Code Security Assessment

Sandbox V1

Jan 7th, 2022

Table of Contents

Summary

This report has been prepared for sandbox to discover issues and vulnerabilities in the source code of the Sandbox V1 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	Sandbox V1
Platform	ethereum
Language	Solidity
Codebase	• https://github.com/thesandboxgame/sandbox-smart-contracts • https://github.com/thesandboxgame/sandbox-smart-contracts-private
Commit	**sandbox-smart-contracts** • f7fad443b9a4730ead473598dbc7e36180871336 • 478d4b8391e9aba2f7e13fb66b6abeaaa7b22473 • 752e899abe7d5492227d28470a0bc2a0ae6df d41 • 328a3024d7100b7c645fc3e3338eb96896de852b **sandbox-smart-contracts-private** • 4309dc8a187d65ad422a66d09ad0e91f7e307109 • ab5791bba6f4983916feb14ea706fc13488711eb

Audit Summary

Delivery Date	Jan 07, 2022
Audit Methodology	Static Analysis, Manual Review
Key Components	

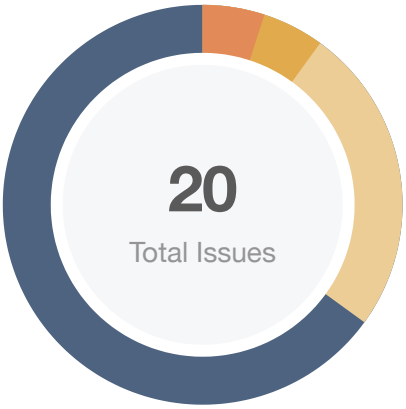
Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
Major	1	0	0	0	0	1
Medium	1	0	0	0	0	1
Minor	5	0	0	3	0	2
Informational	13	0	0	8	0	5
Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
ERC	Land/erc721/ERC721BaseToken.sol	abbef6990a4bb9d3a2f99a6330fa97b9a12de92a0f88ae9e30947e2c3ffcbd83
LBT	Land/erc721/LandBaseToken.sol	ebb6ab14f7766bc12a1d7c98566160d2ac4350c84c2294ba1d0f5623bcbbca48
LSL	LandSale/LandSale.sol	ea4d55a7903b0524d720c274250946ea606724497ae7679f9b75d611d69a1220
LSW	LandSale/LandSaleWithETHAndDAI.sol	392ea7d406daa4cced1f7659f7572eb501ed2623ccda35ea8899d904d94a9591
ABW	contracts_common/BaseWithStorage/Admin.sol	f336e6bd77e29368a3afe4ffecdc9eafe0b2854f2c303d47405a45a85bfcfb6e
MTR	contracts_common/BaseWithStorage/MetaTransactionReceiver.sol	8bae54108e69e81fcffe22425c311814d7339e078ae37e9c1c67c30cf4e4a6e9
AUP	contracts_common/UpgradableProxy/AdminUpgradeabilityProxy.sol	aad54e009cf2a954c392494410a4c7d699da2d7b5bda2aea745e604498439a53
PAU	contracts_common/UpgradableProxy/ProxyAdmin.sol	a94fdf65260ecaf842da22d8d428682f54f2df69c13546bcf58865792f0e7a2b
LKP	Land.sol	049b1ad829349d3deeea557bb19a6e36708520b359551272b5fdd6869b34ec8d

Findings



Critical	0 (0.00%)
Major	1 (5.00%)
Medium	1 (5.00%)
Minor	5 (25.00%)
Informational	13 (65.00%)
Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
GLOBAL-01	Upgradable contracts version	Language Specific	Informational	Resolved
ABW-01	Pull-Over-Push Pattern for changeAdmin()	Logical Issue	Informational	Acknowledged
ERC-01	Restrict access for `burnFrom()`	Logical Issue	Major	Resolved
ERC-02	Pull-Over-Push Pattern for constructor	Logical Issue	Informational	Acknowledged
ERC-03	Missing Return Value Handling	Logical Issue	Informational	Resolved
ERC-04	Proper usage of approveFor	Coding Style	Informational	Resolved
ERC-05	SafeMath Not Used	Mathematical Operations	Minor	Acknowledged
ERC-06	Function Visibility Optimization `_transferFrom`	Control Flow	Minor	Resolved
ERC-07	Assignment Optimization	Gas Optimization	Informational	Acknowledged
ERC-08	Access modifier should be 'internal' instead of 'public'	Control Flow	Medium	Resolved
LBT-01	Pull-Over-Push Pattern for constructor	Logical Issue	Informational	Acknowledged
LBT-02	SafeMath Not Used	Mathematical Operations	Minor	Acknowledged
LBT-03	Variable could be declared as `uint256`	Coding Style	Informational	Resolved

