

SPARK Token audit

Security Assessment

CertiK Assessed on Oct 22nd, 2026





Certik Assessed on Oct 22nd, 2026

SPARK Token audit

The security assessment was prepared by Certik.

Executive Summary

TYPES
ERC-20

ECOSYSTEM
Polygon (POL)

METHODS
Manual Review, Static Analysis

LANGUAGE
Solidity

TIMELINE
Preliminary comments published on 10/16/2026
Final report published on 10/22/2026

Vulnerability Summary



6
Total Findings

0
Resolved

3
Multi-Sig

0
Partially Resolved

3
Acknowledged

0
Declined

3 Centralization

3 Multi-Sig

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 High

High 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.

1 Low

1 Acknowledged

Low 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.

2 Informational

2 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 | SPARK TOKEN AUDIT

Audit Summary

[Executive Summary](#)

[Vulnerability Summary](#)

[Codebase](#)

[Audit Scope](#)

[Approach & Methods](#)

Review Notes

[Overview](#)

[External Dependencies](#)

[Addresses](#)

[Privileged Functions](#)

Findings

[PSA-01 : Centralization Related Risks](#)

[PSA-02 : Initial Token Distribution](#)

[PSA-03 : Centralized Balance Manipulation](#)

[PSA-04 : `renounceOwnership` Succeeds Without Renouncing Ownership](#)

[PSA-05 : Missing Emit Events](#)

[PSA-06 : Inconsistent `transferFrom` Checks](#)

Appendix

Disclaimer

CODEBASE | SPARK TOKEN AUDIT

■ Repository

<https://github.com/SparkUSD/SparkToken>

■ Commit

[737548445a00787edd6279498b8627222c211354](#)

AUDIT SCOPE | SPARK TOKEN AUDIT

SparkUSD/SparkToken



SPUSD_Token_v1.sol

APPROACH & METHODS | SPARK TOKEN AUDIT

This audit was conducted for SPARK Token to evaluate the security and correctness of the smart contracts associated with the SPARK Token audit 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.
- 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 | SPARK TOKEN AUDIT

Overview

The **SPARK Token** project is an ERC20 token around SPUSD issuance and administrative token controls. In-scope components include the `SPUSD` token, which mints an initial supply to a constructor-specified recipient and thereafter supports holder transfers and burns, together with owner-controlled minting, pausing, blacklisting, and burning of blacklisted balances.

External Dependencies

In **SPARK Token**, the module inherits or uses a few of the depending injection contracts or addresses to fulfill the need of its business logic. The scope of the audit treats third party entities as black boxes and assume their functional correctness. However, in the real world, third parties can be compromised and this may lead to lost or stolen assets.

Addresses

The following addresses interact at some point with specified contracts, making them an external dependency. All of the following values are initialized either at deployment time or by specific functions in smart contracts.

SPUSD

- `supplyaddress`.

We assume these contracts or addresses are valid and non-vulnerable actors and implementing proper logic to collaborate with the current project.

Also, the following library/contract are considered as the third-party dependencies:

- <https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v4.9.6/>

Privileged Functions

In the **SPARK Token** project, privileged roles are used to support dynamic runtime updates. These roles are described in the `Centralization Related Risks`, `Centralized Balance Manipulation` and `Initial Token Distribution` findings.

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 be also considered to move to the execution queue of the `Timelock` contract.

FINDINGS | SPARK TOKEN AUDIT



6
Total Findings

0
Critical

3
Centralization

0
High

0
Medium

1
Low

2
Informational

This report has been prepared for SPARK Token to identify potential vulnerabilities and security issues within the reviewed codebase. During the course of the audit, a total of 6 issues were identified. Leveraging a combination of Manual Review & Static Analysis the following findings were uncovered:

ID	Title	Category	Severity	Status
PSA-01	Centralization Related Risks	Centralization	Centralization	● 2/3 Multi-Sig
PSA-02	Initial Token Distribution	Centralization	Centralization	● 2/3 Multi-Sig
PSA-03	Centralized Balance Manipulation	Centralization	Centralization	● 2/3 Multi-Sig
PSA-04	<code>renounceOwnership</code> Succeeds Without Renouncing Ownership	Logical Issue	Low	● Acknowledged
PSA-05	Missing Emit Events	Coding Style	Informational	● Acknowledged
PSA-06	Inconsistent <code>transferFrom</code> Checks	Inconsistency	Informational	● Acknowledged

PSA-01 | Centralization Related Risks

Category	Severity	Location	Status
Centralization	● Centralization	SPUSD_Token_v1.sol: 8, 27, 31, 35, 75, 80, 84, 96	● 2/3 Multi-Sig

Description

In the contract `SPUSD`, the role `owner`, initially assigned to the deployer, has authority over the following functions:

- `mint(address to, uint256 amount)`
- `toBlackList(address forBlock)`
- `fromBlackList(address forUnblock)`
- `burnBlackFunds(address account, uint256 amount)`
- `pause()`
- `unpause()`
- `transferOwnership(address newOwner)` (*inherited from Ownable2Step*)

If the owner's account is compromised, the attacker can mint tokens without a supply cap, dilute holders, control blacklist membership, burn balances of blacklisted accounts, pause or resume all token transfers, minting and burning, and nominate a replacement owner. These powers affect token supply, holder balances and transaction availability. Ownership alone does not authorize transferring existing holder balances without allowance or upgrading the contract implementation.

