SMURF: Privacy-Preserving Multi-Party Graph Mining for AML

Dhruvan Gnanadhandayuthapani[1]
Supervisor: Dr. Zekeriya Erkin[1]
[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 21, 2026

Name of the student: Dhruvan Gnanadhandayuthapani
Final project course: CSE3000 Research Project
Thesis committee: Dr. Zekeriya Erkin, Dr. Nezihe Merve Gürel

Abstract

Money launderers frequently obscure illicit funds by routing transactions across multiple financial institutions, creating complex "scatter-gather" (smurfing) patterns. Although these structural patterns are detectable within a centralised transaction graph, privacy regulations like GDPR prevent banks from directly merging their data. Existing privacy-preserving graph frameworks are limited to pairwise protocols that fail to capture laundering chains involving three or more institutions. To address this, we present SMURF (Scatter-gather Mining Using Randomised Functions), a privacy-preserving framework with a five-phase protocol enabling banks to jointly detect cross-institutional scatter-gather patterns without exposing raw account data. By combining Oblivious Pseudorandom Functions (OPRFs) and Homomorphic Encryption (HE), SMURF allows banks to locally extract transaction pairs, generate anonymous identifiers, and securely aggregate counts via a Central Coordinator entirely in the encrypted domain before collaboratively decrypting final totals to flag suspicious accounts. Empirical evaluations on the synthetic AMLWorld small (HI) dataset with approx. 5 million transactions and 369 scatter-gather accounts show that our system achieves an F_1-score of 82.44% in less than 20 minutes when executed on a 64 core AMD EPYC processor.

1 Introduction

The United Nations Office on Drugs and Crime (UNODC) estimates that the total amount of money laundered globally each year ranges between $800 billion and $2 trillion USD or between 2% and 5% of global GDP [1]. Despite large international efforts, UNODC estimates that less than 1% of these laundered funds are successfully seized or frozen by law enforcement [2]. This extremely low seizure rate allows criminal networks and corrupt officials to seamlessly integrate their proceeds into the legitimate economy, leaving the ordinary citizen to bear the increased tax burden.

The difficulty in intercepting these funds stems from the structural complexity of modern laundering techniques, which typically occurs in three stages: placement, layering, and integration [3]. To detect these complex schemes, financial transactions are increasingly modelled as graphs, where accounts represent nodes, and transactions form directed weighted edges [4, 5, 6]. A prominent graph structure is the *scatter-gather* pattern, illustrated in Figure 1a, which represents a money laundering tactic called "smurfing" [7, 8, 9, 10, 11], in which a large sum of illicit cash is broken down into multiple smaller amounts, scattered to various intermediaries, and finally gathered back into a central account.

In practice, this technique is rarely confined to a single bank, it often spans multiple financial institutions, as shown in Figure 1b, so that the partial subgraph visible to any single institution appears normal [12]. Addressing this requires cross-institutional collaboration. However, centralisation of transaction graphs is not feasible due to privacy regulations such as the GDPR [13] and the Swiss FADP [14]. Although privacy-preserving frameworks such as Collaborative Scatter-Gather Mining (CSGM) [12] avoid centralisation, they only support a two-party setting. Current literature lacks a privacy-preserving framework capable of detecting *scatter-gather* patterns that occur across a general multi-party setting.

Therefore, we formalise this challenge with the following research question: "*How can scatter-gather patterns be detected across multi-party financial networks without revealing sensitive transaction-level data?*"

To address this question, we propose SMURF (Scatter-gather Mining Using Randomised Functions), a privacy-preserving framework with a five-phase protocol that enables financial institutions to jointly detect cross-institutional *scatter-gather* patterns without exposing

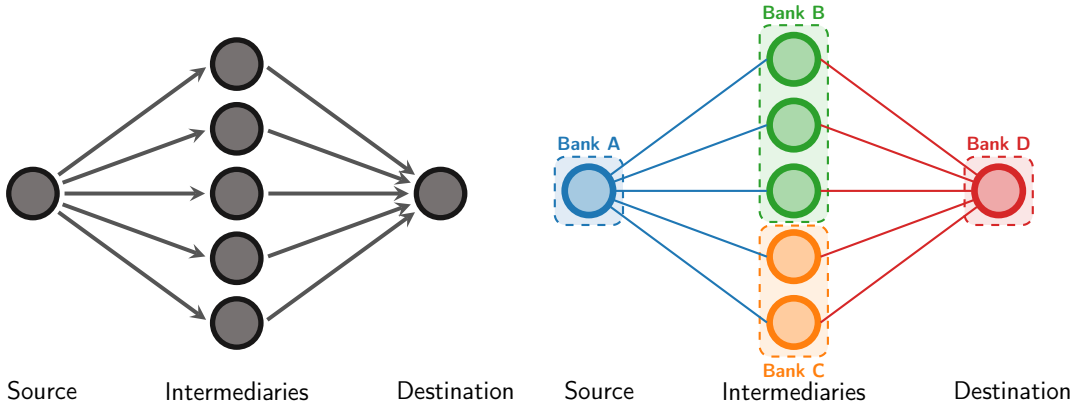

Figure 1: The scatter-gather pattern (a) and its partitioned variant across four banks (b).

raw account data. Under SMURF, banks first locally extract all source-destination pairs connected by internal intermediaries and then mask these raw source-destination identities into anonymous identifiers to privately link them across institutions. Local transaction counts per pair are then encrypted, allowing a Central Coordinator to aggregate global counts entirely in the encrypted domain. Finally, banks collaboratively decrypt only the final totals to score and flag suspicious accounts within their networks.

The remainder of this paper is organised as follows. Section 2 covers prerequisite background knowledge, and Section 3 reviews related work. Section 4 details the problem, the threat model, and the proposed framework, followed by its theoretical analysis in Section 5. Section 6 presents the experimental setup and results. Section 7 discusses these results. Section 8 addresses ethical considerations, and finally Section 9 concludes the paper and discusses the directions for future work.

