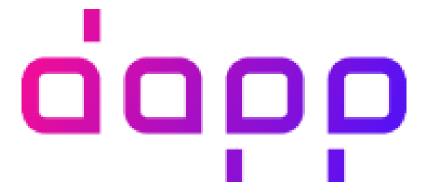
CERTIK AUDIT REPORT FOR DAPP



Request Date: 2019-06-07 Revision Date: 2019-08-06 Platform Name: Ethereum







Contents

Disclaimer	1
About CertiK	2
Exective Summary	3
Vulnerability Classification	3
Testing Summary Audit Score	4 4 4 5
Manual Review Notes	6
Static Analysis Results	7
Formal Verification Results How to read	8
Source Code with CertiK Labels	26





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 Dapp(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/





Exective Summary

This report has been prepared as the product of the Smart Contract Audit request by Dapp. This audit was conducted to discover issues and vulnerabilities in the source code of Dapp's Smart Contracts. Utilizing CertiK's Formal Verification Platform, Static Analysis, and Manual Review, a comprehensive examination has been performed. The auditing process pays special attention to the following considerations.

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessment of the codebase for 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

For every issue 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 conditions, 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 vulnerabilies, but no concern found yet.





Testing Summary

WARNING

DERTIK identified some potential security flaws in this contract and also provided corresponding solutions.





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	An overflow/underflow happens when an arithmetic	2	SWC-101
and Underflow	operation reaches the maximum or minimum size of		
	a type.		
Function incor-	Function implementation does not meet the specifi-	0	
rectness	cation, leading to intentional or unintentional vul-		
	nerabilities.		
Buffer Overflow	An attacker is able to write to arbitrary storage lo-	0	SWC-124
	cations of a contract if array of out bound happens		
Reentrancy	A malicious contract can call back into the calling	0	SWC-107
	contract before the first invocation of the function is		
	finished.		
Transaction Or-	A race condition vulnerability occurs when code de-	0	SWC-114
der Dependence	pends on the order of the transactions submitted to		
	it.		
Timestamp De-	Timestamp can be influenced by minors to some de-	2	SWC-116
pendence	gree.		
Insecure Com-	Using an fixed outdated compiler version or float-	1	SWC-102
piler Version	ing pragma can be problematic, if there are publicly		SWC-103
	disclosed bugs and issues that affect the current com-		
	piler version used.		
Insecure Ran-	Block attributes are insecure to generate random	0	SWC-120
domness	numbers, as they can be influenced by minors to		
	some degree.		





"tx.origin" for	tx.origin should not be used for authorization. Use	0	SWC-115
authorization	msg.sender instead.		
Delegatecall to	Calling into untrusted contracts is very dangerous,	0	SWC-112
Untrusted Callee	the target and arguments provided must be sani-		
	tized.		
State Variable	Labeling the visibility explicitly makes it easier to	0	SWC-108
Default Visibility	catch incorrect assumptions about who can access		
	the variable.		
Function Default	Functions are public by default. A malicious user	0	SWC-100
Visibility	is able to make unauthorized or unintended state		
	changes if a developer forgot to set the visibility.		
Uninitialized	Uninitialized local storage variables can point to	0	SWC-109
variables	other unexpected storage variables in the contract.		
Assertion Failure	The assert() function is meant to assert invariants.	0	SWC-110
	Properly functioning code should never reach a fail-		
	ing assert statement.		
Deprecated	Several functions and operators in Solidity are dep-	0	SWC-111
Solidity Features	recated and should not be used as best practice.		
Unused variables	Unused variables reduce code quality	0	

Vulnerability Details

Critical

No issue found.

Medium

No issue found.

Low

DappToken:

• formatDecimals(), allocateToken(): Potential numerical overflow. See *Review Notes* section for more details.

StandardToken:

- transfer(), transferFrom(): Potential numerical overflow. See *Review Notes* section for more details.
- transfer(), transferFrom(): Missing address check in the following methods, which may lead to value loss.





Manual Review Notes

Source Code SHA-256 Checksum

• DAPPT.sol

 $\verb|c30d8f162df79e643d597751c535b0021c5639707efcba5411bbeda0beb9736e|| \\$

Disucssions

DappToken

- MINOR formatDecimals(): Potential numerical overflow.
- MINOR allocateToken(): Potential numerical overflow but will be guarded by totalSupply.
- INFO increaseSupply(), decreaseSupply(): Potential numerical overflow to be captured by safeAdd() and safeSubtract(). However inconsistent event and resulting balance may happen in case of mistaken operation of the owner.

StandardToken

- MINOR transfer(), transferFrom(): Potential numerical overflow but will be guarded by totalSupply.
- MINOR transfer(), transferFrom(): Missing address check which may lead to value loss:
- INFO Recommend using require() for condition check in transfer and transfer-From. The use of require() permits error message to be emitted for better diagnostic of transaction failure. Example:

```
require(to != address(0), ...)
require(from != to, ...)
require(balances[from] >= value, ...)
require(allowed[from] [msg.sender] >= value, ...)
require(value > 0, ...)
```

- INFO Recommend changing the if (...) throw checks to require(..., ...).
- MINOR Recommend using the pull model instead of the push model to better secure the ownership transfer.

```
address ethFundDeposit;
address proposedEthFundDeposit;
function proposeNewOwner(address newFundDeposit) isOwner() external {
    require(newFundDeposit != address(0), ...);
    proposedEthFundDeposit = newFundDeposit;
}
function claimNewOwner() external {
    require(msg.sender == proposedEthFundDeposit, ...);
    ethFundDeposit = proposedEthFundDeposit;
    proposedEthFundDeposit = address(0);
}
```





Static Analysis Results

INSECURE_COMPILER_VERSION

Line 5 in File DAPPT.sol

5 pragma solidity ^0.4.12;

