CertiK Verification Report For Bodhi

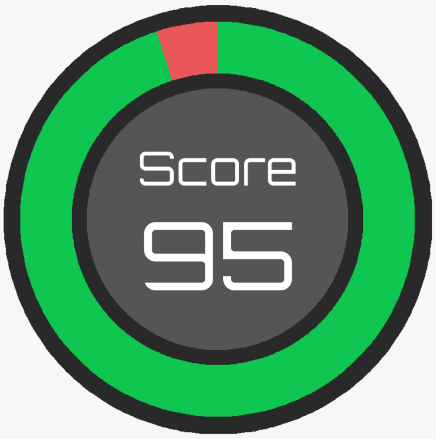
Request Date: 2019-03-01
Revision Date: 2019-03-03

Disclaimer

PASS

CERTIK *believes this smart contract passes security qualifications to be listed on digital asset exchanges.*

Mar 03, 2019



Summary

This audit report summarises the smart contract verification service requested by Bodhi. The goal of this security audit is to guarantee that the audited smart contracts are robust enough to avoid any potential security loopholes.
The result of this report is only a reflection of the source code that was determined in this scope, and of the source code at the time of the audit.

Type of Issues

CertiK smart label engine applied 100% coveraged formal verification labels on the source code, and scanned the code using our proprietary static analysis and formal verification engine to detect the follow type of issues.

Title	Description	Issues	SWC ID
Integer Overflow and Underflow	An overflow/underflow happens when an arithmetic operation reaches the maximum or minimum size of a type.	0	SWC-101
Function incorrectness	Function implementation does not meet the specification, leading to intentional or unintentional vulnerabilities.	0	
Buffer Overflow	An attacker is able to write to arbitrary storage locations of a contract if array of out bound happens	0	SWC-124
Reentrancy	A malicious contract can call back into the calling contract before the first invocation of the function is finished.	0	SWC-107
Transaction Order Dependence	A race condition vulnerability occurs when code depends on the order of the transactions submitted to it.	0	SWC-114
Timestamp Dependence	Timestamp can be influenced by minors to some degree.	4	SWC-116

Insecure Compiler Version	Using an fixed outdated compiler version or floating pragma can be problematic, if there are publicly disclosed bugs and issues that affect the current compiler version used.	0	SWC-102 SWC-103
Insecure Randomness	Block attributes are insecure to generate random numbers, as they can be influenced by minors to some degree.	0	SWC-120
"tx.origin" for authorization	tx.origin should not be used for authorization. Use msg.sender instead.	0	SWC-115
Delegatecall to Untrusted Callee	Calling into untrusted contracts is very dangerous, the target and arguments provided must be sanitized.	0	SWC-112
State Variable Default Visibility	Labeling the visibility explicitly makes it easier to catch incorrect assumptions about who can access the variable.	0	SWC-108
Function Default Visibility	Functions are public by default. A malicious user is able to make unauthorized or unintended state changes if a developer forgot to set the visibility.	0	SWC-100
Uninitialized variables	Uninitialized local storage variables can point to other unexpected storage variables in the contract.	0	SWC-109
Assertion Failure	The assert() function is meant to assert invariants. Properly functioning code should never reach a failing assert statement.	0	SWC-110
Deprecated Solidity Features	Several functions and operators in Solidity are deprecated and should not be used as best practice.	0	SWC-111
Unused variables	Unused variables reduce code quality	0	

Vulnerability Details

Critical

No issue found.

Medium

No issue found.

Low

No issue found.

For every issues found, CertiK categorizes them into 3 buckets based on its risk level:

- Critical: The code implementation does not match the specification, or it could result in loss of funds for contract owner or users.
- Medium: The code implementation does not match the specification at certain condition, or it could affect the security standard by lost of access control.

- Low: The code implementation is not a best practice, or use a suboptimal design pattern, which may lead to security vulnerability, but no concern found yet.

Source Code with CertiK Labels

File BaseContract.sol

```
pragma solidity ^0.4.18;


contract BaseContract {
    struct ResultBalance {
        uint256 totalBets;
        uint256 totalVotes;
        mapping(address => uint256) bets;
        mapping(address => uint256) votes;
    }

    uint8 public constant INVALID_RESULT_INDEX = 255;

    uint8 public numOfResults;
    uint8 public resultIndex = INVALID_RESULT_INDEX;
    uint16 public version;
    ResultBalance[11] internal balances;

    // Modifiers
    modifier validResultIndex(uint8 _resultIndex) {
        require (_resultIndex <= numOfResults - 1);
        _;
    }

    /*
    * @notice Gets the bet balances of the sender for all the results.
    * @return An array of all the bet balances of the sender.
    */
    /*@CTK get_bet_balances
      @tag spec
      @post forall j: uint. (j >= 0 /\ j < numOfResults) -> __return[j] == balances[j
          ].bets[msg.sender]
      */
    function getBetBalances()
        public
        view
        returns (uint256[11])
    {
        uint256[11] memory betBalances;
        /*@CTK set_bet_balances
          @var uint8 i
          @var BaseContract this
          @var uint256[11] betBalances
          @inv forall j: uint. (j >= 0 /\ j < i) -> betBalances[j] == this.balances[j].
              bets[msg.sender]
          @inv i <= this.numOfResults
          @inv this == this__pre
          @post i >= numOfResults
          @post !__should_return
         */
        for (uint8 i = 0; i < numOfResults; i++) {
            betBalances[i] = balances[i].bets[msg.sender];
        }
        return betBalances;
    }

    /*
    * @notice Gets total bets for all the results.
    * @return An array of total bets for all results.
    */
    /*@CTK get_total_bets
      @tag spec
      @post forall j: uint. (j >= 0 /\ j < numOfResults) -> __return[j] == balances[j
          ].totalBets
      */
    function getTotalBets()
        public
        view
        returns (uint256[11])
    {
        uint256[11] memory totalBets;
        /*@CTK set_total_bets
          @var uint8 i
          @var BaseContract this
          @var uint256[11] totalBets
          @inv forall j: uint. (j >= 0 /\ j < i) -> totalBets[j] == this.balances[j].
              totalBets
          @inv i <= this.numOfResults
          @inv this == this__pre
          @post i >= numOfResults
          @post !__should_return
         */
        for (uint8 i = 0; i < numOfResults; i++) {
            totalBets[i] = balances[i].totalBets;
        }
        return totalBets;
    }

    /*
    * @notice Gets the vote balances of the sender for all the results.
    * @return An array of all the vote balances of the sender.
    */
    /*@CTK get_vote_balances
      @tag spec
      @post forall j: uint. (j >= 0 /\ j < numOfResults) -> __return[j] == balances[j
          ].votes[msg.sender]
      */
    function getVoteBalances()
        public
        view
        returns (uint256[11])
    {
        uint256[11] memory voteBalances;
        /*@CTK set_vote_balances
          @var uint8 i
          @var BaseContract this
          @var uint256[11] voteBalances
          @inv forall j: uint. (j >= 0 /\ j < i) -> voteBalances[j] == this.balances[j
              ].votes[msg.sender]
          @inv i <= this.numOfResults
          @inv this == this__pre
          @post i >= numOfResults
          @post !__should_return
         */
        for (uint8 i = 0; i < numOfResults; i++) {
            voteBalances[i] = balances[i].votes[msg.sender];
        }
        return voteBalances;
    }

    /*
    * @notice Gets total votes for all the results.
    * @return An array of total votes for all results.
    */
    /*@CTK get_total_votes
      @tag spec
      @post forall j: uint. (j >= 0 /\ j < numOfResults) -> __return[j] == balances[j
          ].totalVotes
      */
    function getTotalVotes()
        public
        view
        returns (uint256[11])
    {
        uint256[11] memory totalVotes;
        /*@CTK set_total_votes
          @var uint8 i
          @var BaseContract this
          @var uint256[11] totalVotes
          @inv forall j: uint. (j >= 0 /\ j < i) -> totalVotes[j] == this.balances[j].
              totalVotes
          @inv i <= this.numOfResults
          @inv this == this__pre
          @post i >= numOfResults
          @post !__should_return
         */
        for (uint8 i = 0; i < numOfResults; i++) {
            totalVotes[i] = balances[i].totalVotes;
        }
        return totalVotes;
    }
}
```

File Migrations.sol

```
pragma solidity ^0.4.15;

contract Migrations {
  address public owner;
  uint public last_completed_migration;

  modifier restricted() {
    if (msg.sender == owner) _;
  }

  /*@CTK init_migrations
    @post __post.owner == msg.sender
  */
  function Migrations() public {
    owner = msg.sender;
  }

  /*@CTK set_complete
    @pre msg.sender == owner
    @post __post.last_completed_migration == completed
  */
  function setCompleted(uint completed) public restricted {
    last_completed_migration = completed;
  }

  //@CTK NO_ASF
  function upgrade(address new_address) public restricted {
    Migrations upgraded = Migrations(new_address);
    upgraded.setCompleted(last_completed_migration);
  }
}
```

File mocks/StandardTokenMock.sol

```
pragma solidity ^0.4.18;

import '../tokens/StandardToken.sol';

contract StandardTokenMock is StandardToken {
  /*@CTK init_mock_standard_token
    @post __post.balances[_initialAccount] == _initialBalance
    @post __post.totalSupply == _initialBalance
   */
  function StandardTokenMock(address _initialAccount,
          uint256 _initialBalance) public {
    balances[_initialAccount] = _initialBalance;
    totalSupply = _initialBalance;
  }
}
```

File mocks/BasicTokenMock.sol

```
pragma solidity ^0.4.18;

import '../tokens/BasicToken.sol';

contract BasicTokenMock is BasicToken {
  /*@CTK init_mock_basic_token
    @post __post.balances[_initialAccount] == _initialBalance
    @post __post.totalSupply == _initialBalance
  */
  function BasicTokenMock(address _initialAccount, uint256 _initialBalance) public {
    balances[_initialAccount] = _initialBalance;
    totalSupply = _initialBalance;
  }
}
```

