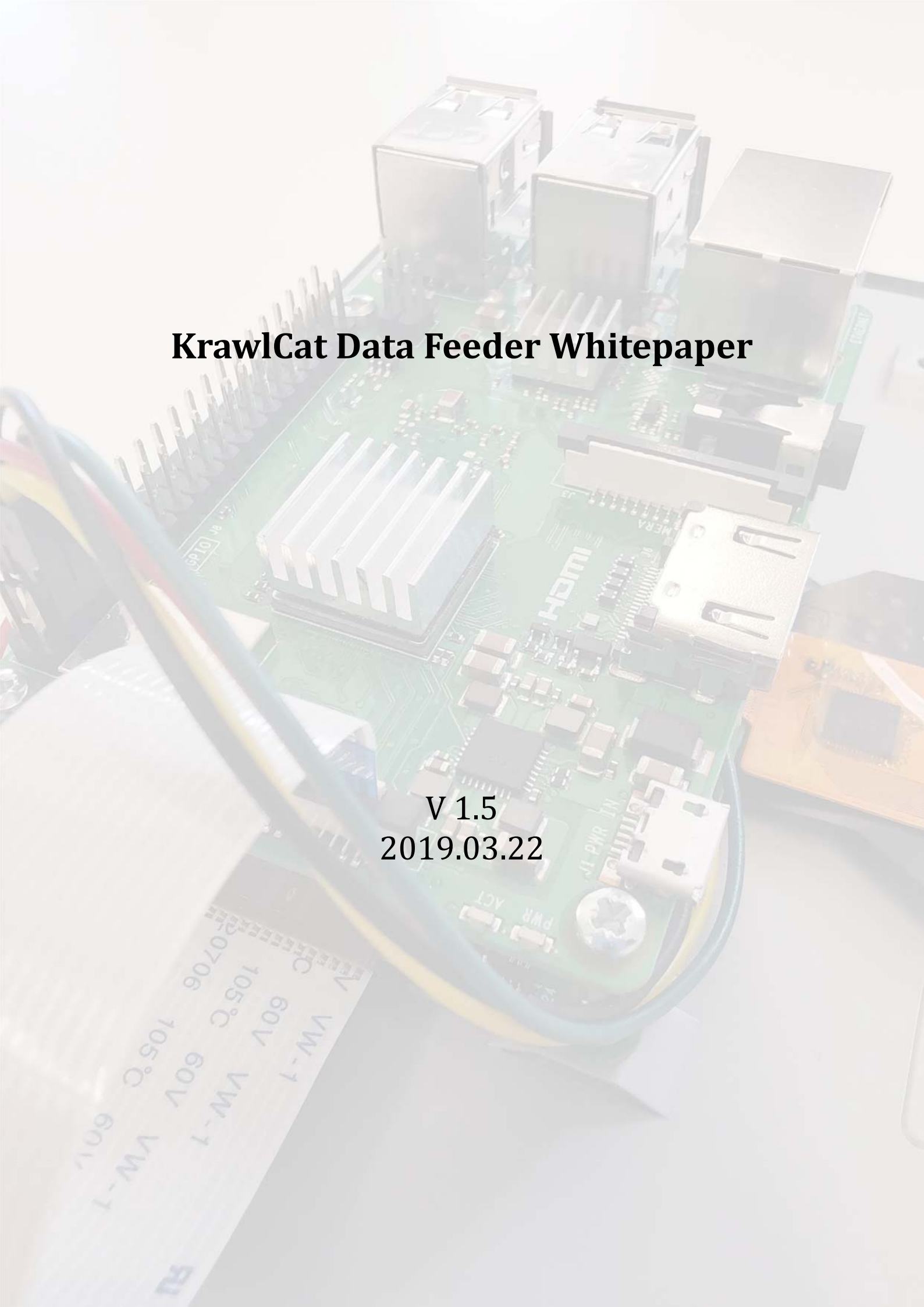


KrawlCat Data Feeder Whitepaper

V 1.5
2019.03.22

1. Problems:

There have been many attempts to integrate blockchain technology with existing industries like supply chain, cloud storage and fin-tech. However, all of these attempts have come short because there is no way to reliably bring off-chain data onto a blockchain. While a blockchain system is safe, the data vendors responsible for providing real-world intelligence to these blockchains are not. With centralized companies responsible for the curation and distribution of this data, there has yet to be a decentralized and trust-less approach to providing off-chain data to a blockchain.

Besides the issue of centralization, these data vendors have a host of problems that affect the fundamentals of any blockchain business that needs off-chain data. Most notably, current data contributors are not compensated for their efforts, and the inefficient uploading speeds of these centralized vendors cannot meet a blockchain's transaction requirements.

As the development of decentralized applications continues, these projects need a trust-less way to bring-off chain data onto their applications. Otherwise, the public blockchains that are hosting these applications will never be able to disrupt the industries that centralized businesses are dominating.

