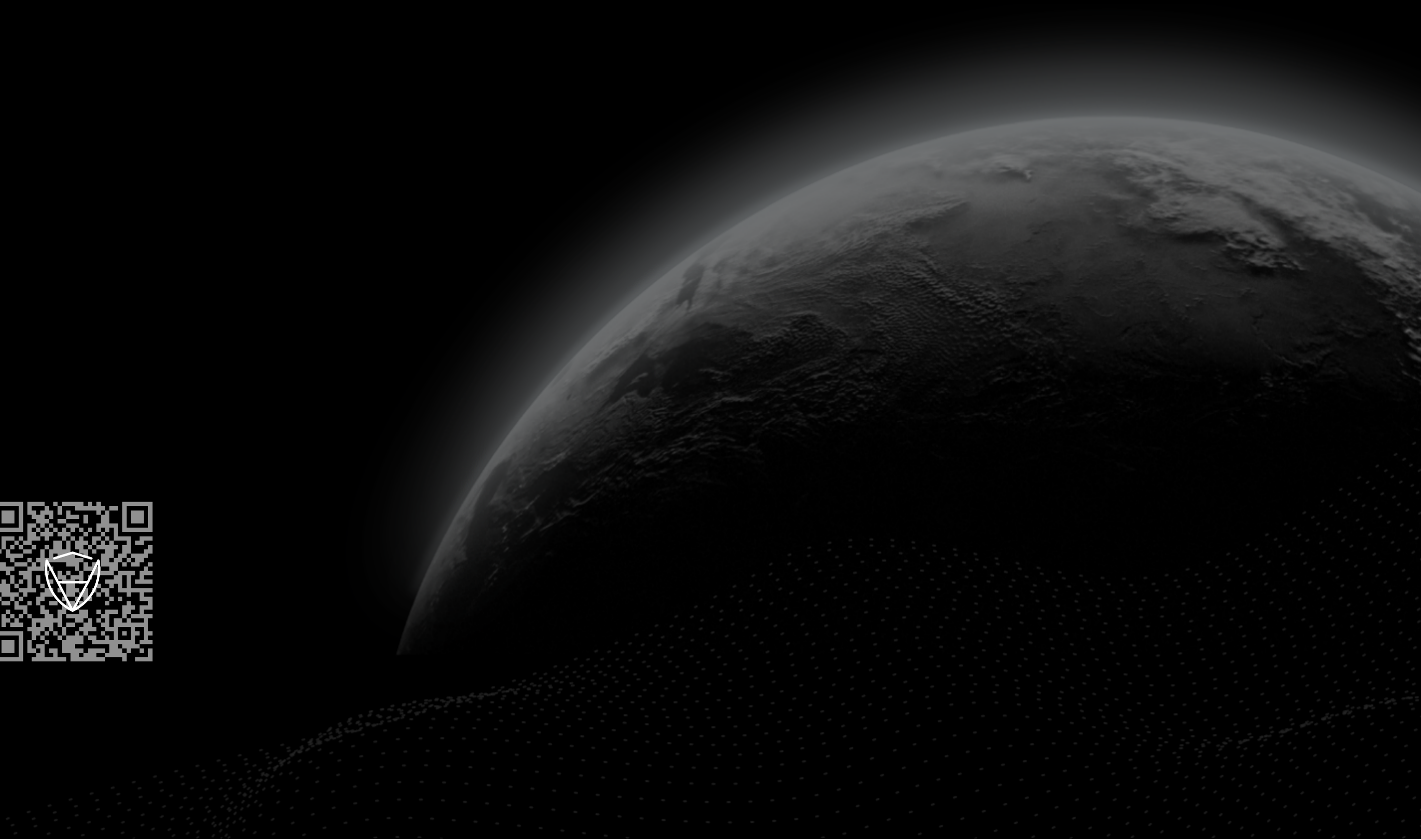
CERTIK

Security Assessment

Tinfun - Audit

CertiK Assessed on Jan 5th, 2024



CertiK Assessed on Jan 5th, 2024

Tinfun - Audit

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

Executive Summary

TYPES	ECOSYSTEM	METHODS
NFT	Ethereum (ETH)	Manual Review, Static Analysis

LANGUAGE	TIMELINE	KEY COMPONENTS
Solidity	Delivered on 01/05/2024	N/A

CODEBASE

https://github.com/TinfunDAO/TinfunContracts/tree/bf94d63ad3409e8a37267bf7c4add106fd6a6dad

