# Using blockchain technologies to improve security in Federated Learning Systems

## Novel defense scheme against model poisoning attack

Andrew Ronald Short
Dept. of Industrial Design and Production Engineering
University of West Attica
Athens, Greece
ashort@uniwa.gr

Helen C. Leligou
Dept. of Industrial Design and Production Engineering
University of West Attica
Athens, Greece
e.leligkou@uniwa.gr

Michael Papoutsidakis
Dept. of Industrial Design and Production Engineering
University of West Attica
Athens, Greece
mipapou@uniwa.gr

Efstathios Theocharis
Dept. of Industrial Design and Production Engineering
University of West Attica
Athens, Greece
stheo@uniwa.gr

***Abstract*—The potential of Federated Learning (FL) deployment increases rapidly as the number of connected devices increases, the value of artificial intelligence is recognized and networking technologies and edge computing evolves. However, as in any distributed system, a set of security issues arise in FL systems. In this paper, we discuss the use of blockchain technology to address diverse security aspects of FL systems and focus on the model poisoning attack for which we propose a novel Blockchain-based defense scheme. An assessment using data from the MNIST database has shown that the proposed approach, which has been designed to be implemented on blockchain technology, offers significant protection against adversaries attempting model poisoning attacks. The approach adopts a novel algorithm for evaluating the model updates, by verifying each model update separately against a verification dataset, without requiring information about the training dataset size, which is often unavailable or easily falsified.**

***Keywords: Blockchain, Federated Learning, Security attacks***

## I. INTRODUCTION

Federated Learning (FL) [1], [2] is a relatively new field of study, whereby collaborating parties can jointly improve a model which is maintained by the coordinator (usually a central server). In this setup, the training is performed individually using locally available data, and the participating entities transmit their model updates to the coordinator. The entities (either end-devices such as mobile phones, or businesses contributing with their data) usually share the same goal, i.e. to make use of the trained model or can otherwise be motivated by financial rewards.

Machine Learning (ML) aims to build a mathematical model by inference, using training data. ML can either be supervised or unsupervised, depending on whether the training data set is labeled or not [3]. In a typical centralized Machine Learning setup, the training data needs to be collected at a central location, which can pose security and privacy risks. Other concerns include the unavailability of training data of good quality, either because of data protection regulations or because of the unwillingness of users to share their private data due to privacy concerns.

Today, large amounts of data are being generated at the edge. Federated Learning overcomes the need to transfer user data to central servers, at the same time safeguarding user privacy. Practical implementations of FL already exist, such as the FL powered application Gboard [4], a keyboard from Google for Android devices, which is used to predict words based on previous input and is improved by the small contributions of each device. With the advent of edge computing, it is foreseen that more machine learning applications will run on edge devices. This is evident by the use of Artificial Intelligence (AI)-specific CPUs in the latest mobile devices from manufacturers, such as Apple and Samsung.

FL improves on privacy, since participating entities contribute with model updates rather than raw data, however as training is performed at the edge, a new attack surface is created. An adversary may intentionally send false model updates in order to affect its performance, a method known as model poisoning [5]. Since high quality datasets are an important factor in training, it may be beneficial to reward users whose model updates improve the overall model performance.

However, a set of security attacks have already been witnessed and countermeasures either rely on adding intelligence or resort to Blockchain technology. Blockchain belongs to the family of Distributed Ledger Technologies (DLT) and as such allows for the operation of an immutable database which can be distributed (and replicated) among multiple nodes. These types of databases offer improvements on security and trust, and offer a high level of transparency. For these reasons, they can be exploited to defend against FL-targeting attacks.

In this article, we present this vivid research landscape and we expand on the model poisoning attack. We propose an algorithm which is suitable to be run in a blockchain network, and provide initial evaluation results. Although blockchain technologies have been employed in FL, to the best of our knowledge, they have not yet been exploited to defend against this attack which we consider of high importance in various application scenarios including also industry 4.0. More specifically, we first describe the main components of any federated learning system and explore the relevant security issues (in section II). In section III, we explore the benefits that the adoption of blockchain technology can bring to FL and in section IV, we review already proposed blockchain based FL solutions. In section V we present our approach and algorithm and in section VI, we present the first results. Section VII concludes the paper.

