Security Assessment

CPPC

Jun 23rd, 2021

Table of Contents

Summary

This report has been prepared for CPPC smart contracts, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing 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 practices 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.

Additionally, this audit is based on all the calculation models of `CbbcLibrary` are correct.

The security assessment resulted in 14 findings that ranged from minor to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases given they are currently missing in the repository;
- Provide more comments per each function for readability, especially contracts are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	CPPC
Platform	Heco
Language	Solidity
Codebase	https://github.com/yedy99/cbbc
Commit	7bd620efd089f3b3baa694019c6e76052f579d51

Audit Summary

Delivery Date	Jun 23, 2021
Audit Methodology	Manual Review
Key Components	

Vulnerability Summary

Total Issues	14
Critical	0
Major	0
Medium	0
Minor	2
Informational	12
Discussion	0

Audit Scope

ID	file	SHA256 Checksum
CER	CbbcERC20.sol	4dcc17fab6a92cac2be411495ea5bd8bd604bc70c3057841f6367886bb6eaf74
CFC	CbbcFactory.sol	31fb62efa2d963dedfaae20abc19e1c89929fdccac47c16f856844f44b6c373e
CLT	CbbcLiquidityToken.sol	db34f1d91529be94bc82163c88ffea4bb59e58b3e3bdc5e1cc7be3c2ef68e6a1
CRP	CbbcRebasePolicy.sol	a86c4ae80051d3314292367cce4cc92f36b1ec134f7142f7e125273350beef82
CRC	CbbcRouter.sol	71d32abc6e2e9c8586f0f3024cc2e8729d6498a1df63c83c549a1210ac24ecff
CTC	CbbcToken.sol	2d39871565aa22688247e37bf73819eb86911a5a1e52613bb92f8c0e259d9907
MOC	MarketOracle.sol	7f9f38b28115ac36119725280f88a8c313b027de2f4ce83511041a945849f068
MOK	MedianOracle.sol	18e9a43ed23483ae1f07572f44812f14cde22aa5d6dc8f18989a8636ca4dfcd0
OCK	Orchestrator.sol	06e8bce516becb6893cfdb6eb49f0dd34d053d88993e9a1647e12fb0941c3a81
ICC	interfaces/ICbbcCallee.sol	446f5669e35c0712d2fa171747e6a20ff8652c4c759362f99eab91914d6e5566
ICE	interfaces/ICbbcERC20.sol	aca53c9b8f4400f0a1b249b6fb45d117cbb1cc9eb5a21d6f69c2d21ee5a2fcdd
ICF	interfaces/ICbbcFactory.sol	9ee28a075570e15b9c9e26243f88d316301e6597a36fd0f356965230c7a31876
ICL	interfaces/ICbbcLiquidityToken.sol	4e2e35cd9b9cfde832feccb7736bc677821a3a3703e053fa6c57735e933e77ca
ICR	interfaces/ICbbcRebasePolicy.sol	3c6aa014e82d9d9b0456bddf7dccf79190e4051662c80d6c88d074a969e3a4b4
ICK	interfaces/ICbbcRouter.sol	04a4f39c19d6786ac54010d7f6c169f20aecec5897dae403ca4f4643e04ae542
ICT	interfaces/ICbbcToken.sol	6a158ede82c67bffc35254df1f3ac966edb169778bc1e96a1cb3a7c28315cbfb
IER	interfaces/IERC20.sol	d655eb9324a07e889100bc54573b13ee06e8b6c23f481fc1de51ec578a0c0c53
IMO	interfaces/IMarketOracle.sol	b4ff8a711cf099b9e0813ede17d4b5e4bdd296b78d1b295f13d60f87332c52c3
IMC	interfaces/IMedianOracle.sol	89f04d0fdeeb7438dfe525a4cda2c78fcb3b05e517b943f600f408c2cf357a39
IOC	interfaces/IOrchestrator.sol	67120a093e8e0932573ccb67b0c90b62eb4521033375f21c0f0607ae54829b84
IWE	interfaces/IWETH.sol	5f30a9968eaa35efc087c9aac078330667368dca92ad77a8b949e518006fc1bb

