TEE based Cross-silo Trustworthy Federated Learning Infrastructure

Wei Yu, Qingqing Li, Dan He, Hao Zhu, Jingyu Hu
Zhiqiang Li, Lejun Zhu, Xiaoping Duan, Chaoqing Zhao
Intel China Ltd.
wei.w.yu@intel.com

Abstract

This paper introduces a TEE based (using Intel SGX) cross-silo trustworthy federated learning infrastructure, which is an excellent pipeline protection including runtime environment and all data (input/output/intermedia data) protection in all parties. It is shown that our implemented infrastructure in TEE is both efficient and flexible. In our implemented infrastructure, AI frameworks are running on libOS Gramine, in order to minimize the change of the applications. This implemented infrastructure can be modified and adopted for commercial usage (please contact author to get codes under NDA).

1 Introduction

Federated Learning (FL) [McMahan and Ramage, 2017] proposed by Google is devised to train machine learning (ML)/deep learning (DL) models without requiring data sharing. However, it still has a few challenges.

Firstly, it cannot always guarantee the privacy of user data due to the existence of a malicious user who purposely tries to steal the data from other users. [Bagdasaryan et al., 2020, Sun et al., 2020, Wang et al., 2020, Xie et al., 2019] install backdoors to the FL model during the model parameter communication, and the resulting model will make an incorrect prediction with specific input data. [Tolpegin et al., 2020, Zhou et al., 2021, Fang et al., 2020, Zhang et al., 2019] study poisoning attack to the FL model by using poisoning training data and poisoning model. Another type of attack to ML model, named inference attack is investigated in [Nasr et al., 2019, Gao et al. 2021], including membership inference attack and category inference attack. [Luo et al. 2021] conducts a feature inference attack to vertical FL model using generative networks. [Li et al. 2021] studies a general framework improving both the fairness and robustness of the FL model against poisoning attack. For backdoor and inference attack, methods based on clipping and smoothing on model parameters [Xie et al., 2021], feedback [Sebastien et al., 2021], model clustering weight clipping [Nguyen et al., 2021] and mixing neural network layers [Antoine et al., 2021] are proposed. Other important techniques used in FL are differential privacy and homomorphic encryption [Liu et al., 2021, Liu et al., 2020, Wei et al., 2020, Stacey et al., 2020, Liu and Yang et al., 2020].

Secondly, the accuracy and efficiency of the pure software-based approaches needs to be improved. The more secure we want; the heavier the workload is. The pure software-based approaches have a much higher computation cost compared to the plain text computation [Naehrig et al., 2011]. The usage of noise in differential privacy also has an adverse effect on the accuracy of the model. So, FL accuracy and efficiency are practical issues to industry projects.

1.1 Related work on TEE

In addition to the above software-based approaches, privacy-preserving FL leveraging trusted execution environments (TEEs) has been proposed recently. TEE is a secure hardware technique for confidential computing on untrusted environment. One well-known application of TEE technique is Intel® Software Guard Extensions (Intel® SGX). SGX is an Intel technology for application developers seeking to protect code and data selected from disclosure or modification. It allows user-level code to allocate private regions of memory, called enclaves, which are designed to be protected from processes even running at higher privilege levels.
[Mo et al., 2021] proposes a practical framework based on greedy layer-wise training and aggregation on SGX, overcoming the constraints posed by the limited TEE memory. [Zhang et al., 2021] provides a scalable collaborative learning system in untrusted infrastructures by distributing the training across multiple SGX enclaves. To protect the gradient during the training, [Fumiyuki et al., 2021] studies a new scheme for differentially private FL. It uses SGX to ensure secure model aggregation on an untrusted server, and the transfer of gradients of models are encrypted. [Zhang et al., 2021] adopts a random grouping algorithm on SGX against the side-channel attack and reduces the probability that the adversaries obtain the gradient information. Other TEE-based approaches for privacy-preserving FL are proposed in [Eugene et al., 2021, Quoc et al., 2021]. OpenFL [Reina et al., 2021] only uses TEE/SGX for running programs so far but not to verify each other with attestation.

