SOLIDIFIED

Audit Report for Yesbit on September 30, 2020

Summary

Audit Report prepared by Solidified covering the Yesbit and KrawCat oracle smart contracts (and their associated components).

Process and Delivery

Two (2) independent Solidified experts performed an unbiased and isolated audit of the code below. The debrief took place on September 30, 2020.

Audited Files

The following contracts were covered during the audit:

- Oracle.sol

Supplied in the repositories:

https://github.com/yesbit/smart-contract-audits

Notes

The audit was based on commit `c1d5dcb80508537ae9ffdc651327497921dced75`.

Intended Behavior

The smart contracts provide oracle solutions for the Yesbit project, in order to supply pricing information on different token pairing. The smart contrat is price oracle that receives updates from an authorized account.

SOLIDIFIED

Audit Report for Yesbit on September 30, 2020

Executive Summary

Solidified found that the Yesbit/KrawlCat contracts contain 0 issues and 4 informational notes.

We recommend all issues are amended, while the notes are up to the team's discretion, as they refer to best practices or optimizations.

Issues found:

Critical	Major	Minor	Notes
0	0	0	4

SOLIDIFIED

Audit Report for Yesbit on September 30, 2020

Issues Found

Critical Issues

No critical issues were found.

Major Issues

No major issues were found.

Minor Issues

No minor issues were found.

Notes

1. Naming collision

There is a naming collision between the function `isHuman()` in `Oracle.sol` and the modifier with the same name. This is acknowledged in the comments.

Recommendation
Consider renaming or removing the function and use the modifier.

Consider removing one of the implementations to simplify the code or reduce dependencies.

2. Tautologies in timing range check

In `Oracle.sol` the functions `setPrice()` and `getPrice()` contain range checks for the time slot:

```
require(_timeslot >= 0 && _timeslot <= 23, "timeslot outOfRange");
```

However, the _timeslot parameter is an unsigned integer, which cannot be negative, making the first check superfluous and confusing.

Recommendation
Remove the `_timeslot >= 0` check.

3. Use of int256 for prices in Oracle.sol

Prices are stored as `int256`. However, they are always treated as positive numbers and range checks are performed to avoid negative numbers. There is no reason to use a signed variable type in this case, as it only serves to introduce unnecessary range checks. It is also inconsistent with the other contracts that use unsigned integers to express prices.

Recommendation
Store prices as `unit256`.

4. Oracle.sol: Unused member variable: ownership

Member variable `ownership` is not being used anywhere inside the Oracle contract.

Recommendation
Remove the unused member variable.

Audit Report for Yesbit on September 30, 2020

Disclaimer

Solidified audit is not a security warranty, investment advice, or an endorsement of Yesbit 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.