Starlit: Privacy-Preserving Federated Learning to Enhance Financial Fraud Detection

Aydin Abadi* Mohammad Naseri† Bradley Doyle‡ Francesco Gini‡
Kieron Guinamard‡ Sasi Kumar Murakonda† Jack Liddell‡ Paul Mellor‡
Steven J. Murdoch§ Hector Page‡ George Theodorakopoulos¶ Suzanne Weller‡

*Newcastle University, †Flower Labs, ‡Privitar, §University College London, ¶Cardiff University

***Abstract*—Federated Learning (FL) is a data-minimization approach enabling collaborative model training across diverse clients with local data, avoiding direct data exchange. However, state-of-the-art FL solutions to identify fraudulent financial transactions exhibit a subset of the following limitations. They (1) lack a formal security definition and proof, (2) assume prior freezing of suspicious customers' accounts by financial institutions (limiting the solutions' adoption), (3) scale poorly, involving either $O(n^2)$ computationally expensive modular exponentiation (where n is the total number of financial institutions) or highly inefficient fully homomorphic encryption, (4) assume the parties have already completed the entity alignment phase, hence excluding it from the implementation, performance evaluation, and security analysis, and (5) struggle to resist clients' dropouts. This work introduces *Starlit*, a novel *scalable* privacy-preserving FL mechanism that overcomes these limitations. It has various applications, such as enhancing financial fraud detection, mitigating terrorism, and enhancing digital health. We implemented *Starlit* and conducted a thorough performance analysis using synthetic data from a key player in global financial transactions. The evaluation indicates *Starlit*'s scalability, efficiency, and accuracy.**

***Index Terms*—Federated Learning, Private Set Intersection, Financial Fraud.**

I. Introduction

Sharing data is crucial in dealing with crime. Collaborative data analysis among law enforcement agencies and relevant stakeholders can significantly enhance crime prevention, investigation, and overall public safety. For instance, in the United Kingdom, Cifas, a non-profit fraud database, and fraud prevention organization that promotes data sharing among its members, reported that its members detected and reported over 350,000 cases of fraud in 2019. This collective effort prevented fraudulent activities amounting to £1.5 billion [23]. The National Data Sharing Guidance, developed by the UK Home Office and Ministry of Justice in 2023, further underscores the importance of data sharing in dealing with crime [20].

Typically, inputs for collaborative data analysis come from different parties, each of which may have concerns about the privacy of their data. Federated Learning (FL) [25] and secure Multi-party Computation (MPC) [26], along with their combination, are examples of mechanisms that allow parties to collaboratively analyze shared data while maintaining the privacy of their input data.

FL is a machine learning framework where multiple parties collaboratively build machine learning models without revealing their sensitive input to their counterparts [25], [16]. Vertical Federated Learning (VFL) is a vital variant of FL, with various applications, e.g., in dealing with crime [3] and healthcare [17]. VFL refers to the FL setting where datasets distributed among different parties (e.g., banks) have some intersection concerning users (e.g., have certain customers' names in common) while holding different features, e.g., customers' names, addresses, and how they are perceived by a financial institution. Horizontal Federated Learning (HFL) is another important variant of FL where participants share the same feature space while holding different users, e.g., customers' attributes are the same, but different banks may have different customers.

Advanced privacy-preserving FL-based solutions aiming to detect anomalies and deal with financial fraud may face a new challenge. In this setting, datasets for financial transactions might be partitioned both vertically and horizontally. For instance, a third-party Financial Service Provider (FSP) may have details of financial transactions including customers' names, and involved banks, while each FSP's partner bank may have some details/features of a subset of these customers. Thus, existing solutions for VFL or HFL cannot be directly applied to deal with this challenge.

A. Our Contributions

In this work, we introduce *Starlit*, a novel scalable privacy-preserving federated learning mechanism that can help enhance financial fraud detection. By devising and utilizing *Starlit* in the context of financial fraud, we address all limitations of the state-of-the-art FL-based mechanisms, proposed in [3], [19], [12]. Specifically, we (1) formally define and prove *Starlit*'s security, (2) do not place any assumption on how suspicious accounts of customers are treated by their financial institutions, (3) make *Starlit* scale linearly with the number of participants (i.e., its overhead is $O(n)$) while refraining from using fully homomorphic encryption, (4) include all phases of *Starlit* in the implementation and evaluation, and (5) make *Starlit* resilient against dropouts of clients.

Starlit offers two compelling properties not found in existing VFL schemes. These include the ability to securely:

- Identify discrepancies among the values of shared features in common users between distinct clients' datasets. For instance, in the context of banking, FSP and a bank can detect if a certain customer provides a different home address to each.

- Aggregate common features in shared users among different clients' datasets, even when these features have varying values. For instance, this feature will enhance FSP's data by reflecting whether FSP and multiple banks consider a certain customer suspicious, according to the value of a flag allocated by each bank to that customer's account.

We have implemented *Starlit* and evaluated its performance using synthetic data which comprises about four million rows. This synthetic data was provided by a major organization globally handling financial transactions.

Starlit stands out as the first solution that simultaneously provides the features mentioned above. To develop *Starlit*, we use a combination of several tools and techniques, such as SecureBoost (for VFL), Private Set intersection (for entity alignment and finding discrepancies among different entities' information), and Differential Privacy to preserve the privacy of accounts' flags (that indicate whether an account is deemed suspicious). Moreover, based on our observation that each dataset's sample (or row), such as a financial transaction, can be accompanied by a random identifier, we allow a third-party feature collector to efficiently aggregate clients' flags without being able to associate the flags values with a specific feature, e.g., customer's name.