Let's start with an example that contextualizes the value of decentralized data feeds:

- A Decentralized Application (Dapp) needs to quote the bitcoin price. But companies that list the Bitcoin price like Coinmarketcap are off-chain. So now, the company will need to check the price from an off-chain resource like the aforementioned Coinmarketcap or an exchange. However this poses several risks to the entity taking this information because:
    - The programmer could fake the price.
    - That exchange may have lower volume relative to the rest of the market; leading to inaccurate pricing.
    - The Dapp must use a server to keep updating the price which could break down.

1.2 Competitive Analysis:

The need for a decentralized data feed was not discovered recently. Not being able to reliably bring off-chain data to blockchains has always been one of the strongest inhibitors to adoption. As such, there have been several attempts to create a decentralized data feed. We will be analyzing some of this competition and highlighting their flaws as a truly decentralized data source:

- Oraclize: Oraclize's decentralized solution works in three steps:
    - You send a query to the Oraclize smart contract

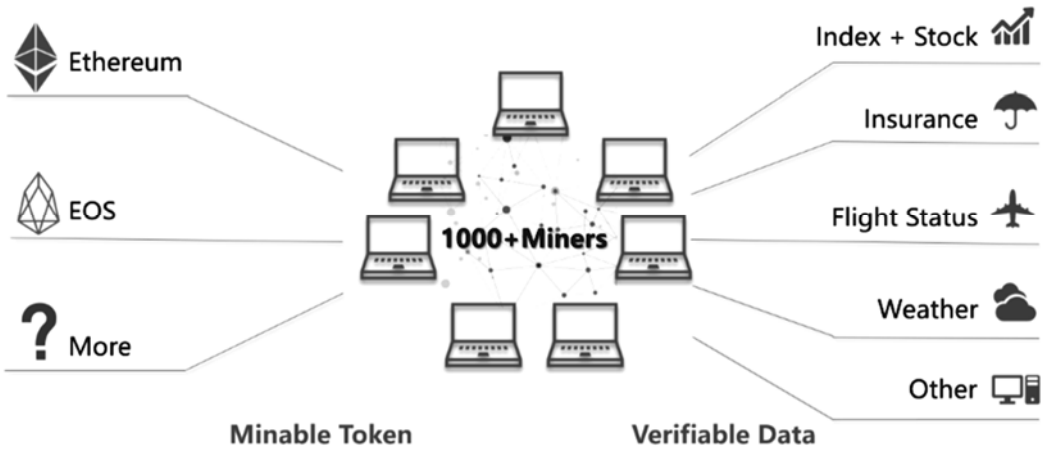
- Oraclize receives your query and makes the relevant request
- Once they receive the data, they'll send a callback function to your smart contract where you'll be able to access the requested data.

The main issue with this process is, although they can provide off-chain data to a blockchain, a centralized entity is still performing this task. Relying on centralized entities to provide the data goes against the ethos of blockchain and decentralization; you are still putting your trust in an external entity. These vendors can compromise the data before it is even uploaded onto a blockchain.

- StreamR: StreamR is an off-chain solution for providing real-world data to blockchain projects. It works similarly to a stock-exchange or advertising-exchange in that anyone can purchase data from vendors in a real-time marketplace.

  The issue with StreamR is that, although their solution offers variety, it lacks the democratic voting mechanism that a truly decentralized solution requires. For a system like StreamR, it is still up to the data-receiver to do their due diligence and make sure they're purchasing the data from a reliable vendor.

2. Solution:



The KrawlCat Data Feeder (Feeder) will break through this innovation bottleneck and provide a decentralized and trust-less data-gateway for blockchain applications to access real-world intelligence.

The Data Feeder does not provide hashing power to blockchains. Instead, Feeders scrape data from credible vendors in finance, weather forecasting, aviation, exchanges, and any other industries that have numerical or categorical data. Together the Data Feeders ensure the curation of this data is decentralized, and that the results cannot be manipulated after the fact.

