CERTIK

Security Assessment

Element.Market

Apr 26th, 2022

Table of Contents

Summary

This report has been prepared for Element.Market to discover issues and vulnerabilities in the source code of the Element.Market 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	Element.Market
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/libillhello/ElementEx
Commit	7a2fc6a49b18b0bf3084d5bb863170ff68702f67

Audit Summary

Delivery Date	Apr 26, 2022 UTC
Audit Methodology	Static Analysis, Manual Review

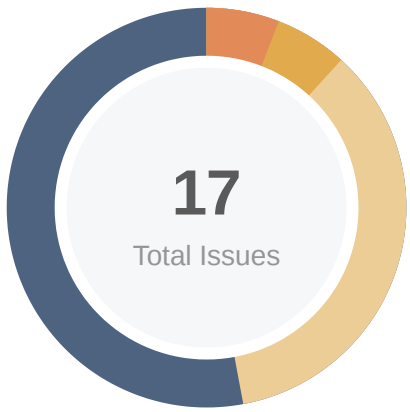
Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Mitigated	Partially Resolved	Resolved
Critical	0	0	0	0	0	0	0
Major	1	0	0	0	0	0	1
Medium	1	0	0	1	0	0	0
Minor	6	0	0	5	0	0	1
Informational	9	0	0	5	0	0	4
Discussion	0	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
LER	contracts/storage/LibERC721OrdersStorage.sol	39ed459665fe4b14ba0bdd11bcca82fdf078f8c1fe306e3f59be222fd3ab677c
IET	contracts/vendor/IEtherToken.sol	99f149c700b39573fe51213bf82d451ac06ea28cf630fc9890c90fcda5819689
LNF	contracts/features/libs/LibNFTOrder.sol	e9e665e0d5c1857a75ed9d8e501220eab20cdf55e6e1d3a203bba95b18618125
LCN	contracts/storage/LibCommonNftOrdersStorage.sol	99ad90075ba88dafdb72b96f85b401062d787328bd27915359f394cbc5782b73
IER	contracts/features/interfaces/IERC721OrdersFeature.sol	b1f9626081855a97adec6948a2f304fd62e232a310297b8f016490a970b2eecf
FER	contracts/fixins/FixinERC721Spender.sol	93bab778bcf13bebb55cc02d18ee994088ab193c43f51083dd375fc6389023ac
ERC	contracts/features/nft_orders/ERC721OrdersFeature.sol	e08a575f240a4c4ecc72a0396716d5526614aa323b103aa6a971a01bbdf2589f
LSK	contracts/storage/LibStorage.sol	d61f50ed3ee44a7354a63e5ed0d88f2614ba0ef75424ef1163f447c2096510b8
NFT	contracts/features/nft_orders/NFTOrders.sol	db63935fec7c8ec68b8fcd6f0dad125edaace03d12bfdfb45a2ac5953f52e541
LSC	contracts/features/libs/LibSignature.sol	e728759ffaaf809479679ef878ef8a7aba199958a29e9dbf5de54f0cf7a3e949
IPV	contracts/vendor/IPropertyValidator.sol	8fc30efc824d47d7730317004992b28011675b1ac99147725d728708028403f5
FEI	contracts/fixins/FixinEIP712.sol	bdc181f74f988e7686dec0e8e21285afab5ffa7368ff667f23f7ae446f810611
IFR	contracts/vendor/IFeeRecipient.sol	54010632d12caee57fbfeef98b131f030780b5c82a6941588d174e74cc90a31b
ITC	contracts/vendor/ITakerCallback.sol	e9092dcf4c21161524c8c7825c905f2e76e85bb39f658f0740bbc1808557293c
FTS	contracts/fixins/FixinTokenSpender.sol	d1d6fae854d51f3706ae6420e179555b1c411f936c3bc550e57a882ebb4dba6a

