CertiK Audit Report For PlayDapp

Request Date: 2019-02-19
Revision Date: 2019-02-26
Platform Name: Ethereum

CERTIK

Contents

Disclaimer

This Report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Verification Services Agreement between CertiK and PlayDapp(the “Company”), or the scope of services/verification, and terms and conditions provided to the Company in connection with the verification (collectively, the “Agreement”). This Report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This Report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without CertiK’s prior written consent.

About CertiK

CertiK is a technology-led blockchain security company founded by Computer Science professors from Yale University and Columbia University built to prove the security and correctness of smart contracts and blockchain protocols.

CertiK, in partnership with grants from IBM and the Ethereum Foundation, has developed a proprietary Formal Verification technology to apply rigorous and complete mathematical reasoning against code. This process ensures algorithms, protocols, and business functionalities are secured and working as intended across all platforms.

CertiK differs from traditional testing approaches by employing Formal Verification to mathematically prove blockchain ecosystem and smart contracts are hacker-resistant and bug-free. CertiK uses this industry-leading technology together with standardized test suites, static analysis, and expert manual review to create a full-stack solution for our partners across the blockchain world to secure 6.2B in assets.

For more information: https://certik.org/

Executive Summary

This report has been prepared for PlayDapp to discover issues and vulnerabilities in the source code of their PLA and TokensPurchased smart contracts. A comprehensive examination has been performed, utilizing CertiK's Formal Verification Platform, Static Analysis, and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practice and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line by line manual review of the entire codebase by industry experts.

Vulnerability Classification

CertiK categorizes issues into 3 buckets based on overall risk levels:

Critical

The code implementation does not match the specification, or it could result in the loss of funds for contract owner or users.

Medium

The code implementation does not match the specification under certain conditions, or it could affect the security standard by lost of access control.

Low

The code implementation does not follow best practices, or use suboptimal design patterns, which may lead to security vulnerabilies further down the line.

Testing Summary

PASS

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

Sep 18, 2019

Type of Issues

CertiK smart label engine applied 100% formal verification coverage on the source code. Our team of engineers ao scanned the source code using our proprietary static analysis tools and code-review methodologies. The following technical issues were found:

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.	0	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.

Static Analysis Results

INSECURE_COMPILER_VERSION

Line 1 in File PLA.sol

```
pragma solidity ^0.5.0;
```

ⓘ Only these compiler versions are safe to compile your code: 0.5.10

INSECURE_COMPILER_VERSION

Line 1 in File TokenPurchased.sol

```
pragma solidity ^0.5.0;
```

ⓘ Only these compiler versions are safe to compile your code: 0.5.10

INSECURE_COMPILER_VERSION

Line 1 in File ERC20.sol

```
pragma solidity ^0.4.24;
```

❗ Version to compile has the following bug: 0.4.24: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2

INSECURE_COMPILER_VERSION

Line 1 in File ERC20Mintable.sol

```
pragma solidity ^0.4.24;
```

❗ Version to compile has the following bug: 0.4.24: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2

INSECURE_COMPILER_VERSION

Line 1 in File ERC20Detailed.sol

```
pragma solidity ^0.4.24;
```

❗ Version to compile has the following bug: 0.4.24: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2

INSECURE_COMPILER_VERSION

Line 1 in File ERC20Capped.sol

```
1 pragma solidity ^0.4.24;
```

❗ Version to compile has the following bug: 0.4.24: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2

INSECURE_COMPILER_VERSION

Line 1 in File ERC20Burnable.sol

```
1 pragma solidity ^0.4.24;
```

❗ Version to compile has the following bug: 0.4.24: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2

INSECURE_COMPILER_VERSION

Line 1 in File Roles.sol

```
1 pragma solidity ^0.4.24;
```

❗ Version to compile has the following bug: 0.4.24: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25:

SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2

INSECURE_COMPILER_VERSION

Line 1 in File MinterRole.sol

```
pragma solidity ^0.4.24;
```

❗ Version to compile has the following bug: 0.4.24: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2

INSECURE_COMPILER_VERSION

Line 1 in File ReentrancyGuard.sol

```
pragma solidity ^0.4.24;
```

❗ Version to compile has the following bug: 0.4.24: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2

INSECURE_COMPILER_VERSION

Line 1 in File SafeMath.sol

```
pragma solidity ^0.4.24;
```

❗ Version to compile has the following bug: 0.4.24: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2

INSECURE_COMPILER_VERSION

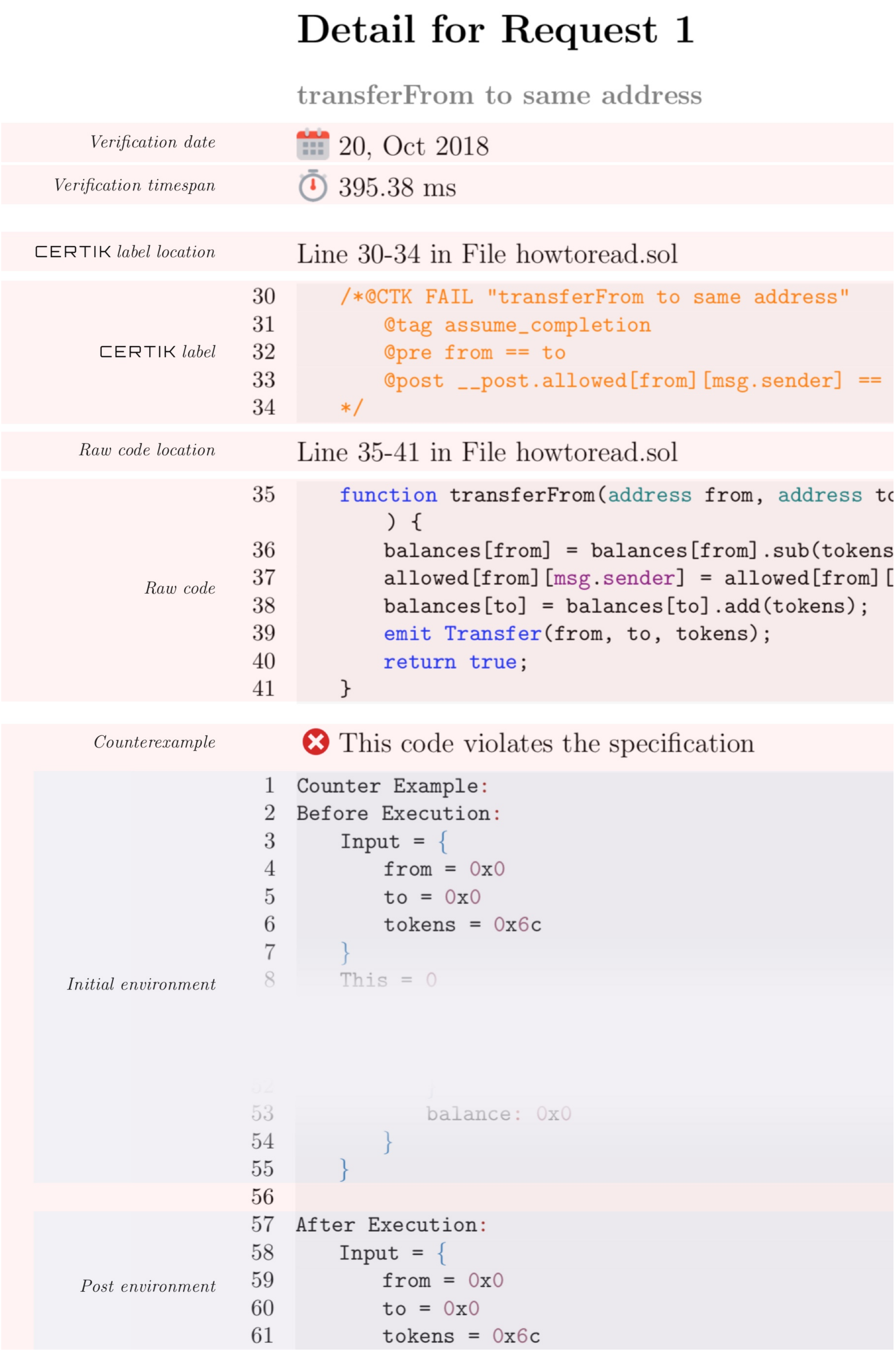
Line 1 in File Ownable.sol

```
pragma solidity ^0.4.24;
```

