DISCIPLINA
Project by TeachMePlease

Rock'n'Block

SMART CONTRACTS EXPERTISE — RIGHT WAY TO SUCCESS
Disciplina contracts audit report

If you have any questions concerning smart contract design and audit, feel free to contact audit@rocknblock.io

Content

1. Overview

Team of Disciplina asked us to perform a review of their ERC20 Token contract code. We performed a review of their code from 12.04.2021- 16.04.2021, and published this document as a write-up of our findings.

1.1 Terms of Reference for the creation of a smart contract

1.1.1 BEP20 Token details:

- Token Name - DISCIPLINA
- Token Symbol - DSCPL
- Decimals - 18
- Type of token - BEP20
- INITIAL SUPPLY - 1,200,000 DSCPL
- MAX Supply - 1,460,000,000 DSCPL

1.1.2 Bridge contract

The Bridge contract is aimed to provide staking of ERC20 Disciplina on Ethereum.

The contract works as follows: the user places tokens on the contract and adds the parameter of the time period he/she would like to have the tokens staked.

If the user wants to stake tokens and receive a corresponding bonus upon unstaking, he/she needs to specify the number of months 1, 3, 6, 12, 24, and the amount of tokens. The withdrawal of the tokens before the specified period is not possible.

If the user only wants to move tokens from one network to another, he/she must indicate 0 months and the number of tokens.

After staking, the contract assigns an ID to it.

1.1.3 Disciplina contract

The Disciplina contract is Binance Smart Chain based. It is aimed to collect the Bridge contract data of the ERC20 Disciplina tokens staking.

Only the address is recorded in the contract as Oracle can enter data into the contract.

Further, the contract enables the user who staked their tokens in the Bridge contract to withdraw tokens with a staking bonus after the specified date. The bonus is provided according to the distribution conditions listed in the attached tab.

Migration Model			**Staking bonus (after the whole stake period)**				
Month	Migration cost		24	12	6	3	1
1	10		60%	25%	12%	7%	5%
2	9		50%	20%	10%	5%	4%
3	8		45%	18%	9%	4%	3%
4	7		40%	16%	8%	4%	2%
5	6		35%	14%	6%	3%	1%
6	5		30%	12%	5%	2%	1%
7	4		25%	11%	4%	1%	1%
8	3		20%	10%	3%	1%	1%
9	2		15%	9%	2%	1%	1%
10-n	1		10%	8%	1%	1%	1%

1.1.4 Token contract functions:tions:

approve	Allows token holders to transferring the right to manage the token on your balance to a third-party address
burn	The token contract allows to all holders burn their tokens
burnFrom	Allows to burn the specified number of tokens from the address that has approved it for the burner address
decreaseApproval	Allows token holders to decrease the amount of tokens, the right to control which is transferred to a third-party address
grantRole	Allows assigning the roles in a contract. This function is only available for the admin role

increaseApproval	Allows token holders to increase the amount of tokens, the right to control which is transferred to a third-party address
mint	The token contract allows the owner or privileged users to mint tokens to a specific address
pause	The token contract allows the owner to pause token transfers and other operations
renounceRole	Allows a user to renounce the assigned role
revokeRole	Allows to revoke the role previously assigned by the admin
transfer	The token contract allows the holder to transfer tokens to a specific address
	Emits Transfer() event when tokens are transferred successfully (include 0 amount transfers)
transferFrom	Allows transfer tokens from the address that previously granted the rights to this operation
	Emits Transfer() event when tokens are transferred successfully (include 0 amount transfers)
unpause	The token contract allows the owner to unpause the token transfers and other operations

1.1.5 Bridge contract functions:

convert	Allows staking the amount of tokens on the available number of months
renounceOwnership	The contract allows the owner to renounce the ownership. After renounceOwnership is called, no one can be the owner of the contract
setHoldTimeOptions	Allows to change the parameters of bonuses. This feature is available after May 1, 2021
transferOwnership	The contract allows the owner to transfer the ownership to another address

1.1.6 Disciplina contract functions:

renounceOwnership	The contract allows the owner to renounce the ownership. After calling renounceOwnership no one can be the owner of the contract
setInfo	Allows the Oracle to write staking data in the contract
setOracle	Allows to appoint an oracle that can write information to the contract
transferOwnership	The contract allows the owner to transfer the ownership to another address
withdraw	Allows the user to withdraw tokens from the contract when the staking date comes

2. Introduction

2.1. Authenticity

The contracts audited are a subset of the contracts and components written in Solidity: https://github.com/Rock-n-Block/AUDIT/commit/9cbf6af512c2f8a04031f724ae045f17d7d1172c

2.2. Scope

The audit reviewed contracts source code provided by the team of Disciplina. Contracts were reviewed in the context of the flattened files, which included three solidity files. The review performed did not assess any scripts, tests, or other non-Solidity files.

2.3. Methodology

This audit was performed as a comprehensive review of the codebase and takes into consideration both the Solidity code, as well as the target platform: Ethereum main network. The Solidity was reviewed not just for common vulnerabilities and antipatterns, but also for its parity with the intent of the deployer, for its efficiency, and for the practices used during development.

2.4. Description of the complex of procedures for reviewing the smart contract

2.4.1 Primary architecture review

- Checking the architecture of the contract.
- The correctness of the code.
- Check for linearity, shortness, and self-documentation.
- Static verification and code analysis for validity and the presence of syntactic errors.

2.4.2 Comparison of requirements and implementation

- Checking the code of the smart contract for compliance with the requirements of the customer code logic, writing algorithms, matching the initial constant values.
- Identification of potential vulnerabilities

2.4.3 Testing according to the requirements

- Control testing of the smart contract for compliance with specified customer requirements.
- Running properties tests of the smart contract in the test net.

