In 2022, over $403.2 million were stolen in DeFi as a result of over 40 price oracle manipulation attacks.

As one of the most common types of attacks in the crypto space together with signature replay attacks, understanding oracle manipulations and how they occur is crucial to being a better engineer and smart contract auditor.

But first, what is a blockchain oracle?

If this is your first time hearing about oracles, you should checkout oursmart contract development and auditing courses on Cyfrin Updraft, completely for free.

In the internet world, anyone can host a website and a server, similar to how anyone can host a node on the blockchain network. However, nodes and servers are not connected to each other; blockchains are closed systems unable to reach the internet's data.

Until oracles.

In the world of blockchains, an oracle serves as a bridge to external systems, such as APIs or other on-chain data. Oracles enable smart contracts to execute actions based on inputs and outputs from the real world or from other smart contracts and chains.

Data is brought into blockchains from external sources through oracles - whose job is to update and expose data so that smart contracts can use the information as triggers to execute blockchain transactions. A type of smart contract themselves, oracles are key to connecting the wider internet and smart contracts across chains.

Oracles and DeFi Protocols

This connection to the real world is particularly important for DeFi protocols, who rely on the on-chain price of an asset in order to function. In DeFi protocols (decentralized finance) specifically, you can imagine any device that delivers the price of an asset (for example, ETH, USD, or BTC) to be considered an oracle.

For example, DeFi protocols might require to know when to liquidate a user's loan. This decision often comes from pricing information fetched from oracles, as they expose the asset's value so that contracts can build on top of it.

Some examples of oracles are: Chainlink, Tellor, or Uniswap.

Note:

Those familiar with Uniswap might be puzzled that Uniswap is on the list. Isn't Uniswap a DEX (decentralized exchange)? Well, DEXes (and other on-chain protocols with liquidity pools) are often treated as oracles, since you can derive the price of an asset based on the ratio of liquidity pools. Uniswap in particular has an extension called "TWAP" (time-weighted average) which is resistant to flash loans, but the price value is inherently lagging and may be more inaccurate. Hint hint though - using liquidity pools as oracles is one of the most common ways to set yourself up for an oracle manipulation attack.

Herein lies the issue.

Oracles and Oracle Price Manipulation Attacks

Oracle manipulation attacks happen when an oracle's price feed is artificially altered.

This manipulation can dramatically affect behavior within DeFi protocols that rely on that oracle for their internal logic. These alterations can create arbitrage opportunities that wouldn't otherwise exist.

Take this example:

1. An attacker manipulates the price of ETH in a liquidity pool.
2. This manipulated pool is then used by an oracle to determine the value of ETH.
3. A DeFi protocol uses this erroneous ETH value to decide when to liquidate a loan.
4. The result? The loan is liquidated earlier than it should be, netting the attacker a tidy profit.

Oracle manipulation attacks occur when we manipulate the information the oracle is sending to the blockchain, thereby impacting the action being executed on-chain to the benefit of the manipulator.

Here is an example of an oracle manipulation attack in code.

How does Oracle Price Manipulation Attacks Work?

Most oracle price manipulation attacks occur through the use of Flash Loans.

Flash loans are a DeFi mechanism which allow users to borrow assets without the need for collateral.

The catch? The funds must be returned to the lender within the same transaction block or the whole operation gets reverted.

Attackers can exploit Flash Loans to alter the price of assets in automated market makers such as Uniswap, changing the spot price of a token before the lender smart contract has a chance to look up the token again.

As all of this happens within the same transaction block, compromised liquidity pools end up distorting the price oracles, which in turn manipulates behavior in a given protocol.

Examples of Oracle Price Manipulation Attacks

Last year, Mango Markets, an online exchange for cryptocurrencies, was attacked through oracle manipulation. $117 million in digital money was taken.

Here's how it happened:

1. Avraham Eisenberg, the attacker, began with $10 million USDC in two different Mango Markets accounts.
2. He used one account to sell a lot of MNGO (Mango's own currency) while his other account bought the same amount.
3. His buying artificially pushed the price of MNGO up quickly on the Mango Markets protocol. The account he used to buy MNGO went from $10M total value to over $400M thanks to the artificially inflated price!
4. With such a high account value, Eisenberg used his MNGO tokens (now worth $400M) as collateral to borrow another token, taking up almost all of Mango Markets' assets since the collateral (MNGO) wasn't actually worth that amount in the market. This caused the price of MNGO to drop, but it was already too late.. Eisenberg had taken all valuable assets!