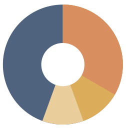
https://github.com/TinfunDAO/TinfunContracts/tree/a2a384a446847743

View All in Codebase Page

Highlighted Centralization Risks

ⓘ Contract upgradeability

Vulnerability Summary



9	5	0	0	4	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined

🟥	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.
🟧	3	Major	1 Resolved, 2 Acknowledged	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.
🟨	1	Medium	1 Resolved	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
▨	1	Minor	1 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.
🟦	4	Informational	3 Resolved, 1 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 | TINFUN - AUDIT

CODEBASE | TINFUN - AUDIT

Repository

https://github.com/TinfunDAO/TinfunContracts/tree/bf94d63ad3409e8a37267bf7c4add106fd6a6dad

https://github.com/TinfunDAO/TinfunContracts/tree/a2a384a4468477439697eca7e791b7318f65cc38

AUDIT SCOPE | TINFUN - AUDIT

1 file audited ● 1 file with Acknowledged findings

ID	Repo	File	SHA256 Checksum
● TRT	TinfunDAO/TinfunContracts	src/TinfunReserve.sol	17d948a9eb78874fd7dea0268bd90a76 b068fd594d65716510dc6e1778fc8620

APPROACH & METHODS | TINFUN - AUDIT

This report has been prepared for Tinfun to discover issues and vulnerabilities in the source code of the Tinfun - 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.

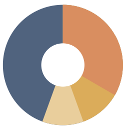
The auditing process pays special attention to the following considerations:

- 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.

FINDINGS | TINFUN - AUDIT



9	0	3	1	1	4
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Tinfun - 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
TRT-01	**Centralization Risks In TinfunReserve.Sol**	**Centralization**	**Major**	**Acknowledged**
TRT-02	**Centralized Control Of Contract Upgrade**	**Centralization**	**Major**	**Acknowledged**
TRT-03	Signature Replay Attack	Logical Issue	Major	Resolved
TRT-04	Signers Can Be Set To Zero Address Which Allows Arbitrary Data To Be Fraudulently Signed	Volatile Code	Medium	Resolved
TRT-05	Potential Cross-Chain Replay Attack	Logical Issue	Minor	Acknowledged
TRT-06	Missing Emit Events	Coding Style	Informational	Acknowledged
TRT-07	The Definition Of The Function `setReservePrice()` Does Not Match The Comment	Coding Style	Informational	Resolved
TRT-08	Incorrect Message	Coding Style	Informational	Resolved
TRT-09	Unused Error Messages	Coding Style	Informational	Resolved

