FedSmarteum: Secure Federated Matrix Factorization Using Smart Contracts for Multi-Cloud Supply Chain

Srini Bhagavan
IBM Corporation
University of Missouri-Kansas City
Kansas City, MO, USA
Email: srini.bhagavan@mail.umkc.edu

Mohamed Gharibi
IBM Corporation
San Jose, CA, USA
Email: gharibi@ibm.com

Praveen Rao
University of Missouri-Columbia
Columbia, MO, USA
Email: praveen.rao@missouri.edu

***Abstract*—With increased awareness comes unprecedented expectations. We live in a digital, cloud era wherein the underlying information architectures are *expected* to be elastic, secure, resilient, and handle petabyte scaling. The *expectation* of epic proportions from the next generation of the data frameworks is to not only do all of the above but also build it on a foundation of trust and explainability across multi-organization business networks. From cloud providers to automobile industries or even vaccine manufacturers, components are often sourced by a complex, not full digitized thread of disjoint suppliers. Building Machine Learning and AI-based order fulfillment and predictive models, remediating issues, is a challenge for multi-organization supply chain automation. We posit that Federated Learning in conjunction with blockchain and smart contracts are technologies primed to tackle data privacy and centralization challenges. In this paper, motivated by challenges in the industry, we propose a decentralized distributed system in conjunction with a recommendation system model (Matrix Factorization) that is trained using Federated Learning on an Ethereum blockchain network. We leverage smart contracts that allow decentralized serverless aggregation to update localized items vectors. Furthermore, we utilize Homomorphic Encryption (HE) to allow sharing the encrypted gradients over the network while maintaining their privacy. Based on our results, we argue that training a model over a serverless Blockchain network using smart contracts will provide the same accuracy as in a centralized model while maintaining our serverless model privacy and reducing the overhead communication to a central server. Finally, we assert such a system that provides transparency, audit-ready and deep insights into supply chain operations for enterprise cloud customers resulting in cost savings and higher Quality of Service (QoS).**

1. Introduction

Blockchain with its shared ledger represents a single system of cryptographic truth, tamper-resistant audit logs, and algorithmic trust, all of which provide the foundation that AI-based complex supply chains need. Blockchain has established itself as a disruptive technology enabling organizations to reinvent, cross borders and collaborate with their business network in previously unimaginable ways. Proof helps engender trust, and blockchains provide irrefutable cryptographic proof of transactions for secure audit trails and thereby, promote trust in the overall system [1]. Figure 1 illustrates the use of Merkle trees to encode blockchain data to be more secure as every parent node is labeled with the cryptographic hash of its child nodes. Smart contracts (autonomous contracts or chain code) are a core fabric component that enables automation across a multi-organization trusted network and its algorithms define the life cycle of one or more business objects.

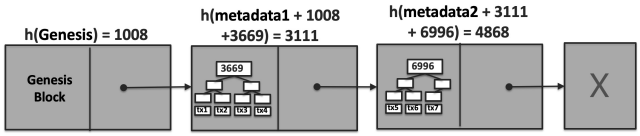

Figure 1: A Hash tree or Merkle tree combined with hash chain promotes efficient search within a block - O(log N)

Multi-organization enterprise customers looking to modernize their automation stack and gravitating toward a *consume anywhere* architecture with a hybrid cloud framework to address the dynamic nature of their businesses and in doing so, they also do want to not worry about accountability, privacy, scalability, and finality of transactions. Recent advances in enterprise blockchain have not only entrenched these enterprise attributes but also proved the potential applicability of blockchain is far greater than just the ubiquitous cryptocurrency. It encompasses a vast range of industries including building a very robust supply chain where it is important to understand data provenance, how goods are sourced, how ethical is the process, how easy is to log, track and prove exceptions and thereby shorten reconciliation cycles. It is imperative for such a supply chain to have the system be fault-tolerant. With blockchain fault-tolerant consensus algorithms, the network continues to operate even in the presence of malicious or careless participants. Just like any established network, they need to be governed member access permissions that are regulated based on business needs i.e., permissioned does not mean private.

1.1. Why Federated Learning?

Businesses continue to burn the midnight oil to protect and preserve security and privacy [2], [3]. However, the effectiveness of AI depends on access to all the data we generate. Motivated by growing private data concerns, as well as the explosion in computing capabilities that end-users possess, a new machine learning paradigm called Federated Learning is gaining popularity. Federated Learning performs Machine Learning by first training a model at each client-side privately and confidentially and subsequently collects and averages models from all participating clients to generate a more generalized model [6]. None of the clients share their confidential data which seems to solve multi-organization data privacy challenges [7]. Healthcare and user's edge devices are two of the main categories that use Federated Learning to train models. However, when deployed to solutions across multi-organizations current Federated Learning architecture has a few challenges such as excessive communications, scalability, and data leakage.

In this paper, we propose a decentralized blockchain network and smart contracts to allow transparent, audit-ready, immutable, and trust-based collaboration between multiple cloud providers and vendors as part of fulfilling a complex customer purchase order. Our target is to improve the enterprise customer's Quality of Service (QoS) attributes when deploying and maintaining a multi-cloud solution. We further intend to delight their experience by providing a fabric to share accurate supply chain insights from multiple parties by implementing a *decentralized* (instead of *centralized*) federated learning matrix factorization model using Ethereum blockchain. The key contributions of our work are as follows:

- A decentralized trusted architecture to overcome the centralized federated learning single server challenges such as the single point of failure and the excessive communications between a single server and the rest of the nodes.
- Smart contracts to orchestrate enterprise customer purchases of multi-cloud assets deployed on a permissioned blockchain network. The novelty of the solution also lies in using smart contracts as the control plane to integrate Federated Learning algorithm to recommend cloud provider resources based on (similar) customers usage patterns.
- Our system can be applied publicly to provide customers and third parties with insights, future predictions, and recommendations while training the model on the localized data without violating their privacy.