File oracles/CentralizedOracle.sol

```
pragma solidity ^0.4.18;

import "./Oracle.sol";

contract CentralizedOracle is Oracle {
    address public oracle;
    uint256 public bettingStartTime;
    uint256 public bettingEndTime;
    uint256 public resultSettingStartTime;
    uint256 public resultSettingEndTime;

    /*
    * @notice Creates new CentralizedOracle contract.
    * @param _version The contract version.
    * @param _owner The address of the owner.
    * @param _eventAddress The address of the Event.
    * @param _numOfResults The number of result options.
    * @param _oracle The address of the CentralizedOracle that will ultimately decide
        the result.
    * @param _bettingStartTime The unix time when betting will start.
    * @param _bettingEndTime The unix time when betting will end.
    * @param _resultSettingStartTime The unix time when the CentralizedOracle can set
        the result.
    * @param _resultSettingEndTime The unix time when anyone can set the result.
    * @param _consensusThreshold The BOT amount that needs to be paid by the Oracle
        for their result to be valid.
    */
    /*@CTK "CentralizedOracle constructor"
      @tag assume_completion
      @post __post.eventAddress == _eventAddress
      @post __post.numOfResults == _numOfResults
      @post __post.oracle == _oracle
      @post __post.bettingStartTime == _bettingStartTime
      @post __post.bettingEndTime == _bettingEndTime
      @post __post.resultSettingStartTime == _resultSettingStartTime
      @post __post.resultSettingEndTime == _resultSettingEndTime
      @post __post.consensusThreshold == _consensusThreshold
    */
    /*@CTK "contruct failed with invalid input"
      @pre _numOfResults == 0 \/ _consensusThreshold == 0
      @post __reverted == true
    */
    function CentralizedOracle(
        uint16 _version,
        address _owner,
        address _eventAddress,
        uint8 _numOfResults,
        address _oracle,
        uint256 _bettingStartTime,
        uint256 _bettingEndTime,
        uint256 _resultSettingStartTime,
        uint256 _resultSettingEndTime,
        uint256 _consensusThreshold)
        Ownable(_owner)
        public
        validAddress(_oracle)
        validAddress(_eventAddress)
    {
        require(_numOfResults > 0);
        require(_bettingEndTime > _bettingStartTime);
        require(_resultSettingStartTime >= _bettingEndTime);
        require(_resultSettingEndTime > _resultSettingStartTime);
        require(_consensusThreshold > 0);

        version = _version;
        eventAddress = _eventAddress;
        numOfResults = _numOfResults;
        oracle = _oracle;
        bettingStartTime = _bettingStartTime;
        bettingEndTime = _bettingEndTime;
        resultSettingStartTime = _resultSettingStartTime;
        resultSettingEndTime = _resultSettingEndTime;
        consensusThreshold = _consensusThreshold;
    }

    /// @notice Fallback function that rejects any amount sent to the contract.
    function() external payable {
        revert();
    }

    /*
    * @notice Allows betting on a result using the blockchain token.
    * @param _resultIndex The index of result to bet on.
    */
    /*CTK "Bodhi bet against certain index"
      @tag assume_completion
      @pre numOfResults > 0
      @post __post.balances[_resultIndex].totalBets == balances[_resultIndex].
          totalBets + msg.value
      @post __post.balances[_resultIndex].bets[msg.sender] == balances[_resultIndex].
          bets[msg.sender] + msg.value
      @post __has_overflow == false
    */
    function bet(uint8 _resultIndex)
        external
        payable
        validResultIndex(_resultIndex)
        isNotFinished()
    {
        require(block.timestamp >= bettingStartTime);
        require(block.timestamp < bettingEndTime);
        require(msg.value > 0);

        balances[_resultIndex].totalBets = balances[_resultIndex].totalBets.add(msg.
            value);
        balances[_resultIndex].bets[msg.sender] = balances[_resultIndex].bets[msg.
            sender].add(msg.value);

        ITopicEvent(eventAddress).betFromOracle.value(msg.value)(msg.sender,
            _resultIndex);

        OracleResultVoted(version, address(this), msg.sender, _resultIndex, msg.value);
    }

    /*
    * @notice CentralizedOracle should call this to set the result. Requires the
        Oracle to approve() BOT in the amount
    *   of the consensus threshold.
    * @param _resultIndex The index of the result to set.
    */
    /*CTK"Set index result"
      @tag assume_completion
      @pre numOfResults > 0
      @post __post.balances[_resultIndex].totalVotes == balances[_resultIndex].
```

```
          totalVotes + this.consensusThreshold
      @post __post.balances[_resultIndex].votes[msg.sender] == balances[_resultIndex].
          votes[msg.sender] + this.consensusThreshold
      @post __has_overflow == false
    */
    function setResult(uint8 _resultIndex)
        external
        validResultIndex(_resultIndex)
        isNotFinished()
    {
        require(block.timestamp >= resultSettingStartTime);
        if (block.timestamp < resultSettingEndTime) {
            require(msg.sender == oracle);
        }

        finished = true;
        resultIndex = _resultIndex;

        balances[_resultIndex].totalVotes = balances[_resultIndex].totalVotes.add(
            consensusThreshold);
        balances[_resultIndex].votes[msg.sender] = balances[_resultIndex].votes[msg.
            sender].add(consensusThreshold);

        ITopicEvent(eventAddress).centralizedOracleSetResult(msg.sender, _resultIndex,
            consensusThreshold);
        OracleResultSet(version, address(this), _resultIndex);
    }
}
```

File oracles/OracleFactory.sol

```
pragma solidity ^0.4.18;

import "./IOracleFactory.sol";
import "./CentralizedOracle.sol";
import "./DecentralizedOracle.sol";
import "../storage/IAddressManager.sol";

contract OracleFactory is IOracleFactory {
    uint16 public version;
    address private addressManager;
    mapping(bytes32 => address) public oracles;

    // Events
    event CentralizedOracleCreated(
        uint16 indexed _version,
        address indexed _contractAddress,
        address indexed _eventAddress,
        uint8 _numOfResults,
        address _oracle,
        uint256 _bettingStartTime,
        uint256 _bettingEndTime,
        uint256 _resultSettingStartTime,
        uint256 _resultSettingEndTime,
        uint256 _consensusThreshold);
    event DecentralizedOracleCreated(
        uint16 indexed _version,
        address indexed _contractAddress,
        address indexed _eventAddress,
        uint8 _numOfResults,
        uint8 _lastResultIndex,
        uint256 _arbitrationEndTime,
        uint256 _consensusThreshold);

    /*
    * @notice Creates new OracleFactory contract.
    * @param _addressManager The address of the AddressManager contract.
    */
    /*@CTK "OracleFactory constructor"
      @post __reverted == false -> (__post.addressManager == _addressManager)
    */
    function OracleFactory(address _addressManager) public {
        require(_addressManager != address(0));

        addressManager = _addressManager;
        // version = IAddressManager(addressManager).currentOracleFactoryIndex();
    }

    function createCentralizedOracle(
        address _eventAddress,
        uint8 _numOfResults,
        address _oracle,
        uint256 _bettingStartTime,
        uint256 _bettingEndTime,
        uint256 _resultSettingStartTime,
        uint256 _resultSettingEndTime,
        uint256 _consensusThreshold)
        public
        returns (address)
    {
        bytes32 hash = getCentralizedOracleHash(_eventAddress, _numOfResults, _oracle,
            _bettingStartTime,
            _bettingEndTime, _resultSettingStartTime, _resultSettingEndTime,
                _consensusThreshold);
        // CentralizedOracle should not exist yet
        require(oracles[hash] == address(0));


        CentralizedOracle cOracle = new CentralizedOracle(version, msg.sender,
            _eventAddress, _numOfResults, _oracle,
            _bettingStartTime, _bettingEndTime, _resultSettingStartTime,
                _resultSettingEndTime, _consensusThreshold);
        oracles[hash] = address(cOracle);

        CentralizedOracleCreated(version, address(cOracle), _eventAddress,
            _numOfResults, _oracle, _bettingStartTime,
            _bettingEndTime, _resultSettingStartTime, _resultSettingEndTime,
                _consensusThreshold);

        return address(cOracle);

    }

    function createDecentralizedOracle(
        address _eventAddress,
        uint8 _numOfResults,
        uint8 _lastResultIndex,
        uint256 _arbitrationEndTime,
        uint256 _consensusThreshold)
        public
        returns (address)
    {
        bytes32 hash = getDecentralizedOracleHash(_eventAddress, _numOfResults,
            _lastResultIndex, _arbitrationEndTime,
            _consensusThreshold);
        // DecentralizedOracle should not exist yet
        require(oracles[hash] == address(0));

        DecentralizedOracle dOracle = new DecentralizedOracle(version, msg.sender,
            _eventAddress, _numOfResults,
            _lastResultIndex, _arbitrationEndTime, _consensusThreshold);
        oracles[hash] = address(dOracle);

        DecentralizedOracleCreated(version, address(dOracle), _eventAddress,
            _numOfResults, _lastResultIndex,
            _arbitrationEndTime, _consensusThreshold);

        return address(dOracle);
    }

    function getCentralizedOracleHash(
        address _eventAddress,
        uint8 _numOfResults,
        address _oracle,
        uint256 _bettingStartTime,
        uint256 _bettingEndTime,
        uint256 _resultSettingStartTime,
        uint256 _resultSettingEndTime,
        uint256 _consensusThreshold)
        private
        pure
        returns (bytes32)
    {

        return keccak256(_eventAddress, _numOfResults, _oracle, _bettingStartTime,
            _bettingEndTime,
            _resultSettingStartTime, _resultSettingEndTime, _consensusThreshold);
    }

    function getDecentralizedOracleHash(
        address _eventAddress,
        uint8 _numOfResults,
        uint8 _lastResultIndex,
        uint256 _arbitrationEndTime,
        uint256 _consensusThreshold)
        private
        pure
        returns (bytes32)
    {
        return keccak256(_eventAddress, _numOfResults, _lastResultIndex,
            _arbitrationEndTime, _consensusThreshold);
    }
}
```