2 Background

2.1 Additive Homomorphic Encryption

Homomorphic encryption (HE) allows computations to be performed directly on ciphertexts without prior decryption [15]. Because our protocol only requires addition, there is no need for a less efficient fully homomorphic scheme. Therefore, we use the additive homomorphic properties of the Paillier cryptosystem [16]. Let $\mathsf{Enc}_{\mathsf{pk}}(\cdot)$ denote encryption under a public key pk. Given two plaintexts m_1 and m_2, the scheme satisfies

$$\mathsf{Enc}_{\mathsf{pk}}(m_1) \cdot \mathsf{Enc}_{\mathsf{pk}}(m_2) \equiv \mathsf{Enc}_{\mathsf{pk}}(m_1 + m_2) \pmod{n^2}. \tag{1}$$

In order to use the Paillier encryption securely in a multi-party setting, we need to prevent parties from decrypting intermediary values. Therefore, we use a distributed version of the Paillier cryptosystem [17]. In this scheme, the private key sk is shared between the n parties during a setup phase without a trusted dealer [18] using Shamir's secret sharing over the integers [19]. Decryption requires cooperation between any m parties. Each bank computes a partial decryption using its local share. Any set of at least m valid partial decryptions can be combined to recover the full plaintext, while any fewer than m reveals no information about the underlying message.

2.2 Oblivious Pseudorandom Functions (OPRFs)

An Oblivious Pseudorandom Function (OPRF) is a two-party protocol that allows a Client with an input x and a Server with a secret key k to jointly evaluate a pseudorandom function $F(k, x)$ [20, 21]. The Client successfully learns the final output $F(k, x)$ without learning any information about the Server's key k, while the Server remains completely blind to both the Client's input x and the resulting output $F(k, x)$.

An important property of an OPRF is that it is deterministic. Evaluating the protocol multiple times on identical inputs under the same key yields identical outputs. This allows independent parties that execute the protocol with a matching input x to derive a matching shared token $\tau = F(k, x)$. Because τ is generated obliviously under a secret key, it acts as an opaque identifier that reveals no information about the input x to an observer or a compromised server [20, 22].

To implement this efficiently, we use the unverifiable 2HashDH OPRF protocol introduced in [23] as a lightweight alternative to the earlier *verifiable* construction [24], which is also standardised in RFC 9497 [20]. We use an elliptic curve group $\mathbb{G}$ of prime order p, a hash-to-curve function $H_1 : \{0, 1\}^* \to \mathbb{G}$, and a base cryptographic hash function H_2 (e.g., SHA-256 [25]). The Server has a secret scalar $k \in \mathbb{Z}_p^*$. First, the Client blinds its input x by computing $A = r \cdot H_1(x)$ using a randomly sampled scalar $r \in \mathbb{Z}_p^*$, and transmits A to the server. The Server then evaluates its key on the blinded point and returns $B = k \cdot A = k \cdot r \cdot H_1(x)$. Due to the commutativity of scalar multiplication, the Client can unblind this result by computing $C = r^{-1} \cdot B = k \cdot H_1(x)$. Finally, the global pseudorandom identifier is derived using the base hash function as $\tau = H_2(x \parallel C) = F(k, x)$.

3 Related Work

Recent advances in Graph Neural Networks (GNNs) have significantly improved money laundering detection via transaction graph analysis. For example, standard message-passing GNNs adapted for directed multigraphs are able to identify scatter-gather motifs and directed cycles with high precision on centralised datasets [26]. However, these centralised approaches assume complete data visibility, which is legally unfeasible due to data privacy regulations such as GDPR [13] and Swiss FADP [14]. To address data silos, federated graph learning allows institutions to collaborate on AML detection by exchanging embedding representations instead of raw graphs [27]. However, FL is designed to learn generalised statistical behaviours rather than to execute exact structural queries.

Consequently, Secure Multi-Party Computation (SMPC) has been explored for exact deterministic motif matching. The Collaborative Scatter-Gather Mining (CSGM) framework leverages Locality-Sensitive Hashing (LSH) and Bloom filters to detect cross-institutional patterns privately [12]. However, its pairwise Private Set Intersection (PSI) protocol suffers from severe scalability and communication complexities in multi-institution networks [28]. To bypass these pairwise bottlenecks, modern architectures combine cryptographic primitives like Oblivious Pseudorandom Functions (OPRF) and threshold cryptography. For example, secure oblivious transfer protocols can determine intersections of suspicious accounts without revealing the underlying data [29]. Ultimately, a gap remains for deterministic multi-party motif detection: centralised GNNs require full data visibility [26], while FL only works for generalised statistical models [27]. Our framework bridges this gap using oblivious aligned identifiers and secure homomorphic aggregation, enabling the exact identification of cross-institutional scatter-gather patterns across an arbitrary number of banks.

4 Our Proposal

4.1 Problem Definition

Let the global transaction network be defined as a distributed directed graph $G = (\mathcal{V}, \mathcal{E})$. An edge $(i, j) \in \mathcal{E}$ indicates that account i transfers money to account j. Let $\mathcal{B} = \{B_1, B_2, \ldots, B_n\}$ be a finite set of n distinct financial institutions operating within a shared ecosystem. The global graph G is partitioned so that each institution institution $B_i \in \mathcal{B}$ is the owner of a private local vertex set $\mathcal{V}_i \subset \mathcal{V}$, where $\bigcup_{i=1}^{n} \mathcal{V}_i = \mathcal{V}$. To comply with Know Your Customer (KYC) standards, financial institutions are required to gather basic information about both the initiator and the recipient of each transaction [13, 30, 14]. This regulatory rule remains applicable even when accounts are held in different institutions. Therefore, there exists an overlap between the vertex sets of any two interacting banks (i.e., $\mathcal{V}_i \cap \mathcal{V}_j \neq \emptyset$, where $i \neq j$).

For any individual account vertex $v \in \mathcal{V}_k$ managed by an institution, we define its in-neighborhood as

$$N^-(v) = \{\, s \in \mathcal{V}_k \mid (s, v) \in \mathcal{E}_k \,\}, \tag{2}$$

and conversely, we define its out-neighborhood as

$$N^+(v) = \{\, t \in \mathcal{V}_k \mid (v, t) \in \mathcal{E}_k \,\}. \tag{3}$$

Because an institution B_k possesses the full edge visibility for its own accounts, it can locally observe the exact identities of any $s \in N^-(v)$ and $t \in N^+(v)$ that interact with its internal nodes, even if those accounts belong to an external bank's vertex set.

