PDS^2: A user-centered decentralized marketplace for privacy preserving data processing

Lodovico Giaretta*§, Ioannis Savvidis†§, Thomas Marchioro‡§,
Šarūnas Girdzijauskas*, George Pallis†, Marios D. Dikaiakos†, Evangelos Markatos‡

* Division of Software and Computer Systems
KTH Royal Institute of Technology
Stockholm, Sweden
{lodovico, sarunasg}@kth.se

† Department of Computer Science
University of Cyprus
Nicosia, Cyprus
{savvidis.ioannis, gpallis, mdd}@cs.ucy.ac.cy

‡ Institute of Computer Science
Foundation for Research and Technology Hellas
Heraklion, Greece
{marchiorot, markatos}@ics.forth.gr

***Abstract*—We envision PDS^2, a decentralized data marketplace in which consumers submit their tasks to be run within the platform, on the data of willing providers. The goal of PDS^2 is to ensure that users maintain full control on their data and do not compromise their privacy, while being rewarded for the value that their data generates. In order to achieve this, our marketplace architecture employs blockchain technology, privacy-preserving computation and decentralized machine learning.**

We then compare different potential solutions and identify the Ethereum blockchain, trusted execution environments and gossip learning as the most suitable for the implementation of PDS^2. We also discuss the main open challenges that are left to tackle and possible directions for future work.

***Index Terms*—iot, blockchain, machine learning, privacy**

I. Introduction

Machine learning (ML) is seeing increased adoption in many different industries, providing services and generating profit. As such, access to vast amounts of data is becoming an important asset for companies and organizations of all sizes, who are therefore eager to collect and store as many data as possible to feed their models. This leads to the creation of large, private data silos and makes it harder for smaller organizations to compete with market leaders, as they lack the leverage to collect large amounts of data and the resources to exploit them. Furthermore, in this process, the data providers often lose control of their data. Once collected by an organization, the original provider has no way of knowing how the data are used, nor getting a share of the value that the organization extracts. This becomes particularly problematic when the data providers are individual users of applications or devices that collect sensitive or personally-identifiable information. Such users typically have little knowledge of how their data are stored and used by organizations and third parties.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 813162. The content of this paper reflects the views only of their author (s). The European Commission/ Research Executive Agency are not responsible for any use that may be made of the information it contains.

§ These authors contributed equally to the paper.

Decentralized blockchain-based data marketplaces have been introduced [1–3], which promise to "democratize" data storage and access. Such marketplaces allow data providers to be rewarded for the value that their data creates for the consumers and lower the entry barrier for smaller entities to access large amounts of heterogeneous data. Unfortunately, to the best of our knowledge, the issues of data control and privacy are not fully solved by these marketplaces. In most cases, the purchased data can be copied outside the platform, where the original provider has no longer control on them. Furthermore, many marketplaces present a business-centered approach. They provide a means for businesses to extract value from their data, but do not cater to the average smart device user, for whom data production is a secondary aspect of owning the device.

To address these issues, we propose PDS^2 (Privacy-Preserving Decentralized Data Sharing System), which is envisioned in this work. Its main goal is to ensure that users maintain full control on their data and do not need to compromise their privacy. The platform should present a flexible, user-centered design that takes into account the needs of all the actors involved. At the same time, it should maintain the desirable properties of previous marketplaces, including decentralization, remuneration and data democratization.

PDS^2 is a trustless marketplace architecture, free of any privileged entity. Data processing is performed in a decentralized network using encrypted computation techniques that guarantee data providers have exclusive control and access to their data. The design is strongly user-centered and provides all actors with ample flexibility and full control, to accommodate their different needs.

This work does not aim to provide a full, production-ready implementation, nor to introduce new technological solutions. Rather, our contribution consists in defining an architecture where well-established or emerging technologies can be embedded. After identifying the stakeholders and their needs, we lay out the components and their interactions within PDS^2, and the technological requirements to implement them. We

then provide a review of state of the art solutions, discussing their suitability in the scope of our architecture. Also, while PDS^2 generalizes to many kinds of workloads, we focus on ML training tasks, as they represent one of the most relevant and valuable data aggregation workloads in the industry.

The rest of the paper is structured as follows. Section II presents the main actors in PDS^2, their requirements and incentives, and the high-level architecture of the marketplace, abstracted from specific implementation methods. Section III then surveys the most promising technologies to implement each core aspect of the platform. Section IV details some of the challenges that need to be addressed in any implementation of PDS^2. Section V reviews similar data marketplaces, while section VI provides directions for future works.

