



Article

Privacy and Trust Redefined in Federated Machine Learning

Pavlos Papadopoulos ^{1,*} , Will Abramson ¹, Adam J. Hall ¹, Nikolaos Pitropakis ^{1,2} and William J. Buchanan ¹

¹ Blockpass ID Lab, School of Computing, Edinburgh Napier University, Edinburgh EH10 5DT, UK; will.abramson@napier.ac.uk (W.A.); adam.hall@napier.ac.uk (A.J.H.); N.Pitropakis@napier.ac.uk or nikolaos.pitropakis@8bellsresearch.com (N.P.); B.Buchanan@napier.ac.uk (W.J.B.)

² Eight Bells LTD, Nicosia 2002, Cyprus

* Correspondence: pavlos.papadopoulos@napier.ac.uk

Abstract: A common privacy issue in traditional machine learning is that data needs to be disclosed for the training procedures. In situations with highly sensitive data such as healthcare records, accessing this information is challenging and often prohibited. Luckily, privacy-preserving technologies have been developed to overcome this hurdle by distributing the computation of the training and ensuring the data privacy to their owners. The distribution of the computation to multiple participating entities introduces new privacy complications and risks. In this paper, we present a privacy-preserving decentralised workflow that facilitates trusted federated learning among participants. Our proof-of-concept defines a trust framework instantiated using decentralised identity technologies being developed under Hyperledger projects Aries/Indy/Ursa. Only entities in possession of Verifiable Credentials issued from the appropriate authorities are able to establish secure, authenticated communication channels authorised to participate in a federated learning workflow related to mental health data.

Keywords: trust; machine learning; federated learning; decentralised identifiers; verifiable credentials



Citation: Papadopoulos, P.; Abramson, W.; Hall, A.J.; Pitropakis, N.; Buchanan, W.J. Privacy and Trust Redefined in Federated Machine Learning. *Mach. Learn. Knowl. Extr.* **2021**, *3*, 333–356. <https://doi.org/10.3390/make3020017>

Academic Editor: Edgar Weippl

Received: 30 January 2021

Accepted: 24 March 2021

Published: 29 March 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Machine Learning (ML) and Deep Neural Networks (DNN) gained popularity in the last few years due to technology advancement. ML and DNN infrastructures can analyse a vast amount of information to predict a certain case [1–3]. DNN is a part of ML that tries to replicate a human's brain neurons' functionalities to achieve a prediction. This analysis and prediction functionalities can deal with complex problems that were previously considered to be unsolvable. ML predictions become more valuable when the analysis involves highly sensitive private data such as health records. Consequently, data holders cannot simply share their private data with ML algorithms and experts [4]. Many defensive techniques proposed in the past as countermeasures to the information leakage of sensitive data, such as anonymisation and obfuscation techniques [5]. Other research focused on non-iterative Artificial Neural Network (ANN) approaches for data security [6,7]. Nevertheless, due to the advancement of technology, similar techniques cannot anymore guarantee the privacy of the underlying data [8,9]. Malicious users are able to reverse and reconstruct, anonymised and obfuscated data, in order to identify the identities of the data subjects.

It is common knowledge that data is the most valuable asset of our century. Since ML algorithms require vast amounts of it, it is frequently targeted by malicious parties. Several attacks exist that can hack, reconstruct, reverse or poison ML algorithms [10,11]. The common goal of these attacks is to identify the underlying data. Several of them require access during the ML algorithm training to succeed, while others are able to interfere later in the testing or publication phase. In the literature, there are several defensive methods and techniques proposed against the aforementioned mistreats. However, when reinforcing a ML algorithm with security and privacy features against adversarial attacks, there is

an impact on efficiency, thus ending up with the produced predictions not related to the associated tasks and with often a lower accuracy. Hence, a balance between tolerable defence and usability is a critical point of interest that many researchers are trying to solve. Most of the aforementioned attacks are applicable and target centralised ML infrastructures, in which the model owners have access and acquire all the training data [12].

Due to the importance of the ML field and the associated privacy risks, a new division of ML was created, namely Privacy-Preserving ML (PPML) [13]. This area is focused on the advancement and development of countermeasures against information leakage in ML, by shedding light on the privacy of the underlying data subjects. When a few of these ML risks and attacks were introduced, they were purely theoretical; hence, due to the rapid advancement and evolution of ML, attackers led to the exploitation of those weaknesses in order to breach, steal, and profit from this data. Several techniques and countermeasures were proposed in the PPML field. It is a common belief that if data never leave their holders possession to be used from a ML algorithm, then the data privacy is higher. The most extended and researched area related to this is Federated Learning (FL) [14–17].

In a FL scenario, the ML model owners are able to send their model to the data holders for training. From a high-level perspective, this scenario is secure; however, there are still many security flaws that need to be solved [10]. For example, the ML model owners could reverse their model and identify the underlying training data [18,19]. A suggested solution is to use a secure aggregator, often an automated procedure or program, as a middle-ware, which aggregates all the participants' trained models and then sends the updates to the ML model owners [20]. This solution is robust against several attacks [18,19,21–23], but still involves issues, such as the possibility of a Man-In-The-Middle (MITM) attack that is able to interfere and trick both parties or even the scenario where one or more participants are malicious. In the latter scenario, malicious data providers can poison the ML model [24–26] in order to miss-classify specific predictions in its final testing phase; thus, the ML model owner could never distinguish a poisoned model from a benign. This model poisoning scenario could happen in a healthcare auditing scheme, where the trusted auditing organisation uses a ML algorithm to audit other healthcare institutions to predict financial profits and losses in the future. A potential malicious healthcare institution is able to poison the ML model using indistinguishable data that could lead to false predictions from the final trained model by miss-classifying the economic losses of the healthcare institution and approve its operation, as usual.

The aforementioned attacks share some common issues and concerns such as the lack of trust between the participating parties, or the lack of a secure communication channel to transmit private ML model updates. In this work, we redefine the privacy and trust in federated machine learning by creating a Trusted Federated Learning (TFL) framework as an extension of the privacy-preserving technique to facilitate trust amongst federated machine learning participants [27]. In our scheme, the ML participants need to get a certification before their participation, from a trusted governmental body such as the National Health Service (NHS) Trust in the United Kingdom. Following, the ML training procedure is distributed among the participants similarly to FL. The main difference is that the model updates are being sent through a secure communication end-to-end encrypted channel. Before learning commences, the respective parties must authenticate themselves against some predetermined policy. Policies can be flexibly determined based on the ecosystem, trusted entities and associated risk; this paper gives an example of using a healthcare scenario. The proof-of-concept developed in this paper is built using open-source Hyperledger technologies such as Aries/Indy/Ursa [28–30] and developed within the PyDentity-Aries FL project [31,32] of the OpenMined open-source privacy-preserving machine learning organisation. Our scheme is based on advanced privacy-enhancing attribute-based credential cryptography [33,34] and is aligned with emerging decentralised identity standards; Decentralized Identifiers (DIDs) [35], Verifiable Credentials (VCs) [36] and DID Communication [37]. The implementation enables participating entities mutually authenticate digitally signed attestations (Credentials), issued by trusted entities specific to

the use-case. The presented authentication mechanisms could be applied to any regulatory workflow, data collection, and data processing and are not limited solely to the healthcare domain. The contributions of our work could be summarised as follows:

- We enable stakeholders in the learning process to define and enforce a trust model for their domain through the application of decentralised identity standards. We also extended the credentialing and authentication system by separating Hyperledger Aries agents and controllers into isolated entities.
- We present a decentralised peer-to-peer infrastructure, namely TFL, which uses DIDs and VCs in order to perform mutual authentication and federated machine learning specific to a healthcare trust infrastructure. Development and evaluation of explicitly designed libraries for federated machine learning through secure Hyperledger Aries communication channels.
- We demonstrate performance improvement upon our previous trusted federated learning state-of-the-art without sacrificing the privacy guarantees of the authentication techniques and privacy-preserving workflows.

Section 2 provides the background knowledge and describes the related literature. Furthermore, Section 3 outlines our implementation overview and architecture, followed by Section 4, in which we provide an extensive security and performance evaluation of our system. Finally, our work concludes with Section 5 that draws the conclusions, limitations, and outlines approaches for future work.

2. Background Knowledge and Related Work

Recent ML advancements can accurately predict specific circumstances using relevant data. Hence, that led businesses and organisations to collect vast amounts of data to predict a situation before their competitors. The rationale is often to analyse people's behaviour patterns to predict the next trend they will follow [38]. However, the European Union tried to minimise and constrain this massive collection of data with the General Data Protection Regulation (GDPR) legislation [39].

Another field that can take advantage of the recent ML progression is the healthcare sector. However, in that case, the underlying data used for ML training is sensitive and private. Thus, its privacy must be ensured first prior to the improvement of its ML predictions. The aforementioned procedure's complexity raises when it is being outsourced to a third-party organisation specialising on the ML task; since the ML practitioners have the expertise to solve the task, but a healthcare organisation is holding the required sensitive data.

2.1. Trust and the Data Industry

The notion of trust has been defined as domain and context-specific since it specifies the amount of control a party provides to another [40,41]. It is often represented as a calculation of risk since it can only be restrained and not fully eradicated [42]. Accordingly, patients trust healthcare institutions when giving their consent to collect their data. However, huge volumes of medical data can be valuable in-context to ML algorithms that aim to predict particular cures or conditions.

In 2015, Royal Free London NHS Trust outsourced patients sensitive data to a third-party ML company, particularly DeepMind, to train ML algorithms for the early detection of kidney failure [43,44]. However, this sensitive data usage was not regulated, raised concerns about data privacy and later judged as illegal by the Information Commissioner's Office [45]. This misbehaviour did not cause other researchers to use sensitive data for ML predictions and led them to obtain proper authorisation from the Health Research Authority first and then use the sensitive medical records to analyse retinal imaging automatically [46], and the segmentation of tumour volumes and organs of risk [47].

2.2. Decentralised Identifiers

Recently, Decentralised Identifiers (DIDs) were established as a digital identifier in a World Wide Web Consortium (W3C) working group [35], that can magnify trust in distributed environments. DIDs can be controlled solely by their owners and grant a person the ability to be authenticated similar to a login system, but without relying on a trusted third-party company. Consequently, DIDs are often stored in distributed ledgers such as blockchain ledgers, which are not managed by a single authority. Distributed storage systems such as Ethereum, Bitcoin and Sovrin ledgers, or InterPlanetary File System (IPFS) are often used to store DID specifications, each with their own resolution method [48]. An outline of a DID document that would have been resolved from *did:example:123456789abcdefghi*, using the DID method *example* and the identifier *123456789abcdefghi*, can be seen in Listing 1. A DID document consists of:

- ID-the DID that resolves to this document
- Public key
- Authentication protocols
- Service endpoints

Listing 1. An example DID document.

```

1 {
2   "@context": "https://example.org/example-method/v1",
3   "id": "did:example:123456789abcdefghi",
4   "publicKey": [{
5     "id": "did:example:123456789abcdefghi#keys-1",
6     "type": "RsaVerificationKey2018",
7     "controller": "did:example:123456789abcdefghi",
8     "publicKeyPem": "-----BEGIN PUBLIC KEY...END PUBLIC KEY-----\r\n"
9   }],
10  "authentication": [
11    "did:example:123456789abcdefghi#keys-1",
12  ],
13  "service": [{
14    "id": "did:example:123456789abcdefghi#agent",
15    "type": "AgentService",
16    "serviceEndpoint": "https://agent.example.com/8377464"
17  }]
18 }

```

DID specifications assure the interoperability across the DID schemes in order to interact and resolve a DID from any storage system. Nonetheless, Peer DIDs implementations are used in peer-to-peer connections that do not require any storage system, in which each peer stores and maintains their own list of DID documents [49].