In the contract `SPUSD`, the role `pendingOwner`, nominated by the current owner, has authority over the following function:

- `acceptOwnership()` (*inherited from Ownable2Step*)

If the pending owner's account is compromised while its nomination remains active, the attacker can accept ownership and obtain every owner capability described above. Before acceptance, the pending-owner role alone cannot mint, manage the blacklist, burn other accounts' tokens, pause or unpause the contract, or nominate another owner.

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 (2/3, 3/5) 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.

I Alleviation

[CertiK, 10/21/2026]: As of **October 21, 2026**, the privileged role status of the contracts is as follows:

SPUSD

Address

0xE979926C0BF5Af833a98a0C2e9818e8B637F6d0A

Owner

- 0x46E074fA40495287aDea1442A2CF0c31670C7E7a - Safe Multi-Sig; 2/3 threshold multi-sig contract created by the official Safe Proxy Factory v1.4.1 at 0x4e1DCf7AD4e460CfD30791CCC4F9c8a4f820ec67

Safe contract owners:

- 0x6F8c35026324d3553cf60CDB32007453c7dBdD42
- 0xECE4f5Db5A3Ba6FD156E9648890a7c414c21252C
- 0x579FF1c2CA0A687809E9622E9A2c6e9b766e86b2

PSA-02 | Initial Token Distribution

Category	Severity	Location	Status
Centralization	● Centralization	SPUSD_Token_v1.sol: 11, 18	● 2/3 Multi-Sig

Description

All initial SPUSD tokens, totaling `initialsupply * 10**6` base units with six decimals, are minted to the single deployer-selected `supplyaddress`; the source does not specify its address or require it to be an EOA. This is a centralization risk because the deployer or the owner(s) of the EOAs can distribute tokens without obtaining the consensus of the community. Any compromise to these addresses may allow a hacker to steal and sell tokens on the market, resulting in severe damage to the project.

Recommendation

It is recommended that the team be transparent regarding the initial token distribution process. The token distribution plan should be published in a public location that the community can access. The team should make efforts to restrict access to the private keys of the deployer account or EOAs. A multi-signature (2/3, 3/5) wallet can be used to prevent a single point of failure due to a private key compromise. Additionally, the team can lock up a portion of tokens, release them with a vesting schedule for long-term success, and deanonymize the project team with a third-party KYC provider to create greater accountability.

Alleviation

[SPARK Token, 10/20/2026]: Link to the token distribution plan: <https://medium.com/@spusd/introducing-spark-token-stability-meets-speed-in-digital-payments-94283413bdd3>

Multi-sig wallet address: 0x46e074fa40495287adea1442a2cf0c31670c7e7a

Signer 1: 0x6f8c35026324d3553cf60cdb32007453c7dbdd42

Signer 2: 0xece4f5db5a3ba6fd156e9648890a7c414c21252c

Signer 3: 0x579ff1c2ca0a687809e9622e9a2c6e9b766e86b2

[CertiK, 10/21/2026]:

As of **October 21, 2026**, the largest token holder of the contracts is as follows:

SPUSD

Address

`0xE979926C0BF5Af833a98a0C2e9818e8B637F6d0A`

Owner / Largest Token Holder

- 0x46E074fA40495287aDea1442A2CF0c31670C7E7a - Safe Multi-Sig; 2/3 threshold multi-sig contract created by the official Safe Proxy Factory v1.4.1 at `0x4e1DCf7AD4e460CfD30791CCC4F9c8a4f820ec67`. Holds 4,788,100 SPUSD (47.881% of the 10,000,000 total supply) and is also the on-chain `owner()`.

Safe contract owners:

- `0x6F8c35026324d3553cf60CDB32007453c7dBdD42`
- `0xECE4f5Db5A3Ba6FD156E9648890a7c414c21252C`
- `0x579FF1c2CA0A687809E9622E9A2c6e9b766e86b2`

Additionally, the second-largest holder, `0xFef7a3ea4B29D324a1CAb80A6A0524289daBFca3`, holds 18.1110%.

PSA-03 | Centralized Balance Manipulation

Category	Severity	Location	Status
Centralization	● Centralization	SPUSD_Token_v1.sol: 27~29, 31~33, 75~78, 80~82, 84~86	● 2/3 Multi-Sig

Description

The owner can freeze any address via `toBlackList` and later destroy its balance via `burnBlackFunds`. Blacklisted addresses cannot `transfer` or `burn`, and `transferFrom` independently rejects blacklisted holders, recipients, and spenders, so existing approvals cannot be used as an exit. The owner can therefore freeze selected holders while team wallets, routers, and pools remain free to sell; blacklisting a pool itself can halt swaps and liquidity operations. `burnBlackFunds` then calls `_burn` without the holder's consent or allowance, permanently destroying user or pool balances. Unblacklisting restores transfer eligibility only and does not recover burned tokens. Moving ownership to a multisig or timelock reduces key-management risk but leaves these freeze-and-confiscation powers intact. In addition, the owner can mint tokens arbitrarily.

