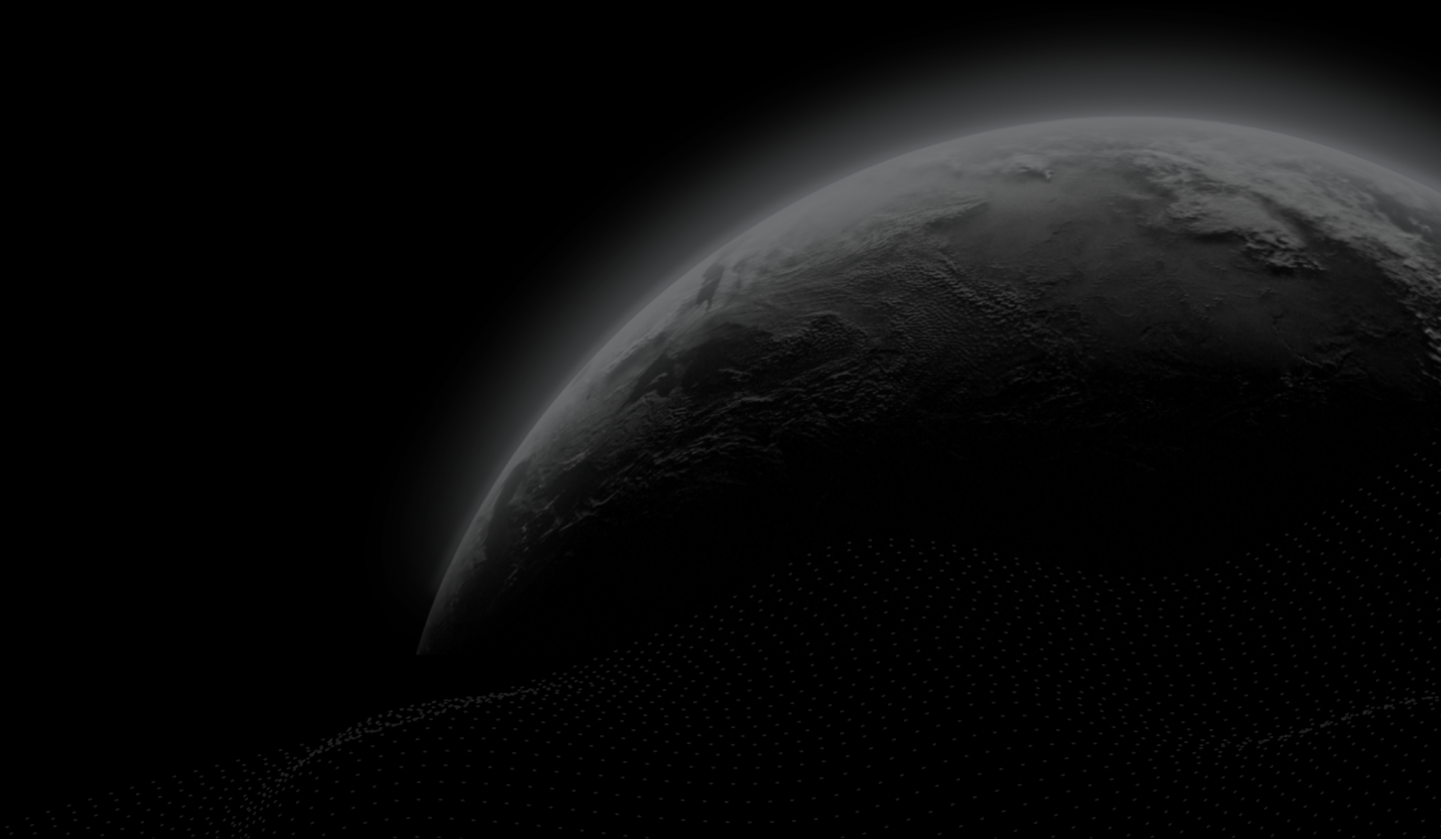
CERTIK

Security Assessment

Solv Yield - Bearing Tokens

CertiK Assessed on Apr 8th, 2024



CertiK Assessed on Apr 8th, 2024

Solv Yield - Bearing Tokens

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

Executive Summary

TYPES	ECOSYSTEM	METHODS
DeFi	Binance Smart Chain (BSC)	Manual Review, Static Analysis

LANGUAGE	TIMELINE	KEY COMPONENTS
Solidity	Delivered on 04/08/2024	N/A

CODEBASE

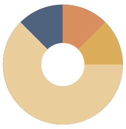
https://github.com/solv-finance/Solv-Yield-Bearing-Tokens/blob/30980706adc1a6e9b075f7f7f1c442159a318166/contracts/SftWrapRouter.sol

View All in Codebase Page

Highlighted Centralization Risks

ⓘ Contract upgradeability

Vulnerability Summary



8	6	0	0	2	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.
🟧	1	Major	1 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.
🟨	5	Minor	4 Resolved, 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.