978-1-7281-7303-0/20/$31.00 ©2020 IEEE
DOI 10.1109/COMPSAC48688.2020.00-96

## II. Federated Learning and Assosiated Security Aspects

In contrast to the centralized nature of traditional ML where data is collected in a large dataset (typically in the cloud) and the model is trained centrally, FL proposes that the ML model can reside on client devices and be trained locally. An aggregation server is required in order to transmit the global model and receive the model updates. Although the individual models being trained on each end-device may have an incomplete picture, considering that the insights of all participating parties are accumulated and that this process is repeated on several rounds, the end model tends to be of high quality. Figure 1 shows the steps involved in FL. The workflow starts by transmitting the initial model (initialized with either random or proxy data) to end devices (blue arrow). The end devices can now perform local training using locally available data (yellow arrow). During the training process, gradients are computed e.g. through a variant of Stochastic Gradient Descent (SGD)[6] on a random portion of the training dataset. In the end, these updates are received by the aggregation server (orange arrow), which in turn computes the updated model, typically by aggregating the model updates. This process is repeated until predefined criteria are met.

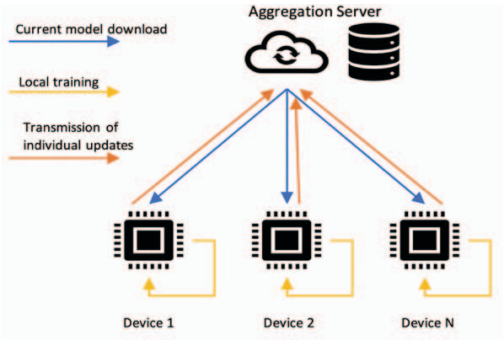


Figure 1 Federated Learning Process

A major benefit of FL is that the end device is able to use the model even when offline. Although the learning process will be affected, the latest version of the global model will still be available. In an FL setup, the data is kept at end devices and never shared with the server. This alone offers a huge improvement in privacy and helps meet requirements of Regulations such as GDPR [7]. However, since even the model update parameters can reveal sensitive information [8], [9], ongoing research is focused on mitigating privacy concerns. It is also worth mentioning that anonymization alone is not an adequate measure for privacy. The process of anonymization is usually carried out by the server and this trust relationship is not always present. Furthermore, researchers have been able to de-anonymize datasets by using external knowledge [10], [11].

Two of the most widely adopted approaches regarding privacy are Secure Multiparty Computation (MPC) [12] and Differential Privacy [13],[14],[15]. MPC enables participating parties to jointly compute an output of a shared function based on their inputs in a way that no information on the inputs is revealed to each other. Several MPC protocols and their effectiveness under different threat models are analyzed in a study by D. Evans 2018 [16]. Differential Privacy aims to conceal personal information in client model update contributions by adding a small amount of noise. In this approach, there is a trade-off between privacy and model accuracy.

With respect to the security aspects of FL, the main vulnerabilities of this scheme stem from the fact that the learning process occurs on the edge, typically on users' devices. This has created a new attack surface where a compromised device could be manipulated in a way to affect the learning process. Specifically, an adversary could either attack the training data (data poisoning) or the model updates, with the intention of affecting the global model (model poisoning) so that its performance deteriorates and ultimately lead to denial of service [17] or in order to make it perform a certain way. It has been demonstrated that an attacker may use malicious model updates in order to alter the behavior of a global model and be able to control its behavior when certain triggers appear (e.g. certain words in word prediction applications), without otherwise affecting the performance of the model on its main task [5]. Because the model still performs well on its main task, this type of attack cannot be easily detected by the aggregation server. In the worst case, when MPC is used, the source of the attack cannot be identified since the server does not have access to the individual model updates. The same study [5] shows that the attack is possible even with a 1% of compromised devices and can survive multiple training rounds.

