DeHacker

Code Security Assessment

ApeCoin Airdrop SmartContract

July 11th, 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	ApeCoin Airdrop Smart Contract
Platform	Ethereum
Website	https://apecoin.com/
Type	ERC-20
Language	Solidity

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	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	2	0	0	2	0	0
Discussion	0	0	0	0	0	0

Audit scope

ID	File	SHA256 Checksum
AGT	AirdropGrapesToken.sol	dcc0c5ef21dfb3c3f36a25379da87a56f1675988f34826f2603fbaee4d154c90
MCK	Migrations.sol	773e9056aee531c9f7a806d4bb7dc458fea321846dc830c73fffcbadcbb5831c

Findings

ID	Category	Severity	Status
AGT-01	Centralization / Privilege	Major	Acknowledged
AGT-02	Control Flow	informational	Acknowledged
AGT-03	Coding Style	informational	Acknowledged

Major

AGT-01| CENTRALIZATION RELATED RISKS

Category	Severity	Location	Status
Centralization / Privilege	Major	AirdropGrapesToken .sol:88,99,189	Acknowledged

Description

In the function claimUnclaimedTokens(),when the function is triggered,the owner will transfer the remainingunclaimed token to the owner's address.

In the contract AirdropGrapesToken.sol,the role onlyOwner has authority over the following functions:
claimUnclaimedTokens():when the function is triggered,the owner will transfer the remaining unclaimedtoken to the owner's address.
startClaimablePeriod():The onlyOwner can start the claimable period at any time.
pauseClaimablePeriod():The onlyOwner can pause the contract any time.

Any compromise to the onlyOwner account may allow a hacker to take advantage of this authority.

Exploit Scenario:
Note:Suppose the attacker gets the private key of the owner.
1.The attacker calls pauseClaimablePeriod() to pause the contract.
2. The attacker calls startClaimablePeriod() and set the claimDuration parameter to be a very small value(e.g.,1)in the contract.
3. The attacker calls claimUnclaimedTokens() to receive all the grapesToken and ETH.
4. The attacker sells grapesToken.

Recommendation

The risk describes the current project design and potentially makes iterations to improve in
the security operationand level of decentralization,which in most cases cannot be resolved

entirely at the present stage.We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked.In general,we strongly recommend
centralized privileges or roles in the protocol be improved via a decentralizedmechanism or smart-contract-based accounts with enhanced security practices,e.g., multi-signature wallets.

Indicatively,here are some feasible suggestions that would also mitigate the potential risk at a different level interms of short-term long-term and permanent:

Short Term:
Timelock and Multi sign (2/3,3/5) combination mitigate by delaying the sensitive operation and
avoiding a single point of key management failure:
Time-lock with reasonable latency,e.g.,hours for awareness on privileged operations;
AND
Assignment of privileged roles to multi-signature wallets to prevent a single point of failure
due to the privatekey compromised;
AND
A medium/blog link for sharing the timelock contract and multi-signers addresses information with the publicaudience.

Long Term
Timelock and DAO,the combination,mitigate by applying decentralization and transparency
Time-lock with reasonable latency e.g.,48 hours for awareness on privileged;
AND
Introduction of a DAO/governance/voting module to increase transparency and user involvement;
AND
A medium/blog link for sharing the timelock contract multi-signers addresses and DAO information with thepublic audience.

Permanent
Renouncing the ownership or removing the function can be considered fully resolved.
Renounce the ownership and never claim back the privileged roles;
OR
Remove the risky functionality.

Noted:Recommend considering the long-term solution or the permanent solution The project team shall make adecision based on the current state of their project,timeline,and project resources.

Informational

AGT-02| AVOID MULTIPLE INITIALIZATIONS

Category	Severity	Location	Status
Control Flow	Informational	AirdropGrapesToken.sol :63~71	Acknowledged

Description

The contract is paused after the deployment and configuration initiation.The constructor initialization may not needto happen at the same time.It will be a good practice to avoid allowing multiple initializations.

Recommendation

Recommend separating the constructor and initialized operation and enabled when your workflow requires it.It will be easier for Etherscan verification as you do not have to deal with messy constructor arguments.It can be used toavoid the 'Stack too deep,try removing local variables' error when passing.

```
contract AirdropGrapesToken {
  bool isInitialized = false;

  function initialize(
    uint256 _param1,
    uint256 _param2,
    uint256 _param3,
    address _param4,

  ) public {
    require(!isInitialized, 'Contract is already initialized!');
    isInitialized = true;

    param1 = _param1;
    ...
    param4 = _param4;

  }
}
```

Informational

AGT-03| MISSING EMIT EVENTS

Category	Severity	Location	Status
Coding Style	Informational	AirdropGrapesToken.sol :99~101	Acknowledged

Description

The function pauseClaimablePeriod() affects the sensitive status of the contract and should emit events asnotifications to users.

Recommendation

Consider adding an event for sensitive action and emit it in the function.

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/

DeHacker

July 2023