1.2 Our work

We implemented a TEE based Cross-silo Trustworthy Federated Learning Infrastructure to help address the above two challenges based on SGX. As SGX enclave size of the 3rd Gen Xeon Scalable processor increased up to 1TB, we can re-design the FL infrastructure for FL workload.

Firstly, the implemented FL infrastructure protects the whole pipeline. All the data and programs are under TEE protection even when it is in runtime environment. Unlike other implementations with TEE/SGX SDK, lib OS Gramine is used to help minimize the change of applications in the infrastructure. The complex software like popular AI frameworks TensorFlow or PyTroch etc. can run on TEE/SGX with defining manifest files only in this way. Meanwhile, all enclaves do remote/local attestation to prove themselves which prevents malicious or even malicious-collude to data/programs in enclaves on the fly. SGX has an integrity check of the loaded data/programs, so the infrastructure is more trustworthy. Within our FL infrastructure, a trusted third party is NOT needed as descripted in Section 2. A logical but protected aggregator by TEE can be deployed to any party.

Secondly, the efficiency of the implemented infrastructure is very high. The efficiency defined as TEE/SGX runtime vs plain text runtime is usually >50% and could be as high as 80~90% which is much higher than purely software-based solutions and it will be discussed in detail in Section 3.

As our target is to build an industry reference infrastructure, a simple Key Management Service (KMS) is also embedded with a GUI interface which is sufficient to most general applications. All operations are logged and trackable. This data can be saved in a secure environment like blockchain. Meanwhile, this TEE-based infrastructure could be used for many works of multiple party collaborations other than FL.

2 Trustworthy Federated Learning with SGX

In this section, we introduce the infrastructure to enable trustworthy federated learning over multiple organizations with the help of SGX technology, so that the data providers can secure their data from illegal use.

The federated learning cluster is composed of two types of workers, governor workers and compute workers, both secured by SGX, which are distributed over each organization, see Figure 1. The governor worker is to attest the compute workers within the local organization as well as governor workers across the other organizations and manage the secured communications across the compute workers (of different organizations). The compute worker is to run the core business, namely the federated learning process. Throughout this paper, the compute worker is run within Gramine libOS, where we can run the regular application within the SGX enclave with little efforts. The governor worker can be run within the Gramine or be developed with SGX SDK.

The remainder of this section is organized as follows. Section 2.1 will introduce the remote attestation process of the governor worker, Section 2.2 will describe the startup of compute worker, and Section 2.3 will provide introduction to the federated learning process.

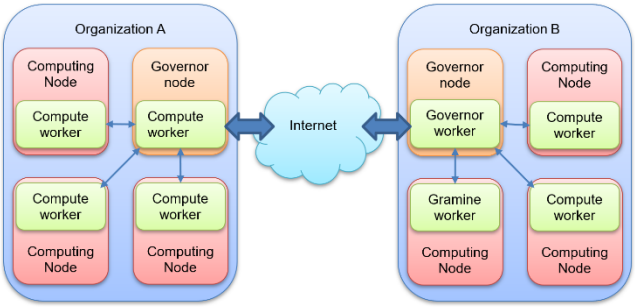

Figure 1 Confidential Collaborative ML Framework

2.1 The remote attestation process

Once a governor worker is started up, it should perform a bi-directional peer-to-peer remote attestation with the governor workers from the other organizations. In consequence, the governor workers from all the organizations form a trusted cluster.

To be concrete, when a governor worker gets started, it should upload its metadata to the storage service. Here a storage service can be any kind service which can be accessed by all the organizations. In our infrastructure, we use the Fabric block chain as a storage service, so that we do not rely on any other third party. The metadata may contain the following information:

- the organization id;
- the id of the governor worker;
- the randomly generated encryption and verification keys of the governor worker;
- the SGX report structure of the governor worker;
- the address of the governor worker.

After uploading the metadata to the storage service, the governor worker will pull the list of governor workers of the other organizations from the storage service and perform the peer-to-peer attestation one by one. The detail of the attestation process is given in Algorithm 1.

