Oracles

Digital Assets - Week 7 (Pre-record)

Rhys Bidder
rhys.m.bidder@kcl.ac.uk

KBS, QCGBF

Autumn 2024

Disclaimer 1: Any opinions expressed, errors or omissions should be regarded as those of the author and not necessarily those of KCL, KBS, QCGBF, QCB, CBI, BoE, or ChainLink Labs.

Disclaimer 2: These notes on cryptography are heavily simplified and leave out many important details. For any real-world security applications these notes should not be relied upon. Instead you should consult appropriate security resources and reliable security professionals.

Disclaimer 3: Cryptoasset transactions are illegal in some jurisdictions. Do not violate these or any other laws. I am not promoting the use of crypto in countries where it is illegal in any form and these slides are not a promotion of crypto or an invitation to participate in crypto-related activities in such countries. They are purely for educational purposes.

Disclaimer 4: I currently am an advisor to Chainlink Labs.

The oracle problem

- Blockchains - and the smart contracts stored/running on them - rely on data stored in their ledgers
- This self-contained nature allows the chain to evolve according to its consensus protocols, ensuring correctness with regard to a very narrow set of data and derived applications
- But for a broader set of applications, data originating off-chain (possibly from the 'real world', possibly from other chains) is needed
- For example, for a derivative contract referencing a stock price from the NYSE, there is no way for the smart contract to know the stock price (it is traded and priced off-chain) unless some service brings the data onto the chain
- Alternatively, an insurance smart contract might need to know the temperature, water level or wind speed at a particular location