• Version to compile has the following bug: 0.4.12: UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ExpExponentCleanup, NestedArrayFunctionCallDecoder, ZeroFunctionSelector, DelegateCallReturnValue, ECRecover-MalformedInput 0.4.13: UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ExpExponentCleanup, NestedArrayFunctionCallDecoder, Zero-FunctionSelector, DelegateCallReturnValue, ECRecoverMalformedInput 0.4.14: UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ExpExponentCleanup, NestedArrayFunctionCallDecoder, ZeroFunctionSelector, Delegate-CallReturnValue 0.4.15: UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ExpExponentCleanup, NestedArrayFunctionCallDecoder, Zero-FunctionSelector 0.4.16: DynamicConstructorArgumentsClippedABIV2, Uninitialized-FunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ExpExponentCleanup, NestedArrayFunctionCallDecoder, ZeroFunctionSelector 0.4.17: Dynamic-ConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ExpExponentCleanup, EventStructWrongData, NestedArrayFunctionCallDecoder, ZeroFunctionSelector 0.4.18: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ExpExponentCleanup, EventStructWrongData, NestedArrayFunctionCallDecoder 0.4.19: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage ExpExponentCleanup, EventStructWrongData, NestedArrayFunctionCallDecoder 0.4.20: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData, NestedArrayFunctionCallDecoder 0.4.21: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructorArgumentsClippedABIV2, UninitializedFunction tor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData, NestedArrayFunctionCallDecoder 0.4.22: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructionPointerInC tor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData, OneOfTwoConstructorsSkipped 0.4.23: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.24: DynamicConstructorArgumentsClipped-ABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrong-Data 0.4.25: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: 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
□ERTIK label location
                        Line 30-34 in File howtoread.sol
                    30
                            /*@CTK FAIL "transferFrom to same address"
                    31
                                @tag assume_completion
                    32
     \Box \mathsf{ERTIK}\ \mathit{label}
                                @pre from == to
                    33
                                @post __post.allowed[from][msg.sender] ==
                    34
    Raw code location
                        Line 35-41 in File howtoread.sol
                            function transferFrom(address from, address to
                    35
                    36
                                balances[from] = balances[from].sub(tokens
                    37
                                allowed[from][msg.sender] = allowed[from][
          Raw\ code
                    38
                                balances[to] = balances[to].add(tokens);
                    39
                                emit Transfer(from, to, tokens);
                    40
                                return true;
                    41
     Counter example \\
                         This code violates the specification
                     1
                        Counter Example:
                     2
                        Before Execution:
                     3
                            Input = {
                                from = 0x0
                     4
                     5
                                to = 0x0
                     6
                                tokens = 0x6c
                     7
                            This = 0
  Initial environment
                                    balance: 0x0
                    54
                    55
                    56
                    57
                        After Execution:
                    58
                            Input = {
                                from = 0x0
                    59
    Post environment
                    60
                                to = 0x0
                    61
                                tokens = 0x6c
```





Method will not encounter an assertion failure.

```
6 06, Aug 2019
19.1 ms
```

Line 20 in File DAPPT.sol

```
20 //@CTK FAIL NO_ASF
```

Line 28-32 in File DAPPT.sol

```
28    function safeAdd(uint256 x, uint256 y) internal returns(uint256) {
29      uint256 z = x + y;
30      assert((z >= x) && (z >= y));
31      return z;
32    }
```

This code violates the specification.

```
1
   Counter Example:
 2
   Before Execution:
 3
       Input = {
 4
           x = 137
           y = 135
 5
 6
 7
       This = 0
       Internal = {
 8
 9
           __has_assertion_failure = false
10
           __has_buf_overflow = false
           __has_overflow = false
11
           __has_returned = false
12
           __reverted = false
13
           msg = {
14
15
             "gas": 0,
             "sender": 0,
16
             "value": 0
17
18
19
20
       Other = {
21
           __return = 0
22
           block = {
23
             "number": 0,
24
             "timestamp": 0
25
26
27
       Address_Map = [
28
29
           "key": "ALL_OTHERS",
            "value": {
30
31
             "contract_name": "SafeMath",
32
             "balance": 0,
33
             "contract": {}
34
35
36
       ]
37
38
   Function invocation is reverted.
```





SafeMath add

```
6 06, Aug 20197 3.44 ms
```

Line 21-27 in File DAPPT.sol

```
21     /*@CTK "SafeMath add"
22     @post (x + y < x || x + y < y) == __reverted
23     @post !__reverted -> __return == x + y
24     @post !__reverted -> !__has_overflow
25     @post !__reverted -> !(__has_assertion_failure)
26     @post !(__has_buf_overflow)
27     */
```

Line 28-32 in File DAPPT.sol

The code meets the specification.

Formal Verification Request 3

Method will not encounter an assertion failure.

```
6, Aug 2019
```

<u> 16.85 ms</u>

Line 34 in File DAPPT.sol

```
//@CTK FAIL NO_ASF
```

Line 42-46 in File DAPPT.sol

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

This code violates the specification.

```
Counter Example:
1
2
   Before Execution:
3
       Input = {
4
           x = 0
           y = 1
5
6
7
       This = 0
8
       Internal = {
           __has_assertion_failure = false
9
10
           __has_buf_overflow = false
11
           __has_overflow = false
```





```
__has_returned = false
12
13
           __reverted = false
14
           msg = {
15
             "gas": 0,
             "sender": 0,
16
             "value": 0
17
18
19
20
       Other = {
           __return = 0
21
22
           block = {
23
             "number": 0,
24
             "timestamp": 0
25
26
27
       Address_Map = [
28
           "key": "ALL_OTHERS",
29
           "value": {
30
             "contract_name": "SafeMath",
31
32
             "balance": 0,
             "contract": {}
33
34
35
36
       ]
37
38
   Function invocation is reverted.
```

SafeMath sub

06, Aug 2019

 \odot 2.02 ms

Line 35-41 in File DAPPT.sol

```
35    /*@CTK "SafeMath sub"
36    @post (x < y) == __reverted
37    @post !__reverted -> __return == x - y
38    @post !__reverted -> !__has_overflow
39    @post !__reverted -> !(__has_assertion_failure)
40    @post !(__has_buf_overflow)
41    */
```

Line 42-46 in File DAPPT.sol

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





Method will not encounter an assertion failure.

