CERTIK

Security Assessment

CRTR

Aug 18th, 2021

Table of Contents

Summary

This report has been prepared for Cre8tor.io to discover issues and vulnerabilities in the source code of the CRTR project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	CRTR
Platform	Ethereum
Language	Solidity
Codebase	https://etherscan.io/address/0x0ac65666f502a6a9de0a1393f42c72d3ff62c40f
Commit	

Audit Summary

Delivery Date	Aug 18, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

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	2	0	0	2	0	0
Informational	7	0	0	7	0	0
Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
CRT	CRTRToken.sol	f69359d60e94111c0b4f9bbab20cc6185d8771492868734dcec70020eb6ee5b3

Findings

Critical	**0** (0.00%)
Major	**1** (10.00%)
Medium	**0** (0.00%)
Minor	**2** (20.00%)
Informational	**7** (70.00%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
CRT-01	Too Many Digits	Coding Style	Informational	Acknowledged
CRT-02	Initial token distribution	**Centralization / Privilege**	**Minor**	Acknowledged
CRT-03	Function Visibility Optimization	Gas Optimization	Informational	Acknowledged
CRT-04	Misleading Error Message	Volatile Code, Language Specific	Informational	Acknowledged
CRT-05	Centralization Risk	**Centralization / Privilege**	**Major**	Acknowledged
CRT-06	Unlocked Compiler Version	Language Specific	Informational	Acknowledged
CRT-07	SafeMath Not Used	Mathematical Operations	Informational	Acknowledged
CRT-08	Possible Reusability Improvement	Gas Optimization	Minor	Acknowledged
CRT-09	Missing Input Validation	Volatile Code	Informational	Acknowledged
CRT-10	Return Variable Utilization	Gas Optimization	Informational	Acknowledged

CRT-01 | Too Many Digits

Category	Severity	Location	Status
Coding Style	Informational	CRTRToken.sol: 288	Acknowledged

Description

Literals with many digits are difficult to read and review.

Recommendation

We recommend modifying as below:

```
uint256 public constant initialSupply = 4 * 1e9 * (10 ** uint256(decimals));
```

CRT-02 | Initial token distribution

Category	Severity	Location	Status
Centralization / Privilege	**Minor**	CRTRToken.sol: 291	Acknowledged

Description

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

Recommendation

We recommend the team to be transparent regarding the initial token distribution process. For example, by detailing the process in a blog post or article.

CRT-03 | Function Visibility Optimization

Category	Severity	Location	Status
Gas Optimization	Informational	CRTRToken.sol: 315, 324, 363, 371, 387, 392, 432, 483, 492, 501, 514, 528	ⓘ Acknowledged

Description

The linked functions are declared as `public` and are not invoked in any of the contracts contained within the project's scope. The functions that are never called internally within the contract should have external visibility.

Recommendation

We advise that the functions' visibility specifiers are set to `external`.

CRT-04 | Misleading Error Message

Category	Severity	Location	Status
Volatile Code, Language Specific	● Informational	CRTRToken.sol: 383, 423, 333	ⓘ Acknowledged

Description

The error message, line 333, `Already owner` is not intended for checking `_newOwner != address(0)`.

The error message in the linked functions, lines 383 and 423, use the term "locked" to talk about a frozen account. Since an account can be blocked or frozen, the message is not clear about the cause of the error.

Recommendation

We advise that the error message is revised to properly reflect the check's purpose.

Here :

```
333  require(_newOwner != address(0), "Error : transfer ownership to the zero address");
```

```
383  require(!freezes[msg.sender], "Sender account is frozen.");
```

```
423  require(!freezes[_from], "From account is frozen.");
```

CRT-05 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	**Major**	CRTRToken.sol: 528, 514, 501, 492, 483, 432, 392, 387, 371, 363, 324, 315	Acknowledged

Description

In the contract, the role `owner` has the authority over all the linked functions.

Any compromise to the `owner` account may allow the hacker to take advantage of this, for example :

- completely destruct the contract by pausing it and then renouncing to ownership
- freeze any account he wants to force people to pay him to get their money back
- lock any amount of tokens from any account he wants for the duration he wants

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being 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 security practices, e.g., Multisignature wallets.

Indicatively, here is some feasible suggestions that would also mitigate the potential risk at the different level 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 the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

CRT-06 | Unlocked Compiler Version

Category	Severity	Location	Status
Language Specific	● Informational	CRTRToken.sol: 5	ⓘ 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 differing compiler version numbers. 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.

CRT-07 | SafeMath Not Used

Category	Severity	Location	Status
Mathematical Operations	● Informational	CRTRToken.sol: 547, 534, 496	ⓘ Acknowledged

Description

SafeMath from OpenZeppelin is not used in the linked functions which makes them possible for overflow/underflow and will lead to an inaccurate calculation result.
In particular in the function `transferWithLockAfter()` and `lockAfter()` where addition overflow allows to set a release date prior to `now`.

Recommendation

We advise the client to use the SafeMath library for all of the mathematical operations.

CRT-08 | Possible Reusability Improvement

Category	Severity	Location	Status
Gas Optimization	Minor	CRTRToken.sol: 492~499, 514~526	Acknowledged

Description

The logic of the functions `lockAfter()` and `transferWithLockAfter()`'s implementation is similar to the logic in functions `lock()` and `transferWithLockAfter()` respectively. Therefore, there's chance to improve the reusability of the project by invoking functions `lock()` and `transferWithLockAfter()` directly.

Recommendation

We advise refactoring the code as follow :

```
492  function lockAfter(address _holder, uint256 _amount, uint256 _afterTime) public onlyOwner {
493      lock(_holder, _amount, now.add(_afterTime));
494  }
```

```
514  function transferWithLockAfter(address _to, uint256 _value, uint256 _afterTime) public onlyOwner returns (bool) {
515      transferWithLock(_to, _value, now.add(_afterTime));
516      return true;
517  }
```

CRT-09 | Missing Input Validation

Category	Severity	Location	Status
Volatile Code	Informational	CRTRToken.sol: 483, 492	Acknowledged

Description

The given input is missing the check for a non-zero value. This could lead to artificially increase the length of `lockInfo[_holder]`, increasing the gas consumption.

Recommendation

We advise adding the check for the passed-in values to prevent unexpected error as below:

```
require(_amount != 0; "Nothing to lock")
```

CRT-10 | Return Variable Utilization

Category	Severity	Location	Status
Gas Optimization	● Informational	CRTRToken.sol: 451	ⓘ Acknowledged

Description

The linked function declarations contain explicitly named `return` variables that are not utilized within the function's code block.

Recommendation

We advise that the linked variables are either utilized or omitted from the declaration. For example:

```
function balanceOf(address _holder) public view returns (uint256) {
   uint256 lockedBalance = 0;
     for(uint256 i = 0; i < lockInfo[_holder].length ; i++ ) {
        lockedBalance = lockedBalance.add(lockInfo[_holder][i].balance);
     }
     return super.balanceOf(_holder).add(lockedBalance);
}
```

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.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

Volatile Code

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

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.

Coding Style

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

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.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING

MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we’re able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.

CERTIK