❗ Version to compile has the following bug: 0.4.24: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2

Formal Verification Results

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*

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

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

Raw code

```
function transferFrom(address from, address to
    ) {
    balances[from] = balances[from].sub(tokens
    allowed[from][msg.sender] = allowed[from][
    balances[to] = balances[to].add(tokens);
    emit Transfer(from, to, tokens);
    return true;
}
```

Counterexample ✖ This code violates the specification

Initial environment

```
Counter Example:
Before Execution:
    Input = {
        from = 0x0
        to = 0x0
        tokens = 0x6c
    }
    This = 0

              balance: 0x0
        }
    }

```

Post environment

```
After Execution:
    Input = {
        from = 0x0
        to = 0x0
        tokens = 0x6c
```

Formal Verification Request 1

PLA

📅 18, Sep 2019
⏱ 218.1 ms

Line 9-14 in File PLA.sol

```
 9  /*@CTK PLA
10    @tag assume_completion
11    @pre _decimals == 0
12    @post __post._totalSupply == _totalSupply + _value
13    @post __post._balances[msg.sender] == _balances[msg.sender] + _value
14   */
```

Line 15-29 in File PLA.sol

```
15  constructor (
16      string memory _name,
17      string memory _symbol,
18      uint256 _value,
19      uint8 _decimals,
20      uint256 _cap
21  )
22      ERC20Detailed (_name , _symbol , _decimals )
23      // ERC20Burnable ()
24      // ERC20Capped (_cap)
25      public
26  {
27      uint256 value = _value * (10 ** uint256(_decimals));
28      _mint(msg.sender, value);
29  }
```

✅ The code meets the specification.

Formal Verification Request 2

TokensPurchased

📅 18, Sep 2019
⏱ 7.89 ms

Line 21-23 in File TokenPurchased.sol

```
21  /*@CTK TokensPurchased
22    @post __post._owner == msg.sender
23   */
```

Line 24-29 in File TokenPurchased.sol

```
24  constructor(IERC20 _token)
25      public
26  {
27      _owner = msg.sender;
28      token = IERC20(_token);
29  }
```

✅ The code meets the specification.

Formal Verification Request 3

validateCheck

📅 18, Sep 2019
⏱ 28.95 ms

Line 53-57 in File TokenPurchased.sol

```
53    /*@CTK validateCheck
54      @tag assume_completion
55      @post _to != address(0)
56      @post _amount > 0
57     */
```

Line 58-62 in File TokenPurchased.sol

```
58    function validateCheck(address _to, uint256 _amount) internal view {
59        require(_to != address(0));
60        require(_amount > 0);
61        require(token.balanceOf(address(this)) >= _amount);
62    }
```

✓ The code meets the specification.

Formal Verification Request 4

totalSupply

📅 18, Sep 2019
⏱ 4.91 ms

Line 25-27 in File ERC20.sol

```
25  /*@CTK totalSupply
26    @post __return == _totalSupply
27   */
```

Line 28-30 in File ERC20.sol

```
28  function totalSupply() public view returns (uint256) {
29    return _totalSupply;
30  }
```

✓ The code meets the specification.

Formal Verification Request 5

balanceOf

📅 18, Sep 2019
⏱ 5.0 ms

Line 37-39 in File ERC20.sol

```
37  /*@CTK balanceOf
38    @post __return == _balances[owner]
39   */
```

Line 40-42 in File ERC20.sol

```
function balanceOf(address owner) public view returns (uint256) {
  return _balances[owner];
}
```

✓ The code meets the specification.

Formal Verification Request 6

allowance

📅 18, Sep 2019
⏱ 5.28 ms

Line 50-52 in File ERC20.sol

```
/*@CTK allowance
  @post __return == _allowed[owner][spender]
 */
```

Line 53-62 in File ERC20.sol

```
function allowance(
  address owner,
  address spender
 )
  public
  view
  returns (uint256)
{
  return _allowed[owner][spender];
}
```

✓ The code meets the specification.

Formal Verification Request 7

transfer

📅 18, Sep 2019
⏱ 280.36 ms

Line 69-76 in File ERC20.sol

```
/*@CTK transfer
  @tag assume_completion
  @pre msg.sender != to
  @post to != address(0)
  @post value <= _balances[msg.sender]
  @post __post._balances[to] == _balances[to] + value
  @post __post._balances[msg.sender] == _balances[msg.sender] - value
 */
```

Line 77-80 in File ERC20.sol

```
77  function transfer(address to, uint256 value) public returns (bool) {
78    _transfer(msg.sender, to, value);
79    return true;
80  }
```

✓ The code meets the specification.

Formal Verification Request 8

approve

18, Sep 2019
14.8 ms

Line 91-95 in File ERC20.sol

```
91  /*@CTK approve
92    @tag assume_completion
93    @post spender != address(0)
94    @post __post._allowed[msg.sender][spender] == value
95   */
```

Line 96-102 in File ERC20.sol

```
96  function approve(address spender, uint256 value) public returns (bool) {
97    require(spender != address(0));
98
99    _allowed[msg.sender][spender] = value;
100   emit Approval(msg.sender, spender, value);
101   return true;
102 }
```

✓ The code meets the specification.