## III. The Potential Benefits of Blockchain Adoption in Federated Learning

Distributed Ledger Technologies are a type of distributed databases collaboratively built (many entities contribute data) and replicated in a number of nodes (replicas of the whole database are kept in many nodes). In essence, they provide a shared and consistent data store, which usually allows multiple entities to contribute data. In contrast to traditional databases, the individual data records cannot be updated or deleted and are usually cryptographically interconnected. Most common categories of DLTs are Blockchains and Directed Acyclic Graphs. Although both of these technologies record transactions in the distributed ledger, they do so in different ways. In the case of blockchain, transactions are bundled into blocks, which once verified, are appended to a chain of previously verified blocks [18]. These blocks ultimately form a linear chain of blocks. In the case of DAGs, each transaction is attached to multiple existing transactions and during this process, these transactions are also being verified. Due to branching of existing transactions, the DAG usually resembles a tree. Furthermore, variants of DLT technologies can be further classified as permissioned or permission-less, allowing for other consensus types, which in turn offer advantages in specific use cases.

Further on, we analyze the benefits of combining the two technologies. In this section we use the term "blockchain" as this is widely used to refer to DLT technologies in total. The blockchain network can be used to store individual

contributions (in the form of model weights) inside blocks. These blocks alter the network's global state, which represents the new global model. It is anticipated that these solutions offer advancements in the following areas:

### A. Data integrity and Reliability

Federated Learning describes a process for collaboratively improving a shared model, and does not deal with security aspects. In its basic implementation, the process is coordinated from a single central server. The entity hosting the solution, could alter the data, either with malicious intent or unintentionally. Blockchain networks on the other hand are a proven technology that is inherently secure. All blocks are cryptographically interconnected, so in the case that a block is altered, this would be easily identifiable.

### B. Reliability

Since a blockchain network is decentralized, full copies of the ledger are maintained by multiple nodes. For this reason, there is no single point of failure since the ledger will be available elsewhere even if a node goes down. In a centrally managed FL scenario, one would need to implement a failover setup, typically involving redundant servers (VMs, Kubernetes), and a separate solution for backup.

### C. Trust

A blockchain is a good candidate to act as a trusted coordinator, because of its security and traceability properties. Blockchain solutions make use of a consensus algorithm, which guarantees that all nodes in a distributed ledger will reach an agreement and converge. These properties can enable trust between corporations in horizontal or federated learning use cases. As a part of the fourth industrial revolution, Industry 4.0 embraces data exchange and the creation of digital twins of the manufacturing processes is at its hype. Therefore, it is very likely that FL will become a priority for manufacturing industries

### D. Possibilities for incentives or rewards

In a Federated Learning setup, it may be important to incentivize users that contribute with quality data. This is crucial as the accuracy of the global model is directly proportional to the quality of the training data. Blockchain is the perfect medium to provide incentives in the form of tokens, which can be exchanged for services or financial rewards.

### E. Auditability – Traceability - Accountability

A data engineer can inspect the model learning process, by inspecting the individual blocks on the blockchain network. This offers increased auditability which might be an essential property in FL applications that require high level of trust on AI decisions such as in the military and manufacturing sector. Since the blocks on the network include the signature of the user initiating the transaction, the user cannot deny the authorship of this transaction. This property, known as non-repudiation, can be used for accountability.

## IV. Blockchain-based Federated Learning Approaches- Relevant Work

H. Kim et al [19] have proposed an architecture (BlockFL), where a blockchain solution is used instead of a central server to facilitate sharing of model updates from client devices. The proposed consensus algorithm is based on Proof of Work (PoW). The blocks on the network contain the model updates and miners are used to verify them and add them to the network. The advantages include incentives for devices that contribute to the training process with larger amounts of data, as well as solving the single point of failure in the case of a central server outage. They also study the effects of a miner's malfunction, imposed energy constraints and number of participating devices in respect to end-to-end latency and robustness.

U. Majeed et al [20] have proposed a similar architecture (FLchain) that includes features from Hyperledger Fabric and Ethereum, in which a separate fabric channel is used for each global learning model. The global model state is calculated after each new block generation. The suggested consensus algorithm is a modified version of Practical Byzantine Fault Tolerance (pBFT) and Proof-of-Word (PoW). The main focus in this solution is to improve auditability and governance.