The rest of the paper is organized as follows: Section 2 discusses related work. Section 3 introduces underlying enterprise challenges while dealing with multiple cloud provider supply chains and parallels in other sectors including healthcare. Section 4 provides the approach of FedSmarteum. Section 5 reports experimental evaluation, time study, and a description of system implementation and the benchmark datasets used. Finally, we conclude in Section 6 and reference possible future work.

2. Related Work

Given the complexity of the proposed system, we have a two-part approach to the related work discussion. The first part focuses on the federated learning (matrix factorization) model used in our technique. The second part evaluates similar systems that use federated learning built on blockchain.

The authors of FedMF [9], Chai et al., introduced a secure matrix factorization framework using federated learning where the model learns using the user's gradients only that are sent to the server instead of the raw preference data. The authors proved that while the gradients seem secure enough, it still might leak some of the user's data [10], [11]. Therefore, they took the implementation of FedMF a step further by implementing the matrix factorization framework using homomorphic encryption [12]. To this end, the implementation of FedMF is a secure framework that can be used to learn a matrix factorization model. The architecture of FedMF still relies on the traditional federated learning approach that utilizes a single centralized server. In industry, when a huge number of parties are included, a single server will not be efficient to coordinate, collect, average all of the client's models. Therefore, a decentralized version of FedMF is required to allow multiple worker nodes to carry the global computational progress.

The authors of BAFFLE [13] created a decentralized aggregator-free blockchain-driven environment that leverages smart contracts to coordinate the federated learning settings and aggregate the user's models. BAFFLE enhances the aggregation process and boosts the computational progress by dividing the global parameters space into chunks with a score and a bid strategy. Chunking the parameters space is due to the limit of the pertaining size of the Ethereum Virtual Machine (EVM) which is 24 kB [14]. Furthermore, due to the expensive SC storage, the models need to be stored in a serialized format. Therefore, the authors use a partitioning algorithm that is used by all of the users to first chunk then serialize the parameters. Finally, the budget for each chunk allows users to decide on which chunks to contribute with, which saves time, capacity, and does not add unnecessary communication on the network. However, the machine learning model type that is used in BAFFLE allows aggregating the user's models in order to generate a more generalized global model. But that is not the case when it comes to recommendation systems models where each user data vector is different than the others since not all of the users have the same items and products. Therefore, it is not possible for matrix factorization techniques to aggregate the user's models to generate a global model. For matrix factorization, each user data vector needs to be updated individually based on that specific user's data. Hence, BAFFLE environment is not applicable for recommendation system models. Furthermore, in our implementation, we store the items data on the worker nodes and keep the user's data localized on the user's machines

while leveraging the benefits of smart contracts for other federated learning tasks.

Tzu-Yu et al. [15] proposed a trust-based system for Collaborative Filtering recommendation systems on the blockchain. The proposed system provides a secure and trust-based system supported by the use of smart contracts in the main blockchain protocol. Using blockchain the authors proposed a framework to collect large volumes of data and allow users to host and share a mutual recommender model that is being used as a public resource. Using incentive mechanisms, users are able to share the model parameters, new images, movies, titles, etc. that allow updating the existing models hosted by a specific user or a group of users. In such scenarios, all users are assumed to be trusted users who will not share malicious information or update the model with false predictions. In addition to the model parameters being shared publicly on the network, the gas fees for the smart contract storage and other operations are very expensive. In our implementation, all of the gradients are stored in IPFS (distributed storage). The hash from IPFS is then shared on the Ethereum network to reduce gas fees instead of sharing the actual data. On top of that, all operations and model updates are computed using the homomorphic encrypted items and gradients vectors.

Blockchain started to make in-roads into the medical fields to facilitate collaboration between patients, healthcare providers, and insurers via a secure, transparent, and immutable network. The authors of HealthMudra Rashmi et al. [16] presented a recommender system that prevents diabetes which is the most challenging disease in the healthcare sector [16]. HealthMudra is built using a blockchain network powered by machine learning algorithms and optimization protocols that mainly use filtering techniques to provide recommendations in order to prevent diabetes. Such implementations are helpful when patients share their health data with different healthcare providers to get helpful recommendations. In our work, we consider federated learning to keep the user data localized. Even when gradients are shared, they are shared after applying homomorphic encryption to perform all of the learning and model updating using the encrypted data and not raw data.

Several studies have discussed the area of federated learning using blockchain which lead to a new paradigm known as FLchain. This technique transfers the current Mobile Edge Computing (MEC) networks into decentralized, secure, and privacy-preserving systems. Nguyen et al. [17] proposed a FLchain network to train a shared model using mobile edge devices. FLchain allows training a shared model while keeping the user's data localized on their devices to benefit from privacy enhancement. A group of MEC servers is initialized with their associated user's devices. Each user carries out the training locally and commits its update to these servers through a transaction that is stored in a Merkle tree. Such approaches like FLchain and BAFFLE are applicable in the case of using specific types of machine learning techniques where averaging all user's models generates a better global model. However, in recommendation systems averaging all user's updates is not the right way to achieve better accuracy since each user's updates are specific to that user. In the recommendation systems area, users with similar behaviors tend to have similar updates, but that cannot be generalized among all of the users on the network.

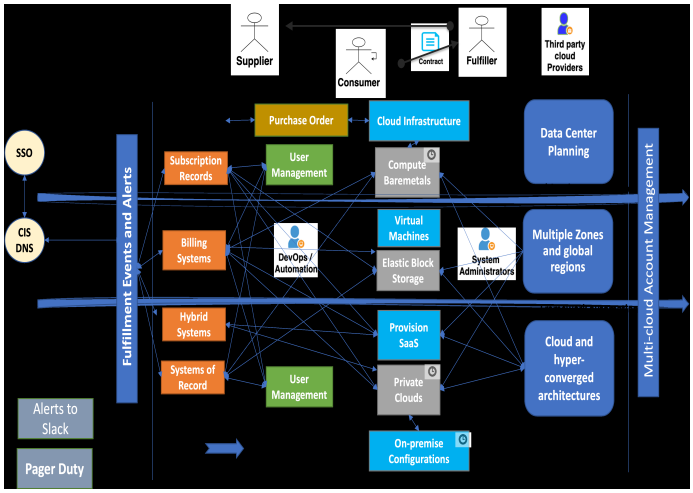

Figure 2: Enterprise cloud operations under the hood.

