QuillAudits

Audit Report
November, 2024

For

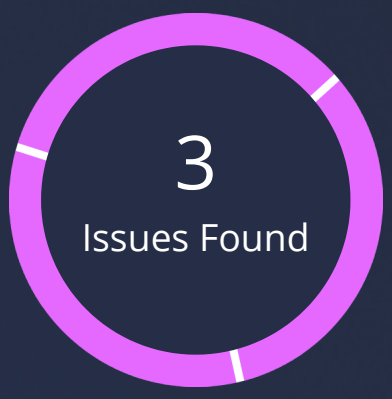
cSIGMA

Table of Content

Executive Summary

Project Name	cSigma Token
Project URL	https://csigma.finance/
Overview	cSigma is an ERC20-compliant token with upgradable and pausable features, designed for secure minting and role-based access control. It allows authorized roles to mint tokens, manage permissions, and pause operations to ensure controlled and compliant token management.
Audit Scope	The Scope of the Audit is to analyse the Security ,Code Quality of Sigma Contract.
Contracts In-Scope	contracts/token/Sigma.sol
Commit Hash	b71f385e5648f48de58788155c2641ddb97b6c8f
Language	Solidity
Blockchain	Ethereum, Arbitrum
Method	Manual Analysis, Functional Testing, Automated Testing
First Review	14th November 2024
Updated Code Received	15th November 2024
Second Review	15th November 2024
Fixed In	NA

Number of Security Issues per Severity



High
Medium
Low
Informational

	High	Medium	Low	Informational
Open Issues	0	0	0	0
Acknowledged Issues	0	0	0	3
Partially Resolved Issues	0	0	0	0
Resolved Issues	0	0	0	0

Checked Vulnerabilities

- Access Management
- Arbitrary write to storage
- Centralization of control
- Ether theft
- Improper or missing events
- Logical issues and flaws
- Arithmetic Correctness
- Race conditions/front running
- SWC Registry
- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- Exception Disorder
- Gasless Send
- Use of tx.origin
- Malicious libraries
- Compiler version not fixed
- Address hardcoded
- Divide before multiply
- Integer overflow/underflow
- ERC's conformance
- Dangerous strict equalities
- Tautology or contradiction
- Return values of low-level calls
- Missing Zero Address Validation
- Private modifier
- Revert/require functions
- Multiple Sends
- Using suicide
- Using delegatecall
- Upgradeable safety
- Using throw

Checked Vulnerabilities

- Using inline assembly
- Style guide violation
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

Throughout the audit of smart contracts, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments, match logic and expected behavior.
- Token distribution and calculations are as per the intended behavior mentioned in the whitepaper.
- Implementation of ERC standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods, and tools were used to review all the smart contracts.

Structural Analysis

In this step, we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

A static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption

In this step, we have checked the behavior of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit

Remix IDE, Foundry, Solhint, Mythril, Slither, Solidity statistic analysis.

Types of Severity

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below.

High Severity Issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

Medium Severity Issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

Low Severity Issues

Low-level severity issues can cause minor impact and are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are four severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Types of Issues

Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved

These are the issues identified in the initial audit and have been successfully fixed.

Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

Partially Resolved

Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.

Informational Issues

1. Unlimited Token Supply Risk

Path

contracts/token/Sigma.sol

Function

Mint

Description

The Sigma contract allows unlimited minting by accounts with the MINTER_ROLE, meaning tokens can be minted beyond the intended maximum supply of 1 billion tokens as stated in the whitepaper. This discrepancy could lead to excessive token issuance, impacting token value, investor trust, and protocol stability.

Recommendation

Implement a total supply cap of 1 billion tokens in the mint function by adding a check to prevent minting if the new supply would exceed this limit.

Csigma Team's Comment

The MINTER_ROLE is a highly restricted role, controlled by cSigma's governance. While the code does not enforce a hard cap, strict internal policies and procedures ensure no minting beyond the 1 billion token limit stated in the whitepaper. This design choice allows flexibility for future adjustments without requiring a contract redeployment. Community governance will oversee and regulate the total supply, ensuring transparency and accountability.
Thus, this issue does not pose a real risk to token holders as the governance structure effectively enforces the supply cap.

