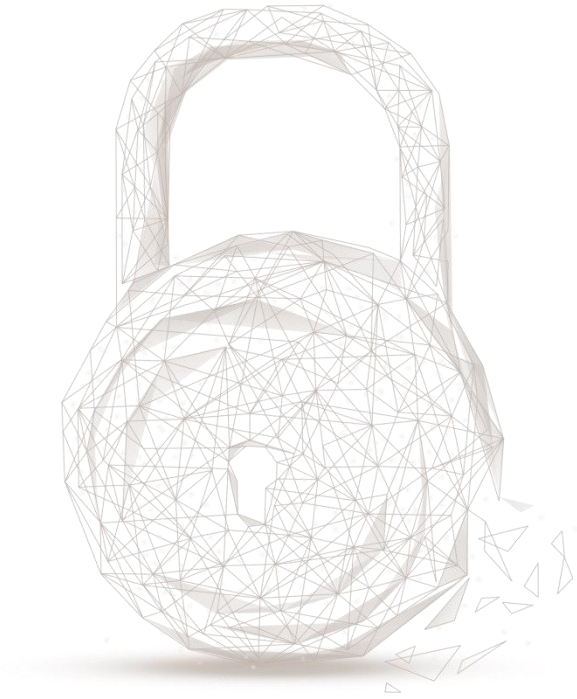
Beosin

成都链安

BEOSIN

Smart contract security audit report



Audit Number：202008191917

Smart Contract Name：

CocosGateway

Smart Contract Address Link：

https://github.com/dcocos-finance/dCocosToken.git

Commit Hash：

0688ff7462bdd72499467eb5fcfbf54102b87c0c

Start Date：2020.08.13

Completion Date：2020.08.19

Overall Result：Pass

Audit Team: Beosin (Chengdu LianAn) Technology Co. Ltd.

Audit Categories and Results:

No.	Categories	Subitems	Results
1	Coding Conventions	Compiler Version Security	Pass
		Deprecated Items	Pass
		Redundant Code	Pass
		SafeMath Features	Pass
		require/assert Usage	Pass
		Gas Consumption	Pass
		Visibility Specifiers	Pass
		Fallback Usage	Pass
2	General Vulnerability	Integer Overflow/Underflow	Pass
		Reentrancy	Pass
		Pseudo-random Number Generator (PRNG)	Pass
		Transaction-Ordering Dependence	Pass
		DoS (Denial of Service)	Pass
		Access Control of Owner	Pass

		Low-level Function (call/delegatecall) Security	Pass
		Returned Value Security	Pass
		tx.origin Usage	Pass
		Replay Attack	Pass
		Overriding Variables	Pass
3	Business Security	Business Logics	Pass
		Business Implementations	Pass

Note: Audit results and suggestions in code comments

Disclaimer: This audit is only applied to the type of auditing specified in this report and the scope of given in the results table. Other unknown security vulnerabilities are beyond auditing responsibility. Beosin (Chengdu LianAn) Technology only issues this report based on the attacks or vulnerabilities that already existed or occurred before the issuance of this report. For the emergence of new attacks or vulnerabilities that exist or occur in the future, Beosin (Chengdu LianAn) Technology lacks the capability to judge its possible impact on the security status of smart contracts, thus taking no responsibility for them. The security audit analysis and other contents of this report are based solely on the documents and materials that the contract provider has provided to Beosin (Chengdu LianAn) Technology before the issuance of this report, and the contract provider warrants that there are no missing, tampered, deleted; if the documents and materials provided by the contract provider are missing, tampered, deleted, concealed or reflected in a situation that is inconsistent with the actual situation, or if the documents and materials provided are changed after the issuance of this report, Beosin (Chengdu LianAn) Technology assumes no responsibility for the resulting loss or adverse effects. The audit report issued by Beosin (Chengdu LianAn) Technology is based on the documents and materials provided by the contract provider, and relies on the technology currently possessed by Beosin (Chengdu LianAn). Due to the technical limitations of any organization, this report conducted by Beosin (Chengdu LianAn) still has the possibility that the entire risk cannot be completely detected. Beosin (Chengdu LianAn) disclaims any liability for the resulting losses.

The final interpretation of this statement belongs to Beosin (Chengdu LianAn).