ID	Title	Category	Severity	Status
LBT-04	Proper usage of `pure`	Coding Style	Informational	Acknowledged
LBT-05	Proper usage of `view`	Coding Style	Informational	Acknowledged
LBT-06	SafeMath Not Used	Mathematical Operations	Minor	Acknowledged
LBT-07	Missing Input Validation	Volatile Code	Minor	Resolved
LBT-08	Missing Input Validation	Volatile Code	Informational	Acknowledged
LKP-01	Usage of `uint` Alias Instead of `uint256`	Coding Style	Informational	Acknowledged
LSL-01	Variable could be declared as `constant`	Gas Optimization	Informational	Resolved

GLOBAL-01 | Upgradable contracts version

Category	Severity	Location	Status
Language Specific	Informational	Global	Resolved

Description

According to the package.json, it seems the codebase referenced the library `@openzeppelin/contracts-upgradeable" with a version higher than "^4.0.0". It called us the attention here because openzeppelin released a hotfix for UUPS contract vulnerability for contract version v4.1.0 to v4.3.1. back in September this year.

Reference: https://forum.openzeppelin.com/t/security-advisory-initialize-uups-implementation-contracts/15301

Recommendation

Recommending to ensure the library @openzeppelin/contracts-upgradeable used is higher than v 4.3.1.

Alleviation

[Sandbox]: The team believed the issue is not impacted, the upgrades package is only being used in solidity v8 contracts where we are using TransparentProxies for deployment while the issue affects the UUPS deployments.

However, the team willing to upgrade the library version on the package.json, the changed is reflected in the

- **Repo:** https://github.com/thesandboxgame/sandbox-smart-contracts-private
- **commit hash:** ab5791bba6f4983916feb14ea706fc13488711eb

ABW-01 | Pull-Over-Push Pattern for changeAdmin()

Category	Severity	Location	Status
Logical Issue	Informational	projects/sandbox-v1/solc_0.5/contracts_common/BaseWithStorage/Admin.sol (4e2ba7f): 17	Acknowledged

Description

The change of `admin` by function `changeAdmin()` overrides the previously set `admin` with the new one without guaranteeing the new `admin` can actuate transactions on-chain.

Recommendation

Recommending to use the pull-over-push pattern to be applied here whereby a new `admin` is first proposed and consequently needs to accept the `admin` status ensuring that the account can actuate transactions on-chain.

Alleviation

[Sandbox]: The team decided to leave as is as we want to set it to the zero address in the future and we will make sure we do not set it by mistake.

ERC-01 | Restrict access for `burnFrom()`

Category	Severity	Location	Status
Logical Issue	Major	projects/sandbox-v1/solc_0.5/Land/erc721/ERC721BaseToken.sol (4e2ba7f): 371	Resolved

Description

The function `burnFrom(from, id)` will enable anyone to burn the item, when the item id's operator is set to its owner.

Recommendation

Recommending to restrict the access in the `burnFrom()` to msg.sneder.

Alleviation

[Sandbox]: The team addressed the issue and reflected in the commit hash 5f2e1a008d8c6e445de26886a59b19a0102d23f8

ERC-02 | Pull-Over-Push Pattern for constructor

Category	Severity	Location	Status
Logical Issue	Informational	projects/sandbox-v1/solc_0.5/Land/erc721/ERC721BaseToken.sol (4e2ba7f): 25~28	Acknowledged

Description

In the constructor, the variable `_admin` is assigned by an explicit `admin` address, and the input is not validated.

Recommendation

Recommending to set `msg.sender` as the initial admin and change the admin using the pull over push pattern later if it is necessary in case of initial human error.

Alleviation

[Sandbox]: The team disagree as we want to ensure the deployment account's only purpose is to deploy contract. It must not have any other responsibilities.

ERC-03 | Missing Return Value Handling

Category	Severity	Location	Status
Logical Issue	Informational	projects/sandbox-v1/solc_0.5/Land/erc721/ERC721BaseToken.sol (4e2ba7f): 45	Resolved

Description

In the function balanceOf the return value is missing in the function declaration.

Recommendation

Recommending to ensure the variable `_balance` should be assigned.

