Privacy Computing with Trusted Execution Environments

Dahui Li
Department of Computer Science and Engineering
Southern University of Science and Technology
Shenzhen, China
lidh2021@mail.sustech.edu.cn

Sicheng Zhou
Department of Computer Science and Engineering
Southern University of Science and Technology
Shenzhen, China
zhousc2021@mail.sustech.edu.cn

***Abstract*—In the era of data-driven decision-making, the ability to harness sensitive data while ensuring privacy has become a paramount concern, especially for government agencies holding critical information such as healthcare records, social security data, and demographic information. We propose a novel platform leveraging Trusted Execution Environments (TEEs) to facilitate secure, privacy-preserving collaborations between government entities and external organizations. By utilizing TEEs, our solution guarantees the confidentiality of sensitive data throughout the entire processing pipeline, from data ingestion to model training and inference. The platform enables collaboration among four key stakeholders: data owners, code owners, platform operators, and consumers of machine learning models. Through remote attestation and secure multi-party computation, we ensure data privacy while enabling the extraction of valuable insights from sensitive data. Our approach offers a scalable and secure framework for privacy-aware machine learning and analytics, with significant potential to drive innovation in fields such as healthcare, public policy, and research.**

***Index Terms*—Privacy-preserving machine learning, Trusted Execution Environments (TEEs), secure collaboration, data privacy, remote attestation, multi-party computation, sensitive data, government data, data sharing, secure computation, machine learning, data confidentiality.**

I. Introduction

In today's data-driven world, the ability to leverage sensitive data for machine learning and analytics while maintaining privacy has become increasingly crucial. Government agencies, in particular, possess vast repositories of valuable data spanning healthcare records, demographic information, and social security details that could drive significant innovations in public services and research. However, the utilization of this data faces substantial challenges due to stringent privacy regulations such as General Data Protection Regulation (GDPR) [1] and Health Insurance Portability and Accountability Act (HIPAA) [2], as well as legitimate concerns about data misuse.

Traditional privacy-preserving approaches like data anonymization and differential privacy often fall short, especially when machine learning models require granular data access. While cryptographic solutions such as Fully Homomorphic Encryption (FHE) [3] and Functional Encryption (FE) [4] offer promising theoretical frameworks, they often introduce significant computational overhead and complexity. To address these challenges, we propose a comprehensive platform leveraging Intel Trust Domain Extensions (TDX) [5] and the Trustflow [6] framework that enables secure collaboration between government agencies and external entities. Our solution creates a trusted computational environment where sensitive data remains protected throughout the entire processing pipeline, from data ingestion to model training and inference.

The platform orchestrates interactions between four key stakeholders, i.e., data owners, code owners, platform owners, and consumers. By implementing remote attestation and secure multi-party computation protocols, our solution ensures data confidentiality while enabling valuable insights to be extracted from sensitive government data. We achieve this through a novel integration of TDX's hardware-based security features and Trustflow's trusted application framework, incorporating automated data communication services and robust attestation mechanisms.

Our key contributions include:

- **Privacy-Preserving Framework**: We propose a comprehensive platform that utilizes Trusted Execution Environments (TEEs) to ensure the privacy and confidentiality of sensitive government data during machine learning model training and inference.
- **Secure Collaboration Mechanism**: Our solution enables secure collaboration between multiple stakeholders by leveraging remote attestation and secure multi-party computation protocols.
- **Scalable Architecture:** The platform is designed to be scalable, allowing for efficient handling of large datasets and enabling cross-institutional collaboration while maintaining high standards of data security and privacy.

- **End-to-End Data Protection:** We ensure that sensitive data remains protected throughout the entire data processing pipeline, from ingestion to analysis, without compromising on performance or accuracy of the resulting machine learning models.

II. Related Work

A. Confidential Computing Application Framework

The typical confidential computing application framework mainly involves four participant roles: platform owner, data owner, code owner, and consumer [7]. **Data owner** is responsible for storing, managing, and controlling large amounts of sensitive data, and has absolute ownership over the data. **Code owner** creates programs, algorithms, models, and other code for accessing, computing, and analyzing. **Platform owner** constructs a cloud environment and establishes a confidential computing platform to provide confidential computing functions and services to external entities. **Consumer** has actual demand for the confidential computing platform and services. They rely on data provided by data owners and code provided by code authors and use the management functions and software provided by platform owners to complete their confidential computing needs.