II. High-Level Architecture

A. Platform Actors

Three types of actors take part in the PDS^2 platform. Two of them, who are present by definition in any marketplace, are buyers and sellers. In addition to them, PDS^2 foresees additional actors, whose role is to maintain its internal infrastructure.

1) Sellers: In PDS^2, the sellers can be any data providers. However, as the goal of this work is to present a user-centered marketplace, the focus will be on individual users of smart application or devices. These *end users* differ from other data providers, such as organizations, in several ways. First, the number of end users can be extremely large, while the amount of data each of them produces is limited. Additionally, end users may have limited technical knowledge of the inner workings of the platform, requiring simple control mechanisms to participate in it. Finally, their data may be extremely sensitive, in some cases usable to track and profile them.

2) Buyers: On the buyers (i.e., data consumers) side, the focus of PDS^2 is on organizations, such as companies and research institutions, who need access to the users' data in order to perform aggregation tasks, such as ML model training. This would otherwise require the collection of all the users' data on the premises or cloud of those organizations.

3) Infrastructure: Finally, the internal, infrastructural roles required by PDS^2, which will be detailed in section II-C, can be taken on by the sellers and buyers themselves, but might also be joined by additional actors who provide their computational resources to the platform.

B. Platform Incentives and Actor Requirements

Actors need to be incentivized to participate in the platform. In particular, a new marketplace must provide each of them with additional benefits compared to existing solutions. Furthermore, each actor presents a number of requirements that the marketplace must fulfill.

The main requirements for the sellers are the following:

- **Data control**, i.e., maintaining full ownership of the data, full control over where it is stored and how and when it is accessed.

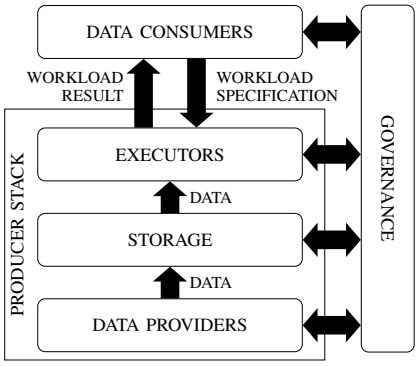

Fig. 1. High-level architecture of the roles and their interactions in PDS^2.

- **Data privacy**, i.e., the guarantee that no entity in the platform will be able to access sellers' data without authorization or infer them based on other information.
- **User-centered data monetization**, which ensures that the value generated by the organizations' workloads is shared back with the users who produced and own the input data.

As all these three aspects are currently missing in most real-world applications, they all play a strong motivating role for the participation of the users as sellers in PDS^2.

On the other hand, buyers, in particular large organizations, would lose the direct access and control of the data that they currently enjoy. However, limited access and control would come with several new benefits, such as lower infrastructural costs and lower legal burden caused by the usage of sensitive data. Furthermore, on PDS^2, organizations would have access to a wider pool of sellers and would be able to request access to any kinds of data made available on the platform, without being limited by what a specific organization has previously collected. The main requirements that the platform should fulfill, from the buyers perspective, are the following:

- **Workload confidentiality**. An organization that pays to execute a workload naturally does not want any other potential consumer to obtain the results for free, by observing the output. Also, the internal details of the workload itself may represent valuable trade secrets of the organization, and should therefore be kept confidential.
- **Data authenticity**. It should be possible for the platform or the workload to identify and reject data that was forged with the intention of affecting the results or claiming undeserved rewards.

Finally, the additional actors that participate in the infrastructure of PDS^2 should be incentivized with a share of the rewards that are offered to the sellers.

C. Platform Roles

The platform consists of five roles, as shown in Fig.1. Each role fulfills one of the core functions in the overall architecture. Each entity that participates in PDS^2, be it an individual user or an organization, can act in multiple roles.

Data consumers prepare and submit workload specifications to the platform. These are binding contracts that specify

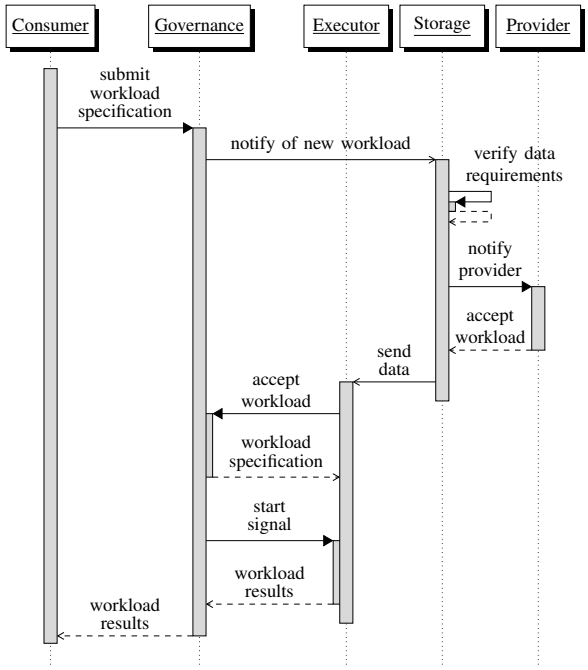

Fig. 2. Sequence diagram of the high-level interactions during the lifetime of a workload in PDS^2.