Audit Results Explained:

Beosin (Chengdu LianAn) Technology has used several methods including Formal Verification, Static Analysis, Typical Case Testing and Manual Review to audit three major aspects of smart contracts CocosGateway, including Coding Standards, Security, and Business Logic. **The CocosGateway contract passed all audit items. The overall result is Pass. The smart contract is able to function properly.**

1. Coding Conventions

Check the code style that does not conform to Solidity code style.

1.1 Compiler Version Security

- Description: Check whether the code implementation of current contract contains the exposed solidity compiler bug.
- Result: Pass

1.2 Deprecated Items

- Description: Check whether the current contract has the deprecated items.
- Result: Pass

1.3 Redundant Code

- Description: Check whether the contract code has redundant codes.
- Result: Pass

1.4 SafeMath Features

- Description: Check whether the SafeMath has been used. Or prevents the integer overflow/underflow in mathematical operation.
- Result: Pass

1.5 require/assert Usage

- Description: Check the use reasonability of 'require' and 'assert' in the contract.
- Result: Pass

1.6 Gas Consumption

- Description: Check whether the gas consumption exceeds the block gas limitation.
- Result: Pass

1.7 Visibility Specifiers

- Description: Check whether the visibility conforms to design requirement.
- Result: Pass

1.8 Fallback Usage

- Description: Check whether the Fallback function has been used correctly in the current contract.
- Result: Pass

2. General Vulnerability

Check whether the general vulnerabilities exist in the contract.

2.1 Integer Overflow/Underflow

- Description: Check whether there is an integer overflow/underflow in the contract and the calculation result is abnormal.
- Result: Pass

2.2 Reentrancy

- Description: An issue when code can call back into your contract and change state, such as withdrawing ETH.
- Result: Pass

2.3 Pseudo-random Number Generator (PRNG)

- Description: Whether the results of random numbers can be predicted.
- Result: Pass

2.4 Transaction-Ordering Dependence

- Description: Whether the final state of the contract depends on the order of the transactions.
- Result: Pass

2.5 DoS (Denial of Service)

- Description: Whether exist DoS attack in the contract which is vulnerable because of unexpected reason.
- Result: Pass

2.6 Access Control of Owner

- Description: Whether the owner has excessive permissions, such as malicious issue, modifying the balance of others.
- Result: Pass

2.7 Low-level Function (call/delegatecall) Security

- Description: Check whether the usage of low-level functions like call/delegatecall have vulnerabilities.
- Result: Pass

2.8 Returned Value Security

- Description: Check whether the function checks the return value and responds to it accordingly.
- Result: Pass

2.9 tx.origin Usage

- Description: Check the use secure risk of 'tx.origin' in the contract.
- Result: Pass

2.10 Replay Attack

- Description: Check the weather the implement possibility of Replay Attack exists in the contract.
- Result: Pass

2.11 Overriding Variables

- Description: Check whether the variables have been overridden and lead to wrong code execution.
- Result: Pass

3. Business Security

Check whether the business is secure.

3.1 Swap dCOCOS tokens (COCOS -> dCOCOS)

- Description:

  The contract implements the *swapDCOCOS* function to convert COCOS tokens to dCOCOS tokens one-to-one. The user (COCOS token holder) pre-approve the contract address, by calling the *transferFrom* function in the COCOS contract, the contract address as a delegate of the user transfers the specified amount of COCOS tokens to the address of this contract itself, and then by calling the *transfer* function in the dCOCOS contract, the contract address transfers the corresponding amount of dCOCOS tokens to the address of the function caller (user); The function can be paused, that is, the function cannot be successfully called when the contract's global pause status is true.

- Related functions: *swapDCOCOS, transferFrom(COCOS contract), transfer(dCOCOS contract)*
- Result: Pass

3.2 Swap COCOS tokens (dCOCOS -> COCOS)

- Description:

  The contract implements the *swapCOCOS* function to convert dCOCOS tokens to COCOS tokens one-to-one. The user (dCOCOS token holder) pre-approve the contract address, by calling the *transferFrom* function in the dCOCOS contract, the contract address as a delegate of the user transfers the specified amount of dCOCOS tokens to the address of this contract itself, and then by calling the *transfer* function in the COCOS contract, the contract address transfers the corresponding amount of COCOS tokens to the address of the function caller (user); The function can be paused, that is, the function cannot be successfully called when the contract's global pause status is true.

