# U.S.-U.K. PETs Prize Challenge: Anomaly Detection via Privacy-Enhanced Federated Learning

**HAFIZ ASIF**[1], **SITAO MIN**[2], **XINYUE WANG**[2], **JAIDEEP VAIDYA**[2] **[Fellow, IEEE]**

[1]Information Systems and Business Analytics Department, Hofstra University, Hempstead, NY 11549 USA

[2]Management Science and Information Systems Department, Rutgers University, Newark, NJ 08901 USA

## Abstract

Privacy Enhancing Technologies (PETs) have the potential to enable collaborative analytics without compromising privacy. This is extremely important for collaborative analytics can allow us to really extract value from the large amounts of data that are collected in domains such as healthcare, finance, and national security, among others. In order to foster innovation and move PETs from the research labs to actual deployment, the U.S. and U.K. governments partnered together in 2021 to propose the PETs prize challenge asking for privacy-enhancing solutions for two of the biggest problems facing us today: financial crime prevention and pandemic response. This article presents the Rutgers ScarletPets privacy-preserving federated learning approach to identify anomalous financial transactions in a payment network system (PNS). This approach utilizes a two-step anomaly detection methodology to solve the problem. In the first step, features are mined based on account-level data and labels, and then a privacy-preserving encoding scheme is used to augment these features to the data held by the PNS. In the second step, the PNS learns a highly accurate classifier from the augmented data. Our proposed approach has two major advantages: 1) there is no noteworthy drop in accuracy between the federated and the centralized setting, and 2) our approach is flexible since the PNS can keep improving its model and features to build a better classifier without imposing any additional computational or privacy burden on the banks. Notably, our solution won the first prize in the US for its privacy, utility, efficiency, and flexibility.

### Keywords

Anomaly detection; differential privacy; federated learning; financial crime; fraud detection; payment network systems

CORRESPONDING AUTHOR: HAFIZ ASIF (hafiz.asif@hofstra.edu).

## I. INTRODUCTION

Data is the lifeblood of the digital economy. The volume of data/information created, captured, copied, and consumed worldwide has grown consistently – from 2 Zettabytes in 2010 to 64.2 Zettabytes in 2020, with a forecast of over 180 Zettabytes by 2025.[1] Data science, AI/ML, and analytics are being used to make sense of all of this data and utilize it. However, as more and more data is collected and analyzed, privacy is increasingly at risk. The risk to privacy has been identified and explicitly called out by the U.S. President in their recently released executive order on Safe, Secure, and Trustworthy Artificial Intelligence.[2]

Privacy Enhancing Technologies (PETs) can be a potential solution to this conundrum, and the national strategy to advance privacy-preserving data sharing and analytics recognizes that PETs can protect privacy by removing personal information, by minimizing or reducing personal data, or by preventing undesirable processing of data, while maintaining the functionality of a system.[3] However, despite the development of advanced PETs such as secure multiparty computation [1], homomorphic encryption [2], differential privacy [3], zero knowledge proofs [4], synthetic data [5], federated learning [6], and trusted execution environments [7], as well as significant development of research papers applying them to solve problems [8], [9], their practical use is still quite limited.

Recognizing this, at the inaugural Summit for Democracy (in 2021), the U.S. and U.K. governments partnered together to propose a set of prize challenges to unleash the potential of these democracy-affirming technologies to make a positive impact[4] [10]. Named the PETs prize challenge, this competition utilized a red team/blue team approach with two types of participants: blue teams developed privacy-preserving solutions, while red teams acted as adversaries to test those solutions.

The competition was structured into two tracks: the first focused on transforming financial crime prevention while the second focused on boosting pandemic response capabilities. In the financial crime track, the objective was to develop solutions that help tackle the challenge of international money laundering, which finances organized crime including human trafficking and terrorist financing, and undermines economic prosperity. The impact of this problem is immense, since money laundering costs up to US $2 trillion each year, according to UN estimates.[5]

Information sharing and collaborative analytics among financial organizations make it much more feasible to detect money laundering and financial fraud. However, it is difficult to realize such information sharing/analytics due to legal concerns with respect to privacy and institutional concerns due to autonomy and with respect to the confidential nature of the information.

[1] https://wwwstatista.com/statistics/871513/worldwide-data-created/
[2] https://www.whitehouse.gov/briefing-room/statements-releases/2023/10/30/fact-sheet-president-biden-issues-executive-order-on-safe-secure-and-trustworthy-artificial-intelligence/
[3] https://www.whitehouse.gov/wp-content/uploads/2023/03/National-Strategy-to-Advance-Privacy-Preserving-Data-Sharing-and-Analytics.pdf
[4] https://drivendata.co/blog/federated-learning-pets-prize-winners-phases-2-3
[5] https://www.un.org/en/coronavirus/illicit-financial-flows

To solve this problem, in the financial crime prevention track, innovators were asked to develop end-to-end privacy-preserving federated learning solutions to detect potentially anomalous payments, leveraging a combination of input and output privacy techniques. Synthetic datasets created by SWIFT, the global provider of secure financial messaging services, were provided to the participating teams and used to develop general solutions. The solutions developed were judged on several different criteria, including privacy, accuracy, efficiency and scalability, adaptability, usability and explainability as well as innovativeness.

This article presents the Rutgers ScarletPets solution for anomaly detection via privacy-enhanced two-step federated learning. Our solution was the winner of the PETs Challenge for transforming financial crime prevention, ranking first in the US.

The two step approach is both accurate and scalable, and it works as follows. In the first step, we mine features based on account-level data and labels, and then use a privacy-preserving encoding scheme to augment these features to the data held by PNS. In the second step, PNS now learns a highly accurate classifier from the augmented data.

To classify a transaction, we first check if it's a simple rule-based anomaly. If it is, then it is labeled as an anomaly. However if the transaction is not a simple anomaly, then the classifier is used to label it.

Overall, our solution has several advantages. Indeed, it won the PETs challenge because along with providing comprehensive privacy guarantees, it had the advantage of being fast, scalable, and accurate (comparable to its centralized counterpart), as well as being highly flexible. By requiring only minimal changes to the existing architecture, it is actually eminently deployable in practice. The implementation code is made available through NIST's official Github account [11].

## II. BACKGROUND, NOTATION, AND BASIC SETUP

We tackle the problem of combating financial fraud in national or international transfers using collaborative and privacy-preserving anomaly detection, posed in the financial crime prevention track of the PETs Prize Challenge [10].

### A. PETS CHALLENGE – FINANCIAL CRIME PREVENTION

The financial track was aimed at developing privacy-preserving financial information sharing and collaborative analytics to detect anomalous payments without compromising the privacy of individuals' or organizational data. Thus, this broad and open effort intended at building trust in PETs in general, and accelerating adoption and the development of efficient privacy-preserving federated learning solutions in general to counter financial crimes such as money laundering that costs up to $2 trillion each year [12].

The challenge was organized in three phases. In the phase I, participants were provided a synthetic global transactional dataset of a *Payment Network Systems (PNS)* (which was created by SWIFT) [13]. Using the insights from this data, the participants proposed their approaches and outlined them as concept papers that were then evaluated by a review panel. The qualified proposals moved to the Phase II, wherein the proposed approaches were implemented and then evaluated by the review panel as well as by execution of the protocols on a standardized online platform. Additionally, the approaches were evaluated for their privacy and security guarantees by a separate set of participants; this constituted the Phase III of the challenge.

## B. GOALS AND OBJECTIVE

As per the problem description,

> *"the analytical objective* [*of the PETs Challenge is*] *to train a model* [over PNS's and banks' data] *that enables PNS to identify anomalous transactions" while preserving privacy, i.e., the solution "is able to provably ensure that sensitive information in the datasets remain confidential to the respective data owners across the machine learning lifecycle"* [14].

The main goal here is to combine insights from transaction-level data, seen in the PNS, and the account level (meta) data known only to the respective bank. Thus, if the parties, i.e., banks and PNS collaborate, then collaborative machine learning can be used to build a classifier, i.e., decision model — to identify the anomalous transaction before fully executing it via the network. The use of such collaborative analytics will boost system performance by lowering the risks of fraud, the average processing time, and the resource-wastage in processing the transactions across all parties, i.e., all banks and the PNS.

Since all of this information, contained in transactions and bank's data, is sensitive and confidential, it is imperative to protect privacy while learning the model. Ideally, no party should learn any additional information by contributing its data to build the classifier, and the model should not expose the private and sensitive information of individuals or banks. Thus, the use of PETs-based solutions, such as federated learning, homomorphic encryption [15], secure multiparty protocols [16], garbled circuits [17], and differential privacy, are befitting this setting. In particular, federated learning solutions were encouraged for they avoid sharing of raw data, and instead rely on learning and sharing parameters of a model to complete the learning task.

To achieve the objective, it is imperative to (1) understand the basic setup of PNS [14] (discussed in Section II-C), and (2) how data is distributed and what information about transactions and their processing is known to banks and PNS (discussed in Section II-D). Note that (1) is needed to know how the data is distributed among the parties (i.e., banks and PNS), and hence, to construct an effective federated learning algorithm that is distributed in nature; (2) is needed to determine exactly what information is known to which party when a transaction is processed, and hence, to quantify information leakage appropriately.

## C. PNS SETUP, TRANSACTIONAL AND ACCOUNT META DATA

A payment network system such as SWIFT [18] consists of advanced Internet protocol-based messaging platform (and network), which enables the participating institutions (e.g., banks) to send standardized messages, such as pertaining to processing/execution of a transaction, to each other (see Fig. 1 for a basic setup of a PNS). Namely, each message/ transaction first goes to PNS and is then sent to the next recipient. In the problem setting, it was assumed that the PNS, executing a transaction, sees all the information in the transaction and stores it, which gives us the transactional data, $\text{PNS}(\mathscr{T})$, at PNS site. Note that we abuse the notation and use PNS to refer to a party that has the view of PNS, i.e., a party that controls the network and sees all the messages passing through it.

Let us now look at the banks connected via PNS. Say there are $N$ banks (participating in PNS), each with its customers who can send and/or receive money via wire transfers, i.e., transactions executed via PNS (see Fig. 2 for an example transaction). Each bank has the account information (i.e., metadata), for example, bank account number, account name, street address, country, and an account-level feature, called `Flag`: which captures the status or behavior of the account, e.g., whether the account is operating normally (`Flag` = 00), under monitoring (`Flag` = 05), or suspended (`Flag` = 06). See Fig. 3 and compare it to Fig. 2 for differences in transaction level data and account (i.e., banks) level data (for more details see [14]). Also, note that it is possible, and often it's the case, that account level information in the transaction is incorrect.

While making a transaction (i.e., transferring funds), all the account information (i.e., bank, account number, name, and address) of the ordering account (i.e., the one sending the funds) and beneficiary account except for `Flag`'s values (as well as the recipient bank and other transaction details, e.g., amount and currency) is shared with PNS (i.e., the PNS system).

**C. SUFFICIENCY OF SINGLE `Flag` FEATURE—**As long as the account level characterization is given by a set of finite features, one account-level feature, e.g., `Flag`, is sufficient to capture the necessary correlations. Let us say the account level characterization features are defined over a finite space (this assumption is supported by the data provided by PNS). Since finite space, defined by multiple features (each of which is finite) can be mapped to a single finite feature via one-to-one mapping, we simplify the account-level features by considering only a single feature `Flag`.

