CERTIK

Preliminary Comments

Dcentraland - Rentals

Aug 25th, 2022

Table of Contents

Summary

This report has been prepared for Dcentraland to discover issues and vulnerabilities in the source code of the Dcentraland - Rentals 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	Dcentraland - Rentals
Platform	Ethereum
Language	Solidity
Codebase	common-contracts : https://github.com/decentraland/common-contracts/ rentals-contract : https://github.com/decentraland/rentals-contract/
Commit	dd1732c84f66130bd29ecea88d6855ed66b3ec9f aa419a79cd2e2dfe5bc3d838eecceed1dfef0387

Audit Summary

Delivery Date	Aug 25, 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	3	0	0	3	0	0	0
Medium	0	0	0	0	0	0	0
Minor	3	0	0	2	0	0	1
Informational	1	0	0	1	0	0	0
Discussion	1	0	0	0	0	0	1

Audit Scope

ID	Repo	Commit	File	SHA256 Checksum
NMT	decentraland/common-contracts	aa419a7	contracts/meta-transactions/NativeMetaTransaction.sol	ac2476d379e2903d8b9c1c7fc59430df5ee6952f41d96c67ef601409ed2f5192
NVB	decentraland/common-contracts	aa419a7	contracts/signatures/NonceVerifiable.sol	0577e0db9c5349a5af15a1a4027d447dc9e6e9b9bd6c9b5c9624b206a1756a0c
IER	decentraland/rentals-contract	dd1732c	contracts/interfaces/IERC721Rentable.sol	d6ab3dd729f6cce67e82b14038de077d8787e06f97d7e75258f5e609927dba7e
REN	decentraland/rentals-contract	dd1732c	contracts/Rentals.sol	e4aea9cd1099cac08f8efb85afe6194590f6afbc7558a2aa199fe2662a5d88de

Findings

Critical	0 (0.00%)
Major	3 (37.50%)
Medium	0 (0.00%)
Minor	3 (37.50%)
Informational	1 (12.50%)
Discussion	1 (12.50%)

ID	Title	Category	Severity	Status
CON-01	Centralized Control Of Contract Upgrade	**Centralization / Privilege**	**Major**	Acknowledged
NVB-01	Centralization Risks In `NonceVerifiable.sol`	**Centralization / Privilege**	**Major**	Acknowledged
NVB-02	Unclear Nonce Feature	Volatile Code	Discussion	Resolved
RCK-01	Unchecked ERC-20 `transfer()` / `transferFrom()` Call	Volatile Code	Minor	Acknowledged
REN-01	Centralization Risks In `Rentals.sol`	**Centralization / Privilege**	**Major**	Acknowledged
REN-02	Third Party Dependency	Volatile Code	Minor	Acknowledged
REN-03	Potential Reentrancy Attack (Events)	Volatile Code	Minor	Resolved
REN-05	Too Many Digits	Coding Style	Informational	Acknowledged

CON-01 | Centralized Control Of Contract Upgrade

Category	Severity	Location	Status
Centralization / Privilege	Major	contracts/meta-transactions/NativeMetaTransaction.sol (common-contracts): 7; contracts/signatures/NonceVerifiable.sol (common-contracts): 7	Acknowledged

Description

`NativeMetaTransaction` and `NonceVerifiable` are upgradeable contracts, the owner role can upgrade the contract without the community's commitment. If an attacker compromises the account, he can change the implementation of the contract and drain tokens from the contract.

