#HashEx
BLOCKCHAIN SECURITY

Plxyer

smart contracts preliminary audit report for internal use only

December 2022

hashex.org

contact@hashex.org

Contents

1. Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and HashEx and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (HashEx) owe no duty of care towards you or any other person, nor does HashEx make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties, or other terms of any kind except as set out in this disclaimer, and HashEx hereby excludes all representations, warranties, conditions, and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, HashEx hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against HashEx, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of the use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed. HashEx owns all copyright rights to the text, images, photographs, and other content provided in the following document. When using or sharing partly or in full, third parties must provide a direct link to the original document mentioning the author (hashex.org).

2. Overview

HashEx was commissioned by the Plxyer team to perform an audit of their smart contract. The audit was conducted between 30/11/2022 and 03/12/2022.

The purpose of this audit was to achieve the following:

- Identify potential security issues with smart contracts
- Formally check the logic behind given smart contracts.

Information in this report should be used for understanding the risk exposure of smart contracts, and as a guide to improving the security posture of smart contracts by remediating the issues that were identified.

The code is available at 0xE79869AeCc5ba7d29710d3718Fc55820fc43D959 in Ethereum mainnet.

2.1 Summary

Project name	Plxyer
URL	http://plxyer.com
Platform	Ethereum
Language	Solidity

2.2 Contracts

Name	Address
Plxyer	0xE79869AeCc5ba7d29710d3718Fc55820fc43D959

3. Found issues

High	1 (33%)
Info	2 (67%)

C1. Plxyer

ID	Severity	Title	Status
C1-01	High	Pausable transfers	Open
C1-02	Info	Lack of error messages	Open
C1-03	Info	Use case of rescuseSafeERC20() is unclear	Open

4. Contracts

C1. Plxyer

Overview

Standard ERC20 token contract with pausable transfers under the owner's control.

Issues

C1-01 Pausable transfers

High Open

The owner of the token contract can pause token transfers. If the token owner's account is compromised, an attacker may transfer ownership and make the token non-operational by pausing all transfers.

Recommendation

Renounce the ownership or transfer it to a governance contract.

C1-02 Lack of error messages

Info Open

The `require` statements in the contract lack error messages. An example of the code without error messages:

```
    modifier onlyPauser() {
        require(isPauser(msg.sender));
        _;
    }
```

Recommendation

We recommend adding messages to all `require` statements to ease debugging and improve user experience with the smart contract.

C1-03 Use case of rescuseSafeERC20() is unclear

Info Open

The function `rescuseSafeERC20()` is aimed at providing the possibility to withdraw ERC20 tokens, accidentally sent to the contract. It has the same functionality as the function `rescuseERC20()` but makes unnecessary additional checks if the transfer was successful.

5. Conclusion

1 high severity issue was found during the audit. No issues were resolved in the update.

The high severity issue is a centralisation issue when the owner can pause the operation of the token and users won't be able to transfer the token. Users using the token must trust that the owner and their account are properly secured. We strongly recommend transferring the ownership to a MultiSig account to minimise the risk of private key compromise.

Appendix A. Issues' severity classification

- **Critical.** Issues that may cause an unlimited loss of funds or entirely break the contract workflow. Malicious code (including malicious modification of libraries) is also treated as a critical severity issue. These issues must be fixed before deployments or fixed in already running projects as soon as possible.
- **High.** Issues that may lead to a limited loss of funds, break interaction with users, or other contracts under specific conditions. Also, issues in a smart contract, that allow a privileged account the ability to steal or block other users' funds.
- **Medium.** Issues that do not lead to a loss of funds directly, but break the contract logic. May lead to failures in contracts operation.
- **Low.** Issues that are of a non-optimal code character, for instance, gas optimization tips, unused variables, errors in messages.
- **Informational.** Issues that do not impact the contract operation. Usually, informational severity issues are related to code best practices, e.g. style guide.

Appendix B. List of examined issue types

- Business logic overview
- Functionality checks
- Following best practices
- Access control and authorization
- Reentrancy attacks
- Front-run attacks
- DoS with (unexpected) revert
- DoS with block gas limit
- Transaction-ordering dependence
- ERC/BEP and other standards violation
- Unchecked math
- Implicit visibility levels
- Excessive gas usage
- Timestamp dependence
- Forcibly sending ether to a contract
- Weak sources of randomness
- Shadowing state variables
- Usage of deprecated code

 contact@hashex.org

 @hashex_manager

 blog.hashex.org

 linkedin

 github

 twitter

#HashEx
BLOCKCHAIN SECURITY