# A Decentralized Oracle Architecture for a Blockchain-Based IoT Global Market

Lorenzo Gigli, Ivan Zyrianoff, Federico Montori, Cristiano Aguzzi, Luca Roffia, and Marco Di Felice

The authors propose an architecture that enables a decentralized IoT global market in which clients pay for data and device owners are rewarded for providing it. Their solution employs a distributed oracle layer on top of smart contracts powered by a distributed global network of IoT devices.

## Abstract

The Internet of Things (IoT) envisions a global market in which it would be possible to easily get data from IoT devices across the globe. However, the potential of this idea still needs to be unlocked. Centralized architectures fall short due to their lack of transparency and tendency to create silos. On the other hand, blockchain technology enables the creation of distributed and trustworthy systems, but its integration with the IoT is still a matter of research. IoT-based scenarios often employ numerous devices for the same sensing task, which may be heterogeneous and unreliable by purpose. In our vision, IoT applications should rely on data and its quality rather than on single providers. For this purpose, we propose an architecture that enables a decentralized IoT global market in which clients pay for data and device owners are rewarded for providing it. Our solution employs a distributed oracle layer on top of smart contracts powered by a distributed global network of IoT devices. The system supports IoT data source decoupling since the end-user can perform semantic queries bounded to specific locations and data types without specifying the target devices. In addition, it features automatic discovery, interoperability mechanisms, and reputation algorithms for the selection of trustworthy data sources. Our results show that the proposed system is robust and consistently provides quality data, even with multiple malicious data sources.

## Why Do We Need an IoT Global Market?

The global network of computers completely reshaped society in the last decades, allowing virtually anyone to be connected to massive sources of information instantly and with minimal costs. On the other hand, the Internet of Things (IoT) has been around for over two decades now [1] and, still, the initial vision of a global system of interconnected devices is far from reality. A decentralized IoT global market would allow the end users to easily and seamlessly gather data from IoT devices connected to a global network. Additionally, anyone could effortlessly connect its device and expose its capabilities throughout the network — monetizing its usage [2]. For instance, smart insurance companies can utilize reliable sensory data as automatic triggers to release contract compensation, inherently benefiting from blockchain security and fraud-proof features — further description of this use case is provided later in the manuscript.

Many approaches have been proposed to fulfill such a vision. However, they all share similar pitfalls. On the one hand, centralized approaches inherently bind users to trust a single entity and to stick to specific standards and technologies [3]. This unavoidably creates silos that are hardly interoperable with one another and do not guarantee the necessary transparency to the final users. On the other hand, decentralized solutions fail to provide the required trust for users to consume data. Trust plays an essential role in the IoT global market since clients need to ensure the quality of the queried data.

Blockchain technology has the potential to become a pivotal enabler for a global IoT market [4], as it creates trustworthiness in totally decentralized systems by sharing among all the nodes of a network a single and immutable history of transactions. However, many challenges still impede unleashing the full potential of blockchain for IoT-based applications. In this article, we tackle four of them.

IoT devices have limited processing and storing capabilities and are often battery-powered, imposing energy efficiency constraints. Thus, it is unfeasible for IoT devices to be blockchain network nodes since they cannot spend energy and computation to verify other transactions and cannot store the transaction history. Furthermore, blockchains themselves are not able to actively access external IoT data.

IoT devices are unreliable per design; they are made to be numerous, inexpensive, and interchangeable. Moreover, they could be more susceptible to tampering than traditional devices since computational-intensive security mechanisms cannot be supported. Hence, a substantial overhead — imposed by the blockchain — is in place to query unreliable data from cheap sensors.

A common and well-known problem of IoT is the lack of interoperability. IoT devices comply with several different communication protocols, data structures, and interfaces, which contribute to a fragmented landscape that hinders the adoption of IoT global markets.

Blockchain and related consensus mechanisms rely upon deterministic outputs. IoT sensor measurements are inherently nondeterministic, meaning that IoT data sources may return different results even if operating in the same conditions. This calls for an accurate method for selecting multiple data sources and safely aggregating their results to ensure trustworthiness.

