CERTIK

Autheo Smart Contract

Security Assessment

CertiK Assessed on Jul 17th, 2026

CertiK Assessed on Jul 17th, 2026

Autheo Smart Contract

The security assessment was prepared by CertiK.

Executive Summary

TYPES	ECOSYSTEM	METHODS
Vesting	Ethereum	Manual Review, Static Analysis

LANGUAGE	TIMELINE
Solidity	Preliminary comments published on 06/04/2026 Final report published on 07/17/2026

Vulnerability Summary

7	5	0	2	0
Total Findings	Resolved	Partially Resolved	Acknowledged	Declined

1	Centralization	1 Acknowledged	Centralization findings highlight privileged roles & functions and their capabilities, or instances where the project takes custody of users' assets.
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 may include logical errors that, under specific circumstances, could result in fund losses or loss of project control.
0	Medium		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.
3	Informational	2 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 | AUTHEO SMART CONTRACT

CODEBASE | AUTHEO SMART CONTRACT

Repository

https://github.com/0xAutheo/mainnet-smart-contracts

Commit

- 38c3f09fd2f0f7ed3269b6728532afae906b7a79
- 0f01bb7056476fde2afb8de29e99ec378e2b4517
- 31550490ec2fb36f2c0ccbbdb28eb15c088124ea

AUDIT SCOPE | AUTHEO SMART CONTRACT

0xAutheo/mainnet-smart-contracts
contracts/AutheoEcosystemVesting.sol
contracts/AutheoEcosystemVesting.sol

APPROACH & METHODS | AUTHEO SMART CONTRACT

This audit was conducted for Autheo to evaluate the security and correctness of the smart contracts associated with the Autheo Smart Contract project. The assessment included a comprehensive review of the in-scope smart contracts. The audit was performed using a combination of Manual Review and Static Analysis.

The review process emphasized the following areas:

- Architecture review and threat modeling to understand systemic risks and identify design-level flaws.
- Identification of vulnerabilities through both common and edge-case attack vectors.
- Manual verification of contract logic to ensure alignment with intended design and business requirements.
- Dynamic testing to validate runtime behavior and assess execution risks.
- Assessment of code quality and maintainability, including adherence to current best practices and industry standards.

The audit resulted in findings categorized across multiple severity levels, from informational to critical. To enhance the project's security and long-term robustness, we recommend addressing the identified issues and considering the following general improvements:

- Improve code readability and maintainability by adopting a clean architectural pattern and modular design.
- Strengthen testing coverage, including unit and integration tests for key functionalities and edge cases.
- Maintain meaningful inline comments and documentations.
- Implement clear and transparent documentation for privileged roles and sensitive protocol operations.
- Regularly review and simulate contract behavior against newly emerging attack vectors.

REVIEW NOTES | AUTHEO SMART CONTRACT

Overview

The **Autheo** vesting system provides native `THEO` allocation management through a single UUPS-upgradeable vesting contract.

- The contract manages predefined ecosystem allocation rounds using canonical `bytes32` round identifiers.
- Round parameters are stored on-chain through a data-driven `RoundConfig` model instead of being hardcoded as immutable tokenomics constants.
- Administrators register and seal round configurations before TGE, then create one vesting schedule per round.
- Funds flow into the contract through native `THEO` deposits via `receive()`, and vested amounts are released directly to beneficiaries using native-coin transfers.
- Each schedule supports an optional TGE unlock and daily linear vesting after a configured cliff.
- The contract tracks scheduled and released totals to expose solvency information and identify excess native `THEO` balance.

Core Contracts

AutheoEcosystemVesting

The contract implements a native `THEO` vesting module for Autheo ecosystem allocations. It supports twelve predefined allocation round identifiers, each configured by an administrator before TGE with a token cap, cliff duration, vesting period, and TGE unlock percentage. Once a round is sealed, the administrator can create a single beneficiary schedule for the full configured cap, using the round ID as the schedule ID. Beneficiaries or administrators can release TGE-unlocked amounts after TGE and release linearly vested amounts after the applicable cliff. The contract also provides an administrator-only batch release function for processing multiple schedules in one transaction. Native `THEO` is funded through plain deposits. The contract is upgradeable through the UUPS pattern and uses role-based access control for configuration, pausing, upgrades, and default administrative utilities.