## D. DATA DISTRIBUTION & PARTIES' KNOWLEDGE

For each transaction that is executed via PNS, PNS is provided with the ordering and beneficiary accounts' information — which can be *incorrect* — amount and currency details, and the receiving bank for the transaction.[6]

To simplify the exposition, we conceptualize the notion of *full transaction* from an ordering account in bank $i$, let's call it $i$-Bank, to the beneficiary account in $j$-Bank (where $i$ and $j$ are two unique integers between 1 and $N$ that denote individual, non-identical, banks). Each full transaction consists of all the information known (i.e., sent) to PNS (as part of the transaction) as well as the information of ordering and beneficiary accounts, which are only known to the banks $i$-Bank and $j$-Bank.[7]

[6]Note that one complete wire, i.e., an end-to-end transaction, may consist of multiple single hop transactions. For example, a transaction from Bank#1 to Bank#2, first goes to Bank#3, and then from Bank#3 to Bank#2; this is akin to network routing protocol, where a bank (Bank#1) may not know how to send funds to desired bank (Bank#2) but it knows another bank (Bank#3) that can send the funds to the desired bank (Bank#2). We focus on identifying fraud on a single transaction level, that is, if the transaction at any hop is identified as anomalous, the end-to-end transaction is labeled as anomalous.

[7]Note that any full transaction is an individual transaction, i.e., it is not an end-to-end transaction.

**a: Federated data distribution:** Let $\mathcal{T}$ be the set of full transactions under consideration (e.g., pertaining to the training data). $\mathcal{T}$ is distributed both vertically and horizontally such that

- Vertical partitions: $\mathcal{T}$ is vertically partitioned between PNS and the banks: $\mathrm{PNS}(\mathcal{T})$ consists of transaction-level features (or attributes) known to PNS and $\mathrm{Bank}(\mathcal{T})$ consists account-level meta data (e.g., correct account information and the corresponding `Flag` values).
- Horizontal partitions: $\mathrm{Bank}(\mathcal{T})$ is horizontally partitioned among banks such that only the ordering account's bank knows the attributes of the ordering account and the beneficiary account's bank knows the attributes beneficiary account. We use $i - \mathrm{Bank}(\mathcal{T})$ to denote the account level data for the transactions where $i$-Bank's account was ordering or beneficiary.

Thus, for any full transaction, $\tau \in \mathcal{T}$, from $i$-Bank's ordering account to $j$-Bank's beneficiary account, $\mathrm{PNS}(\tau)$ gives the transaction-level data available at PNS site, while $\mathrm{ord}(\tau)$ gives account-level data of the ordering account from $i - \mathrm{Bank}(\mathcal{T})$, and $\mathrm{ben}(\tau)$ gives account-level data of the beneficiary account from $j - \mathrm{Bank}(\mathcal{T})$. Furthermore, we use $\mathrm{oFlag}(\tau)$ and $\mathrm{bFlag}(\tau)$ respectively to denote the ordering and beneficiary accounts' flag value;[8] and use $\mathrm{Label}(\tau)$ to denote the label for the transaction, $\mathrm{Label}(\tau) = 1$ for anomalous and $\mathrm{Label}(\tau) = 0$ for normal.

**b: Knowledge of the parties in PNS—**PNS knows all the transaction-level features as they pass via PNS. Thus, they are contained in $\mathrm{PNS}(\mathcal{T})$. Additionally, as per our understanding, PNS also gets to know the *status reason code* for transactions [14]: the codes reveal to PNS if the transaction was executed successfully or was *un*successfully terminated or denied due to invalid account information or some non-normal status of the account.

Thus, we assume that following is the minimal **information leakage for PNS**: it knows if a transaction is successfully executed; and if a transaction is denied due to invalid account information or "non-normal" account status; it knows that the failure was due to either, but it cannot distinguish between the two reasons. In addition, since end-to-end transactions are visible to PNS, it also knows the bank-membership of the ordering and receiving account.

The **information leakage for banks** processing a transaction (whether the sender or the receiver) is such that they know $\mathrm{PNS}(\mathcal{T})$; furthermore, we assume that a bank processing the transaction knows the reason code, and if the bank contains the ordering (or beneficiary) account, it knows know the valid account level data.

## E. DIFFERENTIAL PRIVACY

Differential privacy (DP) [3] aims to preserve individual's privacy while publishing statistical information about a database. DP provides formal guarantees that the distribution of query results changes only slightly with the addition or removal of a single record in the database. Formally, a mechanism (randomized algorithm) $\mathscr{M}$ is $\epsilon$-differentially private if for

[8]Note that for any partial transaction $\tau'$ in $\mathrm{PNS}(\mathcal{T})$, the associated flag values are unknown, i.e., $\mathrm{oFlag}(\tau') = \mathrm{bFlag}(\tau') = \perp$, i.e., null.

all possible sets of the outputs, $S \subseteq Range(\mathscr{M})$, and all neighboring database (that differ by a single record) $D$ and $D'$, we have that

$$Pr(\mathscr{M}(D) \in S) \leq e^{\varepsilon} \times Pr(\mathscr{M}(D') \in S),$$

where $\varepsilon \geq 0$, and it quantifies the level of privacy. The higher its value, the weaker the privacy guarantee.

We also employ attribute level differential privacy [19], where the only difference is how the neighboring databases are considered. In attribute-DP, neighbors are considered only in terms of set of sensitive attributes as opposed to all attributes (sensitive or otherwise).

Two important properties of DP that we will rely on are sequential and parallel composition. According to sequential composition, when a single data point (or record) is used in $n$ independent $\varepsilon$-DP computations, the overall DP guarantee reduces to $(n\varepsilon)$-DP. On the other hand, in parallel composition, when mutually exclusive partitions of the database are used by independent $\varepsilon$-DP computations, the overall privacy guarantee remains the same, i.e., $\varepsilon$-DP.

We use Laplace mechanism to achieve DP. For count queries (which just count the number of records in a database satisfying a given property), Laplace mechanism adds independently sampled noise form Laplace distribution (of mean 0 and scale $1/\varepsilon^2$) to the true answer – the perturbed answer is guaranteed to be $\varepsilon$-DP.

### F. BLOOM FILTERS (BF)

In our approach, we will augment PNS($\mathscr{T}$) with a secure and privacy-preserving encoding of account-level features (which are known to banks). And for this encoding scheme, we will use bloom filters (BFs). BF is a probabilistic data structure that allow for the storage and look up of elements. The data stored in a BF is not directly retrievable. Once data is 'inserted', data can be checked to see if it likely has been seen or if it definitely has not. BF is a space-efficient option with an acceptable false positive rate, and has been widely used in many fields [20], [21]; for instance, they are employed for privacy-preserving biometric issue [22]. Many variants of BF exist. We consider the case where the BF is a bit vector of length $\alpha$. All the bits are initialized to 0. To insert an element, $w$ independent hash functions are used to randomly map the element into $w$ positions in the bit vector, which are set to 1. To query if an element is a member, BF maps the element into its bit vectors with the $w$ hash functions. If all the $w$ bits are 1 s, then the element is considered to be in the BF and otherwise not.

## III. TECHNICAL APPROACH

Given the analytical objective, the unique nature of data and how it is distributed across parties, i.e., PNS and banks (discussed in Section II-D), and the requirements for an easily deployable solution, we opt for a domain- and problem-specific solution because this enables an improved trade-off between the practical usability and protection of the data.

We first describe our approach in the centralized setting (where all the data is gathered at one site). Then we propose our novel two-step federated learning anomaly detection method, which extends the centralized setting to the collaborative setting. Note that we use the term ‘federated learning’ in its more general sense of collaborative learning. In the first step of the approach, we mine features based on account-level data (from banks) and labels (from PNS); we then use a novel privacy-preserving encoding scheme to encode these features. In the second step, PNS uses the secure encoding to create augmented data $(\mathrm{PNS}(\mathscr{T})+)$ to build a classifier to identify anomalies. Our federated approach (as outlined shortly):

- is fast and accurate (comparable to its centralized counterpart), and it requires minimal changes to the existing architecture—it actually builds upon it;
- can detect one of the most challenging anomalies, which occur due to mis-specification of information (e.g., incorrect name, account number, or address) without compromising privacy;
- can preserve all sensitive information, not only at the account level but also at the model level (privacy objectives are outlined in Section III-A);
- is also highly flexible; this is because once the data is augmented, any classifier can be built on the augmented data without an active involvement of other parties, allowing drop-in replacement with newer technology;
- is very scalable, allowing new banks to be added in, and is also easily explainable.

## A. PRIVACY OBJECTIVES

The aim is to learn a global anomaly detection model using $\mathrm{PNS}(\mathscr{T})$ and banks’ data, i.e., 1-`BankData`, $2-\mathrm{BankData},\ \ldots,\ N-\mathrm{BankData}$ without sharing raw data, i.e., via federated learning, and without compromising sensitive information. Here, $i$-`BankData`- denotes all the account-level data that $i$-Bank has on its customers.

**Algorithm 1:**

Centralized Approach.

```
Input: $\mathscr{T}$, set of full transactions; $\gamma \in (0,\ 1)$, threshold; differential privacy parameters, $\varepsilon_1,\ \ \varepsilon_2$
Output: Classifier $\mathscr{C}^+$
  1: m-Features ←Mine Flag-based features (with anomaly-ratio > $\gamma$) from $\mathscr{T}(\varepsilon_1 - \mathrm{DP})$
  2: $\mathrm{PNS}(\mathscr{T}')+$ ← Augment Features to $\mathrm{PNS}(\mathscr{T}')$ via bloom filter based encoding
  3: dp-PNS+train ← Make sensitive features in $\mathrm{PNS}(\mathscr{T}')+\varepsilon_2$-differentially private
  4: Learn classifier, $\mathscr{C}^+$ (e.g., using XGBoost) over dp-PNS+train
-----------------------------------------
-----------------------------------
To classify a transaction, first compute its Flag-based features’ values, augment them to PNS’s transaction-based
features, and then use $\mathscr{C}+$ to obtain its classification.
```

*The sensitive information to be protected is all personally identifiable information, namely, account number, name, address, Flags* (when it is non-zero, i.e., the account is flagged), *transaction identifiers, and time-stamps.* Thus, our proposed approach will assure that this information is not leaked or compromised.

We consider a semi-honest threat model, i.e., the parties faithfully follow the protocol, but they may try to gain more information through the messages they send or/and receive. We also assume the parties do not collude with each other; in particular, the PNS site does not collude with any bank. Thus, we will develop and establish the privacy guarantees of our method under the simulation model; namely, they do not gain more information than what is explicitly defined as their input, output, and anything that can be inferred using them (for details see Section IV). Furthermore, we use differential privacy [3] to protect the sensitive information, e.g., various time frequencies (how many transactions are initiated by a sender per unit time) in the classification model, which could be released or shared with parties other than PNS.

## B. APPROACH FOR THE CENTRALIZED SETTING

Let us first discuss how our approach works if all the data was available at one site. Algorithm 1 gives an overview the approach in this centralized setting. Below we give details for each of the steps in Algorithm 1 except for the last one, which simply runs a learning algorithm over the provided differentially private augmented data to obtain a classifier.[9]

**1) ACCOUNT-LEVEL (`Flag`-BASED) FEATURE MINING—**Recall that the aim is to leverage the account-level (meta) data with transaction data to improve the overall detection. To do this, we perform feature mining that focuses on capturing information about anomalous (or normal) behavior of transactions as it pertains to account-level data.