```
## 06, Aug 2019

• 27.76 ms
```

Line 48 in File DAPPT.sol

```
//@CTK FAIL NO_ASF
```

Line 56-60 in File DAPPT.sol

```
56     function safeMult(uint256 x, uint256 y) internal returns(uint256) {
57         uint256 z = x * y;
58         assert((x == 0)||(z/x == y));
59         return z;
60     }
```

This code violates the specification.

```
1
   Counter Example:
 2
   Before Execution:
 3
       Input = {
 4
           x = 11
           y = 159
 5
 6
 7
       This = 0
       Internal = {
 8
 9
           __has_assertion_failure = false
10
           __has_buf_overflow = false
           __has_overflow = false
11
           __has_returned = false
12
           __reverted = false
13
           msg = {
14
15
             "gas": 0,
             "sender": 0,
16
             "value": 0
17
18
19
20
       Other = {
21
           __return = 0
22
           block = {
23
             "number": 0,
24
             "timestamp": 0
25
26
27
       Address_Map = [
28
29
           "key": "ALL_OTHERS",
            "value": {
30
31
             "contract_name": "SafeMath",
32
             "balance": 0,
33
             "contract": {}
34
35
36
       ]
37
38
   Function invocation is reverted.
```





If method completes, integer overflow would not happen.

```
1 06, Aug 2019

○ 5.81 ms
```

Line 130 in File DAPPT.sol

```
130 //@CTK NO_OVERFLOW
```

Line 137-139 in File DAPPT.sol

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

The code meets the specification.

Formal Verification Request 7

Buffer overflow / array index out of bound would never happen.

```
## 06, Aug 2019
```

 \odot 0.37 ms

Line 131 in File DAPPT.sol

```
131 //@CTK NO_BUF_OVERFLOW
```

Line 137-139 in File DAPPT.sol

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

The code meets the specification.

Formal Verification Request 8

Method will not encounter an assertion failure.

```
6, Aug 2019
```

 $\overline{\bullet}$ 0.36 ms

Line 132 in File DAPPT.sol

```
132 //@CTK NO_ASF
```

Line 137-139 in File DAPPT.sol

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





balanceOf

```
6 06, Aug 2019○ 0.39 ms
```

Line 133-136 in File DAPPT.sol

```
/*@CTK balanceOf

dtag assume_completion

post balance == (balances[_owner])

*/
```

Line 137-139 in File DAPPT.sol

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

The code meets the specification.

Formal Verification Request 10

If method completes, integer overflow would not happen.

```
60, Aug 201910.26 ms
```

Line 141 in File DAPPT.sol

```
141 //@CTK NO_OVERFLOW
```

Line 149-153 in File DAPPT.sol

```
function approve(address _spender, uint256 _value) returns (bool success) {
    allowed[msg.sender] [_spender] = _value;
    Approval(msg.sender, _spender, _value);
    return true;
}
```

The code meets the specification.

Formal Verification Request 11

Buffer overflow / array index out of bound would never happen.

```
6, Aug 2019
0.4 ms
```

Line 142 in File DAPPT.sol

```
142 //@CTK NO_BUF_OVERFLOW
```

Line 149-153 in File DAPPT.sol





```
function approve(address _spender, uint256 _value) returns (bool success) {
    allowed[msg.sender][_spender] = _value;
    Approval(msg.sender, _spender, _value);
    return true;
}
```

⊘ The code meets the specification.

Formal Verification Request 12

Method will not encounter an assertion failure.

```
1 06, Aug 2019

○ 0.41 ms
```

Line 143 in File DAPPT.sol

```
143 //@CTK NO_ASF
```

Line 149-153 in File DAPPT.sol

```
function approve(address _spender, uint256 _value) returns (bool success) {

allowed[msg.sender] [_spender] = _value;

Approval(msg.sender, _spender, _value);

return true;

}
```

The code meets the specification.

Formal Verification Request 13

```
approve
```

```
6 06, Aug 20191.53 ms
```

Line 144-148 in File DAPPT.sol

Line 149-153 in File DAPPT.sol

```
function approve(address _spender, uint256 _value) returns (bool success) {

allowed[msg.sender] [_spender] = _value;

Approval(msg.sender, _spender, _value);

return true;

}
```





If method completes, integer overflow would not happen.

```
6.3 ms6.3 ms
```

Line 155 in File DAPPT.sol

```
155 //@CTK NO_OVERFLOW
```

Line 163-165 in File DAPPT.sol

The code meets the specification.

Formal Verification Request 15

Buffer overflow / array index out of bound would never happen.

```
## 06, Aug 2019
```

 $\overline{\bullet}$ 0.43 ms

Line 156 in File DAPPT.sol

```
156 //@CTK NO_BUF_OVERFLOW
```

Line 163-165 in File DAPPT.sol

The code meets the specification.

Formal Verification Request 16

Method will not encounter an assertion failure.

```
6 06, Aug 20190 0.48 ms
```

Line 157 in File DAPPT.sol

```
157 //@CTK NO_ASF
```

Line 163-165 in File DAPPT.sol





allowance

```
6, Aug 2019
```

0.41 ms

Line 158-162 in File DAPPT.sol

```
/*@CTK allowance
/*@CTK allowance

@tag assume_completion
@post (__reverted) == (false)
@post (remaining) == (__post.allowed[_owner][_spender])

#/
Line 163-165 in File DAPPT.sol

function allowance(address owner, address spender) constant returns (uint256)
```

The code meets the specification.

Formal Verification Request 18

If method completes, integer overflow would not happen.

```
6 06, Aug 2019○ 27.81 ms
```

Line 242 in File DAPPT.sol

The code meets the specification.

Formal Verification Request 19

Buffer overflow / array index out of bound would never happen.