B. Privacy-Preserving ML with crypto-based methods

Fully Homomorphic Encryption (FHE) is an advanced encryption method that enables a non-trustworthy third party to perform computations on encrypted data without ever revealing the underlying confidential information. Lee et.al [8] explore the promising potential of FHE as a tool for enhancing privacy in machine learning applications. Zama [9] develops an open-source privacy-preserving machine learning (PPML) set of tools that automatically turn machine learning models into their homomorphic equivalents. Functional Encryption(FE) is another cryptographic method that enables data owners to grant third-party access to perform specified computations without disclosing their inputs. Different from FHE, it also provides computation results in plaintext. Panzade et.al [10] provide a survey of PPML works based on FE, and analyze the performance and usability of the available FE libraries and their applications to PPML.

C. Privacy-Preserving ML with TEE

Wang et.al [11] propose a hybrid framework integrating SGX and HE, called HT2ML, to protect user's data and models. Narra et.al [12] presents Origami, which provides privacy-preserving inference for large deep neural network (DNN) models through a combination of enclave execution, cryptographic blinding, interspersed with accelerator-based computation.

D. Applications

1) BigDL Privacy Preserving Machine Learning: BigDL PPML creates a trusted cluster environment that allows users to run standard big data and AI applications without compromising privacy or security by leveraging Intel SGX and integrating various hardware and software security technologies [13].

2) Confidential Google Kubernetes Engine Nodes: Confidential GKE Nodes are a security feature offered by Google Cloud that enhances data protection for containerized workloads running on Google Kubernetes Engine [14] [15]. Built on top of Compute Engine Confidential VMs, this technology leverages AMD Secure Encrypted Virtualization (SEV) [16] to encrypt data-in-use within the memory of nodes and workloads.

3) Scalable Open Financial Architecture Enclave: SOFAEnclave [17] is a confidential computing middleware developed by Ant Financial as part of their SOFAStack financial-grade distributed architecture [18]. It aims to address key challenges in confidential computing by improving usability and enabling cluster deployment. SOFAEnclave consists of three main components: Occlum, a memory-safe LibOS for Intel SGX enclaves that allows unmodified applications to run securely; KubeTEE, a framework for deploying and managing confidential computing workloads on Kubernetes; and security testing and analysis tools.

4) WeBank Data Privacy Rights: WeDPR [19] is a suite of privacy-preserving solutions developed by the blockchain team at WeBank, designed to address privacy concerns in digital business processes. It provides efficient, scenario-based frameworks for privacy protection, including encrypted ledgers, multi-party encrypted decision-making, ranking, computation, and selective ciphertext disclosure. These solutions are applicable to various fields such as finance, healthcare, auditing, and digital identity.

5) Federated AI Technology Enabler: FATE [20] is an industrial-grade open-source framework that facilitates secure data collaboration for enterprises through homomorphic encryption and multi-party computation (MPC), developed by FedAI. It supports various federated learning scenarios with algorithms like logistic regression, tree-based methods, deep learning, and transfer learning.

III. Background

Trusted Execution Environment (TEE). Trusted Execution Environment (TEE) is a secure region designed to provide confidentiality and integrity for code and data loaded into it. TEEs are typically implemented at the processor level and provide memory encryption, protecting secure regions from potentially malicious privileged software, including operating systems. [21]. A crucial security feature of TEEs is remote attestation,

which enables remote parties to verify the authenticity and integrity of code running within the TEE. Through remote attestation certification, stakeholders can verify execution results, ensuring computational integrity. This capability is particularly valuable in blockchain-based private data sharing, where TEE's confidentiality and integrity guarantees complement blockchain's transparency.

Intel's Trust Domain Extensions (TDX), introduced in the 4th Generation Intel Xeon Scalable Processor, represents a significant advancement in TEE technology. [5] TDX extends confidential computing capabilities to virtual machines through the Secure-Arbitration Mode (SEAM), providing encrypted CPU state and memory, integrity protection, and remote attestation. This technology specifically addresses the needs of cloud computing by enforcing hardware-assisted isolation for virtual machines and minimizing the attack surface exposed to potentially untrustworthy host platforms. This makes TDX particularly valuable for regulated industries and sensitive data holders who need to securely outsource their computations and data to public cloud infrastructures while maintaining end-to-end protection.