Formal Verification Request 9

transfer__from

18, Sep 2019
206.76 ms

Line 110-119 in File ERC20.sol

```
110 /*@CTK transfer_from
111   @tag assume_completion
112   @pre from != to
113   @post to != address(0)
114   @post value <= _allowed[from][msg.sender]
115   @post __post._balances[from] == _balances[from] - value
116   @post __post._balances[to] == _balances[to] + value
117   @post __post._allowed[from][msg.sender] ==
118     _allowed[from][msg.sender] - value
119  */
```

Line 120-133 in File ERC20.sol

```
120  function transferFrom(
121    address from,
122    address to,
123    uint256 value
124  )
125    public
126    returns (bool)
127  {
128    require(value <= _allowed[from][msg.sender]);
129
130    _allowed[from][msg.sender] = _allowed[from][msg.sender].sub(value);
131    _transfer(from, to, value);
132    return true;
133  }
```

✓ The code meets the specification.

Formal Verification Request 10

increaseAllowance

📅 18, Sep 2019
⏱ 42.42 ms

Line 144-149 in File ERC20.sol

```
144  /*@CTK increaseAllowance
145    @tag assume_completion
146    @post spender != address(0)
147    @post __post._allowed[msg.sender][spender] ==
148          _allowed[msg.sender][spender] + addedValue
149   */
```

Line 150-163 in File ERC20.sol

```
150  function increaseAllowance(
151    address spender,
152    uint256 addedValue
153  )
154    public
155    returns (bool)
156  {
157    require(spender != address(0));
158
159    _allowed[msg.sender][spender] = (
160      _allowed[msg.sender][spender].add(addedValue));
161    emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
162    return true;
163  }
```

✓ The code meets the specification.

Formal Verification Request 11

decreaseAllowance

📅 18, Sep 2019

⏱ 47.29 ms

Line 174-179 in File ERC20.sol

```
/*@CTK decreaseAllowance
  @tag assume_completion
  @post spender != address(0)
  @post __post._allowed[msg.sender][spender] ==
        _allowed[msg.sender][spender] - subtractedValue
 */
```

Line 180-193 in File ERC20.sol

```
function decreaseAllowance(
  address spender,
  uint256 subtractedValue
)
  public
  returns (bool)
{
  require(spender != address(0));

  _allowed[msg.sender][spender] = (
    _allowed[msg.sender][spender].sub(subtractedValue));
  emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
  return true;
}
```

✓ The code meets the specification.

Formal Verification Request 12

__transfer

📅 18, Sep 2019
⏱ 99.95 ms

Line 201-208 in File ERC20.sol

```
/*@CTK _transfer
  @tag assume_completion
  @pre from != to
  @post to != address(0)
  @post value <= _balances[from]
  @post __post._balances[to] == _balances[to] + value
  @post __post._balances[from] == _balances[from] - value
 */
```

Line 209-216 in File ERC20.sol

```
function _transfer(address from, address to, uint256 value) internal {
  require(value <= _balances[from]);
  require(to != address(0));

  _balances[from] = _balances[from].sub(value);
  _balances[to] = _balances[to].add(value);
  emit Transfer(from, to, value);
}
```

✓ The code meets the specification.

Formal Verification Request 13

_mint

📅 18, Sep 2019
⏱ 82.84 ms

Line 225-230 in File ERC20.sol

```
225  /*@CTK _mint
226    @tag assume_completion
227    @post account != 0
228    @post __post._totalSupply == _totalSupply + value
229    @post __post._balances[account] == _balances[account] + value
230   */
```

Line 231-236 in File ERC20.sol

```
231  function _mint(address account, uint256 value) internal {
232    require(account != 0);
233    _totalSupply = _totalSupply.add(value);
234    _balances[account] = _balances[account].add(value);
235    emit Transfer(address(0), account, value);
236  }
```

✓ The code meets the specification.

Formal Verification Request 14

_burn

📅 18, Sep 2019
⏱ 160.69 ms

Line 244-250 in File ERC20.sol

```
244  /*@CTK _burn
245    @tag assume_completion
246    @post account != 0
247    @post value <= _balances[account]
248    @post __post._totalSupply == _totalSupply - value
249    @post __post._balances[account] == _balances[account] - value
250   */
```

Line 251-258 in File ERC20.sol

```
251  function _burn(address account, uint256 value) internal {
252    require(account != 0);
253    require(value <= _balances[account]);
254
255    _totalSupply = _totalSupply.sub(value);
256    _balances[account] = _balances[account].sub(value);
257    emit Transfer(account, address(0), value);
258  }
```

✓ The code meets the specification.

Formal Verification Request 15

__burnFrom

📅 18, Sep 2019
⏱ 251.93 ms

Line 267-273 in File ERC20.sol

```
267  /*@CTK _burnFrom
268    @tag assume_completion
269    @post value <= _allowed[account][msg.sender]
270    @post __post._allowed[account][msg.sender] == _allowed[account][msg.sender] - value
271    @post __post._totalSupply == _totalSupply - value
272    @post __post._balances[account] == _balances[account] - value
273   */
```

Line 274-282 in File ERC20.sol

```
274  function _burnFrom(address account, uint256 value) internal {
275    require(value <= _allowed[account][msg.sender]);
276
277    // Should https://github.com/OpenZeppelin/zeppelin-solidity/issues/707 be accepted,
278    // this function needs to emit an event with the updated approval.
279    _allowed[account][msg.sender] = _allowed[account][msg.sender].sub(
280      value);
281    _burn(account, value);
282  }
```

✓ The code meets the specification.

Formal Verification Request 16

mint

📅 18, Sep 2019
⏱ 193.91 ms

Line 17-20 in File ERC20Mintable.sol

```
17  /*@CTK mint
18    @tag assume_completion
19    @post minters.bearer[msg.sender]
20   */
```

Line 21-31 in File ERC20Mintable.sol

```
21  function mint(
22    address to,
23    uint256 value
24  )
25    public
26    onlyMinter
27    returns (bool)
28  {
29    _mint(to, value);
30    return true;
31  }
```

✓ The code meets the specification.

Formal Verification Request 17

ERC20Detailed

📅 18, Sep 2019
⏱ 8.14 ms

Line 16-20 in File ERC20Detailed.sol

```
16  /*@CTK ERC20Detailed
17    @post __post._name == name
18    @post __post._symbol == symbol
19    @post __post._decimals == decimals
20   */
```

Line 21-25 in File ERC20Detailed.sol

```
21  constructor(string name, string symbol, uint8 decimals) public {
22    _name = name;
23    _symbol = symbol;
24    _decimals = decimals;
25  }
```

✓ The code meets the specification.

Formal Verification Request 18

name

📅 18, Sep 2019
⏱ 5.2 ms

Line 30-32 in File ERC20Detailed.sol

```
30  /*@CTK name
31    @post __post._name == _name
32   */
```

Line 33-35 in File ERC20Detailed.sol

```
33  function name() public view returns(string) {
34    return _name;
35  }
```

✓ The code meets the specification.

Formal Verification Request 19

symbol

📅 18, Sep 2019
⏱ 5.11 ms

Line 40-42 in File ERC20Detailed.sol

```
40  /*@CTK symbol
41    @post __return == _symbol
42   */
```

Line 43-45 in File ERC20Detailed.sol

```
43  function symbol() public view returns(string) {
44    return _symbol;
45  }
```

✓ The code meets the specification.