ISSN: 0163-6804
Digital Object Identifier: 10.1109/MCOM.007.2200906

*The authors are with the University of Bologna, Italy.*

To address such challenges, we propose DESMO, a novel architecture — founded by the ONTOCHAIN European project — to enable an IoT Global Market based on distributed data oracles paired with decentralized IoT data sources. Oracles are special applications that connect blockchains to the off-chain world [5]. In order to increase their trustworthiness and maintain decentralization, we adopted a layer of distributed oracles and enabled the client applications to retrieve data from multiple data sources that share the same features in a specified geolocation. We need to trust not only the oracles but — and mainly — the data coming from the IoT devices. For this reason, our architecture includes reputation algorithms for the ranking and automatic selection of trustworthy data sources. Finally, to address IoT inherent heterogeneity, we utilize a well-known standard open interface — the W3C Web of Things (WoT) [6]. The WoT standard allows us to straightforwardly describe the capabilities of the devices by extending already existing Web technologies and enabling integration across platforms and applications domains. We do not elaborate on economic aspects related to payments, since they are outside the scope of the article, although they will be further investigated in the future. In the following sections, we provide further details of our proposal. We outline its unique features and technical characteristics compared to existing blockchain solutions. Next, we describe its architectural design and showcase two applications enabled by the DESMO IoT global market. To exemplify its operation, we conducted a case study that demonstrated the system's robustness to malicious nodes while maintaining data quality. Finally, the article concludes with a discussion about challenges and future research directions, such as scalability — a fundamental feature for a global system.

## Oracle Architectures for IoT Systems

Before diving into the architectural details of our solution, we introduce the reader to the key concepts and issues that serve as the foundation for fully understanding our work. Blockchains are innovative technologies to store and share data between a network of nodes without relying on a single centralized authority. They are instances of Distributed Ledger Technologies (DLTs) that use cryptography to create a secure, immutable, and transparent record of transactions. These technologies have undergone substantial changes recently, evolving dramatically from the first Bitcoin-related implementations. Among the most significant advances, we cite the emergence of blockchain-based platforms such as Ethereum, which introduced the concept of "smart contracts" as executable programs that enforce an agreement between two or more parties in a secure and verifiable way. Thanks to their pre-defined functions, they can store information, process inputs, and write outputs. Their capabilities have opened the doors to exploring new synergies not only related to decentralized finance but to other sectors such as supply chain management, insurance, voting systems, health care, and IoT.

Smart contracts only have access to data stored on the chain. This limitation is because their execution must be deterministic to be fully verifiable by other nodes in the network. Therefore, injecting external data into the blockchain requires an off-chain component, the oracle. An oracle is a software entity capable of retrieving external data and making it available on-chain for smart contracts [3]. Hence, the oracle is not the data source per se, but the layer that queries, verifies, and authenticates external data sources and then forwards such information. The use of oracles brings back the centralization that blockchain was supposed to remove from the equation, reintroducing issues related to architectures with a single point of failure (e.g., bringing corrupt, malicious, and incorrect data to the chain). This dilemma is known as "The Oracle Problem." It deals with the issue of finding a balance between efficiency and decentralization when data is retrieved from the outside through these systems [7]. Numerous approaches have been proposed to solve this problem, and they can be grouped into two macro-categories: centralized and decentralized solutions.

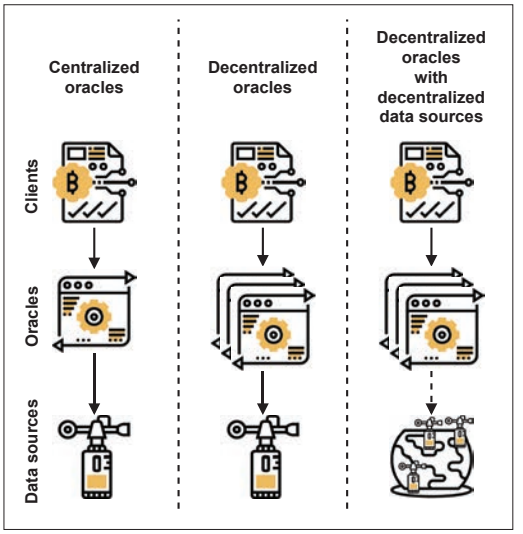

FIGURE 1. Different oracle architectures and their relationship with data sources.

Examining Fig. 1, we can notice that a centralized oracle is based on a single-server architecture and relies on a single data source. Typically, these solutions employ Trusted Execution Environments to secure the critical processes of oracles, in combination with technologies for generating proofs of data authenticity. An example of an oracle that leverages this type of architecture is TownCrier [8]. On the other hand, a distributed oracle is implemented through multiple nodes providing data to the blockchain. Each node usually relies on one or more specific data sources to fulfill the requests. ChainLink [9], for instance, is a decentralized oracle network based on reputation mechanisms and can be considered a general-purpose system. Although some of these oracles can be used to retrieve IoT data, there are specialized solutions that are designed for this task. DiOr-SGX [10] uses multiple oracle servers to minimize the risk of a single point of failure while ensuring data integrity. STB [4] is a distributed and hierarchical blockchain architecture with a peer-to-peer oracle network. It has a lightweight consensus algorithm for IoT-constrained devices and specialized components for scaling and verifying the reliability of external information before storing them on-chain. Finally, OIB [5] is a system to facilitate the deployment of smart contracts-based Indus-