Privacy-preserving machine learning (PPML). Privacy-preserving machine learning (PPML) is an emerging discipline within artificial intelligence aimed at ensuring that sensitive data remains secure and private while still enabling the development and deployment of effective machine learning models [22]–[24]. As the reliance on data-driven technologies increases across industries such as healthcare, finance, and government, concerns over the misuse and exposure of personal data have become paramount. PPML addresses these challenges by employing a variety of techniques designed to safeguard data privacy throughout the machine learning lifecycle. Key methods include *differential privacy* [25], [26], which adds controlled noise to datasets to obscure individual contributions while retaining statistical utility; *homomorphic encryption* [27], [28], which allows computations to be performed on encrypted data without requiring decryption; and *federated learning* [29], [30], a decentralized approach where models are trained locally on user devices and only aggregated updates are shared, rather than raw data. Secure multiparty computation and trusted execution environments further bolster the privacy of computations by ensuring that data remains encrypted even during processing.

By integrating these advanced technologies, PPML enables collaborative and scalable learning across distributed data sources without exposing sensitive information, making it a cornerstone of modern ethical AI practices. In addition to its technical advancements, PPML plays a critical role in helping organizations comply with privacy regulations like GDPR, HIPAA, and CCPA, addressing the growing tension between the demand for high-quality machine learning models and the protection of individual rights. By preserving privacy while unlocking the potential of data, PPML is poised to transform how organizations leverage AI, fostering trust, transparency, and innovation in a privacy-conscious era.

Transport Layer Security (TLS) is a cryptographic protocol designed to provide secure communication over a computer network, primarily for protecting data transmitted between a client (such as a web browser) and a server [31], [32]. TLS ensures that sensitive information, such as passwords, credit card details, and personal data, is securely transmitted over the internet, making it difficult for attackers to intercept or tamper with the data. TLS operates by encrypting the data being transmitted, which prevents unauthorized parties from reading or altering the information during transit. This encryption is achieved using a combination of asymmetric and symmetric cryptography. In the initial phase, TLS uses asymmetric encryption (involving public and private keys) for secure key exchange, allowing both parties to agree on a shared secret key. Once the shared key is established, symmetric encryption is used for the actual data transfer, which is faster and more efficient.

In addition to encryption, TLS provides data integrity by ensuring that the data has not been altered during transmission. It also authenticates the identity of the server (and optionally the client) using digital certificates issued by trusted Certificate Authorities (CAs). This authentication process helps prevent man-in-the-middle attacks, where an attacker might intercept or impersonate one of the communicating parties.

TLS is commonly used in many applications, with the most notable being HTTPS (HyperText Transfer Protocol Secure), which is used to secure web browsing. Other uses of TLS include securing email communications (e.g., SMTP, IMAP, and POP3 over TLS), Virtual Private Networks (VPNs), and instant messaging services. As the internet continues to evolve, TLS remains a cornerstone of securing communications and protecting users' privacy online.

IV. System design

We propose a detailed third-party platform that enables companies to train ML models on sensitive government datasets using Trusted Execution Environments (TEEs). The system structure is shown in Figure 1. We simulate the entire workflow, encompassing four key stakeholders: *data owners, code owners, platform owners, and consumers*:

- *Data owners* securely upload their sensitive datasets to the platform, where they remain encrypted and accessible only within the TEE.