D. Preuveneers et al [21] have implemented Federated Learning on a blockchain, for intrusion detection systems in computer networks, in order to explore auditability and accountability. Their setup relies on a permissioned block-chained network, in order to orchestrate machine learning models using federated learning. These models are then used to classify traffic for Intrusion Detection. The non-repudiation property of the blockchain network allows for enhanced accountability of contributing parties. The implementation is based on MultiChain, an opensource blockchain platform. The calculated overhead of the proposed solution in relation to traditional FL is estimated between 5%-15%.

More recently, similar research was performed by J. Weng et al [22] who implemented a blockchain assisted Federated Learning that focuses on incentive mechanisms and auditability. The setup is implemented on Corda V3.0 (a blockchain network sharing features of Bitcoin and Ethereum) and uses a custom consensus protocol based on the work of Algorand [23]. The learning environment is based on TensorFlow and the results show how the training accuracy increases with more participating parties.

Kang et al [24] have proposed a reputation-based approach that acts as an incentive mechanism in a FL setup. A reputation blockchain network is utilized in order to store weighting reputation opinions from recommenders. Based on these reputation weights and contract theory, an incentive mechanism is designed in order to motivate high reputation workers.

Dillenberger et al [25] in their publication have proposed easy to use tools that can be used with Hyperledger Fabric (but can also be applied to other blockchain solutions) in order to query the ledger and retrieve analytics.

It is evident that none of the above approaches address data or model poisoning attacks which are very important in the

Federated Learning process as they have great impact as already presented above.

## V. Proposed Algorithm to Assess the Quality of Contributed Model Updates

In this section, we propose an algorithm for Federated Learning, that can be incorporated in a blockchain environment and run inside a smart contract, in order to facilitate the learning process and provide protection against model poisoning. We are also presenting a high-level description of the algorithm and results based on its implementation in open-source tools: Keras and TensorFlow.

The steps involved in the operation of the algorithm are shown in Figure 2. As in the case of traditional FL, the process starts with the initialization of the global model and its relevant weights (with either random values or using proxy data). The global model is then distributed to participating parties. Local training is performed on end devices resulting in the generation of model updates in the form of weights. The coordinator upon receiving the updates, evaluates each update separately against a known good validation data set and records the accuracy. If the accuracy increases, the specific model update is considered reliable. If the accuracy decreases, the update is discarded. A qualitative measure for each update can be derived by measuring the distance between the global model accuracy and the accuracy of the global model when averaged with the update.

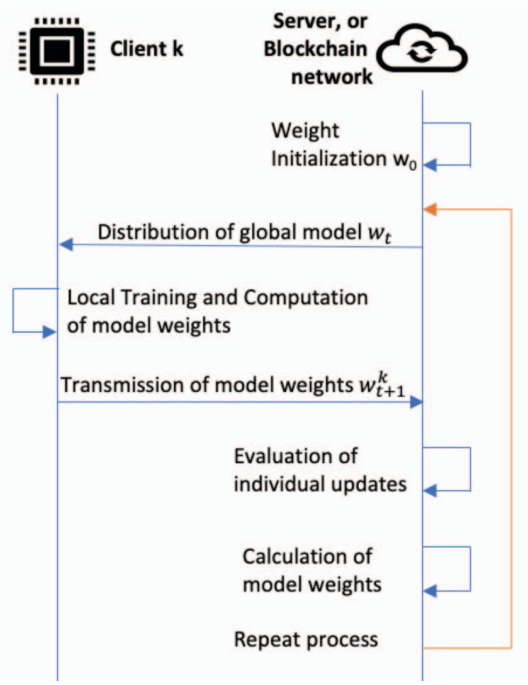


Figure 2 Steps involved in the proposed algorithm

The pseudocode in Algorithm 1 is a high-level overview of the implementation, based on the Federated Learning algorithm and the use of stochastic gradient descent (SGD). During averaging, it intentionally does not take into consideration the clients' data sample size, since in real use cases this would not be known by the server and could be easily falsified by an adversary in order to maximize his effect on the global model. This approach does not measure the quality of each update; instead it only measures if each update increases accuracy in a Boolean logic. Further improvements could be made in order to measure the quality and use this as a metric for an incentive or rewards mechanism.