If we were to implement an attack like this in Solidity, it would look something like:

In this scenario, we may visualize a manipulateMarket

function exchanging ETH for the MANGO token in the following way:

Things to keep in mind

In this case, the attacker, Avraham Eisenberg, openly confirmed he was behind the oracle price manipulation attack, arguing that none of it was illegal.

- However, the SEC filed a charge arguing that Eisenberg's actions might count as market manipulation since he inflated MNGO's trading volume by 2,000% on the day of the exploit, compared to the average price over the previous ten days. The case is still open, and no legal conclusion has been reached on this topic as of yet.
- On the attack itself, it's interesting to keep in mind that the attack here is not to the specific code, but rather to the source of truth, the oracle, which in turn manipulates the protocol's behavior. The code is in fact working as expected.

In the Mango Markets incident, for example, the Mango Markets protocol was leveraging centralized exchanges like FTX as their oracle to define the collateral valuation for its loans.

- How likely a protocol is to be exploited by an Oracle Manipulation Attack is directly correlated with the type of oracle

they are using:

- Protocols using a liquidity pool as their oracle are essentially 99.9% likely to be exploited because of the volatility in prices when leveraging flash loans.
- Oracles like Chainlink, which are decentralized in nature, are significantly harder to break since the attacker would have to manipulate 50% +1 of nodes on a price feed.
- Decentralized exchanges like Uniswap have created their own “oracle” functionality using time-weighted averages of prices (called TWAP) so they are resistant against flash loan attacks. However, the downside is that they are less accurate and are ergo considered a lagging indicator of price values.

How to prevent Price Oracle Manipulation Attacks?

1. Choose the oracle carefully

It's important to know the strengths and weaknesses of different oracle designs, especially how they fit with your specific protocol needs.

- Which data feeds are being aggregated by the oracle?
- Is this data being fetched from centralized exchanges, decentralized ones, or both?
- Which statistical method is being used to aggregate this data into a single output?
- How does the oracle work out dispute mechanisms in case there’s discrepancies between the nodes?

All these questions fit critically into the oracle architecture and must be taken seriously when choosing the right oracle for a protocol. Using a liquidity pool as your oracle guarantees almost 100% that your protocol will get rekt by a flash loan attack, especially when leveraging low liquidity tokens which are easier to manipulate than those with a larger market cap.

1. Have backup systems

Many DeFi projects use a dual oracle system that leverages both (eg.Chainlink Price Feeds) and an on-chain oracle (eg. Uniswap V3 TWAP oracles). This system can switch to another oracle if the first one isn't working properly (like in the case that the data is wrong).

1. Use decentralized oracles over centralized ones

While a centralized oracle can be efficient, the drawbacks are usually bigger than the benefits. Relying on a single entity makes the oracle vulnerable if that specific entity goes offline or is manipulated. Additionally, users have to trust that the centralized party is providing accurate, unbiased data. This centralization also reduces resilience and censorship resistance.

A decentralized oracle service that uses many sources is not completely safe from manipulation, but it's usually too costly to be a problem. Some downsides to decentralized oracles are the additional resources and coordination power needed to operate at scale and reach consensus across multiple independent sources.

Even so, we recommend using decentralized oracles as they are harder to manipulate.

Some examples of decentralized oracles areChainlink, API3, and Synthetix. Examples of centralized oracles are Coinbase and Kraken who expose the price data from their centralized exchange’s order books.

1. Constantly check the oracle’s performance and take protective steps if needed

Don't always trust that the data from an oracle is correct. Instead, check the data regularly. For example, you could write a script that compares the prices from an oracle to other sources and looks for big differences.

If you don't have a backup system, you should have ways to limit the effects of oracle manipulations. One way could be to pause a protocol’s smart contracts if the oracle data source (or the oracle itself) is suspicious.

1. Get an audit

Although audits are not the cure to hacks, bothprivate and competitive audits can significantly decrease the chance of an oracle manipulation attack occurring to your protocol.

If this is of interest to you, make sure to reach out toCyfrin for smart contract audit and get an estimate for your codebase.

Conclusion

Oracle manipulation attacks are a significant threat to the DeFi space today.

While the Solidity code of the smart contracts operates as expected, the vulnerability lies in the reliance on oracles as a source of truth.

By understanding their occurrence and mechanisms, protective measures can be developed to secure DeFi protocols in the future.

If you are concerned about whether your codebase is vulnerable to Oracle Manipulation Attacks,don't hesitate to reach out here for an audit.

Alternatively, check out the ultimate smart contract development learning platform to upskill your smart contract security skills!