TRUF.NETWORK: Decentralized Database for Prediction Markets with Proof of Source and Data Integrity

TRUF.NETWORK Team
truf@truf.network
https://truf.network

Abstract. A mature ecosystem of decentralized, event-based financial contracts requires cryptographically verifiable data integrity. Event-based financial contracts - futures contracts, prediction markets, sports betting, and more - represent quadrillions of dollars of economic activity each year. Contracts of this type, in the context of decentralized finance, rely on event result data from an external third-party - a “source-of-truth” - to settle. At scale, in such an environment, there is an existential risk of manipulation of source-of-truth data, allowing (and even encouraging) malicious actors to cheat. As more of these contracts move on chain, the risk has grown to a level that can no longer be ignored.

We propose a solution to the source-of-truth problem. The use of a permissionless distributed ledger ensures data availability and integrity. The ledger is a SQL variant, requiring the acquisition of a minimal skillset by new network participants while allowing virtually unlimited permutations of the standardized data provided by network participants. Network participants can request attestations of query results, cryptographically signed by validator nodes, providing data that has assured integrity and that can be used to settle event-based financial contracts across all existing and future platforms and protocols.

1. Introduction

The decentralized revolution that began with the release of Bitcoin has evolved, in the ensuing years, from a mere system for the peer-to-peer transfer of "electronic cash” to a thriving global ecosystem of decentralized finance (DeFi) where myriad financial markets and instruments are secured, transferred, and measured on hundreds of different distributed ledger protocols. Governments and major financial institutions that once sought to have users of such decentralized networks criminally prosecuted have now themselves become the major proponents of DeFi and are rushing to migrate the systems using their existing legacy infrastructure over to the powerful new decentralized technology born from that first revolutionary network begun by Satoshi Nakamoto. DeFi has proven to be an efficient and effective tool for tokenization and

trading of not only existing assets but even memes, ideas, and art. Distributed ledgers are powerful tools for providing cryptographic guarantees of data integrity when it comes to exchange of tokens on order books. However, the use of distributed ledgers in prediction markets - where the terms of contracts demand settlement based upon the outcome of some event that takes place outside of the context of the distributed ledger - comes with intrinsic risk. If the source-of-truth providing the results of an event is corrupted, any prediction market leveraging that source-of-truth is subject to total failure.

What is needed is a system where the source-of-truth can be independently validated, using cryptographic proof, across all possible contracts leveraging that source-of-truth, by not only all parties to the contract, but by any party interested in the integrity of a prediction market. We propose a distributed ledger network, functioning as a marketplace for source-of-truth data, where any interested party can independently validate the integrity of an event result, where individual results can be combined to form complex indexes, and where the validating nodes of the network can provide cryptographically certified results that can be used in any contract and where the results, and the queries that generate them, are standardized.

1.1 Philosophy

In general, a contract between two parties, where financial settlement of the contract is based on the outcome of some event that occurs subsequent to the execution of the contract, consists of some sort of escrow arrangement. In the context of DeFi, the most basic event-based contract requires that two parties send funds to an address or account representing the executable contract terms. When the event has happened, a third party, generally referred to as an “oracle,” generates a cryptographic signature representing the event outcome. At the time the contract was created, the oracle’s public key was included in the logic of the contract. Thus, when the signature is provided to the contract, funds can be disbursed to the appropriate party (or parties). Such a process can be considered “cryptographically secure.”

The system we propose meets several criteria that enable its ubiquitous use in DeFi, at universal scale, and providing the highest level of sustainable data integrity. First, the system consists of a permissionless distributed ledger network leveraging a replicated relational database. Data is provided and consumed by network participants in a decentralized manner and the network is secured by economic incentives. Second, events are represented as standardized database queries in a popular structured query language. The results of all queries are deterministic and can be validated by all network participants without restriction. Third, digital signatures, and accompanying metadata, are provided in a format that facilitates their use in any application, platform, or protocol. Fourth, the immutable, replicated record of data provision and cryptographic combined with the network’s economic framework creates a natural market for data where data providers compete on data accuracy and liveness.