Summary of our Contributions. In this work, we:

- Introduce *Starlit*, a novel scalable privacy-preserving federated learning mechanism.
- Formally define *Starlit*'s security using the simulation-based paradigm.
- Implement *Starlit* and conduct a comprehensive evaluation of its performance.

B. Primary Goals and Setting

This paper focuses on a real-world scenario in which a server Srv wants to train a machine-learning model to detect anomalies using its data, and complementary data held by different clients $C = \{\mathrm{C}_1, ..., \mathrm{C}_m\}$. For instance, Srv can be a Financial Service Provider (FSP) such as SWIFT[1], Visa[2], and PayPal[3], facilitating financial transactions and payments between various clients in set C, such as banks, eBay, and Amazon—that aims to detect anomalous transactions.

In this setting, Srv may maintain a database of samples/rows between interacting clients, but it does not possess all the details about the users included in each sample. For instance, in the context of financial transactions, FSP holds a dataset containing samples (i.e., transactions) between the ordering account held by bank C_i and the beneficiary account held by bank C_j. Each sample may contain a customer's name, the amount sent, home address, and information about C_i, and C_j. Each client in C maintains a dataset containing certain customers' account information, including customers' details, their transaction history, and even local assessments of their known financial activities. However, each C_j may not hold all users (e.g., customers) that Srv is interested in.

While Srv is capable of training a model to detect anomalous transactions using its data, it could enhance the analytics by considering the complementary data held by other clients concerning the entities involved in the transactions. The ultimate goal is to enable Srv to collaborate with other clients to develop a model that is significantly better than the one developed on Srv's data alone, e.g., to detect suspicious transactions and ultimately to deal with financial fraud. However, a mechanism that offers the above feature must satisfy vital security and system constraints; namely, (i) the privacy of clients' data should be preserved from their counterparts, and (ii) the solution must be efficient for real-world use cases. The aforementioned setting is an example of FL on vertically and horizontally partitioned data in which each Srv's transaction is associated with a sender C_i (e.g., ordering bank), and receiver C_j, e.g., beneficiary bank. Our solution will enhance Srv's dataset with two primary types of features using the datasets of C_i and C_j:

- **Discrepancy Feature**: This will enhance Srv's data by reflecting whether there is a discrepancy between (i) the (value of the) feature, such as a customer's name and address, it holds about a certain user U under investigation and (ii) the feature held by sending client C_i and receiving client C_j about the same user. For each user, this feature is represented by a pair of binary values $(b_{u,i}, b_{u,j})$, where $b_{u,i}$ and $b_{u,j}$ represents whether the information that Srv holds matches the one held by the sending and receiving clients respectively.
- **Sample's Flag Feature**: This will enhance Srv's data by reflecting whether Srv and a client have the same view of a certain user, e.g., a customer is suspicious. This feature is based on a pair of binary private flags for a certain user, where one flag is held by the sending client and the other one is held by the receiving client. In the context of banking, banks often allocate flags to each customer's account for internal use. The value of this flag is set based on the user's transaction history and determines whether the bank considers the account holder suspicious.

To preserve the privacy of the participating parties' data (e.g., data of non-suspicious customers held by banks) while aligning Srv's dataset with the features above, we rely on a set of privacy-enhancing techniques, such as Private Set Intersection (PSI) and Differential Privacy (DP). Briefly, to enable Srv to find out whether the data it holds about a certain (suspicious) user matches the one held by a client, we use PSI. Furthermore, to enhance Srv's data with the flag feature, each client uses local DP to add noise to their flags and sends the noisy flags to a third-party flag collector which feeds them to the model training phase.

II. Related Work

In this section, we briefly discuss the privacy-preserving FL-based approaches used to deal with fraudulent transactions.

[1] https://www.swift.com

[2] https://www.visa.co.uk/about-visa.html

[3] https://www.paypal.com/uk/home

Lv *et al.* [15] introduced an approach to identify black market fraud accounts before fraudulent transactions occur. It aims to guarantee the safety of funds when users transfer funds to black market accounts, enabling the financial industry to utilize multi-party data more efficiently. It involves data provided by financial and social enterprises. The approach utilizes *insecure* hash-based PSI for entity alignment. This scheme differs from *Starlit* in a couple of ways: (i) *Starlit* operates in a multi-party setting, where various clients contribute their data, in contrast to the aforementioned scheme, which has been designed for only two parties, and (ii) *Starlit* deals with the data partitioned both horizontally and vertically, whereas the above scheme focuses only on vertically partitioned data.

Recently, Arora *et al.* [3] introduced an approach that relies on oblivious transfer, secret sharing, DP, and multi-layer perception. The authors have implemented the solution and conducted a thorough analysis of its performance.

Starlit versus the Scheme of Arora et al. The latter assumes that the ordering bank never allows a customer with a dubious account to initiate transactions but allows the same account to receive money. In simpler terms, this scheme exclusively addresses frozen accounts, restricting its applicability. This setting will exempt the ordering bank from participating in MPC, enhancing the efficiency of the solution. In the real world, users' accounts might be deemed suspicious (though not frozen), yet they can still conduct financial transactions within their bank. The bank may handle such accounts more cautiously than other non-suspicious accounts. In contrast, *Starlit* (when applied to financial transactions context) does not place any assumption on how a bank treats a suspicious account. Unlike the scheme in [3], which depends on an ad-hoc approach to preserve data privacy during training, our solution, *Starlit*, uses SecureBoost, a well-known scheme. Thus, compared to the one in [3], Starlit considers a more generic scenario and relies on an established scheme for VFL.

