CERTIK

Code Security Assessment

Sandbox - PR

Jan 26th, 2022

Table of Contents

Summary

This report has been prepared for Sandbox to discover issues and vulnerabilities in the source code of the Sandbox - PR 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	Sandbox - PR
Platform	polygon
Language	Solidity
Codebase	https://github.com/thesandboxgame/sandbox-smart-contracts-private
Commit	14034d0ebdee1d74f2fe402005e4e1a03532b972

Audit Summary

Delivery Date	Jan 26, 2022
Audit Methodology	Static Analysis, Manual Review
Key Components	LandSwap, ERC2771Handler, PausableWithAdmin

Vulnerability Summary

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

Audit Scope

ID	File	SHA256 Checksum
ERC	solc_0.5/BaseWithStorage/ERC2771Handler.sol	6007170c2f9f1f26ab63b7429c54d6a15275675d10e6c63a5b592569209b271c
LSS	solc_0.5/Swap/LandSwap.sol	06f4e6c502f4e85acdf5403e5d9db8da84739d9b22d7211a43c5bede58c1eac0
PWA	solc_0.5/contracts_common/BaseWithStorage/PausableWithAdmin.sol	327869413f21a251d3a44262948f2081861742e2c3d7a53624f1b909554368da

Overview

The **Sandbox** implements a few contracts to interact with user `lands`:

- `ERC721BaseToken` : NFT representing a land. Users can transfer/burn their `lands`.
- `ERC2771Handler` : For compatibility with Meta Transactions (ERC2771).
- `landswap` : For swapping `lands` from an old Grid (`LandBaseToken`) to a new one.
- `PausableWithAdmin` : Extension of the `landswap` contract, allowing the `Admin` to pause contract if needed.

External Dependencies

The scope of the audit treats third-party entities as black boxes and assumes their functional correctness. However, in the real world, third parties can be compromised and this may lead to lost or stolen assets.

There are a few dependent injection contracts or addresses in the current project:

- `AddressUtils`, `ERC721TokenReceiver`, `ERC721Events`, `SuperOperators`, `MetaTransactionReceiver`, `ERC721MandatoryTokenReceiver` for the contract `ERC721BaseToken`;
- `LandBaseToken`, `ERC721MandatoryTokenReceiver` for the contract `landswap`;
- `Admin` for the contract `Admin`;

We assume these contracts or addresses are valid and non-vulnerable actors and implement proper logic to collaborate with the current project.

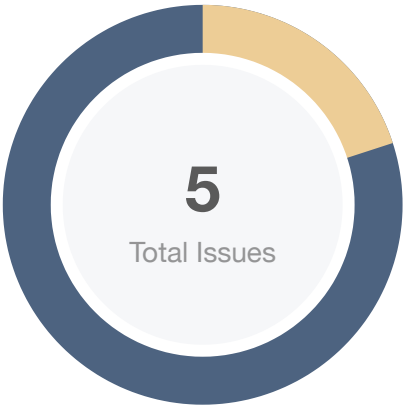
Privileged Functions

The contract `PausableWithAdmin` contains the following privileged functions that are restricted by the `admin`:

- `PausableWithAdmin.pause()`: Pause the contract;
- `PausableWithAdmin.unpause()`: Unpause the contract;

To improve the trustworthiness of the project, dynamic runtime updates in the project should be notified to the community. Any plan to invoke the aforementioned functions should be also considered to move to the execution queue of `Timelock` contract.

Findings



Critical	**0** (0.00%)
Major	**0** (0.00%)
Medium	**0** (0.00%)
Minor	**1** (20.00%)
Informational	**4** (80.00%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
GLOBAL-01	Unlocked Compiler Version	Language Specific	Informational	Acknowledged
ERC-01	Unknown Implementation of Meta Transaction Contract	Control Flow	Informational	Acknowledged
LSS-01	Missing Input Validation	Volatile Code	Minor	Acknowledged
LSS-02	Third Party Dependencies	Volatile Code	Informational	Acknowledged
LSS-03	Lack of Access Control for `burn` Function	Control Flow	Informational	Acknowledged

GLOBAL-01 | Unlocked Compiler Version

Category	Severity	Location	Status
Language Specific	Informational	Global	Acknowledged

Description

The `PausableWithAdmin` 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.

Additionally, it has been noticed that all contracts are compiled with the compiler version over `0.5.2`, which dates from March 2019. It is recommended to update the compiler versions in the contracts, so they are not exposed to potential security issues related to old compiler versions.

Recommendation

In the short term, lock the compiler version.

In the long term, if the contracts are compatible with the version `v0.8.0`, they are recommended to use `v0.8.0`.

Alleviation

[Sandbox]: The team acknowledged the issue and decided to avoid updating the compiler version for all dependent contracts. Unless there is a specific bug in the current version, the contract will keep it as is.