We conceptually think of anomalous transactions having two types: *intrinsic* anomalies and *complex* anomalies. We use *account-level rules* (driven from account-level data) to define intrinsic anomalies—for brevity, we refer to them as *rules*. These rules are solely based on the account level data, and they are used to assess whether a transaction is invalid, i.e., it is an intrinsic anomaly. For instance, the rule that ‘money cannot be wired to an account that does not exist’ is an example of (account-level) rule. Clearly as per the rule, a transaction for a beneficiary account that doesn’t exist is an intrinsic anomaly. Thus, we say *a transaction is an intrinsic anomaly if it is anomalous with high probability (i.e.,* $\approx 1$*) when it fails to satisfy one of the account-level rules*. Note that without the account-level features (derived from account-level data) intrinsic anomalies cannot necessarily be distinguished from normal records.

A *complex* anomaly, on the other hand, is an anomaly that is not an intrinsic anomaly. Namely, complex anomalies can satisfy all the account level rules; when they do not, this information (i.e., violation of some rules) alone is not sufficient to identify them—though it

[9]For the PETs challenge, we evaluated the performance of many classifiers and found that XGBoost outperformed other models for the anomaly detection task and therefore utilized it.

can be necessary to do so. Therefore, building a high-accuracy classifier to identify complex anomalies requires training data consisting of the features derived from both the PNS's data (i.e., $\mathrm{PNS}(\mathcal{T})$) and the banks' data (i.e., 1-`BankData`,...,$N$-`BankData`).

Let us now describe how we mine the account-level, i.e., `Flag` based, rules from the data, which have sufficient support. These rules will be used to create account-level features (which will later be augmented to PNS's data). We consider a pair of `Flag` values, $(f, f')$, where $f$ is for the ordering account, and $f'$ is for beneficiary account. Naturally, each transaction $\tau$ is linked to such a pair, and the pair to the account level rule. Since there are $|\mathrm{Flag}|^2$ many pairs (|`Flag`| =number of possible flag values), we only want the pairs with sufficient support. We call this support *anomaly-ratio* and denote it as $r(f, f')$), defined by (1):

$$r(f, f') = \frac{|\{\tau \in S(f, f') \mid \mathrm{Label}(\tau) = 1\}|}{|S(f, f')|} \tag{1}$$

where $S(f, f')$ is the set of all transactions in $\mathcal{T}$ where ordering and beneficiary `Flag` value are $f$ and $f'$, i.e., $S(f, f') = \{\tau \in \mathcal{T} \mid \mathrm{oFlag}(\tau) = f \text{ and } \mathrm{bFlag}(\tau) = f'\}$.

To account for the rules that depend only on the beneficiary account (or only on the ordering account), we consider $(*, f)$ (or $(f, *)$), where $*$ indicates the independence with respect to the ordering (or beneficiary) account respectively.

To mine the rules, we find all the relevant `Flag`-pairs. To do this, we divide the rules into three categories based on the anomaly-ratios. The first category corresponds to the ratios that are close to zero (or below a threshold, e.g., 0.1), and is ignored completely. The second category—called *simple* rules—corresponds to the ratios that are close to one (e.g., greater than 0.9), resulting in rules to identify intrinsic anomalies. The third category—called *complex* rules—corresponds to the rest not covered by the first or the second categories. The specific thresholds can be specified by the domain experts or empirical analysis. Let `m − Features` denote the set of all the pair corresponding corresponding to simple and complex rules given by the mining process.

**<u>How to achieve privacy:</u>** While mining `Flag`-based features, we guarantee differential privacy (DP) [3] in computing $r(f, f')$ (note we discussed DP in Section II-E). We use $(\varepsilon_1/2)$-DP Laplace mechanism [3] to independently perturb both the numerator ($|S(f, f')|$) and the denominator (the number of anomalies in $S(f, f')$) for each $r(f, f')$. Thus, guaranteeing overall $\varepsilon_1$-DP in this process (which follows from serial and parallel composition of DP[10]).

[10]Note that every $(f, f')$ pair covers a different partition of transaction data, i.e., there are no overlaps ion data while computing anomaly-ratios for different `Flag` pairs, and therefore, parallel composition can apply, giving an overall bound of $\varepsilon_1$.

### 2) BANK AND ACCOUNT-LEVEL FEATURE ENCODING VIA BLOOM FILTERS

—We have two types of account level features that we want to encode: 1) account information (e.g., account number, name, address, etc.) and 2) mined features/rules given by the pairs in `m − Features`. We need 1) so that we can validate whether the specified account-level details in a transaction are correct otherwise the transaction cannot go through and will be considered anomaly (note that this will correspond to simple anomaly and is clearly independent of the `Flag` values). Solving this problem while protecting privacy is a significant obstacle. Next, these encoding, i.e., derived features, are augmented to the PNS's transactions data, $\mathscr{T}$.

Since for each transaction $\tau$ in $\mathscr{T}$ and every pair $(f,\ f')$ in `m − Features`, either $\tau \in S(f,\ f')$ or $\tau \notin S(f,\ f')$, we encode the mined features as membership relations using bloom filters (described in Section II-F), shortly called *filter* (described in detail below). We use bloom filters for their superior performance for rule-validation and the security and privacy properties they provide [23], [24]; additionally, we use them to develop a privacy-preserving encoding and membership evaluation protocol in the federated setting (presented shortly).

We first use a simple example to describe our encoding scheme and then give general methods for different scenarios. Consider $(*,\ f')$: it is encoded by adding to a filter all the accounts with `Flag` value $f'$. Now, for a given transaction and a filter (encoding a rule), we check if the beneficiary account is in the filter: if it is then the filter's output is 1 (i.e., it satisfies the rule for being likely anomalous) and otherwise 0.

After the rules have been encoded, we use the filter's output, corresponding to each feature/rule, as the feature's value and augment it to PNS's data, $\text{PNS}(\mathscr{T})$ — these new features are rule-based features (as they are computed by evaluating the rules). Let $\text{PNS}(\mathscr{T}) +$ denote the PNS data augmented with the account-level features.

Below, we provide three approaches to support a variety of `Flag`-based rule encoding via bloom filters. *First*, we can encode and validate multiple account-related data elements, e.g., number, name, street, country, in one filter. Let `acc_info` contain all the necessary account-level information for an account, i.e., for our setting: `acc_info = acc# ∥ name ∥ street ∥ country`, where ∥ denotes concatenation. Now, instead of the account number (as done earlier) for each account, add its `acc_info` to the filter. This allows validation of the complete account information via one filter.

*Second*, consider simple rules like $(f_1,\ *),\ \ldots,\ (f_a,\ *)$ (or $(*,\ f_1),\ \ldots,\ (*,\ f_a)$). We can encode all of them via one filter by adding `acc_info` for all the accounts whose `Flag` value is in $\{f_1,\ \ldots,\ f_a\}$.[11] To evaluate the rule, we just check if the given `acc_info` is in the filter; and if it is, then the filter's output is 1, meaning that the account has a feature value that likely leads to its transaction being anomalous, and the output is 0 otherwise. We note

[11]Note that a simple naming convention (which explicitly specifies ordering or beneficiary in a bloom filter's name) can be used to distinguish a filter encoding for the beneficiary account from that of the ordering account.

that all the rules in the provided data are of this form; thus one bloom filter is sufficient to capture all the rules.

*Third*, consider more general rules, given by a pair of `Flag` values, i.e., $(f,\ f')$. To encode such a rule, create a filter-pair $(BF,\ BF')$: $BF$ is for the ordering account and $BF'$ is for beneficiary account. Then, to the filter $BF$, we add the account information, `acc_info`, for each account with $\text{Flag} = f$; and to the filter $BF'$, we add `acc_info` for all the accounts with $\text{Flag} = f'$. To evaluate the rule, $(f,\ f')$, we check if `acc_info` for the ordering account is in the filter $BF$ with its output being $b\ \ (\in \{0,\ 1\})$, and that of beneficiary account is in the filter $BF'$ with its output being $b'$. Now, if $2b' + b = 3$ then the transaction is likely anomalous.

From storage and computation complexity, we emphasize that all *account-level feature based rules will lead to smaller sized filters as the flagged accounts, i.e., having the account's status that is anomaly prone, only constitute a tiny proportion of the total accounts.*

**3) EXTRACTING TRANSACTION-LEVEL FEATURES AND ASSURING DP FOR SENSITIVE FEATURES—**Once PNS's data is augmented, we can perform feature extraction and evaluation at the PNS site without involving any other party (i.e., the banks). Here, one can use any befitting learning/classification algorithm and the evaluation metrics. This flexibility is indeed one of the powerful features of our approach as it allows for a continuous improvement by using or incorporating more advanced or newly developed learning and data mining methods without incurring any additional privacy cost or any additional burden on any of the other parties.

**Algorithm 2:**

Privacy-Enhanced Two-Step Federated Learning.

```
Input: PNS($\mathscr{T}$) at PNS; $i-\text{Bank}(\mathscr{T})$ and $i$-BankData (all the account-level data) at $i$-Bank for $i = 1, \ldots, N$; key size, $\lambda$; differential privacy (DP) parameters, $\varepsilon_1,\ \varepsilon_2$
Output: Classifier $\mathscr{C}^+$ at PNS
  FIRST STEP: Secure rule mining and encoding
  1: All banks collaboratively securely generate PRF's, $F$, key $k^*$ (of length $\lambda$)
  2: All banks and PNS securely and collaboratively mine Flag-based rules using Flag values from $i-\text{Bank}(\mathscr{T})$ and Label in PNS's data with $\varepsilon_1$-DP guarantee
  3: All banks collaboratively run secure sum protocol to estimate the number of accounts (across all banks) to be added to each bloom filter (BF) and compute the BF's size
  4: for $i = 1, \ldots, N$ do
  5:     For each rule, i-Bank initializes the local filter and adds to it the secure encoding, i.e., $E[\text{acc\_info}] = F_{k^*}(h(\text{acc\_info}))$, of all the accounts from $i$-BankData that conform to the rule (i.e., have specified Flag values)
  6:     Send all local BFs to PNS
  7: end for
SECOND STEP: Secure aggregation, augmentation, and learning
  8: For each rule, PNS aggregates all the received local filters into a global filter
```

```
9: PNS gets the secure encodings of the ordering and beneficiary accounts for each transaction by sending their hashes to the sender bank.
10: PNS uses the global filters and the received secure encodings to compute values of the Flag-based features and augments them to its training data.
11: $PNS$ extracts features (with $\varepsilon_2$-DP guarantee) from the augmented data and learns $\mathscr{C}^+$
```

**How to achieve privacy:** Firstly, we aim to use non-sensitive data that are strong predictors. For account-level (meta) data based feature, we directly rely on the augmented features (which, in federated setting, will be derived using BF-based privacy-preserving encoding). To protect privacy of the sensitive features (namely, features containing sensitive information or derived using some sensitive information), we use generalization (e.g., hours instead of actual time) as well as attribute differential privacy (discussed in Section II-E). For example, a sensitive feature that is computed using counts related to sensitive information can be made differentially private by perturbing the counts using Laplace mechanism; whereas for averages or mean (or other functions) that have higher 'sensitivity' (i.e., the magnitude of change to be hidden, known as global sensitivity in DP literature [3]), smooth sensitivity based DP methods can be used for the perturbation [25].