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 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

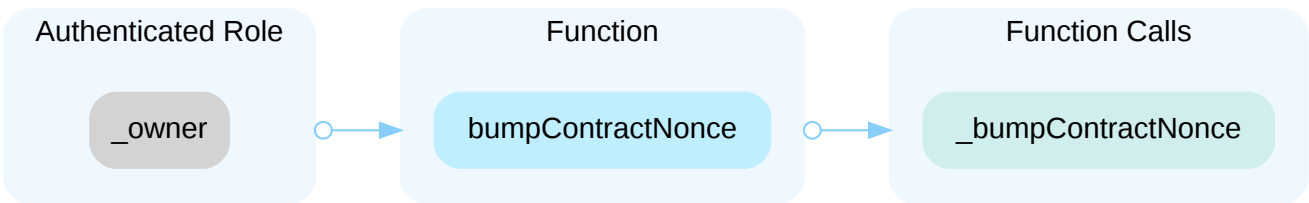
`[CertiK]`: The Dcentraland team will introduce multisig solution in the following update.

NVB-01 | Centralization Risks In `NonceVerifiable.sol`

Category	Severity	Location	Status
Centralization / Privilege	**Major**	contracts/signatures/NonceVerifiable.sol (common-contracts): 27	Acknowledged

Description

In the contract `NonceVerifiable` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and for example, renounce the ownership so `contractNonce` cannot be changed anymore.



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 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

`[CertiK]`: The Dcentraland team will introduce multisig solution in the following update.

NVB-02 | Unclear Nonce Feature

Category	Severity	Location	Status
Volatile Code	Discussion	contracts/signatures/NonceVerifiable.sol (common-contracts): 27~29, 32~34, 39~41	Resolved

Description

In the contract `NonceVerifiable.sol` there is three external functions allowing to change nonces:

- `bumpContractNonce()`;
- `bumpSignerNonce()`;
- `bumpAssetNonce()`.

`bumpContractNonce()` and `bumpSignerNonce()` are never used inside the scope of the audit, even when these nonces are verified meaning that it could be possible to reuse a nonce for a transaction.

Moreover `bumpSignerNonce()` and `bumpAssetNonce()` are not restricted meaning that anybody can interfer with these nonces values.

Recommendation

We advise to restrict the access of the `bumpSignerNonce()` and `bumpAssetNonce()` functions.
We would also recommend making the nonces usable only once by automatically changing a nonce once it has been used.

Alleviation

`[Dcentraland]`: The Rentals contract, which was part of this audit, uses it to verify that signatures have been signed with the correct nonces.

RCK-01 | Unchecked ERC-20 `transfer()` / `transferFrom()` Call

Category	Severity	Location	Status
Volatile Code	Minor	projects/rentals/contracts/Rentals.sol (test): 543, 544	Acknowledged

Description

The return value of the transfer()/transferFrom() call is not checked.

```
543         token.transferFrom(_tenant, _lessor, totalPrice - forCollector);
```

```
544         token.transferFrom(_tenant, feeCollector, forCollector);
```

Recommendation

Since some ERC-20 tokens return no values and others return a `bool` value, they should be handled with care. We advise using the OpenZeppelin's `SafeERC20.sol` implementation to interact with the `transfer()` and `transferFrom()` functions of external ERC-20 tokens. The OpenZeppelin implementation checks for the existence of a return value and reverts if `false` is returned, making it compatible with all ERC-20 token implementations.

Alleviation

`[Dcentraland]`: This contract is intended to be used, with the MANA token, despite having making the token updatable by the owner, it is not something we intend to do.
https://etherscan.io/address/0x0f5d2fb29fb7d3cfee444a200298f468908cc942 the MANA token always returns true after a transfer so checks are unnecessary