Another approach has been developed by Qiu *et al.* [19]. It uses neural networks and shares the same objective as the one by Arora *et al.* However, it strives for computational efficiency primarily through the use of symmetric key primitives. The scheme incorporates the elliptic-curve Diffie-Hellman key exchange and one-time pads to secure exchanged messages during the model training phase. This scheme has also been implemented and subjected to performance evaluation.

Starlit versus the Scheme of Qiu et al. The latter scheme requires each client (e.g., bank) to possess knowledge of the public key of every other client and compute a secret key for each through the elliptic-curve Diffie-Hellman key exchange scheme. Consequently, this approach imposes $O(n)$ modular exponentiation on each client, resulting in the protocol having a complexity of $O(n^2)$, where n represents the total number of clients. In contrast, in *Starlit*, each client's complexity is independent of the total number of clients and each client does not need to know any information about other participating clients. Moreover, the scheme proposed in [19] assumes the parties have already performed the entity alignment phase, therefore, the implementation, performance evaluation, and security analysis exclude the entity alignment phase.

Furthermore, the scheme in [19] fails to terminate successfully even if only one of the clients neglects to transmit its message. In this scheme, each client, utilizing the agreed-upon key with every other client, masks its outgoing message with a vector of pseudorandom blinding factors. The expectation is that the remaining clients will mask their outgoing messages with the additive inverses of these blinding factors. These blinding factors are generated such that, when all outgoing messages are aggregated, the blinding factors cancel each other out. Nevertheless, if one client fails to send its masked message, the aggregated messages of the other clients will still contain blinding factors, hindering the training on correct inputs. In contrast, *Starlit* does not encounter this limitation. This is because the message sent by each client is independent of the messages transmitted by the other clients.

Kadhe *et al.* [12] proposed an anomaly detection scheme, that uses fully homomorphic encryption (computationally expensive), DP, and secure multi-party computation.

Starlit versus the Scheme of Kadhe et al. The latter heavily relies on fully homomorphic encryption. In this scheme, all parties must perform fully homomorphic operations. This will ultimately affect both the scalability and efficiency of this scheme. *Starlit* does not use any fully homomorphic scheme.

All above solutions share another shortcoming, they lack formal security definitions and proofs of the proposed systems.

III. Informal Threat Model

Starlit involves three types of parties:

- Server (Srv). It wants to train a model to detect anomalies using its data, and complementary data held by different clients. The data Srv maintains is partitioned vertically and horizontally across different clients. Each sample in the data includes various features, e.g., a user's name, sender client, and receiver client.
- Clients ($C_1, ..., C_n$). They are different clients (e.g., nodes, devices, or organizations) that contribute to FL by providing local complementary data to the training process.
- Flag Collector (FC). It is a third-party helper that aggregates some of the features held by different clients. FC is involved in *Starlit* to enhance the system's scalability.

We assume that all the participants are honest but curious, as it is formally defined in [11]. Hence, they follow the protocol's description. But, they try to learn other parties' private information. We consider it a privacy violation if the information about one party is learned by its counterpart during the model training (including pre-processing). We assume that parties communicate with each other through secure channels.

IV. Preliminaries

A. Notations and Assumptions

Let $\mathcal{G}$ be a multi-output function, $\mathcal{G}(inp) \rightarrow (outp_1, ..., outp_n)$. Then, by $\mathcal{G}_i(inp)$ we refer to the i-th output of $\mathcal{G}(inp)$, i.e., $outp_i$. In this paper, we consider semi-honest adversaries. We use the simulation-based paradigm of secure multi-party computation [11] to define and discuss the security of the proposed scheme.

B. Private Set Intersection (PSI)

PSI is a cryptographic protocol that enables mutually distrustful parties to compute the intersection of their private datasets without revealing anything about the datasets beyond the intersection. The fundamental functionality computed by any n-party PSI can be defined as $\mathcal{G}$ which takes as input sets $S_1, ..., S_n$ each of which belongs to a party and returns the intersection $S_\cap$ of the sets to a party. More formally, the functionality is defined as: $\mathcal{G}(S_1, ..., S_n) \rightarrow (S_\cap, \underbrace{\perp, ..., \perp}_{n-1})$, where $S_\cap = S_1 \cap S_2, ..., \cap S_n$. In this work, we denote the concrete PSI protocol with $\mathcal{PSI}$.

C. Local Differential Privacy

Local Differential Privacy (LDP) entails that the necessary noise addition for achieving differential privacy is executed locally by each individual. Each individual employs a random perturbation algorithm, denoted as M, and transmits the outcomes to the central entity. The perturbed results are designed to ensure the protection of individual data in accordance with the specified ϵ value. This concept has been formally stated in [9]. We restate it in the full version of our paper [2].

D. Federated Learning

FL allows model training to occur on individual devices/clients contributing private data. This preserves the privacy of the data to some extent by avoiding direct access to them. The process involves training a global model through collaborative learning on local data, and only the model updates, rather than raw data, are transmitted to the central server. This decentralized paradigm is particularly advantageous in scenarios where data privacy is paramount, such as in healthcare or finance, as it enables machine learning advancements without compromising sensitive information.

1) SecureBoost: A Lossless Vertical Federated Learning Framework: SecureBoost, introduced in [8], stands out as an innovative FL framework designed to facilitate collaborative machine learning model training among multiple parties while safeguarding the privacy of their individual datasets. It accomplishes this by leveraging homomorphic encryption to execute computations on encrypted data, ensuring the confidentiality of sensitive information throughout the training procedure. There are two main technical concepts and phases involved in SecureBoost:

- **Secure Tree Construction:** SecureBoost builds boosting trees by utilizing a non-federated tree boosting mechanism called XGBoost [7] and a partially homomorphic encryption scheme, such as Paillier encryption [18], allowing various operations such as majority votes and tree splits to be performed without exposing the underlying plaintext data to the system's participants.
- **Entity Alignment:** To enable collaborative training, SecureBoost conducts entity alignment to recognize corresponding user records across diverse data silos. This process is typically executed through an MPC (such as PSI), guaranteeing the confidentiality of individual identities.