## C. PRIVACY-ENHANCED ANOMALY DETECTION VIA TWO-STEP FEDERATED LEARNING

We now describe how our approach can be extended from the centralized setting to the federated setting. A distinguishing feature of our federated approach is to leverage the knowledge PNS has about the transactions to design a protocol that, on the one hand, improves the accuracy of classification, and on the other, reduces the leakage of sensitive information (Section IV) for more details). Thus, we choose PNS as the central server/aggregator site as well. Even if we choose a different site for the central server, the information-leakage to PNS will remain the same. To extend our approach to privacy-enhanced federated learning, we need to perform the following three main tasks:

1. Securely learn `Flag`-based (i.e., account-level) rules from banks' data and PNS's data.
2. Make the bloom filter based encoding (to be shared with PNS) secure as well as privacy-preserving so that PNS cannot carry out attacks (e.g., brute force or attacks similar to using rainbow tables) to obtain account information encoded in the filters.
3. Securely extract the values for `Flag`-based features for all the transactions in the training data and augment the training data with these features' values.

Tasks #1 and #2 are completed in the first step of the learning process, it requires banks to mine account-level features/rules, and build a privacy-preserving global bloom filter for PNS via federated learning. Once PNS is provided with the bloom filters, it needs to carry out task #3, which it does by asking the sender bank of each transaction to provide secure encoding of the ordering and beneficiary account information for the transactions — note that PNS cannot perform this on its own, a privacy feature of our approach; after receiving the secure encodings, PNS populates the `Flag`-based (i.e., account-level) features, augments them to its data, and learns the classifier (by following other steps discussed in the centralized setting, detailed in the previous section)—this constitutes the second step. Thus, our approach is named two-step federated learning and is outlined in Algorithm 2. Fig. 4 gives an overview of the proposed solution in a simpler way.

We reemphasize that even in the federated setting, once PNS has obtained augmented data, it is free to choose any learning method and feature extraction method to build the classifier, $\mathscr{C}+$, for anomaly detection. Thus, our approach is flexible and agile, *where PNS can keep improving its model and features to build a better classifier without imposing on banks to carry additional computational burden or risk privacy of their customers by sharing more information.*

**1) SECURE AND PRIVACY-PRESERVING FEATURE/RULE MINING—**The banks use a secure protocol (given in [26]) to count and compare the number of anomalies to identify rules. For instance, for a rule $(f, f')$, the banks securely compute if the anomaly-ratio $= |a(f, f')|/|S(f, f')| > p/q$ (where $p \leq q$); here $a(f, f')$ is the set of anomalies in $S(f, f')$ (the set of all transactions with ordering and beneficiary accounts' `Flag` as $f$ and $f'$). This, for example, can be done by computing, $E[q \cdot |a(f, f')|]$ and $E[p \times |S(f, f')|]$ (where $E[\cdot]$ means the value inside is not accessible, e.g., it may be encrypted or consists of distributed random shares) and compare them without ever revealing the the actual values. Note that for $|S(f, f')|, q > 0$,

$$\text{anomaly} - \text{ratio} > p/q \text{ if and only if } q \cdot |a(f, f')| - p \cdot |S(f, f')| > 0$$

To achieve differential privacy, we need to perturb the counts $|a(f, f')|$ and $|S(f, f')|$ in a way that no one party is able to remove noise from the counts. Let $|a(f, f')| = \sum_{i=1}^{N} |a_i(f, f')|$ and $|S(f, f')| = \sum_{i=1}^{N} |S_i(f, f')|$, where $|a_i(f, f')|$ and $|S_i(f, f')|$ are the corresponding counts at the $i$-Bank. Now, we choose two banks, let's say $i$-Bank and $j$-Bank, at random to perturb their counts via $\varepsilon_1$-DP Laplace mechanism. Now, only $i$-Bank or $j$-Bank can remove the added noise, but each one can remove the noise that it added and not the noise added by the other bank. Hence, the counts are always guaranteed to be differentially private. To ensure that PNS doesn't learn anything about the results of rule mining, *the results of comparison are only shared with the banks and never with PNS.*

As noted earlier *for the provided dataset in the PETs challenge, rule mining is completely unnecessary since: simple rules are fixed and all complex anomalies correspond to non-flagged accounts* (as discussed in the implementation part of Section III-B1). Then, the next task in the first step is for banks to carry out secure encoding of the `Flag`-based features, which is explained next.

**2) PRIVACY-PRESERVING RULE ENCODING AND AGGREGATION—**Here, we discuss the second important task in the 1st step of our approach. This consists of creating secure encoding for the `Flag`-based features (i.e., account-level data) using pseudorandom function (PRF, e.g., AES block cipher), $F$, as well as a cryptographic hash function, $h$ (e.g., sha3), using the secure encodings to construct local bloom filters at each bank's site, and then performing aggregation of secure local bloom filters into global bloom filters at PNS's site.

To accomplish the aforementioned, the banks need to collaboratively pick a random key, $k^* \in \{0, \quad 1\}^{\lambda}$, for the PRF while keeping it secret from PNS (or any other party). A PRF, $F$, takes two inputs, (randomly picked[12]) key, $k^*$, and message, $m$; and its output is depicted as $F_{k^*}(m)$. PRF ensures that an adversary who has some messages and their corresponding PRF outputs still cannot predict the output for a new message that it has not yet seen [27]. Additionally, all the banks and PNS need to know the size of bloom filter (corresponding to each `Flag`-based feature) that depends on the error rate $(\kappa)$ one is willing to tolerate and the total number of elements to be added to the filter, e.g., total number of valid non-flagged accounts across all banks. Thus, they have to securely compute the estimates (e.g., in the upper-bound sense) of the total number of accounts, across all banks, to be added to each bloom filter without revealing the number of accounts for any one bank as well. Below, we provide details on how to do this.

**Secure encoding:** Firstly, all the banks need to *securely generate the key*, $k^*$, of length λ: Each $i$-Bank randomly picks a string $k_i$ of length λ and shares it with all the other banks. With this, each bank can get $k^* = k_1 \oplus k_2 \oplus \cdots \oplus k_N$, where $\oplus$ is for bit-wise XOR operation.

Secondly, the banks use the secure sum protocol described below to compute the total number of accounts to be added to a bloom filter as follows. Let's say $n_i(f)$ is the upper-bound estimate of the accounts in $i$-Bank that have $\mathrm{Flag} = f$, and $p$ be a sufficiently big prime number (e.g., greater than any 32-bit long positive number). Now, 1-Bank picks a random number $r \in \{0, \ 1, \ldots, p-1\}$ and sets $\mathrm{sec\_sum}_0 = r$. Next, for $i = 1, \ldots, N$, $i$-Bank computes $\mathrm{sec\_sum}_i = \mathrm{sec\_sum}_{i-1} + n_i(f)$ mod $p$ and sends $s_i$ to $(i+1)$-Bank ($N$-Bank sends $\mathrm{sec\_sum}_N$ to 1-Bank). 1-Bank computes $\mathrm{sum} = \mathrm{sec\_sum}_N - r$ mod $p$ and shares it with all the banks. Banks use `sum` and pre-agreed upon error rate, $\kappa$, to compute the size of the filter. It is easily possible to modify this protocol to make it resistant to collusion between parties as well.

Then, *each bank builds its local secure bloom filters and adds to them the secure encoding of the relevant accounts* (as per the rules following the method discussed in in Section III-B2 for the centralized setting). The name of the bloom filters bear no association with the value of `Flag`, which can simply be achieved by using PRF as a block cipher under CBC (cipher block chaining) mode. This ensures that the correspondence between rules and the filters remains secret. Secure bloom filter assures that when PNS (or any party or an adversary) receives only the secure bloom filter, it cannot carry out an attack (e.g., by brute force or something similar to rainbow tables) to figure out which accounts are in the bloom filter and which are not. To build a secure bloom filter, instead of adding the account information, `acc_info`, to the bloom filter, we add $E[\mathrm{acc\_info}] \quad = F_{k^*}(h(\mathrm{acc\_info}))$ to the bloom filter. Note that given access to such a filter—but not to $F_{k^*}$—one cannot check if an account is

[12]Note that randomly picking a binary string means picking a string uniformly at random from the set of all possible strings of a specified length.

in the filter even if the adversary knows the account information—this follows from the security properties of PRF [27].

**Secure aggregation:** Since each (bloom) filter consists of indexed bits, we can think of each local filter as a binary string of length $\alpha$: Let's say, for $i = 1, \ldots, N$, $x_i$ is the binary string of $i$-Bank corresponding to its secure local bloom filter for a rule. Then, the global filter for the same rule is: $y = x_1 \vee x_2 \vee \cdots \vee x_N$ (where $y_j = x_{1j} \vee x_{2j} \vee \cdots \vee x_{Nj}$). Note that this does not affect the correctness of the bloom filter because the the filter is initialized with all bits as zero and then any bit (in the bit array of the filter) corresponding to an index where a value is hashed is set to one.

Thus, we aggregate/combine all the local filters corresponding to one feature into a global bloom filter by OR-ing the bits at the same indices. To do this, secure local bloom filters are sent to PNS who carries out the aggregation.

Note that by using pseudorandom function before adding the account information to the bloom filters, we stop PNS from carrying out attacks on the filters. *Our approach ensures that PNS—despite having access to the filters—cannot evaluate any of the rules on a new account without a bank's permission, which stop attacks from the PNS side*.

**3) FEATURE AUGMENTATION AND LEARNING THE CLASSIFIER—**Once all the filters have been aggregated, PNS collects all the transactions, $\mathrm{PNS}(\mathscr{T})_i$, originating from each bank, $i$-Bank for $i = 1, \ldots, N$; then for each transaction in $\tau' \in \mathrm{PNS}(\mathscr{T})_i$, PNS sends the hash of the account info., i.e, $h(\mathrm{acc\_info})$, for the ordering and beneficiary accounts of $\tau'$ to $i$-Bank; in response $i$-Bank sends secure encoding, i.e., $F_{k^*}(h(\mathrm{acc\_info}))$, for each of these accounts.

After PNS receives the secure encodings for all the (ordering and benefeciary) accounts, it evaluate rules via the secure global filters, and uses the results to augment its training data.

Now, PNS carries out the same steps as detailed in the centralized setting to learn the $\mathscr{C}^+$.

## IV. PRIVACY ANALYSIS

We consider the *semi-honest adversarial model*, that is, each party (bank or PNS) will follow the given protocol (i.e., the distributed/federated algorithm) but may try to gain more information from the intermediate data/messages it receives. For the basic solution, we assume that none of the parties collude.

This restriction can however be relaxed by using threshold homomorphic encryption [28]. Privacy is ensured both in terms of the federated protocol as well as the model computed to find anomalies. A formal proof using the simulation paradigm is given below to show that the protocol does not reveal any additional information beyond that explicitly stated. Since differential privacy is used to protect account-level feature mining as well as the sensitive features created by PNS, the overall model is protected.

## IV. IDEAL PARADIGM

There is a fully trusted third party, T-party, whom PNS sends its data, i.e., $\text{PNS}(\mathcal{T})$, and each *i*-Banks sends *i*-`BankData` (all the account-level data that it has). T-party carries out the computation as described in the centralized setting and returns:

- *to PNS*: $I_1\ :(n,\ n')$ for each mined rule $(RF(f),\ RF(f'))$ (with anomaly-ratio $> \gamma$ for the rule $(f,\ f')$),[13] where where $RF$ denotes random function, and $n$ (or $n'$) gives the number of accounts with $\text{Flag} = f$ (or $f'$), and when $f$ (or $f'$) is $*$ the corresponding count is zero. $I_2$ : Values of the augmented features, named as $(f)\ \|\ RF(f')$. $I_3$ : Additionally, for each $RF(f)\ \|\ RF(f')$, as leakage, PNS also receives additional feature values corresponding to $RF(f')\ \|\ RF(f)$.[14] Note that in our implementation for the challenge, as part of $I_2$, PNS only gets the total number of non-flagged accounts and the corresponding feature values (let's say 1 if ordering account info. is incorrect or the account is flagged, 2 if the same is true for the beneficiary account, 3 if it's true for both, and 0 otherwise). Thus, for our implementation for the challenge $I_2$ and $I_3$ will be the same, and there will be no additional leakage.