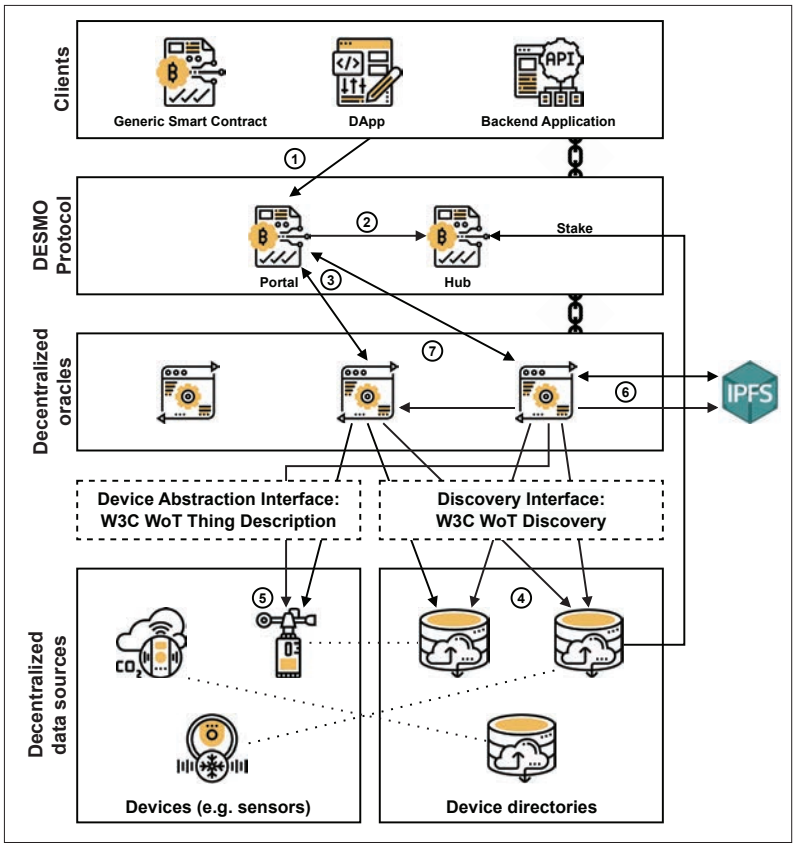


FIGURE 2. The DESMO layered architecture with the steps required in the query resolution process..

trial IoT applications. The core of the system lies in a distributed oracle network that extends the computing capabilities of the contracts.

Looking at the solutions presented, it is possible to realize how the risk of centralization is always around, especially when it comes to selecting and managing data sources [3]. Furthermore, in the specific case of oracle architectures for the IoT, the heterogeneity of devices is not considered, even if a clear methodology for integration and communication would be needed, since IoT devices do not usually have compatible interfaces with current Web technologies, as they employ a different stack of protocols. This article advances state-of-the-art by proposing a novel, fully decentralized oracle system specialized for retrieving IoT data. Differently from the cited studies, our solution detaches the oracles layer entirely from the data sources, pushing to the limit the concept of decentralization and trust. Our objectives are threefold:

- Maintain the highest possible level of decentralization of the system enabling multiple oracle nodes to retrieve data from a set of different data sources each time.
- Treat the data sources as first-class citizens of the system, keeping track of their reputation and rewarding them for the provided data.
- Unify device discovery and communication through a well-defined interface that can serve as a homogeneous abstraction layer for the various actors in our system.

## How to Design a Blockchain-Based System for an IoT Global Market?

In this section, we analyze the DESMO architecture and how it supports the concept of global IoT market. As we will see, the DESMO architecture spans from on-chain components (smart contracts) to off-chain components (oracles, indexers, and data sources). All these concur in achieving a fully decentralized system organized in the following layers:

- The Clients are the buyers of IoT data and can be on-chain or off-chain components.
- The DESMO Protocol layer is composed of smart contracts that register requests, store responses and payments and detain the reputation ranking of data sources.
- The Decentralized oracles layer collects the requests from smart contracts and queries the designated sources.
- The Decentralized IoT data sources layer contains the devices that provide the data and the directories that index them. Users can register new directories by staking tokens. In this way, if a directory behaves fraudulently, the protocol can punish it by draining from these funds.

