SOLIDIFIED

Audit Report for Pixowl The Sandbox Assets (ERC721 / ERC1155 token contract) on September 4th, 2019.

Summary

Audit Report prepared by Solidified for Pixowl covering the Sandbox Asset smart contract (and inherited dependencies).

Process and Delivery

Three (3) independent Solidified experts performed an unbiased and isolated audit of the code below. The debrief took place on September 4th, 2019, and the final results are presented here.

Audited Files

The following contracts were covered during the audit:

- src/Asset.sol
- src/Asset/ERC1155ERC721.sol
- contracts_common/src/Interfaces/ERC1155.sol
- contracts_common/src/Interfaces/ERC1155TokenReceiver.sol
- contracts_common/src/Interfaces/ERC721.sol
- contracts_common/src/Interfaces/ERC165.sol
- contracts_common/src/Interfaces/ERC721Events.sol
- contracts_common/src/Interfaces/ERC721TokenReceiver.sol
- contracts_common/src/Libraries/AddressUtils.sol
- contracts_common/src/Libraries/ObjectLib32.sol
- contracts_common/src/Libraries/SafeMath.sol
- contracts_common/src/BaseWithStorage/SuperOperators.sol
- contracts_common/src/BaseWithStorage/Admin.sol
- src/Asset/GenesisBouncer.sol

Notes

The audit was performed on commit `0e8a86c9e9d8f7b39659af537d6a7990428d71bc` and Solidity compiler version `0.5.9`. Follow up was performed on commit `2c32a7b78296e914fe6157fed2ad51b93cc78c99`.

Intended Behavior

The contract implements both ERC-721 and ERC-1155, in order to represent in-game assets.

SOLIDIFIED

Audit Report for Pixowl The Sandbox Assets (ERC721 / ERC1155 token contract) on September 4th, 2019.

Issues Found

Critical

1. Duplicated token Ids are possible

The mint functions allow for the creation of tokens with duplicate ids, if the minter passes 0x0 as the hash for the tokens. This will lead to duplicates ids, which can cause all sorts of unwanted consequences in the system.

Recommendation
Check if the hash passed is different from 0.

Follow up [17.09.2019]
The issue has been fixed and is no longer present in commit `2c32a7b78296e914fe6157fed2ad51b93cc78c99`.

Major

No major issues have been found.

Minor

2. Check for 0 Address in Access Control

The ERC1155ERC721 contract uses a bouncerAdmin role with special privileges. This can be changed using the `changeBouncerAdmin()` function. There are no checks to prevent this new address to be set to address 0 and avoid unwanted changes from erroneous calls.

Recommendation

In general, it is considered a best practice to check addresses provided in arguments for 0. This is particularly true for addresses that are given special roles.

Follow up [17.09.2019]
Pixowl informed us that this is intentional, as there is a plan to delegate the admin function to address(0x0), increasing the trustlessness of the system.

3. ERC1155 receivers can exploit gas usage

All of the tokens standards that require a call to the receiving contract, such as ERC721, ERC777, ERC1155, ERC223, can be exploited by artificially increasing the gas usage, since the calling function doesn't limit the amount of gas passed to an external contract, as the regular Fallback function does. An attacker could implement a functionality for mining gas tokens, for example, in the receiver function, which will be paid by the sender or the relayer.

Since this behavior is defined in the standard, there isn't a recommended action to take, but stakeholders need to be aware of this possibility.

Notes

4. Unify Compiler Pragmas

Different library contracts seem to have been copied from different sources (OpenZeppelin, Loom, etc.). This leads to a variety of compiler pragmas.

Recommendation
We recommend agreeing on a single compiler version and unifying the pragma declarations throughout the code.

Follow up [17.09.2019]
All contracts, except for shared libraries stored in the contracts_common folder, are using fixed compiler version 0.5.9 in commit `2c32a7b78296e914fe6157fed2ad51b93cc78c99`.

5. Avoid magic variables

Especially in the `generateTokenId` there's a lot of hard coded numbers, which could be replaced by declared constants without any drawback. This is helpful for not only for readers but also diminishes the chances of mistakes further in the development process.

Follow up [17.09.2019]
The issue has been fixed and is no longer present in commit `2c32a7b78296e914fe6157fed2ad51b93cc78c99`.

6. Misleading error message

The error message `"supplies > 0"`, on the function `allocateIds`, is misleading as it states the condition necessary for success, as opposed to all the other error messages which indicates the reason for failure.

Follow up [17.09.2019]
The issue has been fixed and is no longer present in commit `2c32a7b78296e914fe6157fed2ad51b93cc78c99`.

7. Use require instead of assert in SafeMath

Safemath is used to validate inputs several times in the codebase, reverting the transactions through a `require` is the appropriate way to revert on input validation, while also reimbursing the unused gas back to the user.

Follow up [17.09.2019]
The issue has been fixed and is no longer present in commit `2c32a7b78296e914fe6157fed2ad51b93cc78c99`.

Audit Report for Pixowl The Sandbox Assets (ERC721 / ERC1155 token contract) on September 4th, 2019.

Closing Summary

The contracts audited contain one critical issue, and a couple of minor issues.

We recommend the issues are amended, while the notes are up to the client's discretion, as they mainly refer to improving the operation of the smart contract.

Follow up [17.09.2019]
All critical/minor issues were fixed and are no longer present in commit `2c32a7b78296e914fe6157fed2ad51b93cc78c99`.

Disclaimer

Solidified audit is not a security warranty, investment advice, or an endorsement of the Pixowl platform or its products. This audit does not provide a security or correctness guarantee of the audited smart contract. Securing smart contracts is a multistep process, therefore running a bug bounty program as a complement to this audit is strongly recommended.

The individual audit reports are anonymized and combined during a debrief process, in order to provide an unbiased delivery and protect the auditors of Solidified platform from legal and financial liability.

Solidified Technologies Inc.