PReVer: Towards Private Regulated Verified Data

Mohammad Javad Amiri, Tristan Allard, Divyakant Agrawal, Amr El Abbadi

Vision Paper

PReVer: Towards Private Regulated Verified Data

Mohammad Javad Amiri[1] Tristan Allard[2] Divyakant Agrawal[3] Amr El Abbadi[3]
[1]University of Pennsylvania, [2]Univ Rennes, CNRS, IRISA, [3]University of California Santa Barbara
[1]mjamiri@seas.upenn.edu [2]tristan.allard@irisa.fr [3]{agrawal, amr}@cs.ucsb.edu

ABSTRACT

Data privacy has garnered significant attention recently due to diverse applications that store sensitive data in untrusted infrastructure. From a data management point of view, the focus has been on the privacy of stored data and the privacy of querying data at a large scale. However, databases are not solely query engines on static data, they must support updates on dynamically evolving datasets. In this paper, we lay out a vision for privacy-preserving dynamic data. In particular, we focus on dynamic data that might be stored remotely on untrusted providers. Updates arrive at a provider and are verified and incorporated into the database based on predefined constraints. Depending on the application, the content of the stored data, the content of the updates and the constraints may be private or public. We then propose *PReVer*, a universal framework for managing regulated dynamic data in a privacy-preserving manner. We explore a set of research challenges that PReVer needs to address in order to guarantee the privacy of data, updates, and/or constraints and address the consistent and verifiable execution of updates. This opens the space of privacy-preserving data management from the narrow perspective of private queries on static datasets to the larger space of private management of dynamic data.

1 INTRODUCTION

Data privacy[1] has garnered significant attention recently due to diverse applications that store sensitive data in untrusted infrastructure. Data privacy and enabling secure query processing on data using cryptographic techniques have been addressed in several systems [4, 17, 20, 21, 30, 33, 39, 41, 44, 47, 52, 55, 57]. As an example, Prism [47] presents a secret-sharing-based technique to answer queries in the form of private set operations as well as aggregation functions over outsourced databases belonging to multiple owners. The focus of such private database systems, however, is mainly on static databases and they do not support updates on dynamically evolving databases in a satisfactory manner [62].

Privacy-preserving dynamic data management has been addressed in a few systems where the focus is on answering queries in a privacy-preserving manner (e.g., [32, 40, 43, 53, 57, 59]), in supporting high insertion rates (e.g., [61]), or in strengthening the privacy model (e.g., [62]). For instance, DP-Sync [62] addresses the problem of hiding timing database update patterns and guarantees that the entire update history over the outsourced data (in which only a single table schema is supported) is protected using differential privacy techniques. None of these dynamic data management systems, however, support regulated data where data are being updated frequently and updates must satisfy predefined constraints before being incorporated into the data while the privacy of updates, regulations, and/or data needs to be preserved.

Regulated dynamic data have a wide range of applications from single-enterprise settings, where the enterprise data is typically maintained in an outsourced database within an untrusted infrastructure, to multi-enterprise settings, where data is maintained in multiple independent databases owned by a set of mutually distrustful parties. Constraints might be internal constraints defined by the data owners or global regulations defined by external authorities, e.g., some national or governmental institution. For example, in supply chain management, terms of collaborations are internal constraints defined in service level agreements (SLAs) which are agreed upon by all involved participants while in a crowdworking environment, the *Fair Labor Standards Act*[2] (FLSA) is a global regulation that limits the total work hours of a worker per week to not exceed 40 hours.

Verifying incoming updates with respect to such constraints and incorporating the verified updates into data maintained by either untrusted or mutually distrustful infrastructures, on one hand, and guaranteeing the integrity of the stored data, on the other hand, while preserving the privacy of updates, constraints, and data makes the problem challenging. The challenge with respect to privacy is that data managers may not be allowed to (1) view the update, (2) view the data itself and (3) be necessarily aware of the constraints that allow or disallow updates to be incorporated into the databases.

In this paper, we lay out a vision to support privacy-preserving regulated dynamic data. Once data is stored, managing data requires the support of privacy-preserving queries as well as privacy-preserving updates. The underlying infrastructure might consist of a single database owned by a data owner, e.g., an enterprise, or multiple independent databases owned by different distrustful data owners, e.g., different enterprises in a supply chain management. Furthermore, data might be stored remotely on untrusted providers, e.g., cloud servers. While outsourcing data to third parties like the cloud results in higher availability, ease of maintenance, and lower costs, the privacy concerns of data owners need to be addressed. In fact, depending on the application and the underlying infrastructure, the content of the stored data, the content of the updates, and the constraints may be private or public.

In addition to privacy, our vision addresses *integrity* in order to manage regulated dynamic data. Integrity ensures the checking of updates against constraints and incorporating them into data and enforces the consistency and accuracy of the stored data.

We then present *PReVer*, a universal framework for managing regulated dynamic data in a privacy-preserving manner. PReVer, on one hand, needs to guarantee (1) the *integrity* of updates against constraints, (2) the integrity of updates on stored data and (3) the integrity of stored data itself with both single and federated databases and, on the other hand, should be able to preserve (1) the *privacy* of data, (2) the privacy of updates, or/and (3) the privacy of constraints.

[1]We overload below the term *privacy* for referring to both the *privacy of personal data* (our main focus) and to the *confidentiality of data that is non-personal.*

[2]https://www.dol.gov/agencies/whd/flsa

We explore the roadmap leading to systems instantiating the PReVer vision by studying a set of research challenges in two dimensions of privacy and integrity. To guarantee the integrity of updates against constraints and on stored data, PReVer can rely on different privacy-preserving techniques, e.g., fully homomorphic encryption for single database settings [36], secure multi-party computation and/or token-based mechanisms for federated database settings [10], private information retrieval for public data [31]. To guarantee the integrity of stored data, the related information should be stored in an immutable and verifiable manner. PReVer relies on append-only ledgers as an immutable verifiable data structure and uses centralized ledger databases [1, 63] and permissioned blockchains [7–9, 11, 14, 27, 37, 38, 42, 45, 51, 56, 58] as its infrastructure for single and federated database settings respectively.

