Network Units Smart Contract Audit

by Hosho, October 2017

Table of Contents

Technical Summary

This document outlines the overall security of Network Units' smart contract as evaluated by Hosho's Smart Contract auditing team.

The scope of this audit was to analyze and document Network Units' token contract codebase for quality, security, and correctness.

The quality of the contracts is high and has been found free of security issues.

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 Network Units 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 25, 2017. The following contract files and their respective SHA256 fingerprints were evaluated:

File	Fingerprint (SHA256)
infrastructure/ITokenRetreiver.sol	3b4b01c16a6a9c0278436b9af7a8d7f2a1b7182b2a396c67a69382b9477294d6
infrastructure/modifier/InputValidator.sol	cc06d78630510d66cefa6a537e8cf27e1756f66058876a201417ab983daf1ec7
infrastructure/modifier/Owned.sol	2605959f2e37cb6323854406897b981ee6211eb2463e9592692770d9d572c4b6
infrastructure/ownership/IOwnership.sol	baeb0fe923b7e8c0e53d23339e1b06d49c9581a854e342eef710d0d7cdea5892
infrastructure/ownership/ITransferableOwnership.sol	b77b210598b25fdd7d6f309b7cc3bfc2f2aa6f68aec600144209795a47685114
infrastructure/ownership/Ownership.sol	8232872d78c68a10e5867c69f02a294a8ffd5e9871d6da44c86fd348f1e247cc
infrastructure/ownership/TransferableOwnership.sol	f0e062b0a74550d4e0d0e0a7f711ab6e4dd8dbb7e221e0a5bb70421a643c9a7c
integration/wings/IWingsAdapter.sol	435c894ef9cbb7f5a1a9f9450e21b7787a3deb12461a2bf79382e107607db695
source/NUCrowdsale.sol	7209a6ea823a5e70d23b595324a153634c331c126e48fe6973e044d601dad700
source/NUToken.sol	5029cdc66eb3a421869e675a09c303b39b0ceb9c4afd3e134095398fcd687328

source/crowdsale/Crowdsale.sol	`2c1742111fe78a98fca858c90c94a229c3bca71433f8a6e8ce753fe26a6b477b`
source/crowdsale/ICrowdsale.sol	`8643fc3939a55007985f25e0563b4aa72ce6416fc90024f8274fc27ac33a367f`
source/token/IManagedToken.sol	`2d31a7c1268565a8d10f5b2cc09794f6306907d5662263963adc8d162342e816`
source/token/IToken.sol	`8da559d9c2e31d26677e885058e1d884faf302e79b5e60094735672cdc6a7d65`
source/token/ManagedToken.sol	`52a5700a793f7c84ba64eb42fd629f9db653a0a14a9de8d792cdc3e54edc6c31`
source/token/Token.sol	`13ccd606bb0fba0c65db2d3303717a90412d6a3d12b3273bc164b16ddb5f6c50`

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

The Hosho team is pleased with the technical aspects of this contract and believe that this token and presale contract are well written. The whitepaper description matches up with what is expected based on the provided init scripts and the results of testing within Hosho's environment. Limits are properly obeyed and the system issues the correct number of tokens based on the staging testing.

Coverage Report

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

Some lines are not under coverage due to the amount of mutual exclusion that comes from the way the contracts have been constructed. Many of the functions require time-shifting in order to achieve their full functionality and due to the limitations of the testing VM, it's not possible to combine all tests into one unified series. However, uncovered portions in this report, specifically, in regards to the `refund` function, which is the largest portion of the uncovered code, has been tested manually, as well in an alternate series of tests, and it is working as expected.

A number of the branches that are not covered are either impractical to hit as the system would need to be able to generate tokens in excess of the uint256 storage values, which is not possible with the published limits, or must match times exactly.

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

File	% Statements	% Branches	% Functions	% Lines
contracts/infrastructure/ITokenRetreiver.sol	100.00%	100.00%	100.00%	100.00%
contracts/infrastructure/authentication/whitelist/IWhitelist.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/modifier/InputValidator.sol	100.00%	50.00%	100.00%	100.00%
contracts/infrastructure/modifier/Owned.sol	100.00%	50.00%	100.00%	100.00%
contracts/infrastructure/ownership/IOwnership.sol	100.00%	100.00%	100.00%	100.00%
contracts/infrastructure/ownership/ITransferableOwnership.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/integration/wings/IWingsAdapter.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/GLACrowdsale.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/GLAToken.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/crowdsale/Crowdsale.sol	92.40%	75.00%	96.77%	92.05%
contracts/source/crowdsale/ICrowdsale.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/token/IManagedToken.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/token/IToken.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/token/ManagedToken.sol	100.00%	75.00%	100.00%	100.00%
contracts/source/token/Token.sol	100.00%	80.00%	100.00%	100.00%
All files	94.35%	75.45%	98.39%	94.17%

