CERTIK

Security Assessment

WOOFi Swap

Oct 25th, 2021

Table of Contents

Summary

This report has been prepared for Wootech Limited to discover issues and vulnerabilities in the source code of the WOOFi Swap project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing 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 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 a premise that all external contracts and financial model formula were implemented safely.

The security assessment resulted in findings that ranged from critical 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;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	WOOFi Swap
Platform	BSC
Language	Solidity
Codebase	https://github.com/woonetwork/woofi_swap_smart_contracts/tree/main/miscellaneous/dex_for_audit/Certik_Sep_Audit_for_V1
Commit	40eed73a4890d3419e4d0c8981bf5ed3042afc52 6fda30f5c851cbe239dd4452c652898bd34abfdf

Audit Summary

Delivery Date	Oct 25, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
Major	1	0	0	1	0	0
Medium	0	0	0	0	0	0
Minor	2	0	0	0	0	2
Informational	7	0	0	3	0	4
Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
RMC	RewardManager.sol	67977dcc751b2350f718b95069a01a08b7cf07d870621e27aab1ed599c28edb0
WPP	WooPP_newmath.sol	522e8b59474bf836d203e28aaf60cc80a7b04ecdc103f438d3ed5e5c1646f0a5
WRC	WooRouter.sol	ad953aabf23662fd747188e221e3662924459c0f7299af7b473e8902e16be27b

Understandings

Overview

The WooPP contract is a contract for swapping tokens. There are two types of tokens involved in the contract: baseToken and quoteToken. Users can get quoteToken by selling baseToken or get baseToken by selling quoteToken.

The WooRouter contract is a contract used to exchange tokens, users exchange token through the `swap()` function, and a certain fee will be charged for each transaction. If the exchanged tokens do not include quoteToken, the contract will first exchange fromToken to quoteToken, then exchange quoteToken to toToken, the contract will charge fee twice. Fees will be accumulated on the user account designated by the user, and the user with the reward can claim the fee.

The RewardManager contract is a contract for managing rewards.Accounts with approve permissions can add the transaction fees to users as reward.

Privileged Functions

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

The `onlyOwner` modifier:

Contract `WooPP`:

- setStrategist(address strategist, bool flag)
- withdraw(address token, address to, uint256 amount)

Contract `RewardManager`:

- withdraw(address token, address to, uint256 amount)
- withdrawAll(address token, address to)
- approve(address user)
- revoke(address user)

Contract `WooRouter`:

- setPool(address _pool)
- rescueFunds(IERC20 token, uint256 amount)
- destroy()

- setWhitelisted(address target, bool whitelisted)

The `preventReentrant` modifier:

Contract `WooPP`:

- sellBase(address baseToken, uint256 baseAmount, uint256 minQuoteAmount, address from, address to, address rebateTo)
- sellQuote(address baseToken, uint256 quoteAmount, uint256 minBaseAmount, address from, address to, address rebateTo)
- setChainlinkRefOracle(address token, address newChainlinkRefOracle)
- addBaseToken(address baseToken, uint256 threshold, uint256 lpFeeRate, uint256 R, address chainlinkRefOracle)
- removeBaseToken(address baseToken)
- tuneParameters(address baseToken, uint256 newThreshold, uint256 newLpFeeRate, uint256 newR)

Contract `WooRouter`:

- externalSwap(address approveTarget, address swapTarget, address fromToken, address toToken, uint256 fromAmount, address payable to, bytes calldata data)

The `onlyStrategist` modifier:

Contract `WooPP`:

- setPairsInfo(string calldata _pairsInfo)
- setPriceOracle(address newPriceOracle)
- setChainlinkRefOracle(address token, address newChainlinkRefOracle)
- setRewardManager(address newRewardManager)
- addBaseToken(address baseToken, uint256 threshold, uint256 lpFeeRate, uint256 R, address chainlinkRefOracle)
- removeBaseToken(address baseToken)
- tuneParameters(address baseToken, uint256 newThreshold, uint256 newLpFeeRate, uint256 newR)
- withdrawToOwner(address token, uint256 amount)

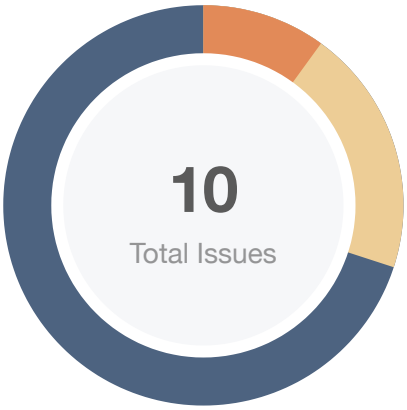
The `onlyApproved` modifier:

Contract `RewardManager`:

- addReward(address user, uint256 amount)
- setPriceOracle(address newPriceOracle