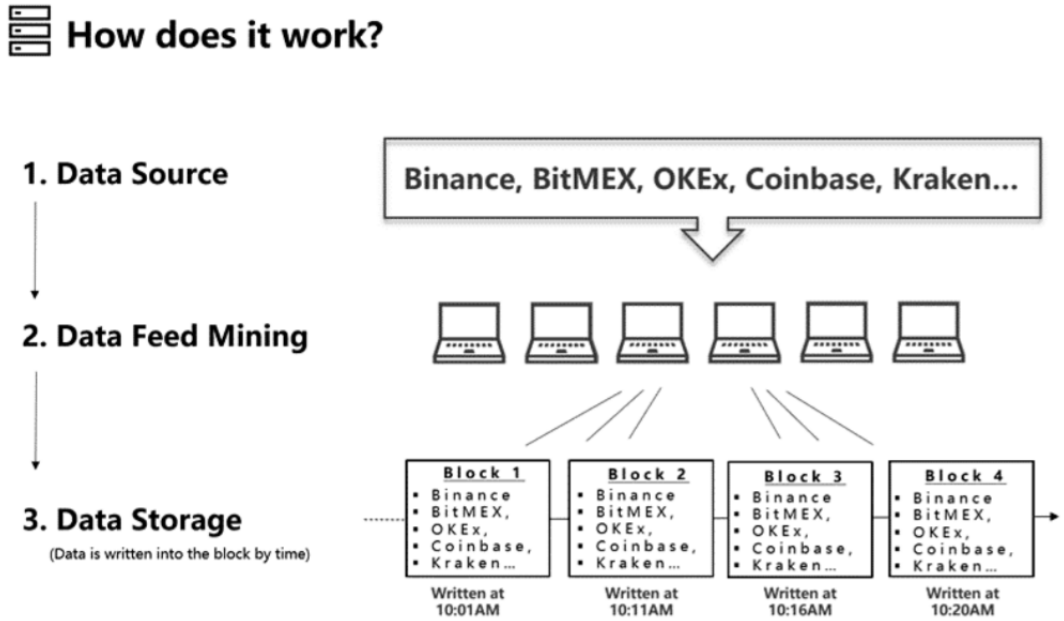
3. Product

KrawlCat Data Feeder is a scraping device for gathering data on the internet, and uploading it to blockchains in real-time. The more Feeders that make up this data feeding network, the more reliable the data would be. Once the network has a sufficient amount of Feeders in the ecosystem, then the Data Feeding process would be considered decentralized and trust-less since the majority approves the truth.

All data feeders are supported by KrawlCat Data Feeding Smart Contract, which acts like a Data Feed Marketplace (marketplace contract), where Feeders choose what data to feed into the blockchain.

Back to the scenario in Section Problem, if that DApp can't found a reliable Bitcoin price data source, then let's see what the process will look using KrawlCat Data Feeder.

> Every Feeder will get the price info from multiple exchanges' APIs, each feeder device will upload the price info to the Marketplace contract to calculate the "most moderate bitcoin price" among thousands of data-submissions from the Feeders. Now the DApp can get this decentralized Bitcoin price info from blockchain.



Although this example illustrates The KrawlCat's ability to fetch financial data, the device is also equipped to fetch numerical or categorical data from Bloomberg, historical NASDAQ Index Google Flight, and Yahoo Weather and virtually any reputable data vendor.

What are users mining?

This process is similar yet different from Bitcoin or any other Cryptocurrency mining. The KrawlCat uses computing power to scrape data from the internet and upload the data to the Data Feed Marketplace Contract, which aggregates all the data and calculates the most moderate data using a Proof of Data consensus. There will be thousands of Feeders working in the network, and acquiring data independently. The Feeder also do hash validations, but the validation only happens when the Feeder tries to upload data to the blockchain, so there is little need for massive computing power.

This process is different from Bitcoin or any other mining that uses a Proof of Work consensus protocol. The KrawlCat uses computing power to scrape data from the internet and upload the data to the Data Feed Marketplace Contract, which aggregates all the data and calculates the most moderate data using a Proof of Data consensus. There will be thousands of Feeders contributing to the network and acquiring data independently. The Feeder also performs hash validations However, the validation only occurs when the Feeder is uploading data to the blockchain, so the device does not require massive amounts of computing power to complete this operation.

Decentralized Scraper Network

If we analyze each of these Feeders as a singular entity, then none of these devices could be considered as a “trust-less data provider”. However, when there are 1,000+ Feeders continuously working together, we can see the decentralized system take hold. Now we have a comprehensive network of devices all working together to prove one thing, determine what the most reliable data-set is for a given-task.

Within the context of The KrawlCat Data Feeder, the idea of “decentralization” is that every Feeder is trying to scrape off-chain data to the blockchain independently, and the “acceptable range” represents the Feeders most widely accepted data-set.

3.1 Data Feeder Device

The KrawlCat hardware comprises of a Linux-based microcomputer and a 7-inch touchscreen. Linux is safe against malware, requires little maintenance, and requires no installation; just connect the device to a power source and you are ready to go. There are 4 major advantage of KrawlCat Feeder Device:

- **Easy to use**
  KrawlCat engineers have hardcoded the entire information collecting program into the device, so all you need to begin using the device is a power source and an internet connection.

  All devices come equipped with a friendly UI design and intuitive touch screen. So no prior technical skills are needed to use the device. You can view and change the device's settings by simply tapping the screen.

  More details on the devices function and aesthetic can be found in the *Feeder User Guide*.

- **Built-in Wallet**
  Each Feeder has a built-in wallet: Far too often are people leaving their digital assets on exchanges, and thinking they are still the custodian for these funds. In reality, this just shows you your balance, but this is not the same as having custody of that balance. The Data Feeder has a built-in hardware wallet, so users can send and receive funds at anytime, while actually possessing custody over these assets. The wallet is based on Linux, and comes with anti-virus software.