2. Technology

2.1 Overview

The system uses a leader-based proof-of-stake (or proof-of-authority) blockchain consensus model. In order for the network to generate and propagate blocks, at least three nodes must be assigned as validators. Nodes connect to the network in a permissionless manner. Nodes that are authorized to do so may create stored procedures, database queries that may be accessed and run by any node. Any node may broadcast a transaction executing a stored procedure and, if the stored procedure changes the state of the relational database, that change will be reflected across all nodes. Any node may broadcast a transaction requesting that validator nodes publish and cryptographically sign the result of a stored procedure. This result is stored in the replicated database and is accessible to all nodes. Though not complex in scope, this process allows nodes to generate a fundamentally infinite number of data permutations by changing the arguments supplied to the available stored procedures and by authorized nodes increasing the number and variety of stored procedures available to all nodes.

2.2 Data Standardization

We define a stream as a sequence of data points representing events within a specific, articulated context. For example, the settled price (data point) on the first trade of every hour within the BTC/USD marketplace on the Binance exchange (specific context) would qualify as a stream. When a datapoint is stored in the ledger, the stream is defined as primitive. When multiple different primitive stream datapoints are retrieved from the database and transformed into a single datapoint using some algorithm, the stream of the datapoints resulting from such transformation is defined as composed.

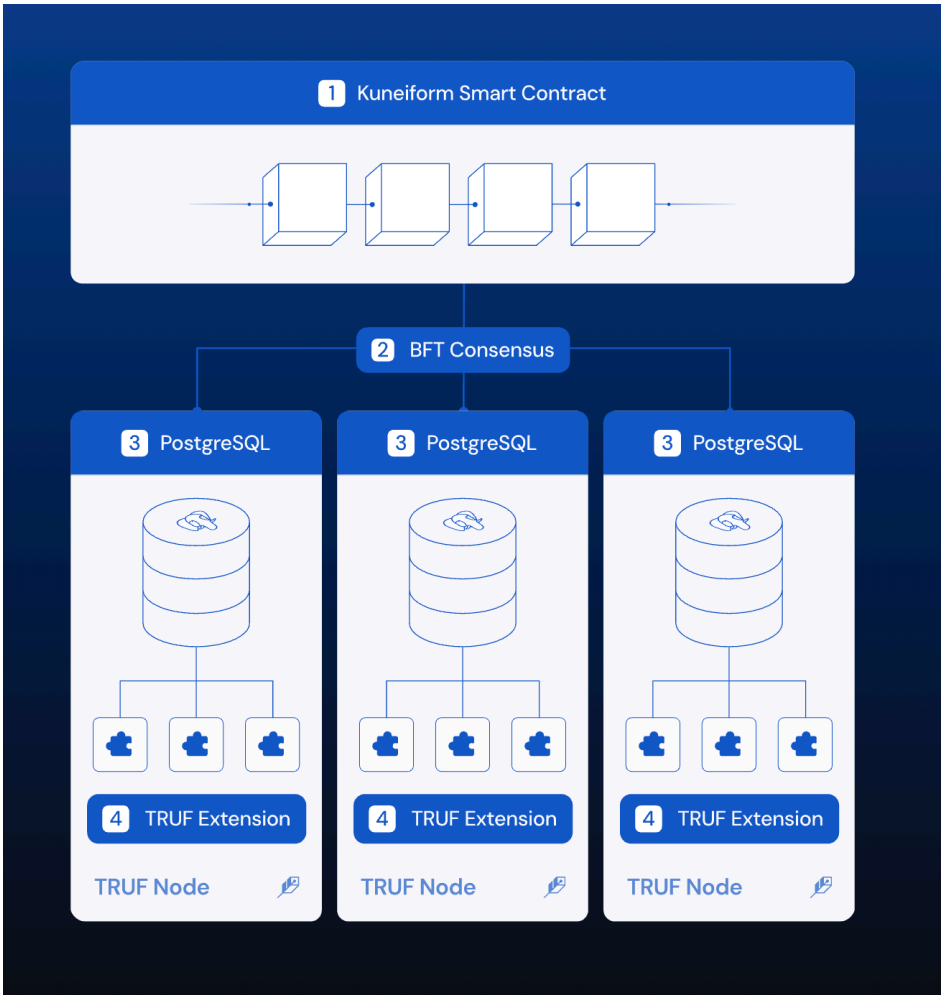
We use a SQL-based distributed ledger to store primitive data points. Because SQL is ubiquitous across many software applications and because most, if not all software professionals have at least come competence with SQL, the decision to use this form of database means that the creation of composed streams, using SQL queries, is simplified and accessible to the maximum number of network participants.