PReVer provides database researchers with a framework to explore how to support the scalable efficient execution of private updates on private data sets with constraints. Addressing this challenge requires providing support for the private verifiability of updates on data that is regulated by private or public constraints in a scalable manner as well as a better understanding of information leakage when updates are verified with respect to constraints, where the updates or the constraints may be private or public. This framework can be instantiated using data that is both private or public and where the data is either centralized or sharded across multiple entities in a federated setting. Finally, solutions can be both centralized or distributed, thus providing designers with a range of challenges to explore, especially given the need for practical scalable performance.

The rest of this paper is organized as follows. Section 2 motivates privacy-preserving dynamic data management using several applications. A model for regulated dynamic data is presented in Section 3. Section 4 presents PReVer and explores its research challenges. Section 5 discusses how PReVer can support a specific application, and Section 6 concludes the paper.

2 MOTIVATION

In this section, we motivate the problem of privacy-preserving dynamic data management using diverse applications. In each case, each of the data, the updates and the constraints may be private or public and data are stored in a single database or multiple databases (i.e., federated setting). Figure 1 presents some of these applications.

2.1 Environmental Sustainability

Environmental regulation certification is an important verification process that is critical for successful environmental sustainability. Consider an organization that wants to verify and certify its environmental sustainability efforts with respect to public quantitative sustainability metrics, e.g., ISO 14000 series of standards or LEED (Leadership in Energy and Environmental Design). These claims are certified by a recognized authority that awards Platinum, Gold, or Silver Certificates to companies based on their environmental sustainability statistics or procedures. Usually, since the statistics reflect potential internal organizational operations and hence the organization would like to be certified without revealing sensitive information to the certifying authority, other parties, or the public. Furthermore, these statistics may need to be continuously updated based on changes in the internal processes in these organizations. The data reflecting the organization statistics are stored at the certifying authority as long as the data and the updates are private to the company and are not visible to the certifying authority. However, the regulations are public and widely advertised and are used by the certifying authority to accept or reject specific updates.

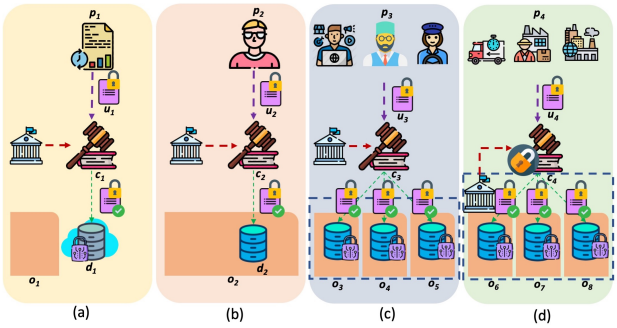

Figure 1: Some possible applications of privacy-preserving dynamic data management

As shown in Figure 1(a), if a private update satisfies a public regulation that is issued by some public regulatory, e.g., the International Organization for Standardization, it is executed on a private database. The private database is owned by data owner, however, it is *outsourced* to a third party, e.g., cloud provider. In this case, the constraints are public, but both the data and the updates need to be private and hidden from the certifying authority. This scenario is typical of many environmental applications, and in general, the certifying authority might also advertise the ranking among a set of competing organizations, based on their environmental sustainability statistic.

2.2 In-Person Conference Participation

Consider a conference which proposes both in-person and online participation, however, requires a valid COVID vaccine certification for in-person participation. In this application, the data, the list of in-person attendees, is public, but the update application of the participants demonstrating their vaccination records is private. Needless to say, the constraints regarding admission are public (e.g., testing, vaccination, etc.). As shown in Figure 1(b), For example, when a private update, e.g., a registration record, satisfies a public constraint, e.g., valid COVID vaccine certification requirement, it is executed on public database, e.g., the list of in-person participants.

2.3 Multi-Platform Crowdworking

The previous applications all illustrate a single database, where data was stored. Now, we explore multi-site infrastructures. In a multi-platform crowdworking environment, multiple independent competing platforms (e.g, Uber, Lyft, etc) perform private updates (i.e., crowdworking tasks) initiated by participants (i.e., requesters and workers). Consider, for example, a driver who is willing to work for both Lyft and Uber. Whenever the driver performs a task, this is considered an update. Each of Lyft and Uber needs to maintain private records for each driver. However, such updates must satisfy public global regulations, e.g., the total work hours of a worker, who might work for multiple crowdworking platforms, per week may not exceed 40 hours to adhere to *Fair Labor Standards Act*[3] (FLSA). In this case, private updates need to be executed on multiple private databases stored on platforms that do not trust each other. However, the platforms need to

[3] https://www.dol.gov/agencies/whd/flsa

cooperate to verify the public regulations without compromising privacy. Figure 1(c) presents such an application with a federated database system comprising multiple autonomous databases.

2.4 Supply Chain Management

Supply chain management is another example of federated database systems where multiple mutually distrustful collaborating enterprises process incoming updates. Lack of trust between different parties is one of the most important problems in supply chain management. Various updates need to be considered, both internal and cross-enterprise, where in contrast to the cross-enterprise updates which are visible to multiple enterprises, the internal updates of each enterprise are confidential, e.g., the internal updates of a Manufacturer demonstrate its internal process for producing a product which the Manufacturer might intend to keep as a secret. In such an application the constraints are written as service level agreements agreed upon by all enterprises, hence may be considered public constraints, while depending on the use case data and updates might be public or private. Figure 1(d) shows a supply chain management application with private data, updates, and constraints.