preconditions that the input data must fulfill, rewards that data providers will receive for submitting valid data, the definition of the workload itself, and any additional conditions, such as minimum amount of data or providers that should have agreed in order for the workload to start executing.

Data providers continuously produce data through their devices, store it in a storage subsystem of their choice, and register it with the marketplace. Data providers are notified of the available workloads for which they have eligible data. They can then choose to submit part of their data to that workload.

The *storage subsystem* is responsible for permanently storing the providers' data. It then matches data against available workloads and gives the executors access to them, when authorized by the providers.

The *executors* provide the computational resources on which the workloads are run. If an executor opts to run a specific workload, and one or more providers opt to use this executor for participating in that workload, the executor will run the workload code on the data provided by them. Decentralized aggregation methods are used to synchronize the results of all executors participating in the same workload, so that the full output can be computed without sharing the input data.

The *governance layer* is responsible for the decentralized orchestration, book-keeping and audit of the platform. It keeps track, among other things, of the available data, the outstanding workloads, the mapping of executors to workloads and the mapping of data to providers and executors. The governance layer is also responsible for distributing rewards and verifying that no actor is behaving maliciously.

D. Execution Workflow

Fig.2 provides a high-level view of the sequence of interactions that take place during the lifetime of a workload.

First, the consumer submits a complete specification of the workload to the governance layer. The storage subsystems of each provider will be notified and will verify whether eligible data is available. If that is the case, the provider will be asked whether to participate in the workload or not.

Once providers accept, they have to identify available executors and submit their data to them, along with certificates confirming that they have indeed accepted to participate in the workload. The executors will then register their own participation with the governance layer. In doing so, they will also submit the certificates from all the participants who sent data to them. This guarantees that all executors have indeed been granted access to a specific set of data for the specific workload in question. Furthermore, the governance layer uses this information to track the contributions of different providers, for the purpose of rewarding them.

Once the conditions set by the consumer in the contract are met, the governance layer instructs the executors to proceed. They then use peer-to-peer communications to compute the workload results in a decentralized manner. The final results are submitted to the governance layer, making them available for the consumer to retrieve.

E. Platform Requirements

Given the goals of PDS^2 and the requirements of the different actors, the technologies used to implement the various aspects of the platform must fulfill a number of requirements.

First, the details of the data and of the workload computation must be invisible to all actors involved, except for the providers and consumers, respectively. Even the executors, while running the actual workload, should preferably be incapable of accessing this information. If technological solutions guarantee that the executors have no direct access to data and code, and no way to tamper with the results without being detected, trust in them becomes unnecessary.

Second, the local computations, which are performed by each executor on a subset of the data, should be aggregated in a decentralized fashion in order to produce the full result. This aggregation should be tamper-proof, free from any bias and should not leak information of the input data.

Third, all actions in the platform should be automatically audited by the governance layer, in a trustless decentralized fashion. This should guarantee that the data, workload and results have not been tampered with by any actor, that all and only the data of willing providers have been fairly used, and that all clauses in each workload contract, such as rewards, are fully discharged.

F. Architectural Flexibility

One of the core properties of this architecture is its flexibility. Each of the three internal roles of the platform (storage, executors, governance) can be implemented using any existing

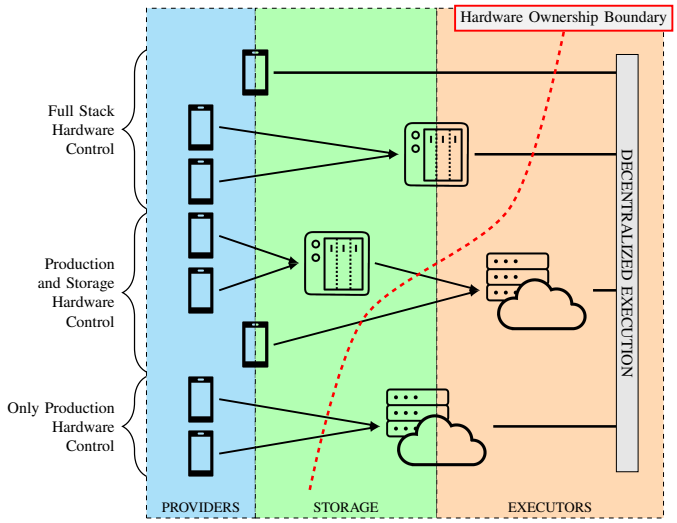

Fig. 3. Different possible hardware configurations for providers, with storage and computation happening either on owned hardware or on third-party servers. Arrows represent data transfers while thick black lines indicate direct participation in the decentralized execution.

or future technology that satisfies the requirements set out in section II-E.