Finally, it is worth noting the idea of a supply chain using blockchain is nascent and full potential can be achieved with the digital transformation of existing, entrenched supply chain infrastructures. Bhagavan et al. [1] discussed utilizing permissioned blockchain to process multi-cloud customer orders. However, such networks do not have AI-based optimization techniques implemented using smart contracts to obtain data insights from the collected raw data across multiple cloud vendors and suppliers.

3. Problem Formulation

Recent surveys show 60% of consumers are willing to change in support of a traceable (provenance of materials) and sustainable supply chain and are willing to pay up to a 35% premium to support a circular economy (reuse of materials) [4]. Consumers and businesses are fatigued by massive data sprawls and are gravitating toward universal governance, transparent data discovery, and provenance to enable their mission-critical enterprise purchase decisions driven by insights from data available to them. In this paper, we discuss two major challenges:

- Cloud provider physical resources are often sourced by a complex, not full digitized thread of disjoint suppliers who behind the scenes are attempting to fulfill their parts of the order expeditiously. This often results in disjoint communication over email, Slack, fax, spreadsheets, and other software and is not conducive to scale beyond organizational boundaries. Figure 2 reveals under the hood complexity even with a single cloud provider supply chain. The task gets more challenging when multiple parties are included to fulfill the supply chain network.

- Using AI to obtain insights on private data is paramount to compete in the marketplace. To accomplish this task federated learning can be considered as a solution. However, current techniques rely on the traditional federated learning settings that utilize a single central server. In industries where multiple parties are involved and predictive models are limited by organization boundaries, figure 3, a single server will not be efficient to coordinate, collect, average all of the client's models. A decentralized version of federated learning is to be considered to allow multiple worker nodes to carry the global computational progress and facilitate fulfillers to make AI-based decisions.

We discuss some of the pertinent challenges and motivating use cases in more detail in the following sections.

3.1. Policing B2B Multi-organization cloud supply chains

An enterprise-level supply chain is characterized as a complex system orchestration between geo-dispersed suppliers, buyers, and fulfillers that could be spread across multiple organizations, and with people and systems in various roles performing life cycle activities. International Data Corporation [5] estimates 81% of enterprise organizations use a multi-cloud approach (often mixed with hybrid cloud), which empowers them to selectively extend their cloud footprint and deploy application workloads based on cloud provider strengths, cost, and disaster recovery needs. For cloud providers, even with self-service, typical enterprise cloud customer sales and procurement are addressed by the sales department drawing up purchase orders which are fulfilled behind the scenes by pockets of automation along the way. Furthermore, it is an arduous task to automate when multiple organizations are involved.

Recent trends in cloud computing encourage customers to keep compute close to their existing data which may be deployed on any cloud provider. Figure [2] captures a view of the entrenched scope of order orchestration. Terms of purchase are recorded by multi-organization sales using prevailing tools and practices, with some of them being archaic. Requested cloud providers and their data centers may have to replenish specific compute or storage types, set up cloud accounts, and propagate data to all billing and subscription systems to validate the order. As shown in Figure [3], any automation for fulfillment that may also use AI for predictions typically stops at the company's firewall making order fulfillment progress opaque to the enterprise customer. Manual order reconciliation is costly and time-consuming as every cascading fulfiller in a multi-cloud supply chain will have their own system of record which often has strict rules about how data can be shared across the business network. For example, the personal information and identity of the customer may be required to process an order automatically or billing reconciliation when an incorrect compute node configuration was deployed, but

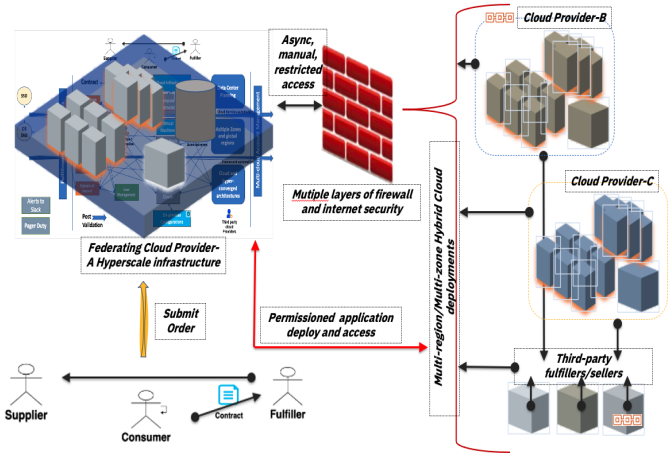

Figure 3: Information silos in multi-organization business flows

fulfiller business rules are written to prevent such data to be shared across organizational boundaries. It is commonplace for cloud providers or customers to change contract terms, leading to further delays. For example, Elastic File System storage with higher performance IOPS and lower latency may no longer be available in the desired region due to supplier issues. The customer is then presented with an alternate data center or asked if the use of elastic block storage instead will satisfy their needs.

3.2. Healthcare supply chain seams exposed

COVID-19 pandemic is a global challenge and demands no less than a global response. It has compressed innovation cycles across the health, manufacturing, and computing industry by at least a factor of ten. The opportunities to improve care management, patient outcome, and patient engagement are significant. However, recent health and bioinformatics exploration has exposed our vulnerability not just to the virus, but also the dependency we have on our infrastructure which in many cases has failed us particularly in the areas of trust, scalability, immutability, security, and privacy. Several recent events amplify existing gaps in our supply chain infrastructure. Even though vaccines were developed with unprecedented speed, there were unexpected delays in manufacturing and distribution. Key ingredients required for vaccines were compromised and the manufacturer was unaware until it was too late to correct. Government and hospitals lack shipment delivery of vaccines to help normalize the supply chain based on population density and demographics. In many instances, hackers were able to target groups and gain control of distribution systems for malicious use. Contact tracing, an important epidemiological tool, has proven critical to tracking and containing the spread of the current pandemic, yet its implementation has been fraught with logistical and privacy concerns including disturbing the balance between safety and liberty.

