Anonymous and Efficient Authentication Scheme for Privacy-Preserving Distributed Learning

Subhajit Ghosh

Roll: 2102041

This report is submitted for the mini project

Department of Computer Science and Engineering
Indian Institute of Information Technology Guwahati

October 2022

Abstract

This report provides an overview of the anonymous and efficient authentication scheme for privacy-preserving distributed learning. This work has helped to alleviate some of the drawbacks of privacy-preserving in distributed learning or has innovated new technologies for various applications. I attempted to identify the paper's problem(s) while also comprehending the methodology (that is, the working model and algorithms).

Table of Contents

1. Introduction
2. Proposed Methodology
3. Results
4. Problem found in the methodology
5. Conclusions

Chapter 1

Introduction

A tremendous amount of data has been generated and transmitted to cloud servers due to the integration of machine learning (ML) and cloud computing, which has been developed rapidly in the past decade. It leads to significant communication and computation overheads.

To resolve this issue, the concept of distributed learning (DL) is used.

In this, the participating devices first train the ML model locally with the raw data, and once computed, it sends the model parameters to the cloud. The cloud then aggregates into a global ML model. Since there is no need to upload the user data to the cloud, the risk of privacy leakage is reduced in distributed learning (DL).

DL is still subject to privacy violations when the cloud server is not fully trusted. In distributed scenarios, it shows that the cloud is able to recover the input of an ML model with the knowledge of full or partial model parameters. Once the raw data is recovered, the personal information of the participants is disclosed. At the cloud server side, achieving secure parameter aggregation without revealing the raw parameters is a challenge in DL.

To tackle this challenge, deploying encryption techniques or differential privacy (DP) are two primary approaches.

1) **Encryption techniques**, such as homomorphic encryption and secure multiparty computation, are employed to design secure data exchange and aggregation protocols. Benefiting from the protocols, the cloud is able to calculate the global model without accessing the model parameters of individual participants.

Cons:

- Encryption techniques is not generalised for all ML algorithms in DL.
- It also consumes a lot of communication and computation resources because it uses a lot of cryptographic primitives.

2) **Differential privacy (DP)** is deployed to obfuscate information by introducing random noise to the raw data or model parameters. DP-based schemes are generalised for all ML algorithms in DL.

Cons:

- The randomness reduces data utility, bringing undesired accuracy degradation to ML models.
- Model convergence with longer latency.
- DP is ineffective for accuracy-sensitive situations.

To overcome the existing cons, the anonymous and efficient authentication scheme for privacy-preserving distributed learning is used, which is generalised for all ML algorithms without data reduction.

The proposed scheme is based on the fact that if the participants in the DL system are anonymous, then although the raw data is recovered, the adversary cannot link the data to the corresponding participant's real identity, thereby achieving privacy preservation.

- In this scheme, it deploys anonymous authentication. It is effective for all ML algorithms without reducing data utility. It can also be integrated with the defence schemes to detect poisoning attacks and free-rider attacks.
- The new generalised certificateless signature scheme PCLS, which is built on pairing-based cryptography, The designed signature scheme has an anonymous and efficient authentication protocol that supports batch verification (AAPBV).
- An algorithm of dynamic batch verification (DBV) which can dynamically adjust the batch size during batch verification. In conventional batch verification, the batch size is not adjustable. All the messages are discarded if the batch verification fails. When the failure occurs, DBV is launched to adjust the batch size and redo verification.

The major problems addressed in this paper are:

- Adversary is able to derive the raw data (user identity) by analysing the obtained machine learning models.
- Significantly reduces the time consumption in batch verification, achieving high computational efficiency.

Apart from this, the proposed protocol address:

- Confidentiality
- Anonymity
- Mutal Authentication
- Unlinkability
- Unforgery
- Forward Security
- Backward Security
- Non-repudiation

Chapter 2

Proposed Methodology

2.1 System Overview

- **Participants:** It refer to individuals that are equipped with intelligent devices, such as laptop, smart phone, tablet, and so on.
- **Cloud Server:** It receives the model parameters from participants and aggregates into a global ML model
- **Trusted Authority (TA):** TA is a powerful system manager that is responsible for system initialization, participant registrations, key generation, identity management, and so on.