Using these neighbourhoods, we define a scatter-gather candidate with an intermediary node $v \in \mathcal{V}_k$ as a tuple (s, v, t) such that $s \in N^-(v)$ and $t \in N^+(v)$. Globally, a scatter-gather pattern exists between a source account s and a destination account t if they are bridged by a large number of distinct intermediaries. Formally, we define the global intersection count function $I(s, t)$ as

$$I(s, t) = |\{v \in \mathcal{V} \mid s \in N^-(v) \wedge t \in N^+(v)\}|. \tag{4}$$

Given a threshold $\theta \in \mathbb{Z}^+$, the multi-party computational objective is to identify all pairs $(s, t) \in \mathcal{V} \times \mathcal{V}$ that satisfy $I(s, t) \geq \theta$. The problem challenge dictates that this objective must be executed under strict privacy constraints: no institution B_i should learn any individual membership of $N^-(v)$ or $N^+(v)$ held by another institution B_j ($j \neq i$), nor should any central entity learn the raw identities of s, v, or t.

4.2 Threat Model

To evaluate the security and privacy guarantees of our framework, we adopt a *semi-honest* (honest-but-curious) adversarial model [31]. We assume that all participating institutions $B_i \in \mathcal{B}$ strictly follow the protocol steps and execute the cryptographic operations exactly as specified. This model fits distributed financial networks, where participants are legally regulated and heavily audited entities that are unlikely to risk the detection of malicious protocol deviations. However, they remain incentivised to learn any actionable information about their competitors. Consequently, institutions act as passive adversaries that log incoming network messages. A curious participant may analyse these communication transcripts to deduce private transactions or sensitive account-level information belonging to other banks. This model explicitly excludes active attacks, such as injecting forged routing paths, altering transaction weights, or sending malformed payloads.

4.3 Protocol Overview

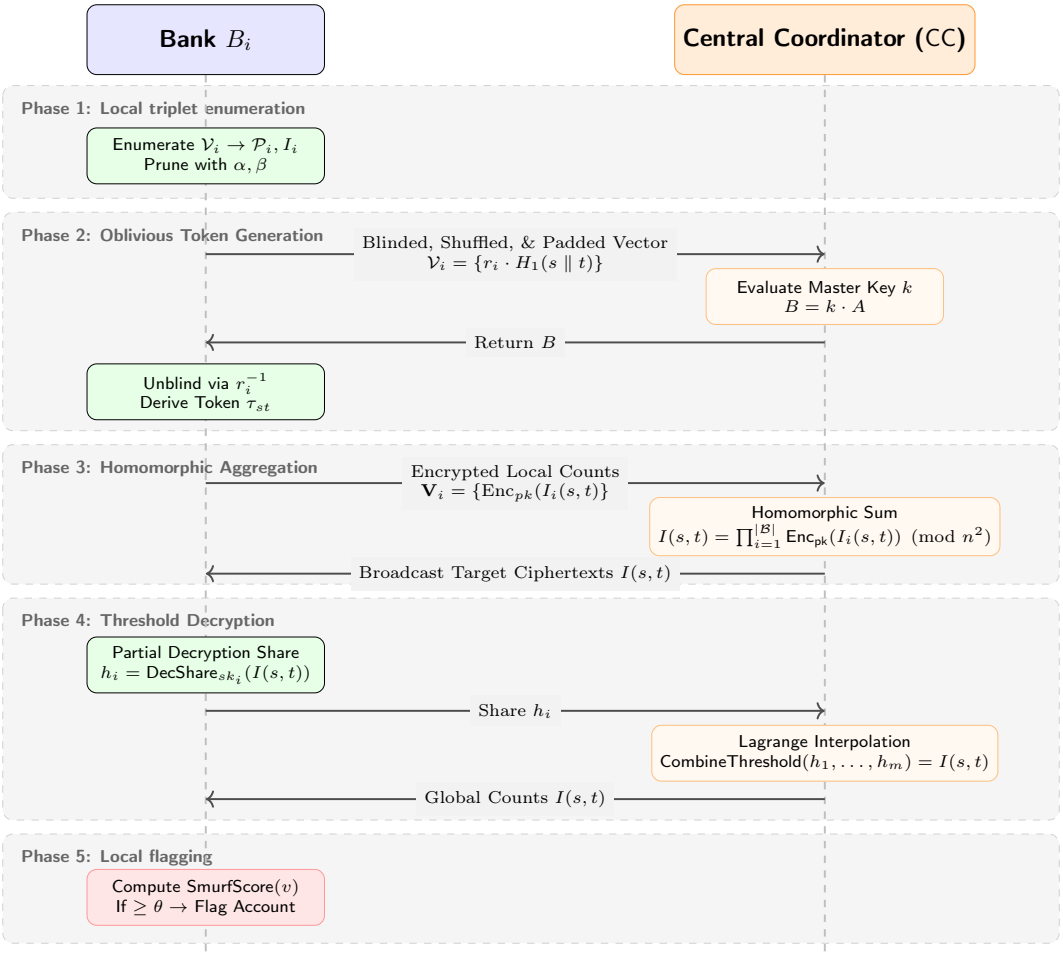

Figure 2: The SMURF framework architecture, depicting the sequential interactions between a bank B_i and the Central Coordinator CC across the five protocol phases.

To securely compute the global interaction count $I(s,t)$ over a distributed network, the SMURF framework (Figure 2) exploits an elegant property of the problem: while individual financial institutions cannot observe the whole pattern, they maintain full visibility over their immediate local neighbourhoods. Specifically, for a vertex $v \in \mathcal{V}_i$, the hosting bank B_i can locally link a preceding source $s \in N^-(v)$ to a succeeding destination $t \in N^+(v)$.

The protocol is executed in five sequential phases. First, in **Local Triplet Enumeration**, institutions locally extract all (s,v,t) transactions connected by internal intermediaries. Next, during **Oblivious Token Generation**, banks and the CC evaluate an OPRF to mask raw (s,t) pairs into anonymous global identifiers. In the **Homomorphic Aggregation** phase, banks encrypt their local counts and send them to the CC, which additively sums them up entirely in the encrypted domain. In the fourth phase, **Threshold Decryption**, banks collaboratively decrypt only these final aggregated totals. Finally, during **Local Flagging**, institutions use these global counts to flag suspicious accounts.