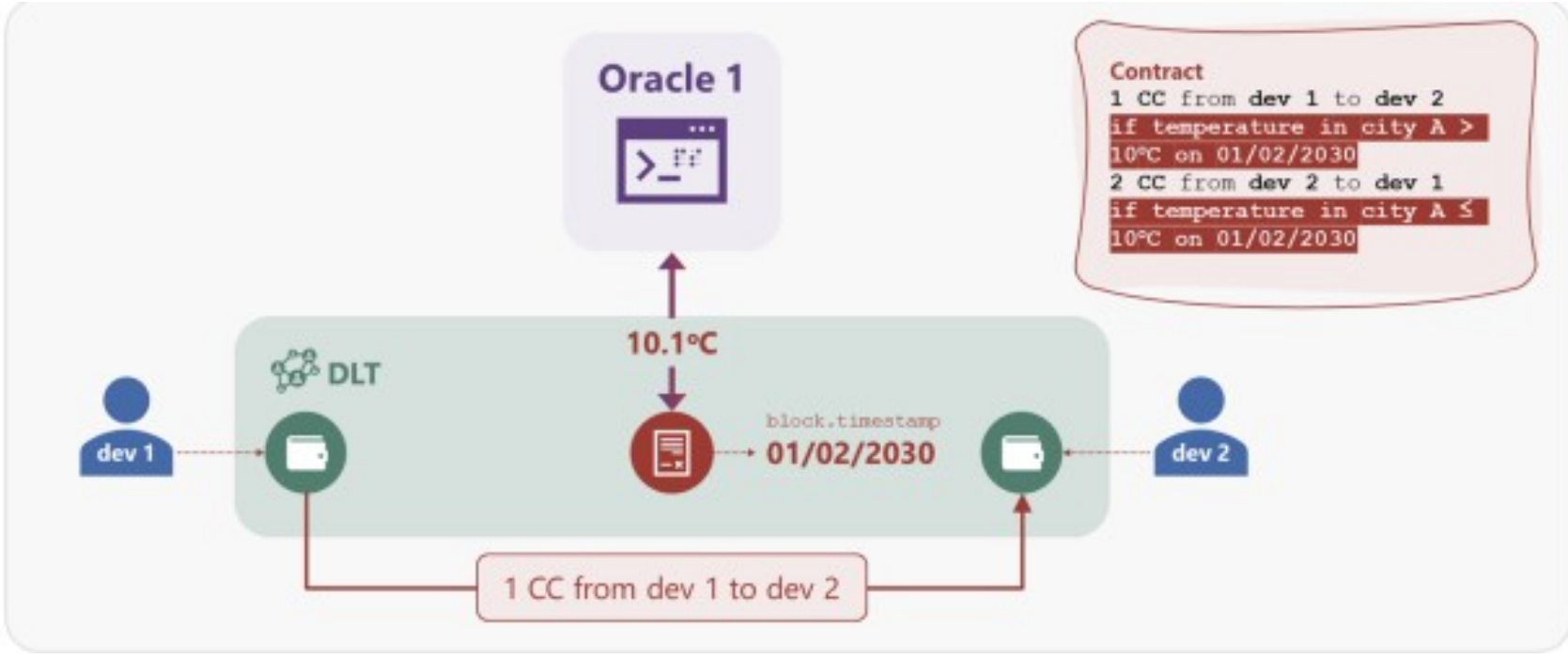
The oracle problem

- The oracle problem is how to introduce this data to the blockchain in a reliable way
  - It may also be desirable for the reverse operation, conveying blockchain information back to the 'real world'
- Oracle services are third parties that collect and disseminate (or at least make available) off-chain data, on the chain
  - They may also do various calculations (simple example - taking median of values from multiple data sources) that would be prohibitively costly on-chain

Oracle use cases

Many interesting examples are listed here and here

- Price feeds (e.g. recall DAI and AAVE needed feeds to continually evaluate collateralization / borrower health)
- Verifiable random numbers (block chains need to operate deterministically on each node, making it difficult to generate true or 'acceptably approximate' randomness)
- Proof of reserves - on and off chain (stablecoin backing, cross chain tokens, assets underlying tokenized RWA. . .)
- Automation of smart contract activities (may rely on real world devices, IoT, data from legacy systems - remember wholesale CBDC 'trigger' mechanisms)
- Cross chain interoperability (L2s, rollups, sidechains, and connections between L1s)



Oracles and smart contracts. *Source:*BIS, Sep 2023

Oracle formats

- Some oracles (e.g. the original data feed offering from Chainlink provide 'push' services, whereby data is supplied to the blockchain at intervals that depend on:
  - *Heartbeats:* Time since last fresh value obtained
  - *Deviation triggers:* Has the last value deviated substantially from value on some other source (e.g. on a CEX)
  - In more (less) volatile periods the deviation trigger (heartbeat) will typically determine the feed
  - Data arrives on chain even when not explicitly requested (implying gas fees that ultimately need to be funded somehow)
- Others (e.g. Pyth and Chainlink data streams) operate a low latency 'pull' model
  - Especially suited to uses requiring high frequency data (e.g. pricing derivatives, arbitrage)