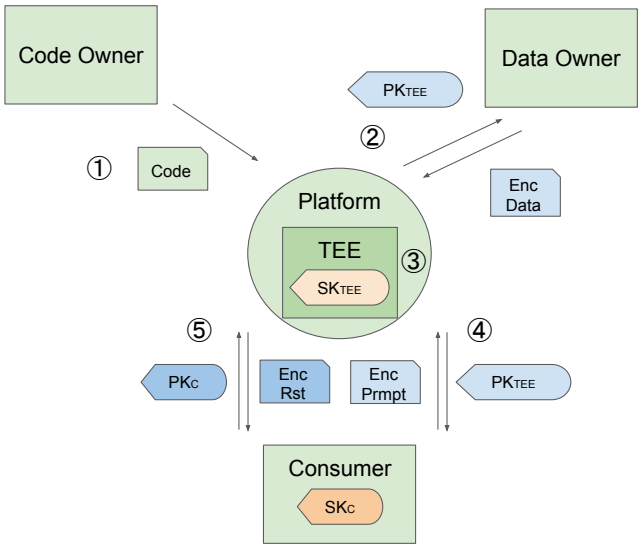

Fig. 1: The proposed workflow. PK: Public Key. SK: Private Key. 1. The *code owner* sends the code to the *platform*; 2. The *platform* sends the PK_{TEE} to the *data owner* and the *data owner* sends back the encrypted data which can only be decrypted by the *platform*; 3. The *platform* trains the model using the code and the data; 4. The *platform* sends PK_{TEE} to the *consumer* who uses this key to encrypt the prompt and then sends it back; 5. The *consumer* sends the PK_C to the *platform* which uses this key to encrypt the inference result and then sends it back.

- *Code owners* contribute their proprietary algorithms and model architectures, which are executed within the secure enclave, ensuring intellectual property protection.
- *The platform owner* manages the infrastructure, orchestrating the secure computation environment and facilitating interactions between parties.
- *Consumers*, who may be researchers or businesses, can request specific analyses or model training without directly accessing the underlying data or algorithms.

Our platform utilizes remote attestation to verify the integrity of the TEE and the code running within it, establishing trust among all parties. We implement secure multi-party computation protocols to enable collaborative model training while maintaining data confidentiality. Additionally, the platform incorporates differential privacy techniques to further protect against potential inference attacks on the trained models. By simulating this end-to-end workflow, we aim to demonstrate the feasibility and effectiveness of our solution in enabling privacy-preserving ML collaborations.

V. Implementation

Our implementation is built upon the TDX VM platform and the open-source Trustflow framework. While TDX serves as the cornerstone for ensuring data security, Trustflow provides the reference architecture for our development workflow. Due to TDX environment configuration constraints, our current implementation operates in Trustflow's simulation mode. The implementation architecture is structured into four primary components: *data encryption and authorization, data communication, model training and inference, and TEE attestation*. We leveraged Trustflow's existing implementations for model training, data authorization, and encryption while developing our own solutions for data communication and TEE attestation.

For *data encryption and authorization*, Trustflow provides a trusted application called the capsule manager that handles key management and data authorization policy administration. Data owners generate their authentication certificates and private keys, then securely upload their data encryption keys, authorization policies, authentication certificates, and identifiers to the capsule manager through Trustflow's CLI tools. These uploaded credentials facilitate subsequent computation and authentication processes. When data consumers need to perform computations, they must provide their certificates and sign the computation task to the trusted application (teeapp) responsible for computation management. The teeapp then automatically verifies with the capsule manager whether the data consumer is an authorized computing entity and whether the proposed computation method complies with the predefined data authorization policies. If the requirements are met, the computation proceeds according to the specified parameters, and encrypted results are returned. After receiving the encrypted results through a secure channel, data consumers must initiate a decryption vote. Only upon the data owner's approval can the data consumer obtain the data decryption key through the CLI tool.

In terms of *data communication*, we implemented a trusted data communication application that automates the manual data transfer operations previously required in the Trustflow workflow. Unlike Trustflow's workflow, which required manual data copying into trusted applications, our implementation incorporates a communication service within the trusted application. This service enables data owners to upload data to the trusted application via REST API over TLS-encrypted channels, triggering model training and inference. Data consumers can similarly retrieve encrypted computation results through this service.

For *model training and inference*, we utilized Trustflow's built-in trusted applications in sequence: data intersection, dataset partitioning, XGBoost [33] model training, and XGBoost model prediction for data processing, training, and inference. Throughout this pipeline, all inputs and outputs between trusted applications remain encrypted. Without proper data au-

thorization, the encryption keys for these data remain securely stored within the capsule manager and teeapp, preventing unilateral access by any party.

