

# Hats Finance / July 2021

## Files in scope

All solidity files in <https://github.com/hats-finance/hats-contracts/tree/46e7a21fd32be2cfd1b9f309f54dbc3b0d5dbcc>

## Current status

All issues have been fixed by the developer. There are no know issues in <https://github.com/hats-finance/hats-contracts/tree/fdf42ebe324332492d6e7b46f6663226435532fc>.

## Issues

### 1. emergencyWithdraw function doesn't account for paid out claims when calculating withdrawable tokens

*type: incorrect implementation / severity: major*

In `HATMaster.emergencyWithdraw` the amount of withdrawable tokens should be adjusted to account for the paid out claims the same way it's done in `HATMaster.withdraw`, otherwise users can withdraw more tokens than they should based on their relative share of ownership in the pool.

*status - fixed*

Issue has been fixed and is no longer present in

<https://github.com/hats-finance/hats-contracts/tree/fdf42ebe324332492d6e7b46f6663226435532fc>

### 2. reward per share in updatePool are calculated using pool token supply as total, but in withdraw it's assumed share supply was used as total

*type: incorrect implementation / severity: major*

In `HATMaster.updatePool` when reward per share is calculated, instead of using the sum of all user shares as the total, pool tokens owned by the pool are used. The amount of pool tokens is likely to be lower than the amount of pool shares, meaning the reward per share amount will be too high and will lead to overpaying of rewards in `deposit` and `withdraw` functions.

*status - fixed*

Issue has been fixed and is no longer present in

<https://github.com/hats-finance/hats-contracts/tree/fdf42ebe324332492d6e7b46f6663226435532fc>

### 3. HATToken contract should implement increaseApproval and decreaseApproval methods

*type: security / severity: minor*

`HATToken` contract should implement methods `increaseApproval` and `decreaseApproval` methods to prevent ERC20 approval attack.

*status - fixed*

Issue has been fixed and is no longer present in

<https://github.com/hats-finance/hats-contracts/tree/fdf42ebe324332492d6e7b46f6663226435532fc>

#### 4. HATMaster.swapBurnSend method is open to a flash loan based price manipulation attack

*type: security / severity: major*

`HATMaster.swapBurnSend` does a uniswap trade on behalf of other users and is callable by everyone, this allows an attacker to pump a token price using a flash loan financed buy, execute `HATMaster.swapBurnSend` making the contract buy at the increased price and then sell the originally bought tokens with profit.

*status - fixed*

Issue has been fixed and is no longer present in

<https://github.com/hats-finance/hats-contracts/tree/fdf42ebe324332492d6e7b46f6663226435532fc>

#### 5. TokenLockFactory.createTokenLock allows anybody to frontrun HATVaults call of the function and block it

*type: security / severity: major*

`TokenLockFactory.createTokenLock` has an issue in that anybody can call it, so a call from `HATVaults` can be frontrun and blocked by creating the lock contract before it can.

*status - fixed*

Issue has been fixed and is no longer present in

<https://github.com/hats-finance/hats-contracts/tree/fdf42ebe324332492d6e7b46f6663226435532fc>