- to i-Bank: $I_1\ :(n,\ n')$ for each mined rule $(f,\ f')$ (with anomaly-ratio $> \gamma$). $I_2$ : list of the ordering and beneficiary accounts for the transactions initiated by the *i*-Bank (note that this information is always known to the bank that initiates a transaction).

## IV. REAL PARADIGM

The real paradigm corresponds to the execution of two-step federated learning method, which reveals the following information to the parties:

- to PNS: $R_1\ :(n,\ n')$ for each mined rule $(f,\ f')$ (with anomaly-ratio $> \gamma$) with feature names as $F_{k^*}(f)\ \|\ F_{k^*}(f')$, where $F_{k^*}$ is pseudorandom function, $F_{k^*}$, used as a block cipher in CBC mode. $R_2$ : bloom filters for each $F_{k^*}(f)\ \|\ F_{k^*}(f')$ (which have been populated using $F_{k^*}$ and $h$ for the relevant accounts by all the banks). $R_3$ : Secure encodings of the ordering and beneficiary accounts for each transactions initiated by the *i*-Bank for $i = 1,\ \ldots,\ N$.

- *to i-Bank:* $R_1\ : k_j$ (a randomly picked string of length λ) (from *j*-Bank). $R_2\ :(n,\ n')$ for each mined rule $(f,\ f')$ $R_3$ : Partial secure sum, i.e., $\text{sec\_sum}_{i-1}$ for each mined rule $(f,\ f')$. $R_4$ : cryptographic hashes of the ordering and beneficiary accounts for the transactions initiated by the *i*-Bank.

We will now prove that the view of a non-colluding probabilistic polynomial time (PPT) *semi-honest* adversary controlling any party in the real paradigm can be simulated by the **view** of the same party in the ideal paradigm; i.e., the two views, one in the ideal

[13]This is computed with $\varepsilon_1$-differentially privacy guarantee.

[14]This accounts for PNS swapping the ordering accounts with beneficiary accounts.

paradigm and the other in the real paradigm, are computationally indistinguishable to any PPT adversary (for more details, we refer the reader to [29]). Note that here, by *view* we mean all the information that a party has including its input, results, and intermediate messages received as well as all information it can construct (in polynomial time) from this.

### Indistinguishability of PNS's view:

Let us first look at an adversary controlling PNS in the ideal paradigm. $I_1$ and $R_1$ are the same. Because PRF's output is (computationally) indistinguishable from that of RF's, $RF(f) \parallel RF(f')$ is indistinguishable from $F_{k^*}(f) \parallel F_{k^*}(f')$ for the adversary.

We now show how the adversary can simulate the view of PNS from real paradigm for $R_2$ and $R_3$. For each feature, $RF(f) \parallel RF(f')$, the adversary initializes a pair of bloom filters $(\hat{BF},\ \hat{BF}')$ of sizes given by $(n,\ n')$ and $\kappa$, which are known to PNS and the adversary. Then using the values of the feature, $RF(f) \parallel RF(f')$, given by $I_2$, it adds the account info. to the relevant BFs. For example, if for a transaction, $RF(f) \parallel RF(f') = 1$ then it adds the ordering account's info., i.e., `acc_info`, as $RF(h(\text{acc\_info}))$ to $\hat{BF}$ if $RF(f) \parallel RF(f') = 2$ then it adds the beneficiary account's info. to $\hat{BF}'$, and if $RF(f) \parallel RF(f') = 3$ then it adds the ordering account's info. to $\hat{BF}$ and the beneficiary account's info. to $\hat{BF}'$ respectively. Furthermore, it keeps counts of total number of accounts added to each bloom filter, let's say $z$ and $z'$ are respectively the counts for $\hat{BF}$ and $\hat{BF}'$; then for $M = n - z$ and $j = 1,\ \ldots,\ M$, the adversary adds $RF(j)$ to $\hat{BF}$, and for $j = M + 1,\ \ldots,\ M + n' - z',\ RF(j)$ to $\hat{BF}$. The adversary does the same for $I_3$ but swaps the info. of ordering account with that of the beneficiary account. The same procedure is repeated for each of the augmented feature.

$R_3$ can be simulated by considering all the accounts info. added to the bloom filters (i.e., $RF(h(\text{acc\_info}))$'s) from the above. The simulatability claim of the view follows from the indistinguishability of PRF from RF.

### Indistinguishability of i-Bank's view.

$R_1$ can be simulated by generating a randomly picked string of length $\lambda$. $I_1$ and $R_2$ are the same. As for $R_3$, i.e., $\text{sec\_sum}_{i-1}$, since $r$ is uniformly distributed over $\{0,\ \ldots,\ p - 1\}$, so is $\text{sec\_sum}_{i-1}$. Thus, the $i$-Bank in real paradigm gets the sum, e.g., $n$ or $n'$, when $i = 1$, and uniformly distributed $\text{sec\_sum}_{i-1}$ when $i \neq 1$. When in ideal paradigm the adversary is controlling 1-Bank, it knows the corresponding sum from $I_1$, otherwise it can generate an (distributionally) indistinguishable $\text{sec\_sum}_{i-1}$ by picking a random number. Lastly, $R_4$ can be generate by hashing the account given in $I_2$. Thus, we have shown that even at the bank level, the adversary can simulate the view. This completes the proof.

**A. INEVITABLE INFORMATION LEAKAGE—**We now describe and analyze the information leakage that is inevitable; this is due to the fact that using all banks' data together with PNS's data, i.e., `Flag` values, results in a better classifier, $\mathscr{C}^+$. *We argue that over time a party who is given $\mathscr{C}^+$ to classify transactions can learn a probabilistic association of* `Flag` *values for accounts.*

Note that $\mathscr{C}^+$ is better because it can identify more anomalies than $\mathscr{C}$, a classifier learned only over data without `Flag` related features (e.g., PNS's data). In particular, $\mathscr{C}^+$, with high probability, can identify all the anomalies that $\mathscr{C}$ can identify. But there are additional anomalies that $\mathscr{C}^+$ can identify (which are missed by $\mathscr{C}$); this is due to the fact that `Flag` —which is directly or indirectly used by $\mathscr{C}^+$—is useful in identifying these additional anomalies. This information can be used to infer probabilistically correct associate of `Flag` values for an account.

**attack I:** Noting the aforementioned, let's say a party, $P$ (e.g., PNS), is given $\mathscr{C}^+$ to classify transactions. Overtime, $P$ gathers transaction and compile a dataset $P(\mathscr{T}_n)$ of $n$ transactions. $P$ remove from $P(\mathscr{T}_n)$ the features related to `Flag`, and use this dataset to learn a classifier $\mathscr{C}$.

Next, $P$ uses $\mathscr{C}^+$ and $\mathscr{C}$ to label all transactions. Any transaction that $\mathscr{C}^+$ labels as anomalous but $\mathscr{C}$ labels as normal, is anomalous due to `Flag` value of the account being used (directly or indirectly) in $\mathscr{C}^+$. Furthermore, for each such anomaly, $P$ inspects the paths of the decision tree(s) of $\mathscr{C}^+$ that are taken by these anomalies and how they compare to other anomalies and normal records. Using this information and the data from $P(\mathscr{T}_n)$, $P$ estimates the distribution, giving the probabilistic association of `Flag` values, for the accounts present in the transactions in $P(\mathscr{T}_n)$.

**Ways to reduce this leakage:** *One*, use a secure and distributed version of $\mathscr{C}^+$ so that no one party can access $\mathscr{C}^+$, and the classification result is only revealed to the relevant banks. This, however, will be extremely slow and will increase the transaction processing time undesirably. *Two*, randomize values for `Flag`. For instance, instead of `Flag`, use $E[\text{Flag}]$, a randomized form of `Flag`, obtained by encrypting all the values of `Flag` with semantically secure encryption. This will hide the precise values of the `Flag` but the distribution will still be inevitably revealed. Thus, the two approaches give a way to trade-off security and utility.

Let us now look at the information leakage for PNS and banks and possible attacks to extract more information.

**PNS's knowledge:** PNS knows $\text{PNS}(\mathscr{T}_n)$. Among other information, PNS can learn: (1) what's the valid/invalid account details for an account; (2) which transactions were processed successfully and which ones failed; (3) which banks an account belongs to; (4) and does an account's `Flag` value is such that it makes the account's transactions anomalous. We also note that (without any prior knowledge) PNS doesn't know what accounts a bank has if it has not appeared in a transaction from $\text{PNS}(\mathscr{T}_n)$. Below we outline the attacks that can help PNS gain some of the non-obvious information mentioned above.

**attack II:** The account details for a successful transaction are correct. If a transaction is failed (i.e., anomalous) due to invalid info., we assume that with high probability, it will be made again. Thus, by comparing new transactions with the older ones (e.g., using edit distance over the ordering and beneficiary account details and Euclidean distance over the amount, time etc.), PNS can infer, with high probability, what are the valid and invalid details for an account.

**attack III:** Let us say the classifier $\mathscr{C}^{+}$ (learned over $\mathrm{PNS}(\mathscr{T}_n)+$, i.e., the data consisting of transaction level and `Flag`-based features) is given to PNS to identify anomalies. Now, using its data, i.e., $\mathrm{PNS}(\mathscr{T})$, and the strategy outlined in attack I, PNS can obtain probabilistic association of `Flag` values for accounts.

**i-Bank's knowledge:** Each $i$-Bank knows $i$-`BankData` (account level data for its accounts), $i\mathrm{Bank}(\mathscr{T}_n)$ (all transactions for which $i$-Banks was sender or receiver bank). Note that for the accounts in transactions in $i\mathrm{Bank}(\mathscr{T}_n)$, $i$-Bank can also carry out *attack II*. Furthermore, the bank can also carryout attack I. This is possible even when $i$-Bank doesn't have $\mathscr{C}^{+}$. $i$-Bank can build an approximation of $\mathscr{C}^{+}$ using its account-level data and use it instead of $\mathscr{C}^{+}$ to carry out *attack I*.

## V. EXPERIMENTAL EVALUATION AND RESULTS

### A. EXPERIMENTAL SETUP

**1) DATASET—**The dataset for evaluation is a synthetic dataset provided by the PETs competition organizer. The dataset contains approximately three million synthetic transaction records and account data from a payment network system and a group of banks. The percentage of anomaly transactions is 1%. As described in Section II-C and II-D, the transaction data is hosted at a payment network system client, which contains features such as the transaction amount, transaction frequency, and the amount for the currency used in each transaction. The account data, which contains account meta information are horizontally partitioned and hosted by multiple bank clients. The provided account data has one important account-level feature, `Flag`, which is finite and indicates the status of the account. Details of the dataset and the feature prepossessing can be found in the appendix.