E. Flower: A Federated Learning Implementation Platform

We implement *Starlit* within *Flower* [5]. This framework offers several advantages, including scalability, ease of use, and language and ML framework agnosticism.

V. System Design

Starlit consists of two main phases: (i) feature extraction and (ii) training. During the feature extraction phase, the two types of features (discussed in Section I-B) are retrieved in a privacy-preserving manner, the data is aligned, and then passed onto a third party, called "Feature Collector (FC)". The use of FC drastically simplifies the training phase from n-party down to 2-party VFL, which will enable the system to scale to a large number of banks. Figure 1 outlines the interaction between the parties in *Starlit*. In Phase 1, each client initially engages with Srv to identify discrepancies in specific user features. Additionally, in the same phase, each client interacts with Srv to extract flags for certain users. Subsequently, each client combines the results of discrepancy extraction with the outcomes of flag extraction, sending the pair along with a random ID (known also to Srv) to FC. Moving on to Phase 2, FC and Srv collaborate to train the VFL model using FC's collected features, Srv's local data, and SecureBoost.

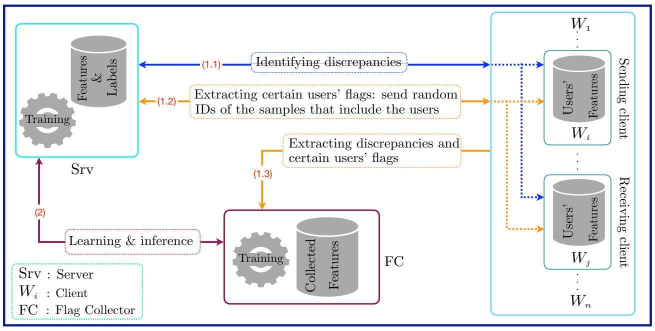


Fig. 1: Outline of parties' interactions in *Starlit*.

This procedure may still leave the chance of an inference attack during model training/deployment. To address this issue, we use LDP, where any flag values that leave the client are obfuscated via a randomization strategy. Note that this protection is an additional layer on top of what is already offered by the SecureBoost protocol, which only shares encrypted (aggregated) gradient information.

VI. Formal Security Definition

In this section, we introduce a generic formal definition, that we call *Celestial*. It establishes the primary security requirements of privacy-preserving (V)FL schemes such as *Starlit*. *Celestial* involves three types of parties, (i) a service provider Srv, (ii) a feature collector FC, and (iii) a set of clients $\{C_1, ..., C_n\}$ contributing their private inputs. Informally, *Celestial* allows Srv to generate a (global) model given its initial model and the inputs of $C_1, ..., C_n$. To achieve a high level of computational efficiency and scalability, in *Celestial*, we involve a third-party FC that assists Srv with computing the model (by interacting with C_is and retrieving the features they hold). The functionality $\mathcal{F}$ that *Celestial* computes takes an input initial model θ from Srv, a set S_i from every C_i, and no input from FC. It returns to Srv an updated model θ'. It returns nothing to the rest of the parties. Hence, $\mathcal{F}$ can be formally defined as follows.

$$\mathcal{F}(\theta, S_1, ..., S_n, \bot) \rightarrow (\theta', \underbrace{\bot, ..., \bot}_{n}, \bot) \quad (1)$$

Since (i) FC interacts with $\text{C}_1, ..., \text{C}_n$ and collects some features from them and (ii) Srv generates the model in collaboration with $\text{C}_1, ..., \text{C}_n$ and FC, there is a possibility of leakage to the participating parties. Depending on the protocol that realizes $\mathcal{F}$ this leakage could contain different types of information. For instance, it could contain (a) each C_i's local model outputs and corresponding gradients (a.k.a. intermediate results) when using gradient descent [21] in VFL, (b) the output of entity aligning procedure, (c) information about features, or (d) nothing at all. We define this leakage as an output of a leakage function defined as follows:

$$\mathcal{L}(inp) \rightarrow (l_1, l_2, ..., l_{n+2}) \quad (2)$$

$\mathcal{L}(inp)$ takes all parties (encoded) inputs, denoted as inp. It returns leakage l_1 to Srv, l_2 to FC, and leakage l_i to C_{i-2}, for all i, where $3 \leq i \leq n+2$.

We assert that a protocol securely realizes $\mathcal{F}$ if (1) it reveals nothing beyond a predefined leakage to a certain party and (2) whatever can be computed by a party in the protocol can be obtained from its input and output only. This is formalized by the simulation paradigm. We require a party's view during the execution of the protocol to be simulatable given its input, output, and the leakage that has been defined for that party.