2.2 Model Flow

- Step 1: The cloud server initializes a global model and distributes the model to all participants.
- Step 2: After receiving the global model, each participant continues to train the model on its local dataset and obtains the updated model parameters W_n. W_n is then uploaded to the cloud server.
- Step 3: The cloud server aggregates the model parameters. The global ML model is corrected with W_g and sent to the participants for next-round training.
- Step 4: Repeat Steps 2, 3 until the global model converges.

2.3 Pairing based certificateless signature scheme (PCLS)

PCLS, there are two main modifications compared with the typical certificateless signature scheme.

- Instead of setting the signer's real identity as the partial public key, the hash value of the signer's real identity is the partial public key, protecting the real identities from disclosure.
- Modified the signature generation and message verification to support batch verification, such that the authentication is more efficient.

The PCLS scheme involves the following steps:

- **Setup:**
  Let G1 denote a cyclic additive group of a prime order p.
  G2 denotes a cyclic multiplicative group with the same prime order p.
  p satisfies $p > 2l$, where l is the security parameter.
  P is a generator of G1 and $e: G1 \times G1 \rightarrow G2$ is a bilinear pairing operator.

TA randomly selects $X_{TA} \in Z^* p$ as its private key and computes $Y_{TA} = X_{TA} P$ as its public key.
TA picks two hash functions H1 and H2, where
H1: $\{0, 1\}^* \rightarrow G1$
H2: $\{0, 1\}^* \times G1 \rightarrow Z^* p$

TA publishes (l, p, P, e, G1, G2, H1, H2, YT A) as the system parameters.
Private key X_{TA} is kept by TA and remains as a secret.

- **Key-Extract:**
  The signer randomly selects $X^{'} \in Z^{*}p$ as its partial private key and calculates $Y^{'} = X^{'} P$ as its partial public key.
  The signer computes $Y^{''} = H1(I D)$ as the other partial public key and sends $Y^{''}$ to TA, where ID is its true identity.
  Given $Y^{''}$, TA then sets $X^{''} = X_{TA} Y^{''}$ and sends to the signer through a secure channel. The signer uses $(X^{'}, X^{''})$ as its full private key and $(Y, Y^{''})$ as its full public key.

- **Sign:**
  Before sending a message m, the signer randomly chooses $r \in Z^* p$ and signs the message as follows:
  $k = H2(m \| Y^{'} \| Y^{''} \| Y_{TA}, r P)$
  $U = X^{'} Y_{TA} - X^{''} k$
  The signature is $\sigma = (k, U)$ on message m.

- **Verify:**
  After receiving the message m with signature σ, the verifier checks
  $e(Y' , Y_{TA}) = e(U, P)e(Y^{''}, Y_{TA})^k$

  Proof –
  $e(U, P)e(Y'', Y_{TA})^k$
  $= e(U, P)e(Y^{''}, X_{TA} P)^k$
  $= e(U, P)e(X^{''}, P)^k$
  $= e(U + k X^{''}, P)$
  $= e(X' Y_{TA}, P)$
  $= e(Y_{TA}, Y^{'})$
  $= e(Y' , Y_{TA})$

  The message is accepted if the checking holds.

2.4 Anonymous authentication protocol AAPBV

The proposed AAPBV involves four stages, including initialization, registration, authentication, and batch verification.

In the **initialization stage**, the TA initializes the system parameters and publishes to the participants and the cloud.

In the **registration stage**, a participant registers to TA with its real identity and generates full public key and private key assisted by TA.

In the **authentication stage**, a participant generates a valid signature, signs the message, and uploads to the cloud. The cloud then authenticates the participant and decrypts the message. To improve the computational efficiency in large-scale DL, the cloud can authenticate a batch of messages at the same time.

In the **batch verification stage**, the batch verification may fail if there are illegal participants who upload messages signed by invalid signatures. In this case, the cloud launches the DBV algorithm to dynamically shrink the batch size and redo verification. Thus, the illegitimate messages may be filtered. In this way, the number of discarded messages is reduced and more legal messages are saved for machine learning.

- Initialization stage –
  TA initializes the system and generates the system parameters given the security parameter l.
  Let G1 denote a cyclic additive group of a prime order p > 2l.
  G2 denotes a cyclic multiplicative group with the same prime order p.
  P is a generator of G1 and e: G1 × G1 → G2 is a bilinear pairing operator.
  TA picks three secure hash functions H1, H2, and H3,
  where H1: {0, 1}* → G1,
  H2: {0, 1}*×G1 → Z* p,
  H3 : {0, 1}* → Z* p.
  TA randomly selects $X_{TA} \in Z^*$ p as its private key and computes $Y_{TA} = X_{TA}$ P as its public key.
  The private key X_{TA} is kept by TA and remains a secret. (l, p, P, e, G1, G2, H1, H2, H3, Y_{TA}) is published as the system parameters.

