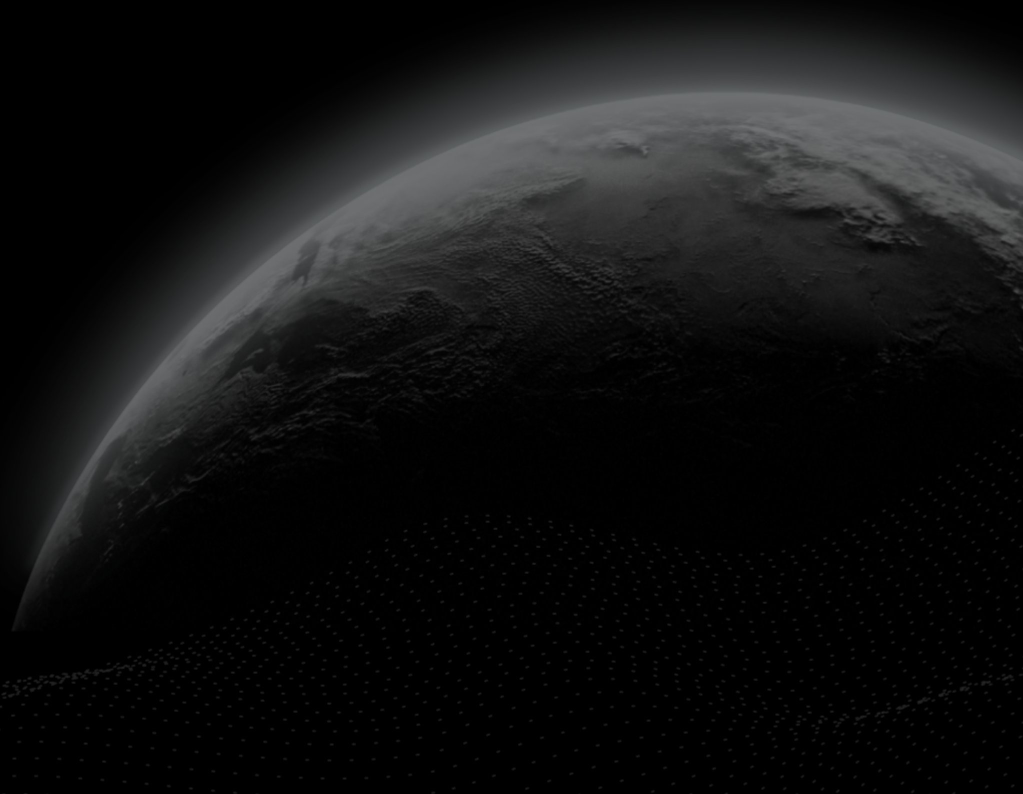
CERTIK

Security Assessment

Swapscanner - audit

CertiK Verified on Apr 6th, 2023



CertiK Verified on Apr 6th, 2023

Swapscanner - audit

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

Executive Summary

TYPES	ECOSYSTEM	METHODS
Service, Staking	Klaytn	Manual Review, Static Analysis

LANGUAGE	TIMELINE	KEY COMPONENTS
Solidity	Delivered on 04/06/2023	N/A

CODEBASE	COMMITS
https://github.com/Swapscanner/klaystaking-core/tree/main/contracts ...View All	4a46ac03c122204a928c0125b96ef116f52ad4db ...View All

Vulnerability Summary

5	5	0	0	0	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.
🟨	2	Medium	2 Resolved	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
🟨	3	Minor	3 Resolved	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.
🟦	0	Informational		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 | SWAPSCANNER - AUDIT

