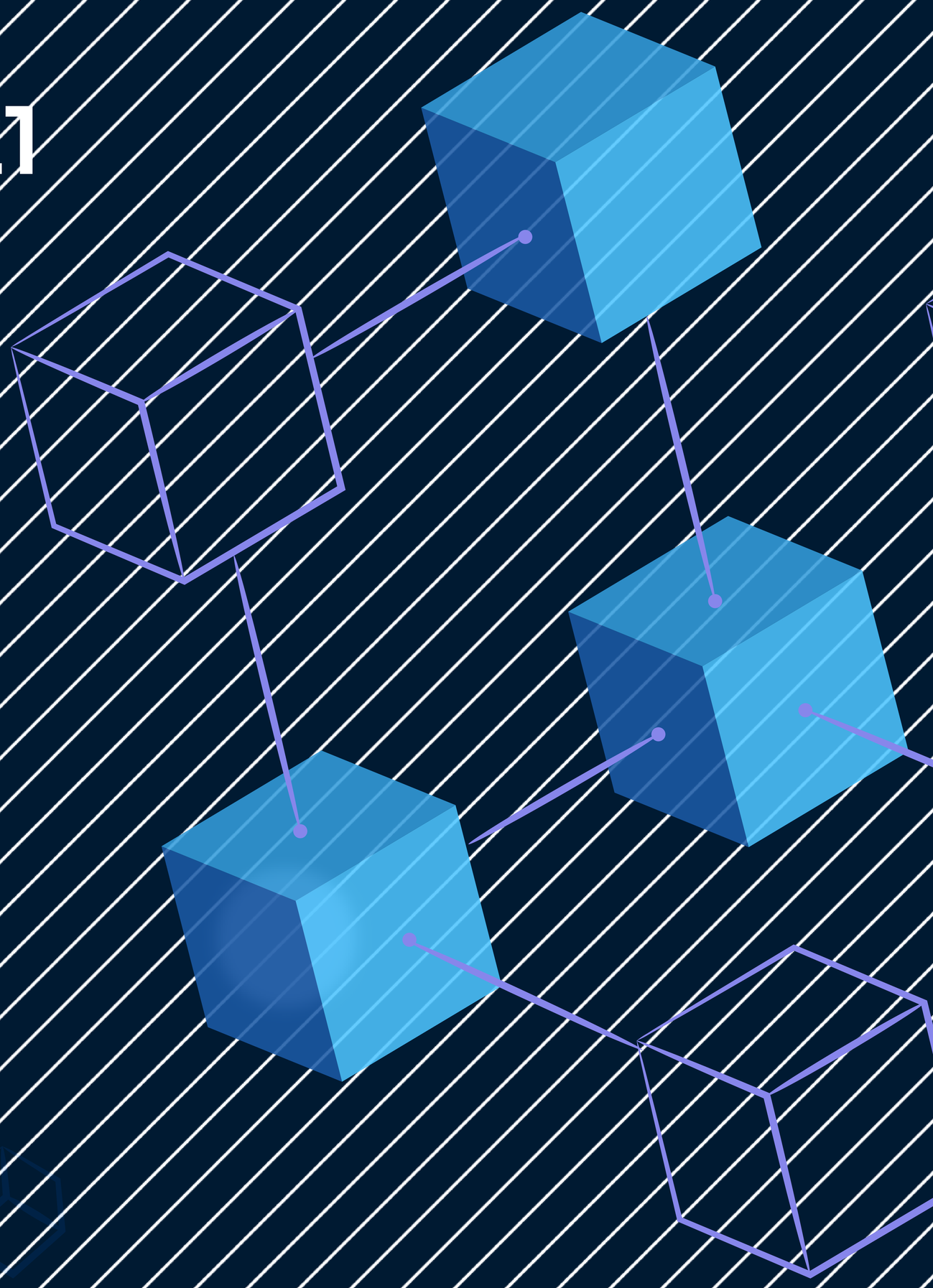
QuillAudits

Audit Report
November, 2021



For

DIVINER PROTOCOL

Contents

Scope of Audit	01
Check Vulnerabilities	01
Techniques and Methods	02
Issue Categories	03
Number of security issues per severity.	03
Introduction	04
A. Contract - Diviner Token	05
High Severity Issues	05
Medium Severity Issues	05
Low Severity Issues	05
Informational Issues	05
Functional Tests	06
Automated Tests	08
Closing Summary	11

Scope of the Audit

The scope of this audit was to analyze and document the Diviner Token Token smart contract codebase for quality, security, and correctness.

Checked Vulnerabilities

We have scanned the smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that we considered:

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

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

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token 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 analysed 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
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 analysed, 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 behaviour 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, Truffle, Truffle Team, Solhint, Mythril, Slither, Solidity statistic analysis, Theo.

Issue Categories

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.

Risk-level	Description
High	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	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	Low-level severity issues can cause minor impact and or 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 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.