```
60, Aug 20190.53 ms
```

Line 243 in File DAPPT.sol

```
243 //@CTK NO_BUF_OVERFLOW
```

Line 252-257 in File DAPPT.sol





```
function setTokenExchangeRate(uint256 _tokenExchangeRate) isOwner external {
   if (_tokenExchangeRate == 0) throw;
   if (_tokenExchangeRate == tokenExchangeRate) throw;
   tokenExchangeRate = _tokenExchangeRate;
}
```

The code meets the specification.

Formal Verification Request 20

Method will not encounter an assertion failure.

```
60, Aug 20190.52 ms
```

Line 244 in File DAPPT.sol

```
Line 252-257 in File DAPPT.sol

function setTokenExchangeRate(uint256 _tokenExchangeRate) isOwner external {
   if (_tokenExchangeRate == 0) throw;
   if (_tokenExchangeRate == tokenExchangeRate) throw;

tokenExchangeRate = _tokenExchangeRate;
}
```

The code meets the specification.

Formal Verification Request 21

setTokenExchangeRate

```
6 06, Aug 2019
4.66 ms
```

Line 245-251 in File DAPPT.sol

Line 252-257 in File DAPPT.sol

```
function setTokenExchangeRate(uint256 _tokenExchangeRate) isOwner external {
    if (_tokenExchangeRate == 0) throw;
    if (_tokenExchangeRate == tokenExchangeRate) throw;
    tokenExchangeRate = _tokenExchangeRate;
}
```





If method completes, integer overflow would not happen.

```
6 06, Aug 2019
38.3 ms
```

Line 295 in File DAPPT.sol

```
295 //@CTK NO_OVERFLOW
```

Line 308-316 in File DAPPT.sol

```
308
        function startFunding (uint256 _fundingStartBlock, uint256 _fundingStopBlock)
            isOwner external {
309
            if (isFunding) throw;
310
            if (_fundingStartBlock >= _fundingStopBlock) throw;
311
            if (block.number >= _fundingStartBlock) throw;
312
313
            fundingStartBlock = _fundingStartBlock;
            fundingStopBlock = _fundingStopBlock;
314
315
            isFunding = true;
316
```

The code meets the specification.

Formal Verification Request 23

Buffer overflow / array index out of bound would never happen.

```
6 06, Aug 20190 0.69 ms
```

Line 296 in File DAPPT.sol

```
296 //@CTK NO_BUF_OVERFLOW
```

Line 308-316 in File DAPPT.sol

```
308
        function startFunding (uint256 _fundingStartBlock, uint256 _fundingStopBlock)
            isOwner external {
309
            if (isFunding) throw;
310
            if (_fundingStartBlock >= _fundingStopBlock) throw;
            if (block.number >= _fundingStartBlock) throw;
311
312
            fundingStartBlock = _fundingStartBlock;
313
314
            fundingStopBlock = _fundingStopBlock;
315
            isFunding = true;
316
```

The code meets the specification.

Formal Verification Request 24

Method will not encounter an assertion failure.

```
## 06, Aug 2019
```

 \bullet 0.63 ms





Line 297 in File DAPPT.sol

```
297
    //@CTK NO_ASF
    Line 308-316 in File DAPPT.sol
308
        function startFunding (uint256 _fundingStartBlock, uint256 _fundingStopBlock)
            isOwner external {
309
           if (isFunding) throw;
310
           if (_fundingStartBlock >= _fundingStopBlock) throw;
311
           if (block.number >= _fundingStartBlock) throw;
312
313
           fundingStartBlock = _fundingStartBlock;
314
           fundingStopBlock = _fundingStopBlock;
315
           isFunding = true;
316
```

The code meets the specification.

Formal Verification Request 25

startFunding

```
6, Aug 2019
23.44 ms
```

Line 298-307 in File DAPPT.sol

```
298
        /*@CTK startFunding
299
          @tag assume_completion
300
          @post msg.sender == ethFundDeposit
301
          @post !isFunding
302
          @post _fundingStartBlock < _fundingStopBlock</pre>
303
          @post block.number < _fundingStartBlock</pre>
304
          @post __post.fundingStartBlock == _fundingStartBlock
          @post __post.fundingStopBlock == _fundingStopBlock
305
306
          @post __post.isFunding
307
```

Line 308-316 in File DAPPT.sol

```
308
        function startFunding (uint256 _fundingStartBlock, uint256 _fundingStopBlock)
            isOwner external {
309
            if (isFunding) throw;
            if (_fundingStartBlock >= _fundingStopBlock) throw;
310
311
            if (block.number >= _fundingStartBlock) throw;
312
313
            fundingStartBlock = _fundingStartBlock;
314
            fundingStopBlock = _fundingStopBlock;
315
            isFunding = true;
316
```





If method completes, integer overflow would not happen.

```
6, Aug 2019
22.31 ms
```

Line 319 in File DAPPT.sol

```
319 //@CTK NO_OVERFLOW
```

Line 328-331 in File DAPPT.sol

```
328  function stopFunding() isOwner external {
329    if (!isFunding) throw;
330    isFunding = false;
331 }
```

✓ The code meets the specification.

Formal Verification Request 27

Buffer overflow / array index out of bound would never happen.

```
6 06, Aug 20196 0.54 ms
```

Line 320 in File DAPPT.sol

```
320 //@CTK NO_BUF_OVERFLOW
```

Line 328-331 in File DAPPT.sol

```
328  function stopFunding() isOwner external {
329    if (!isFunding) throw;
330    isFunding = false;
331 }
```

The code meets the specification.

Formal Verification Request 28

Method will not encounter an assertion failure.

```
60, Aug 20190.61 ms
```

Line 321 in File DAPPT.sol

```
321 //@CTK NO_ASF
```

Line 328-331 in File DAPPT.sol

```
328 function stopFunding() isOwner external {
329     if (!isFunding) throw;
330     isFunding = false;
331 }
```





stopFunding

```
6 06, Aug 20196 3.92 ms
```

Line 322-327 in File DAPPT.sol

```
322  /*@CTK stopFunding
323     @tag assume_completion
324     @post msg.sender == ethFundDeposit
325     @post isFunding
326     @post !__post.isFunding
327     */
```

Line 328-331 in File DAPPT.sol

```
328  function stopFunding() isOwner external {
329    if (!isFunding) throw;
330    isFunding = false;
331 }
```

The code meets the specification.

Formal Verification Request 30

If method completes, integer overflow would not happen.