As illustrated by Fig. 2, the top layer of the platform contains the clients, which can be different: smart contracts — for example, protocols — that need to access IoT data to perform automatic actions, Dapps implementing IoT use cases, server-side applications in need of retrieving IoT data in a trusted and distributed way. Immediately below, we find the DESMO distributed protocol, a set of smart contracts that cooperate to manage the various aspects of the system. The Portal contract is the entry point of the client, receiving requests for data and initiating the data retrieval process. The Hub is the repository for registered data sources and their reputation score, which quantifies the trustability of each source. Finally, the Token contract (omitted in Fig. 2) holds the currency that powers the system's economy. The amount paid by each client is distributed between the oracle nodes and the data sources participating in the process. The oracles layer contains the worker nodes capable of executing the DESMO oracle application, which queries a defined set of data sources and computes a result to the client request through a consensus algorithm between oracles, as explained later. The bottom layer includes the data sources, and its structure is divided into two distinct blocks: directories and end devices. Directories are responsible for indexing the metadata of physical devices and making them discoverable via a well-defined interface that supports both semantic and geo-spatial queries. On the other hand, to be compatible with the system, devices must expose a semantic descriptor that allows both directories and oracles to interact with them. We decided to use the emerging W3C WoT standard in our implementation as it extends existing web technologies and provides a homogeneous interface to access IoT devices that abstract from their particular interfaces and heterogeneous network protocols. Specifically, directories, namely Thing Description Directories (TDDs), will have to implement the W3C WoT Discovery specification[6] with the addition of support for spatial queries. Devices will have to be described by a W3C WoT Thing Description (TD)[6] and be registered on one of the TDDs to be found and used by the system. To store off-chain, accessible, nondeterministic data we make use of IPFS (InterPlanetary File System) [11]. IPFS is a decentralized, peer-to-peer file system that provides storage and retrieval of data on the Internet.

It replaces traditional methods with a content-based addressing system for files and their versions. IPFS allows for saving arbitrarily big files and, by writing only their hash on the actual blockchain, we ensure always to store a constant amount of data on-chain.

The complete flow of a single data request is then depicted in Fig. 2.

**Step 1:** A client submits a new request to the system by calling a function of the Portal smart contract. The payload of a request includes the semantic query for identifying the desired data type and the geospatial filter — for example, the temperature in the metropolitan area of New York City.

**Step 2:** The Portal contract asks the Hub for the TDDs to associate with the request. TDDs, as said, can become part of the system (i.e., included in the list detained by the Hub) by staking a certain amount of tokens. Each of them starts with a neutral reputation value — we set it to 0 — and gets selected by the Hub to reply to a data request through a round-robin selection process. The process uses reputations as weights, so high-reputation TDDs are more likely to be selected.

**Steps 3 and 4:** A subset of oracle instances takes on the request and queries the selected TDDs. It is essential to highlight that each oracle makes the same query on each of the elected TDDs and collects the descriptors of all devices that semantically match the request.

**Step 5:** The oracles retrieve sensor data from the end devices and need to reach a consensus on which data point best represents reality and how good and reliable the sensor data is. The process that implements the above actions (the "consensus process") is depicted over five phases in Fig. 3, and supports the explanation of steps 5, 6, and 7 of the query resolution process. We can abstract from the concept of TDD and assume that each oracle queries the same data sources — that is, the sensors — and obtains from each of them a single data point corresponding to the sensor reading. In Fig. 3, data points generated by different sources — phase I of the consensus process — are depicted as squares of different colors. In this case, each oracle, within a single request round, ends up with as many data points as the number of queried data sources, represented in phase II of the consensus process. Note that two oracles may obtain two different data points from the same source. This inherently belongs to the nature of IoT and can be dictated by several factors: for instance, oracle queries could have taken place at two different moments in time, thus triggering two different sensor readings, causing nondeterminism. Nonetheless, within a single request round, we expect a data source to reply consistently to multiple oracles, meaning that data points shall differ negligibly. The subsequent steps will address the nondeterminism problem.

**Step 6:** Oracles store the reply they obtained from data sources onto IPFS. In phase III of the consensus process, the collection of replies on IPFS is represented as a matrix, where each row is associated with a single oracle and each column with a single data source. Upon saving the reply on IPFS, each oracle also hashes its content and saves the hash onto the blockchain. This ensures the amount of space that a single client request occupies on-chain to be solely dependent on the number of selected oracles (which is constant),

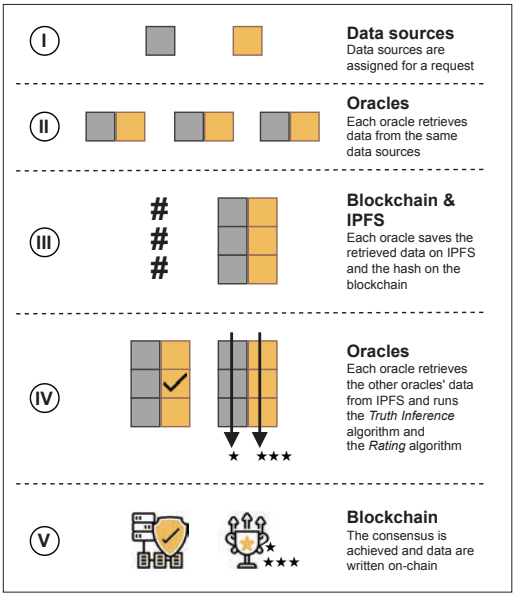


FIGURE 3. DESMO data gathering and consensus process.

not on the number of queried sources (which can be arbitrarily high, depending on how much the client pays). In phase IV of the consensus process, oracles can use the hashes stored on the Portal to retrieve the full matrix from IPFS. At this point, each of them executes two actions: The *Truth Inference algorithm*, which outputs a single data point in the matrix — the "inferred truth" — that is believed to be the closest to the ground truth, and The *Rating algorithm*, which outputs a score for each data source on top of their output with respect to the inferred truth. The score will be integrated with the overall reputation of the data source. The Truth Inference algorithm is a function $\tau$ that takes in input the matrix of results $M$ and returns a single sensor reading $\tau \in M$. The Rating algorithm is a function $\rho$ that takes in input $M$ and returns an array of $n$ scores $S$, where $n$ is the number of queried data sources. Each score must be constrained to a definite interval, in our case we defined scores to be between −1 and 1.

