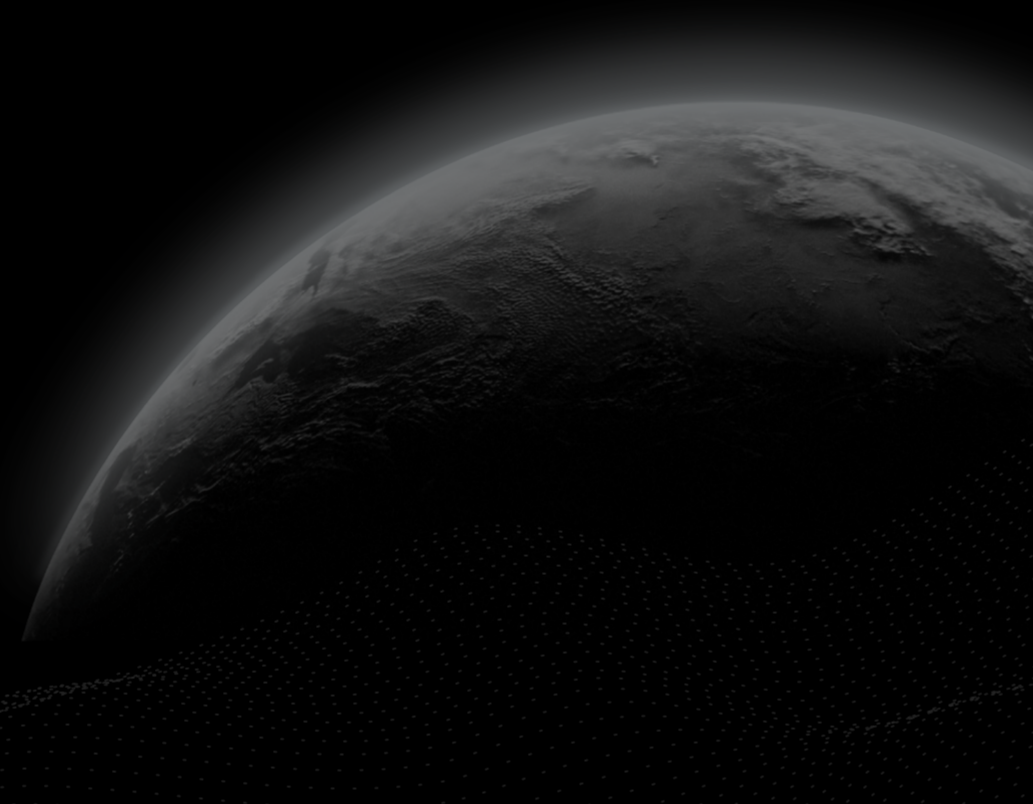
CERTIK

Security Assessment

Coresky-Audit

CertiK Verified on Feb 28th, 2023



CertiK Verified on Feb 28th, 2023

Coresky-Audit

The security assessment was prepared by CertiK, the leader in Web3.0 security.

Executive Summary

TYPES	ECOSYSTEM	METHODS
DeFi	Ethereum (ETH)	Manual Review, Static Analysis
LANGUAGE	**TIMELINE**	**KEY COMPONENTS**
Solidity	Delivered on 02/28/2023	N/A

CODEBASE

https://github.com/csgithub007/core_contract/tree/24c931d99cf63c874317c4097c06a52f01217072

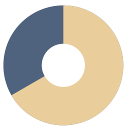
...View All

COMMITS

24c931d99cf63c874317c4097c06a52f01217072

...View All

Vulnerability Summary



9	0	0	0	9	0	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined	Unresolved

🟥	0	Critical		Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
🟧	0	Major		Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
🟨	0	Medium		Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
🟨	6	Minor	6 Acknowledged	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
🟦	3	Informational	3 Acknowledged	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | CORESKY-AUDIT

CODEBASE | CORESKY-AUDIT

Repository

https://github.com/csgithub007/core_contract/tree/24c931d99cf63c874317c4097c06a52f01217072