CODEBASE | SWAPSCANNER - AUDIT

Repository

https://github.com/Swapscanner/klaystaking-core/tree/main/contracts

Commit

4a46ac03c122204a928c0125b96ef116f52ad4db

AUDIT SCOPE | SWAPSCANNER - AUDIT

18 files audited ● 5 files with Resolved findings ○ 13 files without findings

ID	File	SHA256 Checksum
● CNK	CNStakedKLAYV1.sol	2c05d9ac391e5f45b35861e94167418240e86 6d84ce46f08efb7b5929ce7358d
● CNL	CNStakedKLAYV2.sol	a774253b3f2d50c6e19143365a131bb110c2e 2d393da34d206848df9246a4b11
● FCS	FeeCalculator.sol	74a058815576288b6625dec6cdcc85bb6c25b 916bf6fdea950d88f519a46b616
● PSK	ProxyStakedKLAY.sol	4ffd689488af2ac3ecfa4127fe6026ca4a990b4 97375a91261808333b56adc09
● PSA	ProxyStakedKLAYUnstakeable.sol	c9f45e089f10f91197826d20891b5c4cc3d835 b4539c6ff9135052c0019eac4a
○ CNI	cnstakinginterfaces/CNStakingInterface.sol	fd2df365a1a245d11c095b3400b675cf160c36 231f8a1ad5012172503cf89115
○ CNV	cnstakinginterfaces/CNStakingV1Interface.sol	63c833176e4644d4c9c6fe3ad1426f08ef474c a0207d9147eaac694fabcf4d8b
○ CSV	cnstakinginterfaces/CNStakingV2Interface.sol	e6c06bee5b571e0bab59567eeb2f9daa24608 a00006d5977e53846b27e663dd6
○ IPS	interfaces/IProxyStakedKLAY.sol	93b001d92f7df3da82bb671a002d562b21368 82ed0405a4dc2bb9f2c7e8fe0a8
○ IPK	interfaces/IProxyStakedKLAYClaimCheck.sol	4eba53d87717c83a70108646a168fce0401c1 7e883f17035999621df1cc503c0
○ ESS	libraries/EtherStrings.sol	55d5f822a7364bb39c3f2f5017cd78e35054ab 629a5f03183e893db11d21e574
○ FSB	libraries/Fonts.sol	01908b7e549aae269300b094273685f1ca6d3 5a421c4022657f1d89d5b503bfb
○ SMS	libraries/SharesMath.sol	301206164a042d10375f8e45d248732200509 d9db2882a4dac9901aa8a7bfefb
○ TSS	libraries/TimestampStrings.sol	70f3c166e00fe9722760300587c401e57688f4 e69f22549234cc8fd4813a2d75

ID	File	SHA256 Checksum
● ERC	ERC20ProgrammaticBalance.sol	6ad8691dbd196025706a1bf3c0b5e73d4806495ed889a48cd03a89f848640a71
● ERP	ERC20ProgrammaticBalanceStats.sol	2b42dcf1705ce5d400523cab767de26e0cf1037a542fe4917b1ce68f2e5f5b89
● ERV	ERC20VotesCustomBalance.sol	df3d257704f2af748a3368fb96663c86a335618959620e02bf3d04f9f5d06cc0
● PSL	ProxyStakedKLAYClaimCheck.sol	11c6c21bdf60027d7261a36fa5ae2e7fac14520ed2023555c059a16ad3a28344

APPROACH & METHODS | SWAPSCANNER - AUDIT

This report has been prepared for Swapscanner to discover issues and vulnerabilities in the source code of the Swapscanner - 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 | SWAPSCANNER - AUDIT

Description

In the contract `ProxyStakedKLAY` , the role _owner has authority over the function `setFee()` . Any compromise to the _owner account may allow the hacker to change the fee address and fee %.

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 recommend carefully managing 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., multi-signature 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.

Status/Alleviations

[Swapscanner team]: We acknowledge the risk related to the owner's authority over changing the fee-receiving address and fee percentage. We will address this issue by implementing your team's recommended short-term and long-term suggestions, such as introducing a timelock, multi-signature wallets, and a DAO/governance/voting module to enhance transparency and decentralization.