Personal patient data is protected information and AI and machine learning algorithms have to work around that

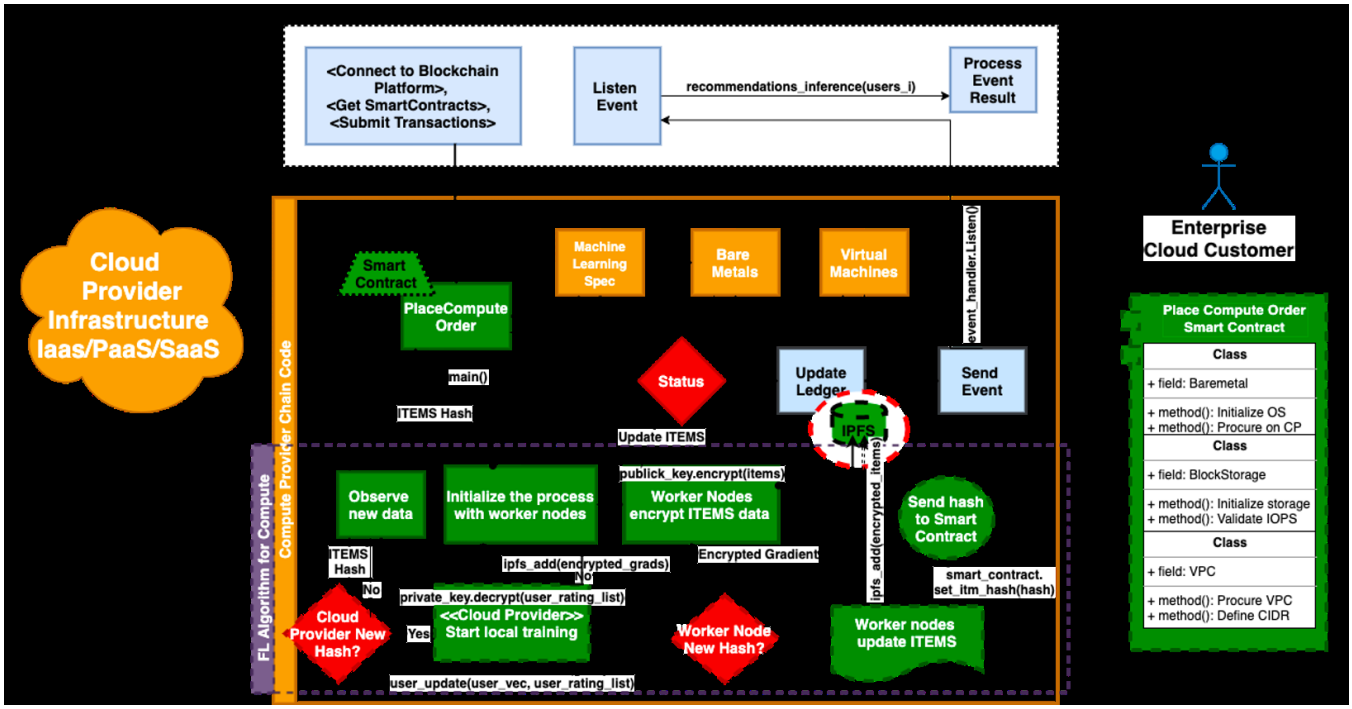

Figure 4: Smart contract orchestrated federated learning matrix factorization for multi-cloud enterprise customer order.

to make deep inroads for automation and analytics. As a team of German scientists noted in a 2019 paper in Nature Partner Journals of Digital Medicine: *Hidden in isolated databases, incompatible systems, and proprietary software, the data are difficult to exchange, analyze, and interpret. This slows down medical progress, as technologies that rely on these data — artificial intelligence, big data, or mobile applications — cannot be used to their full potential* [8]. It is almost impossible to reach healthcare data due to the data sensitivity and the number of regulations and protocols that need to be done in order to run an experiment or train a model. Furthermore, the provenance and authenticity of training data are critical in AI BOTS that perform the role of a physician's assistant. Recommendations are only one half of the solution, while the other half should focus on source data for the machine learning algorithms, why should we trust it, and how much impact did it have on my treatment outcome. Food manufacturers are thinking about the next step to save lives where they can trackback foodborne illnesses to bad ingredients. All of these are real, current problems in the healthcare industry that pass the *Fit for Blockchain Test*.

Hence, we conclude that blockchain is the fabric or tool of choice for multi-organization automation as it embodies an organizational and algorithmic trust based on advanced cryptography in a shared ledger to help form a single system of truth across business networks. Smart (autonomous) contracts are triggered when proper conditions are met without human intervention, and consensus amongst all impacted parties is sought, met, and reconciled without involving intermediaries. Validated transactions are recorded in a shared ledger and are immutable forming a tamper-resistant audit log which not only helps with regulatory compliance but also when properly used can non-repudiate and pinpoint exactly when and how deviation occurred and radically shorten the reconciliation process and improve customer QoS. Bitcoin and Ethereum are two examples of blockchain decentralized networks that allow people to exchange cryptocurrency without violating their privacy. They demonstrate how peer-to-peer (P2P) networks provide security and transparency and allow individuals to be in charge of their own data while performing computational operations privately. With the help of blockchain and federated learning, a collaboration between several parties is possible even when there is no trust between these parties. In addition, we have an obligation to show the lineage of where training data came from. Trust and explainability are paramount as we move into the world where AI bots are used liberally. Blockchain can help with many of these aspects given that many of these traits are organic to the blockchain platform.