```
6 06, Aug 2019○ 22.09 ms
```

Line 334 in File DAPPT.sol

```
334 //@CTK NO_OVERFLOW
```

Line 343-346 in File DAPPT.sol

```
function setMigrateContract(address _newContractAddr) isOwner external {
   if (_newContractAddr == newContractAddr) throw;
   newContractAddr = _newContractAddr;
}
```

The code meets the specification.

Formal Verification Request 31

Buffer overflow / array index out of bound would never happen.

```
6, Aug 2019
0.59 ms
```

Line 335 in File DAPPT.sol

```
335 //@CTK NO_BUF_OVERFLOW
```

Line 343-346 in File DAPPT.sol





```
343  function setMigrateContract(address _newContractAddr) isOwner external {
344    if (_newContractAddr == newContractAddr) throw;
345    newContractAddr = _newContractAddr;
346  }
```

The code meets the specification.

Formal Verification Request 32

Method will not encounter an assertion failure.

```
6 06, Aug 20190 0.6 ms
```

Line 336 in File DAPPT.sol

```
JOSTE NO_ASF

Line 343-346 in File DAPPT.sol

function setMigrateContract(address _newContractAddr) isOwner external {
   if (_newContractAddr == newContractAddr) throw;
   newContractAddr = _newContractAddr;
```

The code meets the specification.

Formal Verification Request 33

setMigrateContract

```
1 06, Aug 2019

○ 3.46 ms
```

346

Line 337-342 in File DAPPT.sol

```
/*@CTK setMigrateContract
338     @tag assume_completion
339     @post msg.sender == ethFundDeposit
340     @post _newContractAddr != newContractAddr
341     @post __post.newContractAddr == _newContractAddr
342     */
```

Line 343-346 in File DAPPT.sol

```
function setMigrateContract(address _newContractAddr) isOwner external {
if (_newContractAddr == newContractAddr) throw;
newContractAddr = _newContractAddr;
}
```





If method completes, integer overflow would not happen.

```
6, Aug 2019
20.98 ms
```

Line 349 in File DAPPT.sol

```
Joint 349 //@CTK NO_OVERFLOW
Line 358-361 in File DAPPT.sol

function changeOwner(address _newFundDeposit) isOwner() external {
   if (_newFundDeposit == address(0x0)) throw;
   ethFundDeposit = _newFundDeposit;
}
```

The code meets the specification.

Formal Verification Request 35

Buffer overflow / array index out of bound would never happen.

```
60, Aug 20190.53 ms
```

Line 350 in File DAPPT.sol

```
Joseph Court No_BUF_OVERFLOW

Line 358-361 in File DAPPT.sol

function changeOwner(address _newFundDeposit) isOwner() external {
    if (_newFundDeposit == address(0x0)) throw;
    ethFundDeposit = _newFundDeposit;
}
```

The code meets the specification.

Formal Verification Request 36

Method will not encounter an assertion failure.

```
60, Aug 20190.5 ms
```

Line 351 in File DAPPT.sol

```
//@CTK NO_ASF
Line 358-361 in File DAPPT.sol

function changeOwner(address _newFundDeposit) isOwner() external {
    if (_newFundDeposit == address(0x0)) throw;
    ethFundDeposit = _newFundDeposit;
}
```





change Owner

```
6, Aug 2019
3.67 ms
```

Line 352-357 in File DAPPT.sol

```
/*@CTK changeOwner

@tag assume_completion

@post msg.sender == ethFundDeposit

@post _newFundDeposit != address(0)

@post __post.ethFundDeposit == _newFundDeposit

*/

Line 358-361 in File DAPPT.sol
```

```
358  function changeOwner(address _newFundDeposit) isOwner() external {
359     if (_newFundDeposit == address(0x0)) throw;
360     ethFundDeposit = _newFundDeposit;
361 }
```





Source Code with CertiK Labels

File DAPPT.sol

```
1 /**
 2
   *Submitted for verification at Etherscan.io on 2019-03-08
 3 */
 4
 5 pragma solidity ^0.4.12;
 6
 7
   contract IMigrationContract {
       function migrate(address addr, uint256 dappt) returns (bool success);
 8
 9
10
   /* taking ideas from FirstBlood token */
11
12 contract SafeMath {
13
14
       /* function assert(bool assertion) internal { */
       /* if (!assertion) { */
15
16
            throw; */
17
       /* } */
       /* }
18
               // assert no longer needed once solidity is on 0.4.10 */
19
20
       //@CTK FAIL NO_ASF
21
       /*@CTK "SafeMath add"
22
         Opost (x + y < x \mid \mid x + y < y) == \_reverted
         @post !__reverted -> __return == x + y
23
24
         @post !__reverted -> !__has_overflow
25
         @post !__reverted -> !(__has_assertion_failure)
26
         @post !(__has_buf_overflow)
27
        */
28
       function safeAdd(uint256 x, uint256 y) internal returns(uint256) {
29
         uint256 z = x + y;
         assert((z \ge x) \&\& (z \ge y));
30
31
         return z;
32
33
34
       //@CTK FAIL NO_ASF
35
       /*@CTK "SafeMath sub"
36
         Qpost (x < y) == \_reverted
37
         @post !__reverted -> __return == x - y
         @post !__reverted -> !__has_overflow
38
         @post !__reverted -> !(__has_assertion_failure)
39
40
         @post !(__has_buf_overflow)
41
42
       function safeSubtract(uint256 x, uint256 y) internal returns(uint256) {
43
         assert(x >= y);
44
         uint256 z = x - y;
45
         return z;
46
47
48
       //@CTK FAIL NO_ASF
49
       /*@CTK "SafeMath mul"
         Opost ((x == 0) \&\& (((x * y) / x) != y)) == (\_reverted)
50
         @post !__reverted -> __return == x * y
51
         @post !__reverted == !__has_overflow
52
53
         @post !__reverted -> !(__has_assertion_failure)
        @post !(__has_buf_overflow)
```