Definition 1 (Security of *Celestial*)**.** Let $\mathcal{F}$ be the functionality presented in Relation 1. Also, let $\mathcal{L}$ be the above leakage function, presented in Relation 2. We assert that protocol Γ securely realizes $\mathcal{F}$, in the presence of a semi-honest adversary, if for every non-uniform PPT adversary $\mathcal{A}$ for the real model, there exists a non-uniform PPT adversary (or simulator) Sim for the ideal model, such that for every party P, where $P \in \{\text{Srv}, \text{C}_1, ..., \text{C}_n, \text{FC}\}$, the following holds:

$$\{\mathsf{Sim}^{\mathcal{F}, \mathcal{L}_1}_{\mathsf{Srv}}(\theta, \theta')\}_{inp} \overset{c}{\equiv} \{\mathsf{View}^{\mathcal{A}, \Gamma}_{\mathsf{Srv}}(inp)\}_{inp} \quad (3)$$

$$\{\mathsf{Sim}^{\mathcal{F}, \mathcal{L}_2}_{\mathsf{FC}}(\bot, \bot)\}_{inp} \overset{c}{\equiv} \{\mathsf{View}^{\mathcal{A}, \Gamma}_{\mathsf{FC}}(inp)\}_{inp} \quad (4)$$

$$\{\mathsf{Sim}^{\mathcal{F}, \mathcal{L}_{i+2}}_{\mathsf{C}_i}(S_i, \bot)\}_{inp} \overset{c}{\equiv} \{\mathsf{View}^{\mathcal{A}, \Gamma}_{\mathsf{C}_i}(inp)\}_{inp} \quad (5)$$

where $1 \leq i \leq n$.

VII. Starlit's Phases in Detail

A. Privacy-Preserving Feature Extraction

In this section, we elaborate on the two primary privacy-preserving mechanisms that we designed to extract features.

1) Finding Features' Discrepancies: Let $T = \{t_{u,1}, ..., t_{u,m}\}$ be a subset of features that Srv holds for a user U. Consider the scenario where Srv wants to check with a pair of clients (C_i, C_j) if there is a discrepancy between some of the features in T that Srv, C_i, and C_j hold, without revealing and being able to learn anything else. This approach could provide information about anomalous transactions. In the domain of financial transactions, we analyzed synthetic data provided to us and identified key features possessed by FSP for each transaction (with FSP acting as Srv). These features include: (i) $customer_{name}$, (ii) $countryCity_{zipcode}$, and (iii) $street_{name}$ for both the ordering and beneficiary banks. Each bank, per user, maintains various features such as $customer_{name}$, $countryCity_{zipcode}$, and $street_{name}$ (with an associated flag).

Different parties may hold different perspectives on the value of these features. Discrepancies can arise from various factors. For instance, a user may have supplied divergent information to different parties. In the given scenario, a customer might hold accounts with both the ordering and beneficiary banks but could have provided inconsistent details, such as their address, to these banks. Additionally, there is a possibility that the values maintained by Srv have been tampered with, potentially by external entities [4], [24]. Thus, incorporating a feature that signals disparities between a client's data and Srv's data can enhance the accuracy of models.

To detect discrepancies while preserving privacy we use PSI, a method that safeguards the privacy of non-suspicious users' data maintained by the involved parties. The PSI outcomes serve as additional features in the FL model. Specifically, Srv and each client C_i participate in an instance of PSI, receiving a set of strings from Srv and the client. The PSI returns the intersection to C_i. For each user, C_i adds a binary feature b to its dataset (if not already present). If a user's details are in the intersection, b is set to 1; otherwise, it is set to 0. Figure 2 presents this procedure in detail. Hence, we not only employ PSI (as a subroutine in SecureBoost) for entity alignment, but we also leverage it to enhance the accuracy of the final model. Note that the outcome of the protocol in Figure 2 will be transmitted to FC in the second phase (collecting flags of suspicious users), presented below.

2) Collecting Flags of Users: Each user's sample may be accompanied by a flag whose value is computed and allocated by a client. For instance, in the context of financial transactions, for each user's account that a bank holds, there is a flag indicating whether the bank considers the account suspicious. This flag type offers extra information crucial for anomaly detection. Nevertheless, these flags are treated as private information and cannot be directly shared with Srv.

To align the flags with the Srv's dataset without revealing them, we rely on the following observation and idea. The key observation is that each user's sample, which is held by Srv and includes both sender and receiver clients, can be assigned an ID selected uniformly at random from a sufficiently large domain. In certain cases, such as financial transactions, each sample (representing a transaction) already comes with a

- **Parties:** Srv and C$_i$.
- **Input:**
  - ◇ Srv's input, for each user U, is a set T_{Srv} of strings (taken from a dataset DS_{Srv}), where each string has the form $t_{u,1}||t_{u,2}||...||t_{u,m}$ and $t_{u,1}$ is a user's unique ID.
  - ◇ C$_i$'s input, for each user U, is a set T_{C_i} of strings (from its dataset DS_{C_i} of all users), where each string has the form $t_{u,1}||t_{u,2}||...||t_{u,m}$.
- **Output:** Updated dataset DS_{C_i}.

---

1) Srv and C$_i$ invoke an stance of PSI protocol: $\mathcal{PSI}(T_{\text{Srv}}, T_{\mathrm{C}_i}) \rightarrow T_{\cap}$.
2) Given $T_{\cap}$, C$_i$ parses each element of $T_{\cap}$ as $t_{u,1}||t_{u,2}||...||t_{u,m}$.
3) If binary feature b is not in DS_{C_i}, then C$_i$ adds b to each user's feature.
4) C$_i$ sets b as follows. For every $t_{u,j} \in DS_{\mathrm{C}_i}$:
   - Sets $b = 1$, when $t_{u,j} \in S_{\cap}$.
   - Sets $b = 0$, otherwise.
5) C$_i$ returns DS_{C_i}.

Fig. 2: PSI-based method to identify discrepancies.

random ID. As a random string, this ID divulges no specific information about a user's features. For each user's sample, Srv can generate this ID and share this ID (along with a unique feature in the sample) with the clients involved in that sample. Accordingly, if each client groups each ID with a set of binary flags and sends them to FC, FC cannot glean significant information about the user's features linked to those IDs. Based on this observation, we rely on the following idea to extract the flags.