Regarding *TEE attestation*, we aim to perform VM measurements upon the initialization of the capsule manager and teeapp, generating a quote to verify the legitimate state of the TDX VM for both data owners and consumers. Another measurement and quote generation occurs after result computation, providing cryptographic proof that the results were genuinely generated by the TEE computation program. This dual attestation mechanism assures data owners that their encryption keys are protected from malicious programs while confirming to data consumers that the received results originate from their intended computation. Due to the environmental configuration constraints of TDX, we will potentially implement this in the future.

VI. Results

We deployed and tested the complete workflow using the Trustflow framework and our applications in an Ubuntu 20.04 environment, which was installed through Windows Subsystem for Linux (WSL). The environment consists of four Docker containers with distinct roles: one container runs the capsule manager for key management and authorization policy control, another container executes trusted applications for model training and inference, while Alice and Bob, as data owners, each operate their own container for key generation and transmission of policies and requirements.

Figure 2 demonstrates the operational status of our application, showing its capability to successfully receive and process both training and inference requests while executing the corresponding script files. Figure 3 and Figure 4 illustrate the simulation of training and inference requests sent by Alice and Bob using the cURL tool. Figure 5 showcases the final inference results obtained after completing the entire workflow, which includes data intersection, dataset partitioning, training, inference, and result decryption through voting protocols. These results validate that our framework successfully accomplishes the entire workflow as designed.

VII. Discussion

While our implementation leverages the TDX VM and the Trustflow framework to build a secure and privacy-preserving platform, there are still several areas where the current approach could be improved.

Limited flexibility in model integration. One of the primary limitations of our implementation is the tight coupling with Trustflow's architecture, particularly its component specification language. This restricts our ability to easily integrate alternative machine learning models into the workflow. While we have utilized the XGBoost model provided by Trustflow for this evaluation, integrating other models requires significant adjustments and adherence to strict specifications, which limits flexibility. Future work could explore ways to decouple the model training components from Trustflow to enable seamless integration of various machine learning algorithms.

Simulation mode constraints. Due to environmental configuration limitations, our current implementation operates in Trustflow's simulation mode rather than the fully operational mode. While this provides a useful reference for developing the workflow, it may not accurately represent the performance and scalability of the system in a production environment. The performance of the platform in real-world deployments, with full TDX capabilities and proper environmental configurations, remains an area for further investigation.

Data communication overhead. The custom trusted data communication application we developed addresses the need for secure data transmission but introduces additional complexity and overhead to the process. While it automates previously manual steps in the Trustflow workflow, there are still concerns regarding its scalability and efficiency when handling large datasets. Optimizing this communication layer to minimize latency and overhead could enhance the overall system performance.

Scalability and real-world deployment. While the current implementation works within the constraints of the simulation mode, scaling the platform for real-world deployments presents additional challenges. The current architecture may face bottlenecks as the number of data owners, consumers, and computations increases. Further research is needed to evaluate the system's scalability in a production environment, especially in terms of managing increased data traffic and handling larger datasets in real-time.

In conclusion, while our platform provides a solid foundation for secure and privacy-preserving collaboration, addressing the above limitations will be crucial for enhancing its performance, flexibility, and scalability in real-world applications.

VIII. Conclusion and future works

In this paper, we presented a privacy-preserving platform for secure collaboration on sensitive government data, leveraging TEEs and the open-source Trustflow framework. Our implementation successfully integrates key components such as model training, data encryption, data communication, and TEE attestation to ensure data confidentiality and enable secure computation across multiple stakeholders. By utilizing TDX and Trustflow, we provided a framework that facilitates secure machine learning and analytics on sensitive data while maintaining strict adherence to privacy regulations.

```
root@docker-desktop:/home/PC-TEE/website# python app.py
 * Serving Flask app 'app'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on https://127.0.0.1:50555
 * Running on https://192.168.65.3:50555
Press CTRL+C to quit
File saved to prompts/test_table
File saved to /host/testdata/breast_cancer/test_table
Executing command: bash predict.sh
192.168.65.3 - - [22/Dec/2024 08:51:15] "POST /predict HTTP/1.1" 200 -
Executing command: bash train.sh
192.168.65.3 - - [22/Dec/2024 08:58:42] "POST /start_training HTTP/1.1" 200 -
File saved to prompts/test_table
File saved to /host/testdata/breast_cancer/test_table
Executing command: bash predict.sh
192.168.65.3 - - [22/Dec/2024 08:59:15] "POST /predict HTTP/1.1" 200 -
```