**Algorithm 1** – Discarding of unreliable model updates

$w_0$ are the weights of the initial model, λ is the learning rate of the global model, η is the learning rate of the local model, K contains all Clients, Pk contains all data samples of client k, B is the local batch size, GetAccuracy is a function which returns the accuracy of the specified model against the verification dataset, $r_{t+1}^k$ is the received (candidate) update from end-device k to be used in the next model update, and $w_{t+1}$ is the weighted average of all verified updates.

**procedure** SERVER
    initialize $w_0$
    **for** each round t = 1,2,… do
        **for** each client k in parallel do
            $r_{t+1}^k \leftarrow ClientUpdate(w_t)$
            if VerifyUpdate($w_t$,$r_{t+1}^k$) then
                $w_{t+1}^k = r_{t+1}^k$
            end if
        **end for**
        $w_{t+1} = \lambda w_t + (1-\lambda)\sum_{k=1}^{K}\frac{1}{K}w_{t+1}^k$
    **end for**
**end procedure**
**procedure** ClientUpdate($w_t$)
    B ← split Pk to smaller sets
    **for** all b ∈ B do
        W←w-η∇f($w_t$,b)
    **end for**
    **return** W *(computed weights obtained by minimizing loss function f($w_t$,b))*
**end procedure**
**procedure** VerifyUpdate($w_t$,$r_{t+1}^k$)
    $p_{t+1} = \lambda w_t + (1-\lambda)r_{t+1}^k$
    a = GetAccuracy($p_{t+1}$)
    b = GetAccuracy($w_t$)
    **if** a > b then
        return True
    **else**
        return False
    **end if**
**end procedure**

The *SERVER* procedure initializes the module weights by either random numbers, or created using proxy data. Then, for each round, it distributes the current model version ($w_t$) to all clients. After the local training process is finished, the individual candidate model updates $r_{t+1}^k$ are retrieved and verified by the procedure *VerifyUpdate*. In the simplest form, this procedure will return a Boolean value, depending on whether the model update improved the overall model

performance against a verification dataset. The new model version $w_{t+1}$ is calculated by averaging all the model updates for which the *VerifyUpdate* procedure returned True. The configurable parameter λ is used to specify the learning rate, affecting the impact of the training round on the global model.

The *ClientUpdate* procedure is responsible for the local training process. First, the data samples Pk are split into smaller batches of size B, which are then used to minimize a loss function f. The reason for choosing a smaller batch size for training is because it speeds up the training process significantly, especially when the end device is low powered such as a mobile phone. The method of splitting a large sample size in smaller batch sizes for training is known as Stochastic Gradient Descend (SGD)[6]. The configurable parameter η, is used to specify the local learning rate, and affects the impact of the training process on the specific model update.

The *VerifyUpdate* procedure is used to compare the performance of the current model $w_t$, against the performance of the weighted average of $w_t$ with the candidate update $r_{t+1}^k$. The performance is evaluated against a verification dataset which is chosen before-hand, and remains the same throughout all rounds.

The novelties of the proposed scheme are: a) we propose an alternative way to evaluate the model updates, i.e. based on the accuracy improvements they bring, which obviates the need to know the dataset size each device possesses which can be falsified, b) due to the execution of the logic in a smart contract, the logic cannot be compromised, c) we exploit the traceability capacity of the blockchain to discourage malicious users which could try to poison the model.

When implementing the algorithm inside a smart contract, the following should be considered with respect to the blockchain framework: a) the smart contract must be able to execute external tools, such as libraries used for model evaluation, b) all nodes must have access to the same verification dataset and c) this dataset should be kept private from clients. The above requirements suggest that a private permissioned blockchain network such as Hyperledger Fabric is a better candidate.

## VI. Experimental Setup and Simulation Results

The following simulation uses Tensorflow and Keras to measure the effectiveness of the above algorithm. We perform FL using the popular MNIST database consisting of a database of 60.000 handwritten digit images and 10.000 verification images and corresponding labels. The model consists of an input layer (receives flattened images of size 28x28), a fully connected layer, and an output layer of size 10 which is used as the classifier. The optimizer selected is Adam[26], an adaptive learning rate algorithm that is based on SGD with default base learning rate of 0.001. The simulation assumes 10 participating parties. In each simulation round, a (changing) subset of the nodes are honest and the rest are adversaries. The training images are first split into 10 chunks which are used by the 10 parties respectively. While both honest and adversaries train on their corresponding training images, the honest use the correct training labels while the adversaries use an altered label set. The labels in this malformed set were assigned to a static number. The intention of the adversary in this scenario is to make the global model always predict this number. During simulation, we experimented with additional malformed label sets such as randomly chosen labels; however, the static number approach was found to have greater negative impact on the model accuracy. Each simulation runs for 10 rounds and each local training is performed over 1 epoch. After each round, the model weights are averaged and distributed to the other parties.