```
55
        function safeMult(uint256 x, uint256 y) internal returns(uint256) {
56
57
          uint256 z = x * y;
          assert((x == 0)||(z/x == y));
 58
59
          return z;
        }
 60
 61
62
    }
63
    contract Token {
64
 65
        uint256 public totalSupply;
        function balanceOf(address _owner) constant returns (uint256 balance);
 66
        function transfer(address _to, uint256 _value) returns (bool success);
 67
        function transferFrom(address _from, address _to, uint256 _value) returns (bool
 68
            success);
 69
        function approve(address _spender, uint256 _value) returns (bool success);
 70
        function allowance(address _owner, address _spender) constant returns (uint256
71
        event Transfer(address indexed _from, address indexed _to, uint256 _value);
72
        event Approval(address indexed _owner, address indexed _spender, uint256 _value);
73
    }
74
75
76
    /* ERC 20 token */
77
    contract StandardToken is Token {
78
79
        //@CTK NO_OVERFLOW
80
        //@CTK NO_BUF_OVERFLOW
        //@CTK NO_ASF
81
 82
        /*@CTK transfer
          @tag assume_completion
 83
 84
          @post _to != address(0)
          @post (balances[msg.sender] >= _value && _value > 0) -> success == true
 85
 86
          @post (balances[msg.sender] >= _value && _value > 0 && msg.sender != _to) -> (
              __post.balances[_to] == balances[_to] + _value)
          @post (balances[msg.sender] >= _value && _value > 0 && msg.sender != _to) -> (
87
              __post.balances[msg.sender] == balances[msg.sender] - _value)
          @post (balances[msg.sender] < _value || _value == 0) -> success == false
 88
          @post (balances[msg.sender] < _value || _value == 0 || msg.sender == _to) ->
 89
              __post.balances[_to] == balances[_to]
 90
          @post (balances[msg.sender] < _value || _value == 0 || msg.sender == _to) ->
              __post.balances[msg.sender] == balances[msg.sender]
91
        function transfer(address _to, uint256 _value) returns (bool success) {
 92
 93
          if (balances[msg.sender] >= _value && _value > 0) {
            balances[msg.sender] -= _value;
 94
95
            balances[_to] += _value;
96
            Transfer(msg.sender, _to, _value);
97
           return true;
          } else {
 98
99
            return false;
100
        }
101
102
103
        //@CTK NO_OVERFLOW
104
        //@CTK NO_BUF_OVERFLOW
105
        //@CTK NO_ASF
106
        /*@CTK transferFrom
```





```
107
          @tag assume_completion
108
          @pre _to != address(0)
109
          @post (balances[_from] >= _value && allowed[_from][msg.sender] >= _value &&
              _value > 0) -> success == true
110
          @post (balances[_from] >= _value && allowed[_from][msg.sender] >= _value &&
              _value > 0) -> (__post.allowed[_from][msg.sender] == allowed[_from][msg.
          @post (balances[_from] >= _value && allowed[_from][msg.sender] >= _value &&
111
              _value > 0 && _from != _to) -> (__post.balances[_from] == balances[_from] -
              _value)
          @post (balances[_from] >= _value && allowed[_from] [msg.sender] >= _value &&
112
              _value > 0 && _from != _to) -> (__post.balances[_to] == balances[_to] +
              _value)
113
          @post (balances[_from] < _value || allowed[_from] [msg.sender] < _value || _value</pre>
               == 0) -> success == false
114
          @post (balances[_from] < _value || allowed[_from] [msg.sender] < _value || _value</pre>
               == 0) -> (__post.allowed[_from] [msg.sender] == allowed[_from] [msg.sender])
          @post (balances[_from] < _value || allowed[_from] [msg.sender] < _value || _value</pre>
115
               == 0 || _from == _to) -> __post.balances[_to] == balances[_to]
          @post (balances[_from] < _value || allowed[_from] [msg.sender] < _value || _value</pre>
116
               == 0 || _from == _to) -> __post.balances[_from] == balances[_from]
117
118
        function transferFrom(address _from, address _to, uint256 _value) returns (bool
            success) {
119
          if (balances[_from] >= _value && allowed[_from][msg.sender] >= _value && _value
              > 0) {
120
            balances[_to] += _value;
            balances[_from] -= _value;
121
122
            allowed[_from][msg.sender] -= _value;
123
            Transfer(_from, _to, _value);
124
            return true;
125
          } else {
126
            return false;
127
          }
128
        }
129
130
        //@CTK NO_OVERFLOW
131
        //@CTK NO_BUF_OVERFLOW
132
        //@CTK NO_ASF
133
        /*@CTK balanceOf
134
          @tag assume_completion
135
          @post balance == (balances[_owner])
136
137
        function balanceOf(address _owner) constant returns (uint256 balance) {
138
            return balances[_owner];
139
140
141
        //@CTK NO_OVERFLOW
142
        //@CTK NO_BUF_OVERFLOW
143
        //@CTK NO_ASF
144
        /*@CTK approve
145
          @tag assume_completion
          @post (__post.allowed[msg.sender][_spender]) == (_value)
146
147
          @post (success) == (true)
148
         */
149
        function approve(address _spender, uint256 _value) returns (bool success) {
150
            allowed[msg.sender][_spender] = _value;
151
            Approval(msg.sender, _spender, _value);
```