Different technologies may coexist in a single platform, as long as API compatibility is maintained between different roles. For example, different users may use different storage subsystems, based on their particular needs. And consumers may direct the executors to use one of several decentralized aggregation mechanisms, to better suit each specific workload. This allows the platform to seamlessly evolve to integrate technological advancements and shifts in requirements.

Another dimension of flexibility that plays an important role in the user-centered design of PDS^2 is hardware control. While data provider, storage and executor are logically distinct roles, they do not need to be separate in terms of ownership or physical location. Providers can outsource data storage and/or execution to third parties, or can choose to retain control of the entire stack, using their own hardware, as shown in Fig.3.

III. Building Blocks

Given the technological requirements set in section II-E, different potential solutions are examined below. Section III-A discusses the use of blockchain in the implementation of the governance layer. Section III-B compares different techniques to achieve oblivious computation, i.e. to allow the executors to operate on data they cannot directly access. Finally, section III-C analyzes approaches to aggregate results across multiple executors in ML training tasks.

A. Blockchain Technology

Considering the requirements of PDS^2 for decentralization, data integrity and trust, blockchain constitutes a natural building block for the governance layer. It is used for the registration of all actors, by using their blockchain addresses, as well as the registration of datasets and workloads, by means of their hashes. The blockchain itself is not used for storing any datapoints or code, given their size and their potential sensitivity. Smart contracts define the procedures and rules that govern the marketplace. A separate smart contract instance is deployed for managing the lifetime of each workload and validate all of its steps. Each of these smart contracts must follow the basic requirements of PDS^2, but can be personalized to encode the specific requirements of the providers and consumers involved in it. Finally, the blockchain is responsible for distributing rewards via transactions.

Among the many existing blockchain implementations, Ethereum [4] represents the most suitable in this regard. It is a mature platform with a large ecosystem. It offers the ability to develop Turing-complete smart contracts, which enable the complex validation behaviours described. Ethereum also provides tools and specifications that aid in the management of the assets traded on PDS^2. It provides the ERC-20 and ERC-721 standards [5, 6] for *fungible* and *non-fungible* tokens respectively. The former is used to represent divisible, non-unique assets, such as currency, and could be used to handle any kind of rewards offered by the consumers, which would be split among the providers. The latter can instead be used for indivisible, unique assets and can be particularly useful to model data and workload code in PDS^2.

B. Privacy Preserving Data Processing

Here we present the most known techniques for privacy preserving data processing, which have been widely studied especially in the context of machine learning and inference. We focus on lossless techniques which preserve full information since lossy techniques such as data anonymization are often not suitable to make data-driven predictions.

Homomorphic encryption is a set of encryption techniques that allow performing calculations on encrypted data [7]. Homomorphic encryption techniques provide confidentiality guarantees derived from cryptographic principles, making them highly reliable. However, they introduce large overheads in the computation caused by the additional operations performed on encrypted data. Hence they are impractical for most applications, particularly when dealing with a massive amount of data as for the case of IoT.

In *secure multiparty computation* (SMC), the data provider and the data consumer collaborate, sometimes with the help of an untrusted third party, to jointly compute a function over their respective inputs while keeping them secret. Several techniques have been developed to perform SMC [8], allowing to reduce the overhead in comparison to homomorphic encryption. However, the active participation required from the data provider coupled with delays introduced during communication makes it difficult to employ SMC for applications that use many operations.

Trusted execution environments (TEEs) are used as a "neutral ground" to perform computation, where neither of the parties can interfere. TEEs create isolated regions in trusted hardware that cannot be accessed even by the hardware owner,

providing guarantees on privacy and integrity in the code execution. Several works focused on trusted code execution with Intel SGX, a set of instructions compatible with most modern Intel processors that allow creating TEEs called enclaves. In the context of machine learning and inference, several frameworks have been developed for supporting model training, and prediction on SGX [9–11]. A twofold advantage of TEEs is that they introduce smaller overheads compared to homomorphic encryption and unlike secure multiparty computation, no active participation of the data provider is required. Moreover, the trusted hardware constitutes an intermediate node where the data provider has no access to the data consumer's model, while the data providers' data are not accessible by the data consumer. Consequently, TEEs reduce the probability of information leakage drastically. On the other hand, TEEs require a hardware infrastructure, which has a cost comprising both the initial purchase of it and the power consumption to keep such infrastructure alive. Furthermore, moving the computation to the TEE requires to trust the trusted hardware, which may be prone to exploits. In the case of SGX, it has been shown that side-channel leaks are possible but can be avoided using oblivious primitives [12].