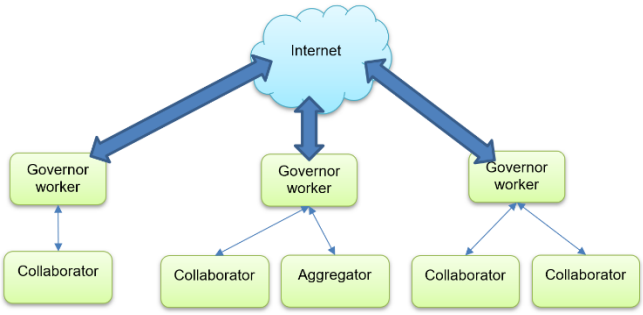

Figure 2 Horizontal federated learning

2.2 The startup of the compute worker

Algorithm 1 Peer-to-peer remote attestation

Input: worker_id //the id of the peer worker
Parameter: measures // the list of trusted measures
Output: true if the attestation passes and false otherwise

```
1:  Let nonce = randomly generated nonce;
2: // Ask the peer worker to generate the SGX quote
3:  Let quote = generate_quote(worker_id, nonce);
4:  //check whether the quote is generated by hardware
5:  if  verify_quote(quote) == false:
6:      return false;
7:  endif
8:  Let worker = get_worker(worker_id);
9:  // Calculate the report data
10: Let report_data = generate_report_data(
11:                              worker, nonce);
12: //check whether the calculated report data matches
13: // the report data in sgx quote
14: if  report_data != get_report_data(quote):
15:     return false;
16: endif
17: if get_measure(quote) not in measures:
18:     return false;
19: endif
20: return true;
```

All the compute workers within one organization are managed by the local governor worker. Once a compute worker is started, it should register itself to the governor worker. To achieve this, the compute worker should send the metadata detailed in Section 2.1 to the governor worker, and then a bi-directional peer-to-peer attestation is performed between the governor worker and the compute worker. If the registration succeeds, the governor worker should upload the metadata of the compute worker to the storage service, so that the other organizations can be aware of it.

Note that the attestations among the compute workers are not needed. Since the compute worker trusts the governor worker, it should also trust the other compute workers attested by the governor worker. By this method, we can add and remove the compute workers within the cluster easily. All we need to do is to update the list of attested compute workers within the governor worker. Moreover, since the encryption keys and verifications keys are generated randomly, and protected by SGX, we can update the keys of the SGX workers without the help of Certificate Authority (CA) when those keys are expired.

The communications among the SGX workers can be conducted over untrusted network. The randomly generated keys of the SGX workers help protect privacy.

The compute workers within the same organization can talk to one another directly. However, when two compute workers from different organizations want to talk, the packets should be redirected by the governor workers. Thus, the governor workers act as virtual routers of the network. To be specific, each packet should be attached with the work id of the compute worker. When the local governor worker receives the packet, it will check the corresponding governor worker of the destinate compute worker and redirect the packet.

2.3 The process of federated learning

In this subsection, we provide an infrastructure to deploy the horizontal federated learning process over the cluster depicted in Figure 1. Note that this does not mean our infrastructure can only be used for horizontal federated learning. In fact, the Gramine libOS has provided us unlimited possibility, and we can run all kinds of machine learning tasks within it.

We provide a decentralized federated learning protocol in this subsection, which means that the aggregator can be selected randomly from the joined compute workers, so that we do not rely on any trusted third party. The whole procedure can be divided into the following steps.

1. **Create a workflow.** When some organizations want to do a federated learning job, the administrator of one organization should create a workflow by the governor worker, and then the governor worker will reveal it to the other governor workers. Afterwards, the administrator from each organization adds the corresponding compute workers to the workflow.

2. **Select an aggregator.** Now the compute workers within the workflow can talk to one another by the method introduced in Section 2.2. Before they really do learning job, they should select an aggregator first. In our demo, the aggregator is selected by the workflow owner manually. However, we can select the aggregator by a random way. For example, each compute worker can generate a random number, and the one with the minimal random number can be chosen as the aggregator. Since the binary of the compute worker is well verified by remote attestation, this simple random method is secure enough. Once the aggregator is selected, the other compute workers will be regarded as collaborators, see Figure 2.