Findings



Critical	**0** (0.00%)
Major	**1** (5.88%)
Medium	**1** (5.88%)
Minor	**6** (35.29%)
Informational	**9** (52.94%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
GLOBAL-01	Unknown Implementation Of Interfaces	Volatile Code	Minor	Acknowledged
GLOBAL-02	Third Party Dependencies	Volatile Code	Minor	Acknowledged
GLOBAL-03	Financial Models	Logical Issue	Minor	Acknowledged
GLOBAL-04	Unlocked Compiler Version	Language Specific	Informational	Acknowledged
ERC-01	Potential Gas Exhaustion	Volatile Code	Minor	Acknowledged
ERC-02	Functions With `_` As Name Prefix Are Not `private` Or `internal`	Coding Style	Informational	Resolved
ERC-03	Improper Usage Of `public` And `external` Type	Gas Optimization	Informational	Resolved
ERC-04	Missing Error Messages	Coding Style	Informational	Resolved
FTS-01	Redundant Code Components	Volatile Code	Informational	Resolved
NFT-01	Incorrect `ITakeCallBack` Address	Logical Issue	Major	Resolved
NFT-02	Potential Unable To Receive ETH	Logical Issue	Medium	Acknowledged
NFT-03	Lack Of Zero Address Validation	Coding Style	Minor	Resolved
NFT-04	Potential Division By Zero	Logical Issue	Minor	Acknowledged

ID	Title	Category	Severity	Status
NFT-05	Redundant Calculation For `erc20FillAmount`	Gas Optimization	Informational	Acknowledged
NFT-06	Logic Issue Of Function `_buyNFT()`	Logical Issue	Informational	Acknowledged
NFT-07	Conditions Are Never Met	Logical Issue	Informational	Acknowledged
NFT-08	Mathematical Verification	Logical Issue	Informational	Acknowledged

GLOBAL-01 | Unknown Implementation Of Interfaces

Category	Severity	Location	Status
Volatile Code	Minor	Global	Acknowledged

Description

There is no contract implementation present for the interfaces `IEtherToken`, `IFeeRecipient`, `IPropertyValidator`, and `ITakerCallback` in the codebase. The scope of the audit treats 3rd party entities as black boxes and assumes 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 `NFTOrders`, and `ERC721OrdersFeature` requires interaction with outside protocols. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

Alleviation

The team acknowledged this issue and they stated the following:

"IEtherToken is the same as IWETH, they only use deposit(), withdraw(), it’s safe.

IFeeRecipient is the callback interface of feeRecipient and will be implemented as needed, it’s safe.

IPropertyValidator is the callback interface of property order and will be implemented as needed, it’s safe.

ITakerCallback is the callback interface of the taker and will be implemented as needed, it’s safe. "

GLOBAL-02 | 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 `exchange proxy`, and `ERC721 asset` protocols. The scope of the audit treats 3rd party entities as black boxes and assumes 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 `ERC721OrdersFeature` requires interaction with exchange proxy, and `ERC721 asset` protocols 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 team acknowledged this issue and they stated the following:

"The ERC721OrdersFeature will delegateCalled from a delegated proxy named `ElementEx`, they confirmed it's safe to interact with."

GLOBAL-03 | Financial Models

Category	Severity	Location	Status
Logical Issue	Minor	Global	Acknowledged

Description

Element ERC721OrdersFeature is based on 0x Protocol V4. It’s an Off-Chain + On-Chain mechanism, it has the following potential issues:

1. The order transaction fees and the recipient of fees are specified by the order maker, therefore the platform may not receive any fees.
2. The buyer or seller may bear the risk of the transaction due to the fluctuation of the NFT trading price by the buyer or seller.

Financial models of blockchain protocols need to be resilient to attacks. It needs to pass simulations and verifications to guarantee the security of the overall protocol. Financial models are not in the scope of the audit.

Recommendation

We would like to confirm with the client if the current implementation aligns with the original project design.

Alleviation

The team acknowledged this issue and they stated the following:

"The off-chain server of Element will verify every maker order’s param, so if the fee param is incorrect, it will not pass the check, and will not insert to element’s off-chain order book."

GLOBAL-04 | Unlocked Compiler Version

Category	Severity	Location	Status
Language Specific	Informational	Global	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 differing compiler version numbers. 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 `v0.8.13` the contract should contain the following line:

```
pragma solidity 0.8.13;
```

Alleviation

The team acknowledged this issue and they stated the following:

"Element’s contract will deploy use 0.8.13, and will verify the source code on etherescan.com, and the web page will show the version of the compiler. Most open-source projects do not lock the version, so it’s safe if the deployer knows this."