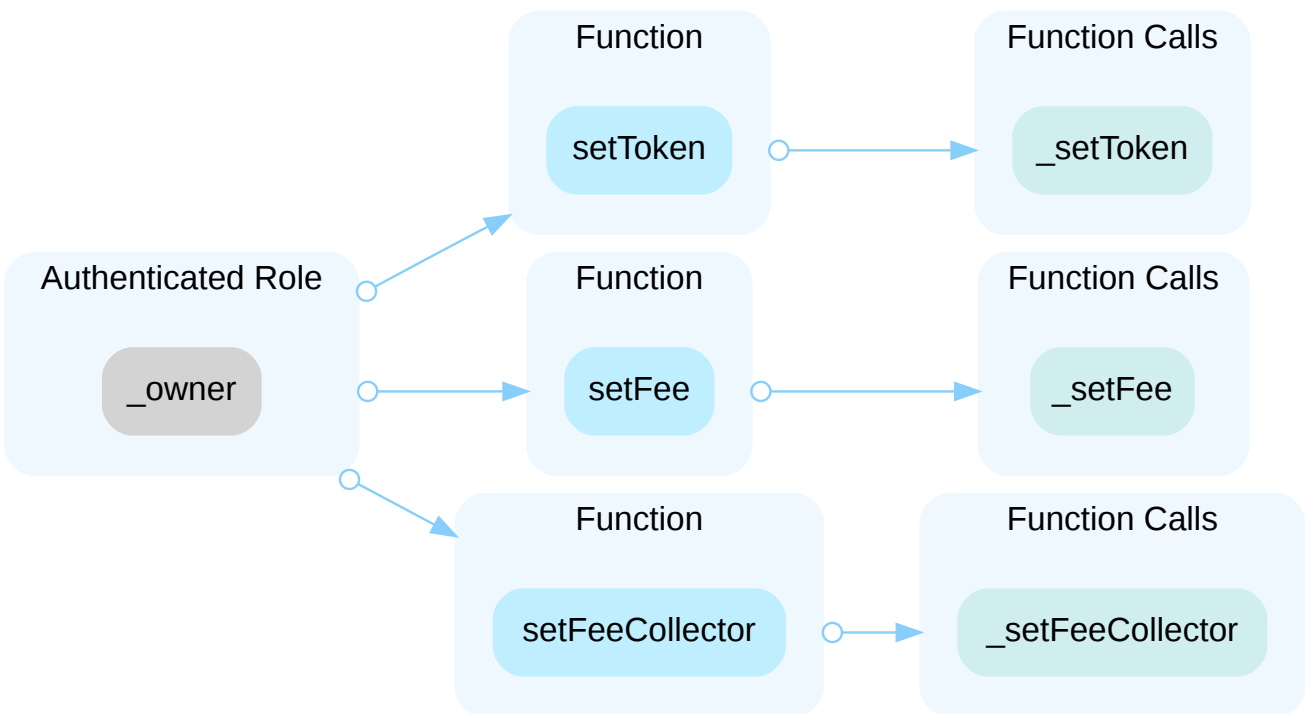
REN-01 | Centralization Risks In `Rentals.sol`

Category	Severity	Location	Status
Centralization / Privilege	**Major**	contracts/Rentals.sol (rentals-contract): 139, 145, 151	Acknowledged

Description

In the contract `Rentals` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and for example:

- set a designed ERC20 address as the `token` address of the contract to perform a specific reentrancy attack (as described in **REN-02**);
- set an address they control as the fee collector and set the highest fees allowed to divert as many tokens as possible.



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 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

`[CertiK]`: The Dcentraland team will introduce multisig solution in the following update.

REN-02 | Third Party Dependency

Category	Severity	Location	Status
Volatile Code	Minor	contracts/Rentals.sol (rentals-contract): 294, 423, 498, 507, 519	Acknowledged

Description

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

Recommendation

We understand that the business logic requires interaction with the third parties. We encourage the team to constantly monitor the statuses of third parties to mitigate the side effects when unexpected activities are observed.

Alleviation

`[CertiK]`: The Dcentraland team acknowledged the finding.

REN-03 | Potential Reentrancy Attack (Events)

Category	Severity	Location	Status
Volatile Code	Minor	contracts/Rentals.sol (rentals-contract): 277, 279, 308, 310, 470, 475, 479, 481~492, 543, 544	Resolved

Description

This finding has a minor impact because the reentrancy only causes out-of-order events.

A reentrancy attack can occur when the contract creates a function that makes an external call to another untrusted contract before resolving any effects. If the attacker can control the untrusted contract, they can make a recursive call back to the original function, repeating interactions that would have otherwise not run after the external call resolved the effects.