ID	file	SHA256 Checksum
CLC	libraries/CbbcLibrary.sol	fecd53b031b772046942d4240ff5733761714d89c7a6462e0977840dc2eabbd3
ECD	libraries/ECDSA.sol	71c06c8eee325b97ec37be3ac7144b5e9d377bdedc6c6b5cf5a68701bf5d11b3
ERC	libraries/ERC20Detailed.sol	cd6a2bf7d2e366a0988d910ba13483a5586e66cfd15d5c45a24c62931255be3e
IKP	libraries/Initializable.sol	536c615ca5964be5ec03ce0f33790b645d10165b18dbbab354af9c8f4db1fee0
MCK	libraries/Math.sol	583c62cbf6081da419a7d0a39d1668f5b6b55a1e5ff90b7825a3bb36ae407b68
OCP	libraries/Ownable.sol	ae1d9c20da7a0e6c8f368e6cadd39a2773a92e3b4372c788cd2bf5d3b07da6a3
SMI	libraries/SafeMathInt.sol	6a613e93d0b22b67305569c713929d4652c16ea643193cf8ca264509442d1e4f
SCK	libraries/Select.sol	673f07656552f4e0651d62b6b882480e9d60512a20c2a563d93b103924cdcda0
THC	libraries/TransferHelper.sol	b9cc3170309b524d05c9e73b5e690accf8c1ba65469803f463e67c3f5c5f68c0
UIL	libraries/UInt256Lib.sol	023327ea95fda72f1a3c144e1bee294b56bfd0378212e42d150105d215d8ab13
WLC	libraries/WhiteList.sol	9e2b4af6b95a0c3e282872a14dafbcdcbed8c4e6db472925f64bc26fcaafc5f6

Privileged Functions

The project contains the following privileged functions that are restricted by the `onlyOwner` modifier. They are used to modify the contract configurations and address attributes. We grouped these functions below:

Contract `CbbcFactory`:

- function setAlpha(uint alpha_buy_, uint alpha_sell_)
- function setLiquidityToken(address settleToken, address liquidityToken)
- function setCbbcToken(address settleToken, address tradeToken, uint8 leverage, ICbbcToken.CbbcType cbbcType, address cbbcToken)
- function removeLiquidityToken(address settleToken, address liquidityToken)
- function removeCbbcToken(address settleToken, address tradeToken, uint8 leverage, ICbbcToken.CbbcType cbbcType, address cbbcToken)
- addSettleToken(address addAddress, string calldata symbol)
- function removeSettleToken(address removeAddress)
- function addTradeToken(address addAddress, string calldata symbol)
- function removeTradeToken(address removeAddress)
- function addLeverage(uint8 leverage)
- function removeLeverage(uint8 leverage)

Contract `CbbcLiquidityToken`:

- function setRouter(address router_)

Contract `CbbcRebasePolicy`:

- function setMarketOracle(IMarketOracle marketOracle_)
- function setOrchestrator(address orchestrator_)
- function setRebaseLag(uint256 rebaseLag_)
- function setRebaseTimingParameters(uint256 minRebaseTimeIntervalSec_,uint256 rebaseWindowOffsetSec_, uint256 rebaseWindowLengthSec_)

Contract `CbbcToken`:

- function setRouter(address router_)
- function setBeta(int beta_)

Contract `MarketOracle`:

- function setInterestRateOracle(address token, address oracle)
- function setBidAskSpreadOracle(address token, address oracle)

- function setDailyPriceVolatilityOracle(address token, address oracle)
- function setDailyVolumeOracle(address token, address oracle)
- function setPriceOracle(address token, address oracle)

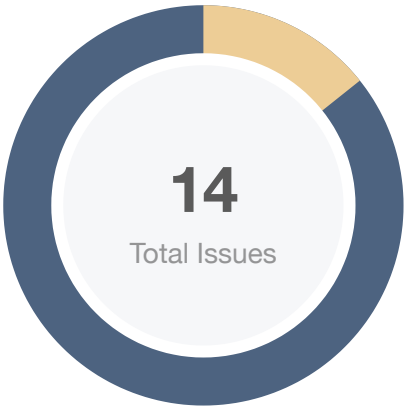
Contract `MedianOracle`:

- function setReportExpirationTimeSec(uint256 reportExpirationTimeSec_)
- function setReportDelaySec(uint256 reportDelaySec_)
- function setMinimumProviders(uint256 minimumProviders_)
- function addProvider(address provider)
- function removeProvider(address provider)

Contract `Orchestrator`:

- function addTransaction(address destination, bytes memory data)
- function removeTransaction(uint256 index)
- function setTransactionEnabled(uint256 index, bool enabled)

Findings



Critical	**0** (0.00%)
Major	**0** (0.00%)
Medium	**0** (0.00%)
Minor	**2** (14.29%)
Informational	**12** (85.71%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
CER-01	Lack of Input Validation	Logical Issue	Informational	Resolved
CFC-01	Missing Event Emitting	Logical Issue	Informational	Resolved
CFC-02	Unprotected Setting for `liquidityToken` and `cbbcToken`	Logical Issue	Minor	Resolved
CLT-01	Missing Event Emitting	Logical Issue	Informational	Resolved
CLT-02	Lack of Input Validation	Logical Issue	Informational	Resolved
CRC-01	Code Optimization	Gas Optimization	Informational	Resolved
CRC-02	Hard Code about Address `DATA_PROVIDER`	Logical Issue	Informational	Acknowledged
CRP-01	Missing Event Emitting	Logical Issue	Informational	Resolved
CRP-02	Initialization about `deviationThreshold`	Logical Issue	Informational	Acknowledged
CTC-01	Unprotected Setting for `rebasePolicy`	Logical Issue	Minor	Resolved
CTC-02	Unused Variables	Gas Optimization	Informational	Resolved
ERC-01	State Variable Shadowing	Coding Style	Informational	Resolved
MOK-01	Missing Event Emitting	Logical Issue	Informational	Resolved
OCK-01	Missing Event Emitting	Logical Issue	Informational	Resolved

CER-01 | Lack of Input Validation

Category	Severity	Location	Status
Logical Issue	Informational	CbbcERC20.sol: 52	Resolved

Description

Addresses should be checked before transferring to make sure they are not zero addresses. This suggestion also applies to other similar places.

Recommendation

Consider adding validation to check whether `to` is zero address like bellow:

```
function _transfer(address from, address to, uint value) private {
    require(to != address(0), "CBBC: can't transfer to zero address");
    balanceOf[from] -= value;
    balanceOf[to] += value;
    emit Transfer(from, to, value);
}
```

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

CFC-01 | Missing Event Emitting

Category	Severity	Location	Status
Logical Issue	Informational	CbbcFactory.sol: 152, 161, 168	Resolved

Description

It's sensitive to change settings but no events are emitted. This suggestion also applied to other similar places.

Recommendation

Consider emitting events when performing sensitive actions.

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

CFC-02 | Unprotected Setting for `liquidityToken` and `cbbcToken`

Category	Severity	Location	Status
Logical Issue	Minor	CbbcFactory.sol: 161, 168	Resolved

Description

Currently, anyone can set `liquidityToken` and `cbbcToken` since they can pass `address(this)` as `liquidityToken` or `cbbcToken` so that `msg.sender==liquidityToken/msg.sender==cbbcToken` will be always true. If some people call these methods maliciously, dirty data will be generated. And once they are set, they cannot be modified. For example, if `settleToken` is bound to a wrong `liquidityToken`, there is no way to revert it.

Recommendation

Consider adding a role to call these functions and adding functions to remove the dirty data or error settings.

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

CLT-01 | Missing Event Emitting

Category	Severity	Location	Status
Logical Issue	Informational	CbbcLiquidityToken.sol: 74, 79	Resolved

Description

It's sensitive to change settings but no events are emitted. This suggestion also applied to other similar places.

Recommendation

Consider emitting events when performing sensitive actions.

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

CLT-02 | Lack of Input Validation

Category	Severity	Location	Status
Logical Issue	Informational	CbbcLiquidityToken.sol: 62	Resolved

Description

Addresses should be checked before transferring to make sure they are not zero addresses. This suggestion also applies to other similar places.