4.4 Local Triplet Enumeration

The protocol begins locally within each individual institution. Each bank B_i parses its local vertex set $\mathcal{V}_i$ to extract localised components of a global scatter-gather motif. Because any hosted account could structurally function as an intermediary, the institution identifies all combinations of valid incoming and outgoing edges across all accounts. The step-by-step procedure is detailed in Algorithm 1. Formally, for a given candidate intermediary $v \in \mathcal{V}_i$, the bank enumerates the set of triplets defined as

$$\mathcal{P}_i = \left\{ (s, v, t) \;\middle|\; v \in \mathcal{V}_i,\ s \in N^-(v),\ t \in N^+(v),\ s \neq t \right\}. \tag{5}$$

Financial transaction graphs are frequently modelled as scale-free networks[1] reflecting the heavy-tailed degree distributions of interbank networks [32, 33]. While evidence for strict scale-free structure in consumer transaction networks is more nuanced [34], high-degree nodes remain a major source of combinatorial explosions during path enumeration.

To ensure computational efficiency across these networks, the enumeration algorithm enforces two pruning bounds. Formally, a candidate vertex v is discarded from evaluation if

$$|N^-(v)| \cdot |N^+(v)| > \alpha \quad \text{or} \quad \max\left(|N^-(v)|, |N^+(v)|\right) > \beta, \tag{6}$$

where α restricts the combinatorial explosion at a single node, and β limits the maximum in- or out-degree. While these bounds exclude a small fraction of highly connected vertices, Appendix D provides empirical validation that varying α and β yields a negligible impact on overall detection performance.

4.5 Oblivious Token Generation

To align cross-institution transaction boundaries (s, t) without disclosing raw account identities or revealing edge distributions across banks, the protocol performs the distributed token generation routine based on the 2HashDH OPRF protocol defined in Section 2.2.

Each bank $B_i \in \mathcal{B}$ maps its local transaction pairs to $P_{st} = H_1(s \parallel t) \in \mathbb{G}$ and blinds it using a private scalar $r_i \in \mathbb{Z}_p^*$. To obscure transaction volume variations from the CC, banks append these entries into a vector $\mathcal{V}_i$ padded with random curve coordinates up to the fixed uniform global capacity N, and then shuffle them before sending it to the CC.

Upon receipt, the CC evaluates its master key scalar $k \in \mathbb{Z}_p^*$ across all slots of the received vectors by computing $k \cdot P$ for each element $P \in \mathcal{V}_i$. Upon receiving the processed vectors back, each bank shuffles its vector a second time. This step breaks the positional linkability of elements, preventing the CC from mapping tokens to their initial positions during the subsequent Homomorphic Aggregation phase. The bank then unblinds the elements using $r_i^{-1} \pmod p$ to yield the point

$$C_{st} = k \cdot H_1(s \parallel t). \tag{7}$$

Finally, each bank computes the globally aligned pseudorandom token using the base hash function H_2 (e.g., SHA-256 [25])

$$\tau_{st} = H_2((s \parallel t) \parallel C_{st}). \tag{8}$$

This token τ_{st} serves as the global opaque identifier for downstream alignment.

[1] A scale-free network is characterized by a degree distribution that follows a power law, $P(k) \sim k^{-\gamma}$, where a few hub nodes possess a disproportionately high number of connections.

Algorithm 1 Local Triplet Enumeration

Require: Private local vertex set $\mathcal{V}_i$, neighbourhoods $N^-(v)$ and $N^+(v)$ for all $v \in \mathcal{V}_i$, threshold parameters α and β
Ensure: Local triplet set $\mathcal{P}_i$, local count map I_i

$\mathcal{P}_i \leftarrow \emptyset$ ▷ Set of all local (s, v, t) triplets
$I_i \leftarrow \emptyset$ ▷ Map pair (s,t) to its count
$\mathcal{V}_{\mathsf{smurf}} \leftarrow \{v \in \mathcal{V}_i \mid N^-(v) \neq \emptyset \wedge N^+(v) \neq \emptyset\}$ ▷ All possible intermediary candidates
for each $v \in \mathcal{V}_{\mathsf{smurf}}$ **do**
  if $|N^-(v)| \times |N^+(v)| > \alpha$ **or** $|N^-(v)| > \beta$ **or** $|N^+(v)| > \beta$ **then**
    continue ▷ Pruning using α and β
  end if

  for each $s \in N^-(v)$ **do**
    for each $t \in N^+(v)$ **do**
      if $s \neq t$ **then**
        $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup \{(s, v, t)\}$
        if $(s, t) \notin \mathsf{keys}(I_i)$ **then**
          $I_i(s, t) \leftarrow 0$
        end if
        $I_i(s, t) \leftarrow I_i(s, t) + 1$
      end if
    end for
  end for
end for
return $\mathcal{P}_i, I_i$

4.6 Homomorphic Aggregation

Once the globally aligned identifiers τ_{st} are established, the protocol securely aggregates the localised counts into global counts. To compute the global counts without exposing individual bank counts, the protocol leverages the additive homomorphic properties of the Paillier cryptosystem defined in Section 2.1

Initially, each bank $B_i \in \mathcal{B}$ maps its local (s, t) count to the unblinded identifier τ_{st}. To preserve privacy against volume-based side-channel leaks, every bank again constructs a uniformly padded vector $\mathcal{V}_i$ of fixed capacity N. For a valid identifier with a local count $I_i(s, t)$, the bank populates the corresponding slot with an encryption $\mathsf{Enc}_{\mathsf{pk}}(I_i(s, t))$. The remaining empty slots up to capacity N with the random curve coordinates are populated with an encryption of zero $\mathsf{Enc}_{\mathsf{pk}}(0)$.

The vectors $\mathcal{V}_i$ are then shuffled and forwarded to the CC. The coordinator then aggregates matching tokens across institutions. When identical tokens τ_{st} collide, it means multiple banks observed the same (s, t) pair. The CC homomorphically sums the counts by multiplying the corresponding ciphertexts

$$I(s,t) = \prod_{i=1}^{|\mathcal{B}|} \mathsf{Enc}_{\mathsf{pk}}(I_i(s,t)) \pmod{n^2} = \mathsf{Enc}_{\mathsf{pk}}\left(\sum_{i=1}^{|\mathcal{B}|} I_i(s,t)\right) \pmod{n^2}. \tag{9}$$

4.7 Threshold Decryption

Following aggregation, the CC holds all the encrypted counts $I(s,t)$ for every observed identifier τ_{st}. To flag the intermediaries, these aggregated counts must be safely decrypted. To safely expose these counts without allowing any single institution to unilaterally decrypt frequencies, the participants execute the distributed threshold decryption protocol introduced in Section 2.1.