Figure 3 shows the degradation of accuracy in this FL setup with varying percentage of adversaries when individual updates are not verified (traditional FL). We notice degraded model performance, both in terms of speed (training rounds) and convergence. Specifically, when more than 10% of participants are adversaries, the model accuracy is below 90%, with little or no convergence.

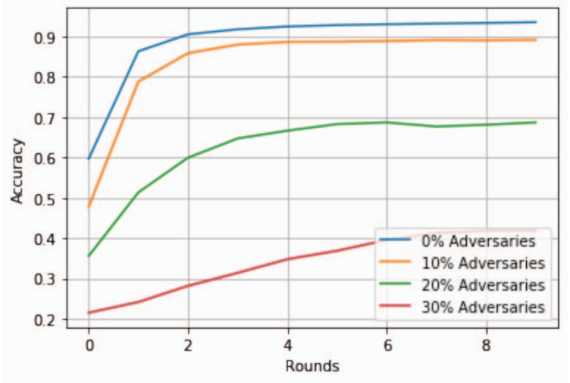


Figure 3 Performance with traditional FL algorithm in the presence of adversaries

Figure 4 shows how the algorithm has correctly identified the falsified model updates and the model accuracy is able to converge in the presence of 30% adversaries. During the first 2 rounds, the difference in accuracy levels is noticeable and is attributed to the fact that the model is trained with less data (since a considerable amount is discarded). The situation is rectified with each successive round.

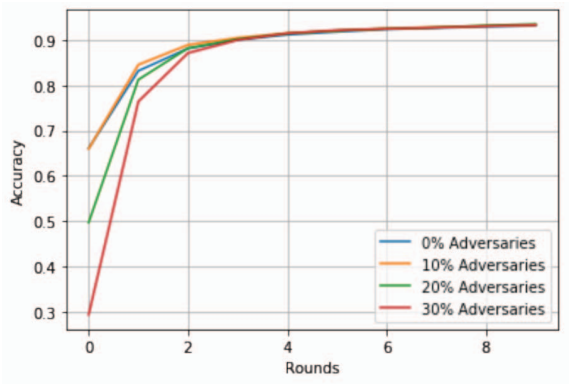


Figure 4 Performance with model update protection in the presence of adversaries

## VII. Conclusions and further work

In this paper, we have examined the current research related to the use of Blockchain Technologies in the field of Federated Learning and we have identified a potential for blockchain technology to improve security of the FL process. We have further proposed an algorithm that is able to run inside a smart contract of a blockchain network. In contrast to the techniques currently available, where researchers relied on the use of data sample size[19],[22] or reputation[24] as a quality metric, our solution measures the accuracy of the model update directly. In this way, we do not need to assume that participants of the Federated Learning process are honest. The first results show that the algorithm provides a high level of protection against model poisoning attacks. We anticipate that results obtained can be considered as a baseline when implementing the algorithm on various blockchain technologies and that the algorithm can be extended in order to offer rewards on a blockchain network, relative to the quality of each contribution. In our next steps, we will a) perform exhaustive tests to study our algorithm along different parameters and b) implement it in an open source blockchain platform.

## References


[1] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated Machine Learning," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, Jan. 2019, doi: 10.1145/3298981.

[2] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated Learning: Strategies for Improving Communication Efficiency," pp. 1–10, 2016.

[3] Z. Ghahramani, "LNAI 3176 - Unsupervised Learning," pp. 72–112, 2004.

[4] A. Hard *et al.*, "Federated Learning for Mobile Keyboard Prediction," 2018.

[5] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How To Backdoor Federated Learning," Jul. 2018.