FINDINGS | SWAPSCANNER - AUDIT

5	0	0	2	3	0
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Swapscanner - audit. Through this audit, we have uncovered 5 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
FCS-03	Loss Of Precision And Rounding Inconsistency Could Cause Fee To Exceed MAX_FEE_PERCENTAGE	Logical Issue	Medium	● Resolved
GLOBAL-01	Out Of Scope Dependencies	Volatile Code	Medium	● Resolved
CON-01	Missing Receive Function	Logical Issue	Minor	● Resolved
PSA-01	Checks Effects Interaction Pattern Not Used	Control Flow, Volatile Code	Minor	● Resolved
PSK-01	Potential To Game Reward Payout If Reward Schedule Is Known And Non-Linear	Logical Issue	Minor	● Resolved

FCS-03 | LOSS OF PRECISION AND ROUNDING INCONSISTENCY COULD CAUSE FEE TO EXCEED MAX_FEE_PERCENTAGE

Category	Severity	Location	Status
Logical Issue	● Medium	FeeCalculator.sol: 46, 67	● Resolved

Description

In the `_calculateFee` function, its return value can exceed the `MAX_FEE_PERCENTAGE` restriction in some cases, due to rounding down of the `/` in line 46, and the rounding up in line 67.

Proof of Concept

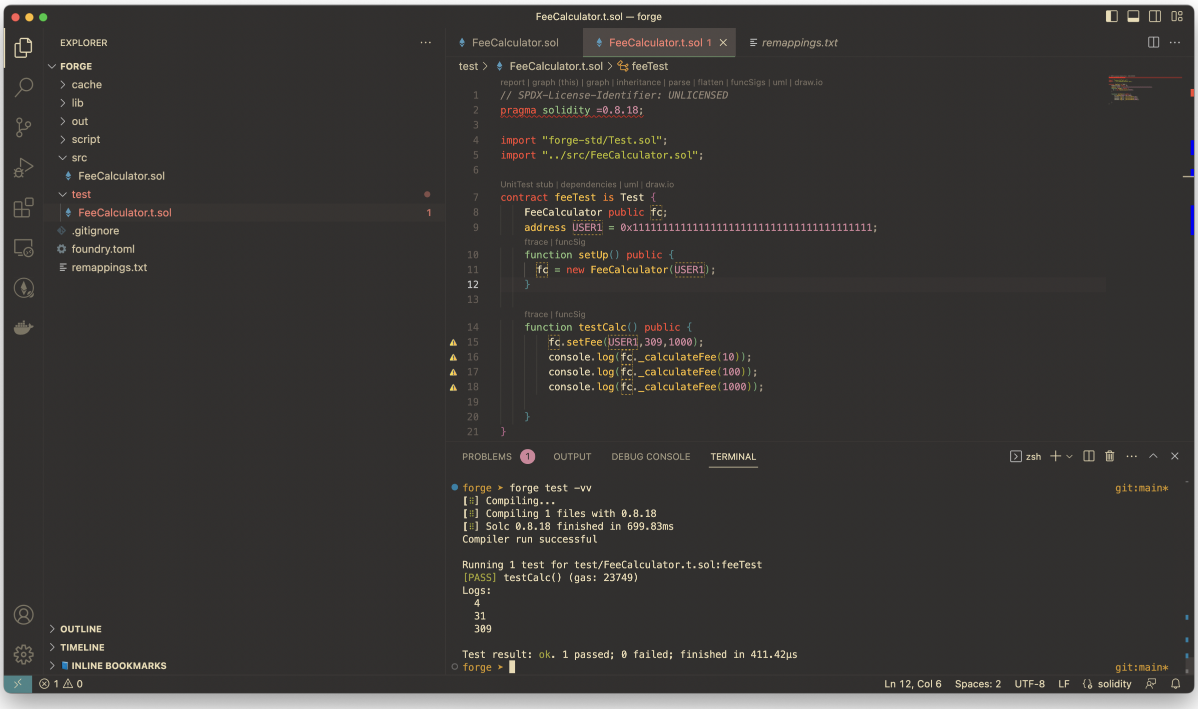
```
// SPDX-License-Identifier: UNLICENSED
pragma solidity =0.8.18;

import "forge-std/Test.sol";
import "../src/FeeCalculator.sol";

contract feeTest is Test {
    FeeCalculator public fc;
    address USER1 = 0x1111111111111111111111111111111111111111;
    function setUp() public {
      fc = new FeeCalculator(USER1);
    }

    function testCalc() public {
        fc.setFee(USER1,309,1000);
        console.log(fc._calculateFee(10));
        console.log(fc._calculateFee(100));
        console.log(fc._calculateFee(1000));

    }
}
```



```
// SPDX-License-Identifier: UNLICENSED
pragma solidity =0.8.18;

import "forge-std/Test.sol";
import "../src/FeeCalculator.sol";

contract feeTest is Test {
    FeeCalculator public fc;
    address USER1 = 0x1111111111111111111111111111111111111111;
    function setUp() public {
      fc = new FeeCalculator(USER1);
    }

    function testCalc() public {
        fc.setFee(USER1,309,1000);
        console.log(fc._calculateFee(10));
        console.log(fc._calculateFee(100));
        console.log(fc._calculateFee(1000));

    }
}
```

```
forge ➤ forge test -vv
[⠊] Compiling...
[⠊] Compiling 1 files with 0.8.18
[⠊] Solc 0.8.18 finished in 699.83ms
Compiler run successful

Running 1 test for test/FeeCalculator.t.sol:feeTest
[PASS] testCalc() (gas: 23749)
Logs:
  4
  31
  309

Test result: ok. 1 passed; 0 failed; finished in 411.42µs
forge ➤
```