**Step 7:** Finally, in phase V of the consensus process, oracles need to reach a consensus by all executing $\tau(M)$ and $\rho(M)$ and perform majority voting, then the two results are written on the Portal contract. Note that both functions are deterministic, which means that, if all oracles execute them onto the same matrix $M$, they should end up with the same results. The Portal then stores the result of $\tau(M)$, which is the reply to the client request, and uses the result of $\rho(M)$ to update the reputation of the sources in the Hub contract. This is done, for each source, by linearly combining its score with its previous reputation, which is constrained between −1 and 1 as well, in order to obtain again a reputation value between −1 and 1. Upon performing this operation, the Hub contract may choose to blacklist a source, if its reputation falls below a certain threshold. This operation, as we will see in the section about the Case Study, is necessary to mitigate the chances for attacks as well as untrusted or defective data sources.

The interaction of DESMO with the blockchain is a crucial aspect of its architecture, specifically regarding the transactions generated from data queries. When a user makes an initial query, both

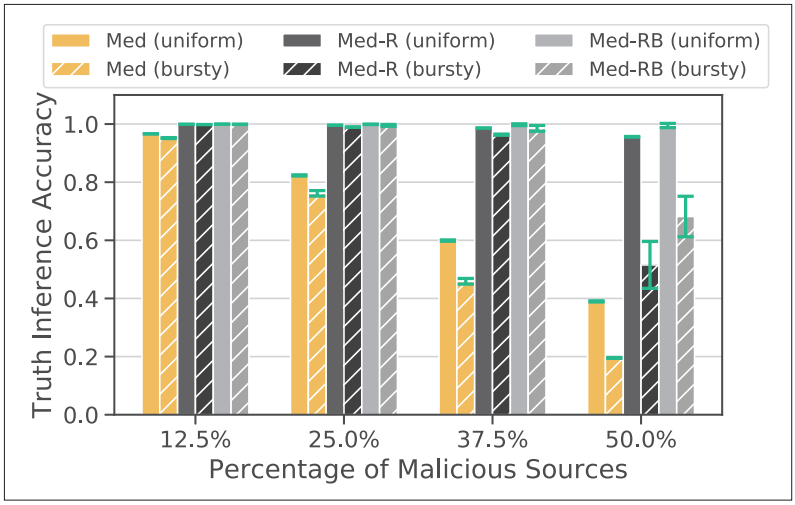


FIGURE 4. Truth Inference Accuracy: the percentage of requests by a client that get satisfied within a tolerance threshold.

the query and the final result are recorded on the blockchain, resulting in two separate transactions. Further, each oracle involved in a query stores the hash of the queried data on the blockchain to make it available for other oracles to calculate the Truth Inference algorithm — as outlined in Step 6. The output generated by each oracle is also recorded in the blockchain to enable the process described in Step 7. Current scalable blockchains have a high throughput of thousands of transactions per second (TPS) and are pushing to keep increasing those numbers — for example, Solana can process more than 8,000 TPS [12]. Considering those numbers and an average of five oracles involved in each query, theoretically DESMO could support millions of daily queries. A first implementation of the different components of the architecture can be found on GitHub[13].

## Applications of a Blockchain-Based IoT Global Market

In this section, we explore new applications that are enabled by DESMO. We selected one insurance scenario that leverages sensor data and one urban scenario related to noise pollution.

### Smart Insurances

Currently, insurance claims are a long and costly process. Insurances companies desire to combat fraud, at the same time, to reduce time and costs by automating the claim-related administrations and execution process. On the other hand, customers want rapid compensation and clear insurance terms. A trustworthy automatic process for insurance will be beneficial for both parties.

An example application enabled by DESMO is insurance in the intelligent farming domain. Suppose that a given company commercializes insurance for farmers to protect their crops against extreme environmental hazards — for example, storms. The customer and the company agreed on specific terms to trigger the farmers' payment. Those conditions and their associated triggers are established on a smart contract that acts as a client of DESMO — interacting with the Portal contract.