- Registration stage-
  Cloud and participants both need to register to the TA.
  Cloud
  1. It randomly selects $X'_c \in Z^*p$ as its partial private key.
  2. Calculates $Y'_c = X'$ as its partial public key.
  3. It sends its identity ID_c to the TA via a secure channel.

TA then computes $Y''_c = H1(I\,D)$, $X''_c = X_{TA}Y''_c$ and returns key pair (X''_c , Y''_c) to the cloud. (ID_c, X''_c , Y''_c) is stored in the record of TA.

The cloud publishes (Y'_c, Y''_c) as its full public key and keeps (X'_c, X''_c) as its full private key.

Similarly the participant $U_n(n \in [1, N])$, computes its full public and private key.

The participant U_n publishes (Y'_n, Y''_n) as its full public key and secretly keeps (X'_n, X''_n) as its full private key.

- Authentication stage-
  Participant U_n, before uploading the updated model parameters W_n to the cloud, it generates the signature on and signs W_n as follows.
  1. Randomly pick $r, v \in Z^*p$.
  2. Calculate $D_n = H2(W_n \| Y'_n \| Y''_n \| Y'_c)$.
  3. Calculate $k_n = H2(W_n \| Y'_n \| Y''_n \| Y'_c, r P)$.
  4. Calculate $U_n = X'_n Y_{TA} - X''_n k_n$.
  5. $\sigma_n = (k_n, U_n)$.
  6. Calculate $Z_n = vP$, $Z'_n = vY'_c$.
  7. Calculate $W^*_n = Enc_S(W_n \| Y'_n \| Y''_n \| Y'_c, Zn)$
  8. Send the update request message $req_n = (W^*_n, \sigma_n, k_n, U_n, D_n, Z'_n)$ to the cloud.

  After receiving the request req_n, the cloud authenticate the message as follows

  1. Recover $Z_n = X'^{-1}_c Z'_n$.
  2. Calculate $W_n \| Y'_n \| Y''_n \| Y'_c = Dec_S(W^*_n, Z_n)$.
  3. Check $D_n = H2(W_n \| Y'_n \| Y''_n \| Y'_c)$. If the equation holds, continue to the next step or authentication failed and stop it.
  4. Check the validity of σ_n following the Verify step in the proposed PCLS. It rejects the request if σ_n is invalid. Otherwise, continue to the next step.
  5. The cloud establishes a session key $S^1_n = H3(D_n \| W_n \| Z_n)$ for the next first session. For the subsequent sessions, the session key is calculated as $S^{j+1}_n = H3(S^j_n \| Z_n), \forall j \geq 1$.

It is noted that, with the content of W_n, the cloud can detect the data poisoning attacks and free-rider attacks by integrating with the existing detection schemes

- Batch verification stage –
  The cloud can verify the validity of all received signatures via batch verification. Given a list of requests $R = \{req_n = (W^*_n, \sigma_n, k_n, U_n, D_n, Z'_n)\}^N_{n} = 1$, the batch verification follows
  . $e(\sum_{n=1}^N Y'_n, Y'_{TA}) = e(\sum_{n=1}^N U_n, P)\, e(\sum_{n=1}^N k_n Y''_n, Y_{TA})$

  If holds, all the signatures in the list are valid. Otherwise, the cloud launches the DBV algorithm to find an appropriate batch size and redo batch verification.

  If there is a failure in batch verification, the cloud picks a verification rate €, where € $\in (0, 1)$ is a percentage value, and randomly selects M = € N requests from the list R. The M requests are batch verified following above equation. If equation holds, the cloud then uses the M message to aggregate and update the global ML model. Otherwise, the cloud repeats the above steps at most η iterations.

If the verification fails in the all η iterations, the cloud continues to lower the batch size with M = €M. It will continue until the batch size M is less than λN, where $\lambda \in (0, 1)$ and λN defines the threshold of batch size.