Example of a Chainlink datafeed

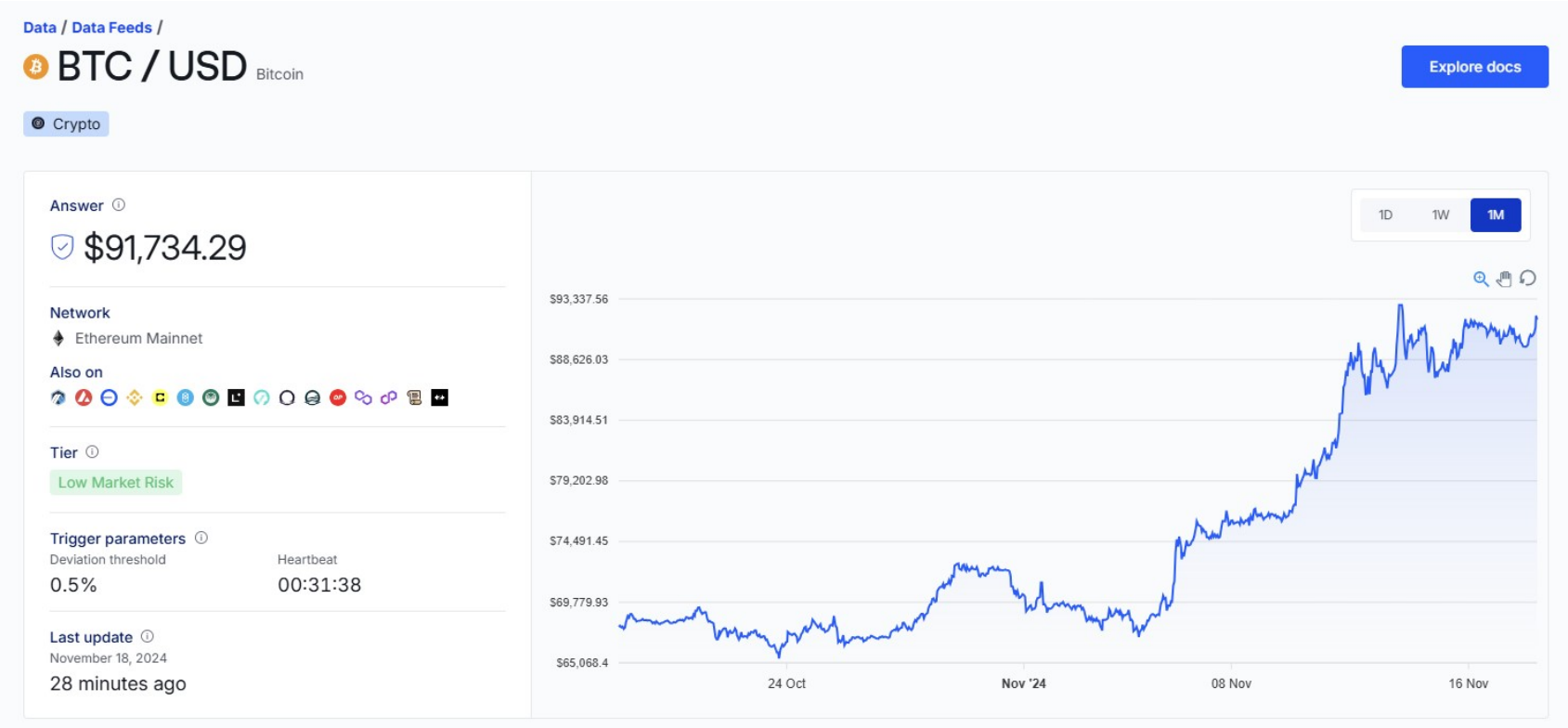
- A simple example of Chainlink data feed usage is found here
  - Deploy it yourself and interact with it using Remix!
  - A pyth example is here
- There are various contract addresses for different data
  - This example shows how to obtain a price of BTC in USD, using a simple smart contract
  - When deployed, the Smart contract establishes a data feed object that interacts with an 'aggregator' contract
  - The aggregator adddreess on Sepolia is: 0x1b44F3514812d835EB1BDB0acB33d3fA3351Ee43
  - As usual, you can examine this contract's activity on Etherscan

```
contract DataConsumerV3 {
    AggregatorV3Interface internal dataFeed;

    /**
   * Network: Sepolia
   * Aggregator: BTC/USD
   * Address: 0x1b44F3514812d835EB1BDB0acB33d3fA3351Ee43
   */
    constructor() {
        dataFeed = AggregatorV3Interface(
            0x1b44F3514812d835EB1BDB0acB33d3fA3351Ee43
        );
    }

    /**
   * Returns the latest answer.
   */
    function getChainlinkDataFeedLatestAnswer() public view returns (int) {
        // prettier-ignore
        (
            /* uint80 roundID */,
            int answer,
            /*uint startedAt*/,
            /*uint timeStamp*/,
            /*uint80 answeredInRound*/
        ) = dataFeed.latestRoundData();
        return answer;
    }
}
```