Decentralised Identifiers Communication Protocol

Hyperledger Aries is an open-source project [28], that uses decentralised identifiers to provide a public key infrastructure for a set of privacy-enhancing attribute-based credential protocols [34]. Hyperledger Aries implements DID Communication (DIDComm) [37], a communication protocol similar to one first outlined by David Chaum [50]. DIDComm is an asynchronous encrypted communication protocol that uses information from the DID document, such as the public key and their associated endpoint, in order to exchange secure messages; the authenticity and integrity of the messages are verifiable. DIDComm protocol is actively developed by the Decentralised Identity Foundation [51].

An example using the DIDComm protocol can be seen in Algorithm 1, in which Alice and Bob want to communicate securely and privately. Alice encrypts and signs a message for Bob. Alice's endpoint sends the signature and the encrypted message to Bob's endpoint. Bob can verify the message's integrity by resolving the DID and checking if it corresponds

to Alice's public key, decrypt and read the message. All the associated information required for this interaction are defined in each person's respective DID document. The encryption techniques used by DIDComm include ElGamal [52], RSA [53] and elliptic curve-based [54].

Algorithm 1 DID Communication Between Alice and Bob [27]

- 1: Alice has a private key sk_a and a DID Document for Bob containing an endpoint ($endpoint_{bob}$) and a public key (pk_b).
 - 2: Bob has a private key (sk_b), and a DID Document for Alice containing her public key (pk_a).
 - 3: Alice encrypts plaintext message (m) using pk_b and creates an encrypted message (e_b).
 - 4: Alice signs e_b using her private key (sk_a) and creates a signature (σ).
 - 5: Alice sends (e_b, σ) to $endpoint_{bob}$.
 - 6: Bob receives the message from Alice at $endpoint_{bob}$.
 - 7: Bob verifies σ using Alice's public key pk_a
 - 8: **if** $Verify(\sigma, e_b, pk_a) = 1$ **then**
 - 9: Bob decrypts e_b using sk_b .
 - 10: Bob reads the plaintext message (m) sent by Alice
 - 11: **end if**
-

2.3. Verifiable Credentials

Verifiable Credentials (VCs) [36], is a set of tamper-proof claims that used by three different entities, *Issuers*, *Holder*s and *Verifiers*, as it can be seen in Figure 1. VC model specification became a W3C standard in November 2019. A distributed ledger is often used for the storage of the credential schemes, DIDs, and Issuers' DID documents.

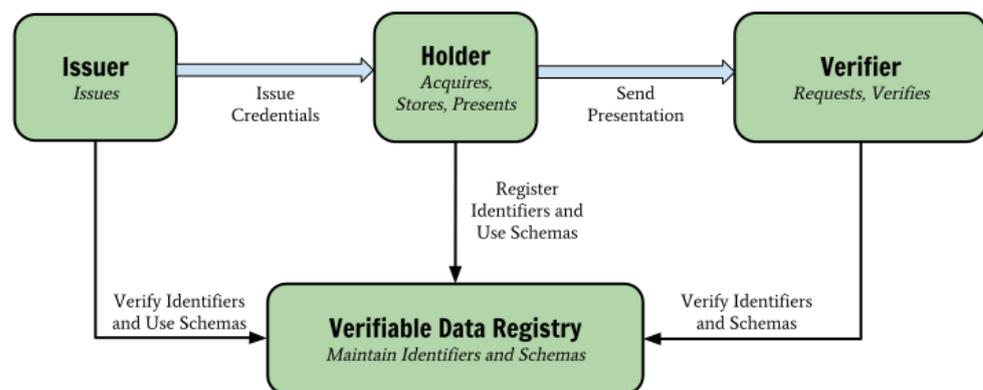


Figure 1. Verifiable Credential Roles [36].

The Issuer to create a new credential needs to generate a signature using their private key corresponded to their public key defined in their DID document. There are three valid categories of signature schemes such as Camenisch-Lysyanskaya (CL) signatures [33,55], Linked Data signatures [56] and JSON Web signatures [57]. Hyperledger Aries uses CL signatures to create a blinded link secret, in which credentials are tied to their intended entities by including a private number within them, without the Issuers be aware of their values. It is a production implementation of a cryptographic system for achieving security without authentication first outlined in 1985 [58].

The Verifier in order to accept the received credential from its Holder needs to confirm the following:

1. The Issuer's DID can be resolved to a DID document stored on the public ledger. The DID document contains the public key that can be used to ensure the credential's integrity.
2. The credential Holder can prove the blinded linked secret by creating a zero-knowledge proof to demonstrate it.
3. The issuing DID has the authority to issue this kind of credential. The signature solely proves integrity, but if the Verifier accepts credentials from any Issuers, it would be prone to obtain fraudulent credentials. It is possible to form a legal document outlining the operating parameters of the ecosystem [59].
4. The Issuer has not revoked the presented credential. This is done by checking that a revocation identifier for the credential is not present within a revocation registry (a cryptographic accumulator [60]) stored on the public ledger.
5. Finally, the Verifier needs to check that the credential attributes meet authorisation criteria in the system. It is common for a credential attribute to be valid only for a certain period.

All the communication between the participating entities transmits peer-to-peer through a DIDComm protocol. It should be noted that a Verifier does not require to contact the credential's Issuer to verify a credential.

2.4. Docker Containers

All the participating entities presented in our work, take the form of Docker containers [61]. Docker containers are lightweight, autonomous, virtualised systems similar to virtual machines [62]. The main difference between virtual machines is that Docker containers use the host's underlying operating system and bridge the network traffic in a virtual network card instead of being fully isolated. Moreover, Docker containers are being developed into deployable images that are executed and operate as expected invariably in any system that supports the Docker environment. Hence, applications that could be built using Docker containers are favourable for reproducibility and code replication purposes. However, since Docker containers use a virtual network card in their host machine's to redirect the network traffic, a security testing in their ecosystem varies [63].

2.5. Federated Machine Learning

FL can be expressed as the decentralisation of the ML. Opposed to centralised ML, in a FL scenario, the training data remain at their respective owners instead of transmitting to a central location to be used by a ML practitioner. There are several FL variations such as Vanilla FL, Trusted Model Aggregator, and Secure Multi-Party Aggregation [10,20,27,64]. Consequently, the ML model is primarily distributed among the data holders, who train it using their private data, and then send it back to the ML model owner. ML training decentralisation permits data holders with sensitive data such as healthcare institutions to train useful ML algorithms to predict a cure or a disease. One of the FL advancements, namely Secure Multi-Party Aggregation, developed to further enhance the system's security by encrypting the models into multiple shares and aggregating all the trained models to eliminate the possibility of a malicious ML model owner [17,65].

To measure the accuracy of FL algorithm is similar to the traditional ML. Four metrics described in the list below [66,67], are used for the calculation as follows:

1. True positive (TP): the model correctly predicts the positive prediction; correct
2. True negative (TN): the model correctly predicts the negative prediction; correct
3. False positive (FP): the model incorrectly predicts the positive prediction; false
4. False negative (FN): the model incorrectly predicts the negative prediction; false

The accuracy of the model is calculated from the number of *TPs* and *TNs*, divided by the total number of outcomes, as you can see in Equation (1).

$$\text{Model's Accuracy} = \frac{(\text{Correct Predictions} : TP + TN)}{(\text{Outcomes} : TP + TN + FP + FN)} \quad (1)$$

2.6. Attacks on Federated Learning

Since that in FL, data never leaves their owners' premises, one could naively assume that FL is entirely protected against misuses. However, even if FL is more secure than traditional ML approaches, it is still susceptible to several privacy attacks that aim to identify the underlying training data or trigger a miss-classification on the final trained model [10,68].

Model Inversion attack [18,19,69], is the first of its kind that aim to reconstruct the training data. A potential attacker with access to the target labels can query the final trained model and exploit the returned classification scores to reconstruct the rest of the data.

In Membership Inference attacks [22,23], the attacker tries to identify if some data was part of the training. As with model inversion attacks, the attacker exploits the returned classification scores in order to create several *shadow* models that have similar classification boundaries as the original model under attack.

In Model Encoding attacks [21], the attacker with white-box access to the model tries to identify the training data that have been *memorised* by the model's weights. In a black-box situation, the attacker overfits the original training model in order for it to leak part of the target labels.

From the other side, Model Stealing attacks [70], present the scenario of a malicious participant that tries to *steal* the model. Since the model is being sent to the participants for training, malicious participants can construct a second model that mimics the original model's decision boundaries. In that scenario, the malicious participants could avoid paying usage fees to the original model's ML experts or sell the model to third parties.

Likewise, in Model Poisoning attacks [24–26,71], since the malicious participants contribute to the training of the model, they are able to inject backdoor triggers to the trained model. According to [24], a negligible number of malicious participants is able to poison a large model. Hence, the final trained model would seem legitimate to the ML experts and react maliciously only on the given backdoor trigger inputs. In that case, the malicious participant could potentially *trick* the original model when certain inputs are given. Contrary to the Data Poisoning attacks [12,72–75], in which the poison backdoor triggers are part of the training data, and the ML model's accuracy may drop [76].

In Adversarial Examples [77–82], the attacker tries to *trick* the model in order to classify falsely a prediction. The threat model for this type of attacks is both white-box and black-box; thus the attacker does not require access to the training procedure, with a potential attacking scenario to be a malware that evades the detection of a ML intrusion detection system.

2.7. Defensive Methods and Techniques

Due to the sensitivity of the underlying training data in ML, there are several defensive techniques, albeit several of them are still in the theoretical stage and therefore are not applicable [10,27].

2.7.1. Differential Privacy

The most eminent defensive countermeasure against many privacy attacks in ML is Differential Privacy (DP), a mathematical guarantee that ensures the ML algorithm's output, despite if a particular person's data used for the training procedure [83,84]. The formal mathematical proof can be seen in Equation (2), where the probability A for all C that are in

range (A), is differentially private, if for any two adjusted databases D and D' that alter in only one element exists:

$$P(A(D) \in C) \leq e^\epsilon P(A(D') \in C) \quad (2)$$

DP elaborates noise techniques to protect a ML algorithm from attacks; however, its accuracy drops significantly according to the designated privacy level [85]. Researchers, further extended and relaxed this mathematical proof into (ϵ, δ) -DP, which introduced an extra δ feature that limits the probability for errors [85–88].