Recently, many machine learning frameworks for both privacy preserving training and inference have been developed. Most of them employ the techniques described above and aim to tackle their issues, in particular regarding computational speed. MiniONN [13] reduces the overheads of homomorphic encryption by approximating nonlinear activation functions as piecewise linear functions. It also adopts secret sharing to achieve oblivious computation. Slalom [10] reaches high performance using Intel SGX-based TEEs. Falcon [14], which instead relies on SMC with secret sharing, leverages an untrusted third party to speed-up the computation and is currently the most computationally efficient framework [15]. However, it relies on the assumption that at least 2 out of 3 parties behave honestly.

Although these frameworks have been extensively analyzed by the literature, more practical solutions consist in extensions of popular machine learning libraries such as PyTorch and TensorFlow. Recent works [15] systematically compared the most popular among such privacy-preserving libraries, testing them on image classification models. All of them preserved the accuracy provided by the related native library. Solutions based on combinations homomorphic encryption and SMC showed good performance on smaller models, but failed to scale for larger ones. TEEs solutions, on the other hand, exhibited better scalability.

Moreover, model execution in TEEs offers a number of other advantages. Since the secure environment cannot be accessed even by its owner, it can meet the privacy requirements of two untrusted parties both owning confidential information to be protected, as is the case in PDS^2. In principle, the consumer and the provider do not need to participate at all in the computation, but they can opt to also act as executors, if they own TEE hardware. Their involvement can be regulated by means of smart contracts within the governance layer. Although maintaining the necessary hardware equipment is certainly a downside, most modern Intel processors are compatible with SGX, making the role of executor easily accessible to the wide public. For all these reasons, we identify TEEs as the most promising solution for PDS^2.

C. Decentralized Machine Learning

Given the growing size of machine learning models and datasets, many techniques have been developed to perform training workloads on distributed data [16]. Furthermore, the raising concern for data privacy has pushed research towards massively-distributed techniques, where data are not collected by the consumer, but rather kept close to the provider.

The most prominent of these techniques is federated learning [17], where each executor computes a local gradient, while a central server at the consumer performs the aggregation of all these gradients. Federated learning has been extensively studied [18] and can address the heterogeneity and unreliability of the executors. However, it presents a number of limitations, all related to the presence of a central coordinator. First, this can limit scalability and cause communication bottlenecks. Second, privacy leaks are possible in the training process [19]. Finally, the aggregation of the gradients in a centralized "black box" undermines the transparency of the process and the ability to fairly calculate rewards for the providers.

Recently, several techniques have been proposed to address these limitations, by replacing the central aggregator with a blockchain-based one, with secure multi-party computation, or with trusted hardware [18, 20, 21]. However, an alternative to federated learning exists in gossip learning [22], where collective aggregation is not needed and can thus more readily overcome each of the stated limitations. Gossip learning is based on peer-to-peer communications, in which each node randomly sends and receives model updates from others and merges them with its local updates.

Unfortunately, while gossip learning has been shown to be applicable to many different ML workloads [22–24], no study that we are aware of tested it in actual physical environments or evaluated its practical use for large-scale deep learning training. However, recent studies suggest that gossip learning compares favorably to federated learning [25] and that it can be extended to work in constrained and highly heterogeneous environments [26].

IV. Open Challenges

A. Reward Schemes

The data generated daily is undoubtedly of economic value. However, defining their value is not trivial. According to [27] simplistic solutions such as monetization of data based on size do not work well. This is because data are digital assets whose value is associated with different factors, including the way that they are processed or how many times they are used. In the case study we explore, several datasets are generated and provided by an arbitrary number of users and are utilized for training a predefined ML algorithm. After the training,

the model is purchased by the consumer. Thus one question that emerges is how to share the profits among the users. It is reasonable to assume that each dataset and, consequently, each data provider does not equally contribute to the training's final result.

Deriving from a game-theoretical approach, Shapley value [28] is a promising solution for the aforementioned problem. Shapley value is a function that calculates how to distribute total gains among players in a cooperative game, which in a reward scheme means to determine each data provider's contribution. In PDS^2 this could be done by calculating the ML algorithm's marginal improvement when adding a dataset to it for every possible coalition of the participated datasets and taking the average. However, the complexity of calculating the Shapley value is exponential, and thus it is unfeasible to use it as is. A slice to the problem is also added by the time needed to train a machine learning model [27, 29–31].