3 A MODEL FOR PRIVATE REGULATED DYNAMIC DATA

Managing dynamic data in a privacy-preserving manner while guaranteeing the satisfaction of predefined constraints is challenging. In this section, we present a model for privacy-preserving regulated dynamic data management. To preserve the privacy of regulated dynamic data, a threat model also needs to be specified.

3.1 Participants

The model consists of four participants roles: *data producers*, *data owners*, *data managers*, and *authorities*, where a single entity might assume multiple participant roles.

Data producers. Data producers produce updates. Data producers might be clients of the system with their own privacy concerns, e.g., requesters in crowdworking environments, or might be public sources of information, e.g., a set of public polar-orbiting satellites that collect data for weather, climate, and environmental monitoring applications.

Data owners. Data owners own the data and either maintain data locally or outsource the data to an external third party that manages the data on behalf of the data owner. Data is maintained in either a *single* database or *multiple* databases, each owned by a different data owner. Several data owners may co-exist (e.g., federated contexts) and collaborate.

Data managers. Data managers store and manage data on behalf of data owners. They are responsible for incorporating updates that are consistent with regulations and constraints into the data as well as responding to queries. We focus on updates as privacy-preserving queries have been extensively studied in the literature. If data is stored locally at the data owner, then the data owner also subsumes the role of data manager. However, when data is outsourced to a third party, e.g., a cloud provider, the third party subsumes the role of data management and in general, needs to perform both updates and queries in a privacy-preserving manner. In the case of a single database and to fully demonstrate the privacy challenges, we only focus on outsourced data.

Authority. Authorities are mainly in charge of defining constraints. Depending on the type of constraints, i.e., private or public, the authority is either internal or external. An internal authority is equivalent to the data owner who specifies constraints on incoming updates whereas an external authority is an official institution that issues regulations that need to be satisfied by updates before they are incorporated into the database.

3.2 Constraints, Updates and Data

A set of constraints specifies the set of allowable database states. A constraint is essentially a Boolean function computed over the database and an incoming update and expresses a policy for accepting or rejecting incoming updates. Constraints specified by the data owner (e.g., the well-known database constraints) are also called *internal constraints*. The scope of internal constraints is typically limited to the database(s) of the corresponding data owner: it does not span the databases of multiple data owners. However, constraints may also come from external authorities (e.g., the laws, ethics). We refer to these as *regulations* below. Unlike internal constraints, a regulation may constrain the databases of multiple data owners (e.g., the money earned monthly by a crowdworker across multiple crowdworking platforms). Given the large body of work for expressing and evaluating database constraints based on data-driven declarative query languages (e.g., relational calculus), these languages are thus a natural choice for expressing regulations. Temporal logic extensions may additionally be relevant for regulations because they often express temporal constraints on sliding time windows, e.g., workers cannot work more than 40 hours a week.

An update may involve several participants including at least a data producer and a data manager (e.g., crowdworking application platform). In general, however, an update may originate from the collaboration between several data producers and/or several data managers. For example, consider the multi-platform crowdworking environment. The completion of a crowdworking task is an update that results from the collaboration between two data producers (i.e., a worker and a requester) and at least one data manager (i.e., a crowdworking platform).

An internal constraint is written by an internal authority. As a result, its scope is limited to the database(s) of the corresponding data owner: it does not span the databases of multiple data owners while regulations may constrain the databases of multiple data owners, e.g., the money earned monthly by a crowdworker whatever the crowdworking platform.

3.3 Threat Model

Participants such as data managers or clients may adopt adversarial behaviors against the privacy and/or the integrity of the constraints. The threat models considered in PReVer are the usual adversarial models considered in the literature. The choice of a given threat model is not fixed but will depend on each instantiation of the framework, according to the adversarial contexts targeted.

The usual threat models can qualify the kind of adversarial behaviors expected from any kind of participant: honest participants are not expected to adopt adversarial behaviors, honest-but-curious participants strictly follow the algorithm but aim at performing any possible inference (e.g., a dubious outsourced data manager), covert participants deviate from the algorithm only if they are not detected (with a probability above a given threshold), and malicious adversaries deviate arbitrarily from the algorithm. Furthermore, participants may or may not collude. For example in the multi-platform crowdworking instantiation of

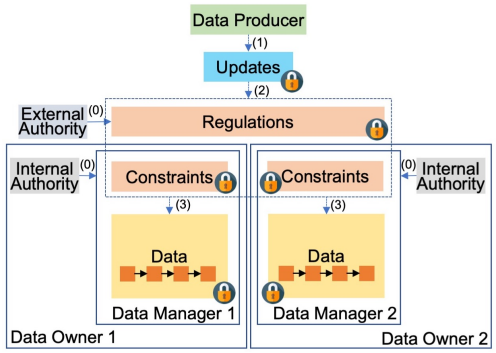

Figure 2: PReVer framework consisting of two databases. (0): Authorities define constraints and regulations, (1): The data producer sends an update, (2): The update is verified with respect to regulations and constraints, and (3): The update is incorporated into data.

PReVer, a covert (colluding) adversary model can be considered where participants (i.e., requesters, workers, platforms) may act as covert adversaries and might collude with each other.

Distinguishing the various roles of participants reflects real-life practices. This is well understood by modern data protection laws (see, e.g., GDPR Article 28 about outsourcing, or GDPR Article 32 about the security of processing). The problem of updates is directly related to these laws. Indeed, when data is personal data, it directly falls under the umbrella of the GDPR, and hence, the updates incorporated in PReVer as well. As a result, it is important to understand the leakage of information due to the enforcement of constraints over private data, and hence the need to be protected. However, we note that the case of private constraints over public data is probably more related to intellectual property rather than personal data protection and thus might fall under different legal frameworks.

