Security Assessment

Chain Partners

Sept 23rd, 2021

Table of Contents

Summary

This report has been prepared for Chain Partners to discover issues and vulnerabilities in the source code of the Chain Partners 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	Chain Partners
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/changerio/cng-contracts/blob/f877c850bd1add1f522bab239e2ae507f43501bc/contracts/token/Token.sol
Commit	

Audit Summary

Delivery Date	Sept 23, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

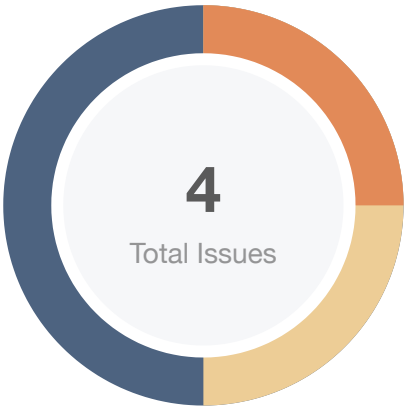
Vulnerability Summary

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	1	0	0	1	0	0
Informational	2	0	0	0	0	2
Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
TCP	Token.sol	4f30144541490cd4427645e7606a88c82ae4fae9a07dbf77e02c248897fce31e

Findings



Critical	0 (0.00%)
Major	1 (25.00%)
Medium	0 (0.00%)
Minor	1 (25.00%)
Informational	2 (50.00%)
Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
TCP-01	Unlocked Compiler Version | Incorrect Versions Of Solidity	Language Specific	Informational	Resolved
TCP-02	Imported Source File Addresses Could Change	Volatile Code	Informational	Resolved
TCP-03	Initial Token Distribution	**Centralization / Privilege**	**Minor**	Acknowledged
TCP-04	Centralization Risk	**Centralization / Privilege**	**Major**	Acknowledged

TCP-01 | Unlocked Compiler Version | Incorrect Versions Of Solidity

Category	Severity	Location	Status
Language Specific	Informational	Token.sol: 3	Resolved

Description

The contract has an 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 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.

`Pragma version^0.8.0;` necessitates a version too recent to be trusted for deployment.

Recommendation

We advise that the compiler version is instead locked at the lowest version possible that the contract can be compiled at.

Moreover, we advise to deploy with any of the following Solidity versions:

- 0.5.16 - 0.5.17
- 0.6.11 - 0.6.12
- 0.7.5 - 0.7.6

And to use a simple pragma version that allows any of these versions. Consider using the latest version of Solidity for testing.

Alleviation

`[Chain Partners]`: We've already deployed contracts with solidity version of v0.8.5+commit.a4f2e591 to use @openzeppelin/contract@4.1.0 and to avoid a solidity bug named KeccakCaching in the commit 0beb83156e4d4f3007d45aa5d3732daba03c4e84

TCP-02 | Imported Source File Addresses Could Change

Category	Severity	Location	Status
Volatile Code	Informational	Token.sol: 5~12	Resolved

Description

Depending on the version of the openzeppelin contracts, the addresses of the files could change, making the importation impossible.

Recommendation

We advise using more "update-proof" links and locking the compiler version to avoid problems with a new version of the openzeppelin libraries.

Alleviation

`[Chain Partners]`: the client heeded the advice and fix the item in the commit 4cb24f163cdbcbc230470f7aff54e066be121d32

TCP-03 | Initial Token Distribution

Category	Severity	Location	Status
Centralization / Privilege	**Minor**	Token.sol: 23	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 through a blog post or an article detailing precisely how the deployment works.

Alleviation

`[Chain Partners]`: We used Token.sol to deploy below types of ERC20 token contracts. We distribute main token (CNG) and sub-tokens (ecoCNG, teamCNG, backCNG, pubCNG, strCNG) to token holders on a reasonable basis.

Reference: https://gist.github.com/4000D/d9a28ba9a9640ed9254eff20a5de30d8

Sub-tokens are swapped to CNG over vesting period of each sub-token, and only CNG is used in our protocol. Currently all CNG tokens are locked in SwapperVault except some tokens that holders have claimed already.

ecoCNG is the only sub-token that needs consensus of the community and distribution of ecoCNG will be aounceed to our community channel.

We believe this is enought to make initial token distribution transparent, even though distribution of a single sub-token like teamCNG, backCNG, pubCNG, strCNG, would not be absolutely transparent (because that kind of sub-tokens cannot be).

`[CertiK]`: This contract gives all tokens to the deployer. If all deployment has been done with transparency then there is no more risk but if this contract will be used again then the risk is still there for the new token.

TCP-04 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	Major	Token.sol: 28~29, 25~26, 36~38, 44~46	Acknowledged

Description

When the contract is deployed, the deployer gets the `DEFAULT_ADMIN_ROLE`, `MINTER_ADMIN_ROLE`, the `PAUSER_ADMIN_ROLE`, the `MINTER_ROLE`, and the `PAUSER_ROLE`.

The `DEFAULT_ADMIN_ROLE` have the authority over the following functions:

- `changeAdminRole()`, `changeRole()` to transfer the `DEFAULT_ADMIN_ROLE` to someone else;
- `grantRole()` to give to an account the `DEFAULT_ADMIN_ROLE`, `MINTER_ADMIN_ROLE` or the `PAUSER_ADMIN_ROLE`;
- `revokeRole()` to remove the `DEFAULT_ADMIN_ROLE`, `MINTER_ADMIN_ROLE` or the `PAUSER_ADMIN_ROLE` from an account;
- `renounceRole()`to renounce to the `DEFAULT_ADMIN_ROLE`.