Both primitive and composed streams can be used as a source-of-truth in event-based financial contracts, but network participants have the capability to acquire a certified event result, cryptographically signed by network validator nodes, in order to ensure data integrity and mitigate the risk of source-of-truth corruption. This certified event result is standardized such that any financial contract (or any party to that contract), DeFi or not, on any network or platform, can easily use the certifying signature to securely settle the contract.

2.3 Distributed Ledger

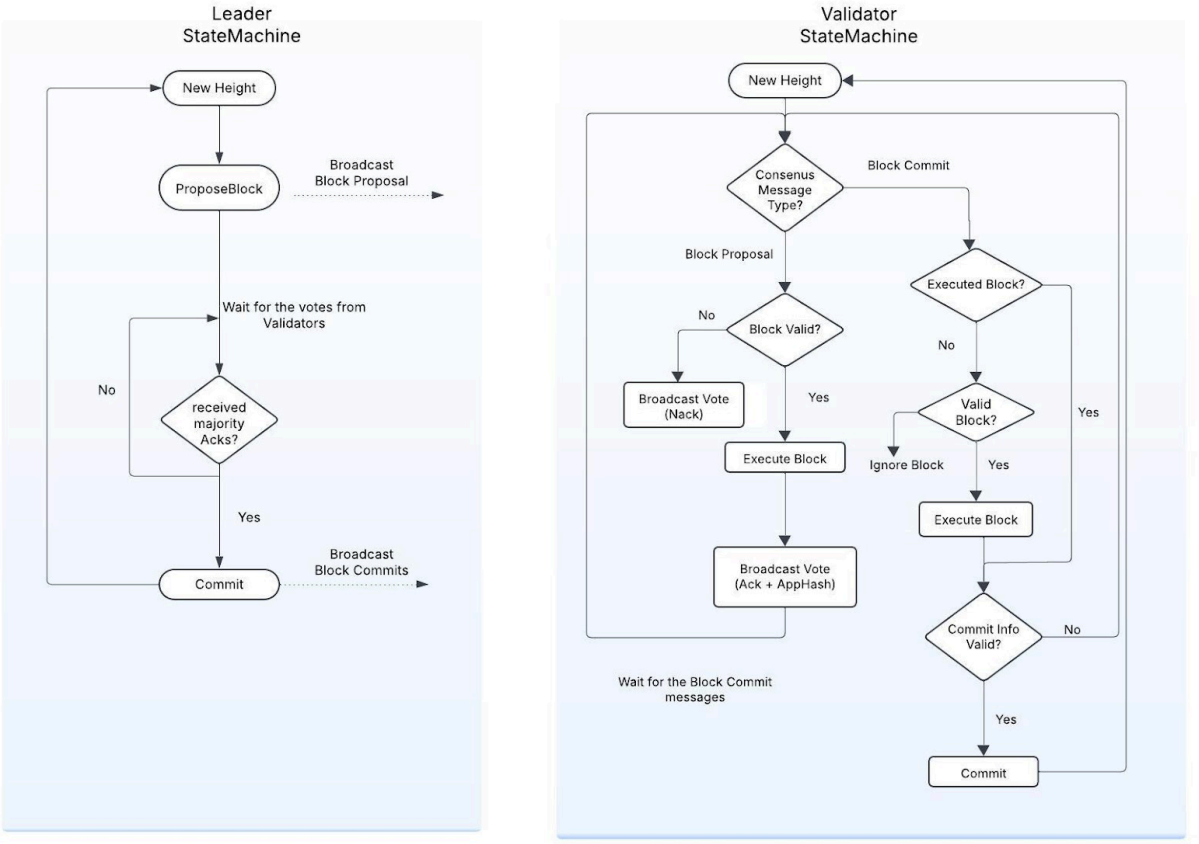
For our blockchain network, we utilize a modified version of the kwil protocol[1], a replicated Postgres SQL database. Database actions such as CREATE, ALTER, INSERT, UPDATE, and DELETE are performed via transactions broadcast to the blockchain network and included, by validator nodes, in blocks propagated throughout the network. In contrast, SELECT actions are performed by individual nodes on their own local databases and, thus, do not have an associated transaction that is broadcast to the network.