4 PREVER FRAMEWORK

PReVer is a universal framework for managing regulated dynamic data in a privacy-preserving manner. PReVer needs to preserve *privacy* while ensuring *integrity* in order to guarantee consistent and verifiable execution of updates. PReVer must therefore provide the context for verifying updates against constraints, applying updates to stored data, and maintaining the consistency of stored data while preserving the privacy of updates, constraints and/or data. Figure 2 presents the PReVer framework with two databases. As updates arrive, PReVer needs to provide mechanisms to verify that updates satisfy regulations provided by external authorities or constraints provided by internal authorities. Once verified, the update needs to be applied to the databases. Finally, PReVer needs to provide techniques that guarantee that databases have not been corrupted.

In general, PReVer needs to deal with several challenges. First, while most privacy-preserving techniques focus on the querying goal, PReVer needs to verify updates that results in additional challenges, e.g., disclosing update patterns [62]. Second, most privacy-preserving techniques do not scale and are computationally expensive. Third, constraints and regulations have not been supported in any of these techniques. They require the untrusted data manager(s) to perform computations on the encrypted database and not only to retrieve tuples.

In this section, we explore the roadmap leading to systems instantiating the PReVer vision. We explore a set of challenges that PReVer needs to address in order to guarantee the privacy-preserving integrity of updates against constraints and on stored data as well as ensuring the privacy-preserving integrity of stored data. For each of these challenges, we highlight the limitations of current privacy-preserving mechanisms.

Single private database. PReVer needs to address the setting where an untrusted data manager maintains the data, e.g., atmosphere monitoring databases application or environmental sustainability application.

Research Challenge 1. *Enable an untrusted data manager to verify updates against constraints and execute updates on private data in a privacy-preserving manner.*

Traditionally, when a query is private, privacy-preserving querying techniques hide it by encrypting it or by generating trapdoors that enable the data manager to perform the search without leaking the query. In PReVer, a constraint can be private, but hiding it requires to go beyond hiding the search values in order to protect the complete Boolean function that implements the constraint. Moreover, an update must eventually be executed on data or discarded depending on the Boolean output of the constraint. Performing or discarding updates must be done without jeopardizing privacy guarantees.

To address Research Challenge 1, PReVer can be built on existing privacy-preserving querying techniques such as *fully homomorphic encryption* [24] (thus providing software protected computation) – or more generally *functional encryption* [25] – and *zero-knowledge succinct non-interactive arguments of knowledge* [35] to address the integrity of constraints (basically requiring the untrusted manager to provide verifiable proofs that they actually perform the correct actions they claim.). Such techniques, however, have considerable overhead [64–66]. Alternatively, PReVer can rely on differentially private indexing, i.e., partial disclosures [57, 61]. However, naive uses of differential privacy lead to rapidly exhausting the limited privacy budget, especially when updates come at a high rate. This results either in an impossibility to support additional updates or in an uncontrolled increase of the noise magnitude. To improve the performance of updates, secure hardware, i.e., hardware protected computation [13, 16, 18, 54, 60] can be used. However, secure hardware has scalability issues.

Multiple private databases. Thus far, we have focused on a single database, however, as our application examples clearly demonstrate, e.g., multi-platform crowdworking or supply chain management, many applications require a federation of databases. These databases need to collectively verify constraints in a distributed manner and apply updates to individual databases without violating the distributed constraints.

Federated contexts require distributed solutions for allowing multiple data managers, trusted or not, to verify updates against constraints and to execute them on stored data in a collaborative manner while protecting the data they host, and, if required, the constraints and the updates.

Research Challenge 2. *Enable a set of trusted and untrusted federated data managers to verify distributed constraints over distributed private data and to perform updates conditionally, all this with sound privacy and integrity guarantees.*

Traditionally, such problems have been solved using *centralized* or *distributed* approaches. In a centralized approach, a token-based mechanism can be used to ensure the verifiability of distributed constraints over distributed private data. A token-based

system, e.g., e-cash, implements constraints and regulations by managing budgets per participant called *tokens*. Tokens are single-use cryptographic tokens (i.e., a nonce signed by a trusted authority stored in a public database once spent) that are generated periodically by an external authority and distributed to corresponding participants. Alternatively, secure multi-party computation [28] can be used as a decentralized approach. Using secure multi-party computation, different data managers could collaborative verify updates against constraints and apply it to their databases (see, e.g., [6, 19] in un-protected contexts).

To address Research Challenge 2, PReVer can be built on a centralized token-based mechanism and decentralized secure multi-party computation. However, token-based mechanisms can only address simple COUNT-aggregate queries and it is unclear how they can be generalized to support other kinds of constraints. Traditional *secure multi-party computations*, on the other hand, either do not handle updates or usually assume each of the distributed parties is trusted (just that they do not trust each other). However, in the context of PReVer when some data managers are untrusted, the challenge is especially hard because these data managers have no access to raw data and consequently, their computation abilities are strongly reduced. Applying updates satisfying the constraints to each federated data manager rises additional challenges similar to those discussed earlier. Furthermore, in a dynamic setting, PReVer can benefit from the efficient incremental techniques. In general constraint enforcement in a privacy-preserving manner has not been sufficiently explored and seems to be an open area for future research.

Public databases. Finally, PReVer needs to address applications, e.g., in-person conference participation during the COVID pandemic, where the data itself is public, but the constraints and updates might be private.

Research Challenge 3. *Enable a data manager to verify updates against constraints over public data and execute the updates with sound privacy guarantees on the updates.*

Private Information Retrieval (PIR) techniques [31, 48] have been traditionally used to address privacy-preserving queries when data is public. PIR allows users to retrieve items from a public database by specifying an index without revealing the items retrieved.

