Internal audit results

Automated tool results

We used several automated code review and static analyzer tools to review our smart contracts.

The following tools were used:

- SmartCheck (https://tool.smartdec.net/)
- Securify (https://securify.ch/)
- Ramix solidity IDE (http://remix.ethereum.org/)
- mythril by ConsenSys (https://github.com/ConsenSys/mythril)
- Solhint (https://github.com/protofire/solhint)

Some warnings or hints suggested by these tools were ignored for various reasons. Here is a list of ignored warnings with explanation:

- SmartCheck
    - https://tool.smartdec.net/knowledge/SOLIDITY_ERC20_TRANSFER_SHOULD_THROW
        - File: CetSolutionsCoin.sol, Lines: 24-31, Severity: 1
        - File: CetSolutionsCoin.sol, Lines: 33-40, Severity: 1
        - **This is a false report, the token throws when necessary. The tool does not inspect super calls.**
    - https://tool.smartdec.net/knowledge/SOLIDITY_NO_PAYABLE_FALLBACK
        - **None of our contracts needs to receive ether through transfer() or send().**
    - https://tool.smartdec.net/knowledge/SOLIDITY_REENTRANCY_EXTERNAL_CALL
        - **All of the calls reported with this warning are towards our own, trusted contracts, or to the open-zeppelin SafeMath library**
    - https://tool.smartdec.net/knowledge/SOLIDITY_UNCHECKED_MATH
        - **This calculation is static.**
- Securify
    - Transactions May Affect Ether Receiver
      A contract is exposed to this vulnerability if a miner (who executes and validates transactions) can reorder the transactions within a block in a way that affects the receiver of ether.
    - Transactions May Affect Ether Amount
      A contract is exposed to this vulnerability if a miner (who executes and validates transactions) can reorder the transactions within a block in a way that affects the amount of ether transferred to the receiver.
    - **These issues can only occur when multiple token purchase transactions are made in the block where the token sale ends because all of the tokens are purchased.**
      Even in this situation no ETH is lost, only some token purchasers do not get their tokens, but receive a refund.
      This issue also requires the attacker is the one who eventually mines this block.
      Considering the above reasons we do not consider this issue serious enough to further complicate the smart contracts.
- Solhint
    - warning Event and function names must be different no-simple-event-func-name
        - **All of the events referred by this warning are part of the ERC20 interface.**
    - warning Explicitly mark visibility of state state-visibility
        - **The referred code is part of the zeppelin library.**

Internal experts review

- ERC20 approve vulnerability
    - **This issue is taken care of, with overridden approve function to only allow transaction that either set the allowance to or from zero, and overridden increase/decreaseApproval to always revert()**
- Implenting speed bumps
    - **Adding speed bumps or time locks to the ICO would not increase security, because the token trading only starts after the sale has ended.**
- Short address attack
    - **The ICO was exposed to the Short Address Attack through the issueToken method. To prevent this, the order of parameters has been switched, so the address is the last parameter therefore if it is short er than 20 bytes and the transaction data is padded, the address will be padded instead of *shifting* the value parameter.**