File oracles/DecentralizedOracle.sol

```
pragma solidity ^0.4.18;

import "./Oracle.sol";

contract DecentralizedOracle is Oracle {
    uint8 public lastResultIndex;
    uint256 public arbitrationEndTime;

    /*
    * @notice Creates new DecentralizedOracle contract.
    * @param _version The contract version.
    * @param _owner The address of the owner.
    * @param _eventAddress The address of the Event.
    * @param _numOfResults The number of result options.
    * @param _lastResultIndex The last result index set by the DecentralizedOracle.
    * @param _arbitrationEndTime The unix time when the voting period ends.
    * @param _consensusThreshold The BOT amount that needs to be reached for this
        DecentralizedOracle to be valid.
    */
    /*@CTK "DecentralizedOracle constructor"
      @tag assume_completion
      @pre _numOfResults > 0
      @pre _consensusThreshold > 0
      @post __post.version == _version
      @post __post.eventAddress == _eventAddress
      @post __post.numOfResults == _numOfResults
      @post __post.lastResultIndex == _lastResultIndex
      @post __post.arbitrationEndTime == _arbitrationEndTime
      @post __post.consensusThreshold == _consensusThreshold
    */
    /*CTK "DecentralizedOracle construct fail with invalid input"
      @pre _numOfResults == 0 \/ _consensusThreshold == 0
      @post __reverted == true
    */
    function DecentralizedOracle(
        uint16 _version,
        address _owner,
        address _eventAddress,
        uint8 _numOfResults,
        uint8 _lastResultIndex,
        uint256 _arbitrationEndTime,
        uint256 _consensusThreshold)
        Ownable(_owner)
        public
        validAddress(_eventAddress)
    {
        require(_numOfResults > 0);
        require(_arbitrationEndTime > block.timestamp);
        require(_consensusThreshold > 0);

        version = _version;
        eventAddress = _eventAddress;
        numOfResults = _numOfResults;
        lastResultIndex = _lastResultIndex;
        arbitrationEndTime = _arbitrationEndTime;
        consensusThreshold = _consensusThreshold;
    }

    /*
    * @notice Vote on an Event result which requires BOT payment.
    * @param _eventResultIndex The Event result which is being voted on.
    * @param _botAmount The amount of BOT used to vote.
    */
    function voteResult(uint8 _eventResultIndex, uint256 _botAmount)
        external
        validResultIndex(_eventResultIndex)
        isNotFinished()
    {
        require(_botAmount > 0);
        require(block.timestamp < arbitrationEndTime);
        require(_eventResultIndex != lastResultIndex);

        // Only accept the vote amount up to the consensus threshold
        uint256 adjustedVoteAmount = _botAmount;
        if (balances[_eventResultIndex].totalVotes.add(_botAmount) > consensusThreshold
            ) {
            adjustedVoteAmount = consensusThreshold.sub(balances[_eventResultIndex].
                totalVotes);
        }

        balances[_eventResultIndex].totalVotes = balances[_eventResultIndex].totalVotes
            .add(adjustedVoteAmount);
        balances[_eventResultIndex].votes[msg.sender] = balances[_eventResultIndex].
            votes[msg.sender]
            .add(adjustedVoteAmount);

        ITopicEvent(eventAddress).voteFromOracle(_eventResultIndex, msg.sender,
            adjustedVoteAmount);
        OracleResultVoted(version, address(this), msg.sender, _eventResultIndex,
            adjustedVoteAmount);

        if (balances[_eventResultIndex].totalVotes >= consensusThreshold) {
            setResult();
        }
    }

    /*
    * @notice This can be called by anyone if this VotingOracle did not meet the
        consensus threshold and has reached
    *   the arbitration end time. This finishes the Event and allows winners to
        withdraw their winnings from the Event
    *   contract.
    * @return Flag to indicate success of finalizing the result.
    */
    function finalizeResult()
        external
        isNotFinished()
    {
        require(block.timestamp >= arbitrationEndTime);

        finished = true;
        resultIndex = lastResultIndex;

        ITopicEvent(eventAddress).decentralizedOracleFinalizeResult();
    }

    /*
    * @dev DecentralizedOracle is validated and set the result of the Event.
    */
    function setResult()
        private
    {
        finished = true;

        uint256 winningVoteBalance = 0;
        for (uint8 i = 0; i < numOfResults; i++) {
            uint256 totalVoteBalance = balances[i].totalVotes;
            if (totalVoteBalance > winningVoteBalance) {
                winningVoteBalance = totalVoteBalance;
                resultIndex = i;
            }
        }

        ITopicEvent(eventAddress).decentralizedOracleSetResult(resultIndex,
            winningVoteBalance);
        OracleResultSet(version, address(this), resultIndex);
    }
}
```

File storage/AddressManager.sol

```
pragma solidity ^0.4.18;

import "./IAddressManager.sol";
import "../libs/Ownable.sol";
import "../tokens/ERC20.sol";

contract AddressManager is IAddressManager, Ownable {
    uint256 public constant botDecimals = 8; // Number of decimals for BOT

    uint16 public currentEventFactoryIndex = 0; // Version of the next upgraded
        EventFactory contract
    uint16 public currentOracleFactoryIndex = 0; // Version of the next upgraded
        OracleFactory contract
    uint256 public eventEscrowAmount = 100 * (10**botDecimals); // Amount of escrow
        deposit needed to create an event
    uint256 public arbitrationLength = 86400; // Number of seconds for arbitration
        period
    uint256 public startingOracleThreshold = 100 * (10**botDecimals); // Consensus
        threshold for CentralizedOracles
    uint256 public thresholdPercentIncrease = 10; // Percentage to increase the
        Consensus Threshold every round
    mapping(address => uint16) public eventFactoryAddressToVersion;
    mapping(address => uint16) public oracleFactoryAddressToVersion;
    mapping(address => bool) private whitelistedContracts;

    // Events
    event BodhiTokenAddressChanged(address indexed _newAddress);
    event EventFactoryAddressAdded(uint16 _index, address indexed _contractAddress);
    event OracleFactoryAddressAdded(uint16 _index, address indexed _contractAddress);
    event EscrowDeposited(address indexed _depositer, uint256 escrowAmount);
    event EscrowWithdrawn(address indexed _eventAddress, address indexed _depositer,
        uint256 escrowAmount);
    event ContractWhitelisted(address indexed _contractAddress);

    // Modifiers
    modifier isWhitelisted(address _contractAddress) {
        require(whitelistedContracts[_contractAddress] == true);
        _;
    }

    function AddressManager() Ownable(msg.sender) public {
    }

    /*
    * @notice Transfer the escrow amount needed to create an Event.
    * @param _creator The address of the creator.
    */
    function transferEscrow(address _creator)
        external
        isWhitelisted(msg.sender)
    {

        ERC20 token = ERC20(bodhiTokenAddress);
        require(token.allowance(_creator, address(this)) >= eventEscrowAmount);

        token.transferFrom(_creator, address(this), eventEscrowAmount);


        EscrowDeposited(_creator, eventEscrowAmount);
    }

    /*
    * @notice Withdraws the escrow for an Event.
    * @param _creator The address of the creator.
    */
    function withdrawEscrow(address _creator, uint256 _escrowAmount)
        external
        isWhitelisted(msg.sender)
    {
        ERC20(bodhiTokenAddress).transfer(_creator, _escrowAmount);

        EscrowWithdrawn(msg.sender, _creator, _escrowAmount);
    }

    /*
    * @dev Adds a whitelisted contract address. Only allowed to be called from
        previously whitelisted addresses.
    * @param _contractAddress The address of the contract to whitelist.
    */
    /*@CTK whitelist_success
        @pre whitelistedContracts[_contractAddress] == true
        @pre _contractAddress != address(0)
        @post __post.whitelistedContracts[_contractAddress] == true
     */
    function addWhitelistContract(address _contractAddress)
        external
        isWhitelisted(msg.sender)
        validAddress(_contractAddress)
    {
        whitelistedContracts[_contractAddress] = true;

        ContractWhitelisted(_contractAddress);
    }

    /// @dev Allows the owner to set the address of the Bodhi Token contract.
    /// @param _tokenAddress The address of the Bodhi Token contract.
    /*@CTK set_bodhi_token_address_success
        @pre _tokenAddress != address(0)
        @pre owner == msg.sender
        @post __post.whitelistedContracts[_tokenAddress] == true
        @post __post.bodhiTokenAddress == _tokenAddress
     */
    function setBodhiTokenAddress(address _tokenAddress)
        public
        onlyOwner()
        validAddress(_tokenAddress)
    {
        bodhiTokenAddress = _tokenAddress;
        whitelistedContracts[_tokenAddress] = true;

        BodhiTokenAddressChanged(bodhiTokenAddress);
        ContractWhitelisted(_tokenAddress);
    }

    /// @dev Allows the owner to set the address of an EventFactory contract.
    /// @param _contractAddress The address of the EventFactory contract.
    /*@CTK set_event_factory_address_success
        @pre owner == msg.sender
        @pre _contractAddress != address(0)
        @post __post.whitelistedContracts[_contractAddress] == true
        @post __post.eventFactoryVersionToAddress[currentEventFactoryIndex] ==
            _contractAddress
        @post __post.eventFactoryAddressToVersion[_contractAddress] ==
            currentEventFactoryIndex
        @post __post.currentEventFactoryIndex == currentEventFactoryIndex + 1
    */
    function setEventFactoryAddress(address _contractAddress)
        public
        onlyOwner()
        validAddress(_contractAddress)
    {
        uint16 index = currentEventFactoryIndex;
        eventFactoryVersionToAddress[index] = _contractAddress;
        eventFactoryAddressToVersion[_contractAddress] = index;
        currentEventFactoryIndex++;

        whitelistedContracts[_contractAddress] = true;

        EventFactoryAddressAdded(index, _contractAddress);
        ContractWhitelisted(_contractAddress);
    }

    /// @dev Allows the owner to set the version of the next EventFactory. In case
        AddressManager ever gets
    ///  upgraded, we need to be able to continue where the last version was.
    /// @param _newIndex The index of where the next EventFactory version should start
        .
    function setCurrentEventFactoryIndex(uint16 _newIndex)
      public
      onlyOwner()
    {
      currentEventFactoryIndex = _newIndex;
    }

    /// @dev Allows the owner to set the address of an OracleFactory contract.
    /// @param _contractAddress The address of the OracleFactory contract.
    /*@CTK set_oracle_factory_address_success
        @pre owner == msg.sender
        @pre _contractAddress != address(0)
        @post __post.whitelistedContracts[_contractAddress] == true
        @post __post.oracleFactoryVersionToAddress[currentOracleFactoryIndex] ==
            _contractAddress
        @post __post.oracleFactoryAddressToVersion[_contractAddress] ==
            currentOracleFactoryIndex
        @post __post.currentOracleFactoryIndex == currentOracleFactoryIndex + 1
    */
    function setOracleFactoryAddress(address _contractAddress)
        public
        onlyOwner()
        validAddress(_contractAddress)
    {
        uint16 index = currentOracleFactoryIndex;
        oracleFactoryVersionToAddress[index] = _contractAddress;
        oracleFactoryAddressToVersion[_contractAddress] = index;
        currentOracleFactoryIndex++;

        whitelistedContracts[_contractAddress] = true;

        OracleFactoryAddressAdded(index, _contractAddress);
        ContractWhitelisted(_contractAddress);
    }

    /// @dev Allows the owner to set the version of the next OracleFactory. In case
        AddressManager ever gets
    ///  upgraded, we need to be able to continue where the last version was.
    /// @param _newIndex The index of where the next OracleFactory version should
        start.
    /*@CTK set_current_oracle_factory_index_success
        @pre owner == msg.sender
        @post __post.currentOracleFactoryIndex == _newIndex
    */
    function setCurrentOracleFactoryIndex(uint16 _newIndex)
      public
      onlyOwner()
    {
      currentOracleFactoryIndex = _newIndex;
    }

    /*
    * @dev Sets the eventEscrowAmount that is needed to create an Event.
    * @param _newEscrowAmount The new escrow amount needed to create an Event.
    */
    /*@CTK set_event_escrow_amount
        @pre owner == msg.sender
        @post __post.eventEscrowAmount == _newEscrowAmount
    */
    function setEventEscrowAmount(uint256 _newEscrowAmount)
        public
        onlyOwner()
    {
        eventEscrowAmount = _newEscrowAmount;
    }

    /*
    * @dev Sets the arbitrationLength that DecentralizedOracles will use.
    * @param _newLength The new length in seconds (unix time) of an arbitration period
        .
    */
    /*@CTK set_arbitration_length
        @tag assume_completion
        @pre owner == msg.sender
        @post __post.arbitrationLength == _newLength
    */
    function setArbitrationLength(uint256 _newLength)
        public
        onlyOwner()
    {
        require(_newLength > 0);

        arbitrationLength = _newLength;
    }

    /*
    * @dev Sets the startingOracleThreshold that CentralizedOracles will use.
    * @param _newThreshold The new consensusThreshold for CentralizedOracles.
    */
    /*@CTK set_starting_oracle_threshold
        @pre owner == msg.sender
        @post __post.startingOracleThreshold == _newThreshold
    */
    function setStartingOracleThreshold(uint256 _newThreshold)
        public
        onlyOwner()
    {
        startingOracleThreshold = _newThreshold;
    }

    /*
    * @dev Sets the thresholdPercentIncrease that DecentralizedOracles will use.
    * @param _newIncrement The new increment amount for DecentralizedOracles.
    */
    /*@CTK set_consensus_threshold_percent_increase
        @pre owner == msg.sender
        @post __post.thresholdPercentIncrease == _newPercentage
    */
    function setConsensusThresholdPercentIncrease(uint256 _newPercentage)
        public
        onlyOwner()
    {
        thresholdPercentIncrease = _newPercentage;
    }

    /// @notice Gets the latest index of a deployed EventFactory contract.
    /// @return The index of the latest deployed EventFactory contract.
    function getLastEventFactoryIndex()
        public
        view
        returns (uint16 lastEventFactoryIndex)
    {
        if (currentEventFactoryIndex == 0) {
            return 0;
        } else {
            return currentEventFactoryIndex - 1;
        }
    }

    /// @notice Gets the latest index of a deployed OracleFactory contract.
    /// @return The index of the latest deployed OracleFactory contract.
    function getLastOracleFactoryIndex()
        public
        view
        returns (uint16 lastOracleFactoryIndex)
    {
        if (currentOracleFactoryIndex == 0) {
            return 0;
        } else {
            return currentOracleFactoryIndex - 1;
        }
    }
}
```