For each user's sample, Srv sends the random ID and a unique feature of the user (e.g., their name or account number) to the related clients. The clients then use their sample information to group each ID with the correct user's flags. It sends this group to FC. When sending a flag for a user to FC, each client also sends to FC the flag b that it generated in Figure 2 (to detect discrepancies). Consequently, FC uses a set (that includes an ID and flags for each user) to create a dataset of flags. This dataset will then be used as the input data for the ML model. The above private information retrieval mechanism is *highly computationally efficient*. This approach still may reveal certain information to the involved parties. Specifically (a) each client gains knowledge of some of their users that are in Srv's dataset, and (b) FC acquires information about which IDs originate from certain clients, enabling the calculation of the number of transactions between each pair of clients. However, the privacy of sensitive information is preserved, as (i) each client remains unaware of details about other participating clients or users' features held at other clients and (ii) FC cannot identify the user involved in a sample. FC only has IDs and a set of flags for each ID. Consequently, FC cannot glean any information about a specific account.

As evident during the feature extraction, each client independently computes its message and sends it to FC without the need to coordinate with other clients. Hence, even if some clients choose not to send their messages, this phase is completed. This is in contrast to the solution proposed in [19] which cannot withstand clients' dropouts.

B. Model Training and Inference

Following the feature extraction phase, Srv and FC jointly possess all the necessary data for training the anomaly detection model. Srv retains a dataset of samples, while FC possesses certain features of samples, i.e., discrepancies and flags (protected by DP). This represents the VFL setting, where only Srv holds the labels to predict. This configuration allows for the utilization of various off-the-shelf protocols suitable for training an ML model, such as those presented in [6], [8], [10]. We use the SecureBoost algorithm (discussed in Section IV-D), which involves the exchange of encrypted (aggregate) gradients between Srv and FC during the training phase. Srv can decrypt the gradients to determine the best feature to split on. Once the model is trained, each party owns the part of the tree that uses the features it holds. Hence, when using the distributed inference protocol in [8], Srv coordinates with the FC to determine the split condition to be used.

Theorem 1 (informal)**.** *Let $\mathcal{F}$ be the functionality defined in Relation 1. If the LDP, SecureBoost, and $\mathcal{PSI}$ schemes are secure, then Starlit securely realizes $\mathcal{F}$, w.r.t. Definition 1.*

We refer to the full version of the paper for a formal statement of Theorem 1 and its proof.

VIII. Implementation of Starlit

We carry out comprehensive evaluations to study *Starlit*'s performance from various aspects, including privacy-utility trade-off, efficiency, scalability, and choice of parameters. In the remainder of this section, we provide an overview of the analysis. Due to space limits, we provided a far more detailed analysis in the full version of the paper [2].

A. The Experiment's Environment

We implement *Starlit* within an FL framework, called Flower (discussed in Section IV-E). We use Python programming language to implement *Starlit*. Experiments were run using AWS ECS cloud with docker containers with 56GB RAM and 8 Virtual CPUs. We adjusted and used the Python-based implementation of the efficient PSI introduced in [14]. We have run experiments to evaluate the performance of this PSI. We conducted the experiments when each party's set's cardinality is in the range $[2^9, 2^{19}]$. Briefly, our evaluation indicates that the PSI's runtime increases from 0.84 to 367.93 seconds when the number of elements increases from 2^9 to 2^{19}. The full version of the paper presents further details on the outcome of the evaluation. Each instance of the PSI, for each account, takes as input string: $account_{number}||$ $customer_{name}||street_{name}$ $||countryCity_{zipcode}$. The output of the PSI is received by the participating bank.

Our experiment involves the utilization of two synthetic datasets. Dataset 1: Synthetic dataset that simulates transaction data obtained from the global payment network of FSP (acting as Srv); Dataset 2: Synthetic dataset related to customers (or users), inclusive of their account information and flags, derived from the partner banks (or clients) of FSP. We refer to the full version of the paper [2] for more details about these datasets.

IX. Empirical Results

In this study, we employ a straightforward approach, utilizing example features extracted from FSP, as provided in the data, in conjunction with four binary values derived from the banks' data. The features extracted from FSP for model training encompass the following: settlement amount, instructed amount, hour, sender hour frequency, sender currency frequency, sender currency amount average, and sender-receiver frequency. We also incorporate four binary flags, indicating the agreement between FSP and the banks on sender and receiver address details, as well as whether the sending and receiving accounts share the same flag for a given account.

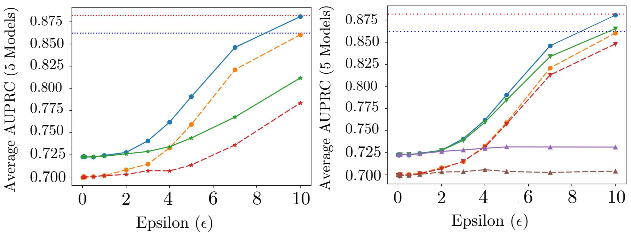
A. Privacy-Utility Trade-off

1) Baseline: To analyze the trade-off between utility and privacy, we establish a benchmark using a *centralized* model constructed within FSP. In this centralized model, all data from banks is revealed in plaintext. The same set of features listed above is extracted. We train a standard XGBoost model with 30 trees. We employ a 5-fold cross-validation with the average precision score as the metric.

2) Evaluation Procedure: The "Area Under the Precision-Recall Curve" (AUPRC) is a metric used to evaluate the performance of an ML classification model. The unit of AUPRC is a value in the range $[0, 1]$. It measures the trade-off between precision and recall and provides a summary of the model's performance across different threshold values for classification. A higher AUPRC indicates better model performance, with 1 being the ideal value representing perfect precision and recall.