- setChainlinkRefOracle(address newRewardChainlinkRefOracle, address newQuoteChainlinkRefOracle)

Findings



Critical	**0** (0.00%)
Major	**1** (10.00%)
Medium	**0** (0.00%)
Minor	**2** (20.00%)
Informational	**7** (70.00%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
GLOBAL-01	Unlocked Compiler Version Declaration	Language Specific	Informational	Resolved
GLOBAL-02	Centralization Risk	**Centralization / Privilege**	**Major**	Acknowledged
RMW-01	Lack Of Access Control	Logical Issue	Minor	Resolved
RMW-02	Missing Emit Events	Coding Style	Informational	Resolved
RMW-03	Additional Rewards Not Transferred	Logical Issue	Informational	Acknowledged
WPP-01	Missing Emit Events	Coding Style	Informational	Resolved
WPP-02	Lack Of Access Control	Logical Issue	Minor	Resolved
WRW-01	Missing Input Validation	Logical Issue	Informational	Resolved
WRW-02	Potential Call Data Attack	Logical Issue	Informational	Acknowledged
WRW-03	Discussion For `internalFallbackSwap`	Logical Issue	Informational	Acknowledged

GLOBAL-01 | Unlocked Compiler Version Declaration

Category	Severity	Location	Status
Language Specific	Informational	Global	Resolved

Description

The compiler version utilized throughout the project uses the `^`, `>=` and `<=` prefix specifier, denoting that a compiler version that is greater than the version will be used to compile the contracts. It is recommend the compiler version should be consistent throughout the codebase.

Recommendation

It is a general practice to alternatively lock the compiler at a specific version rather than allow a range of compiler versions to be utilized to avoid compiler-specific bugs and in so doing be able to identify emerging ones more easily. We recommend locking the compiler at the lowest possible version that supports all the capabilities required by the codebase. This will ensure that the project utilizes a compiler version that has been in use for the longest time and as such is less likely to contain yet-undiscovered bugs.

Alleviation

The team heeded our advice and changed related codes. Code change was applied in commit 6fda30f5c851cbe239dd4452c652898bd34abfdf.

GLOBAL-02 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	Major	Global	Acknowledged

Description

In `WooRouter` contract, the `owner` has the permission to:

1. set pool address through function `setPool()`
2. transfer token to `msg.sender` through function `rescueFunds()`
3. destroy contract through function `destroy()`
4. set whitelist through function `setWhitelisted()`

In `WooPP` contract, the `owner` has the permission to:

1. set strategist through function `setStrategist()`
2. withdraw through function `withdraw()`

In `WooPP` contract, the `onlyStrategist` role has the permission to:

1. set pairs info through function `setPairsInfo()`
2. set price oracle address through function `setPriceOracle()`
3. set chain link ref oracle address through function `setChainlinkRefOracle()`
4. set reward manager address through function `setRewardManager()`
5. add base token through function `addBaseToken()`
6. remove base token through function `removeBaseToken()`
7. update token parameters through function `tuneParameters()`
8. withdraw the contract balance to the owner's account through function `withdrawToOwner`

In `RewardManager` contract, the `owner` has the permission to:

1. withdraw tokens to any account through function `withdraw()`
2. withdraw all tokens to the owner account through function `withdrawAll()`
3. approve users to approved account through function `approve()`
4. revoke users through function `revoke()`

In `RewardManager` contract, the `onlyApproved` roll has the permission to:

1. add reward to any user through function `addReward()`

2. set price oracle address through function `setPriceOracle`
3. set chain link ref oracle price through function `setChainlinkRefOracle` without obtaining the consensus of the community.

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at the different levels in terms of short-term and long-term:

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

Alleviation

The client response:
The risk decribed is controllable based on the design of WOOFi Swap. WOOFi Swap currently only supports one liquidity provider who supplies all capital in the liquidity pool. The owner in contract has the rights to withdraw funds. However, we implemented smart contract based multisignature wallet to be the owner, which requires at least 3 out of the 5 signers to approve any transaction. The signers are from both WOO Network and the market maker (i.e. Kronos) Furthermore, we plan to decentralized the owner of WOOFi Swap via community governance in the future.

Multisig wallet:
0xa0FA9C6fa8a5Dad6BEFF9F12EAd2e7d5e8D14E2c

Singers:
0x3961d488061C02bB4c15c81499056A16552aBb65
0x10D91375116751EcBA21d4E4dD95b5bAc6CafB3C
0xe6BbAce44fbB44a65437A538eC0AEC89663a5b59
0x23fD11e560958cdE53d7F482727FF07a818DeD8B
0xDe95557D3c243e116E40dD8e933c4c7A3939d515

RMW-01 | Lack Of Access Control

Category	Severity	Location	Status
Logical Issue	Minor	RewardManager.sol (WDEX): 38	Resolved

Description

In general, the `init()` function in the contract is called by the owner of the contract. The `init()` function in this contract lacks permission control, anyone can call the `init` function and become the `owner`.