External Dependencies

The **Autheo** protocol relies on a few external dependencies to fulfill the needs of its business logic.

The following are third-party dependency contracts used within the contract:

- @openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol
- @openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol
- @openzeppelin/contracts-upgradeable/proxy/utils/UUPSUpgradeable.sol
- @openzeppelin/contracts-upgradeable/utils/PausableUpgradeable.sol
- @openzeppelin/contracts-upgradeable/utils/ReentrancyGuardUpgradeable.sol
- @openzeppelin/contracts/utils/Address.sol

It is assumed that these functions are trusted and implemented properly within the whole project.

Additionally, it is important to note that the audit scope treats the native `THEO` coin and any deployment-level proxy infrastructure as external components outside the reviewed contract logic.

Privileged Functions

In the **Autheo** protocol, some privileged roles are adopted to ensure the dynamic runtime updates of the project, which are specified in the **Centralization Related Risks** finding.

The advantage of those privileged roles in the codebase is that the client reserves the ability to adjust the protocol according to the runtime required to best serve the community.

It is also worth noting the potential drawbacks of these functions, which should be clearly stated through the client's action/plan. Additionally, if the private keys of the privileged accounts are compromised, it could lead to devastating consequences for the project.

To improve the trustworthiness of the project, dynamic runtime updates in the project should be notified to the community. Any plan to invoke the aforementioned functions should also be considered to move to the execution queue of the `Timelock` contract.

FINDINGS | AUTHEO SMART CONTRACT

7	0	1	0	0	3	3
Total Findings	Critical	Centralization	Major	Medium	Minor	Informational

This report has been prepared for Autheo to identify potential vulnerabilities and security issues within the reviewed codebase. During the course of the audit, a total of 7 issues were identified. Leveraging a combination of Manual Review & Static Analysis the following findings were uncovered:

ID	Title	Category	Severity	Status
ASC-01	**Centralization Related Risks**	**Centralization**	**Centralization**	**Acknowledged**
ASC-02	Off-By-One Error Delays First Vesting Release Until One Day After Cliff Expiry	Logical Issue	Minor	Resolved
ASC-03	Discrepancy Between Lifecycle Diagram And On-Chain TGE Release Logic	Logical Issue	Minor	Resolved
ASC-07	Irrecoverable Vesting Schedule Due To Fixed Beneficiary	Logical Issue	Minor	Resolved
ASC-04	Single Reverting Beneficiary Can Block An Entire Batch Release	Denial of Service	Informational	Acknowledged
ASC-05	Misleading `registerRound` NatSpec Implies Updates Are Supported	Inconsistency	Informational	Resolved
ASC-06	Unused Custom Errors Declared But Never Reverted	Volatile Code	Informational	Resolved

ASC-01 | Centralization Related Risks

Category	Severity	Location	Status
Centralization	**Centralization**		**Acknowledged**

Description

In the contract `AutheoEcosystemVesting`, the role `UPGRADER_ROLE` has authority over the following functions:

- `_authorizeUpgrade()`

Any compromise to the `UPGRADER_ROLE` account may allow an attacker to authorize UUPS implementation upgrades and replace the contract logic with malicious code.

In the contract `AutheoEcosystemVesting`, the role `PAUSER_ROLE` has authority over the following functions:

- `pause()`
- `unpause()`

Any compromise to the `PAUSER_ROLE` account may allow an attacker to halt or resume scheduling and token release operations, disrupting the vesting process.