3) Starlit: In the evaluation of *Starlit*'s implementation, for analyzing AUPRC that can be achieved at a given level of privacy, we modify the flag values that banks send using DP and construct XGBoost models with these noisy features. We use the same parameters as in the baseline model (30 trees and 5-fold cross-validation) and measure the average precision score for the final model on training and test data, averaging over 5 runs to account for the randomness of the privacy mechanism and the training process. SecureBoost does the same computation as XGBoost while constructing the trees albeit on encrypted gradients. Hence, the additional cost will not be on accuracy but rather on performance. We also observed this to be the case from our experimental results.

4) Key Takeaways: Figure 3 provides a summary of our utility-privacy trade-off analysis. Plot(a) in this figure compares the effect on AUPRC of the model when using Randomized Response (RR) and Laplace mechanism with post-processing for achieving LDP. Consistent with the optimality results presented in [22], [13], RR offers a superior utility-privacy trade-off when compared to the Laplace mechanism. Both RR and the Laplace mechanism yield symmetric transformation matrices, meaning an equal probability for converting a 0 to 1 and a 1 to 0. Plot(b) in Figure 3 illustrates the impact on AUPRC when employing RR and privacy mechanisms. This comparison is conducted at the same ϵ value, with the additional constraint of reducing the probability of converting 0 to 1 (and 1 to 0) by 10% compared to the recommendations provided by RR. These recommendations are determined using our game framework. The results demonstrate that even a slight increase in the probability of converting a zero flag to a non-zero value has a significant impact on the model's performance. This observation aligns with intuition, considering the substantial proportion of zero flag values in the dataset.



(a) Rand. Response vs Laplace. (b) Rand. Response vs Asym. matrices.

Fig. 3: Plot(a) compares the effect on AUPRC of the model when using RR and Laplace mechanism with post-processing for achieving LDP. Plot(b) compares the effect on AUPRC when using RR and privacy mechanisms at the same value of ϵ but with the constraint of 10% less probability of converting 0 to 1 (1 to 0) than what is recommended by RR. In Plot(a), red dotted line: non-private-train, blue dotted line: non-private-test, solid blue line: RR-train, solid orange line: RR-test, solid green line: Laplace-train, and solid red line: Laplace-test. In Plot(b), red dotted line: non-private-train, blue dotted line: non-private-test, solid blue line: RR-train, solid orange line: RR-test, solid green line: 10% less $0- > 1$ than RR-train, solid red line: 10% less $0- > 1$ than RR-test, solid purple line: 10% less $1- > 0$ than RR-train, and solid brown line: 10% less $1- > 0$ than RR-test.

B. Efficiency and Scalability

1) Baseline: SecureBoost's training was configured with 10 trees, each with a depth of 3, a dataset sampling rate of 40%, and a "Gradient-based One Side Sampling" (GOSS) sampling of 0.1. This baseline is used to investigate various configurations' impact on efficiency.

2) Starlit: We analyzed *Starlit*'s efficiency with different SecureBoost configurations. The evaluation's results are illustrated in Table I. SecureBoost offers various options that can be employed to enhance efficiency in different settings. For instance, both direct sampling and GOSS sampling offers a means to reduce network and memory overhead by decreasing the volume of data processed in each round of training. The tree depth is also a crucial parameter for improving accuracy while maintaining an appropriate level of efficiency in terms of training time and memory consumption. Also, the integration of *Starlit* with FATE and Flower enables the splitting of large messages into chunks, facilitating more efficient processing. *Starlit* utilizes numerous Flower rounds, with a significant portion of the final rounds remaining empty due to the requirement of a pre-set round number by Flower. This situation has an impact on the network metrics.

TABLE I: *Starlit*'s Runtime using various training parameters. In the table, H represents time in hours and GB refers to gigabyte. The row highlighted in yellow corresponds to the choice of parameters where AUPRC is at the highest level.

Efficiency Metric	Unit	Sampling Approach		Tree's depth	Max Message Size	Result
		Direct Sampling Rate	GOSS			
AUPRC	–	40%	0.1	3	100MB	0.4715
		100%	0.1	3	100MB	0.5786
		40%	Disabled	3	100MB	0.47
		40%	0.3	3	100MB	0.5965
		40%	0.1	5	100MB	0.652
		40%	0.1	3	1GB	0.4715
The total training time	H	40%	0.1	3	100MB	1.1
		100%	0.1	3	100MB	2.21
		40%	Disabled	3	100MB	2.83
		40%	0.3	3	100MB	1.5
		40%	0.1	5	100MB	1.13
		40%	0.1	3	1GB	1
The peak training memory usage	GB	40%	0.1	3	100MB	12.38
		100%	0.1	3	100MB	17.48
		40%	Disabled	3	100MB	18.39
		40%	0.3	3	100MB	13.66
		40%	0.1	5	100MB	16.4
		40%	0.1	3	1GB	12.22
The network disk volume usage	GB	40%	0.1	3	100MB	4.98
		100%	0.1	3	100MB	14.51
		40%	Disabled	3	100MB	16.61
		40%	0.3	3	100MB	7.84
		40%	0.1	5	100MB	5.1
		40%	0.1	3	1GB	4.34
The network file volume usage	GB	40%	0.1	3	100MB	993
		100%	0.1	3	100MB	1270
		40%	Disabled	3	100MB	1256
		40%	0.3	3	100MB	1035
		40%	0.1	5	100MB	1316
		40%	0.1	3	1GB	927

C. Contrasting Starlit with the Baseline

Starlit and the baseline achieve the same level of AUPRC when *Starlit*'s (i) direct sampling rate is 40%, (ii) the tree's depth is 3, and (iii) GOSS is not disabled. *Starlit* achieves its highest AUPRC level (i.e., 0.652) when the tree's depth is set to 5. Remarkably, in this instance, *Starlit*'s AUPRC surpasses even the baseline setting (i.e., 0.652 versus 0.4715). When the tree's depths in *Starlit* and baseline are set to 5 and 3 respectively, then *Starlit* can attain a superior AUPRC level compared to the baseline. However, in this setting, *Starlit* would impose approximately 1.3 times higher cost.

