



Code Security Assessment

Everdome

May 20th, 2023



Contents

CONTENTS	1
SUMMARY	2
ISSUE CATEGORIES	3
OVERVIEW	4
PROJECT SUMMARY	4
VULNERABILITY SUMMARY	4
AUDIT SCOPE	5
FINDINGS	6
MAJOR.....	7
EVE-01 : Centralization Risk in Everdome.sol	7
DESCRIPTION	7
RECOMMENDATION.....	8
INFORMATIONAL.....	9
EVE-02 : Missing Emit Events	9
DESCRIPTION	9
RECOMMENDATION.....	9
INFORMATIONAL.....	10
EVE-03 : Improper Usage of `public` and `external` Type	10
DESCRIPTION	10
RECOMMENDATION.....	10
INFORMATIONAL.....	11
EVE-04 : Unlocked Compiler Version	11
DESCRIPTION	11
RECOMMENDATION.....	11
INFORMATIONAL.....	12
EVE-05 : Variables That Could Be Declared as Immutable	12
DESCRIPTION	12
RECOMMENDATION.....	12
MAJOR.....	13
EVE-06 : Initial Token Distribution	13
DESCRIPTION	13
RECOMMENDATION.....	13
MINOR.....	14
EVE-07 : Missing Input Validation	14
DESCRIPTION	14
RECOMMENDATION.....	14
INFORMATIONAL.....	15
EVE-08 : Redundant Code	15
DESCRIPTION	15
RECOMMENDATION.....	15
DISCLAIMER.....	16
APPENDIX.....	17
ABOUT.....	18



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	Everdome
Platform	Other
Website	https://dimitra.io/
Type	Gaming
Language	Solidity

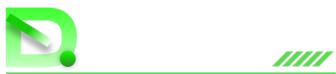
Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
Major	2	0	0	2	0	0
Medium	0	0	0	0	0	0
Minor	1	0	0	1	0	0
Informational	5	0	0	5	0	0
Discussion	0	0	0	0	0	0



Audit scope

ID	File	SHA256 Checksum
EVE	contracts/Everdome.sol	6762ef0a35f78b317a666c60de59dbf7d55a7bbb260c2b67c6c8df74ff7aa5ff



Findings

ID	Category	Severity	Status
EVE-01	Centralization /Privilege	Major	Acknowledged
EVE-02	Coding Style	Informational	Acknowledged
EVE-03	Gas Optimization	Informational	Acknowledged
EVE-04	Language Specific	Informational	Acknowledged
EVE-05	Gas Optimization	Informational	Acknowledged
EVE-06	Centralization /Privilege	Major	Acknowledged
EVE-07	Volatile Code	Major	Acknowledged
EVE-08	Logical Issue	Informational	Acknowledged



