DeHacker

Code Security Assessment

bloXmove

Jan 29th, 2023

Contents

Summary

DeHacker's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.
Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow/underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service/logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Issue Categories

Every issue in this report was assigned a severity level from the following:

Critical severity issues

A vulnerability that can disrupt the contract functioning in a number of scenarios or creates a risk that the contract may be broken.

Major severity issues

A vulnerability that affects the desired outcome when using a contract or provides the opportunity to use a contract in an unintended way.

Medium severity issues

A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.

Minor severity issues

A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.

Informational

A vulnerability that has informational character but is not affecting any of the code.

Overview

Project Summary

Project Name	bloXmove
Platform	Ethereum
Website	https://bloxmove.com/
Type	DeFi
Language	Solidity

Vulnerability Summary

Vulnerability Level	Total	Pending	Mitigated	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
Major	1	0	0	1	0	0
Medium	0	0	0	0	0	0
Minor	0	0	0	0	0	0
Informational	1	0	0	0	0	1
Discussion	0	0	0	0	0	0

Audit scope

ID	File	SHA256 Checksum
BXP	BloXmovePublicSaleClaimContract.sol	d28b19ab7bb77b062e525607959362128 59fd704eb985577a95644739985d5ef
BXT	BloXmoveToken.sol	717de273e430e9da2496d508d196bc14d 25530048f9b38b22fce773223697f6a
BXX	BloXmovevesting.sol	c2c50e6b8009aa803901b42058438e34 69525b6740f880dba75d5e640696c924

Findings

ID	Category	Severity	Status
bloXmove-01	Centralization / Privilege	Major	Acknowleged
BXX-01	Centralization / Privilege	Informational	Resolved

MAJOR

bloXmove-01 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	Major	Global	Acknowledged

Description

In this project, the role owner has the authority over the following functions:
contract BloXmoveToken.sol:
constructor()
owner is the deployer of this contract. In the constructor of this contract, will mint 49000000 BloXmoveToken to owner , mint 1000000 BloXmoveToken to _publicSaleAddress .
contract BloXmoveVesting.sol:
function addTokenGrant()
owner can add some amount of BloXmoveToken as a grant for _beneficiary.
In BloXmovePublicSaleClaimContract.sol, the role owner is equal to publicSaleAddress, which has theauthority over the following functions:
contract BloXmovePublicSaleClaimContract.sol:
function changeReward(uint256 _reward)
function pause()
function unpause()
owner has the permission to transfer any amount of BloXmoveToken from this contract to otheraccounts. owner can change the reward ratio. owner can pause this contract. owner can unpausethis contract. If this contract is paused, users can not claim their tokens.

Recommendation

We advise the client to carefully manage the owner account's private key to avoid any potential risks ofbeing hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced securitypractices, e.g., Multisignature wallets.Indicatively, here is some feasible suggestions that would also mitigate the potential risk at the differentlevel in term of short-term and long-term:
Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to theprivate key;
Introduction of a DAO/governance/voting module to increase transparency and user involvement.

MAJOR

BXX-01 | Mutability Specifiers Missing

Category	Severity	Location	Status
Gas Optimization	Informational	BloXmovevesting.sol: 16, 18, 33	Resolved

Description

The linked variables are assigned only once, either during their contract-level declaration or during theconstructor 's execution.

Recommendation

For the former, we advise that the constant keyword is introduced in the variable declaration to greatlyoptimize the gas cost involved in utilizing the variable. For the latter, we advise that the immutablemutability specifier is set at the variable's contract-level declaration to greatly optimize the gas cost ofutilizing the variables. Please note that the immutable keyword only works in Solidity versions v0.6.5 and up.

Disclaimer

This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Blockchain technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. A report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.

FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Coding Style

Coding Style findings usually do not affect the generated bytecode but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block. timestamp works.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux “sha256sum" command against the target file.

About

DeHacker is a team of auditors and white hat hackers who perform security audits and assessments. With decades of experience in security and distributed systems, our experts focus on the ins and outs of system security. Our services follow clear and prudent industry standards. Whether it's reviewing the smallest modifications or a new platform, we'll provide an in-depth security survey at every stage of your company's project. We provide comprehensive vulnerability reports and identify structural inefficiencies in smart contract code, combining high-end security research with a real-world attacker mindset to reduce risk and harden code.

BLOCKCHAIINS

- Ethereum
- Cosmos
- Eos
- Substrate

TECH STACK

- Python
- Solidity
- Rust
- c++

CONTACTS

- https://dehacker.io
- https://twitter.com/dehackerio
- https://github.com/dehacker/audits_public
- https://t.me/dehackerio
- https://blog.dehacker.io/

Jan 2023