2.7.2. Secure Multi-Party Computation

Secure Multi-Party Computation (SMPC) [89], is a cryptographic function that allows several participants to compute a procedure mutually, such a ML training procedure. Only the outcome of the function is disclosed to the participating parties and not the underlying training information. Using SMPC, gradients and parameters can be computed and updated encrypted in a decentralised manner. In this case, each data item's custody is split into shares to be held by relevant participating entities. SMPC is able to protect ML algorithms against privacy attacks that target the training procedure; however, attacks during the testing phase are still viable.

2.7.3. Homomorphic Encryption

Homomorphic Encryption (HE) [90], is a complex cryptographic protocol, which allows the mathematical computation of encrypted data. The outcome of the computation is still encrypted. HE is a promising method to protect both the training and testing procedures; however, such an intensive technique's high computational cost is not tolerable in real-world situations. Several HE schemes in the literature propose alterations and evaluations of the method [10,91–94].

2.8. Related Work

Our work is not another defensive method or technique that mitigates the aforementioned FL attacks, as seen in Section 2.6. Hence, a comparison with defensive techniques such as Knowledge Distillation [95,96], Anomaly Detection [21], Privacy Engineering [97], Privacy-Preserving Record Linkage [98], Adversarial Training [99], ANTIDOTE [100], Activation Clustering [101], Fine-pruning [102], STRIP [103], or similar, is not comparable and out of the scope of this paper.

The concern related to the privacy of the stored data has been extensively researched in the literature. Many researchers proposed and presented novel infrastructures and concepts that could partially or fully protect data. However, the privacy-preservation of critical data such as medical records often is more important than the actual procedure that it has been used, such as the training of a ML algorithm. There are works that presented the use of another emerging technology such as blockchain, which could be combined with ML. In the work of [104,105] the authors' presented infrastructures that could protect certain private data from the stored records and display of other non-private. However, the feasibility of performing ML in data stored in their blockchain has not been tested and remains an open question.

Another state-of-the-art technology that is similar to our work is the Private Set Intersection (PSI). Using PSI, participants of an infrastructure can compare the private records they share, without disclosing them to the other participants [106]. There are applications that use PSI for privacy-preserving contact tracing systems and machine learning on vertically partitioned datasets [107].

The value of developing an ecosystem and associated governance framework to facilitate the issuance and verification of integrity assured attributes had been considered previously in a different setting [108]. This work develops user-led requirements for a staff passporting system to reduce the administrative burden placed on healthcare professionals as they interact with different services, employers and educational bodies throughout their

careers. These domain-specific systems that digitally define trustworthy entities, policies and information flows have multiple use cases and appear to be a positive indication of the likelihood of broader adoption of the technologies discussed in this paper.

Achieving privacy-enhancing identity management systems has been a focus of cryptographic research since Chaum published his seminal paper in 1985 [58]. The Hyperledger technology stack used in this work is an implementation of a set of protocols formalised by Camenisch and Lysyansky [33,109] and follows a technical architecture closely aligned to the one produced as part of an EU grant ABC4Trust [110]. Self-Sovereign Identity (SSI) has popularised the model of issuer/verifier/holder, with many different projects and implementations building to the emerging standards in this area [111], although not all of these projects use privacy-enhancing cryptography. Furthermore, the major focus of these systems has been the identification and authentication of individuals by issuing and verifying their credentials within a certain context [108,112].

Our work differentiates from the other approaches since we focus on modelling trust relationships between organisations using the mental model of SSI and the technology stack under development in the Hyperledger foundation. More specifically, in our proof-of-concept, we establish trusted connections among only authorised participants and then perform FL over secure communication channels. In our previous work [27], we presented a proof-of-concept that is able to establish trust between the participating parties and perform FL on their data. Thus, to achieve it, we used the basic messages protocol provided by Hyperledger Aries, encoded the ML model and updates into text format and sent it through DIDComm encrypted channels. However, in this paper, we have refactored this functionality into libraries that developed mutually within the OpenMined open-source community in PyDentity [31], for that purpose. Additionally, we have thoroughly presented their communication details in Section 3.2, and have tested their security in Section 4.1.1.

3. Implementation Overview

In our implementation, we used the Hyperledger Aries framework to create a distributed FL architecture [27]. The communication between the participating entities takes place through the DIDComm transport protocol. We present a healthcare trust model in which each participant is in the form of a Docker container. Our architecture can be seen in Figure 2 and consists of three Hospitals, one Researcher, one NHS Trust that issues Hospitals' credentials, and a regulatory authority that issues the Researcher's credentials. The system's technical specifications are as follows: 3.2 GHZ 8th generation Intel Core i7 CPU, with 32 GB RAM and 512 GB SSD. Each Docker container functions as a Hyperledger Aries agent and built using the open-source Hyperledger Aries cloud agent in Python programming language, developed by the Verifiable Organizations Network (VON) team at the Province of British Columbia [113].

3.1. Establishing Trust

We create a domain-specific trust architecture by using DIDs and VCs issued by trusted participants, as presented in Section 3. Furthermore, during the connection establishment between the Hospitals and the Researcher, they need to follow a mutual authentication procedure, in which they present their issued credentials as proof that they are legitimate. The other party can then verify if the received credential has been issued by the public DID of the regulatory authority or the NHS Trust and approve the connection. The credential schema and the DIDs of the credential issuers are written to a public ledger; we used British Columbia VON's [114] development ledger.

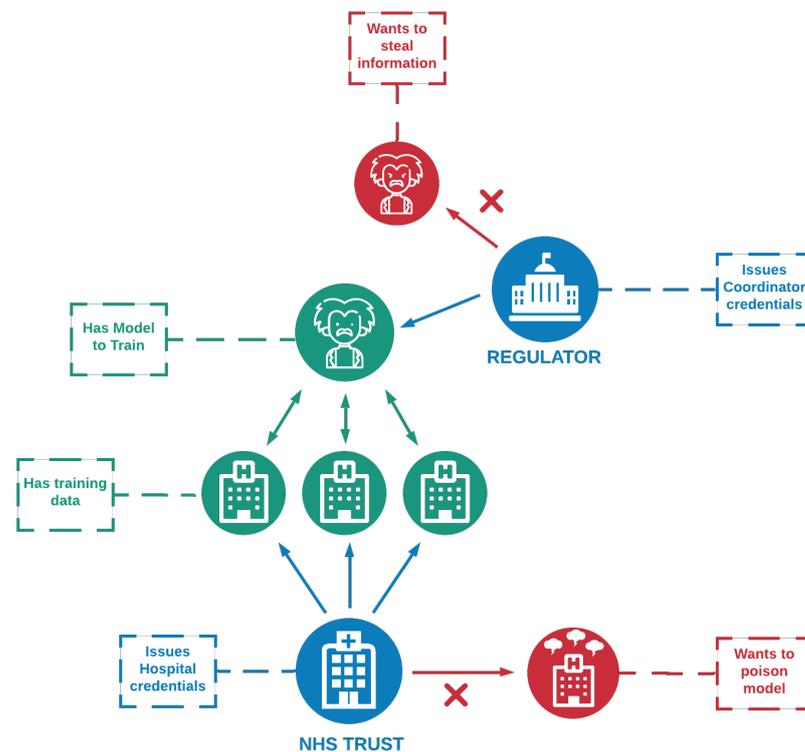


Figure 2. ML Healthcare Trust Model [27].

To create our testbed infrastructure, the steps described in Algorithm 2 followed. After the authentication is completed, the Researcher can initiate a FL mechanism, in which the ML model is being sent encrypted to each Hospital from the list of approved connections sequentially. Each Hospital trains this model using its mental health dataset and sends the model back to the Researcher. Then, the Researcher validates the trained model using its validation dataset and sends it to the next Hospital. The presented procedure continues until all the participants train the ML model. At the end of the training, the Researcher holds a ML model trained by multiple Hospitals that validated using its validation dataset to calculate the model's accuracy and loss. The model's parameters and updates are being sent using the DIDComm transport protocol, the security and performance evaluation of it have been presented and discussed in Section 4.

3.2. Communication Protocol

As presented in Section 3 and Figure 2, our implementation consists of three Hospitals, one Researcher that coordinates the training procedure, an NHS Trust and a Regulator that issue credentials for the Hospitals and the Researcher accordingly. Each participating entity is configured as a Docker container instead of a real-world situation in which each participant would run on their own network. Communication between entities only happens using the DIDComm protocol adding an authenticated encryption layer (see Algorithm 1) on top of the underlying transport protocol, in this case, Hypertext Transfer Protocol (HTTP) at a specified public port. Internally the entity can be represented as a controller and an agent; the controller sends HTTP requests to the agent defined by the admin-API. Received requests act as commands often resulting in the agent sending a DIDComm protocol message to an external agent, for example issuing a credential. Agents that receive a message from another entity post a webhook internally over HTTP, allowing the controller to respond appropriately. Note this can include requesting the agent to send further messages in reply. More details can be seen in Figure 3 and Table 1.