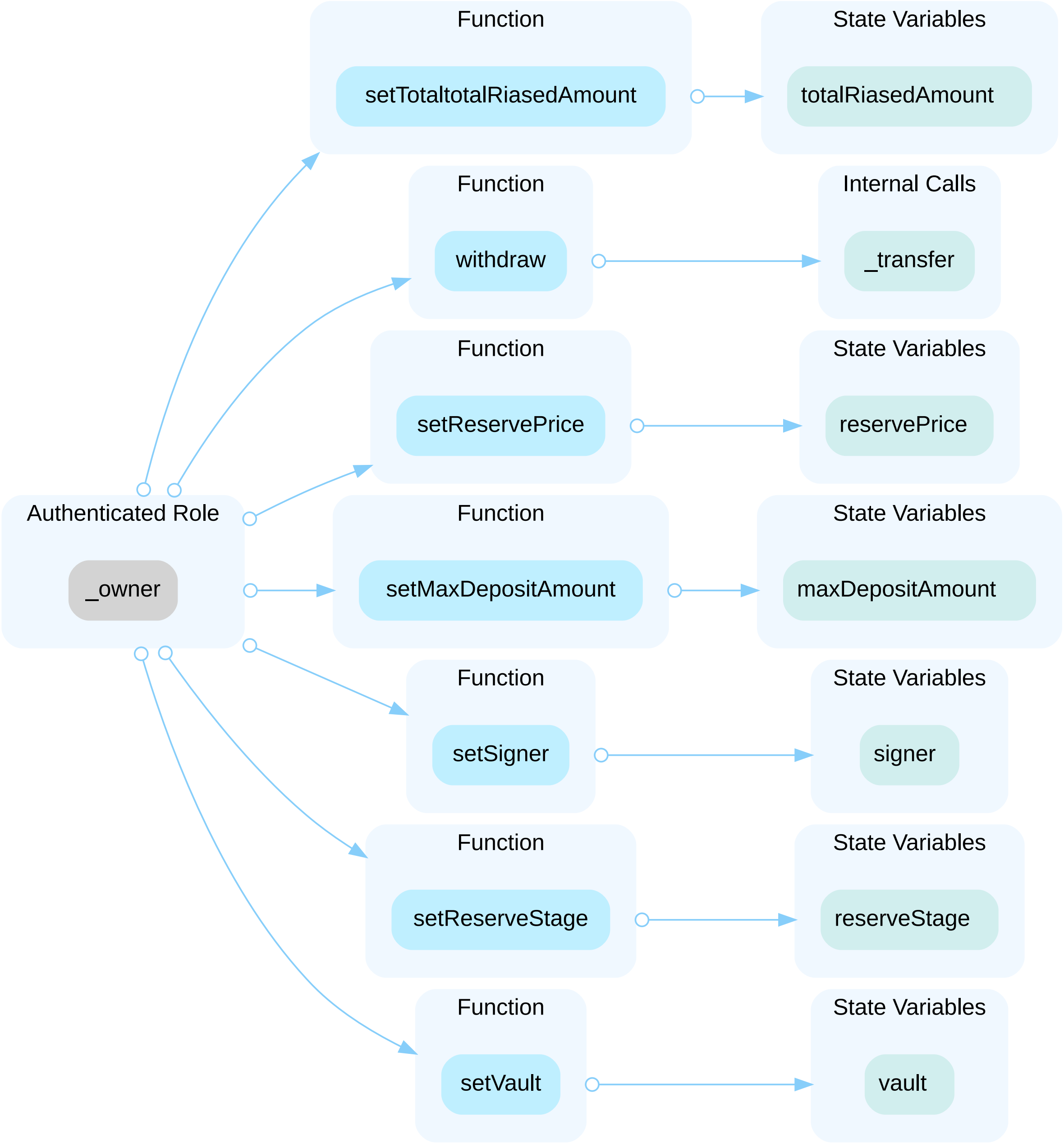
TRT-01 | CENTRALIZATION RISKS IN TINFUNRESERVE.SOL

Category	Severity	Location	Status
Centralization	● **Major**	**src/TinfunReserve.sol (v1): 154, 161, 168, 175, 184, 194, 201**	● **Acknowledged**

Description

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

- set a signer who can sign proof and sign any refund amount to any EOA
- set vault address
- set reserve stage
- set reserve price
- set totalRiasedAmount
- set maxDepositAmount
- withdraw ETH
- set a guardian who can sign proof and sign any public reserve to any EOA



In the contract `TinfunReserve` the role `signer` has authority to sign a signature, which can be used to call `whitelistReserve()` and `refund()` functions.

Any compromise to the `signer` account may allow the hacker to take advantage of this authority and sign a signature for an EOA to withdraw(`refund()`) any amount of ETH.

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets. Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

[Tinfun Team, 01/05/2024]: we will use multisign wallet to manage contract.

TRT-02 | CENTRALIZED CONTROL OF CONTRACT UPGRADE

Category	Severity	Location	Status
Centralization	● **Major**	**src/TinfunReserve.sol (v1): 13**	● **Acknowledged**

Description

In the contract `TinfunReserve` , the role `admin` of the proxy has the authority to update the implementation contract behind the proxy contract.