Another question that emerges is how the prices should be determined. A possible solution to this question is proposed in [32] who suggested to assign values to ML models, instead of the data. The central concept is that given an ML model, an optimal instance is trained. Then based on the budget available to the potential buyer, Gaussian noise is injected into the model to reduce its accuracy. The larger the buyer's budget, the smaller the injected noise variance and the greater the accuracy.

B. Data Authenticity

Another issue to be addressed is ensuring that the data sold by a certain user are authentic and unique. PDS^2 is mainly focused on data collected by IoT devices or sensors, which can be endowed with digital signature. Data should be signed directly by the device to minimize the risk of forgery, and include timestamps to prevent the user from creating multiple copies and reselling them. The signature is verified by executors, as buyers do not have access to the data. If processing of the data by an external application is required, the manufacturer should ensure that data are not tampered during the transfer from the device to the application. Moreover, the resulting information needs to be resigned after such processing. Since data reliability depends on the security of the device and the quality of the sensors, the signature also serves as a "seal of quality". This influences the price of the device according to the trust that buyers have in the manufacturer. However, although efficient schemes for IoT data signature have been proposed in literature [33], often no authentication is provided by the manufacturers, as their products are usually not designed for selling the produced data.

C. Data Discovery and Filtering

An important issue in any data marketplace is how data can be discovered and filtered. In PDS^2, this translates to two main questions: First, how the workload should convey the data requirements of its consumer. Second, how relevant data can be identified and isolated to ensure that only eligible data providers can participate and be rewarded.

One promising direction, which has gained traction in the IoT domain, is semantic data [34, 35]. This consists of annotating the data with machine-understandable semantic metadata, often based on ontologies. Thus, automated reasoning on the contents of the data and their relationships is allowed.

However, while semantic approaches allow defining complex requirements on the input data, verifying them is not trivial. The storage subsystem should perform this verification before notifying a provider of an available workload. However, this subsystem should not have direct access to the data and should base its decisions only on metadata. This leads to a tradeoff between the amount of information leaked by the metadata and the complexity of the verifiable requirements.

As a complementary approach, the executors could verify the more complex requirements directly on the data, using privacy-preserving computation techniques. This would allow leak-free verification of any requirement. However, it would force the providers to participate in a workload without knowing beforehand the amount of relevant data they possess, and thus their compensation. Furthermore, executors would have to spend computational resources to validate irrelevant data, for which they might not be rewarded.

D. Privacy Leaks

While the technologies presented in this paper aim to prevent the consumers from directly accessing the providers' data, some of that information may still leak to them through the results that they download from the platform. Therefore, it is fundamental for any implementation of PDS^2 to take steps to minimize these leakages.

Several previous works have measured the extent of this issue in the training of ML models [36], and various solutions have been proposed, often based on differential privacy [37]. In PDS^2 the executors could statically or dynamically analyze each workload to assess the risk of privacy leaks and apply the most suitable measures to limit it.

V. Related Work

A variety of previous works proposed **data marketplace architectures**. Most decentralized solutions employ Ethereum blockchain to cover one or more roles. In [1, 38, 39], it is used to handle secure transactions between participants and to mitigate the need for a centralized management authority.

Similarly to PDS^2, Sterling [2] allows privacy-preserving ML training. A main difference is the storage of private data keys in smart contracts. This has the advantage of not requiring user intervention to authorize training, but the drawback of requiring a specialized blockchain and of rendering those smart contracts less auditable.

Fernandez et al.[40] also envision a general data marketplace architecture. Their work focuses on different aspects compared to PDS^2 and does not consider the issues of data control, privacy and decentralization. However, certain aspects of their work are orthogonal to those goals and may be ported to PDS^2, such as data discovery, integration and rewarding.

In terms of **storage**, different approaches have been proposed in literature. Zheng et al. [39] describe a network of Data Agents who store users' data encrypted for a fee and are responsible for data re-encryption when purchased by a consumer. Cloud storage is suggested by Zheng et al. [38] for very large datasets along with symmetric-key cryptography combined with Shamir's secret sharing. The decryption key is split and distributed in special nodes called Key Keepers. On the other hand, Özyılmaz et al. [1] used a decentralized file system called Swarm.

