G$_0$ Group

Security Review of
Gnosis Omen FPMM

March 13, 2020

Gnosis Omen FPMM / March 2020

Files in scope

Following contracts:

https://github.com/gnosis/conditional-tokens-market-makers/blob/aeea3f8c966ef5691c3edd0ca9f9d7aab2ab8e4f/contracts/FixedProductMarketMaker.sol
https://github.com/gnosis/conditional-tokens-market-makers/blob/aeea3f8c966ef5691c3edd0ca9f9d7aab2ab8e4f/contracts/FPMMDeterministicFactory.sol
https://github.com/fvictorio/realitio-gnosis-proxy/blob/80a43b1f4a14605f79aa72eb56cbca8df3cfdeae/contracts/RealitioProxy.sol
https://github.com/gnosis/contract-proxy-kit/blob/09b6d7f33a0c3bd3ebe32c310dd2d855d2cefb14/contracts/CPKFactory.sol

Current status

As of March 13th all raised issues have been fixed by the developer

Issues

1. Wrong formula for calculating number of newly minted shares in addFunding

type: correctness / severity: major

In `FixedProductMarketMaker` on `line 174` the calculation of number of newly minted shares is incorrectly implemented as `mintAmount = addedFunds.mul(maxBalance) / poolShareSupply` while it should be `mintAmount = addedFunds.mul(poolShareSupply) / maxBalance`.

status - fixed

Issue has been fixed and is no longer present in https://github.com/gnosis/conditional-tokens-market-makers/blob/5426a1fb1277776e4c7d18cd8ef2eb4438fbc843/contracts/FixedProductMarketMaker.sol

2. Wrong order of operations in addFunding

type: correctness / severity: major

In `FixedPorductMarketMaker` on `line 153` collateral is split before `getPoolalances` is called on `line 160` leading to relative size of the contribution being incorrectly calculated also against tokens that have been created from collateral currently being added instead of just against tokens that have been in the pool before `addFunding` was called, leading to undervaluing the size of the investment at the expense of the investor.

status - fixed

Issue has been fixed and is no longer present in https://github.com/gnosis/conditional-tokens-market-makers/blob/5426a1fb1277776e4c7d18cd8ef2eb4438fbc843/contracts/FixedProductMarketMaker.sol

3. Buy fees are higher than sell fees

type: usability / severity: medium

fees that are paid when buying tokens are higher than fees that are paid when selling tokens. Since in case of buy the fee is: `value * fee` and in case of sell it's `value * fee / (1+fee)` where fee is fee expressed as a number between 0 and 1 and value is value of incoming assets.

status - fixed

Issue has been fixed and is no longer present in https://github.com/gnosis/conditional-tokens-market-makers/blob/5426a1fb1277776e4c7d18cd8ef2eb4438fbc843/contracts/FixedProductMarketMaker.sol

4. Buying tokens in bulk yields lower fees

type: usability / severity: minor

The way fees are implemented, the amount of fees paid depends on whether tokens are bought in bulk or the buy is split into smaller transactions. In other words `buy(10, ...) + buy(10, ...)` buys less tokens than `buy(20, ...)`.

status - fixed

Issue has been fixed and is no longer present in https://github.com/gnosis/conditional-tokens-market-makers/blob/5426a1fb1277776e4c7d18cd8ef2eb4438fbc843/contracts/FixedProductMarketMaker.sol

5. Unnecessary requirement to lock surplus collateral

type: usability / severity: medium

Changes to the fee collection originally proposed here:https://github.com/gnosis/conditional-tokens-market-makers/pull/35 have a cumbersome sideffect of requiring all shareholders to lock additional collateral in the contract to cover fees that haven't been paid out yet and that can be only paid out in full when the pool is copletely liquidated. To rectify this, we have proposed following modifications to the `FixedPorductMarketMaker` contract, that allow withdrawal of collected fees at any time:

```
mapping (address => uint256) private withdrawnFees;
function withdrawFees(address account) public {
    uint rawAmount = collectedFees.mul(balanceOf(account)) / totalSupply();
    uint withdrawableAmount = rawAmount.sub(withdrawnFees[account]);
    if(withdrawableAmount > 0){
        withdrawnFees[account] = rawAmount;
        require(collateralToken.transfer(account, withdrawableAmount),
"witdhdrawal transfer failed");
    }
}
function _beforeTokenTransfer(address from, address to, uint256 amount) internal
{
    if(from != address(0)){
        withdrawFees(from);
    }
    uint withdrawnFeesTransfer = collectedFees.mul(amount) / totalSupply();
    if(from != address(0)){
        withdrawnFees[from] = withdrawnFees[from].sub(withdrawnFeesTransfer);
    } else {
        collectedFees = collectedFees.add(withdrawnFeesTransfer);
    }
    if(to != address(0)){
        withdrawnFees[to] = withdrawnFees[to].add(withdrawnFeesTransfer);
    } else {
        collectedFees = collectedFees.sub(withdrawnFeesTransfer);
    }
}
```

status - fixed

Issue has been fixed and is no longer present in https://github.com/gnosis/conditional-tokens-market-makers/blob/5426a1fb1277776e4c7d18cd8ef2eb4438fbc843/contracts/FixedProductMarketMaker.sol