DeHacker

Code Security Assessment

CARV

October 26th, 2024

Contents

Summary

DeHacker's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best prac tices. 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	CARV
Platform	Polygon
Website	carv.io/home
Type	DeFi
Language	Solidity
Codebase	https://github.com/carv-protocol/carv-contracts/tree/cb470caaaafd38797a86cd18def222f730a1cf50/contracts/,https://github.com/carv-protocol/carv-contracts/tree/a7f1ea88a4ef4ade4c7b7088ff5245419e9feec4/contracts

Vulnerability Summary

Vulnerability Level	Total	Mitigated	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
Major	2	0	0	2	0	0
Medium	3	0	0	2	0	1
Minor	2	0	0	1	0	1
Informational	4	0	0	1	0	3
Discussion	0	0	0	0	0	0

Audit scope

ID	File	SHA256 Checksum
CIO	contracts/CarvINOERC20.sol	03448c96679f88f2ac20fe98310b83a40c0b62ffade307a499ba1afab1cd8b26
CAP	CarvAchievementsProxy.sol	0f1cb50030f53e331cc3242a6db8ec0ab414c715e259cf92ed746680dd35205c
CEC	contracts/CarvEvents.sol	0a6abc18f5e823cd01a0cf57a63a0a783ed66a9f836493647567da3ee29b8178
CEP	CarvEventsProxy.sol	0755dcde37de639d88d9408bcf09bb70a0db90d31928f5957f3de356caa70a93
CIE	CarvINOERC20.sol	876df32092360cb626e801f23bc89873ea84ce00821efed4699801d266505a3e
CEB	CarvEvents.sol	e27aa40a91afdad38f122ea3c013dbc1b0635652ecaad5dc91f8a4b9ac2c15f2
CIN	contracts/CarvINO.sol	aec33af62050d13fe65b341a703119308f4c1af992c51a81774189ab6c605f94
CAB	CarvAchievements.sol	515020928a3949b37cebdeaa5b75428811a533b4b4c3019806190b317b93d04c
CNO	CarvINO.sol	b4edce96319c66e2bcce796964ee0d6555a5e84bab3bb055394fda7320ced2d3
CAC	contracts/CarvAchievements.sol	ad295ff4aef65e81456d91ac1951a62d0cde037bc9fa7e170a28cae3d5e4e469

Findings

ID	Title	Severity	Status
GLOBAL-01	Centralization Risk - UpgradeableContracts	Major	Acknowledged
GLOBAL-02	Unknown Implementations	Informational	Acknowledged
CIO-01	Incompatibility With Deflationary Tokens	Minor	Acknowledged
CKP-01	Unknow Trusted ForwarderImplementation	Medium	Acknowledged
CKP-02	Length Of msg.data Not Check	Minor	Resolved
CKP-03	Missing Emit Events	Informational	Resolved
CKP-04	Unlocked Compiler Version	Informational	Resolved
CKP-05	Proper Usage Of public Andexternal Type	Informational	Resolved
CON-01	Unprotected Upgradeable Contract	Medium	Resolved
CON-02	Unknown Implementation Of Proxy	Medium	Acknowledged
GIT-01	Centralization Related Risks	Major	Acknowledged

MAJOR

GLOBAL-01 | Centralization Risk - Upgradeable Contracts

Issue	Severity	Location	Status
Centralization / Privilege	Major		Acknowledged

Description

The contracts CarvAchievements and CarvEvents are upgradeable since they are sub-contracts of UUPSUpgradeable.

The role proxy, can call the below function upgradeTo in OpenZeppelin Contracts version v4.5.0, to update the implementation by the proxy.

```
    function upgradeTo(address newImplementation) external
virtual onlyProxy {
        _authorizeUpgrade(newImplementation);
        _upgradeToAndCallUUPS(newImplementation, new
bytes(0), false);
    }
```

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

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.

Recommendation

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 (⅔, ⅗) 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

GLOBAL-02 | Unknown Implementations

Issue	Severity	Location	Status
Volatile Code	Informational		Acknowledged

Description

The implementations of imported source files started with @openzeppelin need to be checked whendeploying, and we understand the protocol depends on the 3rd party libraries, however, it is out of this auditscope and should be validated before deploying the protocol in the production environment.

Recommendation

Consider monitoring the implementation of the imported source files to avoid unexpected effects.

MINOR

CIO-01 | Incompatibility With Deflationary Tokens

Issue	Severity	Location	Status
Volatile Code	Minor	contracts/CarvINOERC20.sol (main): 148	Acknowledged

Description

When transferring standard ERC20 deflationary tokens, the input amount may not be equal to the receivedamount due to the charged transaction fee. For example, if a user fills an offer with 100 deflationary tokens(with a 10% transaction fee) in CarvINOERC20 , only 90 tokens actually arrived in the contract. However, theprice of the NFT is 100 tokens from the contract, which causes the contract to lose 10 tokens in such atransaction

Recommendation

We advise the client to regulate the set of pool tokens supported and add necessary mitigation mechanismsto keep track of accurate balances if there is a need to support deflationary tokens.

