Cyber Cube
Smart contract audit

Date: 08th August 2018
Version: 1.0

http://smartcontract.life
contacts@smartcontract.life

Classification of identified problems

CRITICAL – the possibility of theft of the ether / tokens or their blocking without the possibility of restoring access or other loss of ether / tokens due to any party, for example, dividends.
SERIOUS – the possibility of violations of the contract, in which to restore its correct operation, it is necessary to modify the state of the contract manually or completely replace it.
WARNINGS – the possibility of violation of the planned logic of the contract or the possibility of organizing a DoS attack on the contract.
NOTES – all other remarks.

Methodology of audit

The contract code is manually scanned for known vulnerabilities, logic errors, WhitePaper compliance. If necessary, unit tests are written for questionable moments.

Identified problems

1. [CRITICAL]

- None

2. [SERIOUS]

- None

3. [WARNINGS]

- None

4. [NOTES]

4.1. Classes from the openzeppelin-solidity library are not the latest version

Changes in new versions:
- Minor optimizations of the SafeMath library
- Changed the verification of the user's balance when sending tokens - now if the wrong balance is incorrectly specified, the entire gas of the transaction is not burned.

4.2 Event Transaction when creating tokens.

According to the standard, when creating new tokens, you should call the Transfer method with sender address equal 0x0

https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.mdtransfer-1 :
> A token contract which creates new tokens SHOULD trigger a Transfer event with the _from address set to 0x0 when tokens are created.

4.3. Transfer event when burning tokens

By analogy with the previous number, in the `burn` function can be added the call of the Transfer event with the destination address equal to the address (0).
In the openzeppelin-solidity library it's done like this:
https://github.com/OpenZeppelin/openzeppelin-solidity/blob/master/contracts/token/ERC20/BurnableToken.solL43

4.4 Type of decimals

According to the standard, the type for `decimals` must be uint8:
https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.mddecimals

4.5. Writing Large Numbers

For clarity and for risk-reduction, large numbers can be written like this:
1925000000000000000000000000 = 1925000000 * 10 ** 18