Formal Verification Request 20

decimals

📅 18, Sep 2019
⏱ 4.75 ms

Line 50-52 in File ERC20Detailed.sol

```
50  /*@CTK decimals
51    @post __return == _decimals
52   */
```

Line 53-55 in File ERC20Detailed.sol

```
53  function decimals() public view returns(uint8) {
54    return _decimals;
55  }
```

✓ The code meets the specification.

Formal Verification Request 21

ERC20Capped

📅 18, Sep 2019
⏱ 12.03 ms

Line 13-17 in File ERC20Capped.sol

```
13  /*@CTK ERC20Capped
14    @tag assume_completion
15    @post cap > 0
16    @post __post._cap == cap
17   */
```

Line 18-23 in File ERC20Capped.sol

```
18  constructor(uint256 cap)
19    public
20  {
21    require(cap > 0);
22    _cap = cap;
23  }
```

✓ The code meets the specification.

Formal Verification Request 22

cap

📅 18, Sep 2019
⏱ 4.47 ms

Line 28-30 in File ERC20Capped.sol

```
28  /*@CTK cap
29    @post __return == _cap
30   */
```

Line 31-33 in File ERC20Capped.sol

```
31  function cap() public view returns(uint256) {
32    return _cap;
33  }
```

✓ The code meets the specification.

Formal Verification Request 23

_mint

📅 18, Sep 2019
⏱ 461.15 ms

Line 35-40 in File ERC20Capped.sol

```
35  /*@CTK _mint
36    @tag assume_completion
37    @post __post._totalSupply == _totalSupply + value
38    @post __post._totalSupply <= _cap
39    @post __post._balances[account] == _balances[account] + value
40   */
```

Line 41-44 in File ERC20Capped.sol

```
41  function _mint(address account, uint256 value) internal {
42    require(totalSupply().add(value) <= _cap);
43    super._mint(account, value);
44  }
```

✓ The code meets the specification.

Formal Verification Request 24

burn

📅 18, Sep 2019
⏱ 206.77 ms

Line 15-19 in File ERC20Burnable.sol

```
15  /*@CTK burn
16    @tag assume_completion
17    @post __post._totalSupply == _totalSupply - value
18    @post __post._balances[msg.sender] == _balances[msg.sender] - value
19   */
```

Line 20-22 in File ERC20Burnable.sol

```
20  function burn(uint256 value) public {
21    _burn(msg.sender, value);
22  }
```

✓ The code meets the specification.

Formal Verification Request 25

burnFrom

📅 18, Sep 2019
⏱ 368.74 ms

Line 29-33 in File ERC20Burnable.sol

```
29  /*@CTK burnFrom
30    @tag assume_completion
31    @post __post._totalSupply == _totalSupply - value
32    @post __post._balances[from] == _balances[from] - value
33   */
```

Line 34-36 in File ERC20Burnable.sol

```
34  function burnFrom(address from, uint256 value) public {
35    _burnFrom(from, value);
36  }
```

✓ The code meets the specification.

Formal Verification Request 26

has

📅 18, Sep 2019
⏱ 13.04 ms

Line 48-52 in File Roles.sol

```
48  /*@CTK has
49    @tag assume_completion
50    @post account != address(0)
51    @post __return == role.bearer[account]
52   */
```

Line 53-60 in File Roles.sol

```
53  function has(Role storage role, address account)
54    internal
55    view
56    returns (bool)
```

```
{
  require(account != address(0));
  return role.bearer[account];
}
```

✓ The code meets the specification.

Formal Verification Request 27

isMinter

📅 18, Sep 2019
⏱ 45.67 ms

Line 22-25 in File MinterRole.sol

```
/*@CTK isMinter
  @tag assume_completion
  @post __return == minters.bearer[account]
 */
```

Line 26-28 in File MinterRole.sol

```
function isMinter(address account) public view returns (bool) {
  return minters.has(account);
}
```

✓ The code meets the specification.

Formal Verification Request 28

ReentrancyGuard

📅 18, Sep 2019
⏱ 4.79 ms

Line 13-15 in File ReentrancyGuard.sol

```
/*@CTK ReentrancyGuard
  @post __post._guardCounter == 1
 */
```

Line 16-20 in File ReentrancyGuard.sol

```
constructor() internal {
  // The counter starts at one to prevent changing it from zero to a non-zero
  // value, which is a more expensive operation.
  _guardCounter = 1;
}
```

✓ The code meets the specification.

Formal Verification Request 29

SafeMath mul

📅 18, Sep 2019
⏱ 291.52 ms

Line 12-17 in File SafeMath.sol

```
/*@CTK "SafeMath mul"
  @post (a > 0) && (((a * b) / a) != b) -> __reverted
  @post __reverted -> (a > 0) && (((a * b) / a) != b)
  @post !__reverted -> __return == a * b
  @post !__reverted == !__has_overflow
 */
```

Line 18-30 in File SafeMath.sol

```
function mul(uint256 a, uint256 b) internal pure returns (uint256) {
  // Gas optimization: this is cheaper than requiring 'a' not being zero, but the
  // benefit is lost if 'b' is also tested.
  // See: https://github.com/OpenZeppelin/openzeppelin-solidity/pull/522
  if (a == 0) {
    return 0;
  }

  uint256 c = a * b;
  require(c / a == b);

  return c;
}
```

✓ The code meets the specification.

Formal Verification Request 30

SafeMath div

📅 18, Sep 2019
⏱ 12.31 ms

Line 35-39 in File SafeMath.sol

```
/*@CTK "SafeMath div"
  @post b != 0 -> !__reverted
  @post !__reverted -> __return == a / b
  @post !__reverted -> !__has_overflow
 */
```

Line 40-46 in File SafeMath.sol

```
function div(uint256 a, uint256 b) internal pure returns (uint256) {
  require(b > 0); // Solidity only automatically asserts when dividing by 0
  uint256 c = a / b;
  // assert(a == b * c + a % b); // There is no case in which this doesn't hold

  return c;
}
```

✓ The code meets the specification.

Formal Verification Request 31

SafeMath sub

📅 18, Sep 2019
⏱ 11.42 ms

Line 51-55 in File SafeMath.sol

```
/*@CTK "SafeMath sub"
  @post (a < b) == __reverted
  @post !__reverted -> __return == a - b
  @post !__reverted -> !__has_overflow
*/
```

Line 56-61 in File SafeMath.sol

```
function sub(uint256 a, uint256 b) internal pure returns (uint256) {
  require(b <= a);
  uint256 c = a - b;

  return c;
}
```

✓ The code meets the specification.

Formal Verification Request 32

SafeMath add

📅 18, Sep 2019
⏱ 12.32 ms

Line 66-70 in File SafeMath.sol

```
/*@CTK "SafeMath add"
  @post (a + b < a || a + b < b) == __reverted
  @post !__reverted -> __return == a + b
  @post !__reverted -> !__has_overflow
 */
```

Line 71-76 in File SafeMath.sol

```
function add(uint256 a, uint256 b) internal pure returns (uint256) {
  uint256 c = a + b;
  require(c >= a);

  return c;
}
```

✓ The code meets the specification.