Alleviation

[Sandbox]: The team addressed the issue and reflected in the commit hash 478d4b8391e9aba2f7e13fb66b6abeaaa7b22473

ERC-04 | Proper usage of approveFor

Category	Severity	Location	Status
Coding Style	Informational	projects/sandbox-v1/solc_0.5/Land/erc721/ERC721BaseToken.sol (4e2ba7f): 87~91	Resolved

Description

In the function `approveFor`, it can extracted as a common internal function_approveFor() for better code reusability.

Recommendation

Recommending to implement the internal _approve() for better function reusability.

Alleviation

[Sandbox]: The team addressed the issue and reflected in the commit hash 478d4b8391e9aba2f7e13fb66b6abeaaa7b22473.

ERC-05 | SafeMath Not Used

Category	Severity	Location	Status
Mathematical Operations	Minor	projects/sandbox-v1/solc_0.5/Land/erc721/ERC721BaseToken.sol (4e2ba7f): 33, 355, 241	Acknowledged

Description

SafeMath from OpenZeppelin is not used in the following functions which makes them possible for overflow/underflow and will lead to an inaccurate calculation result.

- `_burn()`
- `_transferFrom()`
- `_batchTransferFrom()`

Recommendation

We advise the client to use OpenZeppelin's SafeMath library for all of the mathematical operations.

Reference: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/utils/math/SafeMath.sol

Alleviation

[Sandbox]: The team consider it is of no use if the logic of the contract ensure it will not happen.

ERC-06 | Function Visibility Optimization `_transferFrom`

Category	Severity	Location	Status
Control Flow	Minor	projects/sandbox-v1/solc_0.5/Land/erc721/ERC721BaseToken.sol (4e2ba7f): 3 3	Resolved

Description

The check for the validity of the transfer is missing in the function before making modifications to states. In the current code, _checkTransfer() is called before each call of _transferFrom() so the code is safe. However, this pattern is not guaranteed in future implementations

Recommendation

Recommending to add _checkTransfer() inside of _transferFrom() or wrapped as modifier.

Alleviation

[Sandbox]: The team will leave as is as we might need to have different logic for checking validity in different implementation.

ERC-07 | Assignment Optimization

Category	Severity	Location	Status
Gas Optimization	Informational	projects/sandbox-v1/solc_0.5/Land/erc721/ERC721BaseToken.sol (4e2ba7f): 21	Acknowledged

Description

Saving information of address owner and bool operatorEnabled in a uint256 is of high efficiency. However, this data structure requires developers to stay aware of the changes when they are trying to make conversion between uint256 and address.

Recommendation

Recommend to have a separate mapping for checking whether the operator is enabled.

Alleviation

[Sandbox]: The team will leave as is as we think the optimization benefit out-weight the need to ensure it is reset properly.

ERC-08 | Access modifier should be 'internal' instead of 'public'

Category	Severity	Location	Status
Control Flow	Medium	projects/sandbox-v1/solc_0.5/Land/erc721/ERC721BaseToken.sol (4e2ba7f): 355	Resolved

Description

The _burn() function has a 'public' access modifier, which might be invoked by any address.

Recommendation

Recommending the `_burn()` use the internal access modifier.

Alleviation

[Sandbox]: The team changed the visibility modifier of _burn function from public to internal in the repo:

- **Repo:** https://github.com/thesandboxgame/sandbox-smart-contracts-private
- **Commit hash:** 4309dc8a187d65ad422a66d09ad0e91f7e307109

LBT-01 | Pull-Over-Push Pattern for constructor

Category	Severity	Location	Status
Logical Issue	Informational	projects/sandbox-v1/solc_0.5/Land/erc721/LandBaseToken.sol (4e2ba7f): 39~43	Acknowledged

Description

In the constructor, the variable `_admin` is assigned by an explicit `admin` address, and the input is not validated.

Recommendation

Recommending to set the msg.sender as the initial admin and change the admin using the pull over push pattern later

Alleviation

[Sandbox]: The team disagreed as we want to ensure the deployment account's only purpose is to deploy the contract. It must not have any other responsibilities.

LBT-02 | SafeMath Not Used

Category	Severity	Location	Status
Mathematical Operations	Minor	projects/sandbox-v1/solc_0.5/Land/erc721/LandBaseToken.sol (4e2ba7f): 6	Acknowledged

Description

SafeMath from OpenZeppelin is not used in the following functions which makes them possible for overflow/underflow and will lead to an inaccurate calculation result.

Recommendation

We advise the client to use OpenZeppelin's SafeMath library for all of the mathematical operations.