File libs/Ownable.sol

```
pragma solidity ^0.4.15;

/**
 * @title Ownable contract
 * @dev The Ownable contract has an owner address, and provides basic authorization
     control functions.
 */
contract Ownable {
  address public owner;

  // Modifiers
  modifier onlyOwner() {
    require(msg.sender == owner);
    _;
  }

  modifier validAddress(address _address) {
    require(_address != address(0));
    _;
  }

  // Events
  event OwnershipTransferred(address indexed _previousOwner, address indexed _newOwner
      );

  /// @dev The Ownable constructor sets the original ‘owner‘ of the contract to the
      sender account.
  /*@CTK throw_on_invalid_address
    @post _owner == address(0) -> __reverted == true
  */
  /*@CTK owner_set_on_success
    @pre __reverted == false -> __post.owner == _owner
```

```
   */
  function Ownable(address _owner) public validAddress(_owner) {
    owner = _owner;
  }

  /// @dev Allows the current owner to transfer control of the contract to a newOwner.
  /// @param _newOwner The address to transfer ownership to.
  /*@CTK transferOwnership
    @post __reverted == false -> (msg.sender == owner -> __post.owner == _newOwner)
    @post (owner != msg.sender) -> (__reverted == true)
    @post (_newOwner == address(0)) -> (__reverted == true)
  */
  function transferOwnership(address _newOwner) public onlyOwner validAddress(
      _newOwner) {
    OwnershipTransferred(owner, _newOwner);
    owner = _newOwner;
  }
}
```

File libs/SafeMath.sol

```
pragma solidity ^0.4.11;

library SafeMath {
    /*@CTK SafeMath_add
      @tag spec
      @post __reverted == __has_assertion_failure
      @post __has_assertion_failure == __has_overflow
      @post __reverted == false -> __return == x + y
      @post msg == msg__post
      @post ((x + y < x) || (x + y < y)) == __has_assertion_failure
      @post __addr_map == __addr_map__post
    */
    function add(uint256 x, uint256 y) internal pure returns(uint256) {
        uint256 z = x + y;
        assert((z >= x) && (z >= y));
        return z;
    }

    /*@CTK SafeMath_sub
      @tag spec
      @post __reverted == __has_assertion_failure
      @post __has_overflow == true -> __has_assertion_failure == true
      @post __reverted == false -> __return == x - y
      @post msg == msg__post
      @post (x < y) == __has_assertion_failure
      @post __addr_map == __addr_map__post
    */
    function sub(uint256 x, uint256 y) internal pure returns(uint256) {
        assert(x >= y);
        uint256 z = x - y;
        return z;
    }

    /*@CTK SafeMath_mul
      @tag spec
      @post __reverted == __has_assertion_failure
      @post __has_assertion_failure == __has_overflow
      @post __reverted == false -> __return == x * y
      @post msg == msg__post
      @post (x > 0 && (x * y / x != y)) == __has_assertion_failure
      @post __addr_map == __addr_map__post
    */
    function mul(uint256 x, uint256 y) internal pure returns(uint256) {
        uint256 z = x * y;
        assert((x == 0) || (z / x == y));
        return z;
    }

    /*@CTK SafeMath_div
      @tag spec
      @post __reverted == __has_assertion_failure
      @post y == 0 -> __has_assertion_failure == true
      @post __has_overflow == true -> __has_assertion_failure == true
      @post __reverted == false -> __return == x / y
      @post msg == msg__post
      @post (y == 0) == __has_assertion_failure
      @post __addr_map == __addr_map__post
    */
    function div(uint256 x, uint256 y) internal pure returns(uint256) {
        assert(y != 0);
        uint256 z = x / y;
        assert(x == y * z + x % y);
        return z;
    }
}
```

File tokens/BodhiToken.sol

```
pragma solidity ^0.4.17;

import './StandardToken.sol';
import '../libs/Ownable.sol';

contract BodhiToken is StandardToken, Ownable {
  // Token configurations
  string public constant name = "Bodhi Token";
  string public constant symbol = "BOT";
  uint256 public constant decimals = 8;

  uint256 public constant tokenTotalSupply = 100 * (10**6) * (10**decimals); // 100
      million BOT ever created

  // Events
  event Mint(uint256 supply, address indexed to, uint256 amount);

  /// @notice Creates new BodhiToken contract
  function BodhiToken() Ownable(msg.sender) public {
  }

  /// @notice Allows the owner to mint new tokens
  /// @param _to Address to mint the tokens to
  /// @param _amount Amount of tokens that will be minted
  /// @return Boolean to signify successful minting

  /*@CTK mintByOwner_check
    @post msg.sender != owner -> __reverted == true
  */
  /*@CTK mintByOwner
    @tag assume_completion
    @post __post.balances[_to] == balances[_to] + _amount
    @post __post.totalSupply == totalSupply + _amount
  */
  function mintByOwner(address _to, uint256 _amount) public onlyOwner returns (bool) {
    return mint(_to, _amount);
  }

  /// @dev Mint new tokens
  /// @param _to Address to mint the tokens to
  /// @param _amount Amount of tokens that will be minted
  /// @return Boolean to signify successful minting
  /*@CTK mintCheck
    @tag assume_completion
    @post __has_overflow == false
    @post __post.balances[_to] == balances[_to] + _amount
    @post __post.totalSupply == totalSupply + _amount
    @post __return == true
  */
  function mint(address _to, uint256 _amount) internal returns (bool) {
    uint256 checkedSupply = totalSupply.add(_amount);
    require(checkedSupply <= tokenTotalSupply);

    totalSupply += _amount;
    balances[_to] = balances[_to].add(_amount);

    Mint(totalSupply, _to, _amount);

    return true;
  }
}
```

File tokens/StandardToken.sol

```
pragma solidity ^0.4.11;

import './BasicToken.sol';
import './ERC20.sol';

/**
 * @title Standard ERC20 token
 *
 * @dev Implementation of the basic standard token.
 * @dev https://github.com/ethereum/EIPs/issues/20
 * @dev Based on code by FirstBlood: https://github.com/Firstbloodio/token/blob/master
    /smart_contract/FirstBloodToken.sol
 */
contract StandardToken is ERC20, BasicToken {
  mapping (address => mapping (address => uint256)) allowed;

  /**
   * @dev Transfer tokens from one address to another
   * @param _from address The address which you want to send tokens from
   * @param _to address The address which you want to transfer to
   * @param _value uint256 the amount of tokens to be transferred
   */
  /*@CTK transferFrom
    @tag assume_completion
    @pre _from != _to
    @post __return == true
    @post __post.balances[_to] == balances[_to] + _value
    @post __post.balances[_from] == balances[_from] - _value
    @post __has_overflow == false
  */
  function transferFrom(address _from, address _to,
      uint256 _value) public returns (bool) {
    require(_to != address(0));
    var _allowance = allowed[_from][msg.sender];

    // Check is not needed because sub(_allowance, _value) will already throw if this
        condition is not met
    // require (_value <= _allowance);

    balances[_from] = balances[_from].sub(_value);
    balances[_to] = balances[_to].add(_value);
    allowed[_from][msg.sender] = _allowance.sub(_value);
    Transfer(_from, _to, _value);
    return true;
  }

  /**
   * @dev Approve the passed address to spend the specified amount of tokens on behalf
        of msg.sender.
   * @param _spender The address which will spend the funds.
   * @param _value The amount of tokens to be spent.
   */
  /*@CTK approve_success
    @post _value == 0 -> __reverted == false
    @post allowed[msg.sender][_spender] == 0 -> __reverted == false
  */
  /*@CTK approve
    @tag assume_completion
    @post __post.allowed[msg.sender][_spender] == _value
  */
  function approve(address _spender, uint256 _value) public returns (bool) {
    // To change the approve amount you first have to reduce the addresses‘
    //  allowance to zero by calling ‘approve(_spender, 0)‘ if it is not
    //  already 0 to mitigate the race condition described here:
    //  https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729
    require((_value == 0) || (allowed[msg.sender][_spender] == 0));

    allowed[msg.sender][_spender] = _value;
    Approval(msg.sender, _spender, _value);
    return true;
  }

  /**
   * @dev Function to check the amount of tokens that an owner allowed to a spender.
   * @param _owner address The address which owns the funds.
   * @param _spender address The address which will spend the funds.
   * @return A uint256 specifying the amount of tokens still available for the spender
        .
   */
  /*@CTK get_allowance
    @post __reverted == false
    @post remaining == allowed[_owner][_spender]
    @post this == __post
  */
  function allowance(address _owner, address _spender) public view returns (uint256
      remaining) {
    return allowed[_owner][_spender];
  }
}
```