To address Research Challenge 3, two challenges need to be addressed. First, PIR techniques have been restricted to retrieving a single item in a privacy-preserving manner. This needs to be extended with computational capabilities to verify complex (SQL-like) constraints irrespective of whether they are private or public. If the constraints are private, there is an additional need to consider the non-negligible leak about private constraints resulting from applying updates to public data. Second, while PIR techniques are designed primarily to support private retrieval of information, in PReVer, these techniques need to be extended to support updates. Recently, there have been many attempts to improve the performance of PIR [5, 15, 49], however, more research needs to be conducted to efficiently support updates.

Integrity of stored data. Finally, PReVer needs to guarantee the integrity of stored data in both single and federated databases. The main challenge is to enable any of the participants to verify the integrity of stored data in a privacy-preserving manner.

Research Challenge 4. *Enable any participant to verify the integrity of stored data with sound privacy guarantees.*

In PReVer, data needs to be stored in an *immutable* and *verifiable* manner. Immutability prevents any malicious participants from altering data and verifiability enables all (authorized) participants to check the state of the system. *Append-only ledgers* provide the immutability and verifiability features required by PReVer. Append-only ledgers use *authenticated data structures* such as Merkle trees [50] to ensure that data is tamper-resistant when the underlying infrastructure is untrusted. When there is a single database maintained by a single data manager, the *centralized ledger* technology can be used as the infrastructure of PReVer where a ledger database provides tamper-evidence and verification features in a centralized manner. Alibaba LedgerDB [63], Amazon QLDB [1], and ProvenDB [2] are some of the existing centralized ledger databases. *Verifiable database techniques* [22, 29, 34] have also been used to verify the integrity of stored data. Such techniques, however, are more specific to database queries and their ability to support computation beyond the search is unclear.

To address Research Challenge 4, these techniques need to be extended to the distributed setting. Since an update can be processed on all or a subset of databases, processing updating requires establishing consensus among all involved data managers. Moreover, the data managers might not trust each other. To address these challenges, *permissioned blockchain* systems, e.g., Hyperledger Fabric [14], can be used as the infrastructure of PReVer. While Fabric uses private data collections [3] to ensure confidentiality, its computational overhead is significant and results in reduced throughput. To address the computational overhead of Fabric, PReVer can leverage some of the techniques developed in Qanaat [12]. In particular, to support confidentiality, Qanaat enables every subset of data managers to form a confidential collaboration private from other enterprises thus enabling them to execute updates. Qanaat further provides scalability by partitioning data into data shards. These techniques can be leveraged by PReVer in order to ensure the integrity of stored data in a privacy-preserving manner.

5 AN APPLICATION OF PREVER

The goal of PReVer is to present a general framework for managing regulated dynamic data in a privacy-preserving manner. In general, choosing the right set of techniques depends on three main criteria: (1) data is private or public, (2) the database is single or federated, and (3) the instatiation is centralized or decentralized. In this section, we discuss how PReVer can be used to support multi-platform crowdworking applications. Our goal is to provide a specific instance of PReVer and to illustrate how a particular system, Separ [10], was able to solve several of the challenges discussed in Section 4. In Separ's instantiation of PReVer, the data and the updates are private while the constraints are public, the database is federated, and a centralized token-based approach is chosen. Furthermore, we highlight several of the shortcomings of Separ, thus demonstrating the need for future research to address general solutions for privacy-preserving crowdworking applications.

Separ is a multi-platform crowdworking system, where workers may need to work for or avail from the services of multiple crowdworking platforms, e.g., a driver may work through both Uber and Lift. Separ needs to enforce public global regulations (e.g., future of work labor laws like a limit on the number of hours worked), in a privacy-preserving manner, on a set of distributed independent platforms that are mutually distrustful of

each other. Workers are the data producers and data owners in PReVer, while the multiple platforms are the data managers. Separ uses a trusted third party to act as the authority that expresses public regulations. Platforms do not trust each other and workers do not want to share their personal data (e.g., tasks completed, exact number of hours worked) from one platform to another. However, the global regulatory constraints need to be enforced in a privacy-preserving manner. As a result, the constraints are the (public) upper-bound regulations[4], the data is the (private) task completion database (Separ does not need to instantiate it but keeps it virtual), and the updates are (private) information about the completion of a task (e.g., task completed, time spent, requester, platform).

Separ uses a centralised trusted authority to enforce regulations using single-use, pseudonymous tokens, which ensure that mutually distrustful platforms can cooperate to enforce the distributed constraints for the conditional execution of updates (Research Challenge 2). It relies on the permissioned blockchain system SharPer [9] to guarantee integrity of the global system state (i.e., the tokens spent – Research Challenge 4). The global system state is shared among the mutually distrustful crowdworking platforms using distributed consensus protocols over the blockchain ledgers (for immutability and verifiability).

Separ represents a particular instantiation of PReVer. A general multi-platform crowdworking application would need to address several of Separ's limitations. We discuss some of these here, and leave the solutions for future work. Separ requires a centralized trusted third party authority to issue tokens. This is a serious shortcoming, as a general distributed approach should be used to enforce distributed constraints on updates in a privacy preserving manner. Furthermore, Separ focuses on lower and upper bound constraints. Future systems should support general distributed constraints, e.g., any `SQL` expressed constraints, including `GROUP BY`, `JOIN` and aggregate expressions, which might require the use of *secure multi-party computations*. Furthermore, the adversarial model in Separ assumes that participants do not collude, which is not realistic in many adversarial settings. A general multi-platform solution should assume collusion among participants and provide efficient protocols in such diverse settings. The PReVer framework also highlights some shortcoming in the Separ approach, namely restricting regulations to public constraints as it is quite realistic to assume constraints among a subset of the platforms. Another shortcoming highlighted by

PReVer can be used to support many other applications. For instance, to support supply chain management applications (Figure 1(d)), PReVer can use permissioned blockchains to guarantee the integrity of stored data (Research Challenges 4). To address the integrity requirement in the execution of smart contracts, PReVer can rely on zero-knowledge proofs when the smart contract depends on the private data of a single database: the data manager who knows the secret can run the smart contract on its own, and then prove to everyone else that it did so correctly. However, zero-Knowledge Proofs are not sufficient in settings where the smart contract depends on the secret information of multiple data managers [23]. To address Research Challenge 2, using secure-MPC protocols to support private data on Hyperledger Fabric has been proposed [23] where the execution of the secure-MPC protocol is integrated as part of the smart contract. PReVer can also rely on RIP to address Research Challenge 3 in supply chain management when data is public.