- **Auto Updates**
  Networks can undergo system updates at anytime. If you are not online at the time of this update, this can be very problematic. The Krawl Cat has a auto-update function so users never have to worry about running obsolete software. Currently the Feeder only supports the Ethereum network, but we are working diligently to launch on the EOS network. KrawlCat's engineers will continuously be updating the software to ensure that the Feeder's firmware is always up-to-date.

- **Low Energy Consumption low noise**
  According to the Proof of Data consensus, The KrawlCat doesn't require a substantial amount of computing power to scrape data. Each Feeder is just collecting internet statistics and uploading the statistics to the Data Feeder Smart Contract. The energy consumption for this task is 3-5W, compared to a regular mining devices energy consumption of 1350W. In addition, the fan inside the Feeder generates only 20dB in noise, so the device is almost silent, even when it is running at full capacity.

3.2 Consensus and Implementation

The KrawlCat uses a Proof of Data and Proof of Stake hybrid consensus mechanism for the reliable distribution of this off-chain data.
This section also introduces the key mathematical principles that support the KrawlCat's decentralized architecture.

Proof of Stake

The KrawlCat network will require an enrollment fee of 5,000 Token to join the network.

As a security measure, the Data Feeder Smart Contract has been coded to check the balance of each Feeder. Only if the balance is more than 5,000 tokens can the Feeder be accepted to the network. This high upfront cost was designed to dissuade hackers from trying to compromise the KrawlCat network.

The ownership of Tokens does not directly generate new revenues for miners. Instead, all monetary compensation for the network's miners will come from subscription fees accumulated from businesses paying to access the KrawlCat network.

Proof of Data

During every processing cycle, two data selections will be made based on the mining procedure. Please refer the details stated in Section 3.4.

1. Each Feeder will get the same data-sets that is ordered from data buyers. If we wanted to retrieve the data on the price of Bitcoin, then we would scrape this data from 6 different exchanges. The Feeder will then send this price info to the smart contract.
2. Secondly, the smart contract will generate a "moderate range", so that Feeder contributions are only valid if their submitted data-set falls within this "moderate range". Thousands of Feeders are working simultaneously and getting fed statistics from our list of data vendors. So once the smart contract collects these numbers from the mining pool, the smart contract will filter out submissions that are outside of this reasonable range, and pick a median value from the values that are within this "moderate range". Whichever feeders that provided valid data will be rewarded, while feeders who submit invalid data will be penalized.

Core Math for Decentral Design

The Feeder network is decentralized. Which means, any feeder can send requests to smart contracts, as well as leave or join the network at any time; all without compromising the network's efficiency. Due to a decentralized network's fluid

nature, tracking this system in a linear fashion, with a variable such as time is too complex.

Let's analyze this problem through this example: Buses are scheduled by time. If the bus company wants to know the pick-up flow a bus station, this relationship will be shown through time:

8:00am bus arrives, picks up 5 people
8:10am bus arrives, picks up 3 people
8:21am bus arrives, picks up 7 people (1-minute delay)
8:31am bus arrives. picks up 4 people

In the aforementioned example, all buses are following a 10-minute cycle. But what if buses come randomly? In decentralized system, buses (nodes) can arrive at the station (network) whenever they want, some drivers (nodes) can even choose to quit the job (leave the network). So, a less linear bus schedule might look like this:

8:00am bus arrives, picks up 5 people
8:27am bus arrives, pick up 10 people
9:02am 3 buses arrive, pick up 8 people, 5 people. 7 people respectively
11:59am 2 buses arrive, pick up 21 people, 32 people respectively.

The bus company still wants to track the pick-up flow of all of the buses in their fleet. However, time is no longer an accurate variable because buses can come whenever they want. Time is also an inaccurate variable because since buses come whenever they want, we should be tracking the buses based on how many passengers they pick up, and not if they make their pick-up point on time. Thus, sorting these buses categorically might be a more insightful metric:

Bus 1 picks up 5 people
Bus 2 picks up 10 people
Bus 3 picks up 8 people
Bus 4 picks up 5 people
Bus 5 picks up 7 people
Bus 6 picks up 21 people
Bus 7 picks up 32 people

This concept is known as a harmonic series. The KrawlCat smart contract relies on the concept of Harmonic series as a tool for managing upcoming requests and calculating the balance of each inbound and outbound user.

Mathematical Representation of a Harmonic Series:

$$\sum_{k=1}^{\infty} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots$$

Since all requests made in a decentralized environment happen sporadically, we will not be using time as a variable to count transactions made on the KrawlCat Smart Contract.

Instead, we will monitor transactions made in the pool based on the number of incoming and outgoing miners in the network. The network will re-check the balance of all users in the network, every time someone enters or exits the pool. These triggerable events create what is called a “state”. A state occurs every time someone has entered or exited the networks. By using these states, we are now able to use network participation as the benchmark for determining the balance of those participants in the network, and therefore the balance of the total network.