Recommendation

We recommend ensuring that return value from calculateFee function does not exceed what the MAX_FEE_PERCENTAGE dictates. This can be done by conforming the treatment of rounding in line 46 and 67.

Alleviation

[Swapscanner team]: We have addressed this concern by ensuring that `_calculateFee()` always rounds down. Fixed in the following pull request: https://github.com/Swapscanner/klaystaking-core/pull/26

GLOBAL-01 | OUT OF SCOPE DEPENDENCIES

Category	Severity	Location	Status
Volatile Code	● Medium		● Resolved

Description

The scope of the audit does not include the folder `external` , which includes the staking contract that is an integral component of the system. The majority of in-scope contracts interact directly or indirectly with the staking contract in the `external` folder. Without reviewing the `external` folder, we are unable to verify the security and the behavior of key functions such as `stake()` , `unstake()` , `claim()` , etc. as they all depend on the implementation of the staking contracts in the `external` folder.

As an example, the amount that a user can claim is calculated via the `withdrawlRequestInfo()` function of the `ProxyStakedKLAYUnstakeable` contract, which calls the `getApprovedStakingWithdrawalInfo()` function of the out-of-scope `cnStaking` contract (as seen in line 54 of `CNStakingV1Interface`). If the latter function is implemented incorrectly, users might not be able to claim the KLAY token that he/she is eligible for.

Recommendation

We recommend a thorough review of the `external` folder which is an integral component of the entire system.

Alleviation

[Swapscanner team]: Our system no longer utilizes the CNStakingV1Interface, as it has been deemed deprecated. Instead, we have opted to exclusively employ the CNStakingV2Interface. The latter has undergone security audit, and the report is shown here: https://github.com/klaytn/governance-contracts-audit/tree/main/audit

CON-01 | MISSING RECEIVE FUNCTION

Category	Severity	Location	Status
Logical Issue	● Minor	CNStakedKLAYV1.sol: 14~20; CNStakedKLAYV2.sol: 14~20	● Resolved

Description

There is no `receive()` or `fallback()` function in the contracts, and thus the accrued KLAY token reward cannot be sent directly to the contract.

Recommendation

We'd like to understand if this is the intended design and the mechanism for reward accrual in these contracts, noting that the respective Mock contracts do contain the payable `receive()` functions.

