CERTIK

Code Security Assessment

Join

Mar 3rd, 2022

Table of Contents

Summary

This report has been prepared for Join to discover issues and vulnerabilities in the source code of the Join 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 smart contracts are implemented safely.

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	Join
Platform	BSC
Language	Solidity
Codebase	https://github.com/JoinCoin-Inc/JoinCoin-Contracts
Commit	dbe0a5c910caa2190993a825a4e8d918607bb445

Audit Summary

Delivery Date	Mar 03, 2022
Audit Methodology	Static Analysis, Manual Review

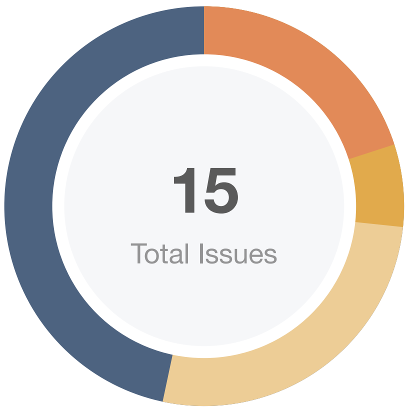
Vulnerability Summary

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

Audit Scope

ID	File	SHA256 Checksum
JCC	JoinCoin.sol	9ce690972aff8f9d9b4f58f6768b28d4370cc276e7ecd39a68d9bb9d64165e3c

Findings



🟥 **Critical**	**0** (0.00%)
🟧 **Major**	**3** (20.00%)
🟨 **Medium**	**1** (6.67%)
🟨 **Minor**	**4** (26.67%)
🟦 **Informational**	**7** (46.67%)
🟩 **Discussion**	**0** (0.00%)

ID	Title	Category	Severity	Status
GLOBAL-01	Third Party Dependencies	Volatile Code	● Minor	ⓘ Acknowledged
GLOBAL-02	Centralization Related Risks	**Centralization / Privilege**	● **Major**	ⓘ Acknowledged
GLOBAL-03	Financial Models	Logical Issue	● Medium	ⓘ Acknowledged
JCC-01	Unchangeable Wallet Address	Logical Issue	● Informational	ⓘ Acknowledged
JCC-02	Token Minted To Centralized Address	**Centralization / Privilege**	● **Major**	ⓘ Acknowledged
JCC-03	Inconsistent Between Code and Error Message	Inconsistency, Logical Issue	● Informational	ⓘ Acknowledged
JCC-04	Missing Emit Events	Coding Style	● Informational	ⓘ Acknowledged
JCC-05	Missing Input Validation	Volatile Code	● Minor	ⓘ Acknowledged
JCC-06	Related States Not Updated	Coding Style, Volatile Code	● Informational	ⓘ Acknowledged
JCC-07	Transfer Amount Zero	Gas Optimization	● Informational	ⓘ Acknowledged
JCC-08	Redundant code	Coding Style	● Informational	ⓘ Acknowledged
JCC-09	Potential Sandwich Attacks	Logical Issue	● Minor	ⓘ Acknowledged
JCC-10	Centralized Risk in `addLiquidity`	**Centralization / Privilege**	● **Major**	ⓘ Acknowledged

ID	Title	Category	Severity	Status
JCC-11	Miscalculation in `swapBack()`	Mathematical Operations	● Minor	ⓘ Acknowledged
JCC-12	Emit Mistakenly	Logical Issue	● Informational	ⓘ Acknowledged

GLOBAL-01 | Third Party Dependencies

Category	Severity	Location	Status
Volatile Code	● Minor	Global	ⓘ Acknowledged

Description

The contract is serving as the underlying entity to interact with third party protocols. The scope of the audit treats 3rd party entities as black boxes and assume their functional correctness. However, in the real world, 3rd parties can be compromised and this may lead to lost or stolen assets. In addition, upgrades of 3rd parties can possibly create severe impacts, such as increasing fees of 3rd parties, migrating to new LP pools, etc.

Recommendation

We understand that the business logic of `JoinCoin` requires interaction with `PancakeSwap`, etc. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

Alleviation

The client acknowledged.

GLOBAL-02 | Centralization Related Risks

Category	Severity	Location	Status
Centralization / Privilege	● Major	Global	ⓘ Acknowledged

Description

In the contract `Ownable`, the role `owner` has the authority over the following function:

- renounceOwnership
- transferOwnership

In the contract `JoinCoin`, the role `owner` has the authority over the following function:

- enableTrading
- removeLimits
- disableTransferDelay
- airdropToWallets
- updateSwapTokensAtAmount
- updateMaxAmount
- excludeFromMaxTransaction
- updateSwapEnabled
- updateBuyFees
- updateSellFees
- excludeFromFees
- setAutomatedMarketMakerPair
- updateMarketingWallet
- updateLiquidityWallet
- withdrawStuckETH

In the contract `JoinCoin`, the `liquidityWallet` can receive LP tokens. The accounts `marketingWallet`, `stakingAddress`, `friendlyWhaleWallet`, `devWallet` can receive BNB from contract.