In recent years many **commercial data marketplaces** have emerged. The IOTA Data Marketplace [3] is hosted on the homonymous blockchain, which employs the Tangle[41] as an alternative to traditional ledger designs. Hyperledger Fabric [42] is employed by Datapace [43] to ensure data integrity and network security, and by GeoDB for data validation and reward distribution among stakeholders. GeoDB [44] also uses a mix of other technologies, such as IOTA for data verification transactions, Google Cloud for data storage and Ethereum for reward transactions. The Datum marketplace [45] also combines multiple technologies, such as BigchainDB and IPFS. All these marketplaces are based on fungible tokens compatible with ERC-20.

VI. Future Work

As this paper presented the high-level architecture of PDS^2 and the most suitable technologies for it, the next logical step consists in producing an implementation that can be used to test the feasibility of the platform. That would allow future works to evaluate different technologies to be used as building blocks, such as rewarding schemes, privacy-preserving processing and decentralized aggregation. Moreover, it is essential to evaluate the extent to which the proposed solution is economically viable and whether the monetary and non-monetary incentives provided to individual players are sufficient to drive platform adoption. In particular, the executors need to be compensated for their computational costs, which must be sustainable and competitive compared to existing solutions. Finally, as PDS^2 aims to be a global, open platform, its scalability is an important aspect that needs to be carefully assessed.

VII. Conclusions

This paper proposed PDS^2, a novel data marketplace architecture, which provides a flexible environment where consumers can enjoy a vast pool of heterogeneous data. At the same time, privacy preservation, full data control and fair rewarding are guaranteed to data providers. After analysing relevant technologies, the Ethereum blockchain, trusted execution environments and gossip learning were indicated as the most suitable candidates. Finally, several challenges were identified, that need to be addressed by any future implementation of this architecture.

References

[1] K. R. Özyilmaz, M. Doğan, and A. Yurdakul, "Idmob: Iot data marketplace on blockchain," in *2018 crypto valley conference on blockchain technology (CVCBT)*. IEEE, 2018, pp. 11–19.

[2] N. Hynes, D. Dao, D. Yan, R. Cheng, and D. Song, "A demonstration of sterling: a privacy-preserving data marketplace," *Proceedings of the VLDB Endowment*, vol. 11, no. 12, pp. 2086–2089, 2018.

[3] "The iota data marketplace," accessed: 2020-12-17. [Online]. Available: https://data.iota.org/

[4] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, 2014.

[5] "Eip-20: Erc-20 token standard." [Online]. Available: https://eips.ethereum.org/EIPS/eip-20

[6] "Eip-721: Erc-721 non-fungible token standard." [Online]. Available: https://eips.ethereum.org/EIPS/eip-721

[7] C. Gentry, "Computing arbitrary functions of encrypted data," *Communications of the ACM*, vol. 53, no. 3, pp. 97–105, 2010.

[8] J. I. Choi and K. R. Butler, "Secure multiparty computation and trusted hardware: Examining adoption challenges and opportunities," *Security and Communication Networks*, vol. 2019, 2019.

[9] N. Hynes, R. Cheng, and D. Song, "Efficient deep learning on multi-source private data," *arXiv preprint arXiv:1807.06689*, 2018.

[10] F. Tramer and D. Boneh, "Slalom: Fast, verifiable and private execution of neural networks in trusted hardware," *arXiv preprint arXiv:1806.03287*, 2018.

[11] R. Kunkel, D. L. Quoc, F. Gregor, S. Arnautov, P. Bhatotia, and C. Fetzer, "Tensorscone: A secure tensorflow framework using intel sgx," *arXiv preprint arXiv:1902.04413*, 2019.

[12] O. Ohrimenko, F. Schuster, C. Fournet, A. Mehta, S. Nowozin, K. Vaswani, and M. Costa, "Oblivious multi-party machine learning on trusted processors," in *25th {USENIX} Security Symposium ({USENIX} Security 16)*, 2016, pp. 619–636.

[13] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via minionn transformations," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 619–631.

[14] S. Wagh, S. Tople, F. Benhamouda, E. Kushilevitz, P. Mittal, and T. Rabin, "Falcon: Honest-majority maliciously secure framework for private deep learning," *arXiv preprint arXiv:2004.02229*, 2020.

[15] V. Haralampieva, D. Rueckert, and J. Passerat-Palmbach, "A systematic comparison of encrypted machine learning solutions for image classification," in *Proceedings of the 2020 Workshop on Privacy-Preserving Machine Learning in Practice*, 2020, pp. 55–59.

[16] Z. Tang, S. Shi, X. Chu, W. Wang, and B. Li, "Communication-efficient distributed deep learning: A comprehensive survey," *arXiv preprint arXiv:2003.06307*, 2020.

[17] H. B. McMahan, E. Moore, D. Ramage, S. Hamp-

son, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," *arXiv:1602.05629 [cs]*, Feb. 2016.