- Related functions: *swapCOCOS, transferFrom(dCOCOS contract), transfer(COCOS contract)*
- Result: Pass

3.3 dCOCOS token mint

- Description:

  The contract implements the internal function *_mintDCOCOS* to call the *mint* function of the dCOCOS contract for minting operations in this contract (there is an upper limit, but the original dCOCOS minting function can mint without cap). The upper limit of the mint set in the contract can be modified through *addDCOCOSSupply* function.

- Related functions: *_mintDCOCOS, mint(dCOCOS contract), addDCOCOSSupply*
- Result: Pass

3.4 Governance setting

- Description:

The contract implements the *setGovernance* function to set the governance administrator address.

- Related functions: *setGovernance*
- Result: Pass

Audited Source Code with Comments:

```
// Beosin (Chengdu LianAn) // File: CocosGateway.sol
/// @title CocosGateway
/// swap COCOS<->dCOCOS 1:1
/// For more information about this token please visit https://dcocos.finance
/// @author reedhong
pragma solidity ^0.5.0; // Beosin (Chengdu LianAn) // Fixing compiler version is recommended.

import '@openzeppelin/contracts/lifecycle/Pausable.sol';
import './IERC20.sol';
import './SafeERC20.sol';

contract CocosGateway is Pausable {
    using SafeERC20 for IERC20; // Beosin (Chengdu LianAn) // Use the SafeERC20 library for safe external ERC20 contract calling.
    using SafeMath for uint256; // Beosin (Chengdu LianAn) // Use the SafeMath library for mathematical operation. Avoid integer overflow/underflow.

    address public governance;

    IERC20 public cocos = IERC20(0x60626db611a9957C1ae4Ac5b7eDE69e24A3B76c5); // Beosin (Chengdu LianAn) // Declare the external COCOS Token contract.
    IERC20 public dcocos = IERC20(0x16cAC1403377978644e78769Daa49d8f6B6CF565); // Beosin (Chengdu LianAn) // Declare the external dCOCOS Token contract.


    uint256 public dcocosTotalSupply = 0; // Beosin (Chengdu LianAn) // Declare the variable 'dcocosTotalSupply' for recording the recorded total token supply of dCOCOS.
    uint256 public dcocosMaxSupply = 2400000000*1e18; // Beosin (Chengdu LianAn) // Declare the variable 'dcocosMaxSupply' for storing the maximum supply of dCOCOS token(the original dCOCOS minting function can mint without cap).

    event MintDCOCOS(uint256 amount); // Beosin (Chengdu LianAn) // Decalre the event 'MintDCOCOS'.
    event SwapDCOCOS(address indexed user, uint256 amount); // Beosin (Chengdu LianAn) // Decalre the event 'SwapDCOCOS'.
    event SwapCOCOS(address indexed user, uint256 amount); // Beosin (Chengdu LianAn) // Decalre the event 'SwapCOCOS'.
    // Beosin (Chengdu LianAn) // Constructor, initialize the contrac pause state and the governance address.
    constructor () public {
        pause();
        governance = tx.origin;
    }
    // Beosin (Chengdu LianAn) // The function 'setGovernance' is defined to set the governance manager address.
    // Beosin (Chengdu LianAn) // Note: Declaring and triggering relevant event in this function is
```

```
recommended.
    function setGovernance(address _governance) public {
        require(msg.sender == governance, "!governance"); // Beosin (Chengdu LianAn) // Require that the caller should be the governance manager address.
        governance = _governance; // Beosin (Chengdu LianAn) // Update the governance manager address.
    }
    // Beosin (Chengdu LianAn) // The function 'addDCOCOSSupply' is defined to increase the maximum supply of dCOCOS token.
    function addDCOCOSSupply(uint256 supply)
        public
    {
        require(msg.sender == governance, "!governance"); // Beosin (Chengdu LianAn) // Require that the caller should be the governance manager address.
        dcocosMaxSupply = dcocosMaxSupply.add(supply); // Beosin (Chengdu LianAn) // Update the 'dcocosMaxSupply'.
    }
    // Beosin (Chengdu LianAn) // The internal function '_mintDCOCOS' is defined to mint tokens to this contract.
    function _mintDCOCOS(uint256 amount) internal{
        uint256 supply = dcocosTotalSupply.add(amount); // Beosin (Chengdu LianAn) // Declare the local variable 'supply' to record the total dCOCOS supply after this minting.
        require(supply <= dcocosMaxSupply, "supply is too large"); // Beosin (Chengdu LianAn) // Require that the 'supply' cannot exceed the 'dcocosMaxSupply'.
        dcocos.mint(address(this),amount); // Beosin (Chengdu LianAn) // Call the function 'mint' of dCOCOS contract to mint tokens to this contract.
        dcocosTotalSupply = supply; // Beosin (Chengdu LianAn) // Update the 'dcocosTotalSupply'.
        emit MintDCOCOS(amount); // Beosin (Chengdu LianAn) // Trigger the event 'MintDCOCOS'.
    }
    // Beosin (Chengdu LianAn) // The function 'swapDCOCOS' is defined to swap COCOS token to dCOCOS token.
    // COCOS->dCOCOS
    function swapDCOCOS(uint256 amount)
        public
        whenNotPaused // Beosin (Chengdu LianAn) // The function can be paused, that is, the function cannot be successfully called when the contract's global pause status is true.
        returns (bool)
    {
        uint256 dcocosBalance =    dcocos.balanceOf(address(this)); // Beosin (Chengdu LianAn) // Declare the local variable 'dcocosBalance' for recording the dCOCOS token balance of this contract.
        if( dcocosBalance < amount){
            _mintDCOCOS(amount.sub(dcocosBalance)); // Beosin (Chengdu LianAn) // Call the internal function '_mintDCOCOS' to mint tokens to thins contract for ensuring the swap balance is sufficient.
        }

        cocos.safeTransferFrom(msg.sender, address(this), amount); // Beosin (Chengdu LianAn) // Call the 'transferFrom' function of specified COCOS Token contract via the function 'safeTransferFrom' in
```