When a query is initiated on an aggregated token's ciphertext $I(s,t)$, the CC broadcasts it to all participants. Each bank B_i acts as a partial decryption node, using its privacy key share sk_i to compute a partial decryption share

$$h_i = \mathsf{DecShare}_{\mathsf{sk}_i}(I(s,t))\,. \tag{10}$$

Once a valid threshold m of honest partial decryption shares is achieved, the CC collects and combines them via Lagrange interpolation to reveal the global plaintext count value:

$$I(s,t) = \mathsf{CombineThreshold}\,(h_1, h_2, \ldots, h_m) = \sum_{i=1}^{|\mathcal{B}|} I_i(s,t). \tag{11}$$

Because the unblinded token τ_{st} is pseudonymous, no participant can infer the underlying account identifiers (s,t) during decryption. Additionally, since the master private key remains distributed across institutions, no isolated participant can unilaterally decrypt arbitrary payloads outside the context of the consensus-driven protocol queries.

4.8 Local Flagging

Once the plaintext global counts $I(s,t)$ are unmasked via collaborative threshold decryption, each institution B_i maps these global counts to its local candidates to finalise anomaly detection. Because each identifier τ_{st} is bound to a specific source-destination pair, each bank can evaluate the true global-level context of its internal vertices.

The framework captures the distributed footprint of a scatter-gather laundering pattern by defining the final risk metric of an internal candidate account $v \in \mathcal{V}_{\mathsf{smurf}}$ as the maximum path recurrence observed across its entire local triplet set $\mathcal{P}_i$

$$\mathsf{SmurfScore}(v) = \max_{(s,v,t)\in\mathcal{P}_i} I(s,t). \tag{12}$$

By sorting internal candidate nodes in descending order according to their computed $\mathsf{SmurfScore}(v)$, each bank can construct a prioritised investigation queue. Accounts whose scores exceed a global threshold parameter θ are flagged as high-confidence illicit smurfing intermediaries. This step completes the protocol.

5 Analysis

In this section, we formally analyse the proposed framework across two dimensions: the theoretical complexities (computation and communication) as well as the privacy guarantees ensured by the underlying cryptographic primitives under the semi-honest model.

5.1 Complexity Analysis

The framework's performance overhead is predominantly dictated by the cryptographic workload rather than the graph traversal itself. To hide traffic volumes, all participants pad their candidate arrays to a uniform maximum capacity N. Consequently, the complexities scale relative to this uniform padding size N rather than the absolute number of transactions, as summarised in Table 1.

Table 1: Complexity Summary For Each of the Five Protocol Phases

	Computational Complexity		**Communication**
Protocol Phase	**Per Bank B_i**	CC	**Per Bank B_i**
Local Enumeration	$\mathcal{O}(\lvert V_i\rvert \cdot \min(\alpha, \beta^2))$	-	-
Oblivious Token Gen.	$\mathcal{O}(N)$	$\mathcal{O}(\lvert\mathcal{B}\rvert \cdot N)$	$\mathcal{O}(N)$
Homomorphic Enc.	$\mathcal{O}(N)$	-	$\mathcal{O}(N)$
Homomorphic Agg.	-	$\mathcal{O}(\lvert\mathcal{B}\rvert \cdot N)$	-
Threshold Decrypt.	$\mathcal{O}(Q)$	$\mathcal{O}(m \cdot Q)$	$\mathcal{O}(Q)$

Computational Complexity The localised enumeration in Algorithm 1 evaluates candidate intermediary nodes. In the worst case, iterating through the neighbourhoods of a vertex v takes $\mathcal{O}(\lvert N^-(v)\rvert \cdot \lvert N^+(v)\rvert)$ time. However, due to the pruning thresholds (α and β), the operations per node are capped at $\mathcal{O}(\min(\alpha, \beta^2))$, causing the overall complexity to scale linearly with the local vertex set $\mathcal{O}(\lvert V_i\rvert \cdot \min(\alpha, \beta^2))$. For token generation, the bank performs one hash-to-curve evaluation and one scalar multiplication per slot requiring $\mathcal{O}(N)$ per bank, while the CC evaluates its master scalar against every point across all banks $\mathcal{O}(\lvert B\rvert \cdot N)$. The Paillier cryptosystem dominates the CPU cycles. A participating bank performs exactly N encryptions to build its padded vector $\mathcal{O}(N)$. The CC executes homomorphic additions across matching tokens via $\mathcal{O}(\lvert\mathcal{B}\rvert \cdot N)$ modular multiplications. During collaborative decryption of Q queried tokens, each bank computes a partial decryption share, yielding a local complexity of $\mathcal{O}(Q)$. Combining a threshold of m shares via Lagrange polynomial interpolation introduces a final recovery cost of $\mathcal{O}(m \cdot Q)$ for the CC.

Communication Complexity Banks upload blinded curve point and download evaluated curve points from the CC, requiring $\mathcal{O}(N)$ bandwidth per bank. For the encryption phase, each bank uploads its uniformly padded vector containing N 4096-bit ciphertexts alongside the 32-byte OPRF tokens. For collaborative decryption of the Q flagged tokens, the CC broadcasts the aggregated target ciphertexts to the network. Each bank returns a partial decryption share modulo n^2, matching the 4096-bit footprint of a standard ciphertext. Thus, the round-trip communication scales directly with the number of queried anomalies as $\mathcal{O}(Q)$.

5.2 Privacy and Security Guarantees

We evaluate the framework under the semi-honest (honest-but-curious) adversarial model described in Section 4.2. The formal security of the protocol relies on three standard cryptographic assumptions: the IND-CPA security of the Paillier cryptosystem [16], the Decisional Diffie-Hellman (DDH) assumption for OPRF blinding [20, 21], and the information-theoretic security of Shamir's Secret Sharing during threshold decryption [19].

5.2.1 Privacy Against the Central Coordinator

The Central Coordinator CC orchestrates the entire protocol but is isolated from plaintext data via four protective layers. First, via volume hiding, all institutions transmit vectors padded with dummy elements up to a uniform capacity N, preventing the CC from deducing individual transaction volumes or subgraph densities. Second, during OPRF evaluation, token blinding ensures the CC only operates on blinded curve points $Y_{st} = r_i \cdot P_{st}$; because the blinding scalar r_i is uniformly sampled and kept secret by the local bank, the point Y_{st} appears uniformly random under the DDH assumption [23]. Third, the banks shuffle the vectors before unblinding, preventing the CC from positionally linking blinded inputs to their unblinded outputs. Finally, payload confidentiality is maintained because the IND-CPA property of Paillier prevents the CC from distinguishing between a ciphertext containing a valid count $\mathsf{Enc}_{\mathsf{pk}}(I_i(s,t))$ and a dummy padding ciphertext $\mathsf{Enc}_{\mathsf{pk}}(0)$.