Fig. 2: TLS data transmission service.

```
(capsule-manager-sdk) root@docker-desktop:/home/transmit# curl -k -v -X POST https://192.168.65.3:50555/start_training
*   Trying 192.168.65.3:50555...
* Connected to 192.168.65.3 (192.168.65.3) port 50555 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* TLSv1.0 (OUT), TLS header, Certificate Status (22):
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.2 (IN), TLS header, Certificate Status (22):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.2 (IN), TLS header, Finished (20):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.2 (OUT), TLS header, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384
* ALPN, server did not agree to a protocol
* Server certificate:
*  subject: C=AU; ST=Some-State; O=Internet Widgits Pty Ltd
*  start date: Dec 21 09:48:17 2024 GMT
*  expire date: Dec 21 09:48:17 2025 GMT
*  issuer: C=AU; ST=Some-State; O=Internet Widgits Pty Ltd
*  SSL certificate verify result: self-signed certificate (18), continuing anyway.
```

Fig. 3: Trigger model training.

However, there are several areas where our approach can be improved. First, the integration of alternative machine learning models into the existing Trustflow workflow requires greater flexibility, and the system's operation in simulation mode limits its real-world performance evaluation. Additionally, the custom data communication application and TEE attestation mechanisms introduce overhead that could affect scalability and efficiency, especially when dealing with large datasets and frequent computations.

In terms of future work, we aim to address these limitations by enhancing the flexibility of model integration, allowing for seamless inclusion of a broader range of machine learning models without strict adherence to Trustflow's specifications; moving beyond simulation mode to fully operationalize the system, allowing for performance evaluation in real-world environments; optimizing the data communication and TEE attestation components to minimize overhead and improve scalability, particularly for large datasets and high-frequency computations; investigating the system's scalability and deployment in real-world settings

```
(capsule-manager-sdk) root@docker-desktop:/home/transmit# curl -k -v -X POST -F "file=@test_table" https://192.168.65.3:50555/predict
Note: Unnecessary use of -X or --request, POST is already inferred.
*   Trying 192.168.65.3:50555...
* Connected to 192.168.65.3 (192.168.65.3) port 50555 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* TLSv1.0 (OUT), TLS header, Certificate Status (22):
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.2 (IN), TLS header, Certificate Status (22):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.2 (IN), TLS header, Finished (20):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.2 (OUT), TLS header, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384
* ALPN, server did not agree to a protocol
* Server certificate:
*  subject: C=AU; ST=Some-State; O=Internet Widgits Pty Ltd
*  start date: Dec 21 09:48:17 2024 GMT
*  expire date: Dec 21 09:48:17 2025 GMT
*  issuer: C=AU; ST=Some-State; O=Internet Widgits Pty Ltd
```

Fig. 4: Trigger model inference.

home > workspace > decrypt_output > predict.table

```
1  score,id,label
2  0.99983,8911834,True
3  5.2e-05,8811842,False
4  0.99983,911408,True
5  0.998086,909220,True
6  0.988456,862261,True
7  0.997474,89511502,True
```

Fig. 5: Decrypted output.

to ensure it can handle the growing demands of secure, privacy-preserving data collaboration.

By addressing these challenges, we believe our platform has the potential to unlock significant breakthroughs in fields such as healthcare, public policy, and research, providing a robust, scalable, and secure solution for privacy-aware machine learning and data analytics.

References

[1] European Commission. General data protection regulation, 2016. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679.

[2] U.S. Department of Health and Human Services. Health insurance portability and accountability act, 1991. https://www.hhs.gov/hipaa/index.html.

[3] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *Cryptology ePrint Archive*, 2012.

[4] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In *Theory of Cryptography: 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings 8*, pages 253–273. Springer, 2011.

[5] Pau-Chen Cheng, Wojciech Ozga, Enriquillo Valdez, Salman Ahmed, Zhongshu Gu, Hani Jamjoom, Hubertus Franke, and James Bottomley. Intel tdx demystified: A top-down approach. *ACM Comput. Surv.*, 56(9), April 2024.

[6] TrustFlow Documentation. https://www.secretflow.org.cn/zh-CN/docs/trustflow/0.4.0b0/. Accessed Dec 2024.