■ 1	Informational	1 Resolved	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 | SOLV YIELD - BEARING TOKENS

CODEBASE | SOLV YIELD - BEARING TOKENS

Repository

https://github.com/solv-finance/Solv-Yield-Bearing-Tokens/blob/30980706adc1a6e9b075f7f7f1c442159a318166/contracts/SftWrapRouter.sol

AUDIT SCOPE | SOLV YIELD - BEARING TOKENS

1 file audited ● 1 file without findings

ID	Repo	File	SHA256 Checksum
● SWR	solv-finance/Solv-Yield-Bearing-Tokens	SftWrapRouter.sol	a8e2ef908ddc6c136aaba3da136875c4578be99101f711b1c09a76a4f36b38c8

APPROACH & METHODS | SOLV YIELD - BEARING TOKENS

This report has been prepared for SOLV to discover issues and vulnerabilities in the source code of the Solv Yield - Bearing Tokens 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.

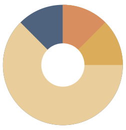
The auditing process pays special attention to the following considerations:

- 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 | SOLV YIELD - BEARING TOKENS



8	0	1	1	5	1
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Solv Yield - Bearing Tokens. Through this audit, we have uncovered 8 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
SWU-09	**Centralized Control Of Contract Upgrade**	**Centralization**	**Major**	● **Acknowledged**
SWR-07	Missing Validation On `swtAddress`	Access Control	Medium	● Resolved
SWR-03	Inherited Contracts Not Initialized In Initializer	Inconsistency	Minor	● Resolved
SWR-04	Unprotected Initializer	Coding Issue	Minor	● Resolved
SWR-05	Test Code Should Be Removed	Coding Issue	Minor	● Resolved
SWR-06	Out-Of-Scope Dependencies	Volatile Code	Minor	● Acknowledged
SWR-08	Inconsistent Support For The Native Token	Inconsistency	Minor	● Resolved
SWR-09	Non-Zero Amount Check Not Performed In The Correct Position	Coding Issue	Informational	● Resolved

SWU-09 | CENTRALIZED CONTROL OF CONTRACT UPGRADE

Category	Severity	Location	Status
Centralization	● **Major**	**SftWrapRouter.sol (31596e2): 17**	● **Acknowledged**

Description

The `SftWrapRouter` contract inherits upgradeable contracts, indicating that it is part of an upgradeable system. Upgradeable contracts often pair with a proxy contract that is responsible for managing contract upgrades. The privileged roles of the proxy often have the authority to update the implementation contract.

Any compromise to the privileged 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

[Solv Team, 04/04/2024]: 根据 DeFi 项目惯例，合约升级权限一般会在上线运行稳定后转给 Timelock、多签地址或投票。Admin 权限会在业务功能稳定后转为 Timelock 或投票模式。

[CertiK, 04/04/2024]: It is suggested to implement the aforementioned methods to avoid centralized failure. Also, CertiK strongly encourages the project team to periodically revisit the private key security management of all addresses related to centralized roles.

SWR-07 | MISSING VALIDATION ON `swtAddress`

Category	Severity	Location	Status
Access Control	● Medium	SftWrapRouter.sol (704ed11): 134	● Resolved

Description

The `swtAddress_` parameter of `unstake()` function is missing a check to ensure it is deployed by the `SftWrappedTokenFactory` , allowing users to specify an arbitrary address as the `swtAddress_` . Calling a user provided address is not safe, especially in a public function with no access control restriction. For example, if a non-standard ERC3525 token is transferred to the router without triggering `onERC3525Received` or `onERC721Received` , a malicious `swtAddress_` contract could be used to transfer the token to the attacker (L149).

Recommendation

Consider adding a check to ensure that the `swtAddress_` is the wrapped token deployed by `SftWrappedTokenFactory` .

Alleviation

[Solv Team, 04/04/2024]: The team heeded the advice and resolved the issue in commit: 2925dd9e864ef6dfd899e94d4ece12c3923b182d.

SWR-03 | INHERITED CONTRACTS NOT INITIALIZED IN INITIALIZER

Category	Severity	Location	Status
Inconsistency	● Minor	SftWrapRouter.sol (704ed11): 31	● Resolved

Description

Contract `SftWrapRouter` extends `ReentrancyGuardUpgradeable` , but the extended contract is not initialized by the current contract. Generally, the initializer function of a contract should always call all the initializer functions of the contracts that it extends.

Recommendation

We recommend initializing the `ReentrancyGuardUpgradeable` .

Alleviation

[Solv Team, 04/04/2024]: The team heeded the advice and resolved the issue in commit: 2925dd9e864ef6dfd899e94d4ece12c3923b182d.

SWR-04 | UNPROTECTED INITIALIZER

Category	Severity	Location	Status
Coding Issue	● Minor	SftWrapRouter.sol (704ed11): 17	● Resolved

Description

The `SftWrapRouter` contract does not protect their initializers. An attacker can call the initializer and assume ownership of the logic contract, whereby she can perform privileged operations that trick unsuspecting users into believing that she is the owner of the upgradeable contract.