Any compromise to these accounts may allow a hacker to take advantage of this authority.

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multi-signature wallets.

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

Short Term:

Timelock and Multi sign ($2/3$, $3/5$) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement;
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles;
  OR
- Remove the risky functionality.

Alleviation

The client acknowledged.

GLOBAL-03 | Financial Models

Category	Severity	Location	Status
Logical Issue	● Medium	Global	ⓘ Acknowledged

Description

The contract `JoinCoin` is a DeFi token deployed on Binance smart chain(BSC).

If both sender and receiver are not excluded from fee, each buy or sell transaction will be charged fees, including market fee, liquidity fee, dev fee, staking fee and friendly whale fee. The buy and sell fees can be set differently by owner at any time. All the fees will be sent to contract first. When meeting certain conditions, they will be used to add liquidity and be converted to BNB. The LP token is for `liquidityWallet`. The converted BNB is sent to market, dev, staking and friendly whale wallets according fee rates. It should be noted that all these wallets are initialized as owner account and the dev, staking and friendly whale wallets can not be changed any more.

Recommendation

We recommend the client to publish the financial models to the community.

Alleviation

The client acknowledged.

JCC-01 | Unchangeable Wallet Address

Category	Severity	Location	Status
Logical Issue	● Informational	JoinCoin.sol: 833~835	ⓘ Acknowledged

Description

The `stakingAddress`, `friendlyWhaleWallet`, `devWallet` are three privileged wallets and can receive BNB from contract. However, they are initialized as owner account in constructor and can never be changed any more. Please make sure this meets your intention.

Recommendation

Please make sure this meets your intention.

Alleviation

The client acknowledged.

JCC-02 | Token Minted To Centralized Address

Category	Severity	Location	Status
Centralization / Privilege	● **Major**	JoinCoin.sol: 856	ⓘ Acknowledged

Description

The amount of `totalSupply` tokens that are minted to the centralized address `msg.sender` who is `owner`, may raise the community's concerns about the centralization issue.

Recommendation

We advise the client to carefully manage the `owner` account's private key and avoid any potential risks of being hacked. We also advise the client to adopt Multisig, Timelock, and/or DAO in the project to manage this specific account in this case.

Alleviation

The client acknowledged.

JCC-03 | Inconsistent Between Code And Error Message

Category	Severity	Location	Status
Inconsistency, Logical Issue	● Informational	JoinCoin.sol: 885, 923	ⓘ Acknowledged

Description

The `airdropWallets.length < 200` means at most 199 wallets can be airdropped per transaction. This might be inconsistent with the error message.

```
885  require(airdropWallets.length < 200, "Can only airdrop 200 wallets per txn due to
gas limits");
```

The `buyTotalFees <= 10` means at most 10% buy fee can be set. This might be inconsistent with the error message.

```
923  require(buyTotalFees <= 10, "Must keep fees at 20% or less");
```

Recommendation

We advice the client to confirm the protocol design and modify the code.

Alleviation

The client acknowledged.

JCC-04 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	● Informational	JoinCoin.sol	ⓘ Acknowledged

Description

The following functions that affect the status of sensitive variables should be able to emit events as notifications to users.

- enableTrading
- removeLimits
- disableTransferDelay
- updateSwapTokensAtAmount
- updateMaxAmount
- updateSwapEnabled
- updateBuyFees
- updateSellFees

Recommendation

Consider adding events for sensitive actions, and emit them in the function.

Alleviation

The client acknowledged.

JCC-05 | Missing Input Validation

Category	Severity	Location	Status
Volatile Code	● Minor	JoinCoin.sol: 953, 958	ⓘ Acknowledged

Description

The given input is missing the check for the non-zero address.

Recommendation

We advise adding the check for the passed-in values to prevent unexpected error.

Alleviation

The client acknowledged.

JCC-06 | Related States Not Updated

Category	Severity	Location	Status
Coding Style, Volatile Code	● Informational	JoinCoin.sol: 958	ⓘ Acknowledged

Description

The `liquidityWallet` is excluded from fee and max transaction in the constructor. However, function `updateLiquidityWallet` does not update these two related states.

Recommendation

We recommend the client to exclude the new liquidity wallet from fee and max transaction and include the old liquidity wallet into fee and max transaction when updating `liquidityWallet`.