Commit

24c931d99cf63c874317c4097c06a52f01217072

AUDIT SCOPE | CORESKY-AUDIT

7 files audited ● 5 files with Acknowledged findings ● 2 files without findings

ID	File	SHA256 Checksum
● MRB	contracts/MarketRegistry.sol	cfdc0834f1b6249c0bafafd0a31f0294fb2c0967b3ee6077039032eac1158783
● MTT	contracts/MarketTokenTransferProxy.sol	375b7ab89ca437ed510e926b47cf778ade5f93291ef91436762f18ee5f720582
● MDB	contracts/MerkleDistributor.sol	fd5965b2b47bf9fa00843cd07daac472dce42253cf949f0fd0c4f6247f0cbe2e
● NFT	contracts/NFTMarket.sol	bf4cbf1846c16218fb6ecfa609606abb07df84428078bb6cd2735ab7202781d9
● NFM	contracts/NFTMarketWrap.sol	df5a93f822eb02ee14d6407a0a864d4b0690b77f4d790af6acadfae08e227eae
● DEP	contracts/Deposit.sol	ddb084fa730e30119230cde3359325578adf063b01f1e689e8fd83703ddc6723
● IDB	contracts/IDeposit.sol	3c47c5e045d1483d9a3e3f864ab2c4a1ce3a3e151c63b50449784c4412d92b7d

APPROACH & METHODS | CORESKY-AUDIT

This report has been prepared for Coresky to discover issues and vulnerabilities in the source code of the Coresky-Audit project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Manual Review and Static Analysis 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:

- Testing the smart contracts against both common and uncommon attack vectors;
- 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.

DECENTRALIZATION EFFORTS | CORESKY-AUDIT

Description

In the contract `Deposit` , the role `DEFAULT_ADMIN_ROLE` has authority over the following functions:

- grantWithdraw
- grantRole
- revokeRole
- renounceRole

In the contract `Deposit` , the role `WITHDRAW_ROLE` has authority over the following functions:

- multicall
- withdrawERC721
- batchWithdrawERC721
- withdrawERC1155
- batchWithdrawERC1155
- renounceRole

In the contract `Deposit` , the role granted by the `DEFAULT_ADMIN_ROLE` has authority over the following functions:

- withdrawERC721
- batchWithdrawERC721
- withdrawERC1155
- batchWithdrawERC1155
- renounceRole

In the contract `MerkleDistributor` , the role `DEFAULT_ADMIN_ROLE` has authority over the following functions:

- grantRole
- revokeRole
- renounceRole

In the contract `MerkleDistributor` , the role `CREATE_ROLE` has authority over the following function:

- launchpad
- renounceRole

In the contract `AuthenticatedProxy` , the role `user` has authority over the following functions:

- setRevoke
- proxy
- proxyAssert

In the contract `AuthenticatedProxy` , the role `authenticated contracts` has authority over the following functions:

- proxy
- proxyAssert

In the contract `OwnableDelegateProxy` , the role `proxyOwner` has authority over the following functions:

- transferProxyOwnership
- upgradeTo
- upgradeToAndCall

In the contract `MarketRegistry` , the role `owner` has authority over the following functions:

- grantInitialAuthentication
- startGrantAuthentication
- endGrantAuthentication
- revokeAuthentication
- transferOwnership
- renounceOwnership

In the contract `MarketTokenTransferProxy` , the role `authenticated contracts` has authority over the following function:

- transferFrom

In the contract `MarketExchange` , the role `owner` has authority over the following functions:

- changeExchangeToken
- changeChainID
- changeExchangeWrap
- changeMinimumMakerProtocolFee
- changeMinimumTakerProtocolFee
- changeProtocolFeeRecipient
- transferOwnership

- renounceOwnership

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

Recommendations

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

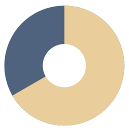
- Remove the risky functionality.

Alleviation

[`Coresky`]: The Deposit contract is used to assist the project party to issue NFT. When the contract needs to be used, the project party deploys it by itself. The DEFAULT_ADMIN_ROLE authority is delivered to the project party, and the project party assigns WITHDRAW_ROLE to the trusted contract address.

The permission setting in other contracts is to ensure the security of the contract and prevent ordinary users from modifying it at will and causing the contract to fail to execute normally.

FINDINGS | CORESKY-AUDIT



9	0	0	0	6	3
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Coresky-Audit. Through this audit, we have uncovered 9 issues ranging from different severity levels. Utilizing the techniques of Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
CON-01	Usage Of `transfer` / `send` For Sending Ether	Volatile Code	Minor	● Acknowledged
CON-02	Pull-Over-Push Pattern	Logical Issue	Minor	● Acknowledged
CON-06	Lack Of Input Validation	Volatile Code	Minor	● Acknowledged
GLOBAL-02	Third Party Dependency	Volatile Code	Minor	● Acknowledged
NFT-01	No Upper Limit	Logical Issue	Minor	● Acknowledged
NFT-02	Missing Zero Address Validation	Volatile Code	Minor	● Acknowledged
CON-04	Missing Error Messages	Coding Style	Informational	● Acknowledged
CON-05	Missing Emit Events	Coding Style	Informational	● Acknowledged
MRB-01	Potential Compiler Error	Compiler Error	Informational	● Acknowledged