Recommendation

We advise calling `_disableInitializers` in the constructor or giving the constructor the `initializer` modifier to prevent the initializer from being called on the logic contract.

Alleviation

[Solv Team, 04/04/2024]: The team heeded the advice and resolved the issue in commit: 2925dd9e864ef6dfd899e94d4ece12c3923b182d.

SWR-05 | TEST CODE SHOULD BE REMOVED

Category	Severity	Location	Status
Coding Issue	● Minor	SftWrapRouter.sol (704ed11): 15	● Resolved

Description

Test code should be removed prior to on-chain deployment:

```
15  import "../lib/forge-std/src/console. Sol";
```

Recommendation

We recommend removing the test code snippet.

Alleviation

[Solv Team, 04/04/2024]: The team heeded the advice and resolved the issue in commit: 2925dd9e864ef6dfd899e94d4ece12c3923b182d.

SWR-06 | OUT-OF-SCOPE DEPENDENCIES

Category	Severity	Location	Status
Volatile Code	● Minor	SftWrapRouter.sol (704ed11): 25	● Acknowledged

Description

The contract is serving as the underlying entity to interact with the out-of-scope contracts `IOpenFundMarket` and deposit tokens to the `IOpenFundMarket` . The scope of the audit treats these entities as black boxes and assumes their functional correctness. However, in the real world, external dependencies and out-of-scope contracts can be compromised and this may lead to lost or stolen assets.

Recommendation

We recommend that the project team constantly monitor the functionality of out-of-scope contracts and dependencies to mitigate any side effects that may occur when unexpected changes are introduced.

Alleviation

[Solv Team, 04/04/2024]: OpenFundMarket 合约由 solv 开发，并由第三方审计机构审计，审计报告：https://github.com/solv-finance/Audit/blob/main/Solv-v3/Solv-Protocol-Open-Fund_audit_report_2023-07-31%20-%20Salus.pdf

SWR-08 | INCONSISTENT SUPPORT FOR THE NATIVE TOKEN

Category	Severity	Location	Status
Inconsistency	● Minor	SftWrapRouter.sol (704ed11): 160	● Resolved

Description

The `createSubscription()` is a payable function, indicating it allows users to transfer native tokens to the router.

Line 164 of the contract calls the function `ERC20TransferHelper.doTransferIn()` to transfer native tokens to the router if `poolInfo.currency` being equal to `ETH_ADDRESS`. However, the router contact does not deposit the received native tokens to the designated contract `IOpenFundMarket(openFundMarket)` (L167), and the `IOpenFundMarket::subscribe()` is not a payable function.

Recommendation

Consider removing the `payable` from the `createSubscription()` if it is unintended to support native tokens.

Alleviation

[Solv Team, 04/04/2024]: The team heeded the advice and resolved the issue in commit: 2925dd9e864ef6dfd899e94d4ece12c3923b182d.

SWR-09 | NON-ZERO AMOUNT CHECK NOT PERFORMED IN THE CORRECT POSITION

Category	Severity	Location	Status
Coding Issue	● Informational	SftWrapRouter.sol (704ed11): 117	● Resolved

Description

The `stake()` function ensures the token amount to be wrapped cannot be zero:

```
117      require(amount_ > 0, "SftWrapRouter: stake amount cannot be 0");
```

Users can transfer tokens directly to the router without triggering the `stake()` function, thereby bypassing the zero-amount check.

Recommendation

Although the `SftWrappedToken` contract would perform a similar check to prevent wrapping zero amount of token, it is recommended to check the token value in `onERC3525Received` and `onERC721Received` if it is intended to emit the "SftWrapRouter: stake amount cannot be 0" error message by the router.

Alleviation

[Solv Team, 04/04/2024]: The team heeded the advice and resolved the issue in commit: 2925dd9e864ef6dfd899e94d4ece12c3923b182d.

OPTIMIZATIONS | SOLV YIELD - BEARING TOKENS

ID	Title	Category	Severity	Status
SWR-01	Unused Inheritances	Code Optimization	Optimization	● Acknowledged

SWR-01 | UNUSED INHERITANCES

Category	Severity	Location	Status
Code Optimization	● Optimization	SftWrapRouter.sol (704ed11): 17	● Acknowledged

Description

The `SftWrapRouter` inherits `AdminControlUpgradeable` and `GovernorControlUpgradeable` , which are initiated but never used. This could potentially lead to confusion and inefficiency in contract deployment and interaction.

Recommendation

Consider removing redundant inheritances.

Alleviation

[Solv Team, 04/04/2024]: The team acknowledged the finding and decided not to change the current codebase.

APPENDIX | SOLV YIELD - BEARING TOKENS

Finding Categories

Categories	Description
Coding Issue	Coding Issue findings are about general code quality including, but not limited to, coding mistakes, compile errors, and performance issues.
Access Control	Access Control findings are about security vulnerabilities that make protected assets unsafe.
Inconsistency	Inconsistency findings refer to different parts of code that are not consistent or code that does not behave according to its specification.
Volatile Code	Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases and may result in vulnerabilities.
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.