3. **Start the learning job.** Once the aggregator is selected, the process of the following learning job is straightforward. The other compute workers will act as collaborators, and they compute the gradients of the neural network based on the local datasets and sent the gradients to aggregators to iterate the global neural network.

3 Evaluations

In this section we present evaluation results of our infrastructure. We implemented 3 workloads within our infrastructure, i.e., ResNet Collaborative Machine Learning (ResNet CML), 3D-Unet horizontal federated machine learning (3D-Unet FML), and Bert-Base horizontal federated machine learning

(Bert-Base FML). We evaluated the workloads in our infrastructure from three perspectives: correctness, performance, and security. For correctness, we run ResNet CML on Linux system, in Gramine without SGX (Use Gramine in the following), and in Gramine with SGX (Use Gramine-SGX in the following)[1] respectively to compare their final loss and accuracy. The executions either on Linux or in Gramine are in plaintext, while the execution in Gramine SGX is protected by hardware encryption. For performance, we measured the time cost of Resnet CML, 3D-Unet FML and Bert-Base FML in different environments. For security, we simulated the attacks by grabbing the training status and model weights in the memory space of Bert-Base FML running in multiple parties.

3.1 Implementation

ResNet CML. In this workload, we simulate 2 parties with 2 nodes, a data owner node, and a requester node. The data owner provides local dataset and the requester requests dataset from data owner to do ResNet training. The dataset is CIFAR-10 which consists of 60000 32x32 colored images in 10 classes. In this workload, we use 10000 images of them to do the training with Batch size 32. We run the training until convergence.

3D-Unet FML. In this workload, we simulate 3 parties with 3 nodes, 2 collaborator nodes and an aggregator node. The 2 collaborators have different training datasets in local, and the aggregator has a testing dataset in local. Each dataset contains 100 622x529 3D colored images. The 2 collaborators do Unet local training first, then send the intermediate results to the aggregator, after that, the aggregator aggregates the results and replies them back to the 2 collaborators for update. This process is repeated for 10 rounds, batch size is 6 and the number of epochs is set to 5 for performance evaluation.

Bert-Base FML. In this workload, we simulate 3 parties with 3 nodes, 2 collaborator nodes and an aggregator node. This workload builds a model to deal with the named-entity recognition task in Chinese Biomedical language understanding. Given a pre-trained schema, the task is to identify and extract entities from the given sentence and classify them into nine categories: disease, clinical manifestations, drugs, etc. The training dataset is CBLUE [Zhang, 2021] and we divided the dataset into two parts randomly so that 2 collaborators have different training datasets in local, and the aggregator has a testing dataset in local. The total dataset consists of 15000 training sentences and 5000 validation sentences and 3000 test sentences. Test data are only saved in the aggregator local to verify the training result. The training process is the same as 3D-Unet except the 2 collaborators load the pre-trained bert-base model [Cui Y, 2021]. This process is repeated for 10 rounds, batch size is 16 and the number of epochs is set to 5 for performance evaluation. During the local training process, we simulated the attack to verify the effectiveness of protection.

3.2 Experimental Setup

We create several Virtual Machines (VM)[2] with SGX feature enabled on one Xeon(R) Platinum 8358 CPU @ 2.60GHz host machine[3]. For ResNet CML, we use 2 VMs and for 3D-Unet FML, we use 3 VMs. Both governor workers and compute workers are running in docker containers[4]. SGX version is 2.15.1 and SGX Data Center Attestation Primitives (DCAP) version is 1.12.1.

3.3 Results

Correctness. As shown in Table 1, whether ResNet CML workload is running in plaintext, in Gramine without SGX, or in Gramine with SGX, the differences in the final loss and accuracy are very small. Therefore, our infrastructure will not cause much impact on the correctness of the workload results.