Status

Acknowledged

2. Pausable Core Token Functions

Path

contracts/token/Sigma.sol

Description

Key ERC20 functions, including approve, increaseAllowance, decreaseAllowance, and _beforeTokenTransfer, are pausable, allowing the contract owner or administrator to halt essential token operations. This could lead to situations where token transfers and trading are unexpectedly disabled, which may impact user experience and liquidity if not transparently managed. The ability to pause core token functions is atypical for ERC20 tokens and could limit token utility, especially for users and integrators who may expect uninterrupted transfer functionality.

Recommendation

Consider limiting the pause functionality to non-core functions or provide clear documentation and communication to token holders and integrators regarding the conditions and scenarios in which pausing will be applied. Alternatively, ensure that the pausing mechanism is decentralized or has additional governance controls to prevent misuse.

Csigma Team's Comment

The pause functionality is an integral security feature, intended to protect users and the ecosystem during emergencies (e.g., detecting an exploit or during contract upgrades). Pausing is not meant for arbitrary or frequent use. This capability is limited to predefined scenarios outlined in our governance policy. Moreover, the pausing feature is controlled through a multi-signature (multi-sig) setup, ensuring decentralized decision-making to prevent misuse.
Therefore, this functionality safeguards user assets and maintains protocol integrity without negatively affecting regular operations under normal circumstances.

Status

Acknowledged

3. Missing NatSpec Documentation

Path

contracts/token/Sigma.sol

Description

The Sigma contract lacks NatSpec (Ethereum Natural Language Specification Format) comments, which are critical for clear documentation, especially in contracts intended for public use. NatSpec comments provide essential descriptions for functions, parameters, and return values, improving readability for developers and auditors, and enhancing user experience with compatible interfaces (e.g., wallets or dApps). The absence of NatSpec could lead to misunderstandings about the functionality and usage of various functions, especially those involving permissions and role management.

Recommendation

Add NatSpec comments for all public and external functions, documenting the purpose of each function, the meaning of input parameters, expected behavior, and any potential restrictions. This will make the contract more accessible to developers, users, and auditors, while also facilitating better integration with developer tools and user interfaces.

Csigma Team's Comment

While NatSpec comments are not currently present in the contract, detailed external documentation accompanies the project, including the token's functionality, permissions, and use cases. This external documentation is readily available for developers and auditors. Additionally, we plan to include NatSpec comments in future updates to improve developer experience and compatibility with third-party tools.
This is a minor informational issue and does not impact the core functionality or security of the contract.

Status

Acknowledged

Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

Closing Summary

In this report, we have considered the security of cSigma Token. We performed our audit according to the procedure described above.

Some issues of informational severity were found. Some suggestions, gas optimizations and best practices are also provided in order to improve the code quality and security posture.

Disclaimer

QuillAudits Smart contract security audit provides services to help identify and mitigate potential security risks in cSigma Token. However, it is important to understand that no security audit can guarantee complete protection against all possible security threats. QuillAudits audit reports are based on the information provided to us at the time of the audit, and we cannot guarantee the accuracy or completeness of this information. Additionally, the security landscape is constantly evolving, and new security threats may emerge after the audit has been completed.

Therefore, it is recommended that multiple audits and bug bounty programs be conducted to ensure the ongoing security of cSigma Token. One audit is not enough to guarantee complete protection against all possible security threats. It is important to implement proper risk management strategies and stay vigilant in monitoring your smart contracts for potential security risks.

QuillAudits cannot be held liable for any security breaches or losses that may occur subsequent to and despite using our audit services. It is the responsibility of cSigma Token to implement the recommendations provided in our audit reports and to take appropriate steps to mitigate potential security risks.

About QuillAudits

QuillAudits is a leading name in Web3 security, offering top-notch solutions to safeguard projects across DeFi, GameFi, NFT gaming, and all blockchain layers. With six years of expertise, we've secured over 1000 projects globally, averting over $30 billion in losses. Our specialists rigorously audit smart contracts and ensure DApp safety on major platforms like Ethereum, BSC, Arbitrum, Algorand, Tron, Polygon, Polkadot, Fantom, NEAR, Solana, and others, guaranteeing your project's security with cutting-edge practices.

1000+
Audits Completed

$30B
Secured

1M+
Lines of Code Audited

Follow Our Journey

Audit Report
November, 2024

For

cSIGMA

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com

audits@quillhash.com