ERC-01 | Unknown Implementation of Meta Transaction Contract

Category	Severity	Location	Status
Control Flow	Informational	solc_0.5/BaseWithStorage/ERC721BaseToken.sol	Acknowledged

Description

The implementation of `_metaTransactionContracts[msg.sender]` is unknown.

In the case that `metaTransaction` contracts are public and propose a `call()` feature, users could call a `metaTransaction` contract, having their call being forwarded to `ERC721BaseToken`.

In this scenario, `msg.sender` received by `ERC721BaseToken` would be the address of `metaTransaction`, which would verify the condition `require(_metaTransactionContracts[msg.sender]`.

If the check can be bypassed through this way, this would be problematic in the functions using this check:

- `ERC721BaseToken.approveFor()`;
- `ERC721BaseToken._checkTransfer()`;
- `ERC721BaseToken._batchTransferFrom()`;
- `ERC721BaseToken.setApprovalForAllFor()`;
- `ERC721BaseToken.burnFrom()`.

For example,

```
301     function setApprovalForAllFor(
302         address sender,
303         address operator,
304         bool approved
305     ) external {
306         require(sender != address(0), "Invalid sender address");
307         require(
308             msg.sender == sender ||
309             _metaTransactionContracts[msg.sender] ||
310             _superOperators[msg.sender],
311             "not authorized to approve for all"
312         );
313 
314         _setApprovalForAll(sender, operator, approved);
315     }
```

The above function will successfully triggered if the caller is from meta transaction contract (i.e., `_metaTransactionContracts[msg.sender]` is `true`).

Recommendation

Recommend carefully setting the `_metaTransactionContracts` array and ensure its correctness.

Alleviation

[Sandbox]: The `basedToken` contract accepts calls from another contract supporting meta-transaction. This was implemented this way to handle meta-transaction from another contract. The team implemented the ERC2771 in the LandSwap contract for this purpose.

LSS-01 | Missing Input Validation

Category	Severity	Location	Status
Volatile Code	Minor	solc_0.5/Swap/LandSwap.sol: 24	Acknowledged

Description

In `LandSwap.sol`, no checks are performed on input parameters to ensure that those are ones that are valid.

In the `LandSwap` contract, in the `initialize()` function :

- `admin` and `trustedForwarder` addresses should be non-zero addresses;
- `oldLand` should not be the same address as `newLand`;
- `admin` should not be the same address as `trustForwarder`.

Recommendation

It is recommended to check the passed-in values to prevent unexpected error

Alleviation

[Sandbox]: The team will cross-check with PR to review the parameters when deploying the contracts. There is no need the further check this on the contract side, as the code depends on those parameters to be correct beside the proposed checks.

LSS-02 | Third Party Dependencies

Category	Severity	Location	Status
Volatile Code	Informational	solc_0.5/Swap/LandSwap.sol: 14	Acknowledged

Description

The contract serves as the underlying entity to interact with the following third-party protocols.

- `_oldLand`
- `_newLand`
- `trustedForwarder`

The scope of the audit treats third-party entities as black boxes and assumes their functional correctness. However, in the real world, third parties can be compromised, which may lead to lost or stolen assets. For instance, if the `trustedForwarder` is initialized to a malicious address, it could cause unexpected errors during the land-swapping process.

Recommendation

Recommend providing the deployed and verified contract. The status of this issue will be updated after contract deployment upon request.

Alleviation

[Sandbox]: The contracts detail are as below:

- _oldLand = 0x50f5474724e0ee42d9a4e711ccfb275809fd6d4a
- _newLand = to be deployed (Audited with Certik)
- trustedForwarder = Check the contract addresses here - deployed smart contract addresses for all networks

LSS-03 | Lack of Access Control for `burn` Function

Category	Severity	Location	Status
Control Flow	Informational	solc_0.5/Swap/LandSwap.sol: 50	Acknowledged

Description

In the current implementation, when the `LandSwap` contract is not in the pause condition, the `burn()` function allows everyone to burn `oldLand` tokens with given `ids`. The `LandSwap` contract receives `oldLand` tokens during the `swap()` invocation and those tokens are supposed to be burnt immediately.

The concern is if those `_oldland` tokens are still valuable to some extent (f.e., causing market fluctuation), a user might call this function at a certain time for his own profit.

Recommendation

If calling `burn()` could cause some unexpected consequence to users/projects, it is recommended to apply an authorized modifier to enforce the access control of this function.

Alleviation

[Sandbox]: The team acknowledged the issue. The point is that even if Sandbox put the burn for Admin, only the `oldLand` token _burn function could be called without any checks (this is the actual vulnerability that triggers this token swap). Eventually, the team thought of burning all the `oldLand` tokens.

Appendix

Finding Categories

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

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.