CON-01 | USAGE OF `transfer` / `send` FOR SENDING ETHER

Category	Severity	Location	Status
Volatile Code	● Minor	contracts/MerkleDistributor.sol: 93~94; contracts/NFTMarket.sol: 1001, 1011, 1021, 1031, 1089, 1093	● Acknowledged

Description

It is not recommended to use Solidity's `transfer()` and `send()` functions for transferring Ether, since some contracts may not be able to receive the funds. Those functions forward only a fixed amount of gas (2300 specifically) and the receiving contracts may run out of gas before finishing the transfer. Also, EVM instructions' gas costs may increase in the future. Thus, some contracts that can receive now may stop working in the future due to the gas limitation. Here is some examples:

```
93            if(refund > 0) payable(msg.sender).transfer(refund);
94            project.receipt.transfer(total);
```

- `MerkleDistributor.claim` uses `transfer()`.

```
1001                        sell.feeRecipient.transfer(makerRelayerFee);
```

- `ExchangeCore.executeFundsTransfer` uses `transfer()`.

Recommendation

We recommend using the `Address.sendValue()` function from OpenZeppelin.

Since `Address.sendValue()` may allow reentrancy, we also recommend guarding against reentrancy attacks by utilizing the Checks-Effects-Interactions Pattern or applying OpenZeppelin ReentrancyGuard.

Alleviation

[`CertiK`]: The dev team explained the issue, the `launchpad` and `sell.feeRecipient` only support EOA. When placing an order on this platform, the team chose the back-end approach of validating `feeRecipient`, and anti-reentry processing has been done in the business logic.

CON-02 | PULL-OVER-PUSH PATTERN

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/MarketRegistry.sol: 36; contracts/MarketTokenTransferProxy.sol: 36; contracts/NFTMarket.sol: 78	● Acknowledged

Description

The change of `owner` by function `transferOwnership()` overrides the previously set `owner` with the new one without guaranteeing the new `owner` is able to actuate transactions on-chain.

Recommendation

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

The following code snippet can be taken as a reference:

```
address public potentialAdmin;

function transferAdmin(address pendingAdmin) external onlyAdmin {
    require(pendingAdmin != address(0), "potential admin can not be the zero
address.")
    potentialAdmin = pendingAdmin;
    emit AdminNominated(pendingAdmin);
}

function acceptAdmin() external {
    require(msg.sender == potentialAdmin, 'You must be nominated as potential admin
before you can accept administer role');
    admin = potentialAdmin;
    potentialAdmin = address(0);
    emit AdminChanged(admin)
}
```

Alleviation

[`CertiK`]: The dev team confirmed the risk and ensured that administrators will be cautious when operating, and there is currently no plan to change the owner. The team will make further behavior restrictions in future version upgrades.

CON-06 | LACK OF INPUT VALIDATION

Category	Severity	Location	Status
Volatile Code	● Minor	contracts/MerkleDistributor.sol: 65; contracts/NFTMarketWrap.sol: 207, 209	● Acknowledged

Description

MerkleDistributor.sol

The function `launchpad()` lacks the verification of `_startTime` , If `_startTime` is greater than `_endTime` , users will not be able to call the `claim()` function. Furthermore, the `_startTime` should be greater then current time.

NFTMarketWrap.sol

The length of the parameters `buySigs` and `sellSigs` should be checked as well.

Recommendation

We recommend adding more robust checks. For example:

```
require(_startTime > block.timestamp, "Start time is past");
require(_endTime > _startTime, "the start time can't be greater than the end
time!");
```

Alleviation

[`CertiK`]: The dev team explained the issue, the launchpad data itself is generated by centralized computation. There is already verified and complete data verification in the centralized computation to ensure business continuity. If the user cannot claim due to data errors, the centralized platform will regenerate a new round of launchpad information to provide users with claim services.

GLOBAL-02 | THIRD PARTY DEPENDENCY

Category	Severity	Location	Status
Volatile Code	● Minor		● Acknowledged

Description

The project is serving as the underlying entity to interact with one or more third party protocols(NFTs/ERC20s). The scope of the audit treats third party entities as black boxes and assume 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 dev team explained the issue and adopt recommendations to continuously monitor the third-party partner's status.

NFT-01 | NO UPPER LIMIT

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/NFTMarket.sol: 580, 591	● Acknowledged

Description

There are no upper boundaries for **changeMinimumMakerProtocolFee() && changeMinimumTakerProtocolFee()** which are used to set `minimumMakerProtocolFee` and `minimumTakerProtocolFee`. It is possible to set the total fee rate up to any arbitrary amount.