In the contract `AutheoEcosystemVesting`, the role `ADMIN_ROLE` has authority over the following functions:

- `registerRound()`
- `sealRound()`
- `scheduleVesting()`
- `releaseTGETokens()`
- `releaseVestingTokens()`
- `batchReleaseVestingTokens()`

Any compromise to the `ADMIN_ROLE` account may allow an attacker to configure round tokenomics before TGE, permanently seal incorrect parameters, assign vesting beneficiaries, and trigger token releases for schedules.

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

- `grantRole()`
- `revokeRole()`
- `setTgeTimestamp()`
- `rescueExcess()`

Any compromise to the `DEFAULT_ADMIN_ROLE` account may allow an attacker to grant or revoke all privileged roles, change the TGE timestamp before launch, and withdraw any balance considered excess by the contract.

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., 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.

Alleviation

[Autheo, 06/10/2026]:

We acknowledge the centralization finding. These roles are intentionally retained for pre-TGE setup, emergency operations, and upgrade governance.

In production, privileged roles should be assigned to controlled operational wallets, preferably multisig wallets, with signer keys protected by secure custody infrastructure. Where appropriate, sensitive operations may be routed through a timelock.

[CertiK, 07/17/2026]:

The team has added the following privileged functions in the commit: 31550490ec2fb36f2c0ccbbdb28eb15c088124ea.

In the contract `AutheoEcosystemVesting`, the role `ADMIN_ROLE` has authority over the following functions:

- `queueBeneficiaryUpdate()`
- `cancelBeneficiaryUpdate()`
- `executeBeneficiaryUpdate()`
- `unfreezeVestingRelease()`

Any compromise to the `ADMIN_ROLE` account may allow an attacker to initiate or execute beneficiary changes, freeze or unfreeze releases through the recovery flow.

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

- `setBeneficiaryUpdateApprover()`
- `setBeneficiaryUpdateApproverThreshold()`

Any compromise to the `DEFAULT_ADMIN_ROLE` account may allow an attacker to control the beneficiary-update approver set and quorum.

It is suggested to implement the aforementioned methods to mitigate centralized failure. Also, CertiK strongly encourages the project team to periodically revisit the private key security management of all addresses related to centralized roles.

ASC-02 Off-By-One Error Delays First Vesting Release Until One Day After Cliff Expiry

Category	Severity	Location	Status
Logical Issue	Minor	contracts/AutheoEcosystemVesting.sol (v1-38c3f09): 683~684	Resolved

Description

The vesting logic is documented to allow the first valid release *exactly* when the cliff ends. However, in `releaseVestingTokens(vestingScheduleId)`, a call made at `block.timestamp == theoTgeTimestamp + alloc.cliffSeconds` still computes `0` unlocked tokens and reverts with `NoTokensAvailableForRelease(vestingScheduleId)`. As a result, beneficiaries cannot claim at cliff expiry as promised and must wait an extra `DAY_IN_SECONDS` before the first release succeeds.

```
683         if (totalUnlocked <= alloc.releasedQuantity)
684             revert NoTokensAvailableForRelease(vestingScheduleId);
```

Recommendation

Align the cliff validation and unlock calculation so that the first release becomes claimable at the exact cliff expiry, as documented.

Alleviation

[Autheo, 06/09/2026]:

We confirm that the current behavior is intended. The configured cliff is a lockup period, and daily linear vesting begins after the cliff. At the exact cliff timestamp, zero full vesting days have elapsed, so no daily amount is claimable. The first daily installment becomes claimable one full day after cliff expiry.

The misleading comment has been corrected to match the implementation. Changes have been reflected in the commit hash: 0f01bb7056476fde2afb8de29e99ec378e2b4517.

ASC-03 Discrepancy Between Lifecycle Diagram And On-Chain TGE Release Logic

Category	Severity	Location	Status
Logical Issue	Minor	contracts/AutheoEcosystemVesting.sol (v1-38c3f09): 43, 54	Resolved

Description

The contract allows any schedule with `tgeUnlockBps > 0` to call `releaseTGETokens()` after TGE. In the 12-round setup, at least two rounds qualify: **Liquidity Res** (`tgeUnlockBps = 10_000` , 100% at TGE) and **Launch Related** (`tgeUnlockBps = 5_000` , 50% at TGE). The remaining rounds have `tgeUnlockBps = 0` and correctly revert with `NoTGEAllocation` if `releaseTGETokens` is called.