MEDIUM

CKP-01 | Unknow Trusted Forwarder Implementation

Issue	Severity	Location	Status
Volatile Code	Medium	contracts/CarvAchievements.sol (main): 34~40, 101~113, 106; contracts/CarvEvents.sol (main): 59~65, 160~173, 165	Acknowledged

Description

Although not explicitly declared, the contract CarvAchievements and CarvEvents implement the EIP-2771.The variable _trustedForwarders serves as a 3rd party forwarder to help it identify the address of theTransaction Signer. The contract CarvAchievements and CarvEvents override the _msgSender() functionto decode the msg.sender from msg.data if it is called from the forwarder. The scope of the audit treatsthese 3rd party forwarders as black boxes and assumes their functional correctness. However, in the realworld, 3rd parties can be compromised and this may lead to lost or stolen assets. In addition, upgrades of3rd parties can possibly create severe impacts.

Recommendation

We understand that the business logic of CarvAchievements and CarvEvents requires interaction withthird-party forwarder contracts. We encourage the team to ensure their functional correctness andconstantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities areobserved.

MINOR

CKP-02 | Length Of msg.data Not Check

Issue	Severity	Location	Status
Logical Issue	Minor	contracts/CarvEvents.sol (main): 165~171; contracts/CarvAchievements.sol (main): 106~113	Resolved

Description

```
if (isTrustedForwarder(msg.sender)) {
assembly {
sender := shr(96,
calldataload(sub(calldatasize(), 20)))
}
} else {
return super._msgSender();
}
```

If the msg.sender is a trusted forwarder, we trust that the last bytes of msg.data are the verified sender address and will extract sender address from the end of msg.data by shr(96, calldataload(sub(calldatasize(), 20))).
If an EOA address is incorrectly set as a trusted forwarder, its call data may be empty so that the calculation sub(calldatasize(), 20) will fail.

Recommendation

Consider adding check on the length of msg.data:

Recommendation

```
if (msg.data.length >= 20 &&
isTrustedForwarder(msg.sender)) {
assembly {
sender := shr(96,
calldataload(sub(calldatasize(), 20)))
}
} else {
return super._msgSender();
}
```

INFORMATIONAL

CKP-03 | Missing Emit Events

Issue	Severity	Location	Status
Coding Style	Informational	contracts/CarvEvents.sol (main): 59, 63, 71, 79, 91, 99, 112, 132, 136; contracts/CarvAchievements.sol (main): 34, 38, 46, 50, 75, 93	Resolved

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

CKP-04 | Unlocked Compiler Version

Issue	Severity	Location	Status
Language Specific	Informational	contracts/CarvAchievements.sol (main): 2; contracts/CarvEvents.sol(main): 2	Resolved

Description

The contract has unlocked compiler version. An unlocked compiler version in the source code of thecontract permits the user to compile it at or above a particular version. This, in turn, leads to differences inthe generated bytecode between compilations due to differing compiler version numbers. This can lead toan ambiguity when debugging as compiler specific bugs may occur in the codebase that would be hard toidentify 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 becompiled at. For example, for version v0.8.11 the contract should contain the following line:

INFORMATIONAL

CKP-05 | Proper Usage Of public And external Type

Issue	Severity	Location	Status
Coding Style	Informational	contracts/CarvINOERC20.sol (main): 113, 130; contracts/CarvINO.sol (main): 107, 124	Resolved

Description

Public functions that are never called by the contract could be declared as external . When the inputs arearrays external functions are more efficient than public functions

Recommendation

We advise the client to consider using the external attribute for functions never called within the contract.

MEDIUM

CON-01 | Unprotected Upgradeable Contract

Issue	Severity	Location	Status
Language Specific	Medium	CarvAchievements.sol (v2): 34; CarvEvents.sol (v2): 50	Resolved

Description

Due to a vulnerability in OpenZeppelin Contracts v4.1.0 through v4.3.1, all projects using the UUPS proxypattern should initialize their implementation contracts.
Reference:

- security-advisory-initialize-uups-implementation-contracts
- uupsupgradeable-vulnerability-post-mortem

When the version of OpenZeppelin Contracts is v4.1.0 through v4.3.1, CarvAchievements will be anupgradeable contract that does not protect its initiliaze functions: CarvAchievements.initialize() .Anyone can delete the contract with: UUPSUpgradeable.upgradeTo(address) andUUPSUpgradeable.upgradeToAndCall(address,bytes) .

When the version of OpenZeppelin Contracts is v4.1.0 through v4.3.1, CarvEvents will be an upgradeablecontract that does not protect its initiliaze functions: CarvEvents.initialize() . Anyone can delete thecontract with: UUPSUpgradeable.upgradeTo(address) andUUPSUpgradeable.upgradeToAndCall(address,bytes) .

Recommendation

We recommend the initializer function is called upon contract deployment or contract creation.

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/

October 26th 2024