5.2.2 Privacy Against Participating Institutions

A curious participating institution B_i seeks to map the localised topology of competitors, but the protocol enforces strict cryptographic isolation. First, for token unlinkability, the tokens are evaluated as $\tau_{st} = k \cdot P_{st}$; because the CC's master scalar k is secret and the elliptic curve satisfies one-wayness [21], a bank cannot invert a token to unmask the plaintext pair (s,t) unless it already exists within its local subgraph. Second, decryption authority is distributed because institutions only observe the final homomorphically aggregated ciphertexts; decrypting these requires a minimum threshold m of independent institutions to cooperate, preventing any unilateral decryption of network states.

5.2.3 Information Leakage and Limitations

Finally, we outline the structural limitations and expected information leakages inherent to the protocol. First, regarding output leakage, when the global plaintext count $I(s,t)$ is revealed via threshold decryption, the querying bank B_i can trivially compute the exact number of transactions that occurred outside its network between s and t, an unavoidable property of multi-party computation [31]. Second, the semi-honest model assumes participants execute the protocol faithfully; because the protocol lacks Zero-Knowledge Proofs (ZKPs) to verify ciphertext contents, an active adversary could inject maliciously large integer values during the encryption to manipulate the global aggregate count. Finally, if the CC were to collude with a participating bank, the bank could share its known plaintext-to-token mappings, allowing the CC to deanonymise those specific OPRF tokens and expose the private transaction edges held by other non-colluding institutions.

6 Experimental Evaluation

This section evaluates the performance of our Rust implementation across two dimensions: classification accuracy and runtime. We implemented our protocol in Rust and benchmarked it using the synthetic AMLWorld dataset [7] under varying graph sizes and laundering intensities. To support reproducibility, our complete source code is publicly available.[2]

[2]The project source code is available at: `https://github.com/dhruvan2006/smurf-rs/`

6.1 Experimental Setup

All benchmarks were executed on a single `c7a.16xlarge` instance provisioned from Amazon Web Services (AWS) [35]. This instance is powered by a fourth-generation AMD EPYC processor with a maximum turbo frequency of 3.7 GHz. It provides 64 dedicated physical CPU cores and 128 GiB of DDR5 memory.

The complete protocol is implemented from scratch in Rust and compiled via `rustc 1.98.0 nightly` with full optimisations enabled under the release profile. The OPRF evaluation is built upon a prime-order elliptic curve group, while additive homomorphic score aggregation utilises a highly optimised distributed homomorphic encryption scheme. To maximise computational throughput, both cryptographic primitives are compiled to exploit the 512-bit wide Advanced Vector Extensions 512 (AVX-512) hardware feature available on our machine. Additionally, the implementation is highly parallelised to distribute the computational workload evenly across all 64 physical cores.

6.2 Dataset

To measure the effectiveness and scaling limits of our system, we evaluated it on two graph sizes (Small and Medium) and two laundering intensities (Low Intensity [LI] and High Intensity [HI]) of the AMLWorld dataset [7]. Although the dataset also features a Large variant, evaluating this was omitted from the current scope due to the extensive computational resources and processing time required. The bank, accounts, transaction, unique edges, and scatter-gather counts for each configuration are detailed in Appendix B while our methodology to bank sharding is discussed in Appendix C.

6.3 Classification Performance

Table 2 presents the classification performance at the optimal threshold (θ) that maximises the F_1-score. Because our homomorphic aggregation evaluates precise integer sums without introducing noise, these metrics match a centralised plaintext baseline.

Table 2: Classification Performance Across AMLWorld at the Threshold With Optimal F_1

Dataset	Intensity	Optimal θ	Accuracy	Precision	Recall	F_1-Score	PR-AUC
Small	Low (LI)	8	99.99%	100.00%	69.09%	81.72%	0.4423
	High (HI)	6	99.97%	88.75%	76.96%	82.44%	0.6874
Medium	Low (LI)	9	99.98%	86.47%	53.36%	65.99%	0.5172
	High (HI)	8	99.94%	88.14%	59.71%	71.19%	0.6671

The system demonstrates high detection accuracy on smaller graphs but exhibits a clear performance degradation as the network scales. On the Small dataset, the model yields peak F_1-scores of 81.72% (LI) and 82.44% (HI). On the Medium dataset, the optimal threshold (θ) increases (from 6 to 8 for HI), resulting in a slight decline in recall, with the final F_1-score settling at 65.99% (LI) and 71.19% (HI). This pattern indicates that as transaction graphs scale, legitimate entities generate a higher density of conflicting benign background noise.

Additionally, performance metrics are consistently higher under High Intensity (HI) configurations compared to their Low Intensity (LI) counterparts across both evaluated dataset sizes. Denser illicit transaction occurrence generate a stronger network signature, allowing the protocol to isolate anomalous structures more effectively from benign background noise.

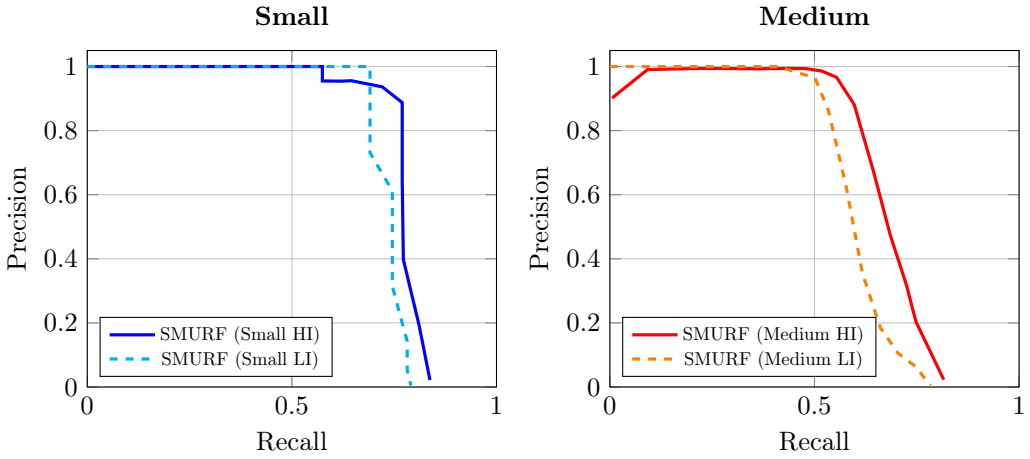

Figure 3: Precision-Recall Curves on Small and Medium AMLWorld datasets for $1 \leq \theta \leq 20$.