Some project documentation describes the post-TGE TGE-release step as **Liquidity Res only**. That wording does not match the on-chain rule. The open question is whether this is purely a documentation mistake, or whether the team intentionally treats Launch Related's TGE portion as a separate operational step that was omitted from the lifecycle diagram.

Recommendation

We would like to confirm this issue with the team. If the intended design only allows TGE unlocking for the `Liquidity Res` and `Launch Related` rounds, we recommend adding the corresponding restrictions in the related functions to ensure that actual execution aligns with the intended design.

Alleviation

[Autheo, 06/09/2026]:

We confirm this was a documentation issue. The intended on-chain rule is that `releaseTGETokens` applies to every scheduled round with `tgeUnlockBps > 0` .

Under current tokenomics, that includes `Liquidity Reserve` at 100% TGE unlock and `Launch Related` at 50% TGE unlock. Documentation has been updated accordingly.

Changes have been reflected in the commit hash: 0f01bb7056476fde2afb8de29e99ec378e2b4517.

ASC-07 | Irrecoverable Vesting Schedule Due To Fixed Beneficiary

Category	Severity	Location	Status
Logical Issue	Minor	contracts/AutheoEcosystemVesting.sol (v2-0f01bb7): 696, 791	Resolved

Description

Each round can only create one vesting schedule (`scheduleId == roundId`), and the beneficiary cannot be changed after `scheduleVesting` . If the beneficiary address cannot receive native THEO (for example, a contract without a payable `receive` / `fallback` function or a contract that reverts on receiving funds), all subsequent release operations for that schedule will fail.

As a result, the allocated TGE and vesting funds remain locked in the contract, with no built-in recovery mechanism available. Resolution would require the beneficiary to become compatible with native THEO transfers or require a contract upgrade.

Recommendation

Consider adding a controlled recovery mechanism, such as beneficiary migration or an alternative recipient, to prevent vesting schedules from becoming permanently unusable due to beneficiary limitations.

Alleviation

[Autheo, 07/17/2026]:

Issue acknowledged. Changes have been reflected in the commit hash: 31550490ec2fb36f2c0ccbbdb28eb15c088124ea.

[CertiK, 07/17/2026]:

The fix introduces an EIP-712 multi-signature based beneficiary change mechanism. The change request is initiated by `ADMIN_ROLE` . After the configured approvers reach the signature threshold, the request enters the queue. Once queued successfully, the release of the corresponding schedule is immediately frozen. After a 7-day waiting period, the change is executed, updating the beneficiary to the new address and unfreezing the schedule. If the change needs to be cancelled, it must be confirmed again through the approver multi-signature process. After cancellation, the release remains frozen, and additional signatures must be collected to call the unfreeze function before it can be resumed. This mechanism provides an on-chain recovery path for incorrect addresses or contract addresses that cannot receive funds while retaining governance control.

The `queue` , `cancel` , and `unfreeze` operations related to beneficiary changes share the same global EIP-712 nonce. After any operation is successfully executed, the nonce is incremented, invalidating other signatures collected based on the previous nonce. We understand this is an intended design used to serialize signature operations and prevent replay, rather than an implementation defect.

When processing beneficiary changes for multiple schedules in parallel, the team should pay attention to the order of signature collection and on-chain submission to avoid situations where already signed operations require signatures to be

collected again because another change operation is executed first.