Workload	loss	acc
ResNet CML (Linux)	1.8927	0.3707
ResNet CML (Gramine)	1.8926	0.3713
ResNet CML (Gramine-SGX)	1.8914	0.3701

Table 1: Loss and acc of Resnet CML

Performance. Table 2 denotes the time cost of ResNet CML, 3D-Unet FML and Bert-Base FML workloads when running in different environments. For Resnet CML, the efficiency of running in Gramine is about 81% of running in plaintext, and that of running in SGX is about 60% of running in plaintext. For 3D-Unet FML, the efficiency of running in Gramine is about 92% of running in plaintext, and that of running in SGX is about 74% of running in plaintext. For Bert-Base FML, the efficiency of running in Gramine is about 73% of running in plaintext, and that of running in SGX is about 61% of running

Workload / Environment	Resnet CML	3D-Unet FML	Bert-Base FML
Linux	51	65	126
Gramine	63	71	172
Gramine-SGX	86	88	207

Table 2: Time cost (s) of workloads

[1] Gramine can run the application in library OS alone or in library OS with SGX.

[2] Each VM has 4 vCPUs, 64GB memory, 40GB disk storage and 16GB EPC size, and is with OS Ubuntu 18.04 and Linux kernel 5.11.

[3] The host has 128 CPU cores on 2 sockets, 256G memory, 1TB disk storage and 64GB EPC size per socket, and is with OS CentOS 8.4 and Linux kernel 5.15.

[4] We use Avalon [Hyperledger 2020] with commitid cf762fd to implement governor worker and Gramine [C. che Tsai, et al 2017] with version 1.1 to implement compute worker. Docker version is 20.10.14 and docker-compose version is 1.24.1.

in plaintext. Therefore, the efficiency of our infrastructure is higher than 50% as shown in Figure 3.

Security. We dump the memory during the training process on the collaborator node and aggregator locally**.** When running in Gramine direct, all data like training status, training data, model weights, can be easily got from the dumped memory. While nothing will be exhibited when running in SGX.

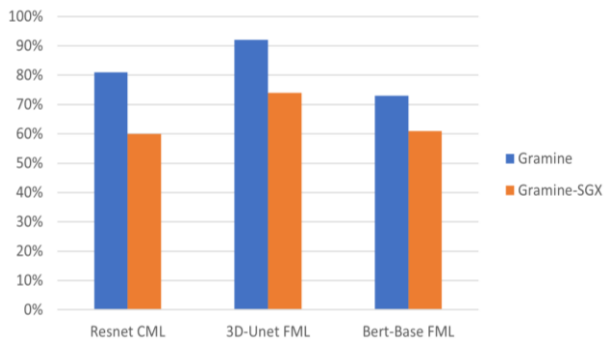

Figure3: Efficiency of workloads (Higher is better)

4 Conclusion

We implemented a TEE based cross-silo trustworthy federated learning infrastructure. It has the following major advantages:

1. An excellent pipeline protection;
2. High efficiency. The efficiency of our tested workload is at least 50%;
3. Easy deployed (docker supported/Setup GUI embedded).

So far, no federated learning infrastructure can be 100% secure. If want to promote the security, more confidential computing techniques could be used together. TEE based FL is not conflicted with other confidential computing techniques like MPC or HE etc. They can be combined to the infrastructure to generate a better solution for a dedicated task.

The limitations of the implemented infrastructure are:
1. Need CPU with SGX [McKeen 2013] features, i.e. 3rd gen Xeon SP or later;
2. So far, the infrastructure does not support heterogeneous computing platform.

Our future works include:
1. Add disaster recovery module into our infrastructure to add/remove nodes with a primary node (like recovery from block chain etc.);
2. So far, the implemented infrastructure does attestation when start. To be more secure, the infrastructure can add a timer to do attestation according to the customer settings.
3. The overall performance could be further improved like the optimization of network communication.

So far, the source codes and docker images of this infrastructure will be shared under NDA. Open source will be scheduled according to market's feedback. A FL infrastructure demo video in Chinese can be viewed in https://ccechina.intel.cn/air/LoadTest.