The kwil protocol utilizes a syntax language called Kuneiform, that allows for the analog of SQL stored procedures to be referenced and executed by all nodes. The similarities to the programs introduced by Ethereum has caused these Kuneiform stored procedures to be referred to as SQL Smart Contracts. The purpose of these smart contracts on kwil is to enable database table creators to place restrictions on the types and structure of queries available to other network participants. The SQL Smart Contract concept can be seen as a highly configurable extension to the functionality traditionally associated with SQL's built-in database permissions.



[1] https://github.com/trufnetwork/kwil-db

Non-validating (“sentry”) nodes may join and leave the network at will. Validating nodes create blocks based on the Roadrunner consensus algorithm. The Roadrunner consensus algorithm is a database-specific consensus mechanism developed by Kwil. It is designed to be efficient and fast, with a focus on low latency and high throughput for database operations. The algorithm draws inspiration from established consensus protocols like Two-Phase Commit, RAFT, and PBFT, but is tailored for the unique requirements of database systems. The Roadrunner algorithm operates on the following core principles: the database state is cryptographically verifiable and replicated on all nodes; once a block is committed, it is final and cannot be reversed; nodes with inconsistent states are excluded from the consensus process; the network remains operational as long as a majority of validators are online and in agreement; validators can hold the leader accountable for invalid blocks and initiate leader replacement if necessary.

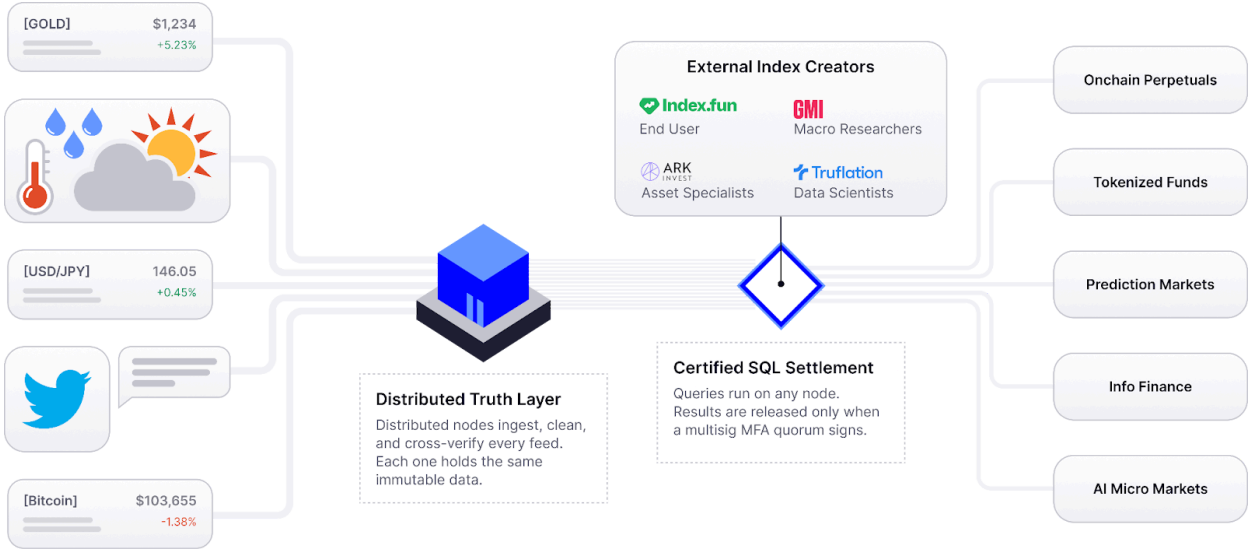


The Roadrunner consensus algorithm involves a two-phase process. In the Propose phase, the leader proposes a new block, and validators vote on it. Validators verify the block and its state changes before voting. In the Commit phase, if a majority of validators agree on the block, the block is committed, and the state changes are applied to the database.