Any compromise to the `admin` account may allow a hacker to take advantage of this authority and change the implementation contract which is pointed by proxy and therefore execute potential malicious functionality in the implementation contract.

Recommendation

We recommend that the team make efforts to restrict access to the admin of the proxy contract. A strategy of combining a time-lock and a multi-signature (⅔, ⅗) wallet can be used to prevent a single point of failure due to a private key compromise. In addition, the team should be transparent and notify the community in advance whenever they plan to migrate to a new implementation contract.

Here are some feasible short-term and long-term suggestions that would mitigate the potential risk to a different level and suggestions that would permanently fully resolve the risk.

Short Term:

A combination of a time-lock and a multi signature (⅔, ⅗) wallet mitigate the risk by delaying the sensitive operation and avoiding a single point of key management failure.

- A time-lock with reasonable latency, such as 48 hours, for awareness of privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to a private key compromised;
  AND
- A medium/blog link for sharing the time-lock contract and multi-signers addresses information with the community.

For remediation and mitigated status, please provide the following information:

- Provide the deployed time-lock address.
- Provide the **gnosis** address with **ALL** the multi-signer addresses for the verification process.

- Provide a link to the **medium/blog** with all of the above information included.

Long Term:

A combination of a time-lock on the contract upgrade operation and a DAO for controlling the upgrade operation mitigate the contract upgrade risk by applying transparency and decentralization.

- A time-lock with reasonable latency, such as 48 hours, for community awareness of privileged operations;
  AND
- Introduction of a DAO, governance, or voting module to increase decentralization, transparency, and user involvement;
  AND
- A medium/blog link for sharing the time-lock contract, multi-signers addresses, and DAO information with the community.

For remediation and mitigated status, please provide the following information:

- Provide the deployed time-lock address.
- Provide the **gnosis** address with **ALL** the multi-signer addresses for the verification process.
- Provide a link to the **medium/blog** with all of the above information included.

Permanent:

Renouncing ownership of the `admin` account or removing the upgrade functionality can *fully* resolve the risk.

- Renounce the ownership and never claim back the privileged role;
  OR
- Remove the risky functionality.

Note: we recommend the project team consider the long-term solution or the permanent solution. The project team shall make a decision based on the current state of their project, timeline, and project resources.

Alleviation

[Tinfun Team, 01/05/2024]: we will use multisign wallet to manage contract.

TRT-03 | SIGNATURE REPLAY ATTACK

Category	Severity	Location	Status
Logical Issue	● Major	src/TinfunReserve.sol (v1): 104, 146, 234	● Resolved

Description

The functions `whitelistReserve()` and `refund()` lack access control, allowing EOA with signed data to execute them. The signed signature can be reused by `whitelistReserve()` and `refund()`, and a malicious EOA can exploit the signature data intended for `whitelistReserve()` and reuse it to execute the function `refund()`, forcing the victim to refund only a very small amount of tokens. The signed message hash should include the address of the contract to prevent the contract from being deployed multiple times.

Proof of Concept

```
    function test_refund() public {
        tinfunReserve.setReserveStage(TinfunReserve.ReserveStage.Reserve);
        address user = randomUsers[0];
        uint256 amount = 100;
        uint256 totalValue = amount * reservePrice;

        bytes memory signature = signMsg(
            keccak256(abi.encodePacked(user, amount)),
            signerPrivateKey
        );

        vm.prank(user, user);
        tinfunReserve.whitelistReserve{value: totalValue}(
            user,
            amount,
            signature
        );

        tinfunReserve.setReserveStage(TinfunReserve.ReserveStage.Refund);

        bytes memory refundSignature = signMsg(
            keccak256(abi.encodePacked(user, totalValue)),
            signerPrivateKey
        );

        vm.startPrank(user, user);

        tinfunReserve.refund(user, amount, signature);

        assertEq(tinfunReserve.refundStatus(user), true);
        assertEq(address(tinfunReserve).balance > 0 ether, true);

        vm.expectRevert(TinfunReserve.AlreadyRefunded.selector);
        tinfunReserve.refund(user, totalValue, refundSignature);

        vm.stopPrank();
    }
```

Recommendation

We recommend adding a nonce to the signature to avoid possible replay attacks.