[4]Separ also supports lower-bound regulations.

6 DISCUSSION AND CONCLUSIONS

In this paper, we lay out a vision to design *PReVer*, a universal framework for managing regulated dynamic data in a privacy-preserving manner. Data in PReVer is maintained by either untrusted or mutually distrustful infrastructures and updates need to be verified and incorporated into databases based on predefined constraints.

Any implementation of the PReVer framework needs to address several critical research questions that we pose to the database community as challenges that need to be addressed to ensure the successful management of regulated privacy preserving dynamic data management systems. In a nutshell database and systems researchers need to clearly demonstrate how to support *scalable efficient consistent and verifiable execution of updates on data with constraints while preserving privacy.*

Designing such systems requires solving multiple challenges:

- **Private verification.** The efficient verification of an update with respect to a given generally formulated constraint where the update and the constraint or both might have privacy constraints with respect to the data manager.
- **Private updates.** The efficient execution of an update on data where the update and the data or both might have privacy constraints with respect to the data manager.
- **Scalable solutions.** These solutions need to scale with respect to the frequency of updates as well as the size of the data.

PReVer thus requires a better understanding of information leakage due to the enforcement of constraints on updates. Furthermore, these challenges need to be addressed in multiple contexts:

- **Private and public databases.** The data is private or public with a stream of updates and data constraints that may be private.
- **Centralized and federated decentralized databases.** The data can be stored in one centralized location or may be sharded across multiple sites, with different organizational supervision.

PReVer provides a framework that would allow database, cryptography and distributed systems researchers to explore implementing parts of this general vision. In particular, the framework facilitates the ability for researchers to restrict their practical implementation to the most appealing context they are interested in, e.g., centralized private data, federated public settings, *etc.*

PReVer presents a framework based on four main components: data owners, data providers, data managers, and regulatory authorities. These abstractly capture the essential and relevant components needed for a focused study of private dynamic data management and would allow researchers to identify the context they wish to explore and hence focus on the efficiency and scalability of their solutions. It is interesting to note that in spite of the variety of contexts, from a storage point of view, the ledger model seems quite versatile for so many varied contexts.

Ideally, and to ensure solutions are actionable, comparisons should be performed with respect to non-private solutions using standardized database benchmarks like TPC and YCSB. Furthermore the distributed solutions should be compared in terms of throughput and latency with standard distributed fault-tolerant protocols, e.g., Paxos [46] and PBFT [26].

ACKNOWLEDGMENTS

This work is funded by NSF grants CNS-1703560, CNS-1815733, and CNS-2104882 and by the ANR grant ANR-16-CE23-0004.

REFERENCES

[1] [n.d.]. AWS. Amazon quantum ledger database (QLDB). https://aws.amazon.com/qldb/.

[2] [n.d.]. A complete guide to the philosophy of ProvenDB. https://www.provendb.com/litepaper/.

[3] [n.d.]. Private Data Collections: A High-Level Overview. https://www.hyperledger.org/blog/2018/10/23/private-data-collections-a-high-level-overview.

[4] Archita Agarwal, Maurice Herlihy, Seny Kamara, and Tarik Moataz. 2019. Encrypted databases for differential privacy. *Proceedings on Privacy Enhancing Technologies* 2019, 3 (2019), 170–190.

[5] Ishtiyaque Ahmad, Yuntian Yang, Divyakant Agrawal, Amr El Abbadi, and Trinabh Gupta. 2021. Addra: Metadata-private voice communication over fully untrusted infrastructure. In *15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21)*.

[6] Gustavo Alonso and Amr El Abbadi. 1995. Partitioned data objects in distributed databases. *Distributed and Parallel Databases* 3, 1 (1995), 5–35.

[7] Mohammad Javad Amiri, Divyakant Agrawal, and Amr El Abbadi. 2019. CAPER: a cross-application permissioned blockchain. *Proc. of the VLDB Endowment* 12, 11 (2019), 1385–1398.

[8] Mohammad Javad Amiri, Divyakant Agrawal, and Amr El Abbadi. 2019. ParBlockchain: Leveraging Transaction Parallelism in Permissioned Blockchain Systems. In *Int. Conf. on Distributed Computing Systems (ICDCS)*. IEEE, 1337–1347.

[9] Mohammad Javad Amiri, Divyakant Agrawal, and Amr El Abbadi. 2021. SharPer: Sharding Permissioned Blockchains Over Network Clusters. In *SIGMOD Int. Conf. on Management of Data*. ACM, 76–88.

[10] Mohammad Javad Amiri, Joris Duguépéroux, Tristan Allard, Divyakant Agrawal, and Amr El Abbadi. 2021. SEPAR: A Privacy-Preserving Blockchain-based System for Regulating Multi-Platform Crowdworking Environments. In *Proceedings of The Web Conference*. 1891–1903.

[11] Mohammad Javad Amiri, Ziliang Lai, Liana Patel, Boon Thau Loo, Eric Loo, and Wenchao Zhou. 2021. Saguaro: Efficient Processing of Transactions in Wide Area Networks using a Hierarchical Permissioned Blockchain. *arXiv preprint arXiv:2101.08819* (2021).

[12] Mohammad Javad Amiri, Boon Thau Loo, Divyakant Agrawal, and Amr El Abbadi. 2021. Qanaat: A Scalable Multi-Enterprise Permissioned Blockchain System with Confidentiality Guarantees. *arXiv preprint arXiv:2107.10836* (2021).

