Security Assessment

NEST Protocol

Jul 15th, 2022

Table of Contents

Summary

This report has been prepared for NEST Protocol to discover issues and vulnerabilities in the source code of the NEST Protocol 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.

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	NEST Protocol
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/NEST-Protocol/NEST-Oracle-V4.0
Commit	21296d12d2d4f5f5209bbc7c398c40ca651ce5d9

Audit Summary

Delivery Date	Jul 15, 2022 UTC
Audit Methodology	Static Analysis, Manual Review

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Mitigated	Partially Resolved	Resolved
Critical	0	0	0	0	0	0	0
Major	2	0	0	2	0	0	0
Medium	1	0	0	0	0	0	1
Minor	2	0	0	0	0	0	2
Informational	6	0	0	3	0	0	3
Discussion	0	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
NBP	NestBatchPlatform2.sol	774617be44035e75d05bc6295aefa50b42b693b26cc8e39bd087198a012c2963
NBM	NestBatchMining.sol	1f5aca4b443d16969424888797f9382aeee048e00f1cb3ece1f5edfca73e7fe1
NBN	NestBase.sol	4f3c689ecd19ab188119c956f7a15cdf56cd6470d2c48fb1dd827952bca160f2

Findings

Critical	**0** (0.00%)
Major	**2** (18.18%)
Medium	**1** (9.09%)
Minor	**2** (18.18%)
Informational	**6** (54.55%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
NEST-01	Financial Models	Volatile Code	Informational	Acknowledged
NBM-01	Centralization Risks In NestBatchMining.sol	**Centralization / Privilege,**	**Major**	Acknowledged
NBM-02	Potential ETH Loss	Logical Issue	Medium	Resolved
NBM-03	Missing Zero Address Validation	Volatile Code	Minor	Resolved
NBM-04	Discussion On The Condition Of `if` Branch	Logical Issue	Minor	Resolved
NBM-07	Missing Error Messages	Coding Style	Informational	Resolved
NBM-08	Discussion On Reward	Logical Issue	Informational	Resolved
NBM-09	Discussion On Parameter `scale`	Coding Style	Informational	Resolved
NBN-01	Centralization Risks In NestBase.sol	**Centralization / Privilege**	**Major**	Acknowledged
NES-01	Missing Emit Events	Coding Style	Informational	Acknowledged
NES-02	Unlocked Compiler Version	Language Specific	Informational	Acknowledged

NEST-01 | Financial Models

Category	Severity	Location	Status
Volatile Code	Informational		Acknowledged

Description

NEST Protocol is a distributed price oracle network on the Ethereum Mainnet. The NEST Oracle Smart Contract 4.0 is a Solidity smart contract implementation of NEST Protocol which provides a unique on-chain Price Oracle through a decentralized mechanism.

Different from other oracle approaches, NEST uses a unique “quotation mining” mechanism to ensure that off-chain price facts are generated on the chain synchronously, and NEST-Price price data is directly generated on the chain, which is designed to solve the industry problem of lack of price facts on the blockchain.(Reference : Nest Protocol White Paper)

Because the share of the sheet has no relation with the scale, and the price sheet with inappropriate prices will be harvested, the miner may choose to open price sheets with very little scale.

The miner needs to double the scale to arbitrage in the previous price sheet, which increases the risk of being arbitraged by other participants.

There are no mechanisms to encourage more price sheets to be posted or taken, and the volume is not large enough to truly reflect the price of the pair.

The following points are worth noting :

1. Each channel supports multiple token pairs, but only the first pair(channel.pairs[0]) can earn the reward.
2. Each channel will reward for the posted price in pairs[0], and the `opener` of channel could call function `NestBatchMining.increase())` to provide reward tokens, but the `opener` also can call function `NestBatchMining.decrease()` to withdraw reward tokens without limitation, which means role `Miner` or `Verifier` will not be able to get reward if the `opener` don't provide reward token or withdraw all of the reward tokens.
3. The amount of the award is related to `rewardPerBlock` in each channel. If the `opener` increases the `rewardPerBlock` without providing more reward tokens, then the `Miner` or `Verifier` will not get the reward they deserve.

Recommendation

This is not a problem and we recommend that users understand the contents of the white paper before using the protocol. Financial models of blockchain protocols need to be resilient to attacks. They need to pass simulations and verifications to guarantee the security of the overall protocol.