4. FedSmarteum

We implemented our work using an Ethereum blockchain in order to provide customers with transparency while keeping the data private. Ethereum leverages the use of smart contracts that coordinates the federated learning process between all the included nodes and cloud parties. In this paper, we train a Matrix Factorization model in a federated way where all the customers' data remain localized on data owner's devices. The items data will be placed

on one of the worker nodes (could be one of the nodes that are included in training the model). Sending and receiving the training data, weights, and gradients on the Ethereum network is too expensive and requires a lot of gas fees, other than the network transfer size limitations. Therefore, we use a distributed database (IPFS) [18] that provides hash-based storage. When storing an object in IPFS, the distributed storage will provide a hash to be used when reading the object from the storage. Consequently, instead of sharing a model, dataset, or gradients set on the blockchain, we store these objects on IPFS and then send that hash to the smart contract which is going to share it with the participating nodes. In this way, all of the nodes will be able to share data on the blockchain using IPFS without incurring expensive fees. The remaining question is, should all the nodes trust IPFS or the worker nodes in order to share their weights or gradients? The answer is no. In our work, we have enabled homomorphic encryption where each node encrypts its data before sharing it on IPFS or on the chain. Using HE, nodes do not need to trust IPFS or any of the worker nodes. HE allows operations on the encrypted data such as addition, multiplication, and subtraction. In our scenario, we are using the encrypted customer's data to update the encrypted items data. Therefore, all the participants keep their data private and share their encrypted gradients only. The smart contract coordinates the federated learning process between the included parties. The purpose of the blockchain network is to keep all the parties included and updated during the process. Blockchain provides transparency between the customers and the cloud providers. All customers will be up to date with their procurement and services while the cloud providers will be able to get insights using the customer's data without violating their privacy.

4.1. FedSmarteum Methodology

As mentioned in section I, federated learning is an approach to allow users to train a shared model on their joint data without exposing users' local data. Federated learning comes under the umbrella of privacy-preserving techniques and can be categorized based on the distribution characteristics of the data [19]. Horizontal federated learning is one of the categories of federated learning where users' data share the same feature space while different users have different samples. Therefore, this matrix factorization model can be considered as an example of horizontal federated learning since the rating data shared with each user has the same feature space, but different users have different samples. In our implementation, we assume all of the users in our scenario are cloud providers. Our technique secures the users further from the central server since this server might be a trusted, but curious server.

4.1.1. Stochastic Gradient Descent for user-level matrix factorization. In this section, we introduce the optimization method that is used in the matrix factorization model [9], [20] which is stochastic gradient descent. We design a *serverless decentralized network* to update the model locally

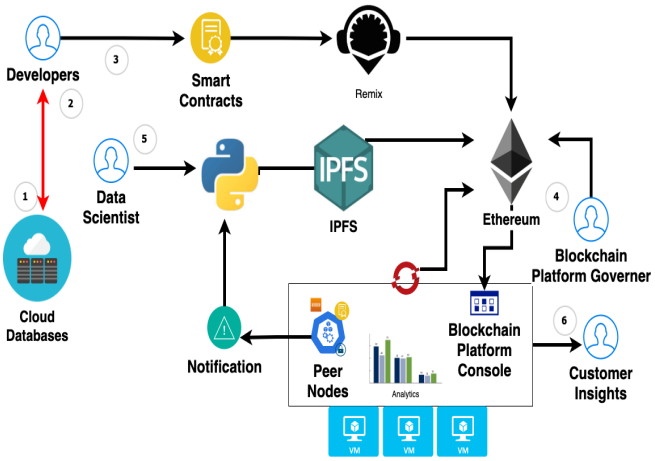

Figure 5: Deployment flow depiction of FedSmarteum.

using worker nodes in a decentralized way instead of relying on a central server.

Assuming we have a m number of items and n number of users where each user rated a number of items (a subset of m). Given $[n] := \{1, 2, ..., n\}$ which is the set of users and $[m] := \{1, 2, ..., m\}$ is the set of items, we can denote $\mathcal{M} \in [n]X[m]$ for the user-item rating pairs and M is the total number of ratings $M = |\mathcal{M}|$. We denote the user i that rated item j by r_{ij}. Assuming the value r_{ij} is given, then the recommendation system is expected to predict all if the items for all of the users by fitting a binary model on the existing ratings. That is computed using user matrix $U \in R^{n \times d}$ and item matrix $V \in R^{m \times d}$ and the output matrix is then used to predict the user i's rating on items j which can be described as $\langle u_i, v_j \rangle$. The authors of FedMF [9] shows how to compute U and V by solving the regularized least squares minimization as in equation 1.

$$\min_{U,V} \frac{1}{M}(r_{i,j} - \langle u_i, v_j \rangle)^2 + \lambda ||U||_2^2 + \mu ||V||_2^2 \quad (1)$$

λ and μ are small values that were added to rescale the penalizer. U and V will be updated using the Stochastic Gradient Descent as in Equations 2 and 3.

$$u_i^t = u_i^{t-1} - \gamma \nabla_{u_i} F(U^{t-1}, V^{t-1}) \quad (2)$$

$$v_i^t = v_i^{t-1} - \gamma \nabla_{v_i} F(U^{t-1}, V^{t-1}) \quad (3)$$

where

$$\nabla_{u_i} F(U, V) = -2 \sum_{j:(i,j)} v_j (r_{i,j} - \langle u_i, v_j \rangle) + 2\lambda u_i \quad (4)$$

$$\nabla_{v_i} F(U, V) = -2 \sum_{i:(i,j)} u_i (r_{i,j} - \langle u_i, v_j \rangle) + 2\lambda v_j \quad (5)$$

Then, this update takes place iteratively until the number of rounds is met.