[7] Dengguo Feng, Yu Qin, Wei Feng, Wei Li, Ketong Shang, and Hongzhan Ma. Survey of research on confidential computing. *IET Communications*, 18(9):535–556, 2024.

[8] Joon-Woo Lee, Hyungchul Kang, Yongwoo Lee, Woosuk Choi, Jieun Eom, Maxim Deryabin, Eunsang Lee, Junghyun Lee, Donghoon Yoo, Young-Sik Kim, and Jong-Seon No. Privacy-preserving machine learning with fully homomorphic encryption for deep neural network. *IEEE Access*, 10:30039–30054, 2022.

[9] Zama. Concrete ML: a privacy-preserving machine learning library using fully homomorphic encryption for data scientists, 2022. https://github.com/zama-ai/concrete-ml.

[10] Prajwal Panzade, Daniel Takabi, and Zhipeng Cai. Privacy-preserving machine learning using functional encryption: Opportunities and challenges. *IEEE Internet of Things Journal*, 11(5):7436–7446, 2024.

[11] Qifan Wang, Lei Zhou, Jianli Bai, Yun Sing Koh, Shujie Cui, and Giovanni Russello. Ht2ml: An efficient hybrid framework for privacy-preserving machine learning using he and tee. *Computers & Security*, 135:103509, 2023.

[12] Krishna Giri Narra, Zhifeng Lin, Yongqin Wang, Keshav Balasubramaniam, and Murali Annavaram. Privacy-preserving inference in machine learning services using trusted execution environments, 2019.

[13] Intel. Bigdl privacy preserving machine learning (ppml) user guide, 2024. https://bigdl.readthedocs.io/en/latest/doc/PPML/Overview/ppml.html.

[14] Google. Google kubernetes engine (gke), 2024. https://cloud.google.com/kubernetes-engine/?hl=en.

[15] Google. Encrypt workload data in-use with confidential google kubernetes engine nodes, 2024. https://cloud.google.com/kubernetes-engine/docs/how-to/confidential-gke-nodes.

[16] AMD. Amd secure encrypted virtualization (sev), 2024. https://www.amd.com/en/developer/sev.html.

[17] Ant Financial. Sofaenclave github, 2024. https://github.com/SOFAEnclave.
[18] Ant Financial. Sofaenclave introduction, 2019. https://www.sofastack.tech/blog/sofa-enclave-confidential-computing/.
[19] WeBank. Wedpr github, 2024. https://github.com/WeBankBlockchain/WeDPR-Lab-Core/.
[20] WeBank. Fate github, 2024. https://github.com/FederatedAI/FATE/.
[21] Mohamed Sabt, Mohammed Achemlal, and Abdelmadjid Bouabdallah. Trusted execution environment: what it is, and what it is not. *IEEE Trustcom/BigDataSE/Ispa*, 1:57–64, 2015.
[22] Runhua Xu, Nathalie Baracaldo, and James Joshi. Privacy-preserving machine learning: Methods, challenges and directions. *arXiv preprint arXiv:2108.04417*, 2021.
[23] Bo Liu, Ming Ding, Sina Shaham, Wenny Rahayu, Farhad Farokhi, and Zihuai Lin. When machine learning meets privacy: A survey and outlook. *ACM Computing Surveys (CSUR)*, 54(2):1–36, 2021.
[24] Abdul Rahman Sani, Muneeb Ul Hassan, and Jinjun Chen. Privacy preserving machine learning for electric vehicles: A survey. *arXiv preprint arXiv:2205.08462*, 2022.
[25] Cynthia Dwork. Differential privacy. In *International colloquium on automata, languages, and programming*, pages 1–12. Springer, 2006.
[26] Cynthia Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer, 2008.
[27] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (Csur)*, 51(4):1–35, 2018.
[28] Xun Yi, Russell Paulet, Elisa Bertino, Xun Yi, Russell Paulet, and Elisa Bertino. *Homomorphic encryption*. Springer, 2014.
[29] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021.
[30] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020.
[31] Eric Rescorla. The transport layer security (tls) protocol version 1.3. Technical report, 2018.
[32] Sean Turner. Transport layer security. *IEEE Internet Computing*, 18(6):60–63, 2014.
[33] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.