ERC-01 | Potential Gas Exhaustion

Category	Severity	Location	Status
Volatile Code	Minor	contracts/features/nft_orders/ERC721OrdersFeature.sol: 125~129, 157~163, 356~361	Acknowledged

Description

The `for` loop within the functions take the unbounded array's length as the maximum iteration times. If the size of the array grows large, iterating through the entire array could be an expensive operation considering there are external calls in the `for` loop.

Recommendation

We recommend setting constraints to the length of the array.

Alleviation

The team acknowledged this issue and they stated the following:

"They have also made a limit to the array length off-chain. This function will call from Element’s frontend web and the function parameters will be verified from the backend server."

ERC-02 | Functions With `_` As Name Prefix Are Not `private` Or `internal`

Category	Severity	Location	Status
Coding Style	Informational	contracts/features/nft_orders/ERC721OrdersFeature.sol: 194~197, 201~204	Resolved

Description

Functions with names starting with `_` should be declared as `private`/`internal`.

Recommendation

Consider changing function visibility to private or removing `_` from the start of the function name.

Alleviation

The team heeded our advice and resolved this issue in commit `85c8c4312fd6c8cd8634eb9c7b892e423bc0ed9c`.

ERC-03 | Improper Usage Of `public` And `external` Type

Category	Severity	Location	Status
Gas Optimization	Informational	contracts/features/nft_orders/ERC721OrdersFeature.sol: 672, 678	Resolved

Description

`public` functions that are never called by the contract could be declared as `external`. `external` functions are more efficient than `public` functions.

Recommendation

Consider using the external attribute for public functions that are never called within the contract.

Alleviation

The team heeded our advice and resolved this issue in commit `85c8c4312fd6c8cd8634eb9c7b892e423bc0ed9c`.

ERC-04 | Missing Error Messages

Category	Severity	Location	Status
Coding Style	Informational	contracts/features/nft_orders/ERC721OrdersFeature.sol: 195, 202, 421, 436	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

The team heeded our advice and resolved this issue in commit `85c8c4312fd6c8cd8634eb9c7b892e423bc0ed9c`.

FTS-01 | Redundant Code Components

Category	Severity	Location	Status
Volatile Code	Informational	contracts/fixins/FixinTokenSpender.sol: 122~124	Resolved

Description

The linked statements do not affect the functionality of the codebase and appear to be either leftovers from test code or older functionality.

Recommendation

We advise to remove the redundant statements for production environments.

Alleviation

The team heeded our advice and resolved this issue in commit `85c8c4312fd6c8cd8634eb9c7b892e423bc0ed9c`.

NFT-01 | Incorrect `ITakeCallBack` Address

Category	Severity	Location	Status
Logical Issue	Major	contracts/features/nft_orders/NFTOrders.sol: 211, 220	Resolved

Description

In the function `_buyNFTEx()`, a taker callback function will be invoked when receiving the NFT token, however, the callback address is incorrect, it should be `params.taker` instead of `msg.sender` since the buyer is `params.taker`.

```
bytes4 callbackResult =
ITakerCallback(msg.sender).zeroExTakerCallback(orderInfo.orderHash,
params.takerCallbackData);
```

Recommendation

We recommend ensuring the callback function will be invoked on correct address.

```
bytes4 callbackResult =
ITakerCallback(params.taker).zeroExTakerCallback(orderInfo.orderHash,
params.takerCallbackData);
```

Alleviation

The team heeded our advice and resolved this issue in commit `85c8c4312fd6c8cd8634eb9c7b892e423bc0ed9c`.

NFT-02 | Potential Unable To Receive ETH

Category	Severity	Location	Status
Logical Issue	Medium	contracts/features/nft_orders/NFTOrders.sol: 107~110, 238~242 contracts/features/nft_orders/ERC721OrdersFeature.sol: 267~270	Acknowledged

Description

As per the Element Contract Architecture, the ERC721OrdersFeature will be called from a delegated proxy. The function `WETH.withdraw()` unwraps WETH and transfers ETH to the delegated proxy contract, then transfers ETH to the taker via calling the function `_transferEth(), so please make sure the function` receive()`is declared in the delegate proxy contract to successfully receive the ETHs, also ensure the function` _fallback()`is not called in the function` receive()` to avoid the risk of gas inefficiency.

```
receive () external payable virtual {
    // no _fallback();
}
```

And the implementation of the delegated proxy contract is not in the scope of this audit.