4.1.2. The Decentralized Matrix Factorization. In our implementation, all users keep their rating data localized without sharing it with anyone. Then, the model is trained on the user's joint data. In order to achieve this goal, we leverage the use of the decentralized matrix factorization approach. This approach decomposes the updating of algorithms into two parts where the first part is performed on the user's device locally and the second part is performed and computed using the worker nodes in a decentralized way. Equation number 2 is performed and computed on the user i's device whereas equation 3 is performed and computed in a decentralized way using the worker nodes. The decentralization part is because of two main reasons: (i) to keep the user's rating data localized, and (ii) to prevent a trusted but curious server from recovering insights from the model.

Algorithm 1 Decentralized User-level matrix factorization

Init: Worker nodes initialize item profile matrix V
Init: User initializes user profile matrix U
Output: Converged U and V
    IPFS and worker nodes keeps latest item-profile for all users
    User local update:
        Smart Contract shares the hash with users to obtain V, perform local update:
            $u_i^t = u_i^{t-1}$ - γ ∇_{u_i} $F(U^{t-1}, V^{t-1})$
            $Gradient_i = \gamma$ ∇_{v_i} $F(U^{t-1}, V^{t-1})$
        Worker nodes update:
            Smart Contract shares IPFS $Gradient_i$ block with worker nodes for user-i
            Perform update: $v_i^t = v_i^{t-1}$ - $Gradient_i$

The general user-level matrix factorization method allows users to keep their data localized on their devices while they share the generated gradients as plain text with the server [20]. The authors of FedMF [9] proved that while the users keep their data localized, the server is still able to decode users' ratings through the gradients. Therefore, the authors have implemented the recommendation system using HE in order to secure the gradients. The server is still able to perform the same updates using the encrypted gradients while maintaining the model accuracy. The encrypted gradients will secure the users from the curious server and other attacks. However, the server will have an overhead updating the encrypted items data using the encrypted gradients. We replaced the server with worker nodes in order to carry out the items update in a decentralized fashion.

Our focus was on the efficiency of a decentralized framework while maintaining the model accuracy. We observed our generated model has similar predictions when testing the recommendation system on the MovieLens [21] rating dataset. However, since our algorithm was implemented in a decentralized way, there is a difference in the execution time as in our proposed approach the worker nodes are carrying out the items updates instead of the central server. Algorithm 1 shows the updating procedures.

4.2. Passing Gradients Over Blockchain

Most of the Blockchains have an upper size limit on the transactions. Ethereum Virtual Machine (EVM) has the limit of 24 kB [14]. Any random smart contract that contains many functions, too much code, with multiple events will hit the size limit instantly. For example, ERC1400 Security Token Standard requires 27 functions and 13 events [14]. With additional application functions and specific code to implement these standards, the limit will easily exceed 24 kB. Therefore, storing a large volume of amount of data on the smart contract is not feasible.

On the other hand, gas fees paid for transactions is expensive. As of August 25, 2021, the gas fee for a single transaction is 0.0013 Ether/transaction [22], with the Ether price today is $3,247.73, each Ethereum transaction costs almost $4. For example, the average size of a photo that was taken using an iPhone-6 is 2-3MB. Hence, depending on the Ether price, buying a car might be cheaper than adding one photo on Ethereum blockchain [23]. Ethereum network can be used due to its security, immutability, and transparency. However, for distributed storage purposes, we have used Interplanetary File System (IPFS) [18]. Using Ethereum and IPFS creates a simple, yet powerful, system of immutable content. This way, all data, and models can be stored on IPFS, while transactions and communications can take place using the Ethereum blockchain. Using IPFS, we are able to timestamp much larger data to be used over the blockchain than the pure blockchain networks. When users add data to IPFS, the protocol returns a hash for the data. This hash is cryptographically guaranteed to be unique to the content so no two sections of data will have the same hash. When the same section of data is added again to IPFS for the second time, the same unique hash will be returned. To retrieve data from IPFS, we use the reverse way. The same cryptographic hash returned from the storage process is then returned for the IPFS in order to retrieve the original data that was stored. By using the IPFS mechanism, we guarantee that our data has not been tampered with. Furthermore, all of the data stored on IPFS during our implementation is homomorphic encrypted. Therefore, the data cannot be read or tampered with by others. The receiver node can then download the encrypted data from IPFS and use the gradients for updating the items data. The smart contract is responsible for this process and directing the hash values between the user's local devices, worker nodes, and training nodes.

After the model is trained, the items data can be used with customer data to provide specific recommendations for that customer. The updated items data will be available in IPFS and worker nodes. The participating cloud providers and other privileged entities in the supply chain can use this data with new customers data to generate predictions and recommendations. The reason for leaving the encrypted items data on worker nodes is due to the fact that training is perpetual and iterative. As additional customers and data are generated, so are the training iterations. The accuracy of the model has a strict relationship with time and the number of users. All participating cloud providers have aqual access

to the model.

Figure 5 represents the deployment architecture for FedSmarteum. We have multiple stakeholders of the system including developers who are responsible for maintaining smart contracts, the data scientists who build and maintain ML algorithms, the blockchain governor who is responsible for setting up and maintaining the network connection and protocols, and finally, the customer who relies on the system for insights to help make business decisions on cloud resource utilization across a multi-organization platform deployment. For example, a customer may be recommended by the model that a specific machine type on data center_X has taken longer to deploy due to unavailability of the desired chipset, and other customers have either chosen to deploy a different machine type of chipset or pick an entirely different region previously. Having this recommendation enables customers to avoid unnecessary delays and cost. Also note that in our system, the customer is a training data provider for data scientists to train the ML models in a federated learning approach.

As alluded to earlier, smart contracts are developed to orchestrate the supply chain flow and federated learning technique and they constitute the backbone of our system. We deploy these smart contracts using the Remix platform on the Ethereum network. Machine Learning engineers and data scientists can then build their models and start the training process. The worker nodes connected to IPFS will then pull the encrypted items data from IPFS, decrypt it, generate gradients, and update the customer's data, and finally encrypt the gradients and push them back to IPFS. IPFS contains the latest items data that can be used to get insights by using the Analytics platform supplied by cloud providers. For example, we can deploy on an IBM Red Hat OpenShift cluster in a containerized environment which gives us ready access to analytic tools. As a cloud provider, we can rely on these insights to provide recommendations and predictions for future customers based on the previous customer's behavior.