2.4 Attestation

In general, an escrow contract requires three parties: two counterparties and an escrow agent. It is the job of the escrow agent to hold the funds deposited by the counterparties into the escrow account and to, ultimately, disburse the funds, to one party or the other, based upon the fulfillment of the conditions of the contract. An escrow agent must be a neutral third party, adjudicating whether contract terms have been met without bias toward either party. In the context of DeFi, the escrow agent role is fulfilled by a smart contract or covenant, a procedure performed by a node, as the result of a transaction, that results in control of escrowed funds transferring to the control of one part or the other. In many cases, the escrow can be settled solely using data that is available to the node because it is data stored on the blockchain network of which the node is a part. In the case where data external to the network is required, a party referred to as an “oracle” provides a cryptographic signature that can be used in the settlement transaction. As in the case of a traditional escrow agent, it is critical that an oracle is impartial. However, in a decentralized, anonymous (or pseudonymous) context, it is virtually impossible for any party to an escrow smart contract to be sure that their counterparty is not colluding with the oracle. This is particularly true if the source-of-truth being used by the oracle as the data point is not publicly available.

The system we propose solves the problem of public availability of the source-of-truth. Anyone can run (or query) a node on our network and validate that the results of a query correspond with oracle data. This gives any node operator that capability to act as an oracle for an escrow transaction. This sort of “self-certification” may be sufficient for occasional escrow transactions between parties that trust one another. However, at scale and in an environment that is adversarial in nature, parties require a greater deal of assured data integrity.



The solution we propose to the data integrity problem is the introduction of a transaction type representing a request, by a network participant to one or more validator nodes, for attestation of the result of a particular query derived from a specific stored procedure and set of arguments. The response, from the validator node, stored as a database row, is a standardized data structure consisting of the stored procedure identifier, the arguments used, the query result, and a cryptographic signature of the other elements by the validator node(s). The signature is encrypted using the public key of the user who signed the requesting transaction. This encryption ensures that all parties can independently verify the integrity of the attestation but only the requestor, who ostensibly paid the transaction fee, has access to the signature.

Attestation data can be used, standalone, as an oracle but it can also be used as a source-of-truth along with existing or future oracle protocols.

