DeHacker

Code Security Assessment

Gomining

Sep 1st, 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	Gomining
Platform	Ethereum
Website	https://gmt.io/
Type	Defi
Language	Solidity

Vulnerability Summary

Vulnerability Level	**Total**	Pending	**Declined**	**Acknowledged**	**Partially Resolved**	**Resolved**
Critical	0	0	0	0	0	0
Major	1	0	0	0	0	1
Medium	0	0	0	0	0	0
Minor	1	0	0	1	0	0
Informational	2	0	0	2	0	0
Discussion	0	0	0	0	0	0

Audit scope

ID	File	SHA256 Checksum
CGM	Context.sol	543c46d0f81fd4e5d9d6a92beef3d2be18badb483b0b4718c819fe3dbbc37587
GMT	GoMiningToken.sol	d099462f9bd3a7103edc5201f3710230ed1f1c6727abdd3a47ac78ea31c45e98
IER	IERC20.sol	5f4e89bc7ee8aeb26b724218151ebe2b5787f2c73b084d3e2b54ef5716223b18
IEC	IERC20Metadata.sol	1f9380710a5a86e156dc3c0feb20e432f75973345a58bee70121d8df89da7c2f
OGM	Ownable.sol	6fda585e8e9903204726fc7447a41a5b25e2f3c52b89a106b581cccb6e7c024e
PGM	Pausable.sol	1d08116ec31b3068802b764d44eeb45357b7b4cc56b96a336e25adad718cf828
SMG	SafeMath.sol	3bf9042f6d35f2cf0389fb8bef53b3ff29d60740a60e92d423798a62ec57cdc9

Findings

ID	Category	Severity	Status
GMT-01	Centralization /Privilege	Minor	Acknowledged
GMT-02	Gas Optimization	Informational	Acknowledged
GMT-03	Gas Optimization	Informational	Acknowledged
GMT-04	Centralization /Privilege	Major	Resolved

MINOR

GMT-01 | Initial token distribution

Category	Severity	Location	Status
Centralization /Privilege	Minor	GoMiningToken.sol: 36	Acknowledged

Description

All of the $GMT tokens are sent to the contract deployer when deploying the contract.

Recommendation

We recommend the team to be transparent regarding the initial token distribution process.

INFORMATIONAL

GMT-02 | Potential Risk On approve /transferFrom Methods

Category	Severity	Location	Status
Gas Optimization	Informational	GoMiningToken.sol: 112~115, 130~138	Acknowledged

Description

These two methods in ERC20 could be used in an attack that allows a spender to transfer more tokensthan the owner of the tokens ever wanted to allow the spender to transfer. Here is the reference link:https://docs.google.com/document/d/1YLPtQxZu1UAvO9cZ1O2RPXBbT0mooh4DYKjA_jp-RLM/edit#

Recommendation

Consider using SafeERC20 .

INFORMATIONAL

GMT-03 | Proper Usage of public and external Type

Category	Severity	Location	Status
Gas Optimization	Informational	GoMiningToken. sol: 42~44, 50 ~52, 67~69	Acknowledged

Description

public functions that are never called by the contract could be declared external . Example: Functions name() , symbol() and decimals() .

Recommendation

Consider using the external attribute for functions never called from the contract.

MAJOR

GMT-04 | Potential centralization risk

Category	Severity	Location	Status
Centralization / Privilege	Major	GoMiningToken.sol: 216~218, 240~242	Resolved

Description

Functions mint and burn are merely called by the owner, and they allow the caller to mint tokens to anyspecified recipient or burn tokens from any specified account. To improve the trustworthiness of thisprotocol, any plan to the mint token or burn token are better to move to the execution queue of Timelockand also add an emit event , or make the owner Multi-sig.

Recommendation

In general, we strongly encourage the centralized privileges or roles in the protocol to be improved via adecentralized mechanism or smart-contract-based accounts with enhanced security practices.

Indicatively, here are some feasible solutions that would also mitigate the potential risk based on yourbusiness flow:

Time-lock with reasonable latency, i.e. 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.

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
- Eos
- Cosmos
- Substrate

TECH STACK

- Python
- Rust
- Solidity
- c++

CONTACTS

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

DeHacker

Sep 2023