5. Evaluation and Implementation Details

We deployed an Ethereum blockchain connected to multiple participating nodes that represent cloud providers and other stakeholders in the multi-organization supply chain. We conducted all of the smart contract executed federated learning evaluation using a well-known MovieLens dataset [24] and our private cloud dataset. We ran the evaluations using a different number of users, items, and rounds. Our implementation relies on decentralized architecture and smart contracts that coordinate the training process between cloud providers and worker nodes. Owing to this, we observed a few extra seconds of delay for each round of execution. This extra time is attributed to the time it takes for the smart contract to initialize the process with the worker nodes and the rest of the additional time is consumed during the process of pushing and pulling the encrypted data from IPFS. It is our intent to demonstrate that for industry applications, we can replace the central server with

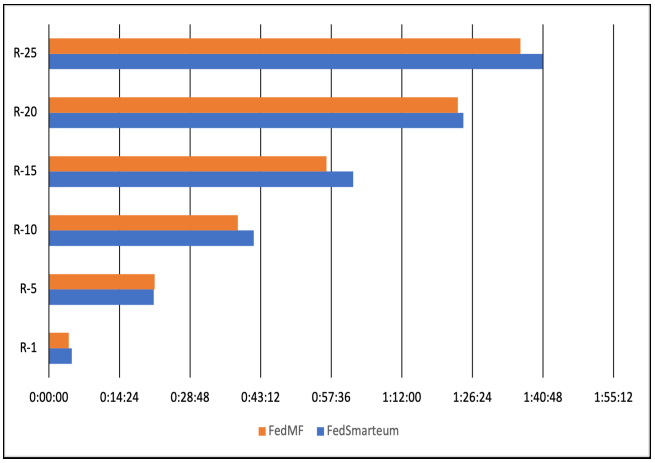

Figure 6: The execution time for FedSmarteum VS. FedMF using our cloud dataset. The experiment includes 20 customers with 40 items.

decentralized distributed worker nodes using blockchain and smart contracts while maintaining the same model accuracy. Table 1 shows the observed difference in time between FedSmarteum and FedMF. Figure 6 compares the execution time between FedSmarteum and FedMF. In this experiment, we ran both models on the cloud dataset including 20 customers with 40 items (multi-cloud resources and services). As can be seen in figure 6, the difference in time between the models is negligible which validates the decentralized approach applicability of FedSmarteum.

TABLE 1: Execution time for FedSmarteum and FedMF.

		FedSmarteum	FedMF
3 Users - 10 Items	R-1	0:00:42	0:00:40
	R-5	0:02:41	0:02:35
	R-10	0:05:34	0:05:08
	R-15	0:08:33	0:07:50
	R-20	0:10:54	0:09:27
	R-25	0:14:31	0:12:59
5 Users - 20 Items	R-1	0:01:55	0:01:48
	R-5	0:08:50	0:07:51
	R-10	0:16:39	0:15:31
	R-15	0:24:41	0:23:41
	R-20	0:31:41	0:28:31
	R-25	0:44:31	0:37:42
10 Users - 40 Items	R-1	0:06:25	0:06:09
	R-5	0:29:30	0:27:52
	R-10	1:02:54	0:57:02
	R-15	1:29:30	1:20:59
	R-20	1:56:15	1:50:39
	R-25	2:26:37	2:19:47

The table shows execution time for FedSmarteum and FedMF. "R" represents the number of rounds. The time is expressed in Hours:Minutes:Seconds.

5.1. Benchmark Description

We conducted our experiments using two datasets. The first one is the MovieLens datasets [21] is a well-known benchmark that has been collected and made available by

the GroupLens Research [25]. The dataset contains 100,000 movie ratings and 3,600 tag applications applied to 9724 movies by 610 users. The dataset is available at the following link: https://grouplens.org/datasets/movielens/.

The second dataset is our private cloud dataset. The premise for this dataset was to represent cloud resources (compute, storage, network, etc.,) procured by different customers across multiple cloud providers in order to deploy their application workloads. The cloud dataset we used includes data for 50 different resources provisioned across multiple cloud providers along with the count for each resource applied to 100 customers.

5.2. Blockchain Implementation and Encryption

To implement the homomorphic encryption, we used the Python programming language. Using Paillier encryption [26], we kept the length of the public key 1024. We used Truffle Suite [27] in order to deploy a synthetic Ethereum blockchain for development and testing purposes. In particular, we used Ganache version 2.5.4 [28] which allows interacting with smart contracts using Python. The smart contracts were deployed using Remix IDE [29]. All smart contracts were written in Solidity programming language using the compiler version ^0.4.21.

IPFS version 0.15.0 was used to allow the distributed encrypted items data to be shared. IPFS powers the Distributed Web using a peer-to-peer hypermedia protocol designed to preserve and grow humanity's knowledge by making the web upgradeable, resilient, and more open [18].

6. Conclusion and Future Work

Computing on the cloud, social, and edge devices generate massive volumes of data every second. AI has cemented its place as the tool of choice for analytic workloads and also helps build predictive models that are able to assist and automate workflows across multi-organizations business networks. Current implementations lean toward federated learning for training AI models on private data. However, federated learning relies on a central server to perform the global computation progress which can lead to several issues related to handling data heterogeneity between parties, coordination of learning process, bias due to lack of verification datasets, single point of failure, and above all communication overhead between parties involved. We proposed a decentralized federated learning matrix factorization technique implemented using a blockchain. Our implementation leverages the use of smart contracts to orchestrate and automate the asset procurement process. This immutable, decentralized architecture reduces the overhead communication between the server and all the nodes while retaining data ownership and privacy with participants. Additionally, we eliminated the single point of failure and proved that decentralized recommendation systems can be implemented using blockchain while maintaining the model accuracy with marginal overhead in computational time. We hope this paper will inspire cloud providers and their supplier community to consider a decentralized framework as they engineer their next generation of supply chain automation.