The goal is to calculate the balance of each user, to tell us the balance of the network in its entirety. So, whenever an event happens, it will trigger a balance change in the entire smart contract. We will define the numerator as the number of claimable tokens for the miners. While the denominator means the number of miners at the time of a triggerable event, and when this “state” was recorded. Thus, it is clear that token for miners to claimnumber of miners means the reward for each miner at the time a triggerable event was activated and a new “state” was recorded.

We call the sum of stated tokens for miners to claimnumber of stated miners the Round Mask. The round mask is simply a historical snapshot of the networks balance, before the most recent “state”

Each feeder entering and existing the system will cause a change to the number of miners and claimable tokens in the network, so we need to record a new state every time one of these triggerable events occurs.

$$\text{RM} = \text{Previous RM} + \frac{\text{token to claim 1}}{\text{number of miners 1}} + \frac{\text{token to claim 2}}{\text{number of miners 2}} + \frac{\text{token to claim 3}}{\text{number of miners 3}}$$

Trigger1	Trigger2	Trigger3
Feeder 207 enter the network	Feeder 792 enter the network	Miner 207 left the network

If Feeder 207 joined the network before the next two inputs, then we call this trigger 1. Before the next Feeder enters or exits the system, Feeder 207 is continuously getting rewards. After a short period, Feeder 792 joins the network, this triggerable event will be called trigger 2. Because the total number of Feeders has changed, the average reward will change accordingly. Next, miner 207 left the network, initiating trigger 3. At event 3 Feeder 207 left the network, so the token rewarded to Feeder 207 is from his efforts during triggers 1,2 & 3.

Formulae

First, we describe the system a bit. If the last time there was a release of tokens was block b_n, and now it is b_j block, then, if a mint is triggered, then we would know $(b_j - b_n) \times \textit{token/ block}$ tokens is released. Since we would want a certain amount of token to be release at a fixed for every period of time.

Rewards Math

Then we define TPR (token per release) or also called PPK, as follows:

$$TPR_n = (b_{n_{start}} - b_{n_{end}}) \times \frac{token}{block}$$

note that $\frac{token}{block}$ is defined as a constant in the system, FYI.

Round Mask:

Round Mask is the series that describes the historical states of the token releases, until the N-th release of tokens, while there are M_N total miners. In other words, Round Mask stores all the historical data that describes that in each release, how many tokens did each miner get (note that the split of tokens between miners is an even split):

$$\begin{aligned} R_M &= \frac{TPR_1}{M_1} + \frac{TPR_1}{M_2} + \frac{TPR_3}{M_3} + ... + \frac{TPR_N}{M_N} \\ &= \sum_{i=1}^{N} \frac{TPR_i}{M_i} \end{aligned} \tag{1}$$

Player Mask:

The Player Mask of the user is the sum of all previous TPRs. This is to make it easy to calculate earnings, as up until current block is the current RM subtracts the user's PM, since user is present and thus is part of the miners during all the in-between token releases includes his/her share. If the user joins after J.

$$\begin{aligned} P_M &= \frac{TPR_1}{M_1} + \frac{TPR_2}{M_2} + ... + \frac{TPR_J}{M_J} \\ &= \sum_{j=1}^{J} \frac{TPR_j}{M_j} \end{aligned} \tag{2}$$

Now the user may choose to withdraw his/her earnings up to now. This means that his mask will have to be modified so data is up to date and accurate. (assuming system is at R_{MN})

$$\begin{aligned} Earnings &= R_{MN} - P_{Mold} \\ &= \frac{TPR_J}{M_J} + \frac{TRP_{J+1}}{M_{J+1}} + ... \frac{TPR_N}{M_N} \\ &= \sum_{i=J}^{N} \frac{TPR_i}{M_i} \end{aligned} \tag{3}$$

$$\begin{aligned} P_{Mnew} &= R_{MN} \\ &= \frac{TPR_1}{M_1} + \frac{TPR_2}{M_2} + ... + \frac{TPR_N}{M_N} \\ &= \sum_{i=1}^{N} \frac{TPR_i}{M_i} \end{aligned} \tag{4}$$