File tokens/BasicToken.sol

```
pragma solidity ^0.4.17;

import './ERC20Basic.sol';
import '../libs/SafeMath.sol';

/**
 * @title Basic token
 * @dev Basic version of StandardToken, with no allowances.
 */
contract BasicToken is ERC20Basic {
  using SafeMath for uint256;

  mapping(address => uint256) balances;

  /**
  * @dev transfer token for a specified address
  * @param _to The address to transfer to.
  * @param _value The amount to be transferred.
  */
  /*@CTK transfer_success
    @pre _to != address(0)
    @pre balances[msg.sender] >= _value
    @pre __reverted == false
    @post __reverted == false
    @post __return == true
  */
  /*@CTK transfer_conditions
    @tag assume_completion
    @pre _to != msg.sender
    @post __post.balances[_to] == balances[_to] + _value
    @post __post.balances[msg.sender] == balances[msg.sender] - _value
  */
  /*@CTK transfer_same_address
    @tag assume_completion
    @tag no_overflow
    @pre _to == msg.sender
    @post this == __post
  */
  function transfer(address _to, uint256 _value) public returns (bool) {
    require(_to != address(0));

    // SafeMath.sub will throw if there is not enough balance.
    balances[msg.sender] = balances[msg.sender].sub(_value);
    balances[_to] = balances[_to].add(_value);
    Transfer(msg.sender, _to, _value);
    return true;
  }

  /**
  * @dev Gets the balance of the specified address.
  * @param _owner The address to query the the balance of.
  * @return An uint256 representing the amount owned by the passed address.
  */

  /*@CTK balanceOf
    @post __reverted == false
    @post balance == balances[_owner]
  */
  function balanceOf(address _owner) public view returns (uint256 balance) {
    return balances[_owner];
  }
}
```

File tokens/CrowdsaleBodhiToken.sol

```
pragma solidity ^0.4.18;

import './BodhiToken.sol';

contract CrowdsaleBodhiToken is BodhiToken {
    uint256 public constant nativeDecimals = 18;

    /// @notice 60 million BOT tokens for sale
    uint256 public constant saleAmount = 60 * (10**6) * (10**decimals);

    // Crowdsale parameters
    uint256 public fundingStartBlock;
    uint256 public fundingEndBlock;
    uint256 public initialExchangeRate;

    // Events
    event TokenPurchase(address indexed purchaser, address indexed beneficiary,
        uint256 value, uint256 amount);

    /// @notice Creates new CrowdsaleBodhiToken contract
    /// @param _fundingStartBlock The starting block of crowdsale
    /// @param _fundingEndBlock The ending block of crowdsale
    /// @param _initialExchangeRate The exchange rate of Ether to BOT
    /// @param _presaleAmount The amount of BOT that will be available for presale
    /*@CTK CrowdsaleBodhiToken
      @pre __reverted == false
      @pre balances[owner] == 0
      @pre totalSupply == 0
      @pre decimals == 8
      @post __post.fundingStartBlock == _fundingStartBlock
      @post __post.fundingEndBlock == _fundingEndBlock
      @post __post.initialExchangeRate == _initialExchangeRate
      @post __post.balances[owner] == __post.totalSupply
      */
    function CrowdsaleBodhiToken(
        uint256 _fundingStartBlock,
        uint256 _fundingEndBlock,
        uint256 _initialExchangeRate,
        uint256 _presaleAmount)
        public
    {
        require(_fundingStartBlock >= block.number);
        require(_fundingEndBlock >= _fundingStartBlock);
        require(_initialExchangeRate > 0);

        // Converted to lowest denomination of BOT
        uint256 presaleAmountTokens = _presaleAmount * (10**decimals);
        require(presaleAmountTokens <= saleAmount);

        assert(nativeDecimals >= decimals);

        fundingStartBlock = _fundingStartBlock;
        fundingEndBlock = _fundingEndBlock;
        initialExchangeRate = _initialExchangeRate;

        // Mint the presale tokens, distribute to a receiver
        // Increase the totalSupply accordingly
        mintByOwner(owner, presaleAmountTokens);
    }

    /// @notice Fallback function to purchase tokens
    function() external payable {
        buyTokens(msg.sender);
    }

    /// @notice Allows buying tokens from different address than msg.sender
    /// @param _beneficiary Address that will contain the purchased tokens
    function buyTokens(address _beneficiary) public payable {
        require(_beneficiary != address(0));
        require(block.number >= fundingStartBlock);
        require(block.number <= fundingEndBlock);
        require(msg.value > 0);

        uint256 tokenAmount = getTokenExchangeAmount(msg.value, initialExchangeRate,
            nativeDecimals, decimals);
        uint256 checkedSupply = totalSupply.add(tokenAmount);

        // Ensure new token increment does not exceed the sale amount
        assert(checkedSupply <= saleAmount);

        mintByPurchaser(_beneficiary, tokenAmount);
        TokenPurchase(msg.sender, _beneficiary, msg.value, tokenAmount);
        owner.transfer(msg.value);
    }

    /// @notice Shows the amount of BOT the user will receive for amount of exchanged
        wei
    /// @param _weiAmount Exchanged wei amount to convert
    /// @param _exchangeRate Number of BOT per exchange token
    /// @param _nativeDecimals Number of decimals of the token being exchange for BOT
    /// @param _decimals Number of decimals of BOT token
    /// @return The amount of BOT that will be received
    function getTokenExchangeAmount(
        uint256 _weiAmount,
        uint256 _exchangeRate,
        uint256 _nativeDecimals,
        uint256 _decimals)
        public
        pure
        returns(uint256)
    {
        require(_weiAmount > 0);

        uint256 differenceFactor = (10**_nativeDecimals) / (10**_decimals);
        return _weiAmount.mul(_exchangeRate).div(differenceFactor);
    }

    /// @dev Function to allow crowdsale participants to mint tokens when purchasing
    /// @param _to Address to mint the tokens to
    /// @param _amount Amount of tokens that will be minted
    /// @return Boolean to signify successful minting
    /*@CTK mintByPurchaser
      @pre __reverted == false
      @post __post.balances[_to] == balances[_to] + _amount
      @post __post.totalSupply == totalSupply + _amount
      */
    function mintByPurchaser(address _to, uint256 _amount) private returns (bool) {
        return mint(_to, _amount);
    }
}
```

How to read

Detail for Request 1

transferFrom to same address

Verification date	20, Oct 2018
Verification timespan	395.38 ms
CERTIK *label location*	Line 30-34 in File howtoread.sol

CERTIK *label*

```
30  /*@CTK FAIL "transferFrom to same address"
31      @tag assume_completion
32      @pre from == to
33      @post __post.allowed[from][msg.sender] ==
34  */
```

Raw code location	Line 35-41 in File howtoread.sol

Raw code

```
35  function transferFrom(address from, address tc
        ) {
36      balances[from] = balances[from].sub(tokens
37      allowed[from][msg.sender] = allowed[from][
38      balances[to] = balances[to].add(tokens);
39      emit Transfer(from, to, tokens);
40      return true;
41  }
```

Counterexample	This code violates the specification

Initial environment

```
1   Counter Example:
2   Before Execution:
3       Input = {
4           from = 0x0
5           to = 0x0
6           tokens = 0x6c
7       }
8       This = 0

53              balance: 0x0
54          }
55      }
56
```

Post environment

```
57  After Execution:
58      Input = {
59          from = 0x0
60          to = 0x0
61          tokens = 0x6c
```

Static Analysis Request

INSECURE_COMPILER_VERSION

Line 1 in File BaseContract.sol

```
pragma solidity ^0.4.18;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File Migrations.sol

```
pragma solidity ^0.4.15;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File StandardTokenMock.sol

```
pragma solidity ^0.4.18;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File BasicTokenMock.sol

```
pragma solidity ^0.4.18;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File CentralizedOracle.sol

```
pragma solidity ^0.4.18;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

TIMESTAMP_DEPENDENCY

Line 95 in File CentralizedOracle.sol

```
95        require(block.timestamp >= bettingStartTime);
```

❗ ”block.timestamp” can be influenced by minors to some degree

TIMESTAMP_DEPENDENCY

Line 96 in File CentralizedOracle.sol

```
96        require(block.timestamp < bettingEndTime);
```

❗ ”block.timestamp” can be influenced by minors to some degree

TIMESTAMP_DEPENDENCY

Line 124 in File CentralizedOracle.sol

```
        require(block.timestamp >= resultSettingStartTime);
```

❗ "block.timestamp" can be influenced by minors to some degree

TIMESTAMP_DEPENDENCY

Line 125 in File CentralizedOracle.sol

```
        if (block.timestamp < resultSettingEndTime) {
```

❗ "block.timestamp" can be influenced by minors to some degree

INSECURE_COMPILER_VERSION

Line 1 in File OracleFactory.sol

```
pragma solidity ^0.4.18;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File DecentralizedOracle.sol

```
pragma solidity ^0.4.18;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

TIMESTAMP_DEPENDENCY

Line 47 in File DecentralizedOracle.sol

```
        require(_arbitrationEndTime > block.timestamp);
```

❗ "block.timestamp" can be influenced by minors to some degree

TIMESTAMP_DEPENDENCY

Line 69 in File DecentralizedOracle.sol

```
        require(block.timestamp < arbitrationEndTime);
```

❗ "block.timestamp" can be influenced by minors to some degree

TIMESTAMP_DEPENDENCY

Line 100 in File DecentralizedOracle.sol

```
        require(block.timestamp >= arbitrationEndTime);
```

❗ "block.timestamp" can be influenced by minors to some degree

INSECURE_COMPILER_VERSION

Line 1 in File AddressManager.sol

```
pragma solidity ^0.4.18;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File Ownable.sol

```
pragma solidity ^0.4.15;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File SafeMath.sol

```
pragma solidity ^0.4.11;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File BodhiToken.sol

```
pragma solidity ^0.4.17;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File StandardToken.sol

```
pragma solidity ^0.4.11;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File BasicToken.sol

```
pragma solidity ^0.4.17;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

INSECURE_COMPILER_VERSION

Line 1 in File CrowdsaleBodhiToken.sol

```
pragma solidity ^0.4.18;
```

ℹ Only these compiler versions are safe to compile your code: 0.4.25

Formal Verification Request 1

get_bet_balances

03, Mar 2019
16.69 ms

Line 29-32 in File BaseContract.sol

```
    /*@CTK get_bet_balances
      @tag spec
      @post forall j: uint. (j >= 0 /\ j < numOfResults) -> __return[j] == balances[j
          ].bets[msg.sender]
      */
```

Line 33-53 in File BaseContract.sol

```
    function getBetBalances()
        public
        view
        returns (uint256[11])
    {
        uint256[11] memory betBalances;
        /*@CTK set_bet_balances
          @var uint8 i
          @var BaseContract this
          @var uint256[11] betBalances
          @inv forall j: uint. (j >= 0 /\ j < i) -> betBalances[j] == this.balances[j].
              bets[msg.sender]
          @inv i <= this.numOfResults
          @inv this == this__pre
          @post i >= numOfResults
          @post !__should_return
         */
        for (uint8 i = 0; i < numOfResults; i++) {
            betBalances[i] = balances[i].bets[msg.sender];
        }
        return betBalances;
    }
```

✓ The code meets the specification

Formal Verification Request 2

get_total_bets

03, Mar 2019
16.35 ms

Line 59-62 in File BaseContract.sol