Democratization via a decentralized architecture provides equal rights for clients to share unbiased data. Our future work will focus on reducing the time required to complete a single iteration and worker node updates using our decentralized federated learning technique and more accurate prediction models for resource procurement. We hope to explore similar techniques to homomorphic encryption that requires less computing and updating time. Reducing the communication between worker nodes, IPFS, and smart contracts will reduce the execution time in each iteration. We are also investigating model governance and how cloud providers can have access to the model based on the provided training data and accuracy before and after including each cloud provider.

7. Acknowledgement

This work was supported in part by the National Science Foundation (NSF) under grant No. 1747751.

References

[1] S. Bhagavan, P. Rao, and T. Ngo, "C3HSB: A transparent supply chain for multi-cloud and hybrid cloud assets powered by blockchain," 2021 IEEE 37th International Conference on Data Engineering Workshops (ICDEW). IEEE, pp. 100-103, April 2021.

[2] U. Jayasinghe, G. M. Lee, A. MacDermott, and W. S. Rhee, "trustchain: A privacy preserving blockchain with edge computing," Wireless Communications and Mobile Computing 2019, July 2019.

[3] "Privacy-preserving computation on blockchains could prevent breaches," a blog by Felix Xu, https://cointelegraph.com/news/privacy-preserving-computation-on-blockchains-could-prevent-breaches, July 2021.

[4] What is Multi-Cloud? Everything You Need to Know https://www.factioninc.com/blog/what-is-multi-cloud/.

[5] What We Learned From IDC's 2019 Multicloud Management Survey. https://www.cloudhealthtech.com/blog/multicloud-management-survey

[6] B. H. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," Artificial Intelligence and Statistics. PMLR, pp. 1273-1282, April 2017.

[7] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," IEEE Transactions on Industrial Informatics 16, no. 10, pp. 6532-6542, October 2019.

[8] M. Lehne, J. Sass, A. Essenwanger, J. Schepers, and S. Thun, "Why digital medicine depends on interoperability," In Nature Partner Journals of Digital Medicine, NPJ digital medicine 2.1 (2019), pp. 1-5, August 2019.

[9] D. Chai, L. Wang, K. Chen, and Q. Yang. "Secure federated matrix factorization," IEEE Intelligent Systems, IEEE 2020.

[10] D. Pasquini, G. Ateniese, and M. Bernaschi, "Unleashing the Tiger: Inference Attacks on Split Learning," arXiv preprint arXiv:2012.02670 (2020), December 2020.

[11] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," In Federated learning. Springer, pp. 17-31, 2020.

[12] G. Craig, "Fully homomorphic encryption using ideal lattices," Proceedings of the forty-first Annual ACM Symposium on Theory of Computing. pp. 169-178, May 2009.

[13] R. Paritosh, and K. Nakayama. “Baffle: Blockchain based aggregator free federated learning,” 2020 IEEE International Conference on Blockchain (Blockchain). IEEE, pp. 72-81, November 2020.

[14] “Ethereum’s Maximum Contract Size Limit is Solved with the Diamond Standard,” A blog by Nick Mudge, https://dev.to/mudgen/ethereum-s-maximum-contract-size-limit-is-solved-with-the-diamond-standard-2189 July 2020.

[15] Y. Tzu-Yu, and R. Kashef. “Trust-Based collaborative filtering recommendation systems on the blockchain,” Advances in Internet of Things 10.4, pp. 37-56, 2020.

[16] B. Rashmi, and D. Datta. “Development of a Recommender System HealthMudra Using Blockchain for Prevention of Diabetes,“ Recommender System with Machine Learning and Artificial Intelligence: Practical Tools and Applications in Medical, Agricultural and Other Industries, pp. 313-327, June 2020.

[17] D. C. Nguyen, M. Ding, Q. V. Pham, P. N. Pathirana, L. B. Le, A. Seneviratne, et al.“Federated learning meets blockchain in edge computing: Opportunities and challenges,” IEEE Internet of Things Journal, April 2021.

[18] “Interplanetary File System (IPFS),” IPFS powers the Distributed Web, https://ipfs.io/.

[19] Q. Yang, Y. Liu, T. Chen, Y. Tong, “Federated machine learning: Concept and applications,” ACM Transactions on Intelligent Systems and Technology (TIST) 10.2 (2019), pp. 1-19, January 2019.

[20] K. Yehuda, R. Bell, and C. Volinsky. “Matrix factorization techniques for recommender systems,” Computer 42.8 (2009), pp. 30-37, 2009.

[21] “MovieLens,” The MovieLens Dataset, https://movielens.org/.

[22] “Ethereum Average Transaction Fee,” Ycharts. https://ycharts.com/indicators/ethereum_average_transaction_fee_eth, 2021.

[23] “The Decentralized Power of Ethereum and IPFS,” A blog by Matt Ober, https://medium.com/pinata/ethereum-and-ipfs-e816e12a3c59 November 2018.

[24] F. M. Harper, and J. A. Konstan, “The movielens datasets: History and context,” ACM Transactions on Interactive Intelligent Systems (TIIS) 5.4 (2015), pp. 1-19, December 2015.

[25] “GroupLens,” MovieLens datasets collected by GroupLens, https://grouplens.org/datasets/movielens/.

[26] “Paillier,” A Python 3 library implementing the Paillier Homomorphic Encryption, https://github.com/data61/python-paillier.

[27] “Truffle Suite,” sweet tools for smart contract, https://www.trufflesuite.com/.

[28] “Ganache,” Ganache one click blockchain, https://www.trufflesuite.com/ganache/.

[29] “Remix,” Remix Ethereum IDE, https://remix.ethereum.org/.