[13] N. Anciaux, Philippe Bonnet, Luc Bouganim, Benjamin Nguyen, P. Pucheral, I. Popa, and Guillaume Scerri. 2019. Personal Data Management Systems: The security and functionality standpoint. *Inf. Syst.* 80 (2019), 13–35.

[14] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, et al. 2018. Hyperledger Fabric: a distributed operating system for permissioned blockchains. In *European Conf. on Computer Systems (EuroSys)*. ACM, 30.

[15] Sebastian Angel, Hao Chen, Kim Laine, and Srinath Setty. 2018. PIR with compressed queries and amortized query processing. In *2018 IEEE symposium on security and privacy (SP)*. IEEE, 962–979.

[16] A. Arasu, Ken Eguro, Manas R. Joglekar, R. Kaushik, D. Kossmann, and Ravishankar Ramamurthy. 2015. Transaction processing on confidential data using cipherbase. *2015 IEEE 31st International Conference on Data Engineering* (2015), 435–446.

[17] Arvind Arasu and Raghav Kaushik. 2013. Oblivious query processing. *arXiv preprint arXiv:1312.4012* (2013).

[18] Sumeet Bajaj and R. Sion. 2014. TrustedDB: A Trusted Hardware-Based Database with Privacy and Data Confidentiality. *IEEE Transactions on Knowledge and Data Engineering* 26 (2014), 752–765.

[19] Daniel Barbard and Hector Garcia-Molina. 1992. The Demarcation Protocol: A technique for maintaining linear arithmetic constraints in distributed database systems. In *International Conference on Extending Database Technology*. Springer, 373–388.

[20] Johes Bater, Xi He, William Ehrich, Ashwin Machanavajjhala, and Jennie Rogers. 2018. Shrinkwrap: efficient sql query processing in differentially private data federations. *Proceedings of the VLDB Endowment* 12, 3 (2018), 307–320.

[21] Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. 2007. Deterministic and efficiently searchable encryption. In *Annual International Cryptology Conference*. Springer, 535–552.

[22] S. Benabbas, R. Gennaro, and Yevgeniy Vahlis. 2011. Verifiable Delegation of Computation over Large Datasets. *IACR Cryptol. ePrint Arch.* 2011 (2011), 132.

[23] Fabrice Benhamouda, Shai Halevi, and Tzipora Halevi. 2019. Supporting private data on hyperledger fabric with secure multiparty computation. *IBM Journal of Research and Development* 63, 2/3 (2019), 3–1.

[24] D. Boneh, Craig Gentry, S. Halevi, Frank Wang, and David J. Wu. 2013. Private Database Queries Using Somewhat Homomorphic Encryption. In *ACNS*.

[25] D. Boneh, A. Sahai, and Brent Waters. 2012. Functional encryption: a new vision for public-key cryptography. *Commun. ACM* 55 (2012), 56–64.

[26] Miguel Castro, Barbara Liskov, et al. 1999. Practical Byzantine fault tolerance. In *Symposium on Operating systems design and implementation (OSDI)*, Vol. 99. USENIX Association, 173–186.

[27] JP Morgan Chase. 2016. Quorum white paper.

[28] David Chaum, Ivan B Damgård, and Jeroen Van de Graaf. 1987. Multiparty computations ensuring privacy of each party's input and correctness of the result. In *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 87–119.

[29] Xiaofeng Chen, Hui Li, J. Li, Q. Wang, Xinyi Huang, W. Susilo, and Yang Xiang. 2020. Publicly Verifiable Databases with All Efficient Updating Operations. *IEEE Transactions on Knowledge and Data Engineering* (2020), 1–1.

[30] Yan Chen, Ashwin Machanavajjhala, Michael Hay, and Gerome Miklau. 2017. Pegasus: Data-adaptive differentially private stream processing. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 1375–1388.

[31] B. Chor, E. Kushilevitz, Oded Goldreich, and M. Sudan. 1998. Private information retrieval. *J. ACM* 45 (1998), 965–981.

[32] Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. 2011. Searchable symmetric encryption: improved definitions and efficient constructions. *Journal of Computer Security* 19, 5 (2011), 895–934.

[33] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. 2010. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*. 715–724.

[34] Dario Fiore, R. Gennaro, and Valerio Pastro. 2014. Efficiently Verifiable Computation on Encrypted Data. *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (2014).

[35] Dario Fiore, Anca Nitulescu, and D. Pointcheval. 2020. Boosting Verifiable Computation on Encrypted Data. *IACR Cryptol. ePrint Arch.* 2020 (2020), 132.

[36] Craig Gentry. 2009. Fully Homomorphic Encryption Using Ideal Lattices. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing (STOC '09)*. 169–178.

[37] Christian Gorenflo, Stephen Lee, Lukasz Golab, and Srinivasan Keshav. 2019. Fastfabric: Scaling hyperledger fabric to 20,000 transactions per second. In *Int. Conf. on Blockchain and Cryptocurrency (ICBC)*. IEEE, 455–463.

[38] Suyash Gupta, Sajjad Rahnama, Jelle Hellings, and Mohammad Sadoghi. 2020. ResilientDB: Global Scale Resilient Blockchain Fabric. *Proceedings of the VLDB Endowment* 13, 6 (2020), 868–883.

[39] Hakan Hacigümüş, Bala Iyer, Chen Li, and Sharad Mehrotra. 2002. Executing SQL over encrypted data in the database-service-provider model. In *SIGMOD Int. Conf. on Management of Data*. 216–227.

[40] Florian Hahn and Florian Kerschbaum. 2014. Searchable encryption with secure and efficient updates. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. 310–320.

[41] Bijit Hore, Sharad Mehrotra, Mustafa Canim, and Murat Kantarcioglu. 2012. Secure multidimensional range queries over outsourced data. *The VLDB Journal* 21, 3 (2012), 333–358.