3.3 Smart Contract Structures whitelist→Prove

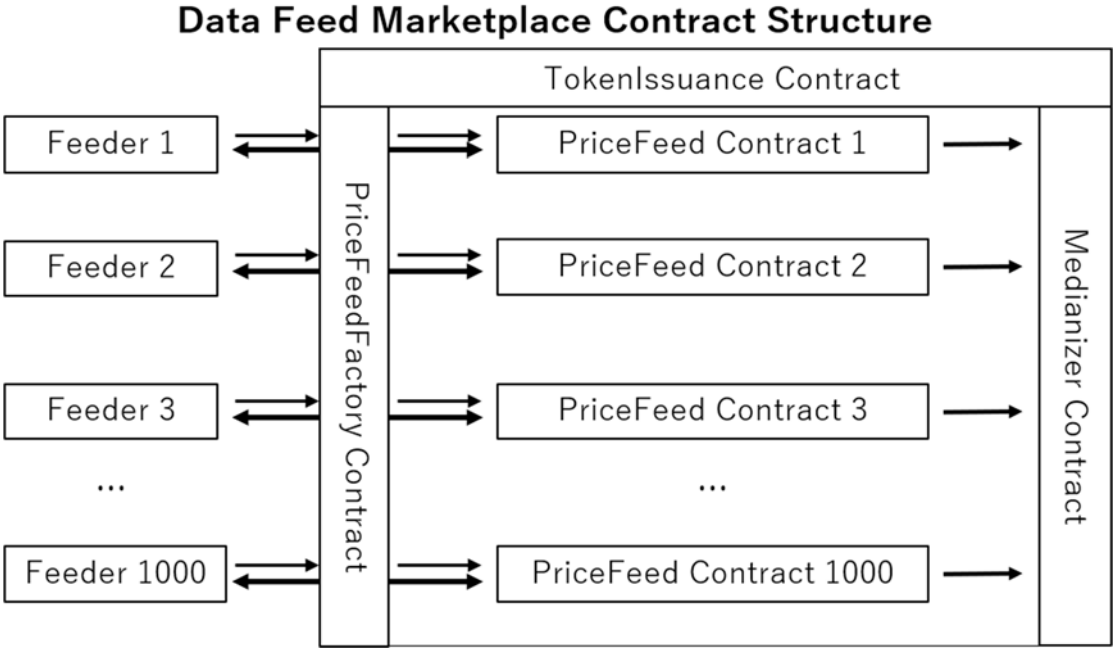
KrawlCat Data Feed Marketplace Smart Contract consist of 4 major contracts:

1. **TokenIssuance Contract:**
   The Token-Issuance contract defines the token distribution amongst all of the Feeders in the network. The Token-Issuance contract acts like a bank in that the contract tracts the balance of all users and is responsible for the distribution of incoming mining profits.

2. **Medianizer Contract:**
   The Medianizer possess a list of wallet addresses that have been approved to join the mining pool. The accepting criteria is whether the Feeder has 5,000 tokens in their Feeder-wallet. Medianizer also sorts through the networks incoming data-submissions, and then finds the median based on these submissions.

3. **PriceFeedFactory Contract:**
   To start mining, new Feeders need to call the Price-Feed-Factory contract to receive a designated PriceFeed contract that corresponds to their specific device. Each mining device will always have a corresponding unique Price-Feed contract. To gain access to the Price-Feed-Factory Contract, Feeders will have to stake 5000 tokens.

4. **PriceFeed Contract:**
   Each Feeder uses their own PriceFeed contract to store the data collected on their device, and then upload the results to the Medianizer. To gain access to the Price-Feed Contract, Feeders will have to continue staking the 5000 tokens that they used to gain access to the Price-Feed-Factory contract.



Design of Marketplace Smart Contract

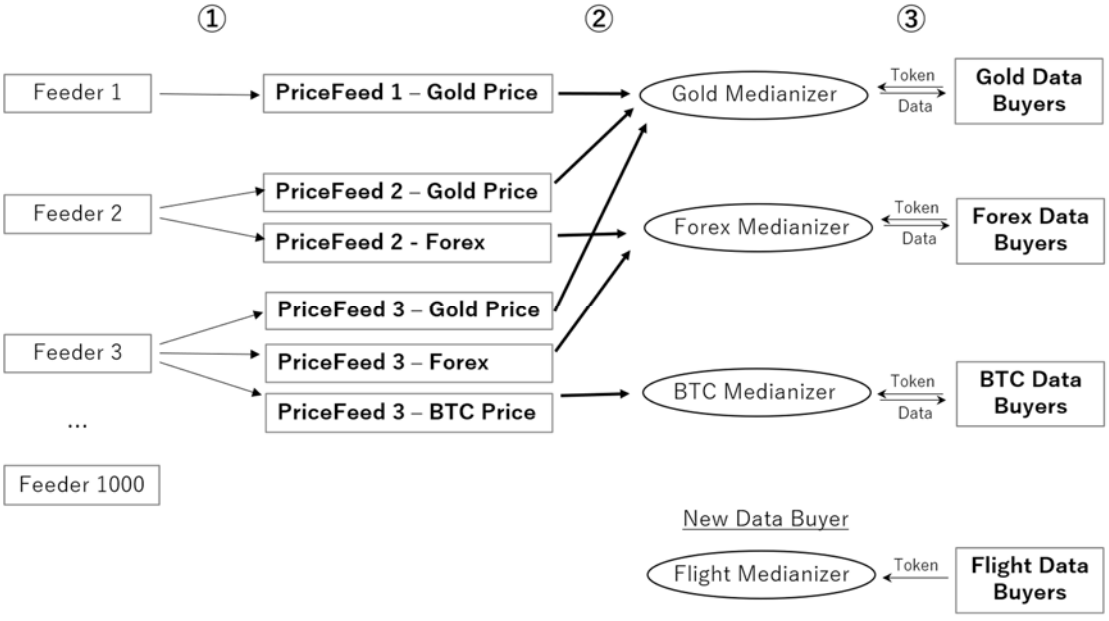
The KrawlCat Data Feed Smart Contract acts like an open marketplace for all data purchasers and providers. The Marketplace contract holds a list for data vendors that each feeder can choose to mine, as well as a portal for data buyers to create new purchase orders. The contract is deployed on Ethereum Network and can be checked by every user in the network.

