Contracts vulnerabilities

Vulnerabilities list:

Involved contracts and level of the bugs

The present document aims to point out some vulnerabilities in the contracts.

Vulnerabilities

1. Deposit function

Severity: Medium

The following method is implemented in the liquidity_lockbox_v2 program:

```
pub fn deposit(ctx: Context<DepositPositionForLiquidity>,
    token_max_a: u64,
    token_max_b: u64,
 )
```

This method facilitates the deposit of wSOL and OLAS tokens to increase OLAS-wSOL position liquidity enabling users to receive fungible token equivalents.

In the current implementation, the *deposit()* function takes *token_max_a* and *token_max_b* as inputs. The calculation of the liquidity amount is solely based on *token_max_a* at the current pool status, which can lead to lower-than-intended liquidity if an attacker manipulates the pool balance, for example, by acquiring a substantial amount of token B. The calculated liquidity is then used to determine deltas for token A and B, which are subsequently utilized in the *increase_liquidity()* instruction of ORCA's whirlpool program. Specifically, if an attacker manipulates the pool balance, a user depositing *token_max_a*

and a small amount of token B could result in skewed underlying balances and lower liquidity than expected. Selling all tokens back into the pool exacerbates the un liquidity underlying imbalance. To mitigate this issue and enhance slippage control, it is advised to update the deposit instruction. The modified function should align with ORCA's *increase_liquidity()* instruction, incorporating an additional parameter, *liquidity_amount*, for a more robust and secure deposit mechanism.