[42] Zsolt István, Alessandro Sorniotti, and Marko Vukolić. 2018. StreamChain: Do Blockchains Need Blocks?. In *Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers (SERIAL)*. ACM, 1–6.

[43] Seny Kamara and Tarik Moataz. 2018. SQL on structurally-encrypted databases. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 149–180.

[44] Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. 2017. Accessing data while preserving privacy. *arXiv preprint arXiv:1706.01552* (2017).

[45] Jae Kwon. 2014. Tendermint: Consensus without mining. (2014).

[46] Leslie Lamport. 2001. Paxos made simple. *ACM Sigact News* 32, 4 (2001), 18–25.

[47] Yin Li, Dhrubajyoti Ghosh, Peeyush Gupta, Sharad Mehrotra, Nisha Panwar, and Shantanu Sharma. 2021. Prism: Private Verifiable Set Computation over Multi-Owner Outsourced Databases. In *SIGMOD Int. Conf. on Management of Data*. 1116–1128.

[48] Carlos Aguilar Melchor, Joris Barrier, Laurent Fousse, and Marc-Olivier Killijian. 2016. XPIR : Private Information Retrieval for Everyone. *Proceedings on Privacy Enhancing Technologies* 2016 (2016), 155 – 174.

[49] Carlos Aguilar Melchor, Joris Barrier, Laurent Fousse, and Marc-Olivier Killijian. 2016. XPIR: Private information retrieval for everyone. *Proceedings on Privacy Enhancing Technologies* 2016 (2016), 155–174.

[50] R. Merkle. 1987. A Digital Signature Based on a Conventional Encryption Function. In *CRYPTO*.

[51] Senthil Nathan, Chander Govindarajan, Adarsh Saraf, Manish Sethi, and Praveen Jayachandran. 2019. Blockchain meets database: design and implementation of a blockchain relational database. *Proceedings of the VLDB Endowment* 12, 11 (2019), 1539–1552.

[52] Rishabh Poddar, Tobias Boelter, and Raluca Ada Popa. 2016. Arx: A Strongly Encrypted Database System. *IACR Cryptol. ePrint Arch.* 2016 (2016), 591.

[53] Raluca Ada Popa, Catherine MS Redfield, Nickolai Zeldovich, and Hari Balakrishnan. 2012. CryptDB: processing queries on an encrypted database. *Commun. ACM* 55, 9 (2012), 103–111.

[54] Christian Priebe, Kapil Vaswani, and Manuel Costa. 2018. EnclaveDB: A Secure Database Using SGX. *2018 IEEE Symposium on Security and Privacy (SP)* (2018), 264–278.

[55] Amrita Roy Chowdhury, Chenghong Wang, Xi He, Ashwin Machanavajjhala, and Somesh Jha. 2020. Cryptϵ: Crypto-assisted differential privacy on untrusted servers. In *SIGMOD Int. Conf. on Management of Data*. 603–619.

[56] Pingcheng Ruan, Dumitrel Loghin, Quang-Trung Ta, Meihui Zhang, Gang Chen, and Beng Chin Ooi. 2020. A Transactional Perspective on Execute-order-validate Blockchains. In *SIGMOD Int. Conf. on Management of Data*. ACM, 543–557.

[57] Cetin Sahin, T. Allard, Reza Akbarinia, A. E. Abbadi, and Esther Pacitti. 2018. A Differentially Private Index for Range Query Processing in Clouds. *2018 IEEE 34th International Conference on Data Engineering (ICDE)* (2018), 857–868.

[58] Ankur Sharma, Felix Martin Schuhknecht, Divya Agrawal, and Jens Dittrich. 2019. Blurring the lines between blockchains and database systems: the case of hyperledger fabric. In *SIGMOD Int. Conf. on Management of Data*. ACM, 105–122.

[59] Emil Stefanov, Charalampos Papamanthou, and Elaine Shi. 2014. Practical Dynamic Searchable Encryption with Small Leakage.. In *NDSS*, Vol. 71. 72–75.

[60] Yuanyuan Sun, Sheng Wang, Huorong Li, and Feifei Li. 2021. Building Enclave-Native Storage Engines for Practical Encrypted Databases. *Proc. VLDB Endow.* 14 (2021), 1019–1032.

[61] Hoang Van Tran, Tristan Allard, Laurent d'Orazio, and Amr El Abbadi. 2021. FRESQUE: A Scalable Ingestion Framework for Secure Range Query Processing on Clouds. In *EDBT*.

[62] Chenghong Wang, Johes Bater, Kartik Nayak, and Ashwin Machanavajjhala. 2021. DP-Sync: Hiding Update Patterns in Secure Outsourced Databases with Differential Privacy. In *SIGMOD Int. Conf. on Management of Data*. ACM, 1892–1905.

[63] Xinying Yang, Yuan Zhang, Sheng Wang, Benquan Yu, Feifei Li, Yize Li, and Wenyuan Yan. 2020. LedgerDB: a centralized ledger database for universal audit and verification. *Proceedings of the VLDB Endowment* 13, 12 (2020), 3138–3151.

[64] Meihui Zhang, Z. Xie, Cong Yue, and Ziyue Zhong. 2020. Spitz: A Verifiable Database System. *Proc. VLDB Endow.* 13 (2020), 3449–3460.

[65] Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. 2017. vSQL: Verifying Arbitrary SQL Queries over Dynamic Outsourced Databases. *2017 IEEE Symposium on Security and Privacy (SP)* (2017), 863–880.

[66] Wenchao Zhou, Yifan Cai, Yanqing Peng, Sheng Wang, Ke Ma, and Feifei Li. 2021. VeriDB: An SGX-based Verifiable Database. In *SIGMOD Int. Conf. on Management of Data*. 2182–2194.