Alleviation

The client confirmed the contract would only be deployed once. The client revised the code and resolved this issue in commit : a2a384a4468477439697eca7e791b7318f65cc38

TRT-04 | SIGNERS CAN BE SET TO ZERO ADDRESS WHICH ALLOWS ARBITRARY DATA TO BE FRAUDULENTLY SIGNED

Category	Severity	Location	Status
Volatile Code	● Medium	src/TinfunReserve.sol (v1): 81, 82, 154	● Resolved

Description

The setter function of the signer/guardian is missing the zero address check. If the signer/guardian is set as zero address, any malicious user is able to create a fraudulent signature that returns `address(0)` from ecrecover/recover function to bypass the signature check.

Recommendation

We recommend checking the new signer is not zero address.

Alleviation

The client revised the code and resolved this issue in commit : a2a384a4468477439697eca7e791b7318f65cc38

TRT-05 | POTENTIAL CROSS-CHAIN REPLAY ATTACK

Category	Severity	Location	Status
Logical Issue	● Minor	src/TinfunReserve.sol (v1): 241	● Acknowledged

Description

Signed messages are not properly verified with the current chain ID, thus allowing attackers to perform replay attacks across chains. Hardcoded or cached chain ID values are also vulnerable since a hard fork may occur and change the chain ID in the future.

Recommendation

We recommend verifying signed messages against the current chain ID by using `block.chainid` or `chainid()` within the same transaction.

Alleviation

[Tinfun Team, 01/05/2024]: only deploy on Ethereum mainnet.

TRT-06 | MISSING EMIT EVENTS

Category	Severity	Location	Status
Coding Style	● Informational	src/TinfunReserve.sol (v1): 154, 161, 168, 175, 184, 194	● Acknowledged

Description

There should always be events emitted in sensitive functions that are controlled by centralization roles.

Recommendation

It is recommended to emit events in sensitive functions that are controlled by centralization roles.

Alleviation

The client acknowledged this finding.

TRT-07 | THE DEFINITION OF THE FUNCTION `setReservePrice()` DOES NOT MATCH THE COMMENT

Category	Severity	Location	Status
Coding Style	● Informational	src/TinfunReserve.sol (v1): 165	● Resolved

Description

The function `setReservePrice()` is intended to set the reserve price, but the comment incorrectly states "Withdraw ETH to owner".

Recommendation

We recommend changing the comment to be consistent with the function name.

Alleviation

The client revised the code and resolved this issue in commit : 0f357cba3efc783bced43fe9bd499173622fbf03

TRT-08 | INCORRECT MESSAGE

Category	Severity	Location	Status
Coding Style	● Informational	src/TinfunReserve.sol (v1): 187	● Resolved

Description

In the function `setTotaltotalRiasedAmount()` , if `_totalRiasedAmount` is greater than `MAX_RESERVE_VALUE` , the error message should be `ExceedMaxReserveValue` instead of `InsufficientValue` .

Recommendation

We recommend that the misinformation be adjusted to correctly express intent.

Alleviation

The client revised the code and resolved this issue in commit : a2a384a4468477439697eca7e791b7318f65cc38

TRT-09 | UNUSED ERROR MESSAGES

Category	Severity	Location	Status
Coding Style	● Informational	src/TinfunReserve.sol (v1): 56, 58, 59	● Resolved

Description

The following error messages are not used anywhere in the contract:

- error NotPublic();
- error InvalidProof();
- error InvalidSigner();

Recommendation

We recommend removing these unused error messages or implementing them in the contract.

Alleviation

The client revised the code and resolved this issue in commit : a2a384a4468477439697eca7e791b7318f65cc38

APPENDIX | TINFUN - AUDIT

Finding Categories

Categories	Description
Coding Style	Coding Style findings may not affect code behavior, but indicate areas where coding practices can be improved to make the code more understandable and maintainable.
Volatile Code	Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases and may result in vulnerabilities.
Logical Issue	Logical Issue findings indicate general implementation issues related to the program logic.
Centralization	Centralization findings detail the design choices of designating privileged roles or other centralized controls over the code.

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.