Algorithm 2 Establishing Trusted Connections [27]

- 1: *Researcher* agent exchanges DIDs with the *Regulator* agent to establish a DIDComm channel.
- 2: *Regulator* offers an *Audited Researcher-Coordinator* credential over this channel.
- 3: *Researcher* accepts and stores the credential in their wallet.
- 4: **for each** *Hospital* agent **do**
- 5: Initiate DID Exchange with *NHS Trust* agent to establish DIDComm channel.
- 6: *NHS Trust* offers *Verified Hospital* credentials over DIDComm.
- 7: *Hospital* accepts and stores the credential.
- 8: **end for**
- 9: **for each** *Hospital* agent **do**
- 10: *Hospital* initiates DID Exchange with *Researcher* to establish DIDComm channel.
- 11: *Researcher* requests proof of *Verified Hospital* credential issued and signed by the *NHS Trust*.
- 12: *Hospitals* generate a valid proof from their *Verified Hospital* credential and respond to the *Researcher*.
- 13: *Researcher* verifies the proof by first checking the DID against the known DID they have stored for the *NHS Trust*, then *resolve* the DID to locate the keys and verify the signature.
- 14: **if** *Hospitals* can prove they have a valid *Verified Hospital* credential **then**
- 15: *Researcher* adds the connection identifier to their list of *Trusted Connections*.
- 16: **end if**
- 17: *Hospital* requests proof of *Audited Researcher* credential from the *Researcher*.
- 18: *Researcher* uses *Audited Researcher* credential to generate a valid proof and responds.
- 19: *Hospital* verifies the proof, by checking the signature and DID of the Issuer.
- 20: **if** *Researcher* produces a valid proof of *Audited Researcher* **then**
- 21: *Hospital* saves connection identifier as a trusted connection.
- 22: **end if**
- 23: **end for**

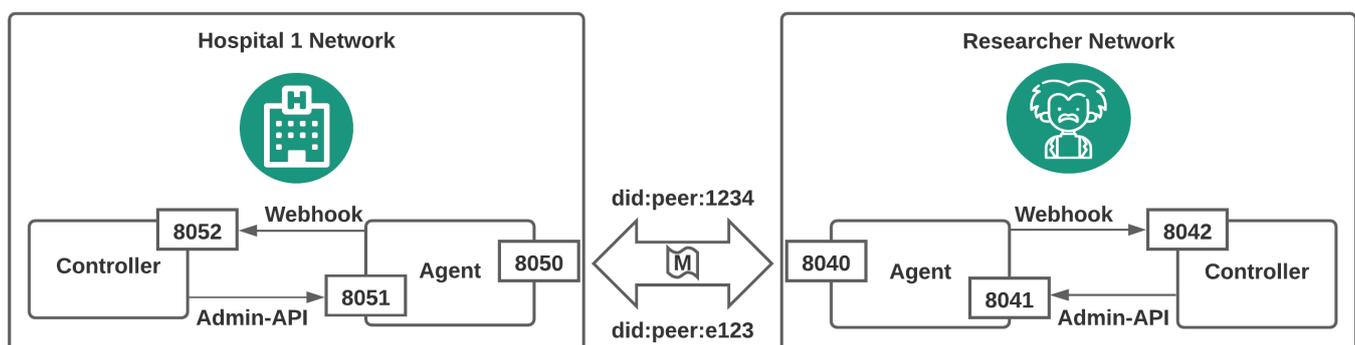


Figure 3. Networking communication architecture.

Table 1. Participating entities communication details.

Name	HTTP Port	Admin-API Port	Webhook Port
Hospital 1	8050	8051	8052
Hospital 2	8060	8061	8062
Hospital 3	8070	8071	8072
Researcher	8040	8041	8042
NHS Trust	8020	8021	8022
Regulator	8030	8031	8032

3.3. Federated Learning Procedure

The FL procedure described in our proof-of-concept is in its most basic form, in which the model and the updates are being sent sequentially to each trusted connection [15,115–117]. In a real-world scenario, this FL process would happen simultaneously, and the model updates would be sent to a secure aggregator to perform a Federated Averaging method [15] to improve the security of the system further. Before the training, the Researcher holds a ML model and a validation dataset, and each Hospital holds its own training dataset. The datasets are from an open-source mental health survey that “that measures attitudes towards mental health and frequency of mental health disorders in the tech workplace” [118], which are pre-processed into appropriate training data and validation data; the original dataset split into four partitions, three training datasets, one for each Hospital and one validation dataset for the Researcher.

Furthermore, we evaluated our infrastructure’s performance related to the model’s accuracy and measured the required resources. Our FL workflow can be seen in Algorithm 3. The focus of this paper is to demonstrate that FL is applicable over Hyperledger Aries agents through the DIDComm protocol in a trusted architecture scenario. Therefore, the ML procedure, classification and parameter-tuning are out of the scope of this paper.

However, the combination of these two emerging fields, private identities and FL, allowed us to mitigate a few existing FL limitations caused by the training participants’ lack of trust. Specifically, these were: (1) Training provided by a malevolent Hospital to corrupt the ML model’s accuracy, and (2) Malicious models being sent to legitimate Hospitals to leak information about the training data.

Algorithm 3 Our Federated Learning workflow [27]

- 1: *Researcher* has *validation* data and a *ML model*, *Hospitals* have *training* data.
 - 2: **while** *Hospitals* have not trained their *training* data **do**
 - 3: *Researcher* benchmarks the *model’s* performance against *validation* data and sends the *model* to the next *Hospital*.
 - 4: *Hospital* trains the *model* with their data and then sends the resulting *model* back to the *Researcher*.
 - 5: **end while**
 - 6: *Researcher* benchmarks the final *model* against *validation* data.
-

4. Evaluation

4.1. Security Evaluation

As presented in our implementation in Section 3, our testbed infrastructure achieves a domain-specific trust framework using verifiable credentials. Hence, the training process involves only authenticated Hospitals and Researchers that communicate through encrypted channels. Our work does not prevent the aforementioned attacks from happening; however, it minimises the possibility of occurring by establishing a trust framework among

the participants. Malicious entities could be checked on their registration to the system and removed on ill behaviour.

A potential threat in our test environment is the possibility of the participants' computer systems getting compromised. In such scenarios, the trusted credential issuers could create legitimate credentials to malicious participants, or a compromised hospital could corrupt the ML training; both scenarios lead to a malicious participant controlling a valid VC for the infrastructure. Another concern is the possibility that a compromised participant may try a Distributed Denial of Service (DDoS) attack [119], which can be mitigated by setting a timeout process within each participant, after several unsuccessful invitations. Several cybersecurity procedures could be in-place within the participants' computer systems that make security concerns and breaches unlikely. OWASP provides several secure practices and guidelines in order to mitigate cybersecurity threats [120]; hence, further defensive mechanisms could be used to extend further the security of the system, such as Intrusion Detection and Prevention Systems (IDPS) [121]. However, this type of attack is out of the scope of this paper.

To evaluate our proof-of-concept's security, we created malicious agents that attempt to take part in the ML procedure by connecting to one of the trusted credential issuers. Any agent without the appropriate VCs, either a verified Hospital or an audited Researcher credential, could not establish an authenticated channel with the other party, as seen in Figure 2. The unauthorised connection requests and the self-signed VC are automatically being rejected. The reason is because they had not been signed by a trusted authority whose DID was known by the entity requesting the proof. The mechanism of the mutual authentication of the VC between participants is not domain-specific to ML and can be expanded to any context [27].

However, our paper is focused on the trust establishment and FL in a distributed DID-based healthcare ecosystem. It is assumed that there is a governance-oriented framework in which the key-stakeholders have a DID written to an integrity-assured blockchain ledger. Identifying the appropriate DIDs and the participating entities related to a particular architecture is out-of-scope of this paper. This paper explores how peer DID connections facilitate participation in the established healthcare ecosystem. Another platform could be developed for the secure distribution of the DIDs between the participating agents [27].

4.1.1. Security Testing

In our implementation, the ML model and its updates are being sent to the Hospitals and the Researcher is using the DIDComm messaging protocol [51]. To verify these communication channels were encrypted, we used network packet sniffers such as the Wireshark and Tcpdump [122], to capture the traffic during the training procedure. As presented in Section 2.4, since the participating entities in our implementation take the form of Docker containers, the captured traffic obtained from a virtual network card in the host machine [63]. During the security testing of our implementation, we observed that when the participating entities connect and provide their proofs to the other party in order to authenticate, information such as the name of the participant, its DID and the provided proof are encrypted. Only in case that the Hyperledger Aries agent and controller reside within the same Docker container, then the information related to the connection establishment sent internally in plain *.json* format (Appendix A, Figure A1). However, this finding is irrelevant in production environments since each participant, its controller and agent would be physical machines or private networks and not Docker containers. We demonstrated this separation of controllers and agents, by executing each one in their own respective Jupyter Notebook [123], as described in Section 3.2.

Furthermore, other critical findings were observed during the training procedure. All the traffic was fully encrypted between all the parties, in each stage of the FL training (Appendix A, Figure A2). We used unsuccessfully various cybersecurity tools such as the Government Communications Headquarters (GCHQ) CyberChef [124] in order to reverse the encrypted content and obtain some information about the training data or the ML

model. That is re-assuring since the traffic related to the training procedure may contain information about the sensitive underlying training data.

4.2. Performance Evaluation

Performance evaluation metrics for each host were recorded during the operation of our workflow. Figure 4a) shows the CPU usage of each agent involved in the learning workflow. The CPU usage of the Researcher raises each time it sends the model to the Hospitals, and the CPU usage of the Hospitals raises when they train the model with their private data. This result is expected and follows the execution of Algorithm 3 successfully. The memory and network bandwidth follow a similar pattern, as it is illustrated in Figure 4b–d). The main difference is that since the Researcher averages and validates each model against the training dataset every time, the memory and network bandwidth increase over time. In these metrics, the ML training procedure transmitted through DIDComm protocols but does not use the designed federated learning libraries.

In Figure 5, we compared the FL training performance with and without the DIDComm protocol. Both architectures are identical, with their only difference in Figure 5a) where the Researcher sends the ML model to each Hospital sequentially through the DIDComm protocol, opposed to Figure 5b), in which the DIDComm protocol is not used, and the ML model is shared among the Hospitals to train it sequentially. We did not plot the memory and network metrics for this experiment since they follow the same pattern with negligible differences as in Figure 4.

Our work aims to demonstrate that since the proposed trust framework is distributed, it is possible to establish a FL workflow. Therefore, we do not focus on improving this FL process and tuning the hyperparameters for more reliable predictions, apart from developing the FL libraries designed for this purpose. The FL training procedure consists of the following hyperparameters: learning rate of 0.01 for 10 training epochs using one-third of the training dataset in batches of size 8. Moreover, we present the ML model's confusion matrix using the Researcher's validation data after each federated training batch, as shown in Tables 2 and 3. That confirms that our ML model was successfully trained at each stage using our distributed mental health dataset [118]. To calculate the model's accuracy, Equation (1) has been used. The two tables provide a comparison between two different activation functions. In Table 2, the Sigmoid linear activation function has been used, as opposed to Table 3, in which we implemented the Rectified Linear activation function (ReLU) [125,126].

Table 2. Classifier's accuracy without hyperparameters' optimisation over training batches using Sigmoid activation function on the original federated learning architecture of [27].

Batch	0	1	2	3
True Positives	0	109	120	134
False Positives	0	30	37	41
True Negatives	114	84	77	73
False Negatives	144	35	24	10
Accuracy	44.1%	74.8%	76.3%	80.2%

Table 3. Classifier's accuracy without hyperparameters' optimisation over training batches using ReLu activation function.