Formal Verification Request 33

SafeMath mod

📅 18, Sep 2019
⏱ 10.83 ms

Line 82-87 in File SafeMath.sol

```
82  /*@CTK "SafeMath mod"
83    @post (b == 0) == __reverted
84    @post !__reverted -> b != 0
85    @post !__reverted -> __return == a % b
86    @post !__reverted -> !__has_overflow
87   */
```

Line 88-91 in File SafeMath.sol

```
88  function mod(uint256 a, uint256 b) internal pure returns (uint256) {
89    require(b != 0);
90    return a % b;
91  }
```

✓ The code meets the specification.

Formal Verification Request 34

Ownable

📅 18, Sep 2019
⏱ 5.05 ms

Line 20-22 in File Ownable.sol

```
20  /*@CTK Ownable
21    @post __post._owner == msg.sender
22   */
```

Line 23-26 in File Ownable.sol

```
23  constructor() internal {
24    _owner = msg.sender;
25    emit OwnershipTransferred(address(0), _owner);
26  }
```

✓ The code meets the specification.

Formal Verification Request 35

owner

📅 18, Sep 2019
⏱ 5.09 ms

Line 31-33 in File Ownable.sol

```
31  /*@CTK owner
32    @post __return == _owner
33   */
```

Line 34-36 in File Ownable.sol

```
34  function owner() public view returns(address) {
35    return _owner;
36  }
```

✓ The code meets the specification.

Formal Verification Request 36

isOwner

📅 18, Sep 2019
⏱ 5.71 ms

Line 49-51 in File Ownable.sol

```
49  /*@CTK isOwner
50    @post __return == (msg.sender == _owner)
51   */
```

Line 52-54 in File Ownable.sol

```
52  function isOwner() public view returns(bool) {
53    return msg.sender == _owner;
54  }
```

✓ The code meets the specification.

Formal Verification Request 37

renounceOwnership

📅 18, Sep 2019
⏱ 22.25 ms

Line 62-66 in File Ownable.sol

```
62  /*@CTK renounceOwnership
63    @tag assume_completion
64    @post _owner == msg.sender
65    @post __post._owner == address(0)
66   */
```

Line 67-70 in File Ownable.sol

```
67  function renounceOwnership() public onlyOwner {
68    emit OwnershipTransferred(_owner, address(0));
69    _owner = address(0);
70  }
```

✓ The code meets the specification.

Formal Verification Request 38

transferOwnership

📅 18, Sep 2019
⏱ 52.6 ms

Line 76-79 in File Ownable.sol

```
76  /*@CTK transferOwnership
77    @tag assume_completion
78    @post _owner == msg.sender
79   */
```

Line 80-82 in File Ownable.sol

```
80  function transferOwnership(address newOwner) public onlyOwner {
81    _transferOwnership(newOwner);
82  }
```

✓ The code meets the specification.

Formal Verification Request 39

_transferOwnership

📅 18, Sep 2019
⏱ 1.34 ms

Line 88-92 in File Ownable.sol

```
88  /*@CTK _transferOwnership
89    @tag assume_completion
90    @post newOwner != address(0)
91    @post __post._owner == newOwner
92   */
```

Line 93-97 in File Ownable.sol

```
93  function _transferOwnership(address newOwner) internal {
94    require(newOwner != address(0));
95    emit OwnershipTransferred(_owner, newOwner);
96    _owner = newOwner;
97  }
```

✓ The code meets the specification.

Source Code with CertiK Labels

File PLA.sol

```
pragma solidity ^0.5.0;

import "openzeppelin-solidity/contracts/token/ERC20/ERC20.sol";
import "openzeppelin-solidity/contracts/token/ERC20/ERC20Detailed.sol";
import "openzeppelin-solidity/contracts/token/ERC20/ERC20Burnable.sol";
import "openzeppelin-solidity/contracts/token/ERC20/ERC20Capped.sol";

contract PLA is ERC20, ERC20Detailed, ERC20Capped, ERC20Burnable {
    /*@CTK PLA
      @tag assume_completion
      @pre _decimals == 0
      @post __post._totalSupply == _totalSupply + _value
      @post __post._balances[msg.sender] == _balances[msg.sender] + _value
     */
    constructor (
        string memory _name,
        string memory _symbol,
        uint256 _value,
        uint8 _decimals,
        uint256 _cap
    )
        ERC20Detailed (_name , _symbol , _decimals )
        // ERC20Burnable ()
        // ERC20Capped (_cap)
        public
    {
        uint256 value = _value * (10 ** uint256(_decimals));
        _mint(msg.sender, value);
    }
}
```

File TokenPurchased.sol

```
pragma solidity ^0.5.0;

import "openzeppelin-solidity/contracts/token/ERC20/SafeERC20.sol";
import "openzeppelin-solidity/contracts/token/ERC20/IERC20.sol";
import "openzeppelin-solidity/contracts/utils/ReentrancyGuard.sol";
import "openzeppelin-solidity/contracts/math/SafeMath.sol";
import "openzeppelin-solidity/contracts/ownership/Ownable.sol";

contract TokensPurchased is ReentrancyGuard,Ownable {
    using SafeMath for uint256;
    using SafeERC20 for IERC20;
    IERC20 private token;

    uint256 public tokensSold;

    event EventPurchased(address _to, uint256 _value);
    event EventAirdrop(address _to, uint256 _value);

    address public _owner;

    /*@CTK TokensPurchased
      @post __post._owner == msg.sender
```

```
     */
    constructor(IERC20 _token)
        public
    {
        _owner = msg.sender;
        token = IERC20(_token);
    }

    function () public payable{
        buyTokens(msg.sender, msg.value);
    }


    function buyTokens(address _to, uint256 _amount) internal nonReentrant { // whenNotPaused
        validateCheck(_to, _amount*10000);

        token.safeTransfer(_to, _amount*10000);

        _owner.transfer(address(this).balance);
        emit EventPurchased(_to, _amount);
    }


    function airDrop(address _to, uint256 _amount) public nonReentrant onlyOwner { // whenNotPaused
        validateCheck(_to, _amount);

        token.safeTransfer(_to, _amount);
        emit EventAirdrop(_to, _amount);
    }

    /*@CTK validateCheck
      @tag assume_completion
      @post _to != address(0)
      @post _amount > 0
     */
    function validateCheck(address _to, uint256 _amount) internal view {
        require(_to != address(0));
        require(_amount > 0);
        require(token.balanceOf(address(this)) >= _amount);
    }
}
```