```
    /*@CTK get_total_bets
      @tag spec
      @post forall j: uint. (j >= 0 /\ j < numOfResults) -> __return[j] == balances[j
          ].totalBets
      */
```

Line 63-83 in File BaseContract.sol

```
function getTotalBets()
    public
    view
    returns (uint256[11])
{
    uint256[11] memory totalBets;
    /*@CTK set_total_bets
      @var uint8 i
      @var BaseContract this
      @var uint256[11] totalBets
      @inv forall j: uint. (j >= 0 /\ j < i) -> totalBets[j] == this.balances[j].
          totalBets
      @inv i <= this.numOfResults
      @inv this == this__pre
      @post i >= numOfResults
      @post !__should_return
     */
    for (uint8 i = 0; i < numOfResults; i++) {
        totalBets[i] = balances[i].totalBets;
    }
    return totalBets;
}
```

✓ The code meets the specification

Formal Verification Request 3

get_vote_balances

03, Mar 2019
17.29 ms

Line 89-92 in File BaseContract.sol

```
/*@CTK get_vote_balances
  @tag spec
  @post forall j: uint. (j >= 0 /\ j < numOfResults) -> __return[j] == balances[j
      ].votes[msg.sender]
  */
```

Line 93-113 in File BaseContract.sol

```
function getVoteBalances()
    public
    view
    returns (uint256[11])
{
    uint256[11] memory voteBalances;
    /*@CTK set_vote_balances
      @var uint8 i
      @var BaseContract this
      @var uint256[11] voteBalances
      @inv forall j: uint. (j >= 0 /\ j < i) -> voteBalances[j] == this.balances[j
          ].votes[msg.sender]
      @inv i <= this.numOfResults
```

```
          @inv this == this__pre
          @post i >= numOfResults
          @post !__should_return
         */
        for (uint8 i = 0; i < numOfResults; i++) {
            voteBalances[i] = balances[i].votes[msg.sender];
        }
        return voteBalances;
    }
```

✓ The code meets the specification

Formal Verification Request 4

get_total_votes

📅 03, Mar 2019
⏱ 15.74 ms

Line 119-122 in File BaseContract.sol

```
    /*@CTK get_total_votes
      @tag spec
      @post forall j: uint. (j >= 0 /\ j < numOfResults) -> __return[j] == balances[j
          ].totalVotes
      */
```

Line 123-143 in File BaseContract.sol

```
    function getTotalVotes()
        public
        view
        returns (uint256[11])
    {
        uint256[11] memory totalVotes;
        /*@CTK set_total_votes
          @var uint8 i
          @var BaseContract this
          @var uint256[11] totalVotes
          @inv forall j: uint. (j >= 0 /\ j < i) -> totalVotes[j] == this.balances[j].
              totalVotes
          @inv i <= this.numOfResults
          @inv this == this__pre
          @post i >= numOfResults
          @post !__should_return
         */
        for (uint8 i = 0; i < numOfResults; i++) {
            totalVotes[i] = balances[i].totalVotes;
        }
        return totalVotes;
    }
```

✓ The code meets the specification

Formal Verification Request 5

set_bet_balances__Generated

📅 03, Mar 2019
⏱ 115.68 ms

(Loop) Line 39-48 in File BaseContract.sol

```
No Snippet Available
```

(Loop) Line 39-51 in File BaseContract.sol

```
No Snippet Available
```

✓ The code meets the specification

Formal Verification Request 6

set_total_bets__Generated

📅 03, Mar 2019
⏱ 78.59 ms

(Loop) Line 69-78 in File BaseContract.sol

```
No Snippet Available
```

(Loop) Line 69-81 in File BaseContract.sol

```
No Snippet Available
```

✓ The code meets the specification

Formal Verification Request 7

set_vote_balances__Generated

📅 03, Mar 2019
⏱ 85.26 ms

(Loop) Line 99-108 in File BaseContract.sol

```
No Snippet Available
```

(Loop) Line 99-111 in File BaseContract.sol

```
No Snippet Available
```

✓ The code meets the specification

Formal Verification Request 8

set_total_votes__Generated

📅 03, Mar 2019
⏱ 96.54 ms

(Loop) Line 129-138 in File BaseContract.sol

```
No Snippet Available
```

(Loop) Line 129-141 in File BaseContract.sol

```
No Snippet Available
```

✓ The code meets the specification

Formal Verification Request 9

init_migrations

📅 03, Mar 2019
⏱ 6.55 ms

Line 11-13 in File Migrations.sol

```
/*@CTK init_migrations
  @post __post.owner == msg.sender
*/
```

Line 14-16 in File Migrations.sol

```
function Migrations() public {
  owner = msg.sender;
}
```

✓ The code meets the specification

Formal Verification Request 10

set_complete

📅 03, Mar 2019
⏱ 8.56 ms

Line 18-21 in File Migrations.sol

```
/*@CTK set_complete
  @pre msg.sender == owner
  @post __post.last_completed_migration == completed
*/
```

Line 22-24 in File Migrations.sol

```
function setCompleted(uint completed) public restricted {
  last_completed_migration = completed;
}
```

✓ The code meets the specification

Formal Verification Request 11

Method will not encounter an assertion failure.

📅 03, Mar 2019
⏱ 28.69 ms

Line 26 in File Migrations.sol

```
//@CTK NO_ASF
```

Line 27-30 in File Migrations.sol

```
function upgrade(address new_address) public restricted {
  Migrations upgraded = Migrations(new_address);
  upgraded.setCompleted(last_completed_migration);
}
```

✓ The code meets the specification

Formal Verification Request 12

init_mock_standard_token

📅 03, Mar 2019
⏱ 9.36 ms

Line 6-9 in File StandardTokenMock.sol

```
/*@CTK init_mock_standard_token
  @post __post.balances[_initialAccount] == _initialBalance
  @post __post.totalSupply == _initialBalance
 */
```

Line 10-14 in File StandardTokenMock.sol

```
function StandardTokenMock(address _initialAccount,
        uint256 _initialBalance) public {
  balances[_initialAccount] = _initialBalance;
  totalSupply = _initialBalance;
}
```

✓ The code meets the specification

Formal Verification Request 13

init_mock_basic_token

📅 03, Mar 2019
⏱ 9.77 ms

Line 6-9 in File BasicTokenMock.sol

```
/*@CTK init_mock_basic_token
  @post __post.balances[_initialAccount] == _initialBalance
  @post __post.totalSupply == _initialBalance
*/
```

Line 10-13 in File BasicTokenMock.sol

```
function BasicTokenMock(address _initialAccount, uint256 _initialBalance) public {
  balances[_initialAccount] = _initialBalance;
  totalSupply = _initialBalance;
}
```

✓ The code meets the specification

Formal Verification Request 14

CentralizedOracle constructor

03, Mar 2019
437.33 ms

Line 25-35 in File CentralizedOracle.sol

```
  /*@CTK "CentralizedOracle constructor"
    @tag assume_completion
    @post __post.eventAddress == _eventAddress
    @post __post.numOfResults == _numOfResults
    @post __post.oracle == _oracle
    @post __post.bettingStartTime == _bettingStartTime
    @post __post.bettingEndTime == _bettingEndTime
    @post __post.resultSettingStartTime == _resultSettingStartTime
    @post __post.resultSettingEndTime == _resultSettingEndTime
    @post __post.consensusThreshold == _consensusThreshold
  */
```

Line 40-71 in File CentralizedOracle.sol

```
  function CentralizedOracle(
      uint16 _version,
      address _owner,
      address _eventAddress,
      uint8 _numOfResults,
      address _oracle,
      uint256 _bettingStartTime,
      uint256 _bettingEndTime,
      uint256 _resultSettingStartTime,
      uint256 _resultSettingEndTime,
      uint256 _consensusThreshold)
      Ownable(_owner)
      public
      validAddress(_oracle)
      validAddress(_eventAddress)
  {
      require(_numOfResults > 0);
      require(_bettingEndTime > _bettingStartTime);
      require(_resultSettingStartTime >= _bettingEndTime);
      require(_resultSettingEndTime > _resultSettingStartTime);
      require(_consensusThreshold > 0);
```

```

        version = _version;
        eventAddress = _eventAddress;
        numOfResults = _numOfResults;
        oracle = _oracle;
        bettingStartTime = _bettingStartTime;
        bettingEndTime = _bettingEndTime;
        resultSettingStartTime = _resultSettingStartTime;
        resultSettingEndTime = _resultSettingEndTime;
        consensusThreshold = _consensusThreshold;
    }
```

✓ The code meets the specification

Formal Verification Request 15

contruct failed with invalid input

📅 03, Mar 2019
⏱ 260.29 ms

Line 36-39 in File CentralizedOracle.sol

```
    /*@CTK "contruct failed with invalid input"
      @pre _numOfResults == 0 \/ _consensusThreshold == 0
      @post __reverted == true
    */
```

Line 40-71 in File CentralizedOracle.sol

```
    function CentralizedOracle(
        uint16 _version,
        address _owner,
        address _eventAddress,
        uint8 _numOfResults,
        address _oracle,
        uint256 _bettingStartTime,
        uint256 _bettingEndTime,
        uint256 _resultSettingStartTime,
        uint256 _resultSettingEndTime,
        uint256 _consensusThreshold)
        Ownable(_owner)
        public
        validAddress(_oracle)
        validAddress(_eventAddress)
    {
        require(_numOfResults > 0);
        require(_bettingEndTime > _bettingStartTime);
        require(_resultSettingStartTime >= _bettingEndTime);
        require(_resultSettingEndTime > _resultSettingStartTime);
        require(_consensusThreshold > 0);

        version = _version;
        eventAddress = _eventAddress;
        numOfResults = _numOfResults;
        oracle = _oracle;
        bettingStartTime = _bettingStartTime;
        bettingEndTime = _bettingEndTime;
        resultSettingStartTime = _resultSettingStartTime;
        resultSettingEndTime = _resultSettingEndTime;
        consensusThreshold = _consensusThreshold;
    }
```

✅ The code meets the specification

Formal Verification Request 16

OracleFactory constructor

📅 03, Mar 2019
⏱ 14.97 ms

Line 38-40 in File OracleFactory.sol

```
    /*@CTK "OracleFactory constructor"
      @post __reverted == false -> (__post.addressManager == _addressManager)
    */
```

Line 41-46 in File OracleFactory.sol

```
    function OracleFactory(address _addressManager) public {
        require(_addressManager != address(0));

        addressManager = _addressManager;
        // version = IAddressManager(addressManager).currentOracleFactoryIndex();
    }
```

✅ The code meets the specification

Formal Verification Request 17

DecentralizedOracle constructor

📅 03, Mar 2019
⏱ 220.57 ms

Line 19-29 in File DecentralizedOracle.sol

```
    /*@CTK "DecentralizedOracle constructor"
      @tag assume_completion
      @pre _numOfResults > 0
      @pre _consensusThreshold > 0
      @post __post.version == _version
      @post __post.eventAddress == _eventAddress
      @post __post.numOfResults == _numOfResults
      @post __post.lastResultIndex == _lastResultIndex
      @post __post.arbitrationEndTime == _arbitrationEndTime
      @post __post.consensusThreshold == _consensusThreshold
    */
```

Line 34-56 in File DecentralizedOracle.sol