ASC-04 Single Reverting Beneficiary Can Block An Entire Batch Release

Category	Severity	Location	Status
Denial of Service	Informational	contracts/AutheoEcosystemVesting.sol (v1-38c3f09): 794	Acknowledged

Description

`AutheoEcosystemVesting.batchReleaseVestingTokens` performs `payable(alloc.beneficiary).sendValue(releaseAmount)` inside its per-schedule loop. If any included `alloc.beneficiary` cannot accept native transfers or deliberately reverts, that transfer failure reverts the entire batch transaction and rolls back releases for other schedules in the same call. This creates a real denial of service for that specific batch execution. It does not, however, amount to a system-wide halt, because an admin can exclude the problematic schedule from later batches or use the single-schedule release paths.

Recommendation

Refactor the batch release flow to prevent a single failing beneficiary transfer from reverting the entire transaction. Prefer a pull-based payout model that records the releasable amount for each beneficiary and lets recipients withdraw separately, rather than pushing native value during batch processing.

Alleviation

[Autheo, 06/10/2026]:

We acknowledge that batch release is atomic and that one failing native transfer can revert that batch execution.

This is an accepted operational tradeoff for the admin batch helper. It does not create a system-wide halt because schedules can be omitted from later batches and releases can still be processed independently.

A pull-payment model may be considered in a future design if stronger isolation inside batch execution becomes a requirement.

ASC-05 Misleading `registerRound` NatSpec Implies Updates Are Supported

Category	Severity	Location	Status
Inconsistency	Informational	contracts/AutheoEcosystemVesting.sol (v1-38c3f09): 405	Resolved

Description

The `@notice` on `registerRound` states that the function is used to “Register (or update) a round's tokenomics configuration,” but the implementation does not allow updates. If `roundConfigs[roundId].isRegistered` is already true, the call reverts with `RoundAlreadyRegistered(roundId)`. The only way to set parameters for a round is a single pre-seal registration; after that, `sealRound` locks the config and `registerRound` cannot be called again for the same `roundId`. The accompanying `@dev` text correctly notes that once registered, the round cannot be updated, which contradicts the `@notice`.

This mismatch can mislead operators, auditors, or integrators into believing mistaken parameters can be corrected via a second `registerRound` call before sealing.

Recommendation

Review and correct the inconsistent NatSpec comments to accurately reflect that registration is one-time and updates are not supported.

Alleviation

[Autheo, 06/10/2026]:

We confirm that update-before-seal is intended business logic.

Changes were made, and `registerRound` now supports updates while the round remains unsealed and before TGE. Once `sealRound` is called, any further attempt to call `registerRound` for that round reverts with `RoundAlreadySealed`, preserving the immutability guarantee.

Changes have been reflected in the commit hash: 0f01bb7056476fde2afb8de29e99ec378e2b4517.

ASC-06 | Unused Custom Errors Declared But Never Reverted

Category	Severity	Location	Status
Volatile Code	Informational	contracts/AutheoEcosystemVesting.sol (v1-38c3f09): 292, 304	Resolved

Description

In `AutheoEcosystemVesting.sol` , two custom errors are defined in the errors block but are never used anywhere in the contract.

```
error InvalidAmount(uint256 amount);
error RoundCapExceeded(uint256 available, uint256 requested);
```

These are dead declarations that appear in the contract ABI but can never be emitted at runtime.

Recommendation

Consider removing unused code to maintain code clarity.

Alleviation

[Autheo, 06/10/2026]:

The unused custom errors have been removed from the contract, and stale documentation references were cleaned up.

Changes have been reflected in the commit hash: 0f01bb7056476fde2afb8de29e99ec378e2b4517.

APPENDIX | AUTHEO SMART CONTRACT

Finding Categories

Categories	Description
Centralization	Centralization findings detail the design choices of designating privileged roles or other centralized controls over the code.
Logical Issue	Logical Issue findings indicate general implementation issues related to the program logic.
Denial of Service	Denial of Service findings indicate that an attacker may prevent the program from operating correctly or responding to legitimate requests.
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.

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.

Elevate Your Web3 Journey

CertiK is the largest Web3 security platform combining formal verification with audits and comprehensive security solutions.