The insurance company smart contract defines features — for example, wind speed — to be queried in a specific geolocation without specifying the set of devices that provide such data. The data sources providers could be both sensors deployed by a trusted partner of the insurance company, as well as already deployed devices in the farm and nearby locations. DESMO Truth Inference and Rating algorithm will assure data quality and prioritize the selections of trustworthy data sources to avoid malicious users tampering with the system — for example, falsifying a storm's conditions. In case the conditions are met, the smart contract automatically releases the tokens to the farmers to keep their businesses running.

### Urban Noise Pollution Monitoring

Noise pollution is a common hazard in urban environments, and its sources range from social activity to construction. Public authorities commonly incentivize noise mitigation by imposing fines on noise emitters above a certain threshold. However, it is challenging to enforce noise regulations in large urban environments. We envision such systems to leverage the usage of DESMO. The appointed public authority would hire several small and medium enterprises to cover a city with noise monitoring sensors. Different set companies could cover different neighborhoods, with some overlap. Another third-party actor — employed by the public authorities — would deploy a smart contract that queries the DESMO Portal contract to get noise pollution data in different parts of the city. Once a code violation is detected, the smart contract automatically warns the competent authorities to take appropriate action. The advantage of using DESMO in such a setup is threefold:

- DESMO geo-spatial filter allows clients to utilize different granularity of boundaries in successive queries to narrow the precise location of the noise pollution source.
- Given the truth inference and rating algorithm, malfunctioning data sources would not hinder the overall system data quality, and their data would be requested less often. Consequently, reliable data sources would be queried frequently and — as each request in DESMO is paid — be more profitable, incentivizing the continuous maintenance of the system.
- Different companies would deploy sensors with diverse technologies regarding communication protocols and data structure. DESMO, by design, handles IoT heterogeneity via W3C WoT solutions.

## System Evaluation on a Case Study

We developed a case study to assess the proposed system's resilience and effectiveness in a theoretical scenario in presence of malicious data sources. As we motivate our work on trustworthiness, we need to punish or ban possible malicious data sources while maintaining satisfactory data quality. The case study comprises clients interested in environmental temperature values and 1,000 sensor sources capable of producing temperature readings registered to various TDDs.

These systems can be prone to collusion attacks, where a group of malicious sources may cooperate in injecting false data on the blockchain. In our scenario, we represent this attack as follows: the temperature ground truth is set to 25°C}, and the

response from the Truth Inference algorithm is considered to be *accurate* if it falls within a tolerance of ±3°C. Honest sources produce temperature values according to a normal distribution, with the mean equal to the ground truth and the standard deviation uniformly generated between 0 and 4/3 of the tolerance, representing a few honest sources making occasional mistakes. We modeled malicious sources to generate temperature readings by replacing the ground truth with a fairly distant common value (10°C) to simulate a collusion attack, where all attackers concur in redirecting the output to a fake verdict. We then simulated a period composed of 5,000 epochs; a single epoch corresponds to a client request, a response by the system, and a single run of the Truth Inference and the Rating algorithms. We experimentally found that all configurations running more than a few hundred of epochs yield consistent results, hence our choice.

During the first 2,500 epochs, we assume that 50 percent of the sources are already registered in the system, and all of them are honest — this assumption of a "warm start" is realistic because most of the blockchain-based applications run an initial *genesis phase*, where the chain is populated with a solid history of valid blocks in a controlled way. During the remaining 2,500 epochs, we add the remaining sources, including the malicious ones. These can join the network following either a *uniform* distribution, thus randomly picking an epoch, or a *bursty* distribution. In the latter case, we ensure that *all* malicious sources join the network at once, simulating the worst case of colluding sources operating simultaneously.

Our first experiment aims to validate the solidity of the Rating algorithm and the usefulness of blacklisting sources that are identified as malicious. For this reason, we ran simulations adopting a single Truth Inference metric: the median — that is, we consider $\tau(M)$ to return the median of all sensor values in $M$, obtained with a single request. We name this baseline algorithm "*Med*." We then compare its performance against "*Med-R*," in which we additionally perform the Rating algorithm. In particular, our implementation of $\rho(M)$ outputs, for each $\tau \in M$, the distance of t from $\tau(M)$, normalized to output a value between 1 and −1 (in our case, if the distance is more than 3°C, the score will be below 0), picking the minimum value for each source. Next, the reputation of each source is calculated as a convex combination of $R$ and $\rho(M)$, tuned by a parameter $\alpha$, where $R$ is the old reputation value, and $\alpha$ is a parameter that tunes how much the new score affects the reputation — in the simulations, we set it to 0.5. Finally, we show the performance of "Med-RB," where we perform both the above Rating algorithm and the blacklisting step, excluding sources with a reputation below a threshold.