Alleviation

[Swapscanner team]: The KLAY token rewards are generated by Klaytn through a direct increase in the balance of the reward address, as demonstrated in the following code snippets:

https://github.com/klaytn/klaytn/blob/243598f312ab6f1fb051c68fcb1ecf90eb842bbe/consensus/istanbul/backend/engine.go#L506 https://github.com/klaytn/klaytn/blob/243598f312ab6f1fb051c68fcb1ecf90eb842bbe/reward/reward_distributor.go#L111 https://github.com/klaytn/klaytn/blob/243598f312ab6f1fb051c68fcb1ecf90eb842bbe/blockchain/state/statedb.go#L450

The Klaytn team has confirmed that the reward-generation process does not require an explicit call to the receive() or fallback() functions, as the balance of the reward address is increased.

PSA-01 | CHECKS EFFECTS INTERACTION PATTERN NOT USED

Category	Severity	Location	Status
Control Flow, Volatile Code	● Minor	ProxyStakedKLAYUnstakeable.sol: 90~101	● Resolved

Description

In the `_processWithdrawalRequest()` function, the user / claimCheckOwner is sent its eligible native token before the `claimCheckTokenId` is burned. This represents a deviation from the "checks-effects-interaction" pattern, as the "effects" here (burning the claimCheckTokenId) is after the "interaction" (native token transfer). While reentrancy might be guarded by the out-of-scope staking contract as the comments from line 91-92 indicate, adherence to the "checks-effects-interaction" pattern is still considered best practice and helpful in preventing reentrancy risks.

Recommendation

We recommend strictly following the Checks-Effects-Interactions Pattern to avoid potential reentrancy issues

Alleviation

[Swapscanner team]: we have implemented a fix by burning the claimCheckTokenId before making the external call in the following pull request: https://github.com/Swapscanner/klaystaking-core/pull/27

PSK-01 | POTENTIAL TO GAME REWARD PAYOUT IF REWARD SCHEDULE IS KNOWN AND NON-LINEAR

Category	Severity	Location	Status
Logical Issue	● Minor	ProxyStakedKLAY.sol: 71~73, 82~101	● Resolved

Description

The amount of KLAY token that users can claim relative to other users is dependent on the relative "shares" of the user, as well as the total KLAY token pool including both the staked tokens and the reward tokens. If user A has knowledge that a large amount of reward will be added at a given block X, and the other users do not have such knowledge, then user A could benefit by staking a large amount of KLAY tokens immediately before block X, and thus inflating his share relative to other users. Subject to the unstaking / claiming waiting period, such a trade could be profitable for user A if the incremental reward is higher than the cost of capital (e.g. from borrowing the necessary KLAY tokens for a certain period of time)

Recommendation

We recommend using mechanism such as linear reward accrual to prevent users with asymmetric information to game reward payout.

Alleviation

[Swapscanner team]:

The KLAY rewards are generated during the finalization phase of each block created by the GC node (https://github.com/klaytn/klaytn/blob/243598f312ab6f1fb051c68fcb1ecf90eb842bbe/consensus/istanbul/backend/engine.go#L506). These rewards are relatively small compared to the interest for the one-week lockup period, as they are determined by the gas fee per block (which has a limit; https://github.com/klaytn/klaytn/blob/243598f312ab6f1fb051c68fcb1ecf90eb842bbe/blockchain/gaspool.go#L44-L46) in addition to the basic reward (https://github.com/klaytn/klaytn/blob/243598f312ab6f1fb051c68fcb1ecf90eb842bbe/consensus/istanbul/backend/engine.go#L473-L500).

When a user stakes their tokens, the sweep() function is called before the share calculation takes place. This step guarantees that users' shares are calculated accurately, even in cases where large rewards have accumulated and sweep() has not been triggered. As a result, the potential issue of users exploiting the system due to asymmetric information is effectively mitigated in our current implementation.

We have also implemented multiple measures to ensure the sweep() function is called frequently, such as automatic triggering during token transfers, stake, and unstake function calls. In addition, our team will manually call the sweep() function at regular, short intervals to further reduce the possibility of any exploitation.

APPENDIX | SWAPSCANNER - AUDIT

Finding Categories

Categories	Description
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.

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.