Smart contract security audit

Capsule Protocol

v.1.0

Table of Contents

1.0 Introduction

1.1 Project engagement

Bspg l e Bcac k `cp md 0 . 00* the A_nqsjc Protocol ("Capsule") c l e_ecb AW] QCA rm _sbgr q k _pr am l rp_arq rf_r r f cw apc_rcb, R f c c l e_ec k c l r u_q rca f l ga_j g l l_rspc _ l b dmasqcb m l gbc l rgdwg l e qcaspgrw dj_u q g l rfc bcqge l _ l b g k njc k c l r_rgm l md r f c am l rp_arq, A_nqsjc npm t gbcb AW] QCA u gr f _aacqq rm r f cgp ambc pcnmqgrmpw _ l b u f gr cn_ncp

Capsule gq _ long term defi-lending protocol designed for highly illiquid digital assets (also known as " Alts" and "MicroCaps" with transaction buffers and open customizable terms for lenders.

1.2 Disclaimer

It should be noted that this audit is not an endorsement of the reliability or effectiveness of the contract, rather limited to an assessment of the logic and implementation. In order to ensure a secure contract that's able to withstand the network's fast-paced and rapidly changing environment, we at CY_SEC recommend that the Capsule team put in place a bug bounty program to encourage further and active analysis of the smart contract.

2.0 Coverage

2.1 Target Code and Revision

For this audit, we performed research, investigation, and review of the Capsule contract followed by issue reporting, along with mitigation and remediation instructions outlined in this report. The following code files are considered in-scope for the review:

Source:

Capsule.flatten.sol

CapsuleToken.flatten.sol

CapsuleTokenChild.flatten.sol

2.2 Attacks made to the contract

In order to check for the security of the contract, we tested several attacks in order to make sure that the contract is secure and follows best practices.

№	Issue description.	Checking status
1	Compiler warnings.	PASSED
2	Race conditions and Reentrancy. Cross-function race conditions.	PASSED
3	Possible delays in data delivery.	PASSED
4	Oracle calls.	PASSED
5	Front running.	PASSED
6	Timestamp dependence.	PASSED
7	Integer Overflow and Underflow.	PASSED
8	DoS with Revert.	PASSED
9	DoS with block gas limit.	PASSED
10	Methods execution permissions.	PASSED
11	Economy model. If application logic is based on an incorrect economic model, the application would not function correctly and participants would incur financial losses. This type of issue is most often found in bonus rewards systems, Staking and Farming contracts, Vault and Vesting contracts, etc.	PASSED
12	The impact of the exchange rate on the logic.	PASSED
13	Private user data leaks.	PASSED
14	Malicious Event log.	PASSED
15	Scoping and Declarations.	PASSED

16	Uninitialized storage pointers.	PASSED
17	Arithmetic accuracy.	PASSED
18	Design Logic.	PASSED
19	Cross-function race conditions.	PASSED
20	Safe Zeppelin module.	PASSED
21	Fallback function security.	PASSED
22	Overpowered functions / Owner privileges	PASSED

3.0 Security Issues

3.1 High severity issues [0]

No high severity issues found.

3.2 Medium severity issues [0]

No medium severity issues found.

3.3 Low severity issues [1]

1.Lock transfer

Issue:

There is a possibility of transfer in the DigiToken.flatten.sol contract even if one of the recipient or sender is locked whitelisted.

Recommendation:

It would be better to check that both addresses are lock whitelisted, not only one of them.

Owner privileges

1. Owner can change the childChainManagerProxy address in CapsuleTokenChild.flatten.sol contract.
2. ChildChainManagerProxy address can mint tokens to any address using function deposit in CapsuleTokenChild.flatten.sol contract.
3. Owner can initiate token collateral transfer as per the whitepaper
4. Owner can add and remove from the lock whitelist in CapsuleToken.flatten.sol contract.
5. ChildChainManagerProxy address can mint tokens to any address using function deposit in DigiTokenNFTChild.flatten.sol contract.
6. Owner can lock the protocol for new deposits an loan origination.

Fix:

1. Solved by the dev team.
2. It is a function inherited from the Matic network so it cannot be modified directly from Capsule. There is no possible solution as it is a dependency on the MATIC acting as a bridge to layer2 network.
3. N/A - see whitepaper
4. N/A
5. N/A - see Summary & Recomendations
6. N/A. Withdrawals of matured assets is not blocked.

4.0 Summary of the audit

There were no critical incidents found. The MINT function named in the report has no danger because the Capsule team does not have access to its execution, it is only possible that it be executed from Polygon/MATIC.. The protocol may be locked by the Capsule team, as the whitepaper envisions. This prevents new deposits and loan origination, but does not affect user withdrawals of locked tokens.

5.0 Additional Notes

There were no critical incidents found. The MINT function named in the report has no danger since the Capsule team does not have access to its execution, it is only possible that it be executed from Polygon/MATIC.

The protocol uses a child-token paradigm to be compatible with POLYGON/MATIC layer2 bridge solution and thus exposes the minting function to Polygon. While there is no evidence or data to suggest that Polygon has any security flaw, and there are countless number of similar layer2 deployments without issue, there may be a case for not using layer2, but that is the protocol's business decision.

,