Test Results

Contract: Network Units Crowdsale

- ✓ Amount of wei raised is correct;
- ✓ Should have a start that is before the end (98ms);
- ✓ Should have the presale timing set correctly;
- ✓ Should be not in presale phase before the `beneficiary` is set (48ms);
- ✓ Should not allow the crowdsale to be sent funds in a non `InProgress` stage (44ms);
- ✓ Should allocate tokens, and update amount raised (541ms);
- ✓ Should be in presale phase when starting to send funds (45ms);
- ✓ Should transfer external tokens from itself and the token contract to the owner (190ms);
- ✓ Should transfer external tokens from itself with the child being at 0 tokens (150ms);
- ✓ Should not transfer any tokens if both balances are 0. (124ms);
- ✓ Should not allow the crowdsale to be sent funds under the minimum amount required during presale (89ms);
- ✓ Should allow the crowdsale to `init` and set balances (2444ms);
- ✓ Should not allow the crowdsale to be sent funds that would send it over the maximum presale amount during presale. (78ms);
- ✓ Should return correct balances for eth and tokens for the various stakeholders (104ms);
- ✓ Should not issue a refund until the crowdsale is over;
- ✓ Should be able to finalize the crowdsale and rate should become 0 (1284ms);

- ✓ Should not allow additional eth deposits once the crowdsale has ended. (50ms);
- ✓ Should allow presale eth to be withdrawn by the stakeholders immediately after the crowdsale has ended (489ms);
- ✓ Should not have any tokens available for immediate withdrawal by the stakeholders immediately after the crowdsale has ended (299ms);
- ✓ Should not let anyone but the `beneficiary` destroy the contract; and
- ✓ Should not let the `beneficiary` destroy the contract before 180 days have passed (40ms)

Contract: ERC-20 Compliant Token

- ✓ Should deploy with Network Units Token as the name of the token;
- ✓ Should deploy with NU 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 (4194ms);
- ✓ Should transfer tokens from 0x1bbb1269032bfd0b0fe0851235fc798af6bd3c9b to 0x64a91312c9386f6e8031973b36bda59e224e6b26 (133ms);
- ✓ Should not transfer negative token amounts (51ms);
- ✓ Should not transfer more tokens than you have (44ms);
- ✓ Should allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to authorize 0x341106cb00828c87cd3ac0de55eda7255e04933f to transfer 1000 tokens (91ms);
- ✓ Should not allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to authorize 0x341106cb00828c87cd3ac0de55eda7255e04933f to transfer an additional 1000 tokens once authorized, and authorization balance is > 0 (70ms);
- ✓ Should allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to zero out the 0x341106cb00828c87cd3ac0de55eda7255e04933f authorization (89ms);
- ✓ Should allow 0xdaef8d8c30eeb858b8c774a8d7d5e92a552bb0d9 to authorize 0x53353ef6da4bbb18d242b53a17f7a976265878d5 for 1000 token spend, and 0x53353ef6da4bbb18d242b53a17f7a976265878d5 should be able to send these tokens to 0x341106cb00828c87cd3ac0de55eda7255e04933f (234ms);
- ✓ Should not allow 0x53353ef6da4bbb18d242b53a17f7a976265878d5 to transfer negative tokens from 0xdaef8d8c30eeb858b8c774a8d7d5e92a552bb0d9 (43ms); and
- ✓ Should not allow 0x53353ef6da4bbb18d242b53a17f7a976265878d5 to transfer more tokens than permitted from 0xdaef8d8c30eeb858b8c774a8d7d5e92a552bb0d9 (58ms)

Contract: Network Units Ownership

- ✓ Should return if someone is an owner or not; and
- ✓ Should return the contract owner

Contract: Network Units Token

- ✓ Should not allow ETH to be sent to the contract;
- ✓ Should allow the crowdsale to init and set balances (3026ms);
- ✓ Should be locked against transfers to start;
- ✓ Should not permit transfers while locked; and
- ✓ Should unlock when the crowdSale ends (1295ms)

Contract: Network Units Crowdsale Time-Shifting Tests

- ✓ Should allocate tokens, and update amount raised during presale (818ms);
- ✓ Should exit presale once the first time-marker is hit;
- ✓ Should add refundable funds for a payment made after the first stage (334ms);
- ✓ Should not be able to finalize the crowdsale if the amount raised is too low (41ms);
- ✓ Should not allow excessively small payments inside of the ICO. (88ms);
- ✓ Should add refundable funds for a payment made after the first stage, and show as refundable (1008ms);
- ✓ Should be able to finalize the crowdsale and rate should become 0 (1248ms);
- ✓ Should return correct balances for eth and tokens for the various stakeholders (97ms);
- ✓ Should allow presale ETH to be withdrawn by the stakeholders immediately after the crowdsale has ended (448ms);
- ✓ Should not have any more eth to send after being drained (485ms);
- ✓ Should not have any tokens available for immediate withdraw by the stakeholders immediately after the crowdsale has ended (268ms);
- ✓ Should allow all token releases after 365 days (712ms); and
- ✓ Should let the beneficiary self-destruct the contract after 2 years (343ms)

Closing Statement

We are grateful to have been given the opportunity to work with the Network Units team and Frank Bonnet, the Solidity developer, whom we worked closely with during the audit.

As a small team of experts, having backgrounds in all aspects of blockchain, cryptography, and cybersecurity, we can say with confidence that Network Units' contracts are free of any glaring issues and that Frank has shown himself to be a highly capable Solidity developer.

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

YoSub Kwon