File openzeppelin-solidity/contracts/token/ERC20/ERC20.sol

```
pragma solidity ^0.4.24;

import "./IERC20.sol";
import "../../math/SafeMath.sol";

/**
 * @title Standard ERC20 token
 *
 * @dev Implementation of the basic standard token.
 * https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md
 * Originally based on code by FirstBlood: https://github.com/Firstbloodio/token/blob/master/smart_contract/FirstBloodToken.sol
 */
```

```
contract ERC20 is IERC20 {
  using SafeMath for uint256;

  mapping (address => uint256) private _balances;

  mapping (address => mapping (address => uint256)) private _allowed;

  uint256 private _totalSupply;

  /**
  * @dev Total number of tokens in existence
  */
  /*@CTK totalSupply
    @post __return == _totalSupply
   */
  function totalSupply() public view returns (uint256) {
    return _totalSupply;
  }

  /**
  * @dev Gets the balance of the specified address.
  * @param owner The address to query the balance of.
  * @return An uint256 representing the amount owned by the passed address.
  */
  /*@CTK balanceOf
    @post __return == _balances[owner]
   */
  function balanceOf(address owner) public view returns (uint256) {
    return _balances[owner];
  }

  /**
   * @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 allowance
    @post __return == _allowed[owner][spender]
   */
  function allowance(
    address owner,
    address spender
   )
    public
    view
    returns (uint256)
  {
    return _allowed[owner][spender];
  }

  /**
  * @dev Transfer token for a specified address
  * @param to The address to transfer to.
  * @param value The amount to be transferred.
  */
  /*@CTK transfer
```

```
    @tag assume_completion
    @pre msg.sender != to
    @post to != address(0)
    @post value <= _balances[msg.sender]
    @post __post._balances[to] == _balances[to] + value
    @post __post._balances[msg.sender] == _balances[msg.sender] - value
   */
  function transfer(address to, uint256 value) public returns (bool) {
    _transfer(msg.sender, to, value);
    return true;
  }

  /**
   * @dev Approve the passed address to spend the specified amount of tokens on behalf of msg.sender.
   * Beware that changing an allowance with this method brings the risk that someone may use both the old
   * and the new allowance by unfortunate transaction ordering. One possible solution to mitigate this
   * race condition is to first reduce the spender's allowance to 0 and set the desired value afterwards:
   * https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729
   * @param spender The address which will spend the funds.
   * @param value The amount of tokens to be spent.
   */
  /*@CTK approve
    @tag assume_completion
    @post spender != address(0)
    @post __post._allowed[msg.sender][spender] == value
   */
  function approve(address spender, uint256 value) public returns (bool) {
    require(spender != address(0));

    _allowed[msg.sender][spender] = value;
    emit Approval(msg.sender, spender, value);
    return true;
  }

  /**
   * @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 transfer_from
    @tag assume_completion
    @pre from != to
    @post to != address(0)
    @post value <= _allowed[from][msg.sender]
    @post __post._balances[from] == _balances[from] - value
    @post __post._balances[to] == _balances[to] + value
    @post __post._allowed[from][msg.sender] ==
      _allowed[from][msg.sender] - value
   */
  function transferFrom(
    address from,
    address to,
    uint256 value
```

```
)
  public
  returns (bool)
{
  require(value <= _allowed[from][msg.sender]);

  _allowed[from][msg.sender] = _allowed[from][msg.sender].sub(value);
  _transfer(from, to, value);
  return true;
}

/**
 * @dev Increase the amount of tokens that an owner allowed to a spender.
 * approve should be called when allowed_[_spender] == 0. To increment
 * allowed value is better to use this function to avoid 2 calls (and wait until
 * the first transaction is mined)
 * From MonolithDAO Token.sol
 * @param spender The address which will spend the funds.
 * @param addedValue The amount of tokens to increase the allowance by.
 */
/*@CTK increaseAllowance
  @tag assume_completion
  @post spender != address(0)
  @post __post._allowed[msg.sender][spender] ==
        _allowed[msg.sender][spender] + addedValue
 */
function increaseAllowance(
  address spender,
  uint256 addedValue
)
  public
  returns (bool)
{
  require(spender != address(0));

  _allowed[msg.sender][spender] = (
    _allowed[msg.sender][spender].add(addedValue));
  emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
  return true;
}

/**
 * @dev Decrease the amount of tokens that an owner allowed to a spender.
 * approve should be called when allowed_[_spender] == 0. To decrement
 * allowed value is better to use this function to avoid 2 calls (and wait until
 * the first transaction is mined)
 * From MonolithDAO Token.sol
 * @param spender The address which will spend the funds.
 * @param subtractedValue The amount of tokens to decrease the allowance by.
 */
/*@CTK decreaseAllowance
  @tag assume_completion
  @post spender != address(0)
  @post __post._allowed[msg.sender][spender] ==
        _allowed[msg.sender][spender] - subtractedValue
 */
function decreaseAllowance(
  address spender,
  uint256 subtractedValue
)
  public
  returns (bool)
{
  require(spender != address(0));

  _allowed[msg.sender][spender] = (
    _allowed[msg.sender][spender].sub(subtractedValue));
  emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
  return true;
}

/**
* @dev Transfer token for a specified addresses
* @param from The address to transfer from.
* @param to The address to transfer to.
* @param value The amount to be transferred.
*/
/*@CTK _transfer
  @tag assume_completion
  @pre from != to
  @post to != address(0)
  @post value <= _balances[from]
  @post __post._balances[to] == _balances[to] + value
  @post __post._balances[from] == _balances[from] - value
 */
function _transfer(address from, address to, uint256 value) internal {
  require(value <= _balances[from]);
  require(to != address(0));

  _balances[from] = _balances[from].sub(value);
  _balances[to] = _balances[to].add(value);
  emit Transfer(from, to, value);
}

/**
 * @dev Internal function that mints an amount of the token and assigns it to
 * an account. This encapsulates the modification of balances such that the
 * proper events are emitted.
 * @param account The account that will receive the created tokens.
 * @param value The amount that will be created.
 */
/*@CTK _mint
  @tag assume_completion
  @post account != 0
  @post __post._totalSupply == _totalSupply + value
  @post __post._balances[account] == _balances[account] + value
 */
function _mint(address account, uint256 value) internal {
  require(account != 0);
  _totalSupply = _totalSupply.add(value);
  _balances[account] = _balances[account].add(value);
  emit Transfer(address(0), account, value);
}

/**
 * @dev Internal function that burns an amount of the token of a given
```

```
   * account.
   * @param account The account whose tokens will be burnt.
   * @param value The amount that will be burnt.
   */
  /*@CTK _burn
    @tag assume_completion
    @post account != 0
    @post value <= _balances[account]
    @post __post._totalSupply == _totalSupply - value
    @post __post._balances[account] == _balances[account] - value
   */
  function _burn(address account, uint256 value) internal {
    require(account != 0);
    require(value <= _balances[account]);

    _totalSupply = _totalSupply.sub(value);
    _balances[account] = _balances[account].sub(value);
    emit Transfer(account, address(0), value);
  }

  /**
   * @dev Internal function that burns an amount of the token of a given
   * account, deducting from the sender's allowance for said account. Uses the
   * internal burn function.
   * @param account The account whose tokens will be burnt.
   * @param value The amount that will be burnt.
   */
  /*@CTK _burnFrom
    @tag assume_completion
    @post value <= _allowed[account][msg.sender]
    @post __post._allowed[account][msg.sender] == _allowed[account][msg.sender] - value
    @post __post._totalSupply == _totalSupply - value
    @post __post._balances[account] == _balances[account] - value
   */
  function _burnFrom(address account, uint256 value) internal {
    require(value <= _allowed[account][msg.sender]);

    // Should https://github.com/OpenZeppelin/zeppelin-solidity/issues/707 be accepted,
    // this function needs to emit an event with the updated approval.
    _allowed[account][msg.sender] = _allowed[account][msg.sender].sub(
      value);
    _burn(account, value);
  }
}
```

File openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol

```
pragma solidity ^0.4.24;

import "./ERC20.sol";
import "../../access/roles/MinterRole.sol";

/**
 * @title ERC20Mintable
 * @dev ERC20 minting logic
 */
contract ERC20Mintable is ERC20, MinterRole {
  /**
   * @dev Function to mint tokens
   * @param to The address that will receive the minted tokens.
   * @param value The amount of tokens to mint.
   * @return A boolean that indicates if the operation was successful.
   */
  /*@CTK mint
    @tag assume_completion
    @post minters.bearer[msg.sender]
   */
  function mint(
    address to,
    uint256 value
  )
    public
    onlyMinter
    returns (bool)
  {
    _mint(to, value);
    return true;
  }
}
```

File openzeppelin-solidity/contracts/token/ERC20/ERC20Detailed.sol

```
pragma solidity ^0.4.24;

import "./IERC20.sol";

/**
 * @title ERC20Detailed token
 * @dev The decimals are only for visualization purposes.
 * All the operations are done using the smallest and indivisible token unit,
 * just as on Ethereum all the operations are done in wei.
 */
contract ERC20Detailed is IERC20 {
  string private _name;
  string private _symbol;
  uint8 private _decimals;

  /*@CTK ERC20Detailed
    @post __post._name == name
    @post __post._symbol == symbol
    @post __post._decimals == decimals
   */
  constructor(string name, string symbol, uint8 decimals) public {
    _name = name;
    _symbol = symbol;
    _decimals = decimals;
  }

  /**
   * @return the name of the token.
   */
  /*@CTK name
    @post __post._name == _name
   */
  function name() public view returns(string) {
    return _name;
  }

  /**
   * @return the symbol of the token.
   */
  /*@CTK symbol
    @post __return == _symbol
   */
  function symbol() public view returns(string) {
    return _symbol;
  }

  /**
   * @return the number of decimals of the token.
   */
  /*@CTK decimals
    @post __return == _decimals
   */
  function decimals() public view returns(uint8) {
    return _decimals;
  }
}
```

File openzeppelin-solidity/contracts/token/ERC20/ERC20Capped.sol

```
pragma solidity ^0.4.24;

import "./ERC20Mintable.sol";

/**
 * @title Capped token
 * @dev Mintable token with a token cap.
 */
contract ERC20Capped is ERC20Mintable {

  uint256 private _cap;

  /*@CTK ERC20Capped
    @tag assume_completion
    @post cap > 0
    @post __post._cap == cap
   */
  constructor(uint256 cap)
    public
  {
    require(cap > 0);
    _cap = cap;
  }

  /**
   * @return the cap for the token minting.
   */
  /*@CTK cap
    @post __return == _cap
   */
  function cap() public view returns(uint256) {
    return _cap;
  }

  /*@CTK _mint
    @tag assume_completion
    @post __post._totalSupply == _totalSupply + value
    @post __post._totalSupply <= _cap
    @post __post._balances[account] == _balances[account] + value
   */
  function _mint(address account, uint256 value) internal {
    require(totalSupply().add(value) <= _cap);
    super._mint(account, value);
  }
}
```

File openzeppelin-solidity/contracts/token/ERC20/ERC20Burnable.sol

```
pragma solidity ^0.4.24;

import "./ERC20.sol";

/**
 * @title Burnable Token
 * @dev Token that can be irreversibly burned (destroyed).
 */
contract ERC20Burnable is ERC20 {

  /**
   * @dev Burns a specific amount of tokens.
   * @param value The amount of token to be burned.
   */
  /*@CTK burn
    @tag assume_completion
    @post __post._totalSupply == _totalSupply - value
    @post __post._balances[msg.sender] == _balances[msg.sender] - value
   */
  function burn(uint256 value) public {
    _burn(msg.sender, value);
  }

  /**
   * @dev Burns a specific amount of tokens from the target address and decrements allowance
   * @param from address The address which you want to send tokens from
   * @param value uint256 The amount of token to be burned
   */
  /*@CTK burnFrom
    @tag assume_completion
    @post __post._totalSupply == _totalSupply - value
    @post __post._balances[from] == _balances[from] - value
   */
  function burnFrom(address from, uint256 value) public {
    _burnFrom(from, value);
  }
}
```

File openzeppelin-solidity/contracts/access/Roles.sol

```
pragma solidity ^0.4.24;

/**
 * @title Roles
 * @dev Library for managing addresses assigned to a Role.
 */
library Roles {
  struct Role {
    mapping (address => bool) bearer;
  }

  /**
   * @dev give an account access to this role
   */
  /*CTK add
    @tag assume_completion
    @post account != address(0)
    @post !role.bearer[account]
    @post __post.role.bearer[account]
   */
  function add(Role storage role, address account) internal {
    require(account != address(0));
    require(!has(role, account));

    role.bearer[account] = true;
  }

  /**
   * @dev remove an account's access to this role
   */
  /*CTK remove
    @tag assume_completion
    @post account != address(0)
    @post role.bearer[account]
    @post !__post.role.bearer[account]
   */
  function remove(Role storage role, address account) internal {
    require(account != address(0));
    require(has(role, account));

    role.bearer[account] = false;
  }

  /**
   * @dev check if an account has this role
   * @return bool
   */
  /*@CTK has
    @tag assume_completion
    @post account != address(0)
    @post __return == role.bearer[account]
   */
  function has(Role storage role, address account)
    internal
    view
    returns (bool)
  {
    require(account != address(0));
    return role.bearer[account];
  }
}
```

File openzeppelin-solidity/contracts/access/roles/MinterRole.sol

```
pragma solidity ^0.4.24;

import "../Roles.sol";

contract MinterRole {
  using Roles for Roles.Role;

  event MinterAdded(address indexed account);
  event MinterRemoved(address indexed account);

  Roles.Role private minters;

  constructor() internal {
    _addMinter(msg.sender);
  }

  modifier onlyMinter() {
    require(isMinter(msg.sender));
    _;
  }

  /*@CTK isMinter
    @tag assume_completion
    @post __return == minters.bearer[account]
   */
  function isMinter(address account) public view returns (bool) {
    return minters.has(account);
  }

  function addMinter(address account) public onlyMinter {
    _addMinter(account);
  }

  function renounceMinter() public {
    _removeMinter(msg.sender);
  }

  function _addMinter(address account) internal {
    minters.add(account);
    emit MinterAdded(account);
  }

  function _removeMinter(address account) internal {
    minters.remove(account);
    emit MinterRemoved(account);
  }
}
```

