Flash Loan Oracle Manipulation via Spot Price Oracles

Explainer

Many protocols use **spot prices** (i.e., current on-chain prices) from DeFi pools like Curve, Uniswap, or Balancer to determine the value of assets. However, these prices can be **temporarily manipulated using flash loans**, allowing attackers to create arbitrage opportunities or trick the protocol into mispricing assets.

In this specific case, the `getPrice()` function:

- Pulls spot prices from a Curve pool,
- Takes the **minimum price** among pool tokens,
- Multiplies it by the LP token's virtual price to determine USD value.

If the attacker **lowers one token's price temporarily** (e.g., via a flash loan swap), it drags the minimum down — mispricing the LP token and opening the door for exploits.

Cause

The root issue is **trusting a spot price that can be influenced instantly**, especially when combined with:

- Mathematical operations that amplify the error (e.g., taking the minimum),
- The result being used for asset valuations, collateralization, or lending logic.

Where to Look

1. **Oracle Contracts**:
   - Functions like `getPrice()`, `getLPValue()`, or `getTokenValue()`.
2. **Usage of Spot Prices**:
   - Look for protocols sourcing prices directly from:
     - AMM pools (`getReserves()`, `get_dy()`, etc.),
     - LP token virtual prices,
     - Internal balance ratios.
3. **No TWAP or Delay Mechanism**:

- If prices are pulled **synchronously and used immediately**, this is a red flag.

4. **Aggregated or Min/Max-Based Calculations**:
   - Any time prices are combined (e.g., min(), avg(), weighted sum), manipulation is easier.

Why This Happens

- DeFi protocols are **composable**, so oracles often pull data from AMMs like Curve/Uniswap.
- AMMs are **instantaneously mutable** via large swaps or flash loans.
- If a protocol reads this data **in the same block**, it's reading a **manipulated reality**.
- Attackers can profit by exploiting **time-of-check vs time-of-use (TOCTOU)** conditions.

Recommended Solutions

1. **Use TWAPs (Time-Weighted Average Price)**
   Replace spot price with TWAPs (Uniswap V2/3, Chainlink feeds, or custom TWAPs):
   - Reduces manipulation risk since price is averaged over a longer window.
2. **Incorporate Delay or Oracle Heartbeat**
   Don’t act on the price immediately:
   - Use oracles that update periodically (e.g., Chainlink),
   - Or use price from a block ago or longer.
3. **Bounding Price Changes (Circuit Breakers)**
   Detect if price deviates beyond a threshold from a moving average or last known price:

   ```
   require(abs(currentPrice - previousPrice) < threshold);
   ```

4. **Limit Protocol Actions Based on Volatile Prices**
   For example, only allow limited borrowing or minting if price volatility is too high.
5. **Use Median or Weighted Prices Across Multiple Sources**
   Instead of `min()`, consider `median()` or weights to reduce sensitivity to a single skewed token.