X. Conclusion

In this work, we introduced *Starlit*, a scalable privacy-preserving and demonstrated its applications in dealing with financial fraud, mitigating terrorism, and improving digital health. We formally defined and proved the security of *Starlit* in the simulation-based model. To formally capture the security of *Starlit*, we have defined a set of leakage functions that may hold independent significance. We implemented *Starlit* and conducted a comprehensive analysis of its performance and accuracy, using synthetic data provided by one of the key players facilitating financial transactions worldwide.

In secure FL, the output inevitably leaks some information about participants private inputs, which may deter parties with sensitive or valuable data, especially if they have no stake in the outcome. Future work could extend *Starlit* to reward active contributors, linking FL with the data market [1].

References

[1] Aydin Abadi. Earn while you reveal: Private set intersection that rewards participants. Cryptology ePrint Archive, Paper 2023/030, 2023.

[2] Anonymous. Starlit: Privacy-preserving federated learning to enhance financial fraud detection–extended version. https://github.com/anonymous2012000/Starlit/blob/main/main.pdf, 2025.

[3] Sunpreet S. Arora, Andrew Beams, Panagiotis Chatzigiannis, Sebastian Meiser, Karan Patel, Srinivasan Raghuraman, Peter Rindal, Harshal Shah, Yizhen Wang, Yuhang Wu, Hao Yang, and Mahdi Zamani. Privacy-preserving financial anomaly detection via federated learning & multi-party computation. *CoRR*, abs/2310.04546, 2023.

[4] Tom Bergin and Nathan Layne. Special report: Cyber thieves exploit banks' faith in swift transfer network. *Reuters*, 2016.

[5] Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, and Nicholas D. Lane. Flower: A friendly federated learning research framework. *CoRR*, 2020.

[6] Iker Ceballos, Vivek Sharma, Eduardo Mugica, Abhishek Singh, and Alberto Roman. Splitnn-driven vertical partitioning. *arXiv preprint*, 2020.

[7] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *ACM SIGKDD*, 2016.

[8] Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, Dimitrios Papadopoulos, and Qiang Yang. Secureboost: A lossless federated learning framework. *IEEE Intelligent Systems*, 2021.

[9] John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy, data processing inequalities, and statistical minimax rates. 2013.

[10] Wenjing Fang, Derun Zhao, Jin Tan, Chaochao Chen, Chaofan Yu, Li Wang, Lei Wang, Jun Zhou, and Benyu Zhang. Large-scale secure xgb for vertical federated learning. In *ACM CIKM*, 2021.

[11] Oded Goldreich. *The Foundations of Cryptography - Volume 2, Basic Applications*. Cambridge University Press, 2004.

[12] Swanand Ravindra Kadhe, Heiko Ludwig, Nathalie Baracaldo, Alan King, Yi Zhou, and Keith Houck. Privacy-preserving federated learning over vertically and horizontally partitioned data for financial anomaly detection. *CoRR*, 2023.

[13] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. Extremal mechanisms for local differential privacy. *Advances in neural information processing systems*, 2014.

[14] Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu. Efficient batched oblivious PRF with applications to private set intersection. In *CCS*, 2016.

[15] Boliang Lv, Peizhe Cheng, Cheng Zhang, Hong Ye, Xianzhe Meng, and Xiao Wang. Research on modeling of e-banking fraud account identification based on federated learning. In *IEEE Intl Conf on Dependable, Autonomic and Secure Computing*, 2021.

[16] H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629, 2016.

[17] Dinh C. Nguyen, Quoc-Viet Pham, Pubudu N. Pathirana, Ming Ding, Aruna Seneviratne, Zihuai Lin, Octavia A. Dobre, and Won-Joo Hwang. Federated learning for smart healthcare: A survey. *ACM Comput. Surv.*, 2023.

[18] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, pages 223–238, 1999.

[19] Xinchi Qiu, Heng Pan, Wanru Zhao, Chenyang Ma, Pedro Porto Buarque de Gusmão, and Nicholas D. Lane. Efficient vertical federated learning with secure aggregation. *CoRR*, 2023.

[20] UK Home Office and Ministry of Justice. Data sharing for the criminal justice system guidance, 2023.

[21] Li Wan, Wee Keong Ng, Shuguo Han, and Vincent C. S. Lee. Privacy-preservation for gradient descent methods. In *Proceedings of the 13th ACM KDD*. ACM, 2007.

[22] Yue Wang, Xintao Wu, and Donghui Hu. Using randomized response for differential privacy preserving data collection. In *EDBT/ICDT Workshops*, volume 1558, pages 0090–6778, 2016.

[23] Olivia White, A Madgavkar, Z Townsend, J Manyika, T Olanrewaju, T Sibanda, and S Kaufman. Financial data unbound: The value of open data for individuals and institutions. *McKinsey Global Institute*, 2021.

[24] Davey Winder. This is how hackers accessed 34,942 paypal accounts. *Forbes*, 2023.

[25] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.*, 2019.

[26] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *FSC*, 1982.