Batch	0	1	2	3
True Positives	144	121	121	108
False Positives	0	23	23	36
True Negatives	114	34	33	39
False Negatives	0	80	81	75
Accuracy	100%	60%	59.6%	57%

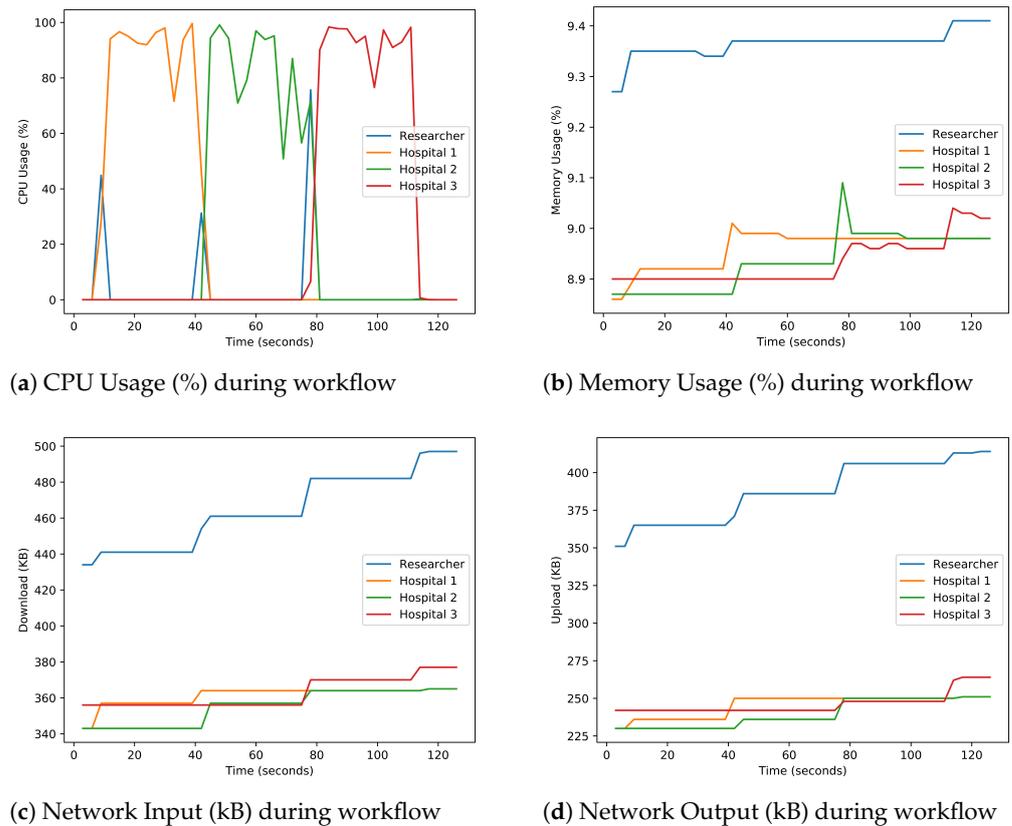


Figure 4. CPU, Memory usage and Network use of Docker container agents during workflow using the original federated learning architecture [27].

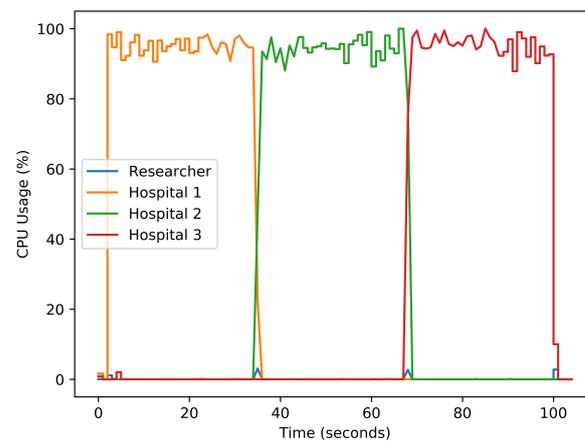
Moreover, we also compared the FL procedure’s accuracy when the ML model is being transmitted through and without the DIDComm protocol, and presented the results in Tables 4 and 5.

Table 4. Classifier’s accuracy without hyperparameters’ optimisation over training batches through the DIDComm protocol.

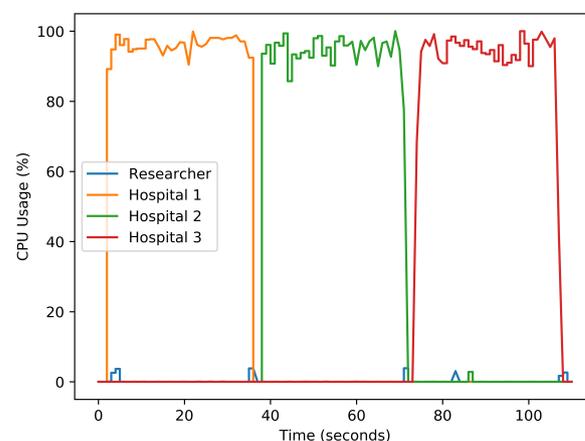
Batch	0	1	2	3
True Positives	0	115	120	135
False Positives	0	29	24	9
True Negatives	113	30	39	44
False Negatives	145	84	75	70
Accuracy	43.7%	56.2%	61.6%	69.3%

Table 5. Classifier’s accuracy without hyperparameters’ optimisation over training batches without the DIDComm protocol.

Batch	0	1	2	3
True Positives	0	113	120	133
False Positives	0	31	24	11
True Negatives	116	35	43	45
False Negatives	142	79	71	69
Accuracy	44.9%	57.3%	63.1%	69%



(a) CPU Usage (%) during workflow and transmission of the model through the DIDComm protocol



(b) CPU (%) during workflow without the use of the DIDComm protocol

Figure 5. CPU Usage comparison of Docker containers during workflow using our novel federated learning libraries.

5. Conclusions and Future Work

In this paper, we extended our previous work [27] by merging the privacy-preserving ML field with VCs and DIDs while addressing trust concerns within the data industry. These areas focus on people's security, privacy and especially on the protection of their sensitive data. In our work, we presented a trusted FL process for a mental health dataset distributed among hospitals. We proved that it is possible to use the established secure channels to obtain a digitally signed contract for ML training or manage pointer communications on remote data [17].

This extension of our previous work [27] retains the same high-level architecture of the participating entities, but the proof-of-concept is a complete refactor of our experimental setup. More specifically, as described in Section 4.1.1, each participant's controller and agent entities are separated into their own isolated Docker containers. That separation is adjacent to a real-world scenario in which each controller and agent reside in different systems (Appendix A). Furthermore, in our technical codebase, we now use our novel libraries written in Python programming language and demonstrated thoroughly using Jupyter notebooks. We further performed an extensive security and performance evaluation in each stage of our proposed infrastructure, which was lacking in our previous work, and our findings are presented in Section 4. The performance metrics identified that the performance of our trusted FL procedure and the accuracy of the ML model are similar,

albeit the model is transmitted through the encrypted DIDcomm protocol. Additionally, using our designed FL libraries, the ML training process completes faster. It should be noted that there are no conflicts with other defensive methods and techniques, and they could be incorporated into our framework and libraries.

While FL is vulnerable to attacks as described in Section 4.1, the purpose of this work is to develop a proof-of-concept for the demonstration that distributed ML can be achieved through the same encrypted communication channels used to establish domain-specific trust. We exhibited how this distributed trust framework could be used by other fields and not FL explicitly. This will allow the application of the trust framework to a wide range of privacy-preserving workflows. Additionally, it allows us to enforce trust, mitigating FL attacks using differentially private training mechanisms [84,85,88]. Various techniques can be incorporated to our framework in order to train a differentially private model; such as Opacus [127], PyDP [128], PyVacy [129] and LATENT [130]. To reduce the model stealing and training data inference risks, the SMPC can be leveraged to split data and model parameters into shares [131].

Our proof-of-concept detailed the established architecture between three hospitals, a researcher, a hospital trust and a regulatory authority. Firstly, the hospitals and the researcher need to obtain a VC from their corresponding trust or regulatory authority, and then follow a mutual authentication process in order to exchange information. Further, the researcher instantiates a basic FL procedure between only the authenticated and trusted hospitals, we refer to this process as Vanilla FL, and then transmits the ML model through the encrypted communication channels using Hyperledger Aries framework. Each hospital receives the model, trains it using their private dataset and sends it back to the researcher. The researcher validates the trained model using its validation dataset to calculate its accuracy. One of the limitations of this work is that the presented Vanilla FL process acts only as a proof-of-concept to demonstrate that FL is possible through the encrypted DIDComm channels. However, to incorporate it in a production environment, it should be extended and introduce a secure aggregator entity, placed in-between the researcher and the hospitals that would act as a mediator of the ML model and updates. In that production environment, the researcher entity would simultaneously send the ML model to all the authorised participants and not have a validation dataset. This is a crucial future improvement we need to undertake to help the research community further. Another potential limitation of our work is training a large-scale convolutional neural network, which left as out-of-scope, but needs to be tested.

Future work also includes integrating the Hyperledger Aries communication protocols, which enables the trust model demonstrated in this work, into an existing framework for facilitating distributed learning within the OpenMined open-source organisation such as PySyft, Duet and PyGrid [17,132,133]. Our focus is to extend the Hyperledger Aries functionalities, the libraries designed for ML communication, and distribute this framework as open-source to the academic and industrial community on the PyDentity project [31]. We hope that our work can motivate more people to work on the same subject. Additionally, our scope is to incorporate and evaluate further PPML techniques to create a fully trusted and secure environment for ML computations.

Author Contributions: All authors contributed to the manuscript's conceptualisation and methodology; P.P., W.A. and N.P. contributed in writing; P.P. performed the security evaluation of the proof-of-concept; W.A. developed the proof-of-concept's used libraries; A.J.H. performed the data preparation; W.J.B. reviewed and edited the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: The research leading to these results has been partially supported by the European Commission under the Horizon 2020 Program, through funding of the SANCUS project (G.A. n 952672).

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: <https://www.kaggle.com/osmi/mental-health-in-tech-survey> (accessed on 1 March 2021).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Security Testing