The financial model of this protocol is not in the scope of this audit.

Alleviation

NEST team acknowledged this finding.

NBM-01 | Centralization Risks In NestBatchMining.sol

Category	Severity	Location	Status
Centralization / Privilege,	**Major**	NestBatchMining.sol: 159, 222, 276	Acknowledged

Description

In the contract `NestBatchMining` :

The role `governance` has authority over the functions listed below:

- Function `setConfig()` : Modifying Nest mining configuration.

For each channel defined in `NestBatchMining`, the role `channel.opener` has authority over the functions listed below:

- Function `modify()` : Modify channel configuration.

The opener of each channel can withdraw assets from this channel, which may be deposited by other users.

- Function `decrease()`: Withdraw assets from this channel.

Any compromise to the privileged accounts may allow the 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., multisignature 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 (⅔, ⅗) 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 of privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key being 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 of 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

[NEST]: The vault and rewards in each channel represent assets that correspond to mined coins and earned ETH (or tokens in other chains), respectively.

Usually, the coins are injected by the opener, but in order to facilitate the injection of coins by the opener, any address is allowed to inject into the channel, and one should not call this interface if one's purpose is not to inject money into the coins, just as one should not transfer money to an unknown address. Simply injecting funds into the channel via this interface is an approval of the action.

NBM-02 | Potential ETH Loss

Category	Severity	Location	Status
Logical Issue	Medium	NestBatchMining.sol: 351	Resolved

Description

In function `post()`, Miners will transfer tokens to post the price for all of the pairs in the channel, and the amount of paid ETH will record in the local variable `fee`.

If the `channel.token0` or `pair.target` is ETH, the value of `fee` will decrease which means the tokens are frozen in this contract. But if more ETH is paid than is needed, the `msg.sender` will lose this part of ETH.

Recommendation

We recommend client to add logic for returning excess ETH.

Alleviation

[NEST]: The `post()` method has an important goal of **minimizing gas consumption**, which directly affects the cost per offer and thus the density of the NEST price. Returning excess ETH to miners or checking excess ETH adds a certain amount of additional gas consumption that needs to be borne by each miner who provides an offer. This is considered expected if failure to do so does not result in the contract or other normal miners' interests being compromised. Miner quoting is a very specialized operation and it is considered a risk that miners should take if they suffer losses due to miscalculations that result in excessive transfers of ETH.

NBM-03 | Missing Zero Address Validation

Category	Severity	Location	Status
Volatile Code	Minor	NestBatchMining.sol: 740	Resolved

Description

Addresses should be checked before assignment or external calls to make sure they are not zero addresses.

```
740     payable(to).transfer(value);
```

- `to` is not zero-checked before being used.

Recommendation

We advise adding a zero-check for the passed-in address value to prevent unexpected errors.

Alleviation

[NEST]: The assets represented by vault and rewards in each channel correspond to mined coins and earned ETH (or tokens in other chains), respectively.

The ETH in the channel is at the discretion of the opener, giving the opener enough freedom, including transferring ETH to a 0 address to indicate destruction.

NBM-04 | Discussion On The Condition Of `if` Branch

Category	Severity	Location	Status
Logical Issue	Minor	NestBatchMining.sol: 855~856	Resolved

Description

The condition of the `if` statement in line 856 is inconsistent with the comment in line 855.

```
855  // Not the same block (or flag is false), calculate the price and update it
856  if (flag || prev != height) {
```

Recommendation

We advise team to determine which of the comments and codes is correct, and then correct the incorrect one.

Alleviation

NEST team confirmed the code logic is correct and modified the comments.

NBM-07 | Missing Error Messages

Category	Severity	Location	Status
Coding Style	Informational	NestBatchMining.sol: 1070	Resolved

Description

The **require** can be used to check for conditions and throw an exception if the condition is not met. It is better to provide a string message containing details about the error that will be passed back to the caller.

Recommendation

We advise adding error messages to the linked **require** statements.

Alleviation

NEST added error messages to the linked statements.