File openzeppelin-solidity/contracts/utils/ReentrancyGuard.sol

```
pragma solidity ^0.4.24;

/**
 * @title Helps contracts guard against reentrancy attacks.
 * @dev If you mark a function `nonReentrant`, you should also
 * mark it `external`.
 */
contract ReentrancyGuard {

  /// @dev counter to allow mutex lock with only one SSTORE operation
  uint256 private _guardCounter;

  /*@CTK ReentrancyGuard
    @post __post._guardCounter == 1
   */
  constructor() internal {
    // The counter starts at one to prevent changing it from zero to a non-zero
    // value, which is a more expensive operation.
    _guardCounter = 1;
  }

  /**
   * @dev Prevents a contract from calling itself, directly or indirectly.
   * Calling a `nonReentrant` function from another `nonReentrant`
   * function is not supported. It is possible to prevent this from happening
   * by making the `nonReentrant` function external, and make it call a
   * `private` function that does the actual work.
   */
  modifier nonReentrant() {
    _guardCounter += 1;
    uint256 localCounter = _guardCounter;
    _;
    require(localCounter == _guardCounter);
  }

}
```

File openzeppelin-solidity/contracts/math/SafeMath.sol

```
pragma solidity ^0.4.24;

/**
 * @title SafeMath
 * @dev Math operations with safety checks that revert on error
 */
library SafeMath {

  /**
  * @dev Multiplies two numbers, reverts on overflow.
  */
  /*@CTK "SafeMath mul"
    @post (a > 0) && (((a * b) / a) != b) -> __reverted
    @post __reverted -> (a > 0) && (((a * b) / a) != b)
    @post !__reverted -> __return == a * b
    @post !__reverted == !__has_overflow
   */
  function mul(uint256 a, uint256 b) internal pure returns (uint256) {
    // Gas optimization: this is cheaper than requiring 'a' not being zero, but the
    // benefit is lost if 'b' is also tested.
    // See: https://github.com/OpenZeppelin/openzeppelin-solidity/pull/522
    if (a == 0) {
      return 0;
    }

    uint256 c = a * b;
    require(c / a == b);

    return c;
  }

  /**
  * @dev Integer division of two numbers truncating the quotient, reverts on division by zero.
  */
  /*@CTK "SafeMath div"
    @post b != 0 -> !__reverted
    @post !__reverted -> __return == a / b
    @post !__reverted -> !__has_overflow
   */
  function div(uint256 a, uint256 b) internal pure returns (uint256) {
    require(b > 0); // Solidity only automatically asserts when dividing by 0
    uint256 c = a / b;
    // assert(a == b * c + a % b); // There is no case in which this doesn't hold

    return c;
  }

  /**
  * @dev Subtracts two numbers, reverts on overflow (i.e. if subtrahend is greater than minuend).
  */
  /*@CTK "SafeMath sub"
    @post (a < b) == __reverted
    @post !__reverted -> __return == a - b
    @post !__reverted -> !__has_overflow
  */
  function sub(uint256 a, uint256 b) internal pure returns (uint256) {
    require(b <= a);
    uint256 c = a - b;

    return c;
  }

  /**
  * @dev Adds two numbers, reverts on overflow.
  */
  /*@CTK "SafeMath add"
    @post (a + b < a || a + b < b) == __reverted
    @post !__reverted -> __return == a + b
    @post !__reverted -> !__has_overflow
   */
  function add(uint256 a, uint256 b) internal pure returns (uint256) {
    uint256 c = a + b;
    require(c >= a);

    return c;
  }

  /**
  * @dev Divides two numbers and returns the remainder (unsigned integer modulo),
  * reverts when dividing by zero.
  */
  /*@CTK "SafeMath mod"
    @post (b == 0) == __reverted
    @post !__reverted -> b != 0
    @post !__reverted -> __return == a % b
```

```
    @post !__reverted -> !__has_overflow
   */
  function mod(uint256 a, uint256 b) internal pure returns (uint256) {
    require(b != 0);
    return a % b;
  }
}
```

File openzeppelin-solidity/contracts/ownership/Ownable.sol

```
pragma solidity ^0.4.24;

/**
 * @title Ownable
 * @dev The Ownable contract has an owner address, and provides basic authorization control
 * functions, this simplifies the implementation of "user permissions".
 */
contract Ownable {
  address private _owner;

  event OwnershipTransferred(
    address indexed previousOwner,
    address indexed newOwner
  );

  /**
   * @dev The Ownable constructor sets the original `owner` of the contract to the sender
   * account.
   */
  /*@CTK Ownable
    @post __post._owner == msg.sender
   */
  constructor() internal {
    _owner = msg.sender;
    emit OwnershipTransferred(address(0), _owner);
  }

  /**
   * @return the address of the owner.
   */
  /*@CTK owner
    @post __return == _owner
   */
  function owner() public view returns(address) {
    return _owner;
  }

  /**
   * @dev Throws if called by any account other than the owner.
   */
  modifier onlyOwner() {
    require(isOwner());
    _;
  }

  /**
   * @return true if `msg.sender` is the owner of the contract.
   */
  /*@CTK isOwner
    @post __return == (msg.sender == _owner)
   */
  function isOwner() public view returns(bool) {
    return msg.sender == _owner;
  }

  /**
   * @dev Allows the current owner to relinquish control of the contract.
   * @notice Renouncing to ownership will leave the contract without an owner.
   * It will not be possible to call the functions with the `onlyOwner`
   * modifier anymore.
   */
  /*@CTK renounceOwnership
    @tag assume_completion
    @post _owner == msg.sender
    @post __post._owner == address(0)
   */
  function renounceOwnership() public onlyOwner {
    emit OwnershipTransferred(_owner, address(0));
    _owner = address(0);
  }

  /**
   * @dev Allows the current owner to transfer control of the contract to a newOwner.
   * @param newOwner The address to transfer ownership to.
   */
  /*@CTK transferOwnership
    @tag assume_completion
    @post _owner == msg.sender
   */
  function transferOwnership(address newOwner) public onlyOwner {
    _transferOwnership(newOwner);
  }

  /**
   * @dev Transfers control of the contract to a newOwner.
   * @param newOwner The address to transfer ownership to.
   */
  /*@CTK _transferOwnership
    @tag assume_completion
    @post newOwner != address(0)
    @post __post._owner == newOwner
   */
  function _transferOwnership(address newOwner) internal {
    require(newOwner != address(0));
    emit OwnershipTransferred(_owner, newOwner);
    _owner = newOwner;
  }
}
```