Recommendation

Please make sure that the relevant functions are called correctly.

Alleviation

The team heeded our advice and changed related codes. Code change was applied in commit 6fda30f5c851cbe239dd4452c652898bd34abfdf.

RMW-02 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	RewardManager.sol (WDEX): 71, 97, 101, 105, 110	Resolved

Description

In contract `RewardManager`, there are numerous functions that can change state variables. However, these functions do not emit event to pass the changes out of chain.

Recommendation

It is recommended emitting events, for all the essential state variables that are possible to be changed during runtime.

Alleviation

The team heeded our advice and changed related codes. Code change was applied in commit 6fda30f5c851cbe239dd4452c652898bd34abfdf.

RMW-03 | Additional Rewards Not Transferred

Category	Severity	Location	Status
Logical Issue	Informational	RewardManager.sol (WDEX): 71	Acknowledged

Description

In the `addReward` function, only the value of pendingReward is updated, and no actual transfer operation is performed. In the WooPP contract, the reward is not transferred to the RewardManager contract.

Recommendation

Please ensure the contract has enough reward.

Alleviation

The client response:
It is intended by design. Reward token will be deposited into the reward manager by admin (with manual check). This process helps limit the loss of reward fund if ever hacked by external attackers.

WPP-01 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	WooPP_newmath.sol (WDEX): 361, 366, 378, 461	Resolved

Description

In contract `WooPP`, there are numerous functions that can change state variables. However, these functions do not emit events to pass the changes out of chain.

Recommendation

It is recommended emitting events, for all the essential state variables that are possible to be changed during runtime.

Alleviation

The team heeded our advice and changed related codes. Code change was applied in commit 6fda30f5c851cbe239dd4452c652898bd34abfdf.

WPP-02 | Lack Of Access Control

Category	Severity	Location	Status
Logical Issue	Minor	WooPP_newmath.sol (WDEX): 62	Resolved

Description

In general, the `init()` function in the contract is called by the owner of the contract. The `init()` function in this contract lacks permission control, anyone can call the `init` function and become the `owner`.

Recommendation

Please make sure that the relevant functions are called correctly.

Alleviation

The team heeded our advice and changed related codes. Code change was applied in commit 6fda30f5c851cbe239dd4452c652898bd34abfdf.

WRW-01 | Missing Input Validation

Category	Severity	Location	Status
Logical Issue	Informational	WooRouter.sol (WDEX): 44, 51	Resolved

Description

The given input is missing the sanity check for the non-zero address in the aforementioned line.

Recommendation

We recommend adding the check for the passed-in values to prevent unexpected error as below:
constructor():

```
44  require(address(_pool) != address(0), "_pool address cannot be 0");
```

setPool():

```
51  require(address(_pool) != address(0), "_pool address cannot be 0");
```

Alleviation

The team heeded our advice and changed related codes. Code change was applied in commit 6fda30f5c851cbe239dd4452c652898bd34abfdf.

WRW-02 | Potential Call Data Attack

Category	Severity	Location	Status
Logical Issue	Informational	WooRouter.sol (WDEX): 146	Acknowledged

Description

The `externalSwap` function has no call restriction. If the owner of the `swapTarget` contract is set as this contract, the hacker can call any function in the `swapTarget` contract that only the owner could call, such as obtaining the owner permission of the `swapTarget` contract (assuming the `swapTarget` contract inherits the `Ownable` contract).

Recommendation

We recommend to add call restrictions to the `externalSwap` function or do not set the owner of the `swapTarget` contract to the `WooRouter` contract.

Alleviation

The client response:
`swapTarget` won't never be set the owner as router contract. In fact, we have two protection mechanisms:

1. `swapTarget` must be in the `whitelist` (no contract is allowed by default, and we will only add a `swapTarget` from `dodoex` in production).
2. The external functions in the `WooRouter` contract are all `nonReentrant`, so `swapTarget` cannot call the functions in the `WooRouter` contract repeatedly.

WRW-03 | Discussion For `internalFallbackSwap`

Category	Severity	Location	Status
Logical Issue	Informational	WooRouter.sol (WDEX): 174	Acknowledged

Description

In the `internalFallbackSwap` function, `swapTarget.call` was used in the end. When `fromToken != _ETH_ADDRESS_` and `swapTarget` and `approveTarget` are not same, `swapTarget` is not approved. This may cause `swapTarget.call` to fail because it is not approved. Why only `approveTarget` is approved?

Recommendation

We recommend the client approving to `swapTarget`.

Alleviation

The client response:
`swapTarget` and `approveTarget` are made and maintained by `dodoex`. By design, `dodoex` will make sure the approval of `swapTarget` will make swap function work correctly.

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

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.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.

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 you ("Customer" or 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, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

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.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING

MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

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.