SOLIDIFIED

Audit Report for B-Protocol on October 02, 2020

Summary

Audit Report prepared by Solidified covering The Sandbox Estate Sale and Fee Distributor smart contracts (and their associated components).

Process and Delivery

Three (3) independent Solidified experts performed an unbiased and isolated audit of the code below. The debrief took place on September 29th, 2020.

Updates were delivered by the client on September 30th, 2020, and have been marked in this audit report.

Audited Files

The following contracts were covered during the audit:

- src/BCdpManager.sol
- src/BCdpScoreConnector.sol
- src/LiquidationMachine.sol
- src/Math.sol
- src/Pool.sol
- src/BProxyActions.sol

Supplied in the repositories:

https://github.com/backstop-protocol/dss-cdp-manager/tree/audit

https://github.com/backstop-protocol/dss-proxy-actions/tree/audit

Notes

The audit was based on commits `1ab35b40e47343af74bcdab8bdc53eff717d6c75` and `3ec578cb588f11fd6e0b14f8786bd3eda4d234de`, Solidity compiler version `0.5.12`.

Update: Fixes were provided in commit `7a284d3a9884afd3065b1e7846785691b1e027ce`.

Intended Behavior

The B-Protocol smart contracts provide an interface for existing lending protocols (MakerDAO in this current version) aimed at avoiding gas wars during liquidation. This is achieved by the selected set of liquidators (members) that have priority for liquidations and share the liquidation proceeds based on a user scoring mechanism.

Executive Summary

Solidified found that the B-protocol contracts contain no issues, and five 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	5

Issues Found

Critical Issues

No critical issues have been found.

Major Issues

No major issues have been found.

Minor Issues

No minor issues have been found.

Notes

1. Loops over member arrays in Pool.sol

The `Pool.sol` contract keeps track of members in array and loops over them, even in access control modifiers. Whilst the array is very small and block gas issues do not apply, this also has an adverse effect, causing transactions from members located towards the end of the list to be more expensive than the same transaction from a member at the beginning.

On `removeElement()`, all items subsequent to the one being removed are moved. If the order of the array is irrelevant, only the last item could be moved into the position of the element being removed.

Recommendation
Consider using mappings in addition to arrays for storing/checking membership. If the order of the member array is irrelevant, consider amending removeItem so that it just moves the last element into the deleted element position.

2. Consider reducing the scope of `emergencyExecute`

The function `emergencyExecute` allows the owner to perform any action on behalf of the contract. While this may seem like a nice to have tool in case something goes wrong, it is also a potent attack vector if the contract's owner keys are misappropriated. B Protocol mentioned in the audit brief that the intention is to use a N/N multisiig as the owner of the contract.

Recommendation
Consider reducing the scope of the function to calls that should be needed in case of emergency.

3. Consider commenting unused function declaration parameters

For instance, replace the following:

```
    function ilks(bytes32 ilk) external pure returns(uint flip, uint chop,
uint dunk) {
        ilk; //shh
        return (0, 1130000000000000000, 0);
    }
```

With:

```
    function ilks(bytes32 /* ilk */) external pure returns(uint flip, uint
chop, uint dunk) {
        return (0, 1130000000000000000, 0);
    }
```

4. Absence of `quitB` function in `BProxyActions.sol`

BProxyActions.sol does not contain an implementation for `quitB(...)`. It's unclear if this is as intended, though all other functions are implemented (`DssProxyActions` (base) also implements all the functions in `DssCdpManager`), so we are reporting this as an informational note.

Recommendation
Consider implementing a proxy call for `quitB(...)`.

5. Consider reviewing architecture of `LiquidationMachine.sol`

LiquidationMachine could inherit `DssCdpManager` instead of being called externally. The address of the contract remains constant and cannot be changed later on.

The LiquidationMachine uses the `DssCdpManager` just to read urn and ilk data which would also be accessible as the base class properties if the LiquidationMachine inherited from the `DssCdpManager`.

The change will result in code that is more readable and save ~2000 gas for each `man` call.

UPDATE: The client has implemented the recommendation

Disclaimer

Solidified audit is not a security warranty, investment advice, or an endorsement of B-Protocol 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.