Reference: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/utils/math/SafeMath.sol

Alleviation

[Sandbox]: The team considers it is of no use if the logic of the contract ensure it will not happen.

LBT-03 | Variable could be declared as `uint256`

Category	Severity	Location	Status
Coding Style	Informational	projects/sandbox-v1/solc_0.5/Land/erc721/LandBaseToken.sol (4e2ba7f): 81	Resolved

Description

The variable x,y,size are using `uint16`, which might cause additional type casting cost.

Recommendation

Recommend using uint256 instead of uint16 for variables x, y and size considering gas saving

Alleviation

[Sandbox]: The team changed the uint16 to uint256 as you recommended in commit f7fad443b9a4730ead473598dbc7e36180871336.

LBT-04 | Proper usage of `pure`

Category	Severity	Location	Status
Coding Style	Informational	projects/sandbox-v1/solc_0.5/Land/erc721/LandBaseToken.sol (4e2ba7f): 47, 53	Acknowledged

Description

The pure functions do not read or modify the state variables, which returns the values only using the parameters passed to the function or local variables present in it.

Recommendation

Recommending to use pure keyword as function decorator for both width() and height().

Alleviation

[Sandbox]: The team acknowledge the issue, but decided no change made in the current version.

LBT-05 | Proper usage of `view`

Category	Severity	Location	Status
Coding Style	Informational	projects/sandbox-v1/solc_0.5/Land/erc721/LandBaseToken.sol (4e2ba7f): 60, 68	Acknowledged

Description

The view functions are read-only function, which ensures that state variables cannot be modified after calling them.

Recommendation

Recommending to use view keyword for the function decorator for function x & y.

Alleviation

[Sandbox]: The team acknowledge the issue, but decided no change made in the current version

LBT-06 | SafeMath Not Used

Category	Severity	Location	Status
Mathematical Operations	Minor	projects/sandbox-v1/solc_0.5/Land/erc721/LandBaseToken.sol (4e2ba7f): 60	Acknowledged

Description

SafeMath from OpenZeppelin is not used in the following functions which makes them possible for overflow/underflow and will lead to an inaccurate calculation result.

- `x()`
- `y()`

Recommendation

We advise the client to use OpenZeppelin's SafeMath library for all of the mathematical operations.

Reference: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/utils/math/SafeMath.sol

Alleviation

[Sandbox]: The team checks for existence in commit 478d4b8391e9aba2f7e13fb66b6abeaaa7b22473, which should not require any SafeMath

LBT-07 | Missing Input Validation

Category	Severity	Location	Status
Volatile Code	Minor	projects/sandbox-v1/solc_0.5/Land/erc721/LandBaseToken.sol (4e2ba7f): 81	Resolved

Description

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

Recommendation

Recommending to check the validity of the recipient `to`.

Alleviation

[Sandbox]: The team addressed the issue and reflected in the commit hash 478d4b8391e9aba2f7e13fb66b6abeaaa7b22473

LBT-08 | Missing Input Validation

Category	Severity	Location	Status
Volatile Code	Informational	projects/sandbox-v1/solc_0.5/Land/erc721/LandBaseToken.sol (4e2ba7f): 81, 175, 291	Acknowledged

Description

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

Recommendation

[Sandbox]: It is currently done just after the coordinates and do not feel like it needs to be changed as coordinates need to be correct anyway.

Alleviation

[Sandbox]: The team acknowledge the issue, but decided no change made in the current version.

LKP-01 | Usage of `uint` Alias Instead of `uint256`

Category	Severity	Location	Status
Coding Style	Informational	projects/sandbox-v1/solc_0.5/Land.sol (4e2ba7f): 34	Acknowledged

Description

According to the coding practice, the `uint` is an alias for `uint256` and both represent the same underlying integer allocation. It is advisable that for clean coding practices the complete form `uint256` should be used instead of the alias `uint`.

Recommendation

Recommending to use uint256 instead of uint.

LSL-01 | Variable could be declared as `constant`

Category	Severity	Location	Status
Gas Optimization	Informational	projects/sandbox-v1/solc_0.5/LandSale/LandSale.sol (4e2ba7f): 100	Resolved

Description

the statemnt `408 (size of the land) is hard coded` could be declared as `constant` since these state variables are never to be changed.

Recommendation

Recommending to declare the variable as `constant`.

Alleviation

[Sandbox]: The team addressed the issue and reflected in the commit hash 328a3024d7100b7c645fc3e3338eb96896de852b

Appendix

Finding Categories

Gas Optimization

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

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.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

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.

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.