External call(s)

```
277         asset.safeTransferFrom(address(this), sender, _tokenId);
```

Events emitted after the call(s)

```
279         emit AssetClaimed(_contractAddress, _tokenId, sender);
```

External call(s)

```
308         asset.setUpdateOperator(_tokenId, _operator);
```

Events emitted after the call(s)

```
310         emit OperatorUpdated(_contractAddress, _tokenId, _operator, sender);
```

External call(s)

```
470             _handleTokenTransfers(_rentParams.lessor, _rentParams.tenant,
_rentParams.pricePerDay, _rentParams.rentalDays);
```

- This function call executes the following external call(s).

- In `Rentals._handleTokenTransfers`,
  - `token.transferFrom(_tenant,_lessor,totalPrice - forCollector)`
- In `Rentals._handleTokenTransfers`,
  - `token.transferFrom(_tenant,feeCollector,forCollector)`

```
475             asset.safeTransferFrom(_rentParams.lessor, address(this),
_rentParams.tokenId);
```

```
479         asset.setUpdateOperator(_rentParams.tokenId, _rentParams.operator);
```

Events emitted after the call(s)

```
481         emit AssetRented(
482             _rentParams.contractAddress,
483             _rentParams.tokenId,
484             _rentParams.lessor,
485             _rentParams.tenant,
486             _rentParams.operator,
487             _rentParams.rentalDays,
488             _rentParams.pricePerDay,
489             extend,
490             _msgSender(),
491             _rentParams.signature
492         );
```

Recommendation

We recommend using the Checks-Effects-Interactions Pattern to avoid the risk of calling unknown contracts or applying OpenZeppelin ReentrancyGuard library - `nonReentrant` modifier for the aforementioned functions to prevent reentrancy attack.

Alleviation

`[CertiK]`: The Dcentraland team heeded the advice and resolved the finding by adding the `ReentrancyGuard` in the commit 427641a4a3e4da50c778ce313cfcb28058de0a16

REN-05 | Too Many Digits

Category	Severity	Location	Status
Coding Style	Informational	contracts/Rentals.sol (rentals-contract): 356	Acknowledged

Description

Literals with many digits are difficult to read and review.

```
356         require(_fee <= 1_000_000, "Rentals#_setFee: HIGHER_THAN_1000000");
```

Recommendation

We advise the client to use the scientific notation to improve readability.

Alleviation

`[CertiK]`: The Dcentraland team acknowledged the finding.

Optimizations

ID	Title	Category	Severity	Status
REN-04	`.length` Called Multiple Times For The Same Variable	Gas Optimization	Optimization	Resolved

REN-04 | `.length` Called Multiple Times For The Same Variable

Category	Severity	Location	Status
Gas Optimization	Optimization	contracts/Rentals.sol (rentals-contract): 218, 219, 222	Resolved

Description

In the contract `Rentals.sol`, the function `acceptListing()` calls the function `.length`, on the same variable, three consecutive times.

This could be avoided by storing the value in a local variable.

Recommendation

We recommend storing the value in a local variable, for example:

```
215         uint length = _listing.pricePerDay.length;
216
217         // Verify that pricePerDay, maxDays and minDays have the same length
218         require(length == _listing.maxDays.length, "Rentals#acceptListing:
MAX_DAYS_LENGTH_MISSMATCH");
219         require(length == _listing.minDays.length, "Rentals#acceptListing:
MIN_DAYS_LENGTH_MISSMATCH");
220
221         // Verify that the provided index is not out of bounds of the listing
conditions.
222         require(_index < length, "Rentals#acceptListing: INDEX_OUT_OF_BOUNDS");
```

Alleviation

`[CertiK]`: The Dcentraland team heeded the advice and resolved the finding in the commit 1ad7dbb994f815cacbea54a4bd6e99b0d0a0f081

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.

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.

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.