**2) EVALUATION METRIC—**The metric used for evaluating the effectiveness of the models is *Area under the Precision-Recall Curve (AUPRC)*, also known as *Average Precision (AP)*. This is a commonly used metric for anomaly detection problems, as it focuses more on the anomalies than the negative class (normal instances). AUPRC is computed as follows:

$$\mathrm{AUPRC} = \sum_{n} (R_n - R_{n-1}) P_n$$

where $P_n$ and $R_n$ are the precision and recall, respectively, when thresholding at the $n$-th individual transaction sorted in order of increasing recall.

**3) METHODS FOR EVALUATION—**We evaluate the following solutions on the provided dataset: **PNS Only:** The solution only uses PNS data, specifically PNS client purely employs the Xgboost model on transaction-level data without using Bank's accounts information such as `Flag`. **Centralized (PNS+Bank):** Our proposed centralized solution, as detailed in Section III-B. **Federated (PNS+Bank):** Our proposed federated solution, as detailed in Section III-C. The evaluation of centralized and federated solutions considers variations in two critical parameters - the acceptable false negative ratio $\kappa$ of the bloom filter, where a lower value gives more accuracy for the bloom filter but requires increased capacity (memory) and computational costs, and the privacy budget, i.e., the value of $\varepsilon > 0$, for differential privacy applied on dataset features (smaller values of $\varepsilon$ provide stronger privacy protection but at the expense of utility such as the effectiveness of the model).

We also compare our solution with the other solutions in the PETs Competition's leaderboard. The evaluation of PETs competition is under one centralized scenario and three federated scenarios with different numbers of bank clients and data splitting.

**4) IMPLEMENTATION DETAILS—**The experiments are implemented in Python, leveraging the Flower framework [30] for federated learning execution, the Xgboost library for model development, and pycryptodome library for all cryptographic operations. Within these experiments, the federated learning environment, including the simulation of multiple clients and communication between clients, was emulated using Flower's built-in simulation module with a multi-processing approach so that experiments could be conducted on a single machine. The machine for running experiments contains a 16-core Intel Core i7 CPU and 32 GB of memory. Security parameters for the system's encryption and decryption, such as the RSA Public key, AES session key, and XOR key, are set to a length of 32.

## B. RESULTS

**1) EFFECTIVENESS OF PROPOSED METHODS—**Fig. 5 shows the final average precision (AP) scores of the anomalous transaction detection task on the dataset and setup where there are four bank clients.

Fig. 5(a) compares the PNS Only solution with Centralized solutions with different differential privacy budgets. As demonstrated in Fig. 5(a), by integrating PNS transaction-level data with Bank account-level data, there is an approximate 6 percent increase in the average precision (AP) score (from 0.91 to 0.97). We note that the impact of different DP budgets on the Centralized solution is not drastic, which may be attributed to the weak correlation of the sensitive attributes overall as well as more pronounced differences between the marginal distribution of anomalies compared to normals.

Fig. 5(b) compares the Centralized and Federated solutions under a DP budget of 1, across varying bloom filter false negative rates for the federated solution. As demonstrated in Fig. 5(b), our proposed federated Solution can achieve a similar average precision (AP) score as the centralized approach under different bloom filter false negative rates, thereby validating the effectiveness of our federated solution.

The overall evaluation results demonstrate that our proposed method can provide excellent detection ability for anomalous transactions in both centralized and federated settings.

**2) EFFICIENCY AND SCALABILITY—**We now highlight the efficiency and scalability of our methods. For this, we measured the overall runtime of clients and server in minutes

(**Time**), the peak memory usage across all the clients during the training phase in gigabytes (**Peak Memory**), and the total size of resources used in communication between clients and server in gigabytes (**Network**).

Table I presents runtime, peak memory usage, and network usage for each party under the scenario where there are four bank clients. The proposed method demonstrates efficiency for both PNS and bank clients, capable of completing tasks within 20 minutes while consuming less than 10 GB of memory. Fig. 6 shows the total runtime, peak memory usage, and network usage under scenarios with different numbers of bank clients. The results from Fig. 6 revealed that our method maintains efficiency as the number of clients increases. This is primarily due to the fact that model fitting, which constitutes the most computationally intensive bottleneck, occurs exclusively on PNS's end. Hence, our approach proves to be highly scalable, adeptly handling the expansion in the number of partitions without significant efficiency degradation.

**3) COMPARISON WITH OTHER SOLUTIONS IN PETS CHALLENGE—**As summarized in Table II, we compare our method against the other solutions from the PETs competition. Our approach can achieve the high average precision (AP) score in the centralized setting, with results of 0.97 AP score close to the PPMLHuskies by a marginal difference of 0.006. Across three distinct federated learning scenarios, our method attained an average precision (AP) score of 0.97, slightly higher than PPMLHuskies and significantly surpassing other leading solutions, such as those from Visa and ILLIDAN. Notably, during the third phase of the PETs competition, which involved adversarial attack testing of various methods, our approach emerged as the winner in the final assessment. However, due to the organizers' decision not to disclose the adversarial test details, those specific results cannot be displayed here. The PETS Challenge results further support the effectiveness and robustness of our solution.

## VI. RELATED WORK

### Financial crime detection.

Financial crime detection has been extensively studied in both research and industry. Traditional approaches primarily rely on rule-based strategies, utilizing human prior knowledge [33]. However, these methods struggle to handle the complexity of patterns and the volume of data. Machine learning and data mining emerged as data-driven approaches, playing a crucial role in this field [34]. The focus of this paper is closely aligned with fraud detection, a typical outlier identification task with highly imbalanced datasets. A substantial body of work in this area included supervised methods like Random Forest and Support Vector Machine, and unsupervised methods such as DBSCAN and Isolation Forest. Deep learning-based anomaly detection has recently drawn great interest due to its ability to handle complex, high-dimensional data, as demonstrated in initiatives like FEAWAD [35] and REPEN [36]. We refer readers to Adbench [37] for a comprehensive evaluation.

Graph and network analysis have also been established as important tools in capturing fraud characteristics in vast financial transaction datasets, where nodes and edges represent entities such as companies, individuals, and transactions [38], [39]. In [40], the authors utilized both labeled data (users are labeled as fraud or not) and unlabeled data (social relations and attributes) and proposed SemiGNN to exploit the representations of users. In [41], the authors developed a group-aware (gang-aware) graph neural network-based approach (GAGNN); in particular, they designed a community-centric encoder to transform the original transactions into graphs and then encoded the graph using both topological and attribute-wise information, the embedding then fed into the prediction layer. Unlike the methods above that assume data centralization, this work aims to identify fraudulent wire transactions in payment networks, where transaction information is distributed among multiple organizations.

### Privacy-preserving federated anomaly detection:

Federated learning-based anomaly detection, allowing multiple clients to collaboratively train a global model without pooling local datasets, is a promising solution to improve outlier detection performance in various applications [42], [43]. Despite its advantages, FL systems were susceptible to several attacks and could compromise the privacy of local datasets [44]. Existing research to address privacy and security concerns investigates integrating privacy-enhancing technologies like Differential Privacy, Homomorphic Encryption, and Secure Multi-party Computation with federated anomaly detection. In [45], the authors developed XORBoost, a multi-party computation-based federated gradient boosted tree model; while this approach can provide a security guarantee, the complexity grows quadratically with the number of nodes and linearly with the number of samples and features, bringing huge computation bottlenecks considering the scale of transactional data. In [46], the authors developed a differentially private Gradient Boosted Decision Tree (GBDT), relying on secure aggregation and differential privacy to guarantee privacy. However, this work focused on horizontally federated settings and is not directly applicable to mixed settings.

### PETs Prize Challenge solutions:

For the PETs Challenge, the authors from [31] proposed a solution leveraging a hybrid of vertical and horizontal federated learning. This approach involved banking clients locally, training an autoencoder on account information, and sharing the embeddings with a central server. The transaction client, such as a Payment Network, then retrieved these embeddings and integrated them with transaction features to train classification models. Gaussian noise was added to the concatenated features to protect data privacy from model inversion and membership inference. To protect model sharing from account clients to the server, account clients encrypt the embedding before sharing with the server. This method achieved an AUPRC score of approximately 0.67 per the PETs leaderboard. However, it presented a risk of leaking private account information. Similar to our work, the solution from Visa Research [32], focusing on “flag” attribute exclusive to banks, employed multi-party computation and noisy aggregations. This approach attained an AUPRC score of around 0.52 per the PETs leaderboard but introduced significant computational overheads.

## VII. DISCUSSIONS AND CONCLUSION

This article introduces a novel privacy-preserving method for payment network systems (PNS) and banks to collaborate and leverage their data to enhance the detection of fraudulent wires using anomaly detection. Our method works in two steps. In the first step, bank clients collaboratively mine account-level features/rules and securely share this information with the PNS, utilizing secure Bloom Filters and secure aggregation. In the subsequent step, the PNS generates additional features by querying the secure aggregated Bloom Filter and trains a classifier on the augmented datasets. Further, to guard against model inversion and membership inference attacks, the PNS protects sensitive features using a differentially private mechanism.

For the PETs challenge, our implementation focused on flag-based rule mining, leveraging the unique “flag” data field held by banks. This approach perfectly suits the challenge when flag information significantly contributes to classification performance, yet it can also be extended to other account-level data. For example, banks can formulate more sophisticated flag-based rules using anomaly ratios, creating additional Bloom Filters and features. Additionally, banks can train local classifiers to determine transaction-associated account risk scores, and encode this information via secure Bloom Filters. The PNS can then use this score data for final predictions, either by generating additional features or through an ensemble approach.

Our secure Bloom Filters and aggregation protocols have broader applications, including in multi-party private set union problems and Federated Submodel Learning (FSL), where group membership is sensitive and needs to be protected. For instance, in FSL, a machine learning model is divided into multiple submodels based on different data types used to train model parts. Users interact with only the relevant submodel for their local data, minimizing communication costs. However, updated submodel indices and values can inadvertently reveal data types that the user has. To preserve user privacy during FSL, both indices and updated values must remain private during reading and writing. Our proposed Bloom Filters and secure aggregation can be incorporated to address these privacy concerns, as demonstrated in [47].

Our solution demonstrates state-of-the-art performance on the PETs challenge datasets, excelling in both accuracy and scalability by leveraging informative rule mining and shifting the computational load to the PNS (or server) side. Furthermore, the drop is performance going from centralized solution to its federated counterpart is negligible, which stands in contrast with other approaches. However, it encounters two limitations. Firstly, the data-dependent nature of rule mining, while effective for account flag data, complicates the extraction of informative rules from other features. To address this, future work will explore data-driven approaches, such as neural networks, to extract valuable representations. Secondly, the efficiency gained from using Bloom Filters is contingent on the availability of informative rules. A significant number of rules may lead to increased communication overhead during the secure aggregation. To tackle this issue, we are considering alternatives like the counting Bloom Filter [48], which may offer a more scalable solution. Moreover, the deployment in real-world applications needs to address challenges such as client dropout, which requires further exploration in future research.

Our research primarily targets outlier detection within the bank-PNS network, therefore, operating under the assumption that the entities involved are non-colluding and semi-honest. However, adapting our methodology to withstand covert or malicious adversaries presents additional challenges. Such adversaries, including potentially malicious banks or those compromised by an attacker, can undermine the classifier's accuracy by sharing local Bloom filters poisoned by incorrectly labeled data. Although existing methods [49], [50] offer strategies to counter such malicious clients, integrating these techniques into our framework needs further investigation. Furthermore, accurate yet private anomaly detection may require the use of models such as sensitive privacy [51], [52], which we plan to explore in the future.

## Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

## ACKNOWLEDGMENT

The content is solely the responsibility of the authors and does not necessarily represent the official views of the agencies funding the research.

This work was supported in part by the National Institutes of Health under Grant R35GM134927 and in part by the National Science Foundation under Grant CNS-2333225.

## Biographies

**HAFIZ ASIF** received the Ph.D. degree from Rutgers University, Newark, NJ, USA. He is currently an Assistant Professor with the Information Systems and Business Analytics Department, Hofstra University, Hempstead, NY, USA. His research is developing SAFE (secure & private, auditable, fair, and equitable) theoretical approaches and practical frameworks to leverage the power of data to solve real-world problems.

**SITAO MIN** is currently working toward the Ph.D. degree with the Department of Management Science and Information Systems, Rutgers University, Newark, NJ, USA. His research interests include data management, data quality, missing data mitigation, and federated learning.

**XINYUE WANG** received the Ph.D. degree from Rutgers University, Newark, NJ, USA. She is currently developing privacy-preserving methods to generate high-fidelity synthetic data, and analyze genomic data.

**JAIDEEP VAIDYA** (Fellow, IEEE) is currently a Distinguished Professor of computer information systems with Rutgers University, Newark, NJ, USA. He has authored or coauthored more than 200 papers in international conferences and journals. His research interests include privacy, security, data management, and data analytics. He is a Fellow of the AAAS and IFIP, and an ACM Distinguished Scientist.

## REFERENCES

[1]. Goldreich O, "Secure multi-party computation," Manuscript. Preliminary Version, vol. 78, no. 110, 1998.

[2]. Gentry C, "Fully homomorphic encryption using ideal lattices," in Proc. 41st Annu. ACM Symp. Theory Comput., 2009, pp. 169–178.

[3]. Dwork C, "Differential privacy: A survey of results," in Proc. Int. Conf. Theory Appl. Models Computation, 2008, pp. 1–19.

[4]. Goldreich O and Oren Y, "Definitions and properties of zero-knowledge proof systems," J. Cryptol, vol. 7, no. 1, pp. 1–32, 1994.

[5]. Assefa SA, Dervovic D, Mahfouz M, Tillman RE, Reddy P, and Veloso M, "Generating synthetic data in finance: Opportunities, challenges and pitfalls," in Proc. 1st ACM Int. Conf. AI Finance, 2020, pp. 1–8.

[6]. McMahan B, Moore E, Ramage D, Hampson S, and y Arcas BA, "Communication-efficient learning of deep networks from decentralized data," in Proc. Artif. Intell. Statist, 2017, pp. 1273–1282.

[7]. Sabt M, Achemlal M, and Bouabdallah A, "Trusted execution environment: What it is, and what it is not," in Proc. IEEE Trustcom/BigDataSE/Ispa, 2015, pp. 57–64.

[8]. Kairouz P et al. , "Advances and open problems in federated learning," Found. Trends Mach. Learn, vol. 14, no. 1–2, pp. 1–210, 2021.

[9]. Vasa J and Thakkar A, "Deep learning: Differential privacy preservation in the era of Big Data," J. Comput. Inf. Syst, vol. 63, no. 3, pp. 608–631, 2023.

[10]. PETsPrizeChallenges, "Privacy-enhancing technologies prize challenges." 2022. Accessed: Jan. 30, 2024. [Online]. Available: https://petsprizechallenges.com/

[11]. Asif H, Min S, Wang X, and Vaidya J, "Scarlet pets implementation code" 2023. [Online]. Available: https://github.com/usnistgov/PrivacyEngCollabSpace/tree/master/tools/de-identification/Scarlet-PETs

[12]. United Nations Office on Drugs and Crime, "Money laundering." Accessed: Jan. 30, 2024. [Online]. Available: https://www.unodc.org/unodc/en/money-laundering/overview.html

[13]. Cipriani M, Goldberg LS, and La Spada G, “Financial sanctions, swift, and the architecture of the international payment system,” J. Econ. Perspectives, vol. 37, no. 1, pp. 31–52, 2023.

[14]. Data Driven, “Transforming financial crime prevention through federated learning with end-to-end privacy,” 2022. [Online]. Available: https://ktn-uk.org/wp-content/uploads/2022/08/PETs-Prize-Challenges_-Financial-Crime-Prevention-Technical-Brief-1.pdf

[15]. Liu Y et al. , “Federated forest,” IEEE Trans. Big Data, vol. 8, no. 3, pp. 843–854, Jun. 2022.

[16]. Abourayya A et al., “Protecting sensitive data through federated co-training,” 2023.

[17]. Bater J, Elliott G, Eggen C, Goel S, Kho A, and Rogers J, “SMCQL: Secure querying for federated databases,” Jun. 22, 2016, arXiv:1606.06808.

[18]. Scott SV and Zachariadis M, The Society for Worldwide Interbank Financial Telecommunication (SWIFT): Cooperative Governance for Network Innovation, Standards, and Community. London, U.K.: Taylor & Francis, 2014.

[19]. Kifer D and Machanavajjhala A, “No free lunch in data privacy,” in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2011, pp. 193–204.

[20]. Hua Y, Xiao B, Veeravalli B, and Feng D, “Locality-sensitive bloom filter for approximate membership query,” IEEE Trans. Comput, vol. 61, no. 6, pp. 817–830, Jun. 2012.

[21]. Deng F and Rafiei D, “Approximately detecting duplicates for streaming data using stable bloom filters,” in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2006, pp. 25–36.

[22]. Gomez-Barrero M, Rathgeb C, Galbally J, Fierrez J, and Busch C, “Protected facial biometric templates based on local gabor patterns and adaptive bloom filters,” in Proc. IEEE 22nd Int. Conf. Pattern Recognit., 2014, pp. 4483–4488.

[23]. Luo L, Guo D, Ma RTB, Rottenstreich O, and Luo X, “Optimizing bloom filter: Challenges, solutions, and comparisons,” IEEE Commun. Surveys Tuts, vol. 21, no. 2, pp. 1912–1949, Secondquarter 2019.

[24]. Bianchi G, Bracciale L, and Loreti P, ““better than nothing” privacy with bloom filters: To what extent?,” in Proc. Int. Conf. Privacy Stat. Databases, 2012, pp. 348–363.

[25]. Nissim K, Raskhodnikova S, and Smith A, “Smooth sensitivity and sampling in private data analysis,” in Proc. 39th Annu. ACM Symp. Theory Comput., 2007, pp. 75–84.

[26]. Niu C et al. , “Secure federated submodel learning,” Nov. 6, 2019, arXiv:1911.02254.

[27]. Bellare M, Canetti R, and Krawczyk H, “Pseudorandom functions revisited: The cascade construction and its concrete security,” in Proc. IEEE 37th Conf. Found. Comput. Sci., 1996, pp. 514–523.

[28]. Paillier P, “Public-key cryptosystems based on composite degree residuosity classes,” in Proc. Int. Conf. Theory Appl. Cryptographic Techn., 1999, pp. 223–238.

[29]. Lindell Y, “Secure multiparty computation for privacy preserving data mining,” in Proc. Encyclopedia Data Warehousing Mining, 2005, pp. 1005–1009.

[30]. Beutel DJ et al. , “Flower: A friendly federated learning research framework,” Jul. 28, 2020, arXiv:2007.14390.

[31]. Zhang H, Hong J, Dong F, Drew S, Xue L, and Zhou J, “A privacy-preserving hybrid federated learning framework for financial crime detection,” Feb. 7, 2023, arXiv:2302.03654.

[32]. Arora S et al. , “Privacy-preserving financial anomaly detection via federated learning & multi-party computation,” Oct. 6, 2023, arXiv:2310.04546.

[33]. Sudjianto A, Nair S, Yuan M, Zhang A, Kern D, and Cela-Díaz F, “Statistical methods for fighting financial crimes,” Technometrics, vol. 52, no. 1, pp. 5–19, 2010.

[34]. Chen Z, Van Khoa LD, Teoh EN, Nazir A, Karuppiah EK, and Lam KS, “Machine learning techniques for anti-money laundering (aml) solutions in suspicious transaction detection: A review,” Knowl. Inf. Syst, vol. 57, pp. 245–285, 2018.

[35]. Zhou Y, Song X, Zhang Y, Liu F, Zhu C, and Liu L, “Feature encoding with autoencoders for weakly supervised anomaly detection,” IEEE Trans. Neural Netw. Learn. Syst, vol. 33, no. 6, pp. 2454–2465, Jun. 2022. [PubMed: 34170831]

[36]. Pang G, Cao L, Chen L, and Liu H, “Learning representations of ultrahigh-dimensional data for random distance-based outlier detection,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2018, pp. 2041–2050.

[37]. Han S, Hu X, Huang H, Jiang M, and Zhao Y, “ADBench: Anomaly detection benchmark,” in Proc. Adv. Neural Inf. Process. Syst, 2022, pp. 32142–32159.

[38]. He X, Vaidya J, Shafiq B, Adam N, and Atluri V, “Preserving privacy in social networks: A structure-aware approach,” in Proc. IEEE/WIC/ACM Int. Joint Conf. Web Intell. Intell. Agent Technol., 2009, pp. 647–654.

[39]. He X, Shafiq B, Vaidya J, and Adam N, “Privacy-preserving link discovery,” in Proc. ACM Symp. Appl. Comput., 2008, pp. 909–915.

[40]. Wang D et al., “A semi-supervised graph attentive network for financial fraud detection,” in Proc. IEEE Int. Conf. Data Mining, 2019, pp. 598–607.

[41]. Cheng D, Ye Y, Xiang S, Ma Z, Zhang Y, and Jiang C, “Anti-money laundering by group-aware deep graph learning,” IEEE Trans. Knowl. Data Eng, vol. 35, no. 12, pp. 12444–12457, Dec. 2023.

[42]. Li L, Fan Y, Tse M, and Lin K-Y, “A review of applications in federated learning,” Comput. Ind. Eng, vol. 149, 2020, Art. no. 106854.

[43]. Silva PR, Vinagre J, and Gama J, “Towards federated learning: An overview of methods and applications,” Wiley Interdiscipl. Rev.: Data Mining Knowl. Discov, vol. 13, no. 2, 2023, Art. no. e1486.

[44]. Yin X, Zhu Y, and Hu J, “A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions,” ACM Comput. Surv, vol. 54, no. 6, pp. 1–36, 2021.

[45]. Deforth K, Desgroseilliers M, Gama N, Georgieva M, Jetchev D, and Vuille M, “XORBoost: Tree boosting in the multiparty computation setting,” in Proc. Privacy Enhancing Technol, 2022.

[46]. Maddock S, Cormode G, Wang T, Maple C, and Jha S, “Federated boosted decision trees with differential privacy,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur, 2022, pp. 2249–2263.

[47]. Niu C et al., “Billion-scale federated learning on mobile clients: A submodel design with tunable privacy,” in Proc. 26th Annu. Int. Conf. Mobile Comput. Netw., 2020, pp. 1–14.

[48]. Guo D, Liu Y, Li X, and Yang P, “False negative problem of counting bloom filter,” IEEE Trans. Knowl. data Eng, vol. 22, no. 5, pp. 651–664, May 2010.

[49]. Tolpegin V, Truex S, Gursoy ME, and Liu L, “Data poisoning attacks against federated learning systems,” in Proc. Comput. Secur. 25th Euro. Symp. Res. Comput. Secur., 2020, pp. 480–501.