References

[McMahan and Ramage, 2017] Federated learning: Collaborative machine learning without centralized training data. Google Research Blog 3.

[Bagdasaryan et al., 2020] Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D. and Shmatikov, V., June. How to backdoor federated learning. In International Conference on Artificial Intelligence and Statistics (pp. 2938-2948). PMLR, 2020.

[Sun et al., 2020] Sun, Ziteng, Peter Kairouz, Ananda Theertha Suresh, and H. Brendan McMahan. Can you really backdoor federated learning? arXiv preprint arXiv:1911.07963 (2019).

[Wang et al., 2020] Wang, Hongyi, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. Advances in Neural Information Processing Systems 33 (2020): 16070-16084.

[Xie et al., 2019] Xie, Chulin, Keli Huang, Pin-Yu Chen, and Bo Li. Dba: Distributed backdoor attacks against federated learning. In International Conference on Learning Representations. 2019.

[Tolpegin et al., 2020] Tolpegin, V., Truex, S., Gursoy, M.E. and Liu, L.. Data poisoning attacks against federated learning systems. In European Symposium on Research in Computer Security (pp. 480-501). Springer, Cham.

[Zhou et al., 2021] Zhou, Xingchen, Ming Xu, Yiming Wu, and Ning Zheng. Deep model poisoning attack on federated learning. Future Internet 13, no. 3 (2021): 73.

[Fang et al., 2020] Fang, Minghong, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local Model Poisoning Attacks to {Byzantine-Robust} Federated Learning. In 29th USENIX Security Symposium (USENIX Security 20), pp. 1605-1622. 2020.

[Zhang et al., 2019]Zhang, Jiale, Junjun Chen, Di Wu, Bing Chen, and Shui Yu. Poisoning attack in federated learning using generative adversarial nets. In 2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), pp. 374-380. IEEE, 2019.

[Nasr et al., 2019] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In SP, pages739–753, 2019.

[Gao et al., 2021] Gao, Jiqiang, Boyu Hou, Xiaojie Guo, Zheli Liu, Ying Zhang, Kai Chen, and Jin Li. Secure Aggregation is Insecure: Category Inference Attack on Federated Learning. IEEE Transactions on Dependable and Secure Computing, 2021.

[Luo et al., 2021] Luo, Xinjian, Yuncheng Wu, Xiaokui Xiao, and Beng Chin Ooi. Feature inference attack on model predictions in vertical federated learning. In 2021 IEEE 37th International Conference on Data Engineering (ICDE), pp. 181-192. IEEE, 2021.

[Li et al., 2021] Li, Tian, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In International Conference on Machine Learning, pp. 6357-6368. PMLR, 2021.

[Sebastien et al., 2021] Andreina, Sebastien, Giorgia Azzurra Marson, Helen Möllering, and Ghassan Karame. Baffle: Backdoor detection via feedback-based federated learning. In 2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS), pp. 852-863. IEEE, 2021.

[Xie et al., 2021] Xie, Chulin, Minghao Chen, Pin-Yu Chen, and Bo Li. Crfl: Certifiably robust federated learning against backdoor attacks. In International Conference on Machine Learning, pp. 11372-11382. PMLR, 2021.

[Nguyen et al., 2021] Nguyen, Thien Duc, Phillip Rieger, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen et al. "FLGUARD: secure and private federated learning." arXiv preprint arXiv:2101.02281 (2021).

[Antoine et al., 2021] Boutet, Antoine, Thomas Lebrun, Jan Aalmoes, and Adrien Baud. "MixNN: Protection of Federated Learning Against Inference Attacks by Mixing Neural Network Layers." arXiv preprint arXiv:2109.12550 (2021).

[Liu et al., 2021] Ruixuan Liu, Yang Cao, Hong Chen, Ruoyang Guo, and Masatoshi Yoshikawa. 2020. Flame: Differentially private federated learning in the shuffle model. In AAAI.

[Liu et al., 2020] Ruixuan Liu, Yang Cao, Masatoshi Yoshikawa, and Hong Chen. 2020. Fedsel: Federated sgd under local differential privacy with top-k dimension selection. In International Conference on Database Systems for Advanced Applications. Springer, 485–501.