Traffic exchanged across DIDComm channel is always encrypted, as in Figure A2. Only the internal traffic during the connection establishment in a single network, such as information exchange between the hospital's agent and controller is not encrypted, as shown in Figure A1. This is not considered an issue since we separated the agent and controller entities to simulate a real-world scenario in which those entities are individual machines.

```
[{"cred_info": {"referent": "3e08c0e8-09dc-41cf-9ec7-5b8c8054b7aa", "attrs": {"hospital_name": "St Thomas's Hospital", "date": "2021-01-12"}, "schema_id": "DukExq9foGb5DjDoRXx8G8:2:Verified Hospital schema:74.42.54", "cred_def_id": "DukExq9foGb5DjDoRXx8G8:3:Cl:3945:default", "rev_reg_id": null, "cred_rev_id": null}, "interval": null, "presentation_referents": [{"0_hospital_name_uuid", "0_date_uuid"}]}]POST /present-proof/records/d0cc59b0-f98f-4858-a787-cd723b8aa984/send-presentation HTTP/1.1
Host: host.docker.internal:8071
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: Python/3.6 aiohttp/3.5.4
Content-Length: 267
Content-Type: application/json

{"requested_predicates": {}, "requested_attributes": {"0_date_uuid": {"cred_id": "3e08c0e8-09dc-41cf-9ec7-5b8c8054b7aa", "revealed": true}, "0_hospital_name_uuid": {"cred_id": "3e08c0e8-09dc-41cf-9ec7-5b8c8054b7aa", "revealed": true}}, "self_attested_attributes": {}}HTTP/1.1 200
OK
Content-Type: application/json; charset=utf-8
Content-Length: 5096
Date: Tue, 12 Jan 2021 18:47:46 GMT
Server: Python/3.6 aiohttp/3.5.4

{"thread_id": "6508fbf2-98bf-4a7f-874f-ca1588bcd8fd", "initiator": "external", "auto_present": false, "updated_at": "2021-01-12 18:47:46.561410Z", "presentation_exchange_id": "d0cc59b0-f98f-4858-a787-cd723b8aa984", "presentation": {"proof": {"proofs": [{"primary_proof": {"eq_proof": {"revealed_attrs": {"date": "10797520622553272758031075333733131562304135744127282393669404927495035677051", "hospital_name": "5517229649884854946323665860005879723745448741412972882555470653127216925176"}, "a_prime": "462932309907306734061587545784214176986800321119152431883072650845817709005638688342063183718206743447660548607351174147579969377485012591507676138428342446009784763934984963137207793653948356211749796624266891886998126228753469195129423833609761562369551503670142804841173552509478120969237142520913939204251800630253103783415957593367383005577302346753641753491141738785640367508803719204899553677848504889852818232506818541801610513593890213583483848597792553767894722716167945421267249794865320330922803279157058922953777001306485230528717092646745869951614133768010527844973083393704985188225507699623030238785671", "e": "49131171799499946080896670661109579789773132149050358000866132208228254676348657430199330532912259117277322709583645424849580907652340520", "v": "41184430801042665754312614844104569749019365626640249720723971329241104786222480300184027911621391783797622691126609738724470502155732954525963117890439522242271801143949611681665790363973859752923274026173727541524520046088413490231411559947952412113111364823147672602863090032519729800565223976656146953583475157254855375439472951659798891903555023079883428319476844678489619890112934965464951153340537727560027017233916011370309471600456712917485381214274225185238905593302781097593471390074438638217521139621971128629113908175110989215519277734345296820261208922924590590358503611513380176603846210550633463673791727190194009258757364647064271318940816374284539928694258886042982978350070682048284858001294945613598680169345195049795890985352992272540078493177870286359581944596318446008609675175309467103543021208490872763751943558989723980266988858265339114401365918601370669696997048538819449084863418127214471634", "m": {"master_secret": "9925636480664232046849756440879208360376325094518404125259013158940263183340633095887715240050566169540460181233211952100900521558066386450379886020549993398126387071521169809290"}, "m2": "6005135661828887604539811977575016432775086032082947073932461309100260722013300212420939226368151887650202612120903828127299799424361172700203417019299962970237551963499863887192"}, "ge_proofs": [], "non_revoc_proof": null}, "aggregated_proof": {"c_hash": "1713383664788883141719421710751591746514217336453438300625276164816228617095", "c_list": [[1, 110, 182, 136, 247, 18, 213, 22, 2, 246, 252, 115, 186, 17, 91, 222, 210, 57, 196, 73, 252, 222, 154, 116, 186, 163, 83, 225, 167, 176, 197, 102, 165, 221, 184, 105, 68, 125, 85, 16, 102, 208, 62, 25, 181, 205, 78, 202, 83, 98, 253, 235, 207, 143, 236, 223, 210, 241, 99, 205, 45, 217, 11, 99, 188, 21, 54, 113, 125, 103, 9, 77, 19, 214, 34, 4, 191, 169, 248, 238, 215, 166, 230, 181, 201, 205, 179, 62, 154, 96, 26, 5, 95, 171, 115, 14, 53, 233, 156, 220, 8, 37, 161, 221, 32, 80, 35, 160, 22, 132, 138, 113, 130, 133, 230, 38, 245, 53, 58, 13, 101, 13, 33, 251, 121, 162, 195, 206, 89, 4, 49, 25, 168, 122, 50, 96, 250, 132, 218, 132, 190, 14, 17, 16, 149, 62, 189, 25, 1, 77, 235, 250, 128, 23, 180, 174, 18, 69, 198, 15, 199, 53, 0, 89, 28, 209, 175, 131, 129, 221, 54, 133, 44, 59, 168, 140, 82, 193, 190, 155, 89, 221, 158, 221, 142, 95, 106, 90, 132, 48, 175, 117, 66, 110, 219, 253, 72, 161, 195, 117, 139, 115, 82, 16, 200, 190, 208, 145, 142, 30, 231, 145, 212, 140, 242, 104, 45, 43, 172, 43, 140, 127, 63, 78, 96, 168, 153, 216, 233, 137, 179, 28, 30, 29, 199, 253, 192, 194, 216, 114, 185, 72, 74, 171, 216, 233, 191, 72, 182, 253, 51, 255, 196, 252, 188, 176, 135]}]}, "requested_proof": {"revealed_attrs": {"0_hospital_name_uuid": {"sub_proof_index": 0, "raw": "St Thomas's Hospital", "encoded": "5517229649884854946323665860005879723745448741412972882555470653127216925176"}, "0_date_uuid": {"sub_proof_index": 0, "raw": "2021-01-12", "encoded": "10797520622553272758031075333733131562304135744127282393669404927495035677051"}}, "self_attested_attrs": {}, "unrevealed_attrs": {}, "predicates": {}, "identifiers": [{"schema_id": "DukExq9foGb5DjDoRXx8G8:2:Verified Hospital schema:74.42.54", "cred_def_id": "DukExq9foGb5DjDoRXx8G8:3:Cl:3945:default", "rev_reg_id": null, "timestamp": null}], "presentation_request": {"name": "Proof of Verified Hospital", "version": "1.0", "nonce": "231376535924841821213592488537406547647", "requested_attributes": {"0_date_uuid": {"name": "date", "restrictions": [{"issuer_id": "DukExq9foGb5DjDoRXx8G8"}]}, "0_hospital_name_uuid": {"name": "hospital_name", "restrictions": [{"issuer_id": "DukExq9foGb5DjDoRXx8G8"}]}}, "requested_predicates": {}, "state": "presentation_sent", "created_at": "2021-01-12 18:47:46.291136Z", "connection_id": "8c53ba0c-3faf-41e5-b9f8-2cca012cbb1a"}
```

Figure A1. Communication between agent and controller is not encrypted during the connection establishment.

9. Ahmad, O.F.; Stoyanov, D.; Lovat, L.B. Barriers and Pitfalls for Artificial Intelligence in Gastroenterology: Ethical and Regulatory issues. *Tech. Gastrointest. Endosc.* **2019**, *22*, 150636. [[CrossRef](#)]
10. Kairouz, P.; McMahan, H.B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A.N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. Advances and open problems in federated learning. *arXiv* **2019**, arXiv:1912.04977.
11. Coordinated by TECHNISCHE UNIVERSITÄT MÜNCHEN. FeatureCloud-Privacy Preserving Federated Machine Learning and Blockchaining for Reduced Cyber Risks in a World of Distributed Healthcare. 2019. Available online: <https://cordis.europa.eu/project/id/826078> (accessed on 1 March 2021).
12. Muñoz-González, L.; Biggio, B.; Demontis, A.; Paudice, A.; Wongrassamee, V.; Lupu, E.C.; Roli, F. Towards poisoning of deep learning algorithms with back-gradient optimization. In Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, Dallas, TX, USA, 3 November 2017; pp. 27–38.
13. Al-Rubaie, M.; Chang, J.M. Privacy-preserving machine learning: Threats and solutions. *IEEE Secur. Priv.* **2019**, *17*, 49–58. [[CrossRef](#)]
14. McMahan, H.B.; Moore, E.; Ramage, D.; Hampson, S.; Agüera y Arcas, B. Communication-efficient learning of deep networks from decentralized data. *arXiv* **2016**, arXiv:1602.05629.
15. Konečný, J.; McMahan, H.B.; Yu, F.X.; Richtárik, P.; Suresh, A.T.; Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv* **2016**, arXiv:1610.05492.
16. Bonawitz, K.; Eichner, H.; Grieskamp, W.; Huba, D.; Ingerman, A.; Ivanov, V.; Kiddon, C.; Konecny, J.; Mazzocchi, S.; McMahan, H.B.; et al. Towards federated learning at scale: System design. *arXiv* **2019**, arXiv:1902.01046.
17. Ryffel, T.; Trask, A.; Dahl, M.; Wagner, B.; Mancuso, J.; Rueckert, D.; Passerat-Palmbach, J. A generic framework for privacy preserving deep learning. *arXiv* **2018**, arXiv:1811.04017v2.
18. Fredrikson, M.; Lantz, E.; Jha, S.; Lin, S.; Page, D.; Ristenpart, T. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In Proceedings of the 23rd USENIX Security Symposium USENIX Security 14, San Diego, CA, USA, 20–24 August 2014; pp. 17–32.
19. Fredrikson, M.; Jha, S.; Ristenpart, T. Model inversion attacks that exploit confidence information and basic countermeasures. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015; pp. 1322–1333.
20. Bonawitz, K.; Ivanov, V.; Kreuter, B.; Marcedone, A.; McMahan, H.B.; Patel, S.; Ramage, D.; Segal, A.; Seth, K. Practical Secure Aggregation for Federated Learning on User-Held Data. *arXiv* **2016**, arXiv:1611.04482.
21. Song, C.; Ristenpart, T.; Shmatikov, V. Machine learning models that remember too much. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 587–601.
22. Shokri, R.; Stronati, M.; Song, C.; Shmatikov, V. Membership inference attacks against machine learning models. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2017; pp. 3–18.
23. Salem, A.; Zhang, Y.; Humbert, M.; Berrang, P.; Fritz, M.; Backes, M. ML-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv* **2018**, arXiv:1806.01246.
24. Bagdasaryan, E.; Veit, A.; Hua, Y.; Estrin, D.; Shmatikov, V. How to backdoor federated learning. *arXiv* **2018**, arXiv:1807.00459.
25. Bhagoji, A.N.; Chakraborty, S.; Mittal, P.; Calo, S. Analyzing federated learning through an adversarial lens. *arXiv* **2018**, arXiv:1811.12470.
26. Liu, Y.; Ma, S.; Aafer, Y.; Lee, W.C.; Zhai, J.; Wang, W.; Zhang, X. *Trojaning Attack on Neural Networks*; Purdue University Libraries e-Pubs: West Lafayette, IN, USA, 2017.
27. Abramson, W.; Hall, A.J.; Papadopoulos, P.; Pitropakis, N.; Buchanan, W.J. A Distributed Trust Framework for Privacy-Preserving Machine Learning. In *Trust, Privacy and Security in Digital Business*; Gritzalis, S., Weippl, E.R., Kotsis, G., Tjoa, A.M., Khalil, I., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 205–220.
28. Hyperledger. Hyperledger Aries. Available online: <https://www.hyperledger.org/projects/aries> (accessed on 1 March 2021).
29. Hyperledger. Hyperledger Indy. Available online: <https://www.hyperledger.org/use/hyperledger-indy> (accessed on 1 March 2021).
30. Hyperledger. Hyperledger Ursa. Available online: <https://www.hyperledger.org/use/ursa> (accessed on 1 March 2021).
31. OpenMined. PyDentity. 2020. Available online: <https://github.com/OpenMined/PyDentity> (accessed on 1 March 2021).
32. OpenMined. PyDentity-Aries FL Project. 2020. Available online: <https://github.com/OpenMined/PyDentity/tree/master/projects/aries-fl> (accessed on 1 March 2021).
33. Camenisch, J.; Lysyanskaya, A. A signature scheme with efficient protocols. In *International Conference on Security in Communication Networks*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 268–289.
34. Camenisch, J.; Dubovitskaya, M.; Lehmann, A.; Neven, G.; Paquin, C.; Preiss, F.S. Concepts and languages for privacy-preserving attribute-based authentication. In *IFIP Working Conference on Policies and Research in Identity Management*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 34–52.
35. Reed, D.; Sporny, M.; Longely, D.; Allen, C.; Sabadello, M.; Grant, R. Decentralized Identifiers (DIDs) v1.0. 2020. Available online: <https://w3c.github.io/did-core/> (accessed on 1 March 2021).
36. Sporny, M.; Longely, D.; Chadwick, D. Verifiable Credentials Data Model 1.0. Technical Report; W3C. 2019. Available online: <https://www.w3.org/TR/2019/REC-vc-data-model-20191119/> (accessed on 1 March 2021).