Recommendation

We recommend adding reasonable boundaries for the fees.

Alleviation

[`CertiK`]: The team has already discussed this issue. The users can see clear fee rate information when placing an order. At the same time, the system will calculate the required handling fee and related fee rate for the user when the user places an order. The user can view it when signing the transaction. The specific fee rate information of the transaction signed by the user, and the information cannot be changed after the user signs, so there is no possibility of arbitrary settings affecting the user's transaction.

NFT-02 | MISSING ZERO ADDRESS VALIDATION

Category	Severity	Location	Status
Volatile Code	● Minor	contracts/NFTMarket.sol: 559, 602, 1567	● Acknowledged

Description

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

```
559         exchangeWrap = _exchangeWrap;
```

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

```
602         protocolFeeRecipient = newProtocolFeeRecipient;
```

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

```
1567        protocolFeeRecipient = protocolFeeAddress;
```

- `protocolFeeAddress` 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

[`Certik`]: The dev team confirmed the issue. The `exchangeWrap` admin will do it with caution and there are currently no plans to change `exchangeWrap`. The `protocolFeeRecipient` is allowed to be set to zero address. The team will optimize this issue when upgrading to future version.

CON-04 | MISSING ERROR MESSAGES

Category	Severity	Location	Status
Coding Style	● Informational	contracts/MarketRegistry.sol: 37, 86, 129, 143, 202, 278, 293, 311, 331, 356, 391, 400, 417, 439, 451; contracts/MarketTokenTransferProxy.sol: 37, 86, 129, 143, 171, 282, 297, 315, 335, 360, 395, 404, 421, 443, 454; contracts/NFTMarket.sol: 70, 79, 131, 132, 355, 500, 501, 502, 503, 1677, 1691, 1719, 1745, 1820, 1835, 1853, 1873, 1898, 1933, 1942, 1959, 1981, 1992	● Acknowledged

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

[`CertiK`]: The dev team explained the issue and is considering the suggestion for future version optimizations.

CON-05 | MISSING EMIT EVENTS

Category	Severity	Location	Status
Coding Style	● Informational	contracts/MarketRegistry.sol: 125, 139, 154, 198; contracts/MarketTokenTransferProxy.sol: 125, 139, 154; contracts/NFTMarket.sol: 536, 543, 555, 565, 576, 587, 598, 1673, 1687, 1702	● 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

[`CertiK`]: The dev team explained the issue and is considering the suggestion for future version optimizations.

MRB-01 | POTENTIAL COMPILER ERROR

Category	Severity	Location	Status
Compiler Error	● Informational	contracts/MarketRegistry.sol: 3	● Acknowledged

Description

In Solidity versions 0.4.13 to 0.4.21, compiling the aforementioned code gives the following error:

- `ParserError: Expected identifier, got 'LParen'` .

Recommendation

It is recommended to modify the minimum Solidity version to 0.4.22. For example:

```
3  pragma solidity ^0.4.22;
```

Alleviation

[`Coresky`]: The dev team checked the issue and the code compiles fine.

OPTIMIZATIONS | CORESKY-AUDIT

ID	Title	Category	Severity	Status
CON-03	State Variable Should Be Declared Constant	Gas Optimization	Optimization	● Acknowledged

CON-03 | STATE VARIABLE SHOULD BE DECLARED CONSTANT

Category	Severity	Location	Status
Gas Optimization	● Optimization	contracts/MarketRegistry.sol: 117; contracts/MarketTokenTransferProxy.sol: 117; contracts/NFTMarket.sol: 1665	● Acknowledged

Description

State variables that never change should be declared as `constant` to save gas.

```
117       uint public DELAY_PERIOD = 2 weeks;
```

- `DELAY_PERIOD` should be declared `constant`.

```
117       uint public DELAY_PERIOD = 2 weeks;
```

- `DELAY_PERIOD` should be declared `constant`.

```
1665       uint public DELAY_PERIOD = 2 weeks;
```

- `DELAY_PERIOD` should be declared `constant`.

Recommendation

We recommend adding the `constant` attribute to state variables that never change.

Alleviation

[`CertiK`]: The dev team explained the issue and is considering the suggestion for future version optimizations.

APPENDIX | CORESKY-AUDIT

Finding Categories

Categories	Description
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.
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.
Compiler Error	Compiler Error findings refer to an error in the structure of the code that renders it impossible to compile using the specified version of the project.

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 | CERTIK

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.

CertiK | **Securing** the **Web3** World

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.