[Wei et al., 2020] Wei, Kang, Jun Li, Ming Ding, Chuan Ma, Howard H. Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H. Vincent Poor. "Federated learning with differential privacy: Algorithms and performance analysis." IEEE Transactions on Information Forensics and Security 15 (2020): 3454-3469.

[Stacey et al., 2020] Truex, Stacey, Ling Liu, Ka-Ho Chow, Mehmet Emre Gursoy, and Wenqi Wei. LDP-Fed: Federated learning with local differential privacy. In Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking, pp. 61-66. 2020.

[Liu and Yang et al., 2020] Liu, Yang, Yan Kang, Chaoping Xing, Tianjian Chen, and Qiang Yang. "A secure federated transfer learning framework." IEEE Intelligent Systems 35, no. 4 (2020): 70-82.

[Naehrig et al., 2011] Naehrig, Michael, Kristin Lauter, and Vinod Vaikuntanathan. "Can homomorphic encryption be practical?" In Proceedings of the 3rd ACM workshop on Cloud computing security workshop, pp. 113-124. 2011.

[Mo et al., 2021] Mo, Fan, Hamed Haddadi, Kleomenis Katevas, Eduard Marin, Diego Perino, and Nicolas Kourtellis. PPFL: privacy-preserving federated learning with trusted execution environments. In Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services, pp. 94-108. 2021.

[Zhang et al., 2021] Zhang, Chengliang, Junzhe Xia, Baichen Yang, Huancheng Puyang, Wei Wang, Ruichuan Chen, Istemi Ekin Akkus, Paarijaat Aditya, and Feng Yan. Citadel: Protecting Data Privacy and Model Confidentiality for Collaborative Learning. In Proceedings of the ACM Symposium on Cloud Computing, pp. 546-561. 2021.

[Fumiyuki et al., 2021] Kato, Fumiyuki, Yang Cao, and Masatoshi Yoshikawa. OLIVE: Oblivious and Differentially Private Federated Learning on Trusted Execution Environment. arXiv preprint arXiv:2202.07165 (2022).

[Zhang et al., 2021] Zhang, Yuhui, Zhiwei Wang, Jiangfeng Cao, Rui Hou, and Dan Meng. "ShuffleFL: gradient-preserving federated learning using trusted execution environment." In Proceedings of the 18th ACM International Conference on Computing Frontiers, pp. 161-168. 2021.

[Eugene et al., 2021] Kuznetsov, Eugene, Yitao Chen, and Ming Zhao. "SecureFL: Privacy Preserving Fed-erated Learning with SGX and TrustZone." In 2021 IEEE/ACM Symposium on Edge Computing (SEC), pp. 55-67. IEEE, 2021.

[Quoc et al., 2021] Quoc, Do Le, and Christof Fetzer. "SecFL: Confidential Federated Learning using TEEs." arXiv preprint arXiv:2110.00981 (2021).

[Reina et al., 2021] Reina, G. A. , Gruzdev, A. , Foley, P. , Perepelkina, O. , Sharma, M. , & Davidyuk, I. , et al. (2021). Openfl: an open-source framework for federated learning.

[Zhang, 2021] Zhang, Ningyu, et al. "CBLUE: A Chinese Biomedical Language Understanding Evaluation Benchmark." arXiv:2106.08087 (2021).

[Cui Y, 2021] Cui Y, Che W, Liu T, et al. Pre-training with whole word masking for Chinese BERT[J]. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2021, 29: 3504-3514.

[C. che Tsai, et al 2017] C. che Tsai, D. E. Porter, and M. Vij, Graphene-sgx: A practical library OS for unmodified applications on SGX," in 2017 USENIX Annual Technical Conference (USENIX ATC 17), (Santa Clara, CA), pp. 645{658, USENIX Association, July 2017

[Hyperledger 2020] Hyperledger: Hyperledger Avalon (2020), https://www.hyperledger.org/use/avalon, last accessed on 24-07-2020