Figure 4 shows the results obtained by executing the three algorithms over different percentages of malicious sources, with the maximum being set to 50 percent, as a higher value, according to the well-known 51 percent problem, would compromise the entire network [14]. The bar chart shows the Truth Inference accuracy, the ratio of client requests that get an accurate response, for uniform and bursty arrival rates. Results show how the rating step significantly affects resilience against a certain number of malicious sources. We also expect a much better

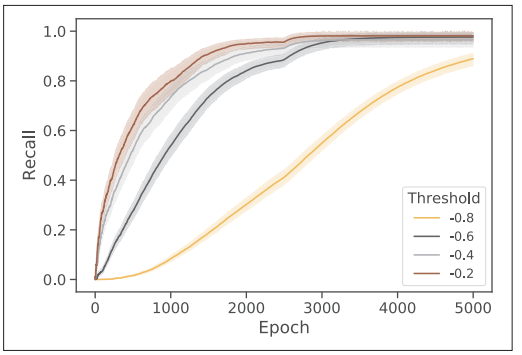


**FIGURE 5.** Blacklisting Recall: the percentage of all the malicious sources that the system is able to detect and ban over time.

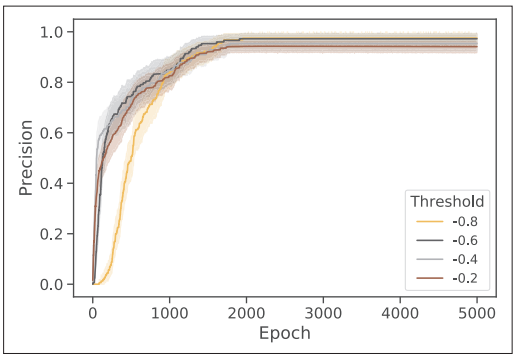


**FIGURE 6.** Blacklisting Precision: the percentage of all the banned sources over time that are actually malicious.

effect against defective or low-quality sources, as in our simulations, malicious sources are performing a joint attack, which is the worst possible scenario. The chart also depicts how blacklisting always has a better impact on inference accuracy. In particular, it allows the system to hold up as many as 50 percent of malicious sources joining all at once, still yielding a high accuracy — around 0.7. Simulations were performed taking into account different values of blacklisting threshold and multiple repetitions.

Furthermore, we study the impact of our Med-RB strategy in terms of "blacklisting precision and recall." The rationale is that different use cases may privilege certain requirements over others. For instance, it may be more important to timely get rid of all malicious sources, while tolerating a reasonable number of honest sources to be kicked out as well (high recall). Conversely, it may be affordable to keep few malicious sources, while ensuring that all banned sources are malicious (high precision). The first case is shown in Fig. 5 over time for different suitable values of "blacklisting threshold." We can see the recall values to be consistent with the expectations: higher thresholds mean less permissive scenarios, thus higher recall. However, we notice that only the most permissive threshold underperforms and, yet, yields a high recall value — above 0.8 — at the end of the simulation. An even better result is shown in Fig. 6, where, no matter the threshold, the precision stabilizes between 0.9 and 1.0, with only tiny differences that are in line with the expectations: the more permissive the policy, the higher the precision and vice versa. High precision scores indicate high reliability and the ability of the technique to identify and kick out malicious data sources with high accuracy, regardless of the threshold. The evaluation confirms that Med-RB is a solid baseline for our scenario, as its

only additional cost is to store the result of the Rating algorithm on-chain, which can be easily controlled by setting an upper bound to the number of selected sources. Med-RB can be used as a first building block for further evolution, including more parameters.

## What are the Relevant Challenges and Research Directions?

Although our system enables different applications, its implementation in complex scenarios poses significant challenges. We list three promising future research directions for decentralized oracles for distributed IoT sources.

### Challenge 1: Truth Inference and Rating Algorithms

Though efficient, as the case study showcased, both the Truth Inference and the Rating algorithms are naive and prone to errors in complex situations. Simple advances in the algorithm could increase its robustness. For instance, we can pair trustworthy oracles with new or low-score data sources to have a high probability of inferring that one node is malicious and vice versa. Assigning trust in distributed decentralized systems is a vast research field that was invigorated by the advent of the blockchain. In particular, guaranteeing IoT devices' trustworthiness is still a considerable challenge. Currently, solutions leverage complex mathematical models that consider various parameters, including the social relationship between the data requester and the device [15].

### Challenge 2: Blockchain Scalability

IoT devices produce a large quantity of data, and we are proposing a system that has the potential to be global. Consequently, scalability is a must. Traditional blockchain strategies struggle to handle the massive number of transactions that a global IoT market will demand. Further, there is the need to rapidly and reliably verify the device's external data (i.e., off-chain) since data is stored permanently in the blockchain. Lightweight — though reliable — consensus algorithms are necessary. There are efforts in the community to tackle this issue, but the entire problem still needs to be solved. One promising direction to provide scalability is blockchain sharding which artificially partitions the workload of one single transaction procession to several members working in parallel. To this aim, we cite STB[4], a blockchain that utilizes sharding and a lightweight consensus to enable IoT scalability.