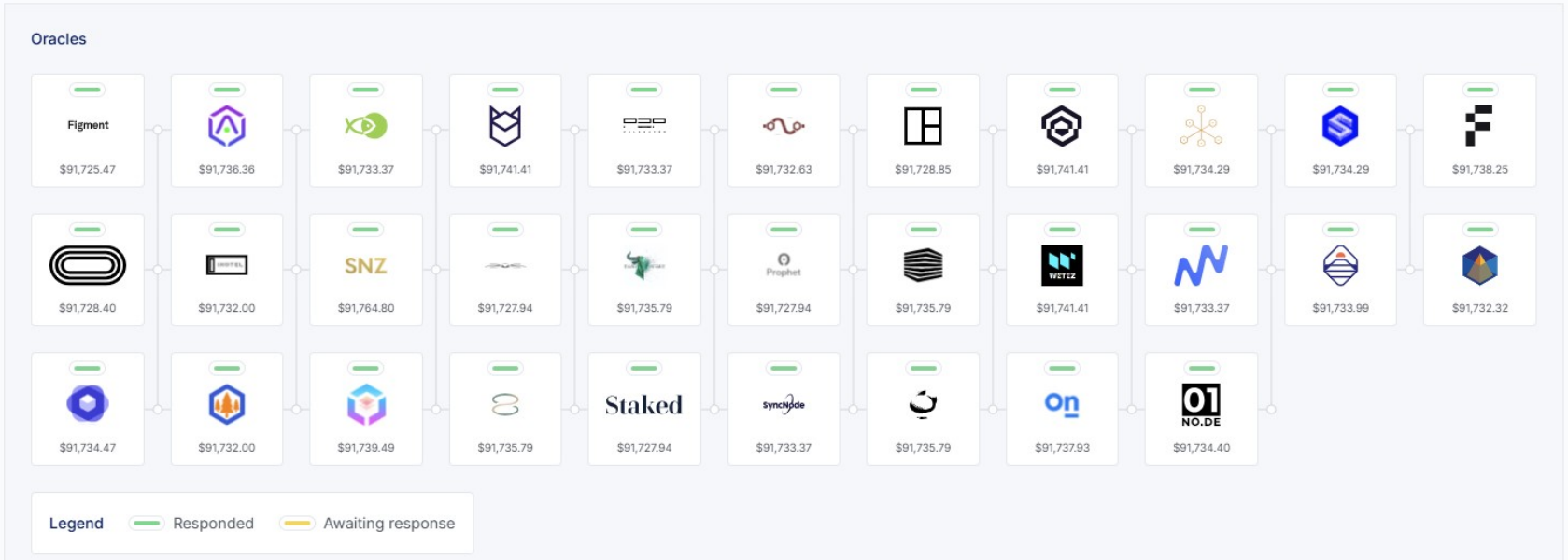
Smart contract for basic use of a chainlink data feed. *Source:* Chainlink - Using data feeds

Sources of data - Chainlink

- Chainlink operates by allowing the permissionless establishment of decentralized oracle networks (DONs)
- Nodes in the network (note these are different from nodes in the *blockchain* network) obtain an - hopefully - validate/process data
- The protocols of the DON then outputs a 'single answer' (perhaps by taking medians, or adjusting for volume if price feeds come from exchanges) that can be used in smart contracts



Ethereum mainnet feed for BTC/USD. *Source:* Chainlink



Oracle responses ⓘ
Minimum of 21
31 / 31

Oracle	Latest answer	Date	Details
01Node Responded	$91,734.402683	November 18, 2024 at 08:00 UTC	
Alpha Chain Responded	$91,736.3591925	November 18, 2024 at 08:00 UTC	
Artifact Responded	$91,735.78908161	November 18, 2024 at 08:00 UTC	
Blockdaemon Responded	$91,741.41	November 18, 2024 at 08:00 UTC	