3. Economics

3.1 Attestation

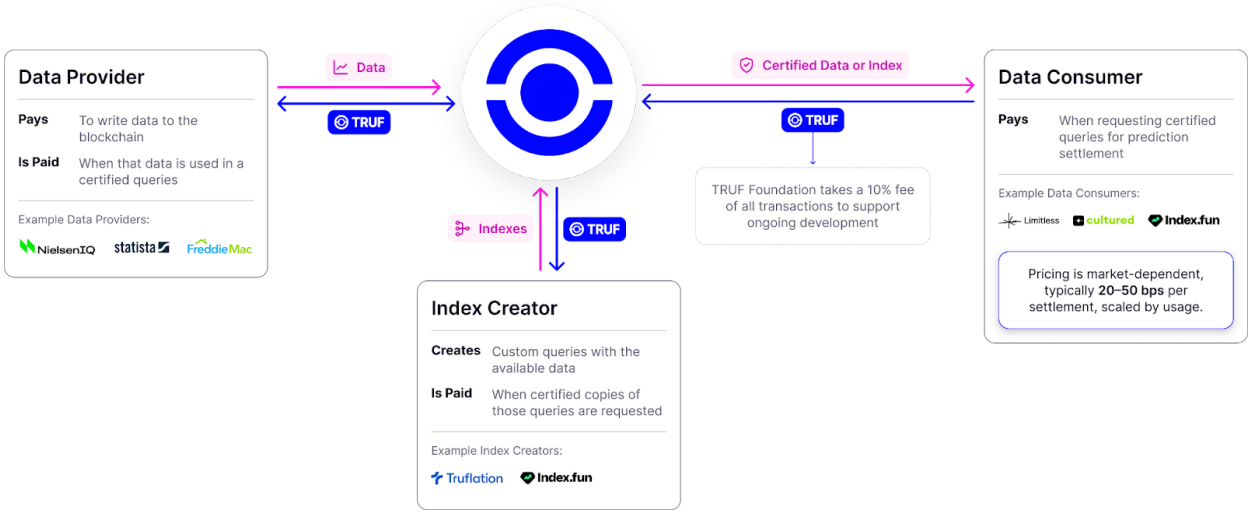
Unlike distributed ledger networks such as Bitcoin or Ethereum, the system we propose has no native token intrinsic to the operation of the network. However, any node with sufficient permissions can deploy a stored procedure that has, as part of the logic, a mechanism for accounting for changing balances between addresses or accounts. By leveraging a smart contract on another distributed ledger network, such as an EVM smart contract on Ethereum, a “bridge” mechanism can be implemented. An example bridge mechanism would be utilized by, first, an Ethereum user depositing tokens into the EVM smart contract address. Next, that deposit would be cryptographically validated by a stored procedure on our network, giving control of the deposited token balance to the same public key that made the deposit. The user could then transfer some portion of the token balance to another user via a transaction on our network. This second user could then withdraw tokens from the EVM smart contract through the use of a stored procedure that removed token value from the system while producing cryptographic proof necessary for the withdrawal from the EVM smart contract.

While the number and type of tokens that could be bridged onto our network is theoretically unlimited, the most efficient means of using the token bridge model described is to limit tokens to only those fundamental to implementing economic incentives that sustain network viability.

3.2 Block Rewards

Distributed ledger technology is sufficiently mature such that proven sustainable models have been sufficiently proven. The simplest economic model is also the most robust. That simple model consists of users paying a fee, in a particular digital asset

(token), as part of any transaction broadcast to the network. If a transaction is accepted by a block producer - miner or validator - and is included in a block, the block producer collects the fee. Additionally, some networks have a built-in token emission schedule per-block that block producers are also able to collect. Any newly created tokens collected by the block producer along with any transaction fees for a given block is referred to as the “block reward.” Utilizing a token bridge and properly implemented stored procedures that require token transfer actions, validators on our network can implement the equivalent of a block reward system, at least in terms of a transaction fee requirement.



There are three general types of transactions that can be broadcast: transactions that deploy stored procedures; transactions that insert or update rows in the SQL database; transactions that are attestation requests resulting in cryptographically certified query results generated by validator nodes. By using a market-based approach to the pricing of these transactions, validator node activity can be profitable and, thus sustainable. Pricing of transactions can be done according to computational complexity, similar to the Ethereum “gas” model, but attestation transaction can also be priced according to the economic value of those transactions by using algorithms that analyze the contracts, in the broader distributed ledger landscape, where the certified query results are likely to be used. This latter pricing model is akin to actuarial calculations in insurance where the premium paid is relative to the value of the assets at risk. In other words, attestations can be priced according to the total escrowed value of known contracts that require a certified query result in order for the contract to be settled.

3.3 Accuracy Via Competition

The economic framework presented above can be used to incentivize data providers to broadcast only accurate data. Broadcasting event data for storage in the database as a primitive stream carries a cost, in the form of transaction fees. By sharing a portion of the transaction fee, charged for certified attestation, with the data providers whose streams are used in the result, validators can allow data providers to profit from the

primitive streams they maintain. Because there is an immutable record of all certified attestations, available to all nodes, relative popularity of primitive streams and, by extension, data providers can be measured. Additionally, the replicated database can be used, alongside secondary sources outside of the network, to verify the accuracy of any given primitive stream. Given the permissionless nature of the network, data providers can compete on accuracy, with more provably accurate data providers able to compete for a greater share of transaction fees, displacing less accurate providers of data about the same events. This unique model ensures data integrity is provided by the technical aspects of the system while data accuracy is ensured by the system's economic aspects.