```
    function DecentralizedOracle(
        uint16 _version,
        address _owner,
        address _eventAddress,
        uint8 _numOfResults,
        uint8 _lastResultIndex,
        uint256 _arbitrationEndTime,
        uint256 _consensusThreshold)
        Ownable(_owner)
        public
        validAddress(_eventAddress)
    {
        require(_numOfResults > 0);
        require(_arbitrationEndTime > block.timestamp);
        require(_consensusThreshold > 0);

        version = _version;
        eventAddress = _eventAddress;
        numOfResults = _numOfResults;
        lastResultIndex = _lastResultIndex;
        arbitrationEndTime = _arbitrationEndTime;
        consensusThreshold = _consensusThreshold;
    }
```

✓ The code meets the specification

Formal Verification Request 18

whitelist_success

03, Mar 2019
34.91 ms

Line 72-76 in File AddressManager.sol

```
    /*@CTK whitelist_success
        @pre whitelistedContracts[_contractAddress] == true
        @pre _contractAddress != address(0)
        @post __post.whitelistedContracts[_contractAddress] == true
     */
```

Line 77-85 in File AddressManager.sol

```
    function addWhitelistContract(address _contractAddress)
        external
        isWhitelisted(msg.sender)
        validAddress(_contractAddress)
    {
        whitelistedContracts[_contractAddress] = true;

        ContractWhitelisted(_contractAddress);
    }
```

✓ The code meets the specification

Formal Verification Request 19

set_bodhi_token_address_success

03, Mar 2019
40.05 ms

Line 89-94 in File AddressManager.sol

```
    /*@CTK set_bodhi_token_address_success
        @pre _tokenAddress != address(0)
        @pre owner == msg.sender
        @post __post.whitelistedContracts[_tokenAddress] == true
        @post __post.bodhiTokenAddress == _tokenAddress
     */
```

Line 95-105 in File AddressManager.sol

```
    function setBodhiTokenAddress(address _tokenAddress)
        public
        onlyOwner()
        validAddress(_tokenAddress)
    {
        bodhiTokenAddress = _tokenAddress;
        whitelistedContracts[_tokenAddress] = true;

        BodhiTokenAddressChanged(bodhiTokenAddress);
        ContractWhitelisted(_tokenAddress);
    }
```

✓ The code meets the specification

Formal Verification Request 20

set_event_factory_address_success

03, Mar 2019
70.89 ms

Line 109-116 in File AddressManager.sol

```
    /*@CTK set_event_factory_address_success
        @pre owner == msg.sender
        @pre _contractAddress != address(0)
        @post __post.whitelistedContracts[_contractAddress] == true
        @post __post.eventFactoryVersionToAddress[currentEventFactoryIndex] ==
            _contractAddress
        @post __post.eventFactoryAddressToVersion[_contractAddress] ==
            currentEventFactoryIndex
        @post __post.currentEventFactoryIndex == currentEventFactoryIndex + 1
    */
```

Line 117-131 in File AddressManager.sol

```
    function setEventFactoryAddress(address _contractAddress)
        public
        onlyOwner()
        validAddress(_contractAddress)
    {
        uint16 index = currentEventFactoryIndex;
        eventFactoryVersionToAddress[index] = _contractAddress;
        eventFactoryAddressToVersion[_contractAddress] = index;
        currentEventFactoryIndex++;

        whitelistedContracts[_contractAddress] = true;

        EventFactoryAddressAdded(index, _contractAddress);
        ContractWhitelisted(_contractAddress);
    }
```

✓ The code meets the specification

Formal Verification Request 21

set_oracle_factory_address_success

📅 03, Mar 2019
⏱ 68.32 ms

Line 145-152 in File AddressManager.sol

```
    /*@CTK set_oracle_factory_address_success
        @pre owner == msg.sender
        @pre _contractAddress != address(0)
        @post __post.whitelistedContracts[_contractAddress] == true
        @post __post.oracleFactoryVersionToAddress[currentOracleFactoryIndex] ==
            _contractAddress
        @post __post.oracleFactoryAddressToVersion[_contractAddress] ==
            currentOracleFactoryIndex
        @post __post.currentOracleFactoryIndex == currentOracleFactoryIndex + 1
    */
```

Line 153-167 in File AddressManager.sol

```
    function setOracleFactoryAddress(address _contractAddress)
        public
        onlyOwner()
        validAddress(_contractAddress)
    {
        uint16 index = currentOracleFactoryIndex;
        oracleFactoryVersionToAddress[index] = _contractAddress;
        oracleFactoryAddressToVersion[_contractAddress] = index;
        currentOracleFactoryIndex++;

        whitelistedContracts[_contractAddress] = true;

        OracleFactoryAddressAdded(index, _contractAddress);
        ContractWhitelisted(_contractAddress);
    }
```

✓ The code meets the specification

Formal Verification Request 22

set_current_oracle_factory_index_success

📅 03, Mar 2019
⏱ 20.68 ms

Line 172-175 in File AddressManager.sol

```
/*@CTK set_current_oracle_factory_index_success
    @pre owner == msg.sender
    @post __post.currentOracleFactoryIndex == _newIndex
*/
```

Line 176-181 in File AddressManager.sol

```
function setCurrentOracleFactoryIndex(uint16 _newIndex)
  public
  onlyOwner()
{
  currentOracleFactoryIndex = _newIndex;
}
```

✓ The code meets the specification

Formal Verification Request 23

set_event_escrow_amount

📅 03, Mar 2019
⏱ 20.52 ms

Line 187-190 in File AddressManager.sol

```
/*@CTK set_event_escrow_amount
    @pre owner == msg.sender
    @post __post.eventEscrowAmount == _newEscrowAmount
*/
```

Line 191-196 in File AddressManager.sol

```
function setEventEscrowAmount(uint256 _newEscrowAmount)
    public
    onlyOwner()
{
    eventEscrowAmount = _newEscrowAmount;
}
```

✓ The code meets the specification

Formal Verification Request 24

set_arbitration_length

📅 03, Mar 2019
⏱ 29.33 ms

Line 202-206 in File AddressManager.sol

```
    /*@CTK set_arbitration_length
        @tag assume_completion
        @pre owner == msg.sender
        @post __post.arbitrationLength == _newLength
    */
```

Line 207-214 in File AddressManager.sol

```
    function setArbitrationLength(uint256 _newLength)
        public
        onlyOwner()
    {
        require(_newLength > 0);

        arbitrationLength = _newLength;
    }
```

✓ The code meets the specification

Formal Verification Request 25

set_starting_oracle_threshold

03, Mar 2019
19.1 ms

Line 220-223 in File AddressManager.sol

```
    /*@CTK set_starting_oracle_threshold
        @pre owner == msg.sender
        @post __post.startingOracleThreshold == _newThreshold
    */
```

Line 224-229 in File AddressManager.sol

```
    function setStartingOracleThreshold(uint256 _newThreshold)
        public
        onlyOwner()
    {
        startingOracleThreshold = _newThreshold;
    }
```

✓ The code meets the specification

Formal Verification Request 26

set_consensus_threshold_percent_increase

03, Mar 2019
19.95 ms

Line 235-238 in File AddressManager.sol

```
    /*@CTK set_consensus_threshold_percent_increase
        @pre owner == msg.sender
        @post __post.thresholdPercentIncrease == _newPercentage
    */
```

Line 239-244 in File AddressManager.sol

```
    function setConsensusThresholdPercentIncrease(uint256 _newPercentage)
        public
        onlyOwner()
    {
        thresholdPercentIncrease = _newPercentage;
    }
```

✓ The code meets the specification

Formal Verification Request 27

throw_on_invalid_address

03, Mar 2019
16.0 ms

Line 25-27 in File Ownable.sol

```
  /*@CTK throw_on_invalid_address
    @post _owner == address(0) -> __reverted == true
  */
```

Line 31-33 in File Ownable.sol

```
  function Ownable(address _owner) public validAddress(_owner) {
    owner = _owner;
  }
```

✓ The code meets the specification

Formal Verification Request 28

owner_set_on_success

03, Mar 2019
0.36 ms

Line 28-30 in File Ownable.sol

```
  /*@CTK owner_set_on_success
    @pre __reverted == false -> __post.owner == _owner
   */
```

Line 31-33 in File Ownable.sol

```
  function Ownable(address _owner) public validAddress(_owner) {
    owner = _owner;
  }
```

✓ The code meets the specification

Formal Verification Request 29

transferOwnership

📅 03, Mar 2019
⏱ 29.2 ms

Line 37-41 in File Ownable.sol

```
/*@CTK transferOwnership
  @post __reverted == false -> (msg.sender == owner -> __post.owner == _newOwner)
  @post (owner != msg.sender) -> (__reverted == true)
  @post (_newOwner == address(0)) -> (__reverted == true)
*/
```

Line 42-45 in File Ownable.sol

```
function transferOwnership(address _newOwner) public onlyOwner validAddress(
    _newOwner) {
  OwnershipTransferred(owner, _newOwner);
  owner = _newOwner;
}
```

✓ The code meets the specification

Formal Verification Request 30

SafeMath_add

📅 03, Mar 2019
⏱ 19.47 ms

Line 4-12 in File SafeMath.sol

```
  /*@CTK SafeMath_add
    @tag spec
    @post __reverted == __has_assertion_failure
    @post __has_assertion_failure == __has_overflow
    @post __reverted == false -> __return == x + y
    @post msg == msg__post
    @post ((x + y < x) || (x + y < y)) == __has_assertion_failure
    @post __addr_map == __addr_map__post
  */
```

Line 13-17 in File SafeMath.sol

```
  function add(uint256 x, uint256 y) internal pure returns(uint256) {
      uint256 z = x + y;
      assert((z >= x) && (z >= y));
      return z;
  }
```

✓ The code meets the specification

Formal Verification Request 31

SafeMath_sub

📅 03, Mar 2019
⏱ 15.21 ms

Line 19-27 in File SafeMath.sol

```
/*@CTK SafeMath_sub
  @tag spec
  @post __reverted == __has_assertion_failure
  @post __has_overflow == true -> __has_assertion_failure == true
  @post __reverted == false -> __return == x - y
  @post msg == msg__post
  @post (x < y) == __has_assertion_failure
  @post __addr_map == __addr_map__post
*/
```

Line 28-32 in File SafeMath.sol

```
function sub(uint256 x, uint256 y) internal pure returns(uint256) {
    assert(x >= y);
    uint256 z = x - y;
    return z;
}
```

✅ The code meets the specification

Formal Verification Request 32

SafeMath_mul

📅 03, Mar 2019
⏱ 114.11 ms

Line 34-42 in File SafeMath.sol