The `MINTER_ADMIN_ROLE` have the authority over the following functions:

- `changeRole()` to transfer the `MINTER_ADMIN_ROLE` to someone else;
- `grantRole()` which allows the `MINTER_ADMIN_ROLE` to give someone the `MINTER_ROLE`;
- `revokeRole()` to remove the `MINTER_ROLE` from an account;
- `renounceRole()`to renounce to the `MINTER_ADMIN_ROLE`.

The `PAUSER_ADMIN_ROLE` have the authority over the following functions:

- `changeRole()` to transfer the `PAUSER_ADMIN_ROLE` to someone else;
- `grantRole()` which allows the `PAUSER_ADMIN_ROLE` to give someone the `PAUSER_ROLE`;
- `revokeRole()` to remove the `PAUSER_ROLE` from an account;
- `renounceRole()`to renounce to the `PAUSER_ADMIN_ROLE`.

The `MINTER_ROLE` have the authority over the following functions:

- `changeRole()` to transfer the `MINTER_ROLE` to someone else;
- `mint()` to mint how many tokens the winter wants to anyone he wants.

The `PAUSER_ROLE` have the authority over the following functions:

- `changeRole()` to transfer the `PAUSER_ROLE` to someone else;
- `pause()` which allows a pauser to stop transferring, burning, and minting tokens;
- `unpause()` to reverse the effect of the `pause()` function.

Any compromise to a `DEFAULT_ADMIN_ROLE` account may allow the hacker to take advantage of this and :

- transfer the role to an address he controls;
- renounce the role and leave the contract without `DEFAULT_ADMIN_ROLE`, so without the possibility to change the `MINTER_ADMIN_ROLE` and the `PAUSER_ADMIN_ROLE`;
- give the `MINTER_ADMIN_ROLE` and the `PAUSER_ADMIN_ROLE` to any address he wants and revoke these roles for anyone he wants.

Any compromise to a `MINTER_ADMIN_ROLE` account may allow the hacker to take advantage of this and :

- transfer the role to an address he controls;
- give the `MINTER_ROLE` to any address he wants and revoke the role for anyone he wants.
- renounce the role and leave the account without this privilege.

Any compromise to a `PAUSER_ADMIN_ROLE` account may allow the hacker to take advantage of this and :

- transfer the role to an address he controls;
- give the `PAUSER_ROLE` to any address he wants and revoke the role for anyone he wants.
- renounce the role and leave the account without this privilege.

Any compromise to a `MINTER_ROLE` account may allow the hacker to take advantage of this and:

- transfer the role to an address he controls;
- mint the number of tokens he wants to addresses he controls.

Any compromise to a `PAUSER_ROLE` account may allow the hacker to take advantage of this and :

- transfer the role to an address he controls;
- pause the contract, which will necessitate another account with the `PAUSER_ROLE` to unpause it;
- unpause the contract when it's paused (for security reasons for example).

Recommendation

We advise the client to carefully manage the `deployer` account's private key to avoid any potential risks of being hacked since he is the only one having all these privileges at the beginning.

We also advise dividing up the roles so that there is a balance between the different elements or being completely transparent about who has what role and what it does.

If there is a division of the roles, we advise the client to carefully manage the `DEFAULT_ADMIN_ROLE` 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.

It should be noted that :

- everything that a `DEFAULT_ADMIN_ROLE` account can do another `DEFAULT_ADMIN_ROLE` account can undo it. One can also revoke the `DEFAULT_ADMIN_ROLE` of another `DEFAULT_ADMIN_ROLE` account. So if an account with this privilege is compromised, the contract can be completely sabotaged by the hacker.
- if any account other than a `DEFAULT_ADMIN_ROLE` one is compromised, then the `DEFAULT_ADMIN_ROLE` has enough privileges to reassign roles as before.
- if there is no more `DEFAULT_ADMIN_ROLE` account, then a compromise to a `MINTER_ADMIN_ROLE` or a `PAUSER_ADMIN_ROLE` is more problematic but the effects could eventually be partially mitigated by other accounts having the same privileges.
- even if a `MINTER_ROLE` doesn't have huge privileges if an account with this role is compromised the hacker could mint huge amounts of token and devaluates completely the value of the token.
- if there is only one `MINTER_ADMIN_ROLE` account and if it is compromised then the consequences could be irreversible (for example the `MINTER_ROLE` granted to an address controlled by the hacker and renouncing of the `MINTER_ADMIN_ROLE`) and cause the contract to be unusable.
- same thing for the `PAUSER_ADMIN_ROLE` (for example the `PAUSER_ROLE` granted to an address controlled by the hacker, all the other `PAUSER_ROLE` account revoked and then renouncing of the `PAUSER_ADMIN_ROLE`). The contract could be paused by the hacker forever.

Alleviation

`[Chain Partners]`: Currently it is not sure to use some governance mechanism for our protocol, but at least we will use Gnosis's Safe Contract to hold roles of Token.sol contract like the table in the gist

reference link https://gist.github.com/4000D/d9a28ba9a9640ed9254eff20a5de30d8.

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.

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.

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.