Alleviation

The client acknowledged.

JCC-07 | Transfer Amount Zero

Category	Severity	Location	Status
Gas Optimization	● Informational	JoinCoin.sol: 975~978	ⓘ Acknowledged

Description

It is unnecessary to transfer amount 0 to another account.

Recommendation

We recommend to add a `require` for gas optimization in front of the function `_transfer()` instead.

```
require(amount > 0, 'Transfer amount must be greater than zero');
```

Alleviation

The client acknowledged.

JCC-08 | Redundant Code

Category	Severity	Location	Status
Coding Style	● Informational	JoinCoin.sol: 984, 999	ⓘ Acknowledged

Description

1. The judgment conditions at Line 973 already include the judgment conditions at Line 984.
2. The if-condition at Line 983 and Line 999 are the same.

Recommendation

We recommend removing the redundant codes. Keeping code clear makes code easy to understand.

Alleviation

The client acknowledged.

JCC-09 | Potential Sandwich Attacks

Category	Severity	Location	Status
Logical Issue	● Minor	JoinCoin.sol: 1087, 1103~1104	ⓘ Acknowledged

Description

A sandwich attack might happen when an attacker observes a transaction swapping tokens or adding liquidity without setting restrictions on slippage or minimum output amount. The attacker can manipulate the exchange rate by frontrunning (before the transaction being attacked) a transaction to purchase one of the assets and make profits by backrunning (after the transaction being attacked) a transaction to sell the asset.

The following functions are called without setting restrictions on slippage or minimum output amount, so transactions triggering these functions are vulnerable to sandwich attacks, especially when the input amount is large:

- uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens()
- uniswapV2Router.addLiquidityETH()

Recommendation

We recommend setting reasonable minimum output amounts, instead of 0, based on token prices when calling the aforementioned functions.

Alleviation

The client acknowledged.

JCC-10 | Centralized Risk In `addLiquidity`

Category	Severity	Location	Status
Centralization / Privilege	● Major	JoinCoin.sol: 1095~1108	ⓘ Acknowledged

Description

```
// add the liquidity
uniswapV2Router.addLiquidityETH{value: ethAmount}(
    address(this),
    tokenAmount,
    0, // slippage is unavoidable
    0, // slippage is unavoidable
    liquidityWallet,
    block.timestamp
);
```

The `addLiquidity` function calls the `uniswapV2Router.addLiquidityETH` function with the `to` address specified as `liquidityWallet` for acquiring the generated LP tokens from the `JOIN-BNB` pool. As a result, over time the `liquidityWallet` address will accumulate a significant portion of LP tokens.If the `liquidityWallet` is an EOA (Externally Owned Account), mishandling of its private key can have devastating consequences to the project as a whole.

Recommendation

We advise the `to` address of the `uniswapV2Router.addLiquidityETH` function call to be replaced by the contract itself, i.e. `address(this)`, and to restrict the management of the LP tokens within the scope of the contract's business logic. This will also protect the LP tokens from being stolen if the `liquidityWallet` account is compromised. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or via smart-contract based accounts with enhanced security practices, f.e. Multisignature wallets.

Indicatively, here are some feasible solutions that would also mitigate the potential risk:

- Time-lock with reasonable latency, i.e. 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent 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 acknowledged.

JCC-11 | Miscalculation In `swapBack()`

Category	Severity	Location	Status
Mathematical Operations	● Minor	JoinCoin.sol: 1132~1135	ⓘ Acknowledged

Description

The `ethForMarketing`, `ethForDev`, `ethForStaking` and `ethForFriendlyWhale` are calculated mistakenly. The correct `ethForMarketing` is `ethBalance * tokensForMarketing / (totalTokensToSwap - tokensForLiquidity / 2)`, so does the `ethForDev`, `ethForStaking` and `ethForFriendlyWhale`.

Recommendation

We recommend the client to use this more accurate way.

Alleviation

The client acknowledged.

JCC-12 | Emit Mistakenly

Category	Severity	Location	Status
Logical Issue	● Informational	JoinCoin.sol: 1152	Acknowledged

Description

In function `swapBack()`, event `SwapAndLiquify` is emitted mistakenly. The `tokensForLiquidity` is set as zero at Line 1140. It is falsely used in the event.

```
1152  emit SwapAndLiquify(amountToSwapForETH, ethForLiquidity, tokensForLiquidity);
```

Recommendation

We recommend changing it this way:

```
1152  emit SwapAndLiquify(amountToSwapForETH, ethForLiquidity, liquidityTokens);
```

Alleviation

The client acknowledged.

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.

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.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

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.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

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.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

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.

CERTIK