Number of issues per severity

Type	High	Medium	Low	Informational
Open	0	0	0	0
Acknowledged	0	0	0	0
Closed	0	0	0	0

Introduction

During the period of November 18, 2021 to November 22, 2021 - QuillAudits Team performed a security audit for Diviner Token smart contracts.

The code for the audit was taken from following the official link: https://github.com/diviner-protocol/contracts

Note	Date	Commit hash
Version 1	November 18	e0964a6939473234b7e6bd0b8c088ee51a190fa9

Issues Found

A. Contract – Diviner Token

High severity issues

No issues were found.

Medium severity issues

No issues were found.

Low severity issues

No issues were found.

Informational issues

No issues were found.

Functional test

Evn: Ganache

Function name	Input	Output	TX	Status
mint	"0xA375ed96769CE0d4EDC8EF47f4C7166e3e30A718","1000000"	true	0xea95950dd07f62323190bbabddeea6d2a6692e0e003ef768299e74a992bc16df	Passed
burn	1000000	true	0xf0f518c7a8cac8a8b6c5c6fa0aa921528d5950fcf086785a231f2ff87751daa0	Passed
balanceOf	0xA375ed96769CE0d4EDC8EF47f4C7166e3e30A718	true	call0xA375ed96769CE0d4EDC8EF47f4C7166e3e30A7180xe1dBb549d65397302C6cEB45FEad46255Bd143B40x70a08231000000000000000000000000a375ed96769ce0d4edc8ef47f4c7166e3e30a718	Passed
transfer	"0xA375ed96769CE0d4EDC8EF47f4C7166e3e30A718","1000000"	true	0xab4405bfcc9fc184ac04d14d5963438891fd4a6fd56033fb329dcfb04a4a6130	Passed
approve	"0xA375ed96769CE0d4EDC8EF47f4C7166e3e30A718","1000000"	true	0x7d10e4b4bacb417b52cf7bd78eb659e58bbd24e5ffbdebd3089ab3778fa59915	Passed

allowance	"0xBC9aA9C9F90a4f95738a5A2b1794219680F49013","0xA375ed96769CE0d4EDC8EF47f4C7166e3e30A718"	1000000	call0xBC9aA9C9F90a4f95738a5A2b1794219680F490130xe1dBb549d65397302C6cEB45FEad46255Bd143B40xdd62ed3e000000000000000000000000bc9aa9c9f90a4f95738a5a2b1794219680f4901300000000000000000000000000000000a375ed96769ce0d4edc8ef47f4c7166e3e30a718	Passed
decreaseAllowance	"0xA375ed96769CE0d4EDC8EF47f4C7166e3e30A718","500000"	true	0x520a8cc71de3a1fae661f2f8944755613472dc1773fbc3540318a0b6fdd08eb2	Passed
increaseAllowance	"0xA375ed96769CE0d4EDC8EF47f4C7166e3e30A718","500000"	true	0x25bc6dfdea610cd2ca626452b2128c14b1ae218418cdec2c0e9a29503415ccc6	Passed
transferFrom	"0xBC9aA9C9F90a4f95738a5A2b1794219680F49013","0xA375ed96769CE0d4EDC8EF47f4C7166e3e30A718","1000000"	true	0x5ec63d3538b7ad5fa96b77b4192c2ac71d4933815049e1818becceb28e27ee85	Passed