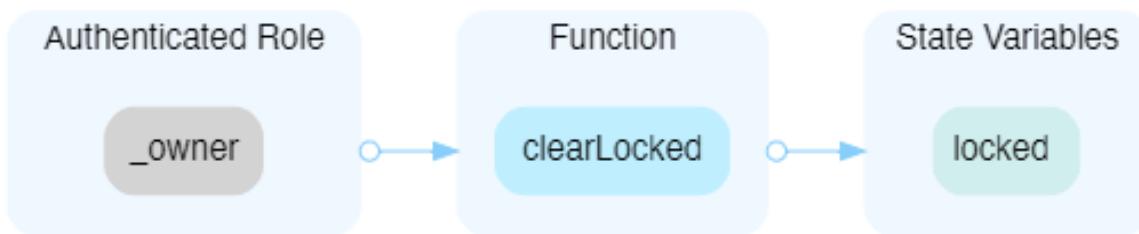
Major

EVE-01 | Centralization Risk In Everdome.sol

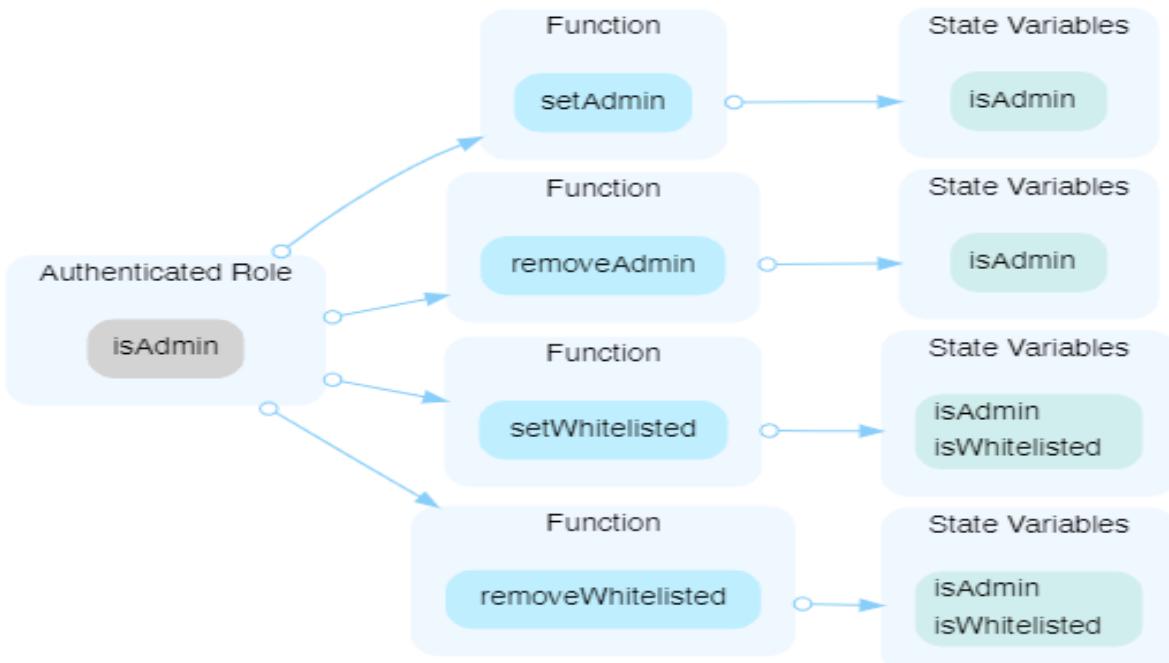
Category	Severity	Location	Status
Centralization / Privilege	Major	contracts/Everdome.sol: 59~63, 54~57, 48~52, 43~46, 39	Acknowledged

Description

In the contract Everdome the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and unlock the token transfer.



In the contract Everdome the role `isAdmin` has authority over the functions shown in the diagram below. Any compromise to the `isAdmin` account may allow the hacker to take advantage of this authority and manage the admin and the whitelist.





Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and 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 decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms 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., 48 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 private key compromised;

AND

A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

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 operations;
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 the public 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.



Informational

EVE-02 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	contracts/ Everdome.sol: 59~ 63, 54~57, 48~52, 43~46, 39~41	Acknowledged

Description

There should always be events emitted in the sensitive functions that are controlled by centralization roles.

Recommendation

It is recommended emitting events for the sensitive functions that are controlled by centralization roles.



Informational

EVE-03 | Improper Usage Of public And external Type

Category	Severity	Location	Status
Gas Optimization	Informational	contracts/ Everdome.sol: 225, 199, 59, 54, 48, 43, 39	Acknowledged

Description

public functions that are never called by the contract could be declared as external . externalfunctions are more efficient than public functions.

Recommendation

Consider using the external attribute for public functions that are never called within the contract.



Informational

EVE-04 | Unlocked Compiler Version

Category	Severity	Location	Status
Language Specific	Informational	contracts/ Everdome.sol: 2	Acknowledged

Description

The contract has unlocked compiler version. An unlocked compiler version in the source code of the contract permits the user to compile it at or above a particular version. This, in turn, leads to differences in the generated bytecode between compilations due to different compiler versions. This can lead to an ambiguity when debugging as compiler specific bugs may occur in the codebase that would be hard to identify over a span of multiple compiler versions rather than a specific one.

Recommendation

We advise that the compiler version is instead locked at the lowest version possible that the contract can be compiled at. For example, for version v0.6.2 the contract should contain the following line :

```
pragma solidity 0.8.0;
```



Informational

EVE-05 | Variables That Could Be Declared As Immutable

Category	Severity	Location	Status
Gas Optimization	Informational	contracts/ Everdome.sol: 19~20, 17~18	Acknowledged

Description

The linked variables assigned in the constructor can be declared as immutable . Immutable state variables can be assigned during contract creation but will remain constant throughout the lifetime of a deployed contract. A big advantage of immutable variables is that reading them is significantly cheaper than reading from regular state variables since they will not be stored in storage.

Recommendation

We recommend declaring these variables as immutable. Please note that the immutable keyword only works in Solidity version v0.6.5 and up.



Major

EVE-06 | Initial Token Distribution

Category	Severity	Location	Status
Centralization / Privilege	Major	contracts/ Everdome.sol: 33~36	Acknowledged

Description

All of the DOME tokens are sent to the contract _owner when deploying the contract. This could be a centralization risk as the deployer can distribute DOME tokens without obtaining the consensus of the community.

Recommendation

We recommend the team to be transparent regarding the initial token distribution process, and the team shall make enough efforts to restrict the access of the private key.



Minor

EVE-07 | Missing Input Validation

Category	Severity	Location	Status
Volatile Code	Minor	contracts/ Everdome.sol: 34	Acknowledged

Description

The given input owner is missing the check for the non-zero address.

Recommendation

We advise adding the check for the passed-in values to prevent unexpected error.



Informational

EVE-08 | Redundant Code

Category	Severity	Location	Status
Logical Issue	Informational	contracts/ Everdome.sol: 3	Acknowledged

Description

Could not identify the imported file hardhat/console.sol .

Recommendation

We advise the client to remove this redundant import.



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.

BLOCKCHAINS

	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

May 2023