NBM-08 | Discussion On Reward

Category	Severity	Location	Status
Logical Issue	Informational	NestBatchMining.sol: 396, 613~614, 698, 726	Resolved

Description

According to our understanding, for each channel, there is no essential difference between `pair[0]` and the other pairs, but in real logic only `pair[0]` has a reward. Please teach us why only `pair[0]` has a reward and teach us what part of the official documentation is this reward mechanism documented in.

```
396     // Only pairIndex 0 has reward
```

```
613     // Only pairIndex 0 has reward
```

Recommendation

We recommend reviewing this logic to ensure it meets the design intent.

Alleviation

[NEST]: It is the basic logic of NEST to have multiple quote pairs in a channel at the same time in order to make better use of the coins and to minimize the average quote cost so that fewer coins are used to drive the quotes. All quote pairs within a channel need to be quoted at the same time, and the bonus coins are calculated based on the quote interval, so it does not make sense to give a bonus to each quote pair.

Due to the design of the data structure, it is just possible to use the interval between two quotes of the quote array to calculate the number of mined coins due, and a normal quote will always have quote pair 0, so the interval of quote pair 0 is used as the criteria for calculating the bonus mined coins.

The Whitepaper mainly describes the mechanism of NEST price formation, but does not explain the specific implementation and optimization strategy, which is expected if it does not lead to undesirable consequences.

NBM-09 | Discussion On Parameter `scale`

Category	Severity	Location	Status
Coding Style	Informational	NestBatchMining.sol: 357	Resolved

Description

Since the value of `scale` must be 1, `scale` is not needed in the multiplication formula.

```
356    // 1. Check arguments
357    require(scale == 1, "NOM:!scale");
```

```
379    // Freeze token0
380    fee = _freeze(balances, channel.token0, cn * uint(channel.unit) * scale, fee);
```

```
387    fee = _freeze(balances, pair.target, scale * equivalent, fee);
```

Also, `scale` can be replaced by fixed value 1 when emit event `Post` or call function `_create()` in function `post()`.

```
394    // 6. Create token price sheet
395    emit Post(channelId, cn, msg.sender, pair.sheets.length, scale, equivalent);
396    // Only pairIndex 0 has reward
397    _create(pair.sheets, accountIndex, uint32(scale), uint(config.pledgeNest), cn == 0
? 1 : 0, equivalent);
```

In section 2.3 of white paper, the scale looks like it can be any value. Please describe why the value of `scale` must be 1 in the function `post()`.

Recommendation

We recommend reviewing the logic to ensure it meets the design intent.

Alleviation

[NEST]: The white paper mainly describes the price formation mechanism of NEST, and in the actual implementation, some restrictions are to be made. Limiting the quote size to 1 is to unify the initial quote

size, different quote sizes do not bring benefits, and keeping the `scale` parameter is for extending development.

`_create()` will be called by the `post()` and `take()` methods. After initialization, the value of `scale` should be 1, and then `take()` will double it.

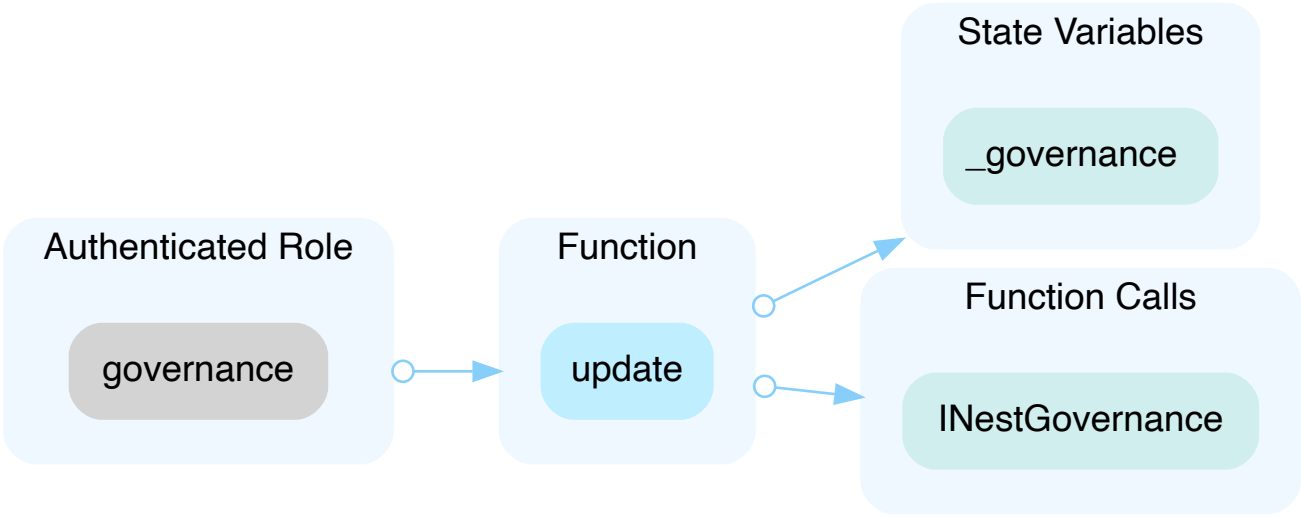
NBN-01 | Centralization Risks In NestBase.sol