SafeERC20 library, this contract delegate the caller transfers the specified amount of COCOS tokens to this contract.

dcocos.safeTransfer(msg.sender, amount); **// Beosin (Chengdu LianAn) // Call the 'transfer' function of specified dCOCOS Token contract via the function 'safeTransfer' in SafeERC20 library, this contract transfers the specified amount of dCOCOS tokens to the function caller.**

emit SwapDCOCOS(msg.sender, amount); **// Beosin (Chengdu LianAn) // Trigger the event 'SwapDCOCOS'.**

return true;

}

// Beosin (Chengdu LianAn) // The function 'swapDCOCOS' is defined to swap dCOCOS token to COCOS token.

```
// dCOCOS->COCOS
function swapCOCOS(uint256 amount)
    public
    whenNotPaused // Beosin (Chengdu LianAn) // The function can be paused, that is, the function cannot be successfully called when the contract's global pause status is true.
    returns (bool)
{
    uint256 cocosBalance =    cocos.balanceOf(address(this)); // Beosin (Chengdu LianAn) // Declare the local variable 'cocosBalance' for recording the COCOS token balance of this contract.
    require(cocosBalance >= amount, "amount is too large"); // Beosin (Chengdu LianAn) // Require that the swap amount cannot exceed the COCOS token balance of this contract.

    dcocos.safeTransferFrom(msg.sender, address(this), amount); // Beosin (Chengdu LianAn) // Call the 'transferFrom' function of specified dCOCOS Token contract via the function 'safeTransferFrom' in SafeERC20 library, this contract delegate the caller transfers the specified amount of dCOCOS tokens to this contract.

    cocos.safeTransfer(msg.sender, amount); // Beosin (Chengdu LianAn) // Call the 'transfer' function of specified COCOS Token contract via the function 'safeTransfer' in SafeERC20 library, this contract transfers the specified amount of COCOS tokens to the function caller.
    emit SwapCOCOS(msg.sender, amount); // Beosin (Chengdu LianAn) // Trigger the event 'SwapCOCOS'.

    return true;
}

}
```

成都链安

BEOSIN

Official Website

https://lianantech.com

E-mail

vaas@lianantech.com

Twitter

https://twitter.com/Beosin_com