0x Guard

Smart contracts security assessment

Final report

Tariff: Simple

Sprocket

February 2022

0xguard.com

hello@0xguard.com

Contents

Introduction

The report has been prepared for the Sprocket team.

The audited code is available at @TS-Banking-Group/Sprocket Github repository and was audited after commit e0169e2.

Standard ERC20 token contract with access to mint functions only by owner. ERC20 interface is implemented with the use of OpenZeppelin libraries, which is considered the best practices.

Update: a recheck has been done after commit 82bcfcb.

Name	Sprocket
Audit date	2022-02-10 - 2022-02-10
Language	Solidity
Platform	Ethereum

Contracts checked

Name	Address
Sprocket Token	https://etherscan.io/token/0xa9364D93a4a6D8Dc51781236DeD2Dc4325b668B8

Procedure

We perform our audit according to the following procedure:

Automated analysis

- Scanning the project's smart contracts with several publicly available automated Solidity analysis tools
- Manual verification (reject or confirm) all the issues found by the tools

Manual audit

- Manually analyse smart contracts for security vulnerabilities
- Smart contracts' logic check

Known vulnerabilities checked

Title	Check result
Unencrypted Private Data On-Chain	passed
Code With No Effects	passed
Message call with hardcoded gas amount	passed
Typographical Error	passed
DoS With Block Gas Limit	passed
Presence of unused variables	passed
Incorrect Inheritance Order	passed
Requirement Violation	passed
Weak Sources of Randomness from Chain Attributes	passed
Shadowing State Variables	passed
Incorrect Constructor Name	passed
Block values as a proxy for time	passed
Authorization through tx.origin	passed
DoS with Failed Call	passed
Delegatecall to Untrusted Callee	passed
Use of Deprecated Solidity Functions	passed
Assert Violation	passed
State Variable Default Visibility	passed

Reentrancy	passed
Unprotected SELFDESTRUCT Instruction	passed
Unprotected Ether Withdrawal	passed
Unchecked Call Return Value	passed
Floating Pragma	passed
Outdated Compiler Version	passed
Integer Overflow and Underflow	passed
Function Default Visibility	passed

Classification of issue severity

High severity	High severity issues can cause a significant or full loss of funds, change of contract ownership, major interference with contract logic. Such issues require immediate attention.
Medium severity	Medium severity issues do not pose an immediate risk, but can be detrimental to the client's reputation if exploited. Medium severity issues may lead to a contract failure and can be fixed by modifying the contract state or redeployment. Such issues require attention.
Low severity	Low severity issues do not cause significant destruction to the contract's functionality. Such issues are recommended to be taken into consideration.

Issues

High severity issues

No issues were found

Medium severity issues

1. Open access to mint function (Sprocket Token)

Open access to the mint() function can lead to fraudulent activity with the owner or a compromised owner account, this is big risks for token users.

Recomendation: Use a multisig wallet and put it behind a Timelock contract by giving it owner rights. After this the severity of the issue may be lowered.

Update: The ownership of the token was transferred to a 2/2 multisig wallet.

Low severity issues

1. Fixed unstable pragma version (Fixed) (Sprocket Token)

The version of Solidity is set to 0.8.0. We do not recommend using first major release versions of the compiler as it may not have bugfixes. See Solidity version changelog for more information.

Recommendation: Use for example the 0.8.6 version of the compiler.

Update: This issue was fixed in the update.

2. Not optimal access modifier (Sprocket Token)

The function mint(address,uint256) should be declared external to save gas on calling it.

3. Uses number instead of decimals() (Sprocket Token)

The constructor() uses a hardcoded number for decimals uint256(18) instead of using decimals() functions. This may lead to discrepancies if the token contract is updated and redeployed.

Conclusion

Sprocket Sprocket Token contract was audited. 1 medium, 3 low severity issues were found.

Update: 1 low severity severity issue was fixed in the update. The ownership of the token was transferred to a 2/2 multisig wallet (transaction) which mitigates risks of the owner's account being compromised.

Disclaimer

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 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 without 0xGuard prior written consent.

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 0xGuard 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.

Slither output

Context._msgData() (SprocketToken.sol#98-101) is never used and should be removed

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (SprocketToken.sol#3) allows old versions

Pragma version^0.8.0 (SprocketToken.sol#81) allows old versions

Pragma version^0.8.0 (SprocketToken.sol#103) allows old versions

Pragma version^0.8.0 (SprocketToken.sol#175) allows old versions

Pragma version^0.8.0 (SprocketToken.sol#202) allows old versions

Pragma version^0.8.0 (SprocketToken.sol#505) allows old versions

Pragma version^0.8.0 (SprocketToken.sol#546) allows old versions

solc-0.8.11 is not recommended for deployment

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Redundant expression "this (SprocketToken.sol#99)" inContext (SprocketToken.sol#93-102)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Variable ERC20._totalSupply (SprocketToken.sol#236) is too similar to MyToken.constructor(string,string,uint256).totalSupply_ (SprocketToken.sol#555)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar

renounceOwnership() should be declared external:

- Ownable.renounceOwnership() (SprocketToken.sol#153-156)

transferOwnership(address) should be declared external:

- Ownable.transferOwnership(address) (SprocketToken.sol#162-166)

name() should be declared external:

- ERC20.name() (SprocketToken.sol#258-260)

symbol() should be declared external:

- ERC20.symbol() (SprocketToken.sol#266-268)

decimals() should be declared external:

- ERC20.decimals() (SprocketToken.sol#283-285)

totalSupply() should be declared external:

- ERC20.totalSupply() (SprocketToken.sol#290-292)

balanceOf(address) should be declared external:

- ERC20.balanceOf(address) (SprocketToken.sol#297-299)

transfer(address,uint256) should be declared external:

- ERC20.transfer(address,uint256) (SprocketToken.sol#309-312)

approve(address,uint256) should be declared external:

- ERC20.approve(address,uint256) (SprocketToken.sol#328-331)

transferFrom(address,address,uint256) should be declared external:

- ERC20.transferFrom(address,address,uint256) (SprocketToken.sol#346-354)

increaseAllowance(address,uint256) should be declared external:

- ERC20.increaseAllowance(address,uint256) (SprocketToken.sol#368-371)

decreaseAllowance(address,uint256) should be declared external:

- ERC20.decreaseAllowance(address,uint256) (SprocketToken.sol#387-393)

burn(uint256) should be declared external:

- ERC20Burnable.burn(uint256) (SprocketToken.sol#519-521)

burnFrom(address,uint256) should be declared external:

- ERC20Burnable.burnFrom(address,uint256) (SprocketToken.sol#534-541)

mint(address,uint256) should be declared external:

- MyToken.mint(address,uint256) (SprocketToken.sol#566-568)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external

0x Guard