[18] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated learning: A survey on enabling technologies, protocols, and applications," *IEEE Access*, vol. 8, pp. 140 699–140 725, 2020.

[19] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 2512–2520.

[20] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng, and Q. Yan, "A blockchain-based decentralized federated learning framework with committee consensus," *arXiv preprint arXiv:2004.00773*, 2020.

[21] U. Majeed and C. S. Hong, "Flchain: Federated learning via mec-enabled blockchain network," in *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2019, pp. 1–4.

[22] R. Ormándi, I. Hegedűs, and M. Jelasity, "Gossip Learning with Linear Models on Fully Distributed Data," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 4, pp. 556–571, Feb. 2013.

[23] I. Hegedűs, Á. Berta, L. Kocsis, A. A. Benczúr, and M. Jelasity, "Robust Decentralized Low-Rank Matrix Decomposition," *ACM Trans. Intell. Syst. Technol.*, vol. 7, no. 4, pp. 62:1–62:24, May 2016.

[24] Á. Berta and M. Jelasity, "Decentralized Management of Random Walks over a Mobile Phone Network," in *2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, Mar. 2017, pp. 100–107.

[25] I. Hegedűs, G. Danner, and M. Jelasity, "Decentralized learning works: An empirical comparison of gossip learning and federated learning," *Journal of Parallel and Distributed Computing*, vol. 148, pp. 109–124, 2021.

[26] L. Giaretta and Š. Girdzijauskas, "Gossip learning: off the beaten path," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 1117–1124.

[27] M. Paraschiv and N. Laoutaris, "Valuating user data in a human-centric data economy," *arXiv preprint arXiv:1909.01137*, 2019.

[28] A. E. Roth, *The Shapley value: essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988.

[29] R. Stanojevic, N. Laoutaris, and P. Rodriguez, "On economic heavy hitters: shapley value analysis of 95th-percentile pricing," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, 2010, pp. 75–80.

[30] A. Ghorbani and J. Zou, "Data shapley: Equitable valuation of data for machine learning," *arXiv preprint arXiv:1904.02868*, 2019.

[31] R. Jia, D. Dao, B. Wang, F. A. Hubis, N. Hynes, N. M. Gürel, B. Li, C. Zhang, D. Song, and C. J. Spanos, "Towards efficient data valuation based on the shapley value," in *Proceedings of Machine Learning Research*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and M. Sugiyama, Eds., vol. 89. PMLR, 16–18 Apr 2019, pp. 1167–1176.

[32] L. Chen, P. Koutris, and A. Kumar, "Towards model-based pricing for machine learning in a data marketplace," in *Proceedings of the 2019 International Conference on Management of Data*, 2019, pp. 1535–1552.

[33] X. Li, H. Wang, Y. Yu, and C. Qian, "An iot data communication framework for authenticity and integrity," in *2017 IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI)*. IEEE, 2017, pp. 159–170.

[34] A. J. Jara, A. C. Olivieri, Y. Bocchi, M. Jung, W. Kastner, and A. F. Skarmeta, "Semantic web of things: an analysis of the application semantics for the iot moving towards the iot convergence," *International Journal of Web and Grid Services*, vol. 10, no. 2-3, pp. 244–272, 2014.

[35] A. Rhayem, M. B. A. Mhiri, and F. Gargouri, "Semantic web technologies for the internet of things: Systematic literature review," *Internet of Things*, vol. 11, p. 100206, 2020.

[36] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 739–753.

[37] T. Zhu and P. S. Yu, "Applying differential privacy mechanism in artificial intelligence," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 1601–1609.

[38] X. Zheng, R. R. Mukkamala, R. Vatrapu, and J. Ordieres-Mere, "Blockchain-based personal health data sharing system using cloud storage," in *2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom)*. IEEE, 2018, pp. 1–6.

[39] S. Zheng, L. Pan, D. Hu, M. Li, and Y. Fan, "A blockchain-based trading platform for big data," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2020, pp. 991–996.

[40] R. C. Fernandez, P. Subramaniam, and M. J. Franklin, "Data market platforms: trading data assets to solve data problems," *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 1933–1947, Jul. 2020.

[41] S. Popov, "The tangle," 2016.

[42] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the thirteenth EuroSys conference*, 2018, pp. 1–15.

[43] D. Draskovic and G. Saleh, "Decentralized data marketplace based on blockchain," p. 16, 2017.

[44] GeoDB Blockchain Limited, "Geodb," 2020.

[45] R. Haenni, "Datum Network - The decentralized data marketplace," p. 37, 2017.