6.4 Runtime Performance

To evaluate computational efficiency, we measure execution times across the primary protocol phases (Table 3). Local Triplet Enumeration and Oblivious Token Generation (OPRF) demonstrate high efficiency, executing in under 7 and 13 seconds, respectively, even on the Medium dataset. The primary computational bottleneck is the homomorphic encryption and decryption phase. Paillier processing dominates the end-to-end execution time, scaling from approximately 1,154 seconds (19 minutes) on Small-HI to 5,779 seconds (1.6 hours) on Medium-HI, despite hardware acceleration via AVX-512 and multi-core parallelisation.

Table 3: Execution Time per Protocol Phase Across AMLWorld

Dataset	Intensity	Triplet (ms)	OPRF (ms)	Paillier (s)	Total (s)
Small	Low (LI)	1766	3225	1510.1	1521.7
	High (HI)	1377	2508	1154.7	1173.9
Medium	Low (LI)	6293	11,350	5264.7	5340.1
	High (HI)	6944	12,374	5779.7	5828.1

7 Discussion

7.1 Interpretation of Classification Performance

Empirical evaluations demonstrate that SMURF successfully uncovers distributed cross-bank scatter-gather motifs without suffering a "privacy penalty". Because our homomorphic pipeline operates entirely over precise integer arithmetic rather than probabilistic approximations or noisy perturbations, the classification metrics match a centralised plaintext baseline exactly. The system demonstrates exceptional performance on the High Intensity (HI) configurations, yielding a peak F_1-score of 82.44% on Small-HI and retaining 71.19% on Medium-HI. This indicates that dense laundering activities create a highly visible structural footprint that our protocol can reliably isolate from benign financial activities. The observed precision remains high across both graph sizes, achieving 88.75% on Small-HI and 88.14% on Medium-HI. Scaling from the Small to the Medium dataset introduces a higher volume of legitimate transactions that structurally mimic a scatter-gather pattern, therefore requiring a higher optimal filtering threshold (θ) that slightly suppresses recall.

7.2 Practical Deployment and Feasibility

Evaluating the runtime numbers shows some clear bottlenecks for real-world deployment. The Local Triplet Enumeration and Oblivious Token Generation (OPRF) phases are highly efficient, completing in 10 seconds even for the Medium dataset containing more than 31 million transactions. This confirms that our distributed alignment pipeline successfully bypasses the severe multi-institution communication bottlenecks characteristic of traditional pairwise Private Set Intersection (PSI) methods.

The primary computational bottleneck lies in in the Additive Homomorphic phase. Because semantic security requires populating vectors up to a fixed uniform global capacity N to conceal local transaction volumes, encrypting and decrypting the Paillier ciphertexts dominates the end-to-end execution time. This overhead scales linearly with N, resulting in processing times that range from 19 minutes on Small-HI to around 1.6 hours on Medium-HI.

Although these execution requirements render the system unsuitable for real-time transaction blocking, they establish its feasibility as a high-throughput batch-processing architecture. A runtime of around 1.6 hours to securely analyse tens of millions of distributed transactions allows institutions to effortlessly run SMURF as an overnight or weekly collaborative AML sweep.

7.3 Privacy vs. Information Leakage Trade-off

The framework successfully satisfies its primary design objective: preventing the disclosure of raw account identities, transaction weights, and institutional subgraphs. By enforcing a uniform padding capacity N, the framework effectively neutralises volume analysis attacks, ensuring the Central Coordinator (CC) cannot deduce the transaction density of any participating bank. However, we do acknowledge some acceptable leakages. When the global count plaintext $I(s, t)$ is revealed during the Threshold Decryption phase, the querying institution inevitably learns the aggregate number of external transactions from s to t. However, this leakage represents the minimum theoretical information required to achieve the multi-party computational objective itself [31]. Because the unblinded tokens are purely pseudonymous, participants cannot reverse-engineer the underlying account identifiers or deduce the wider topological structure of external networks beyond the specific queried pairs.

7.4 Limitations

Despite its strong classification and cryptographic guarantees, the framework possesses certain limitations. First, the heavy reliance on the Paillier cryptosystem induces significant ciphertext expansion and modular exponentiation overhead. Second, the framework operates under a semi-honest adversarial model; because the protocol lacks Zero-Knowledge Proofs (ZKPs) for ciphertext verification, it assumes participants supply valid inputs. A malicious actor could inject arbitrarily inflated integer values to distort the global aggregates. Third, while the precision rate remains exceptionally high, the observed drop in recall on larger, low-intensity networks implies that a small portion of subtle laundering networks may bypass detection if they fall below the global threshold θ. Finally, the motif detection depth is limited to 2-hop scatter-gather patterns. Sophisticated financial crime rings deploying deeper, arbitrary k-hop laundering chains would escape the scope of the local triplet enumeration phase. This highlights a critical future work.

8 Responsible Research

The development of privacy-preserving frameworks for Anti-Money Laundering (AML) carries dual-use responsibilities. Using Oblivious Pseudorandom Functions (OPRF) and Homomorphic Encryption (HE), our framework protects consumer data privacy, as stated in global mandates such as the GDPR [13] and Swiss FADP [14], to eliminate the vulnerability associated with centralised financial data storage. However, publishing structural motifs and optimal thresholds (θ) risks providing an evasion blueprint for adversarial actors. To ensure scientific reproducibility without violating banking secrecy laws, we use a public synthetically generated dataset AMLWorld [7]. However, a notable caveat is potential data bias; our algorithm may be overfit to the specific manner in which the dataset is synthesised, potentially overstating performance relative to real-world financial data. Reproducibility is guaranteed by stating the explicit hardware used, documenting the specific compiler version used, and fixing the exact versions of all dependencies in the codebase. The complete source code is also made public in an open-source repository.