37. Hardman, D. DID Communication. Github Requests for Comments. 2019. Available online: <https://github.com/hyperledger/aries-rfcs/tree/master/concepts/0005-didcomm> (accessed on 1 March 2021).
38. Yeh, C.L. Pursuing consumer empowerment in the age of big data: A comprehensive regulatory framework for data brokers. *Telecommun. Policy* **2018**, *42*, 282–292. [CrossRef]
39. Voigt, P.; Von dem Bussche, A. The eu general data protection regulation (gdpr). In *A Practical Guide*, 1st ed.; Springer International Publishing: Cham, Switzerland, 2017.
40. Young, K.; Greenberg, S. A Field Guide to Internet Trust. 2014. Available online: <https://identitywoman.net/wp-content/uploads/TrustModelFieldGuideFinal-1.pdf> (accessed on 1 March 2021).
41. Hoffman, A.M. A conceptualization of trust in international relations. *Eur. J. Int. Relat.* **2002**, *8*, 375–401. [CrossRef]
42. Keymolen, E. Trust on the Line: A philosophical Exploration of Trust in the Networked Era. 2016. Available online: <http://hdl.handle.net/1765/93210> (accessed on 1 March 2021).
43. Powles, J.; Hodson, H. Google DeepMind and healthcare in an age of algorithms. *Health Technol.* **2017**, *7*, 351–367. [CrossRef]
44. Hughes, O. Royal Free: ‘No Changes to Data-Sharing’ as Google Absorbs Streams. 2018. Available online: <https://www.digitalhealth.net/2018/11/royal-free-data-sharing-google-deepmind-streams/> (accessed on 1 March 2021).
45. Denham, E. *Royal Free-Google DeepMind Trial Failed to Comply with Data Protection Law*; Technical Report; Information Commissioner Office: Cheshire, UK, 2017.
46. De Fauw, J.; Keane, P.; Tomasev, N.; Visentin, D.; van den Driessche, G.; Johnson, M.; Hughes, C.O.; Chu, C.; Ledsam, J.; Back, T.; et al. Automated analysis of retinal imaging using machine learning techniques for computer vision. *F1000Research* **2016**, *5*, 1573. [CrossRef]
47. Chu, C.; De Fauw, J.; Tomasev, N.; Paredes, B.R.; Hughes, C.; Ledsam, J.; Back, T.; Montgomery, H.; Rees, G.; Raine, R.; et al. Applying machine learning to automated segmentation of head and neck tumour volumes and organs at risk on radiotherapy planning CT and MRI scans. *F1000Research* **2016**, *5*, 2104. [CrossRef]
48. W3C Credential Community Group. DID Method Registry. Technical Report. 2019. Available online: <https://w3c-ccg.github.io/did-method-registry/> (accessed on 1 March 2021).
49. Hardman, D. Peer DID Method Specification. Technical Report. 2019. Available online: <https://openssi.github.io/peer-did-method-spec/index.html> (accessed on 1 March 2021).
50. Chaum, D.L. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM* **1981**, *24*, 84–90. [CrossRef]
51. Terbu, O. DIF Starts DIDComm Working Group. 2020. Available online: <https://medium.com/decentralized-identity/dif-starts-didcomm-working-group-9c114d9308dc> (accessed on 1 March 2021).
52. ElGamal, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory* **1985**, *31*, 469–472. [CrossRef]
53. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
54. Wohlwend, J. Elliptic Curve Cryptography: Pre and Post Quantum. Technical Report; MIT Tech. Rep. 2016. Available online: https://math.mit.edu/~apost/courses/18.204-2016/18.204_Jeremy_Wohlwend_final_paper.pdf (accessed on 1 March 2021).
55. Camenisch, J.; Lysyanskaya, A. A Signature Scheme with Efficient Protocols. In *Security in Communication Networks*; Goos, G., Hartmanis, J., van Leeuwen, J., Cimato, S., Persiano, G., Galdi, C., Eds.; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2576, pp. 268–289. [CrossRef]
56. Longley, D.; Sporny, M.; Allen, C. Linked Data Signatures 1.0. Technical Report. 2019. Available online: <https://w3c-dvcg.github.io/ld-signatures/> (accessed on 1 March 2021).
57. Jones, M.; Bradley, J.; Sakimura, N. JSON Web Signatures. Rfc. 2015. Available online: <https://tools.ietf.org/html/rfc7515> (accessed on 1 March 2021).
58. Chaum, D. Security without identification: Transaction systems to make big brother obsolete. *Commun. ACM* **1985**, *28*, 1030–1044. [CrossRef]
59. Davie, M.; Gisolfi, D.; Hardman, D.; Jordan, J.; O’Donnell, D.; Reed, D. The Trust Over IP Stack. RFC 289, Hyperledger. 2019. Available online: <https://github.com/hyperledger/aries-rfcs/tree/master/concepts/0289-toip-stack> (accessed on 1 March 2021).
60. Au, M.H.; Tsang, P.P.; Susilo, W.; Mu, Y. Dynamic Universal Accumulators for DDH Groups and Their Application to Attribute-Based Anonymous Credential Systems. In *Topics in Cryptology—CT-RSA 2009*; Fischlin, M., Ed.; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5473, pp. 295–308.
61. Boettiger, C. An introduction to Docker for reproducible research. *ACM SIGOPS Oper. Syst. Rev.* **2015**, *49*, 71–79. [CrossRef]
62. Smith, J.E.; Nair, R. The architecture of virtual machines. *Computer* **2005**, *38*, 32–38. [CrossRef]
63. Martin, A.; Raponi, S.; Combe, T.; Di Pietro, R. Docker ecosystem—Vulnerability analysis. *Comput. Commun.* **2018**, *122*, 30–43. [CrossRef]
64. Kholod, I.; Yanaki, E.; Fomichev, D.; Shalugin, E.; Novikova, E.; Filippov, E.; Nordlund, M. Open-Source Federated Learning Frameworks for IoT: A Comparative Review and Analysis. *Sensors* **2021**, *21*, 167. [CrossRef]
65. Das, D.; Avancha, S.; Mudigere, D.; Vaidynathan, K.; Sridharan, S.; Kalamkar, D.; Kaul, B.; Dubey, P. Distributed deep learning using synchronous stochastic gradient descent. *arXiv* **2016**, arXiv:1602.06709.
66. Tharwat, A. Classification assessment methods. *Appl. Comput. Inform.* **2020**, *17*, 168–192. [CrossRef]

67. Shah, S.A.R.; Issac, B. Performance comparison of intrusion detection systems and application of machine learning to Snort system. *Future Gener. Comput. Syst.* **2018**, *80*, 157–170. [[CrossRef](#)]
68. Buchanan, W.J.; Imran, M.A.; Rehman, M.U.; Zhang, L.; Abbasi, Q.H.; Chrysoulas, C.; Haynes, D.; Pitropakis, N.; Papadopoulos, P. Review and critical analysis of privacy-preserving infection tracking and contact tracing. *Front. Commun. Netw.* **2020**, *1*, 2.
69. Zhang, Y.; Jia, R.; Pei, H.; Wang, W.; Li, B.; Song, D. The Secret Revealer: Generative Model-Inversion Attacks Against Deep Neural Networks. *arXiv* **2019**, arXiv:1911.07135.
70. Tramèr, F.; Zhang, F.; Juels, A.; Reiter, M.K.; Ristenpart, T. Stealing machine learning models via prediction apis. In Proceedings of the 25th {USENIX} Security Symposium ({USENIX} Security 16, Austin, TX, USA, 10–12 August 2016; pp. 601–618.
71. Nuding, F.; Mayer, R. Poisoning attacks in federated learning: An evaluation on traffic sign classification. In Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy, New Orleans, LA, USA, 16–18 March 2020; pp. 168–170.
72. Sun, G.; Cong, Y.; Dong, J.; Wang, Q.; Liu, J. Data Poisoning Attacks on Federated Machine Learning. *arXiv* **2020**, arXiv:2004.10020.
73. Jagielski, M.; Oprea, A.; Biggio, B.; Liu, C.; Nita-Rotaru, C.; Li, B. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In Proceedings of the 2018 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 20–24 May 2018; pp. 19–35.
74. Biggio, B.; Nelson, B.; Laskov, P. Poisoning attacks against support vector machines. *arXiv* **2012**, arXiv:1206.6389.
75. Laishram, R.; Phoha, V.V. Curie: A method for protecting SVM Classifier from Poisoning Attack. *arXiv* **2016**, arXiv:1606.01584.
76. Steinhardt, J.; Koh, P.W.W.; Liang, P.S. Certified defenses for data poisoning attacks. *arXiv* **2017**, arXiv:1706.03691.
77. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
78. Carlini, N.; Wagner, D. Towards evaluating the robustness of neural networks. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (sp), San Jose, CA, USA, 22–26 May 2017; pp. 39–57.
79. Chen, J.; Jordan, M.I.; Wainwright, M.J. Hopscijumpattack: A query-efficient decision-based attack. *arXiv* **2019**, arXiv:1904.02144.
80. Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z.B.; Swami, A. Practical black-box attacks against machine learning. In Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, Abu Dhabi, United Arab Emirates, 2–6 April 2017; pp. 506–519.
81. Yuan, X.; He, P.; Zhu, Q.; Li, X. Adversarial examples: Attacks and defenses for deep learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 2805–2824. [[CrossRef](#)] [[PubMed](#)]
82. Pitropakis, N.; Panaousis, E.; Giannetsos, T.; Anastasiadis, E.; Loukas, G. A taxonomy and survey of attacks against machine learning. *Comput. Sci. Rev.* **2019**, *34*, 100199. [[CrossRef](#)]
83. Dwork, C. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 1–19.
84. Dwork, C. Differential privacy. In *Encyclopedia of Cryptography and Security*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 338–340.
85. Abadi, M.; Chu, A.; Goodfellow, I.; McMahan, H.B.; Mironov, I.; Talwar, K.; Zhang, L. Deep learning with differential privacy. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 308–318.
86. McMahan, H.B.; Andrew, G.; Erlingsson, U.; Chien, S.; Mironov, I.; Papernot, N.; Kairouz, P. A general approach to adding differential privacy to iterative training procedures. *arXiv* **2018**, arXiv:1812.06210.
87. Dwork, C.; Roth, A. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.* **2014**, *9*, 211–407. [[CrossRef](#)]
88. Mironov, I. Rényi differential privacy. In Proceedings of the 2017 IEEE 30th Computer Security Foundations Symposium (CSF), Santa Barbara, CA, USA, 21–25 August 2017; pp. 263–275.
89. Goldreich, O. Secure multi-party computation. *Manuscript. Prelim. Version* **1998**, *78*. Available online: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.11.2201&rep=rep1&type=pdf> (accessed on 1 March 2021).
90. Fontaine, C.; Galand, F. A survey of homomorphic encryption for nonspecialists. *EURASIP J. Inf. Secur.* **2007**, *2007*, 013801. [[CrossRef](#)]
91. Gentry, C. Fully homomorphic encryption using ideal lattices. In Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing, Bethesda, MD, USA, 31 May–2 June 2009; pp. 169–178.
92. Bost, R.; Popa, R.A.; Tu, S.; Goldwasser, S. Machine learning classification over encrypted data. *NDSS* **2015**, 4324, 4325.
93. Zhang, L.; Zheng, Y.; Kantoa, R. A review of homomorphic encryption and its applications. In Proceedings of the 9th EAI International Conference on Mobile Multimedia Communications, Xi'an, China, 18–19 June 2016; pp. 97–106.
94. Sathya, S.S.; Vepakomma, P.; Raskar, R.; Ramachandra, R.; Bhattacharya, S. A review of homomorphic encryption libraries for secure computation. *arXiv* **2018**, arXiv:1812.02428.
95. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531.
96. Papernot, N.; McDaniel, P.; Wu, X.; Jha, S.; Swami, A. Distillation as a defense to adversarial perturbations against deep neural networks. In Proceedings of the 2016 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2016; pp. 582–597.
97. Gürses, S.; del Alamo, J.M. Privacy engineering: Shaping an emerging field of research and practice. *IEEE Secur. Priv.* **2016**, *14*, 40–46. [[CrossRef](#)]