Data Feeder \ Data Buyer	Gold Data Buyer	FX Data Buyer	BTC Data Buyer	Flight (New data Buyer)
Feeder 1	✓			
Feeder 2	✓	✓		
Feeder 3	✓	✓	✓	
...	...	...	...	
Feeder 1000	...	...	...	

In the example above, the marketplace smart contract has 3 data buyers in the network for Gold price, Foreign Exchange, and the Bitcoin price respectively.

- Feeder 1 choose to only provide data for the Gold price
- Feeder 2 chooses to provide data for the Gold price & Foreign Exchange
- Feeder 3 provides data on all three markets.
- A new data buyer just set up an order for Aviation Info

From the perspective of contract flow, the aforementioned example can be expressed like this:



- Feeder 1 is only working on the Gold Price. Therefore he will only contribute to the gold Medianizer. So the Feeder will receive token rewards exclusively from the Gold Data buyers.
- Feeder 2 is working on both the Gold Price as well as Foreign Exchange. So Feeder 2 will upload data to those two Medianizer contracts, their rewards will also come exclusively from data-purchasers in those markets.
- Feeder 3 will contribute to all three available Medianizer contracts.
- Aviation information buyers have paid a token to the aviation market contract. Feeders will start providing this data shortly.

KrawlCat Data Feeder can handle numerical and categorical data type, The Marketplace accepts most simple-form data orders:

Data Type	**User Scenario examples**
Numerical:	• **Price:** Bitcoin to USD price collected from exchanges • **Stock index:** NASDAQ quote collected from official website
Categorical:	• **Game result:** Raptor wins Magic in the last match • **Deliver statues:** the product is delivered and signed.
Combination:	• **Weather:** the weather is Cloudy, 10℃ to 3℃ • **Flight:** the flight has been delayed for 1hr 20min

3.4 Mining Procedure

1. When you start up the Feeder, the device will automatically perform an update-check to install the latest version of The KrawlCat's firmware.

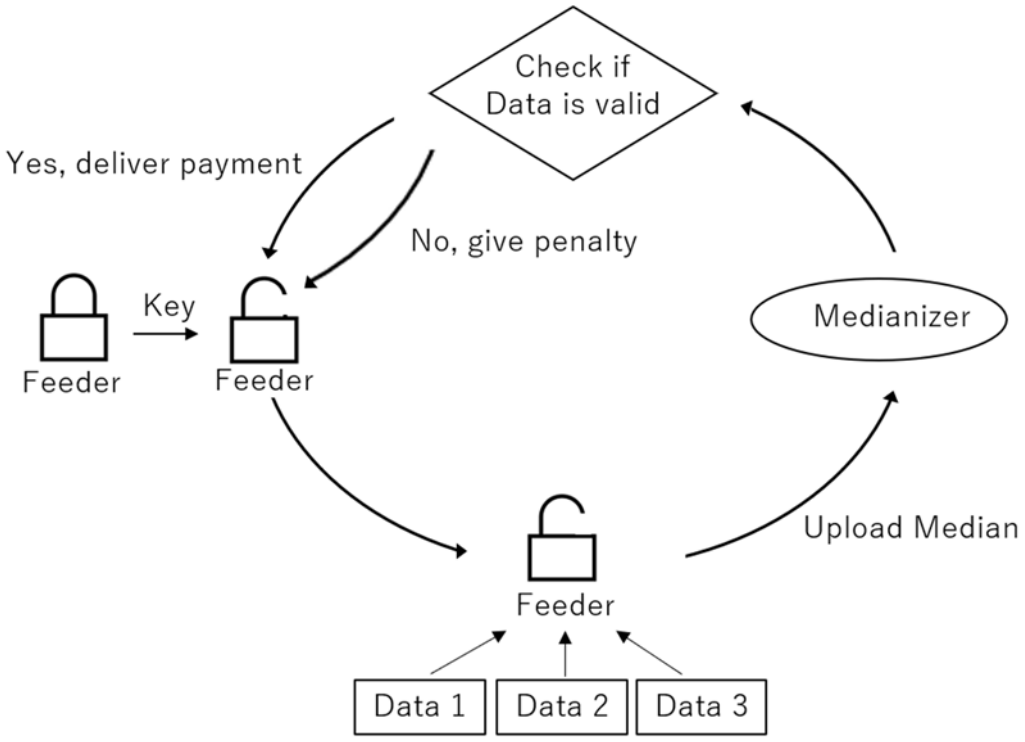
2. Users will register a private key or input an existing key to start mining. During this procedure, the Feeder will create a smart contract to store this data, every Feeder has a corresponding smart contract. Right after this process, the Feeder will call the smart contract to determine if the user has staked the necessary capital and is therefore eligible to join the network.