Recommendation

Consider adding validation to check whether `to` is zero address like bellow:

```
    function _safeTransfer(address token, address to, uint value) private {
        require(to != address(0), "CBBC: can't transfer to zero address");
        (bool success, bytes memory data) = token.call(abi.encodeWithSelector(SELECTOR,
to, value));
        require(success && (data.length == 0 || abi.decode(data, (bool))), 'CBBC:
TRANSFER_FAILED');
    }
```

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

CRC-01 | Code Optimization

Category	Severity	Location	Status
Gas Optimization	Informational	CbbcRouter.sol: 216~217	Resolved

Description

There is no need to refund dust chain token since all `msg.value` has transferred to `liquidityToken`.

Recommendation

Consider commenting mentioned code.

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

CRC-02 | Hard Code about Address `DATA_PROVIDER`

Category	Severity	Location	Status
Logical Issue	Informational	CbbcRouter.sol: 425	Acknowledged

Description

`DATA_PROVIDER` is a hard code address. Please make sure this address is correct?

Alleviation

The development team responded that this address is used to provide data oracle. They will change this address from time to time, in order to avoid malicious data infusion due to privateKey leakage.

CRP-01 | Missing Event Emitting

Category	Severity	Location	Status
Logical Issue	Informational	CbbcRebasePolicy.sol: 128, 136, 148, 166	Resolved

Description

It's sensitive to change settings but no events are emitted. This suggestion also applied to other similar places.

Recommendation

Consider emitting events when performing sensitive actions.

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

CRP-02 | Initialization about `deviationThreshold`

Category	Severity	Location	Status
Logical Issue	Informational	CbbcRebasePolicy.sol: 41	Acknowledged

Description

`deviationThreshold` is not initialized, and it is used in function `withinDeviationThreshold()`. Currently, the result of this function is always false. Is this as intended?

Alleviation

The development team responded that they intionally let `deviationThreshold = 0` so that the users can rebase at any time they would like to. In the future, they may change `deviationThreshold` to 5%.

CTC-01 | Unprotected Setting for `rebasePolicy`

Category	Severity	Location	Status
Logical Issue	Minor	CbbcToken.sol: 137~138	Resolved

Description

Currently, anyone can set `rebasePolicy` since they can pass `address(this)` as `rebasePolicy` so that `msg.sender==rebasePolicy_` will be always true. If anyone who knows the address of `CbbcToken` can change the rebase policy, this is dangerous.

Recommendation

Consider making this function call only by the owner or adding a whitelist in this contract to maintain those addresses that can set `rebasePolicy`, and remove `cbbcToken_.setRebasePolicy(address(this));`(line 216) from contract `CbbcRebasePolicy`, once `CbbcRebasePolicy` was deployed successfully, then call `setRebasePolicy`.

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

CTC-02 | Unused Variables

Category	Severity	Location	Status
Gas Optimization	Informational	CbbcToken.sol: 87~88	Resolved

Description

These two variables `rebasePausedDeprecated` and `tokenPausedDeprecated` are just initialized， but never used.

Recommendation

Consider removing these two variables `rebasePausedDeprecated` and `tokenPausedDeprecated`.

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

ERC-01 | State Variable Shadowing

Category	Severity	Location	Status
Coding Style	Informational	libraries/ERC20Detailed.sol: 48	Resolved

Description

The state variable `ERC20Detailed._____gap` is shadowing `Initializable._____gap`.

Recommendation

Consider removing the state variable shadowing or changing the state variable name.

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

MOK-01 | Missing Event Emitting

Category	Severity	Location	Status
Logical Issue	Informational	MedianOracle.sol: 72, 85, 97	Resolved

Description

It's sensitive to change settings but no events are emitted. This suggestion also applied to other similar places.

Recommendation

Consider emitting events when performing sensitive actions.

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

OCK-01 | Missing Event Emitting

Category	Severity	Location	Status
Logical Issue	Informational	Orchestrator.sol: 68, 76, 91	Resolved

Description

It's sensitive to change settings but no events are emitted. This suggestion also applied to other similar places.

Recommendation

Consider emitting events when performing sensitive actions.

Alleviation

The development team heeded our advice and resolved this issue in commit `b9a9434b56083dc2e8b59354c63e75a5e70c89dc`.

Appendix

Finding Categories

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

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 Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with 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.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.