Category	Severity	Location	Status
Centralization / Privilege	Major	NestBase.sol: 23	Acknowledged

Description

In the contract `NestBase` the role `governance` has authority over the functions shown in the diagram below.

Any compromise to the `governance` account may allow the 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., multisignature 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 (⅔, ⅗) 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 of privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key being 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 of 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

[NEST]: At the beginning of the project it was necessary to retain owner privileges for some parameters, and currently owner privileges have been managed with multi-signature accounts.

NES-01 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	NestBase.sol: 23; NestBatchMining.sol: 222, 247, 276, 291, 734	Acknowledged

Description

There should always be events emitted in the sensitive functions that are controlled by centralization roles.

Recommendation

It is recommended emitting events for the sensitive functions that are controlled by centralization roles.

Alleviation

[NEST]: This contract only does event logging for the more important operations (`open()` and `post()`), other operations, which can be obtained by monitoring data changes.

NES-02 | Unlocked Compiler Version

Category	Severity	Location	Status
Language Specific	Informational	NestBase.sol: 3; NestBatchMining.sol: 3; NestBatchPlatform2.sol: 3	Acknowledged

Description

The contract has unlocked compiler version. An unlocked compiler version in the source code of the contract permits the user to compile it at or above a particular version. This, in turn, leads to differences in the generated bytecode between compilations due to different compiler versions. This can lead to an ambiguity when debugging as compiler specific bugs may occur in the codebase that would be hard to identify over a span of multiple compiler versions rather than a specific one.

Recommendation

We advise that the compiler version is instead locked at the lowest version possible that the contract can be compiled at. For example, for version `v0.6.2` the contract should contain the following line:

```
pragma solidity 0.6.2;
```

Alleviation

[NEST]: Before deployment, all test cases are rerun against the target version of the compiler.

Optimizations

ID	Title	Category	Severity	Status
NBM-05	Lack Of Sanity Check	Volatile Code	Optimization	Acknowledged
NBM-06	Incorrect Comments	Inconsistency	Optimization	Resolved

NBM-05 | Lack Of Sanity Check

Category	Severity	Location	Status
Volatile Code	Optimization	NestBatchMining.sol: 440, 987	Acknowledged

Description

In the following calculation, the transaction will revert when the subtracted number is less than the subtracted number :

```
440    sheet.remainScales = uint32(uint(sheet.remainScales) - takeNum);
```

```
987    return value - tokenValue;
```

Recommendation

We recommend adding a `require` statement to check the input value and revert the call with an explicit error message.

Alleviation

[NEST]: This does not lead to security issues, and the focus is on reducing consumption, considering that `post()` and `take()` are specialized and frequent operations.

NBM-06 | Incorrect Comments

Category	Severity	Location	Status
Inconsistency	Optimization	NestBatchMining.sol: 444	Resolved

Description

File : NestBatchMining.sol (function `take()`, line 444)

The is no variable named `ethNum` in struct `PriceSheet`.

```
444  // sheet.token0Scales + sheet.token1Scales is always two times to sheet.ethNum
445  uint needNest1k = (takeNum << 2) * uint(sheet.nestNum1k) / (uint(sheet.token0Scales)
+ uint(sheet.token1Scales));
```

Recommendation

We recommend correcting the comment by using `sheet.remainScales`.

Meanwhile, the word `ethNum` appears several times in the comments of this project, we recommend that the team make sure the comments are consistent with the code.

Alleviation

NEST team modified the comments.

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.

Volatile Code

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

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.

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.