```
152
           return true;
153
        }
154
155
        //@CTK NO_OVERFLOW
156
        //@CTK NO_BUF_OVERFLOW
        //@CTK NO_ASF
157
158
        /*@CTK allowance
159
          @tag assume_completion
160
          @post (__reverted) == (false)
161
          @post (remaining) == (__post.allowed[_owner][_spender])
162
        function allowance (address _owner, address _spender) constant returns (uint256
163
            remaining) {
164
          return allowed[_owner][_spender];
165
166
167
        mapping (address => uint256) balances;
168
        mapping (address => mapping (address => uint256)) allowed;
169 }
170
171
    contract DappToken is StandardToken, SafeMath {
172
173
        // metadata
174
        string public constant name = "Dapp Token";
175
        string public constant symbol = "DAPPT";
176
        uint256 public constant decimals = 18;
177
        string public version = "1.0";
178
179
        // contracts
180
        address public ethFundDeposit;
                                            // deposit address for ETH for Dapp Team.
181
        address public newContractAddr;
                                             // the new contract for dapp token updates;
182
183
        // crowdsale parameters
184
              public isFunding;
                                             // switched to true in operational state
        uint256 public fundingStartBlock;
185
186
        uint256 public fundingStopBlock;
187
        uint256 public currentSupply;
188
                                             // current supply tokens for sell
        uint256 public tokenRaised = 0;
189
                                           // the number of total sold token
190
        uint256 public tokenMigrated = 0; // the number of total transferted token
191
        uint256 public tokenExchangeRate = 25000;
                                                    // 25000 Dapp tokens per 1 ETH
192
193
        // events
194
        event AllocateToken(address indexed _to, uint256 _value); // allocate token for
            private sale;
        event IssueToken(address indexed _to, uint256 _value); // issue token for public
195
            sale:
196
        event IncreaseSupply(uint256 _value);
197
        event DecreaseSupply(uint256 _value);
198
        event Migrate(address indexed _to, uint256 _value);
199
200
        // format decimals.
201
        //@CTK NO_OVERFLOW
202
        //@CTK NO_BUF_OVERFLOW
203
        //@CTK NO_ASF
204
        /*@CTK formatDecimals
205
          @tag assume_completion
206
          @post __return == _value * 100000000000000000
```





```
207
208
        function formatDecimals(uint256 _value) internal returns (uint256 ) {
209
           return _value * 10 ** decimals;
210
211
212
        // constructor
213
        //@CTK NO_OVERFLOW
214
        //@CTK NO_BUF_OVERFLOW
215
        //@CTK NO_ASF
216
        /*@CTK DappToken
217
         @post currentSupply <= totalSupply</pre>
         @post __post.ethFundDeposit == _ethFundDeposit
218
219
         @post __post.isFunding == false
         @post __post.fundingStartBlock == 0
220
221
         @post __post.fundingStopBlock == 0
222
         223
         @post __post.currentSupply == 5000000000 * 1000000000000000
224
         @post __post.balances[msg.sender] == __post.currentSupply
225
        function DappToken(address _ethFundDeposit, uint256 _currentSupply) {
226
227
           ethFundDeposit = _ethFundDeposit;
228
229
           isFunding = false;
                                                    //controls pre through crowdsale state
230
           fundingStartBlock = 0;
231
           fundingStopBlock = 0;
232
233
           currentSupply = formatDecimals(_currentSupply);
234
           totalSupply = formatDecimals(5000000000);
235
         balances[msg.sender] = totalSupply;
236
           if(currentSupply > totalSupply) throw;
237
238
239
        modifier isOwner() { require(msg.sender == ethFundDeposit); _; }
240
        /// @dev set the token's tokenExchangeRate,
241
242
        //@CTK NO_OVERFLOW
243
        //@CTK NO_BUF_OVERFLOW
244
        //@CTK NO_ASF
245
        /*@CTK setTokenExchangeRate
246
         @tag assume_completion
         @post msg.sender == ethFundDeposit
247
248
         @post _tokenExchangeRate != 0
249
         @post _tokenExchangeRate != tokenExchangeRate
250
         @post __post.tokenExchangeRate == _tokenExchangeRate
251
252
        function setTokenExchangeRate(uint256 _tokenExchangeRate) isOwner external {
253
           if (_tokenExchangeRate == 0) throw;
254
           if (_tokenExchangeRate == tokenExchangeRate) throw;
255
256
           tokenExchangeRate = _tokenExchangeRate;
257
        }
258
259
        /// @dev increase the token's supply
260
        //@CTK NO_OVERFLOW
261
        //@CTK NO_BUF_OVERFLOW
262
        //@CTK NO_ASF
263
        /*@CTK increaseSupply
264
        @tag assume_completion
```





```
265
          @post msg.sender == ethFundDeposit
266
          @post currentSupply + (_value * 10000000000000000) <= totalSupply</pre>
267
          @post __post.currentSupply == currentSupply + (_value * 10000000000000000000)
268
269
        function increaseSupply (uint256 _value) isOwner external {
270
            uint256 value = formatDecimals(_value);
271
            if (value + currentSupply > totalSupply) throw;
272
            currentSupply = safeAdd(currentSupply, value);
273
            IncreaseSupply(value);
274
        }
275
        /// @dev decrease the token's supply
276
277
        //@CTK NO_OVERFLOW
        //@CTK NO_BUF_OVERFLOW
278
279
        //@CTK NO_ASF
280
        /*@CTK decreaseSupply
281
          @tag assume_completion
282
          @post msg.sender == ethFundDeposit
283
          @post (_value * 100000000000000000) + tokenRaised <= currentSupply</pre>
          @post __post.currentSupply == currentSupply - (_value * 10000000000000000000)
284
285
286
        function decreaseSupply (uint256 _value) isOwner external {
287
            uint256 value = formatDecimals(_value);
288
            if (value + tokenRaised > currentSupply) throw;
289
290
            currentSupply = safeSubtract(currentSupply, value);
291
            DecreaseSupply(value);
292
        }
293
294
        /// @dev turn on the funding state
295
        //@CTK NO_OVERFLOW
296
        //@CTK NO_BUF_OVERFLOW
297
        //@CTK NO_ASF
298
        /*@CTK startFunding
          @tag assume_completion
299
300
          @post msg.sender == ethFundDeposit
301
          @post !isFunding
          @post _fundingStartBlock < _fundingStopBlock</pre>
302
303
          @post block.number < _fundingStartBlock</pre>
304
          @post __post.fundingStartBlock == _fundingStartBlock
305
          @post __post.fundingStopBlock == _fundingStopBlock
306
          @post __post.isFunding
307
308
        function startFunding (uint256 _fundingStartBlock, uint256 _fundingStopBlock)
            isOwner external {
309
            if (isFunding) throw;
            if (_fundingStartBlock >= _fundingStopBlock) throw;
310
311
            if (block.number >= _fundingStartBlock) throw;
312
313
            fundingStartBlock = _fundingStartBlock;
            fundingStopBlock = _fundingStopBlock;
314
315
            isFunding = true;
        }
316
317
318
        /// @dev turn off the funding state
319
        //@CTK NO_OVERFLOW
320
        //@CTK NO_BUF_OVERFLOW
321
       //@CTK NO_ASF
```