[6] L. Bottou, "Large-Scale Machine Learning with Stochastic Gradient Descent," *COMPSTAT*, vol. 19th Inter, 2010, doi: 10.1007/978-3-7908-2604-3_16.

[7] European Parliament and Council of the European Union, "Regulation on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (Data Protection Directive)," *Off. J. Eur. Union*, vol. L 119, pp. 1–88, 2016.

[8] M. (University of M. A. Nasr, R. (National U. of S. Shokri, and A. (University of M. A. Houmansadr, "Comprehensive Privacy Analysis of Deep Learning," *Sp*, pp. 1–15, 2018.

[9] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," *Proc. - IEEE Symp. Secur. Priv.*, vol. 2019-May, pp. 691–706, 2019, doi: 10.1109/SP.2019.00029.

[10] A. Narayanan and V. Shmatikov, "How To Break Anonymity of the Netflix Prize Dataset," 2006.

[11] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," *Proc. - IEEE Symp. Secur. Priv.*, pp. 111–125, 2008, doi: 10.1109/SP.2008.33.

[12] K. Bonawitz *et al.*, "Practical Secure Aggregation for Privacy-Preserving Machine Learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security - CCS '17*, 2017, vol. 9, pp. 1175–1191, doi: 10.1145/3133956.3133982.

[13] M. Abadi *et al.*, "Deep learning with differential privacy," *Proc. ACM Conf. Comput. Commun. Secur.*, vol. 24-28-Octo, no. Ccs, pp. 308–318, 2016, doi: 10.1145/2976749.2978318.

[14] R. C. Geyer, T. Klein, and M. Nabi, "Differentially Private Federated Learning: A Client Level Perspective," no. Nips, pp. 1–7, 2017.

[15] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating Noise to Sensitivity in Private Data Analysis," *J. Priv. Confidentiality*, vol. 7, no. 3, pp. 17–51, May 2017, doi: 10.29012/jpc.v7i3.405.

[16] D. Evans, V. Kolesnikov, and M. Rosulek, "A Pragmatic Introduction to Secure Multi-Party Computation," *Found. Trends® Priv. Secur.*, vol. 2, no. 2–3, pp. 70–246, 2018, doi: 10.1561/3300000019.

[17] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local Model Poisoning Attacks to Byzantine-Robust Federated Learning," no. August, 2019.

[18] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," in *21st Century Economics*, 2008.

[19] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained On-Device Federated Learning," *IEEE Commun. Lett.*, vol. PP, no. c, pp. 1–1, 2019, doi: 10.1109/LCOMM.2019.2921755.

[20] U. Majeed and C. S. Hong, "FLchain: Federated Learning via MEC-enabled Blockchain Network," in *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2019, no. September, pp. 1–4, doi: 10.23919/APNOMS.2019.8892848.

[21] D. Preuveneers, V. Rimmer, I. Tsingenopoulos, J. Spooren, W. Joosen, and E. Ilie-Zudor, "Chained Anomaly Detection Models for Federated Learning: An Intrusion Detection Case Study," *Appl. Sci.*, vol. 8, no. 12, p. 2663, Dec. 2018, doi: 10.3390/app8122663.

[22] J. Weng, J. Weng, J. Zhang, M. Li, Y. Zhang, and W. Luo, "DeepChain: Auditable and Privacy-Preserving Deep Learning with Blockchain-based Incentive," *IEEE Trans. Dependable Secur. Comput.*, vol. PP, no. 8, pp. 1–1, 2019, doi: 10.1109/tdsc.2019.2952332.

[23] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand : Scaling Byzantine Agreements for Cryptocurrencies," 2017.

[24] J. Kang, Z. Xiong, and D. Niyato, "Incentive Mechanism for Reliable Federated Learning : A Joint Optimization Approach to Combining Reputation and Contract Theory," *IEEE Internet Things J.*, vol. PP, no. c, p. 1, 2019, doi: 10.1109/JIOT.2019.2940820.

[25] D. N. Dillenberger *et al.*, "Blockchain analytics and artificial intelligence," *IBM J. Res. Dev.*, vol. 63, no. 2, 2019, doi: 10.1147/JRD.2019.2900638.

[26] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," pp. 1–15, Dec. 2014.