Chapter 3

Result

3.1 Computational Efficiency

We evaluate the computational efficiency only based on the three operations,

- Exponentiation
- Multiplication
- Pairing dominate

Operations	**Time(ms)**
Exponentiation in G_2	T_e : 1.5
Multiplication in G_1	T_m : 15.175
Pairing	T_p : 18.06

TABLE I (Operating Time)

Table I lists the running time of the three operations over the constructed experimental platform, where t_e denotes the time consumption of one exponentiation operation in G2, t_m denotes the time consumption of one multiplication operation in G1, and t_p denotes the time consumption of one pairing operation.

Protocols	Sign	Verify
Liu et al. [11]	$5T_m$	$T_e + 4T_m + T_p$
Zhang at al. [12]	$9T_m$	$6T_m$
Yang et al. [14]	T_e+T_m	$T_e + 5T_p$
Shen et al. [15]	$T_e + T_m + T_p$	$T_e + T_m + 2T_p$
Proposed	$4T_m$	$T_e + 2T_m + 3T_p$

Protocols	Sign Value	Verify Value
Liu et al.	75.875	80.26
Zhang at al.	136.575	91.05
Yang et al.	16.675	91.8
Shen et al.	34.735	52.795
Proposed	60.7	86.03

TABLE II(COMPARISON OF THE COMPUTATIONAL TIME WITHOUT BATCH VERIFICATION)

It is noted in table II that, proposed AAPBV outperforms Zhang at al. while consuming more time than Liu et al., Yang et al., Shen et al. in different stages. Thus, the advantage of the proposed protocol is not significant comparing with Liu et al., Yang et al., Shen et al. in a single verification. The reason is that to support batch verification and more security features (e.g., forward security and non-repudiation), the proposed protocol consumes more operations of pairing and multiplication.

Protocols	Time
Yang et al.	$5(n+1)\ T_m + 5T_p$
Shen et al.	$nT_e + nT_m + 2nT_p$
Proposed	$nT_m + 3T_p$

TABLE III (COMPUTATIONAL TIME OF BATCH VERIFICATION)

Table III show the comparison of the computational overhead with batch verification from the theoretical results. Let the batch size be n. For Shen et al., the computational time for batch verification is nTe + nTm + 2nTp, which is the n times of its verification time without batch verification in Table II.

Thus, although Shen et al. supports batch verification, the time efficiency is not improved comparing with the case without batch verification. For the proposed AAPBV and Yang et al., the time cost is linearly dependent on the batch size and Tm.

Compared with the time consumption listed in Table II, AAPBV and Yang et al. not only support batch verification, but also improve the computational efficiency. Shen et al. costs the most computational time, while the proposed AAPBV reduces the time consumption significantly. Particularly, in proposed protocol, the time consumption is reduced by approximately 71% compared with Shen et al. and is reduced by approximately 80% compared with Yang et al. It is also noted that the batch verification in our proposed protocol reduces the time cost approximately by 82% compared with the case that without batch verification.

AAPBV has the best time efficiency in batch verification. To support the batch verification, it employs more operations of pairing and multiplication in single verification. Computational time with batch verification (BV). Slightly sacrificing the time efficiency in the case of without batch verification. Since the final authentication is conducted with batch verification, the sacrifice is acceptable.

Features	Liu et al.	Zhang et al.	Yang et al.	Shen et al.	Proposed
Confidentiality	Yes	Yes	Yes	Yes	Yes
Anonymity	Yes	Yes	No	Yes	Yes
Mutual Authentication	Yes	Yes	No	No	Yes
Unlinkability	Yes	Yes	No	No	Yes
Unforgery	Yes	No	Yes	No	Yes
Forward Security	No	Yes	No	No	Yes
Backward Security	No	No	No	No	Yes
Batch Verification	No	No	Yes	Yes	Yes

TABLE IV (Feature Comparison)

Chapter 4

Problem found in the methodology

- The proposed protocol itself can't detect the internal attacks (e.g., data poisoning attack and free-rider attack) in DL.

  Data poisoning is a type of adversarial attack on machine learning systems in which an adversary injects a few correctly-labeled, minimally-perturbed samples into the training data, causing a model to misclassify a particular test sample during inference.

  So it needs to integrate with the defence scheme.

Chapter 5

Conclusions

The proposed protocol achieves efficient batch verification with a slight sacrifice of computational efficiency in single verification. In addition, benefiting from the designed DBV algorithm, the global model converges quickly under the proposed AAPBV protocol, further improving the computational efficiency in DL.

The proposed protocol guarantees various security properties while supporting batch verification. The protocol provides high efficiency and enhanced security with a small sacrifice in computational costs in single authentication, making it more practical for large-scale DL systems.