Oracles for Ethereum mainnet feed for BTC/USD. *Source:* Chainlink

Sources of data - Pyth

- Pyth relies on 'first-party data from major financial institutions'
  - *Examples:* Jane Street, CBOE, Binance, and OKX
- Obvious (massive) centralization and censorship issues, but maybe financial market participants are ok with that?
  - Less elaborate incentivization/safety mechanisms required by the protocol
  - Data already (hopefully) standardized - less need for processing
  - Helps with speed
- Contrast with Chainlink - though arguably less reliable in broader contexts
  - Recall Ethereum (and L2s) ideally provide a general 'world computer'
  - Data streams now available, for lower latency

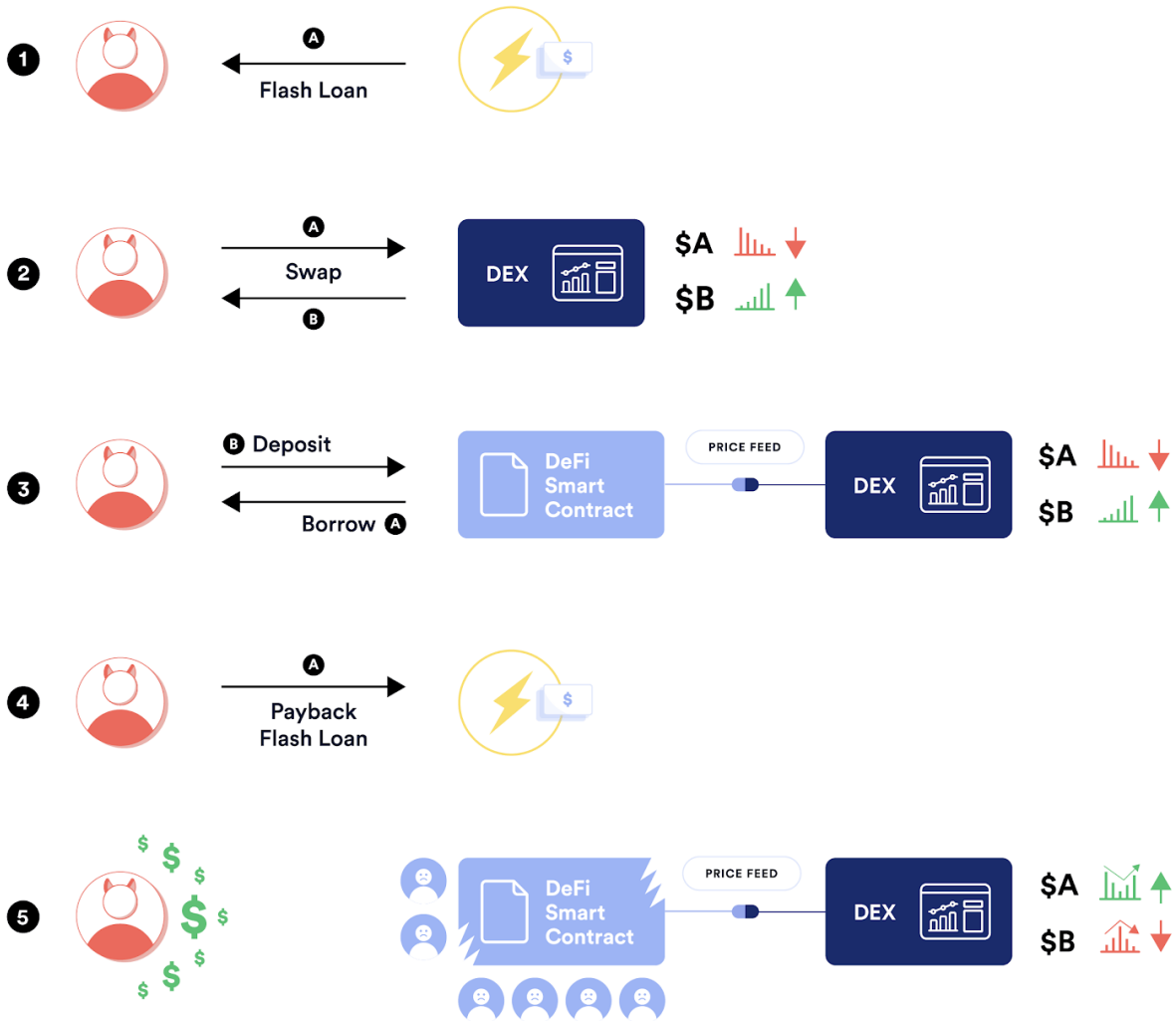
Oracle fragility

- Reliance on oracles can be a source of fragility
  - Are they competent?
  - Are they honest?
  - Can they be manipulated?
- An obvious model is to have a single/small set of oracles
  - But centralized oracles are an obvious target for attacks
  - According to Chainalysis (and see this excellent reading), in 2022 DeFi protocols lost $402m in 41 distinct oracle attacks
- One type of attack is to manipulate prices on exchanges (e.g. obscure/illiquid AMMs), which then influence other protocols whose oracles get price data from those exchanges
  - Recall our discussions of 'slippage' and liquidity in AMMs and the ability to make huge uncollateralized plays with 'flash loans'
- Some oracles used to derive price feeds from a single DEX

Example attack - I

- Here is a simple example of a flash loan-funded attack
  1. Attacker borrows a large amount of token A from a protocol supporting flash loans (e.g. AAVE)
  2. Attacker swaps token A for token B on a DEX (lowering the spot price of token A and increasing the spot price of token B on the DEX)
  3. Attacker deposits the purchased token B as collateral on a DeFi protocol that uses the spot price from the above DEX as its sole price feed, and uses the manipulated spot price to borrow a larger amount of token A than should normally be possible
  4. Attacker uses a portion of borrowed token A to fully pay back the original flash loan and keep the remaining tokens, generating a profit using the protocol's manipulated price feed