### Challenge 3: Distributing Micro-Computational Task

We proposed a system solely for sensing IoT data. A considerable improvement is being capable of requesting tasks to be performed. We could leverage the computational power of edge devices to execute computational tasks in a distributed fashion, close to where the data was generated, improving latency and offering Platform as a Service (PaaS) features — as partially explored in IoT-HiTrust[15]. Another direction is to explore the actuation capabilities of IoT devices. Hence, it would be possible to rent an entire fleet of Industry 4.0 robots for a short duration to manufacture a particular custom-made good. Where and how to deploy the computational task and verify that the requested task was executed are open challenges — with some intersections with the first challenge mentioned.


### Acknowledgments

This project received funding from the EU Horizon 2020 through the NGI ONTOCHAIN program under funding agreement No 957338. Icons designed by Eucalyp from Flaticon.


### References


[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A Survey," *Computer Networks*, vol. 54, no. 15, 2010, pp. 2787–2805.

[2] S. Bajoudah, C. Dong, and P. Missier, "Toward a Decentralized, Trustless Marketplace for Brokered Iot Data Trading Using Blockchain," *Proc. 2019 IEEE Int'l. Conf. Blockchain*, 2019, pp. 339–46.

[3] H. Al-Breiki *et al.*, "Trustworthy Blockchain Oracles: Review, Comparison, and Open Research Challenges," *IEEE Access*, vol. 8, 2020, pp. 85,675–85.

[4] H. Moudoud, S. Cherkaoui, and L. Khoukhi, "Towards a Scalable and Trustworthy Blockchain: IoT Use Case," *Proc. IEEE ICC*, 2021, pp. 1–6.

[5] Y. Du *et al.*, "A Novel Oracle-Aided Industrial Iot Blockchain: Architecture, Challenges, and Potential Solutions," *IEEE Network*, Early Acess, 2022.

[6] W3C. "Web of Things Documentation," https://www.w3.org/WoT/documentation, accessed Dec. 14, 2022.

[7] G. Caldarelli, "Understanding the Blockchain Oracle Problem: A Call for Action," *Information*, vol. 11, no. 11, 2020, p. 509.

[8] F. Zhang *et al.*, "Town Crier: An Authenticated Data Feed for Smart Contracts," Proc. ACM SIGSAC, 2016, pp. 270–82.

[9] L. Breidenbach *et al.*, "Chainlink 2.0: Next Steps in the Evolution of Decentralized Oracle Networks," Apr. 2021; https://research.chain.link/whitepaper-v2.pdf, accessed May 29, 2023.

[10] S. Woo *et al.*, "A Distributed Oracle Using Intel SGX for Blockchain-Based IoT Applications," *Sensors*, vol. 20, no. 9, 2020, p. 2725.

[11] J. Benet, "IPFS-Content Addressed, Versioned, P2P File System," arXiv preprint arXiv:1407.3561, 2014.

[12] G. A. Pierro and R. Tonelli, "Can Solana Be the Solution to the Blockchain Scalability Problem?" *Proc. IEEE SANER*, 2022, pp. 1219–26.

[13] C. Aguzzi *et al.*, "A DEcentralized SMart Oracle for the Internet of Things," Feb. 2023; https://github.com/vaimee/desmo, accessed May 30, 2023.

[14] M. Saad *et al.*, "Exploring the Attack Surface of Blockchain: A Comprehensive Survey," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 3, 2020, pp. 1977–2008.

[15] R. Chen *et al.*, "Trust-Based Service Management for Mobile Cloud IoT Systems," *IEEE Trans. Netw. Serv. Manag.*, vol. 16, no. 1, 2018, pp. 246–63.


### Biographies


Lorenzo Gigli (lorenzo.gigli@unibo.it) is a Ph.D. student at the University of Bologna. His research interests include IoT and Distributed Systems.

Ivan Zyrianoff (ivandimitry.ribeiro@unibo.it) is a Ph.D. student at the University of Bologna. His research focus are IoT and Edge Computing.

Federico Montori [M'19] (federico.montori2@unibo.it) is a Assistant Professor at the University of Bologna. His research interests include IoT and Context-Aware Computing.

Cristiano Aguzzi (cristiano.aguzzi@unibo.it) is a Ph.D. and Research fellow at the University of Bologna, Co-Founder of VAIMEE spinoff. His research focus is connecting the physical world to the Web.

Luca Roffia (luca.roffia@vaimee.com) is an Assistant Professor at the University of Bologna, CEO of VAIMEE spinoff. His research interests include interoperable solutions based on Semantic Web.

Marco Di Felice [M'21] (marco.difelice3@unibo.it) is a Full Professor at the University of Bologna and co-director of the IoT PRISM research laboratory. His research interests include IoT, Edge Computing, and sensor data analytics.



Open Access funding provided by Universita di Bologna within the CRUI CARE agreement.