98. Franke, M.; Gladbach, M.; Sehili, Z.; Rohde, F.; Rahm, E. ScaDS research on scalable privacy-preserving record linkage. *Datenbank-Spektrum* **2019**, *19*, 31–40. [[CrossRef](#)]
99. Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; McDaniel, P. Ensemble adversarial training: Attacks and defenses. *arXiv* **2017**, arXiv:1705.07204.
100. Rubinstein, B.I.; Nelson, B.; Huang, L.; Joseph, A.D.; Lau, S.h.; Rao, S.; Taft, N.; Tygar, J.D. Antidote: Understanding and defending against poisoning of anomaly detectors. In Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement, Chicago, IL, USA, 4–6 November 2009; pp. 1–14.
101. Chen, B.; Carvalho, W.; Baracaldo, N.; Ludwig, H.; Edwards, B.; Lee, T.; Molloy, I.; Srivastava, B. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv* **2018**, arXiv:1811.03728.
102. Liu, K.; Dolan-Gavitt, B.; Garg, S. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 273–294.
103. Gao, Y.; Xu, C.; Wang, D.; Chen, S.; Ranasinghe, D.C.; Nepal, S. Strip: A defence against trojan attacks on deep neural networks. In Proceedings of the 35th Annual Computer Security Applications Conference, San Juan, PR, USA, 9–13 December 2019; pp. 113–125.
104. Stamatellis, C.; Papadopoulos, P.; Pitropakis, N.; Katsikas, S.; Buchanan, W.J. A Privacy-Preserving Healthcare Framework Using Hyperledger Fabric. *Sensors* **2020**, *20*, 6587. [[CrossRef](#)] [[PubMed](#)]
105. Papadopoulos, P.; Pitropakis, N.; Buchanan, W.J.; Lo, O.; Katsikas, S. Privacy-Preserving Passive DNS. *Computers* **2020**, *9*, 64. [[CrossRef](#)]
106. Dachman-Soled, D.; Malkin, T.; Raykova, M.; Yung, M. Efficient robust private set intersection. In *International Conference on Applied Cryptography and Network Security*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 125–142.
107. Angelou, N.; Benaissa, A.; Cebere, B.; Clark, W.; Hall, A.J.; Hoeh, M.A.; Liu, D.; Papadopoulos, P.; Roehm, R.; Sandmann, R.; et al. Asymmetric Private Set Intersection with Applications to Contact Tracing and Private Vertical Federated Machine Learning. *arXiv* **2020**, arXiv:2011.09350.
108. Abramson, W.; van Deursen, N.E.; Buchanan, W.J. Trust-by-Design: Evaluating Issues and Perceptions within Clinical Passporting. *arXiv* **2020**, arXiv:2006.14864.
109. Camenisch, J.; Lysyanskaya, A. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *International Conference on the Theory and Applications of Cryptographic Techniques*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 93–118.
110. Bichsel, P.; Camenisch, J.; Dubovitskaya, M.; Enderlein, R.; Krenn, S.; Krontiris, I.; Lehmann, A.; Neven, G.; Nielsen, J.D.; Paquin, C.; et al. D2. 2 Architecture for Attribute-Based Credential Technologies-Final Version. *ABC4TRUST Project Deliverable*. 2014. Available online: <https://abc4trust.eu/index.php/pub> (accessed on 1 March 2021).
111. Dunphy, P.; Petitcolas, F.A. A first look at identity management schemes on the blockchain. *IEEE Secur. Priv.* **2018**, *16*, 20–29. [[CrossRef](#)]
112. Wang, F.; De Filippi, P. Self-sovereign identity in a globalized world: Credentials-based identity systems as a driver for economic inclusion. *Front. Blockchain* **2020**, *2*, 28. [[CrossRef](#)]
113. Hyperledger. Hyperledger Aries Cloud Agent-Python. 2019. Available online: <https://github.com/hyperledger/aries-cloudagent-python> (accessed on 1 March 2021).
114. Government of British Columbia. British Columbia’s Verifiable Organizations. 2018. Available online: <https://orgbook.gov.bc.ca/en/home> (accessed on 1 March 2021).
115. Nishio, T.; Yonetani, R. Client selection for federated learning with heterogeneous resources in mobile edge. In Proceedings of the ICC 2019-2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–7.
116. Shoham, N.; Avidor, T.; Keren, A.; Israel, N.; Benditkis, D.; Mor-Yosef, L.; Zeitak, I. Overcoming forgetting in federated learning on non-iid data. *arXiv* **2019**, arXiv:1910.07796.
117. Koppurapu, K.; Lin, E. FedFMC: Sequential Efficient Federated Learning on Non-iid Data. *arXiv* **2020**, arXiv:2006.10937.
118. Open Sourcing Mental Illness, LTD. Mental Health in Tech Survey-Survey on Mental Health in the Tech Workplace in 2014. 2016. Available online: <https://www.kaggle.com/osmi/mental-health-in-tech-survey> (accessed on 1 March 2021).
119. Lau, F.; Rubin, S.H.; Smith, M.H.; Trajkovic, L. Distributed denial of service attacks. In Proceedings of the Smc 2000 Conference Proceedings, 2000 IEEE International Conference on Systems, Man and Cybernetics, ‘Cybernetics Evolving to Systems, Humans, Organizations, and Their Complex Interactions’ (Cat. No. 0), Nashville, TN, USA, 8–11 October 2000; Volume 3, pp. 2275–2280.
120. OWASP. TOP 10 2017. *Ten Most Crit. Web Appl. Secur. Risks. Release Candidate* **2018**, *2*. Available online: <https://owasp.org/www-project-top-ten/> (accessed on 1 March 2021).
121. Hall, P. Proposals for Model Vulnerability and Security. 2019. Available online: <https://www.oreilly.com/ideas/proposals-for-model-vulnerability-and-security> (accessed on 1 March 2021).
122. Goyal, P.; Goyal, A. Comparative study of two most popular packet sniffing tools-Tcpdump and Wireshark. In Proceedings of the 2017 9th International Conference on Computational Intelligence and Communication Networks (CICN), Cyprus, Turkey, 16–17 September 2017; pp. 77–81.
123. Kluyver, T.; Ragan-Kelley, B.; Pérez, F.; Granger, B.E.; Bussonnier, M.; Frederic, J.; Kelley, K.; Hamrick, J.B.; Grout, J.; Corlay, S.; et al. *Jupyter Notebooks—a Publishing Format for Reproducible Computational Workflows*; IOS Press: Amsterdam, The Netherlands, 2016; Volume 2016, pp. 87–90.

124. GCHQ. CyberChef—The Cyber Swiss Army Knife. 2020. Available online: <https://gchq.github.io/CyberChef/> (accessed on 1 March 2021).
125. Agostinelli, F.; Hoffman, M.; Sadowski, P.; Baldi, P. Learning activation functions to improve deep neural networks. *arXiv* **2014**, arXiv:1412.6830.
126. Nwankpa, C.; Ijomah, W.; Gachagan, A.; Marshall, S. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv* **2018**, arXiv:1811.03378.
127. PyTorch. Opacus. 2020. Available online: <https://github.com/pytorch/opacus> (accessed on 1 March 2021).
128. OpenMined. PyDP. 2020. Available online: <https://github.com/OpenMined/PyDP> (accessed on 1 March 2021).
129. Waites, C. PyVacy: Privacy Algorithms for PyTorch. 2019. Available online: <https://pypi.org/project/pyvacy/> (accessed on 1 March 2021).
130. Chamikara, M.; Bertok, P.; Khalil, I.; Liu, D.; Camtepe, S. Local differential privacy for deep learning. *arXiv* **2019**, arXiv:1908.02997.
131. Lindell, Y. Secure multiparty computation for privacy preserving data mining. In *Encyclopedia of Data Warehousing and Mining*; IGI Global: Hershey, PA, USA, 2005; pp. 1005–1009.
132. OpenMined. PySyft. 2020. Available online: <https://github.com/OpenMined/PySyft> (accessed on 1 March 2021).
133. OpenMined. Duet. 2020. Available online: <https://github.com/OpenMined/PySyft/tree/master/examples/duet> (accessed on 1 March 2021).