Automated Tests

Slither

```
INFO:Detectors:
BEP20.constructor(string,string).name (token-dpt.sol#419) shadows:
        - BEP20.name() (token-dpt.sol#435-437) (function)
        - IBEP20.name() (token-dpt.sol#121) (function)
BEP20.constructor(string,string).symbol (token-dpt.sol#419) shadows:
        - BEP20.symbol() (token-dpt.sol#449-451) (function)
        - IBEP20.symbol() (token-dpt.sol#116) (function)
BEP20.allowance(address,address).owner (token-dpt.sol#487) shadows:
        - Ownable.owner() (token-dpt.sol#71-73) (function)
BEP20._approve(address,address,uint256).owner (token-dpt.sol#704) shadows:
        - Ownable.owner() (token-dpt.sol#71-73) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
DivinerToken.burn(uint256) (token-dpt.sol#774-777) should emit an event for:
        - _cap = _cap - (amount) (token-dpt.sol#775)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Ownable.constructor().msgSender (token-dpt.sol#63) lacks a zero-check on :
                - _owner = msgSender (token-dpt.sol#64)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Address.isContract(address) (token-dpt.sol#218-229) uses assembly
        - INLINE ASM (token-dpt.sol#225-227)
Address._functionCallWithValue(address,bytes,uint256,string) (token-dpt.sol#341-369) uses assembly
        - INLINE ASM (token-dpt.sol#361-364)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._functionCallWithValue(address,bytes,uint256,string) (token-dpt.sol#341-369) is never used and should be removed
Address.functionCall(address,bytes) (token-dpt.sol#276-281) is never used and should be removed
Address.functionCall(address,bytes,string) (token-dpt.sol#289-295) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (token-dpt.sol#308-320) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (token-dpt.sol#328-339) is never used and should be removed
Address.isContract(address) (token-dpt.sol#218-229) is never used and should be removed
Address.sendValue(address,uint256) (token-dpt.sol#247-256) is never used and should be removed
BEP20._burnFrom(address,uint256) (token-dpt.sol#721-728) is never used and should be removed
Context._msgData() (token-dpt.sol#32-35) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
```

```
INFO:Detectors:
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (token-dpt.sol#87-89)
name() should be declared external:
        - BEP20.name() (token-dpt.sol#435-437)
decimals() should be declared external:
        - BEP20.decimals() (token-dpt.sol#442-444)
symbol() should be declared external:
        - BEP20.symbol() (token-dpt.sol#449-451)
balanceOf(address) should be declared external:
        - BEP20.balanceOf(address) (token-dpt.sol#463-465)
transfer(address,uint256) should be declared external:
        - BEP20.transfer(address,uint256) (token-dpt.sol#475-482)
allowance(address,address) should be declared external:
        - BEP20.allowance(address,address) (token-dpt.sol#487-494)
approve(address,uint256) should be declared external:
        - BEP20.approve(address,uint256) (token-dpt.sol#503-510)
transferFrom(address,address,uint256) should be declared external:
        - BEP20.transferFrom(address,address,uint256) (token-dpt.sol#524-541)
increaseAllowance(address,uint256) should be declared external:
        - BEP20.increaseAllowance(address,uint256) (token-dpt.sol#555-565)
decreaseAllowance(address,uint256) should be declared external:
        - BEP20.decreaseAllowance(address,uint256) (token-dpt.sol#581-596)
mint(uint256) should be declared external:
        - BEP20.mint(uint256) (token-dpt.sol#606-609)
cap() should be declared external:
        - DivinerToken.cap() (token-dpt.sol#740-742)
mint(address,uint256) should be declared external:
        - DivinerToken.mint(address,uint256) (token-dpt.sol#765-767)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

```
INFO:Detectors:
Pragma version0.8.9 (token-dpt.sol#10) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (token-dpt.sol#247-256):
        - (success) = recipient.call{value: amount}() (token-dpt.sol#251)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (token-dpt.sol#341-369):
        - (success,returndata) = target.call{value: weiValue}(data) (token-dpt.sol#350-352)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter DivinerToken.mint(address,uint256)._to (token-dpt.sol#765) is not in mixedCase
Parameter DivinerToken.mint(address,uint256)._amount (token-dpt.sol#765) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (token-dpt.sol#33)" inContext (token-dpt.sol#23-36)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
DivinerToken.slitherConstructorVariables() (token-dpt.sol#737-778) uses literals with too many digits:
        - _cap = 1000000000e18 (token-dpt.sol#738)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
```

SOLHINT LINTER

```
token-dpt.sol
   10:1   error    Compiler version 0.8.9 does not satisfy the ^0.5.8 semver requirement                          compiler-version
   26:3   warning  Explicitly mark visibility in function (Set ignoreConstructors to true if using solidity >=0.7.0)  func-visibility
   26:17  warning  Code contains empty blocks                                                                     no-empty-blocks
   62:3   warning  Explicitly mark visibility in function (Set ignoreConstructors to true if using solidity >=0.7.0)  func-visibility
   95:5   warning  Error message for require is too long                                                          reason-string
  252:5   warning  Error message for require is too long                                                          reason-string
  334:5   warning  Error message for require is too long                                                          reason-string
  419:3   warning  Explicitly mark visibility in function (Set ignoreConstructors to true if using solidity >=0.7.0)  func-visibility
  532:5   warning  Error message for require is too long                                                          reason-string
  587:5   warning  Error message for require is too long                                                          reason-string
  630:5   warning  Error message for require is too long                                                          reason-string
  631:5   warning  Error message for require is too long                                                          reason-string
  636:5   warning  Error message for require is too long                                                          reason-string
  676:5   warning  Error message for require is too long                                                          reason-string
  681:5   warning  Error message for require is too long                                                          reason-string
  708:5   warning  Error message for require is too long                                                          reason-string
  709:5   warning  Error message for require is too long                                                          reason-string
  734:22  warning  Code contains empty blocks                                                                     no-empty-blocks
```

Results

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 the **Diviner Token** platform. We performed our audit according to the procedure described above.

No issues were found during the audit.

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of the **Diviner Token** platform. This audit does not provide a security or correctness guarantee of the audited smart contracts. The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the **Diviner Token** Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

Audit Report
November, 2021

For

DIVINER PROTOCOL

QuillAudits

Canada, India, Singapore, United Kingdom

audits.quillhash.com

audits@quillhash.com