[50]. Hong Y, Vaidya J, and Lu H, “Secure and efficient distributed linear programming,” J. Comput. Secur, vol. 20, no. 5, pp. 583–634, 2012.

[51]. Asif H, Papakonstantinou PA, and Vaidya J, “How to accurately and privately identify anomalies,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., 2019, pp. 719–736.

[52]. Asif H, Vaidya J, and Papakonstantinou PA, “Identifying anomalies while preserving privacy,” IEEE Trans. Knowl. Data Eng, vol. 35, no. 12, pp. 12264–12281, Dec. 2023. [PubMed: 37974954]

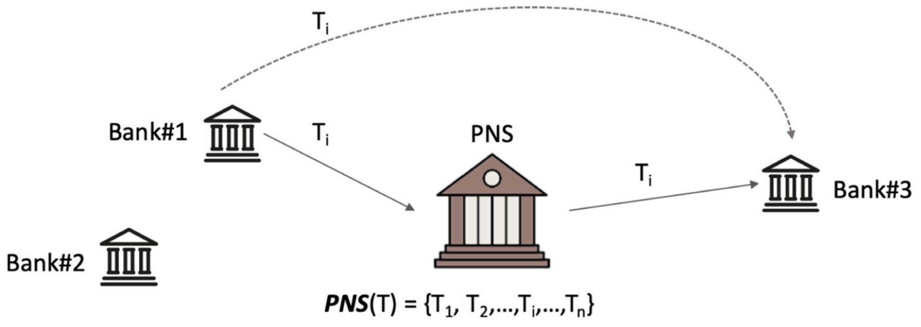
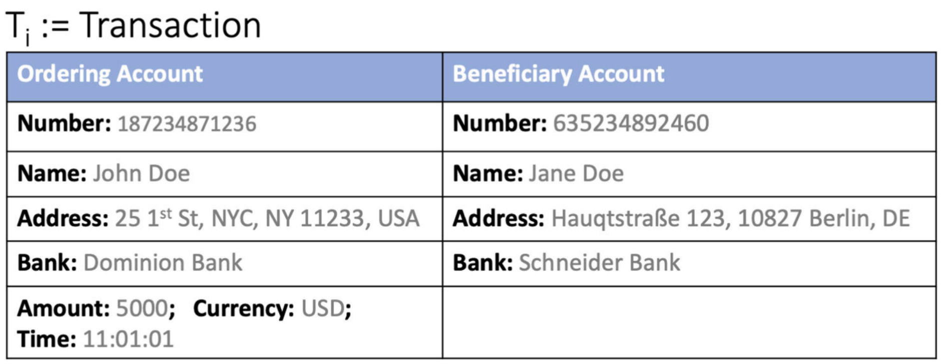

$T_i$
Bank#1
$T_i$
PNS
$T_i$
Bank#3
Bank#2
$\boldsymbol{PNS}(T) = \{T_1, T_2, \ldots, T_i, \ldots, T_n\}$

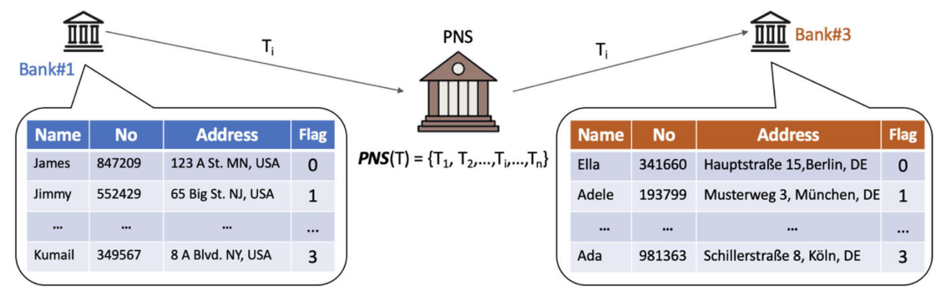
**FIGURE 1.**
Basic setup of Payment Network Systems (PNS). The dotted line depicts the transaction that we want to make from Bank#1 and Bank#3; while the solid line depicts how the transaction goes through the PNS. PNS(T) denotes the data consisting of all the transactions at PNS.



$T_i$ := Transaction

| **Ordering Account** | **Beneficiary Account** |
|---|---|
| **Number:** 187234871236 | **Number:** 635234892460 |
| **Name:** John Doe | **Name:** Jane Doe |
| **Address:** 25 1st St, NYC, NY 11233, USA | **Address:** Hauqtstraße 123, 10827 Berlin, DE |
| **Bank:** Dominion Bank | **Bank:** Schneider Bank |
| **Amount:** 5000; **Currency:** USD; **Time:** 11:01:01 | |

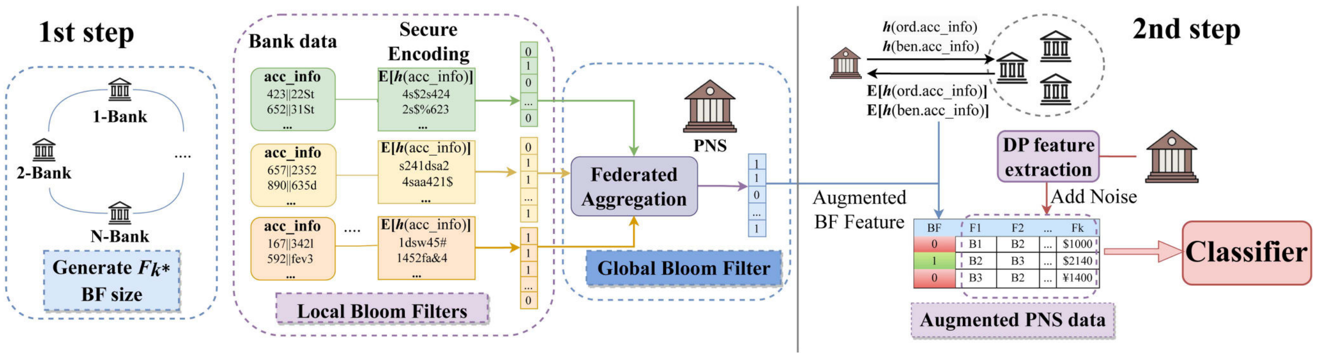
**FIGURE 2.**
Example transaction from NYC to Berlin.



| Name | No | Address | Flag |
|---|---|---|---|
| James | 847209 | 123 A St. MN, USA | 0 |
| Jimmy | 552429 | 65 Big St. NJ, USA | 1 |
| ... | ... | ... | ... |
| Kumail | 349567 | 8 A Blvd. NY, USA | 3 |

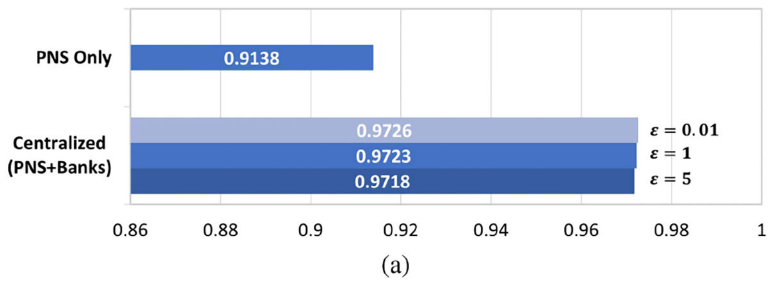
| Name | No | Address | Flag |
|---|---|---|---|
| Ella | 341660 | Hauptstraße 15,Berlin, DE | 0 |
| Adele | 193799 | Musterweg 3, München, DE | 1 |
| ... | ... | ... | ... |
| Ada | 981363 | Schillerstraße 8, Köln, DE | 3 |

**FIGURE 3.**
Account metadata at banks connected via PNS.



| BF | F1 | F2 | ... | Fk |
|---|---|---|---|---|
| 0 | B1 | B2 | ... | $1000 |
| 1 | B2 | B3 | ... | $2140 |
| 0 | B3 | B2 | ... | ¥1400 |

**FIGURE 4.**
2-Step FL overview with a simpler setting (without rule mining).



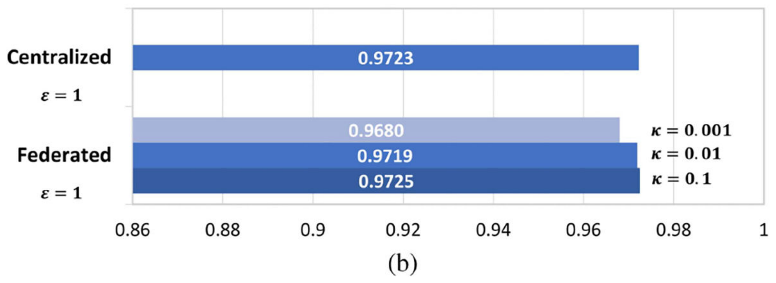


**FIGURE 5.**
Effectiveness w.r.t AUPRC of Proposed Solution. We consider there are four bank clients. For parameters of the proposed solution, $\kappa$ **denotes the false negative rate of the bloom filter**, $\varepsilon$ represents the privacy budget in differential privacy. (a) Comparison between PNS Only and Centralized (PNS+Banks) Solution. (b) Comparison between Centralized and Federated Solution.

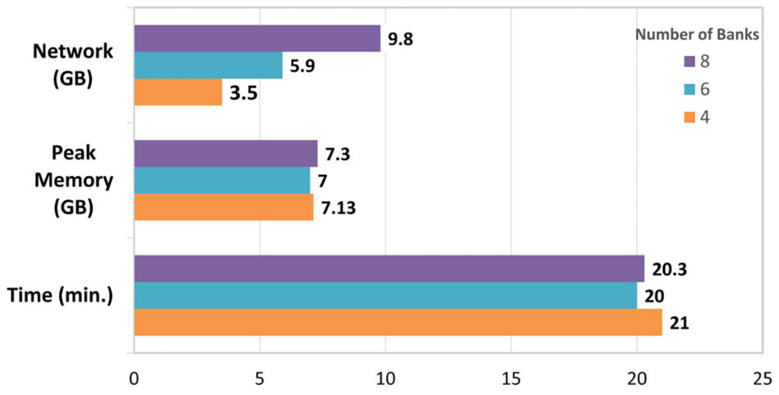


**FIGURE 6.**
Scalability of Proposed Federated Solution.

**TABLE I**

Efficiency of Proposed Federated Solution Under Four Banks Scenario

| | Efficiency | | | | |
|---|---|---|---|---|---|
| | **PNS** | **Bank1** | **Bank2** | **Bank3** | **Bank4** |
| **Time (Minutes)** | 17 | 0.58 | 0.60 | 0.68 | 0.67 |
| **Peak Memory (GB)** | 7.13 | 0.34 | 0.34 | 0.67 | 0.38 |
| **Network (GB)** | 1.44 | 0.32 | 0.31 | 0.72 | 0.71 |

## TABLE II

Evaluation From Leaderboard

| | **Centralized** | **Federated** | | |
|---|---|---|---|---|
| **Method** | **AP** | **AP@N1** | **AP@N2** | **AP@N3** |
| ILLIDAN Lab [31] | 0.4457 | 0.6784 | 0.6466 | 0.6466 |
| PPMLHuskies | 0.9801 | 0.9494 | 0.9610 | 0.9477 |
| Visa [32] | 0.7949 | 0.5219 | 0.5121 | 0.4575 |
| **Ours** | 0.9741 | 0.9751 | 0.9748 | 0.8941* |