Recommendation

We recommend the team makes efforts to restrict access to the private key of the privileged account. A strategy of multi-signature (2/3, 3/5) 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 mint more tokens or engage in similar balance-related operations.

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 multi signature (2/3, 3/5) wallet *mitigate* the risk by avoiding a single point of key management failure.

- 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 **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 DAO for controlling the operation *mitigate* the risk by applying transparency and decentralization.

- Introduction of a DAO, governance, or voting module to increase decentralization, transparency, and user involvement;
AND
- A medium/blog link for sharing the multi-signers' addresses, and DAO information with the community.

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

- 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:

The following actions can *fully* resolve the risk:

- Renounce the ownership and never claim back the privileged role.
OR
- Remove the risky functionality.
OR
- Add minting logic (such as a vesting schedule) to the contract instead of allowing the owner account to call the sensitive function directly.

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.

I Alleviation

[CertiK, 10/21/2026]: As of **October 21, 2026**, the privileged role status of the contracts is as follows:

SPUSD

Address

`0xE979926C0BF5Af833a98a0C2e9818e8B637F6d0A`

Owner

- `0x46E074fA40495287aDea1442A2CF0c31670C7E7a` - Safe Multi-Sig; 2/3 threshold multi-sig contract created by the official Safe Proxy Factory v1.4.1 at `0x4e1DCf7AD4e460CfD30791CCC4F9c8a4f820ec67`

Safe contract owners:

- `0x6F8c35026324d3553cf60CDB32007453c7dBdD42`
- `0xECE4f5Db5A3Ba6FD156E9648890a7c414c21252C`

- `0x579FF1c2CA0A687809E9622E9A2c6e9b766e86b2`

PSA-04 | `renounceOwnership` Succeeds Without Renouncing Ownership

Category	Severity	Location	Status
Logical Issue	● Low	SPUSD_Token_v1.sol: 96	● Acknowledged

Description

`SPUSD` overrides `renounceOwnership` with an empty body. An owner call succeeds on-chain, but `owner` is not set to `address(0)` and no `OwnershipTransferred` event is emitted. Parties that treat a successful `renounceOwnership` transaction as proof of renunciation will wrongly assume privileged functions are disabled, while `mint`, `toBlackList`, `burnBlackFunds`, and `pause` remain available. If the intent is to disable renunciation, the function should revert rather than return successfully.

Recommendation

Revert in `renounceOwnership`, for example `revert("Ownable: renounceOwnership is disabled")`, instead of leaving an empty body that allows the call to succeed.

Alleviation

[SPARK Token, 10/20/2026]: Issue acknowledged. I won't make any changes for the current version.

PSA-05 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	● Informational	SPUSD_Token_v1.sol: 31~33, 35~37	● 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

[SPARK Token, 10/21/2026]: Issue acknowledged. I won't make any changes for the current version.

PSA-06 | Inconsistent `transferFrom` Checks

Category	Severity	Location	Status
Inconsistency	● Informational	SPUSD_Token_v1.sol: 47, 53~68	● Acknowledged

Description

The `transfer` function prohibits transfers to the zero address.

SPUSD_Token_v1.sol

```
39     function transfer(address to, uint256 amount)
40         public
41         virtual
42         override
43         returns (bool)
44     {
45         require(isBlocked[msg.sender] == false, "Your address is blacklisted!");
46     ;
47     @>   require(isBlocked[to] == false, "Recipient address is blacklisted!");
48         require(to != address(0), "ERC20: transfer to the zero address");
49         address owner = _msgSender();
50         _transfer(owner, to, amount);
51         return true;
52     }
```

SPUSD_Token_v1.sol

```
53     function transferFrom(
54         address from,
55         address to,
56         uint256 amount
57     ) public virtual override returns (bool) {
58         require(isBlocked[from] == false, "From address is blacklisted!");
59         require(isBlocked[to] == false, "Recipient address is blacklisted!");
60         require(
61             isBlocked[msg.sender] == false,
62             "Your address is blacklisted!"
63         );
64         address spender = _msgSender();
65         _spendAllowance(from, spender, amount);
66         _transfer(from, to, amount);
67         return true;
68     }
```

However, `transferFrom` has no such check, resulting in inconsistent transfer behavior.

I Recommendation

We recommend that the project team determine the intended consistent behavior for the token, then consider adding or removing the corresponding checks in both functions accordingly.

I Alleviation

[SPARK Token, 10/21/2026]: Issue acknowledged. I won't make any changes for the current version.

APPENDIX | SPARK TOKEN AUDIT

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.
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.
Inconsistency	Inconsistency findings refer to different parts of code that are not consistent or code that does not behave according to its specification.

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.

SPARK Token audit Security Assessment | CertiK Assessed on Oct 22nd, 2026 | Copyright © CertiK