3. To begin mining, Feeders will automatically get data from our data vendors' APIs. Each Feeder will send all of their data to the medianizer where it will calculate the median.

   Please note: The calculation of this median value occurs on the back-end of each mining device. But the Feeder only will upload the median value to the smart contract when:
   1. There is a price discrepancy of more than 1% between the most recent price submission, and the current prices provided by the Vendor's API's.
   2. The last price update was more than 6 hours ago.

4. The Marketplace Contract will collect data from all Feeders in the network. The Smart contract will filter out values that are not “within the moderate range” and calculate the median based off the remaining data that is “within the moderate range”. The contract will then store the median number on the blockchain. Since the majority of the Feeders are providing the same data-point, this final result will be the data that is sent to the businesses that are paying for access to this information.

5. In the meantime, the Marketplace Contract will decide which Feeders provided data within the moderate range, and outside of the range. Feeders within the range will be rewarded, and Feeders outside of the range will be penalized.



4. Token Economy

KrawlCat team will issue KrawlCat Token. KrawlCat token is a utility token based on ERC-20 standard, and will be deployed on Ethereum Network at the first stage. KrawlCat Token is used for identifying the Data Feeder's eligibility, as well as creating liquidity for Data Feed Marketplace orders.

5. Global Strategy

First Stage:
Sell 100 Beta Feeders to a pre-vetted group to establish a functioning network. At this stage, feeders are functioning, yet frequent updates will be made to ensure the stability of Feeder network.

Second Stage:
Open Feeder sales channel and sell 1000 Feeders globally, and start the multi-chain support like EOS network. 1000 Feeders are sufficient at this stage to be considered decentralized, and the Marketplace contract will start engaging commercialized data buying orders.

Third Stage:
KrawlCat team will stop selling Feeders, gradually reduce its role on the Data Feeding Marketplace. The team will opensource the Feeding software to all community, start accepting every third-party Feeder to join the network, and finally let the feeders in the network reach autonomy.

5.1 Distribution Method:

There are two major means of circulation of the token.

1. Since the Feeders are used to inputting this trust-less data into the blockchain, those who pay for the data will transfer the token into the Marketplace contract. The Marketplace contract will then automatically pay token to feeders based off the data contribution.
2. 1 million token will be minted each year, and the allocation of tokens will be equally delivered to normally functioning Feeders.

For the first distribution method, the payment for Feeders comes from the buyer of data source. To activate a data buying order, all data buyers will have to deposit certain amount of tokens according to data type, feeding frequency, and subscription period. Currently, all data buyers should deposit 10,000 tokens to create a buying order, and the token will be paid to Feeders gradually within one year. After one year, the order will be either cancelled or suspended until receiving new payments.

For the second distribution method, the annual token addition goes directly to the Feeders. Feeder Devices are the fundamental supporters of the Data Marketplace, and their existence is crucial to maintain the network and its level of decentralization. The annual token addition will be continuously rewarded to Feeders who are correctly providing data. Those Feeder been temporally penalized will not be rewarded until the punishment is removed.

5.2 User Scenario

1. Decentralized derivatives
Financial derivative is a contract between two parties which derive its value/price from an underlying asset. Decentralized derivatives platform enables people to hedge risk and discovery price using cryptocurrency. For example, dYdX, Market Protocol, and Yesbit are developing such platform. KrawlCat Feeder network can provide all kinds of price feeds and fact results required in derivative contracts.

2. Stablecoin
Stablecoin plays an important role in crypto development. All blockchain applications that require stable pricing, for example decentralized lending, cannot settle the price using a coin that has 10% volatility every day. Dai is a very successful project for stablecoin, KrawlCat Feeders can provide reference price feed for ETH-USD using a decentralized approach.

3. Decentralized Randomness.
Randomness is crucial for most 1) PoS Protocol, 2) Some Applications. Most Proof of Stake need to randomly pick the next verifier. If the picking method can be predicted or biased, the verifier set would be damaged. Furthermore. A trustworthy random number is the very core for some game and apps that generate contents based on random seeds. KrawlCat Feeders can generate verifiable, unable to bias, and secure random number for the network.

4. Off-chain Trigger monitor
Decentralized lending, contract, insurance, gambling, and many other blockchain applications are triggered by off-chain events. KrawlCat Data Feeders provides a decentralized solution to monitor the off-chain data, and announce the trigger to blockchain if the condition meets the preset requirements.