```
/*@CTK SafeMath_mul
  @tag spec
  @post __reverted == __has_assertion_failure
  @post __has_assertion_failure == __has_overflow
  @post __reverted == false -> __return == x * y
  @post msg == msg__post
  @post (x > 0 && (x * y / x != y)) == __has_assertion_failure
  @post __addr_map == __addr_map__post
*/
```

Line 43-47 in File SafeMath.sol

```
function mul(uint256 x, uint256 y) internal pure returns(uint256) {
    uint256 z = x * y;
    assert((x == 0) || (z / x == y));
    return z;
}
```

✅ The code meets the specification

Formal Verification Request 33

SafeMath_div

📅 03, Mar 2019
⏱ 1095.07 ms

Line 49-58 in File SafeMath.sol

```
  /*@CTK SafeMath_div
    @tag spec
    @post __reverted == __has_assertion_failure
    @post y == 0 -> __has_assertion_failure == true
    @post __has_overflow == true -> __has_assertion_failure == true
    @post __reverted == false -> __return == x / y
    @post msg == msg__post
    @post (y == 0) == __has_assertion_failure
    @post __addr_map == __addr_map__post
  */
```

Line 59-64 in File SafeMath.sol

```
  function div(uint256 x, uint256 y) internal pure returns(uint256) {
      assert(y != 0);
      uint256 z = x / y;
      assert(x == y * z + x % y);
      return z;
  }
```

✅ The code meets the specification

Formal Verification Request 34

mintByOwner_check

📅 03, Mar 2019
⏱ 109.23 ms

Line 26-28 in File BodhiToken.sol

```
/*@CTK mintByOwner_check
  @post msg.sender != owner -> __reverted == true
*/
```

Line 34-36 in File BodhiToken.sol

```
function mintByOwner(address _to, uint256 _amount) public onlyOwner returns (bool) {
  return mint(_to, _amount);
}
```

✅ The code meets the specification

Formal Verification Request 35

mintByOwner

📅 03, Mar 2019
⏱ 141.09 ms

Line 29-33 in File BodhiToken.sol

```
/*@CTK mintByOwner
  @tag assume_completion
  @post __post.balances[_to] == balances[_to] + _amount
  @post __post.totalSupply == totalSupply + _amount
*/
```

Line 34-36 in File BodhiToken.sol

```
function mintByOwner(address _to, uint256 _amount) public onlyOwner returns (bool) {
  return mint(_to, _amount);
}
```

✓ The code meets the specification

Formal Verification Request 36

mintCheck

📅 03, Mar 2019
⏱ 93.81 ms

Line 42-48 in File BodhiToken.sol

```
/*@CTK mintCheck
  @tag assume_completion
  @post __has_overflow == false
  @post __post.balances[_to] == balances[_to] + _amount
  @post __post.totalSupply == totalSupply + _amount
  @post __return == true
*/
```

Line 49-59 in File BodhiToken.sol

```
function mint(address _to, uint256 _amount) internal returns (bool) {
  uint256 checkedSupply = totalSupply.add(_amount);
  require(checkedSupply <= tokenTotalSupply);

  totalSupply += _amount;
  balances[_to] = balances[_to].add(_amount);

  Mint(totalSupply, _to, _amount);

  return true;
}
```

✓ The code meets the specification

Formal Verification Request 37

transferFrom

📅 03, Mar 2019
⏱ 87.08 ms

Line 22-29 in File StandardToken.sol

```
/*@CTK transferFrom
  @tag assume_completion
  @pre _from != _to
  @post __return == true
  @post __post.balances[_to] == balances[_to] + _value
  @post __post.balances[_from] == balances[_from] - _value
  @post __has_overflow == false
*/
```

Line 30-43 in File StandardToken.sol

```
function transferFrom(address _from, address _to,
    uint256 _value) public returns (bool) {
  require(_to != address(0));
  var _allowance = allowed[_from][msg.sender];

  // Check is not needed because sub(_allowance, _value) will already throw if this
      condition is not met
  // require (_value <= _allowance);

  balances[_from] = balances[_from].sub(_value);
  balances[_to] = balances[_to].add(_value);
  allowed[_from][msg.sender] = _allowance.sub(_value);
  Transfer(_from, _to, _value);
  return true;
}
```

✅ The code meets the specification

Formal Verification Request 38

approve_success

📅 03, Mar 2019
⏱ 19.41 ms

Line 50-53 in File StandardToken.sol

```
/*@CTK approve_success
  @post _value == 0 -> __reverted == false
  @post allowed[msg.sender][_spender] == 0 -> __reverted == false
*/
```

Line 58-68 in File StandardToken.sol

```
function approve(address _spender, uint256 _value) public returns (bool) {
  // To change the approve amount you first have to reduce the addresses‘
  //  allowance to zero by calling ‘approve(_spender, 0)‘ if it is not
```

```
    // already 0 to mitigate the race condition described here:
    // https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729
    require((_value == 0) || (allowed[msg.sender][_spender] == 0));

    allowed[msg.sender][_spender] = _value;
    Approval(msg.sender, _spender, _value);
    return true;
  }
```

✓ The code meets the specification

Formal Verification Request 39

approve

03, Mar 2019
1.67 ms

Line 54-57 in File StandardToken.sol

```
  /*@CTK approve
    @tag assume_completion
    @post __post.allowed[msg.sender][_spender] == _value
  */
```

Line 58-68 in File StandardToken.sol

```
  function approve(address _spender, uint256 _value) public returns (bool) {
    // To change the approve amount you first have to reduce the addresses`
    // allowance to zero by calling `approve(_spender, 0)` if it is not
    // already 0 to mitigate the race condition described here:
    // https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729
    require((_value == 0) || (allowed[msg.sender][_spender] == 0));

    allowed[msg.sender][_spender] = _value;
    Approval(msg.sender, _spender, _value);
    return true;
  }
```

✓ The code meets the specification

Formal Verification Request 40

get_allowance

03, Mar 2019
5.93 ms

Line 76-80 in File StandardToken.sol

```
  /*@CTK get_allowance
    @post __reverted == false
    @post remaining == allowed[_owner][_spender]
    @post this == __post
  */
```

Line 81-83 in File StandardToken.sol

```
function allowance(address _owner, address _spender) public view returns (uint256
    remaining) {
  return allowed[_owner][_spender];
}
```

✓ The code meets the specification

Formal Verification Request 41

transfer_success

03, Mar 2019
36.65 ms

Line 20-26 in File BasicToken.sol

```
/*@CTK transfer_success
  @pre _to != address(0)
  @pre balances[msg.sender] >= _value
  @pre __reverted == false
  @post __reverted == false
  @post __return == true
*/
```

Line 39-47 in File BasicToken.sol

```
function transfer(address _to, uint256 _value) public returns (bool) {
  require(_to != address(0));

  // SafeMath.sub will throw if there is not enough balance.
  balances[msg.sender] = balances[msg.sender].sub(_value);
  balances[_to] = balances[_to].add(_value);
  Transfer(msg.sender, _to, _value);
  return true;
}
```

✓ The code meets the specification

Formal Verification Request 42

transfer_conditions

03, Mar 2019
34.97 ms

Line 27-32 in File BasicToken.sol

```
/*@CTK transfer_conditions
  @tag assume_completion
  @pre _to != msg.sender
  @post __post.balances[_to] == balances[_to] + _value
  @post __post.balances[msg.sender] == balances[msg.sender] - _value
*/
```

Line 39-47 in File BasicToken.sol

```
function transfer(address _to, uint256 _value) public returns (bool) {
  require(_to != address(0));

  // SafeMath.sub will throw if there is not enough balance.
  balances[msg.sender] = balances[msg.sender].sub(_value);
  balances[_to] = balances[_to].add(_value);
  Transfer(msg.sender, _to, _value);
  return true;
}
```

✓ The code meets the specification

Formal Verification Request 43

transfer_same_address

03, Mar 2019
5.63 ms

Line 33-38 in File BasicToken.sol

```
/*@CTK transfer_same_address
  @tag assume_completion
  @tag no_overflow
  @pre _to == msg.sender
  @post this == __post
*/
```

Line 39-47 in File BasicToken.sol

```
function transfer(address _to, uint256 _value) public returns (bool) {
  require(_to != address(0));

  // SafeMath.sub will throw if there is not enough balance.
  balances[msg.sender] = balances[msg.sender].sub(_value);
  balances[_to] = balances[_to].add(_value);
  Transfer(msg.sender, _to, _value);
  return true;
}
```

✓ The code meets the specification

Formal Verification Request 44

balanceOf

03, Mar 2019
6.19 ms

Line 55-58 in File BasicToken.sol

```
/*@CTK balanceOf
  @post __reverted == false
  @post balance == balances[_owner]
*/
```

Line 59-61 in File BasicToken.sol

```
function balanceOf(address _owner) public view returns (uint256 balance) {
  return balances[_owner];
}
```

✓ The code meets the specification

Formal Verification Request 45

CrowdsaleBodhiToken

03, Mar 2019
384.8 ms

Line 24-33 in File CrowdsaleBodhiToken.sol

```
  /*@CTK CrowdsaleBodhiToken
    @pre __reverted == false
    @pre balances[owner] == 0
    @pre totalSupply == 0
    @pre decimals == 8
    @post __post.fundingStartBlock == _fundingStartBlock
    @post __post.fundingEndBlock == _fundingEndBlock
    @post __post.initialExchangeRate == _initialExchangeRate
    @post __post.balances[owner] == __post.totalSupply
    */
```

Line 34-58 in File CrowdsaleBodhiToken.sol

```
  function CrowdsaleBodhiToken(
      uint256 _fundingStartBlock,
      uint256 _fundingEndBlock,
      uint256 _initialExchangeRate,
      uint256 _presaleAmount)
      public
  {
      require(_fundingStartBlock >= block.number);
      require(_fundingEndBlock >= _fundingStartBlock);
      require(_initialExchangeRate > 0);

      // Converted to lowest denomination of BOT
      uint256 presaleAmountTokens = _presaleAmount * (10**decimals);
      require(presaleAmountTokens <= saleAmount);

      assert(nativeDecimals >= decimals);

      fundingStartBlock = _fundingStartBlock;
      fundingEndBlock = _fundingEndBlock;
      initialExchangeRate = _initialExchangeRate;

      // Mint the presale tokens, distribute to a receiver
      // Increase the totalSupply accordingly
```

```
        mintByOwner(owner, presaleAmountTokens);
    }
```

✓ The code meets the specification

Formal Verification Request 46

mintByPurchaser

03, Mar 2019
196.85 ms

Line 109-113 in File CrowdsaleBodhiToken.sol

```
    /*@CTK mintByPurchaser
      @pre __reverted == false
      @post __post.balances[_to] == balances[_to] + _amount
      @post __post.totalSupply == totalSupply + _amount
      */
```

Line 114-116 in File CrowdsaleBodhiToken.sol

```
    function mintByPurchaser(address _to, uint256 _amount) private returns (bool) {
        return mint(_to, _amount);
    }
```

✓ The code meets the specification