2.5. Risk Assessment

Findings were categorized using a risk rating model based on the OWASP method. Each vulnerability takes into consideration the impact and likelihood of exploitation, as well as the relative ease with which the vulnerability is resolved; findings that permeate throughout the codebase will require much more review and work to solve and are rated higher as a result.

To standardize the evaluation, we define the following terminology based on OWASP Risk Rating Methodology:

• Likelihood represents how likely a particular vulnerability is to be uncovered and exploited in the wild;

• Impact measures the technical loss and business damage of a successful attack;

• Severity demonstrates the overall criticality of the risk;

Likelihood and impact are categorized into three ratings: H, M and L, i.e., high, medium and low respectively. Severity is determined by likelihood and impact and can be classified into four categories accordingly, i.e., Critical, High, Medium, Low shown in following Table

Table: Vulnerability Severity Classification

impact	*High*	**Critical**	**High**	**Medium**
	Medium	**High**	**Medium**	**Low**
	Low	**Medium**	**Low**	**Low**
		High	*Medium*	*Low*
			Likelihood	

2.6. Disclaimer

This document reflects the understanding of security flaws and vulnerabilities as they are known to Rock`n`Block, and as they relate to the reviewed project. This document makes no statements on the viability of the project or the safety of its code. This audit does not represent investment advice and should not be interpreted as such.

3. Findings

3.1. Critical Severity

No critical-severity vulnerabilities were found.

3.2. High Severity

No high-severity vulnerabilities were found.

3.3. Medium Severity

No medium-severity vulnerabilities were found.

3.4. Low Severity

No low-severity vulnerabilities were found.

3.5. Formal remarks

The DSCPL contract, intended for deployment in the Binance Smart Chain, is an ERC20 token. Even though the BSC blockchain supports the ERC20 standard 100%, in our opinion, it would be more correct if the listed below errors that may arise from interactions with the contract refer to the BEP20 standard which is generally accepted by the Binance Smart Chain.

Fail with error

'ERC20Pausable: token transfer while paused'

'ERC20PresetMinterPauser: must have pauser role to pause'

"ERC20Pausable: token transfer while paused"

"ERC20PresetMinterPauser: must have minter role to mint"

"ERC20PresetMinterPauser: must have minter role to mint"

"ERC20PresetMinterPauser: must have pauser role to unpause"

4. Manual testing

4. Testings

4.1. **Successful** Deployment token in test net. Open link

4.1.1 **Successful** Check name, symbol, decimals.
4.1.2 **Successful** Owner of contracts set correctly.
4.1.3 **Successful** Checking the distribution function of tokens. Tokens are distributed correctly. Open link
4.1.4 **Successful** The MINT function. Tokens are minted and sent to an address. Open link
4.1.5 **Successful** Minting more than 1,460,000,000 is not possible. Transaction failed. Open link
4.1.6 **Successful** Pause. Token transfer function is disabled. No one can transfer tokens from address to address. Open link
4.1.7 **Successful** Transfer token function disabled. Transaction failed. Open link
4.1.8 **Successful** Unpause. Token transfer function is enabled. Any holder can transfer tokens from address to address. Open link
4.1.9 **Successful** Transfer token function enabled. Transaction successful.
Open link

4.1.10 **Successful** Burn. Burning tokens from management address. Open link

4.1.11 **Successful** Approve. Approving tokens to another address. Open link

4.1.12 **Successful** BurningFrom. Burning tokens from approved address. Open link

4.1.14 **Successful** TransferFrom. Transfer from approved address. Open link

4.2. **Successful** Deployment Bridge contract in test net. Open link

4.2.1 **Successful** Approve. Approving tokens to Bridge contract. Open link

4.2.2 **Successful** Stake tokens on Bridge Open link

4.3. **Successful** Deployment Disciplina contract in test net. Open link

4.3.1 **Successful** SetInfo. Oraclize address added a staking entry to the Discipli-na contract. Open link

4.3.2 **Successful** Replenishment of the Disciplina contract. Open link

4.3.3 **Successful** Withdraw tokens from Disciplina contract address. Open link If there are not enough tokens on the contract, then there will be an error. Fail with error 'Disciplina: INSUFFICIENT_FUNDS'

4.3.4 **Successful** Withdrawal of tokens ahead of time. Transaction failed. Open link

5. Documents and Resources

5.1. Source

The original source code used can be found in the Rock`n`Block repository: https://github.com/Rock-n-Block/AUDIT/tree/main/DISCIPLINA (commit 9cbf6af512c2f8a04031f724ae045f17d7d1172c)

Contracts description
https://docs.google.com/document/d/1Y08bbZiZtF0zASSdQr8NiWAgvyKLZygk0ufhXUfp-VU/edit

5.2. References

OWASP. Risk Rating Methodology. https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology

6. Conclusion

To sum up, it should be mentioned that all the data and proposals mentioned in this report can be considered as recommendations to the actions that have to be done to ensure the quality and security of the smart contracts. The Rock`n`block experts conducted the verification of the smart contracts.

Based on the results of the review and test, we determined that the smart contracts comply with the specifications specified in the terms of reference.

During the contracts review and tests, none of the critical, high, medium, or low possible vulnerabilities were detected. The code review was simple and clean.
The formatting, naming, and other conventions used were fairly regular, and the inheritance structure was well-organized, resulting in a codebase that was easier to review.

For any questions regarding the review and testing of the smart contract, contact audit@rocknblock.io