```
322
       /*@CTK stopFunding
323
          @tag assume_completion
324
          @post msg.sender == ethFundDeposit
325
          @post isFunding
326
          @post !__post.isFunding
327
328
        function stopFunding() isOwner external {
329
            if (!isFunding) throw;
330
            isFunding = false;
331
        }
332
333
        /// @dev set a new contract for recieve the tokens (for update contract)
334
        //@CTK NO_OVERFLOW
        //@CTK NO_BUF_OVERFLOW
335
336
        //@CTK NO_ASF
337
        /*@CTK setMigrateContract
338
          @tag assume_completion
339
          @post msg.sender == ethFundDeposit
340
          @post _newContractAddr != newContractAddr
341
          @post __post.newContractAddr == _newContractAddr
342
343
        function setMigrateContract(address _newContractAddr) isOwner external {
344
            if (_newContractAddr == newContractAddr) throw;
345
            newContractAddr = _newContractAddr;
346
        }
347
        /// @dev set a new owner.
348
349
        //@CTK NO_OVERFLOW
350
        //@CTK NO_BUF_OVERFLOW
351
        //@CTK NO_ASF
352
        /*@CTK changeOwner
353
          @tag assume_completion
354
          @post msg.sender == ethFundDeposit
355
          @post _newFundDeposit != address(0)
          @post __post.ethFundDeposit == _newFundDeposit
356
357
         */
358
        function changeOwner(address _newFundDeposit) isOwner() external {
359
            if (_newFundDeposit == address(0x0)) throw;
360
            ethFundDeposit = _newFundDeposit;
361
        }
362
363
        /// sends the tokens to new contract
364
        //@CTK NO_OVERFLOW
365
        //@CTK NO_BUF_OVERFLOW
366
        //@CTK NO_ASF
        /*@CTK migrate
367
368
          @tag assume_completion
369
          @post !isFunding
370
          @post newContractAddr != address(0)
371
          @post balances[msg.sender] > 0
372
          @post __post.balances[msg.sender] == 0
373
374
        function migrate() external {
375
            if(isFunding) throw;
376
            if(newContractAddr == address(0x0)) throw;
377
378
            uint256 tokens = balances[msg.sender];
379
            if (tokens == 0) throw;
```





```
380
381
           balances[msg.sender] = 0;
382
383
           IMigrationContract newContract = IMigrationContract(newContractAddr);
384
           if (!newContract.migrate(msg.sender, tokens)) throw;
385
           Migrate(msg.sender, tokens);
                                             // log it
386
387
       }
388
389
       /// @dev sends ETH to Dapp team
390
       //@CTK NO_OVERFLOW
391
       //@CTK NO_BUF_OVERFLOW
392
       //@CTK NO_ASF
393
       /*@CTK transferETH
394
         @tag assume_completion
395
         @post msg.sender == ethFundDeposit
396
       function transferETH() isOwner external {
397
398
           if (this.balance == 0) throw;
399
           if (!ethFundDeposit.send(this.balance)) throw;
400
       }
401
402
       /// @dev allocates Dapp tokens to pre-sell address.
403
       //@CTK NO_OVERFLOW
404
       //@CTK NO_BUF_OVERFLOW
405
       //@CTK NO_ASF
406
       /*@CTK allocateToken
407
         @tag assume_completion
408
         @post msg.sender == ethFundDeposit
         @post _eth > 0
409
410
         @post _addr != address(0)
411
         @post (_eth * 1000000000000000000 * tokenExchangeRate) + tokenRaised <=</pre>
             currentSupply
412
         tokenExchangeRate)
413
         tokenExchangeRate)
414
       function allocateToken (address _addr, uint256 _eth) isOwner external {
415
416
           if (_eth == 0) throw;
417
           if (_addr == address(0x0)) throw;
418
419
           uint256 tokens = safeMult(formatDecimals(_eth), tokenExchangeRate);
420
           if (tokens + tokenRaised > currentSupply) throw;
421
422
           tokenRaised = safeAdd(tokenRaised, tokens);
           balances[_addr] += tokens;
423
424
425
           AllocateToken(_addr, tokens); // logs token issued
426
       }
427
428
       /// buys the tokens
429
       //@CTK NO_OVERFLOW
430
       //@CTK NO_BUF_OVERFLOW
431
       //@CTK NO_ASF
432
       /*@CTK fallback
433
         @tag assume_completion
434
         @post isFunding
```





```
435
          @post msg.value > 0
436
          @post block.number >= fundingStartBlock
          @post block.number <= fundingStopBlock</pre>
437
438
          @post (msg.value * tokenExchangeRate) + tokenRaised <= currentSupply</pre>
          @post __post.tokenRaised == tokenRaised + (msg.value * tokenExchangeRate)
439
440
          @post __post.balances[msg.sender] == balances[msg.sender] + (msg.value *
              tokenExchangeRate)
441
442
        function () payable {
443
            if (!isFunding) throw;
444
            if (msg.value == 0) throw;
445
446
            if (block.number < fundingStartBlock) throw;</pre>
            if (block.number > fundingStopBlock) throw;
447
448
449
            uint256 tokens = safeMult(msg.value, tokenExchangeRate);
450
            if (tokens + tokenRaised > currentSupply) throw;
451
452
            tokenRaised = safeAdd(tokenRaised, tokens);
453
            balances[msg.sender] += tokens;
454
            IssueToken(msg.sender, tokens); // logs token issued
455
        }
456
457 }
```