  5. As the spot prices of token A and B on the DEX are arbitraged back to the true market-wide price, the DeFi protocol is left with an undercollateralized position

Example attack - I



Example attack - II

- Another (real world) example is discussed here - an attack on the Compound protocol
  - Similar to AAVE, Compound is a collateralized lending model (though allows mult asset pools)
- Remember that if a borrower's position becomes under-collateralized (relative to an approved LTV), they can be liquidated
  - Very costly for the borrower
  - Liquidator repays loan, receives a portion of collateral + bonus
- Attacker manipulated (pushed up) the price of DAI on the Coinbase Pro exchange
  - Compound used oracles that passed this price on to pools where DAI had been borrowed
  - LTV essentially plummeted - allowing liquidation
- One **downside** of one of blockchain's strengths - 'composibility' (at least while defi is still buggy. . .)

Responses to oracle fragility

- Possible responses:
  - Centralize and rely on trusted third parties (like Pyth)
  - Try to reduce/eliminate manipulation in a decentralized manner (more like Chainlink)
- Reduce reliance on single block-based prices
  - Time Weighted Average Price (TWAP - nice note from Uniswap V2)
  - Partly a response to flash loan attacks
- Reduce reliance on single sources
  - Volume Weighted Average Price (VWAP - note comparing TWAP vs VWAP)
  - Rely on, say, larger and more liquid exchanges that are hard to manipulate
- But with multiple sources, how can the system be relied upon if permissionless?

DONs

- Chainlink introduced a design to leverage strategic incentives among oracles
  - DON nodes are compensated in LINK tokens for good behavior (providing info and also in raising 'alerts' regarding bad behavior)
  - They must stake LINK to participate
  - There are penalties if their performance / actions are inadequate (e.g. reporting inaccurate or unreliable data)
  - Information on node performance is also made available - allowing 'reputation' to be built
  - Some guardrails (traffic light ratings for feeds) are provided - but ultimately *caveat emptor*