Recommendation

We recommend ensuring the delegated proxy contract can successfully receive ETHs.

Alleviation

The team acknowledged this issue and stated they will ensure the delegate proxy code is safe to receive ETHs.

NFT-03 | Lack Of Zero Address Validation

Category	Severity	Location	Status
Coding Style	Minor	contracts/features/nft_orders/NFTOrders.sol: 51	Resolved

Description

Address should be checked before assignment to make sure it is not zero addresses.

Recommendation

Consider adding a zero check.

Alleviation

The team heeded our advice and resolved this issue in commit `85c8c4312fd6c8cd8634eb9c7b892e423bc0ed9c`.

NFT-04 | Potential Division By Zero

Category	Severity	Location	Status
Logical Issue	Minor	contracts/features/nft_orders/NFTOrders.sol: 346~347	Acknowledged

Description

If the value of `denominator` is 0, the linked operation will fail due to the divide by 0 error, which ultimately makes the invocation to `_resetDutchAuctionTokenAmountAndFees()` function fail.

Recommendation

We recommend adding a validation in the function `_resetDutchAuctionTokenAmountAndFees()`.

Alleviation

The team acknowledged this issue and they stated the following:

"If someone directly call this contract function, and when the denominator=0, the transaction will revert, Same as add require(denominator != 0, “ZERO”), so it’s safe."

NFT-05 | Redundant Calculation For `erc20FillAmount`

Category	Severity	Location	Status
Gas Optimization	Informational	contracts/features/nft_orders/NFTOrders.sol: 93~94, 158~159	Acknowledged

Description

The `erc20FillAmount` calculation is redundant for the NFT transaction since the `orderInfo.orderAmount` and `params.sellAmount` alway equal to 1.

Recommendation

We advise the client to revisit the function and simplify this calculation as below,

```
erc20FillAmount = buyOrder.erc20TokenAmount;
```

```
erc20FillAmount = sellOrder.erc20TokenAmount
```

Alleviation

The team acknowledged this issue and they stated the following:

"Element will support ERC1155 in the future, and the ERC1155OrdersFeature will be inherited from the NFTOrders. So they need these codes to support later features, and it’s not redundant and it’s safe."

NFT-06 | Logic Issue Of Function `_buyNFT()`

Category	Severity	Location	Status
Logical Issue	Informational	contracts/features/nft_orders/NFTOrders.sol: 138~142	Acknowledged

Description

The buyer can invoke the function `_buyNFT()` to buy the NFT, however, we can not find the callback logic to confirm the NFT is received by the buyer(msg.sender). We would like to confirm with the client if the current implementation aligns with the original project design.

```
// Invoke the callback
bytes4 callbackResult = ITakerCallback(msg.sender)
.zeroExTakerCallback(orderInfo.orderHash, params.takerCallbackData);
```

Recommendation

We advise the client to revisit the design and ensure it is intended.

Alleviation

The team acknowledged this issue and they stated the following:

"For the purpose of saving gas, they made two functions _buyNFT() and _buyNFTEx() in contract ERC721OrdersFeature, they believe the callback is an advanced requirement and only supported in _buyNFTEx()."

NFT-07 | Conditions Are Never Met

Category	Severity	Location	Status
Logical Issue	Informational	contracts/features/nft_orders/NFTOrders.sol: 199 contracts/features/nft_orders/ERC721OrdersFeature.sol: 233	Acknowledged

Description

In the function `_buyNFTEx()`/`matchERC721Orders()`, the linked condition is almost never met.

Recommendation

We advise the client to revisit the design and ensure it is intended.

Alleviation

The team acknowledged this issue and stated they reuse the 256-bit of Expiry to save gas.

NFT-08 | Mathematical Verification

Category	Severity	Location	Status
Logical Issue	Informational	contracts/features/nft_orders/NFTOrders.sol: 337, 355~360	Acknowledged

Description

The protocol is using some algorithms, including in the logic of the functions `_resetDutchAuctionTokenAmountAndFees()` and `_resetEnglishAuctionTokenAmountAndFees()`. The Mathematical verification of these algorithms is not in the scope of this audit. The function logic will be checked based on the requirement documents.

Recommendation

We advise the client to revisit the design and ensure it is intended.

Alleviation

The team acknowledged this issue and they stated these two functions are correct and satisfy business logic, and it’s safe.

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.

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.

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.