9 Conclusions and Future Work

This work addressed the challenge of detecting distributed money laundering operations, specifically the scatter-gather pattern, without violating consumer data privacy regulations such as the GDPR [13] and Swiss FADP [14]. While existing literature is largely limited to pairwise protocols that fail to scale or capture multi-party laundering chains exactly, this thesis introduces SMURF: a novel privacy-preserving multi-party graph mining framework featuring a five-phase protocol.

By combining Oblivious Pseudorandom Functions (OPRF) and threshold Additive Homomorphic Encryption (AHE), SMURF enables an arbitrary number of financial institutions to collaboratively align and aggregate transaction chains entirely in the encrypted domain. It is designed in such a way that it preserves perfect utility, matching a centralised plaintext baseline exactly. Our empirical evaluations demonstrate that SMURF achieves an F_1-score of 82.44% on a dataset with approx. 5 million transactions and 369 scatter-gather accounts. Moreover, with an end-to-end execution time of around 1.6 hours for networks with around 31 million transactions, the system proves highly viable as an overnight or weekly audit within regional banking networks.

Despite these strong results, there are still several opportunities for future work. First, our localised triplet enumeration pipeline evaluates transaction networks from a purely topological perspective, flagging anomalies based on edge connections alone. Because the protocol ignores transactional properties such as transaction amounts, timestamps, and velocity, sophisticated actors could slip under structural thresholds. Future iterations could incorporate securely aggregated metadata, such as homomorphically computed net-flow variances, to increase detection accuracy. Second, the current framework is structurally bounded to 2-hop scatter-gather motifs. Real-world laundering syndicates frequently employ deeper, multi-hop layering chains spanning arbitrary lengths. Extending the protocol to support arbitrary k-hop paths remains an essential next step. Finally, the reliance on the Paillier cryptosystem introduces significant modular exponentiation and ciphertext expansion overhead. Subsequent research should explore alternative cryptographic primitives, such as transitioning to an additively homomorphic Elliptic Curve ElGamal (EC-ElGamal) scheme, to drastically reduce communication bandwidth and processing latencies.

References

[1] United Nations Office on Drugs and Crime. "Improving regional investigations on money laundering and asset recovery," UNODC Regional Office for Afghanistan, Central Asia, Iran and Pakistan, Accessed: Jun. 21, 2026. [Online]. Available: https://www.unodc.org/roca/en/NEWS/news_2024/november/improving-regional-investigations-on-money-laundering-and-asset-recovery.html.

[2] T. Pietschmann and J. Walker, "Estimating illicit financial flows resulting from drug trafficking and other transnational organized crimes," United Nations Office on Drugs and Crime (UNODC), Vienna, Research Report, Oct. 2011. Accessed: Jun. 21, 2026. [Online]. Available: https://www.unodc.org/documents/data-and-analysis/Studies/Illicit_financial_flows_2011_web.pdf.

[3] United Nations Office on Drugs and Crime. "Money laundering," UNODC Global Programme against Money Laundering, Accessed: Jun. 21, 2026. [Online]. Available: https://www.unodc.org/unodc/en/money-laundering/overview.html.

[4] J. Blanusa et al., "Graph feature preprocessor: Real-time subgraph-based feature extraction for financial crime detection," in *Proceedings of the 5th ACM International Conference on AI in Finance, ICAIF 2024, Brooklyn, NY, USA, November 14-17, 2024*, ACM, 2024, pp. 222–230. DOI: 10.1145/3677052.3698674.

[5] J. Nicholls, A. Kuppa, and N. Le-Khac, "Financial cybercrime: A comprehensive survey of deep learning approaches to tackle the evolving financial crime landscape," *IEEE Access*, vol. 9, pp. 163 965–163 986, 2021. DOI: 10.1109/ACCESS.2021.3134076.

[6] M. Weber et al., "Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics," *CoRR*, vol. abs/1908.02591, 2019. arXiv: 1908.02591.

[7] E. R. Altman, J. Blanusa, L. von Niederhäusern, B. Egressy, A. Anghel, and K. Atasu, "Realistic synthetic financial transactions for anti-money laundering models," in *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., 2023. [Online]. Available: http://papers.nips.cc/paper_files/paper/2023/hash/5f38404edff6f3f642d6fa5892479c42-Abstract-Datasets_and_Benchmarks.html.

[8] K. Michalak and J. Korczak, "Graph mining approach to suspicious transaction detection," in *Federated Conference on Computer Science and Information Systems, FedCSIS 2011, Szczecin, Poland, 18-21 September 2011, Proceedings*, M. Ganzha, L. A. Maciaszek, and M. Paprzycki, Eds., 2011, pp. 69–75. [Online]. Available: https://ieeexplore.ieee.org/document/6078254/.

[9] M. Starnini et al., "Smurf-based anti-money laundering in time-evolving transaction networks," in *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track - European Conference, ECML PKDD 2021, Bilbao, Spain, September 13-17, 2021, Proceedings, Part IV*, Y. Dong, N. Kourtellis, B. Hammer, and J. A. Lozano, Eds., ser. Lecture Notes in Computer Science, vol. 12978, Springer, 2021, pp. 171–186. DOI: 10.1007/978-3-030-86514-6_11.

[10] M. Lee et al., "Autoaudit: Mining accounting and time-evolving graphs," in *2020 IEEE International Conference on Big Data (IEEE BigData 2020), Atlanta, GA, USA, December 10-13, 2020*, X. Wu et al., Eds., IEEE, 2020, pp. 950–956. DOI: 10.1109/BIGDATA50022.2020.9378346.

[11] X. Li et al., "Flowscope: Spotting money laundering based on graphs," in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, AAAI Press, 2020, pp. 4731–4738. DOI: 10.1609/AAAI.V34I04.5906.

[12] Z. Tian, Y. Ding, X. Yu, E. Gong, J. Liu, and K. Ren, "Towards collaborative anti-money laundering among financial institutions," in *Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025- 2 May 2025*, G. Long, M. Blumestein, Y. Chang, L. Lewin-Eytan, Z. H. Huang, and E. Yom-Tov, Eds., ACM, 2025, pp. 4722–4733. DOI: 10.1145/3696410.3714576.

[13] European Parliament and Council of the European Union. "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data," Accessed: Jun. 18, 2026. [Online]. Available: https://data.europa.eu/eli/reg/2016/679/oj.

[14] Federal Assembly of the Swiss Confederation, *Federal act on data protection (fadp) of 25 september 2020*, SR 235.1, 2020. [Online]. Available: https://www.fedlex.admin.ch/eli/cc/2022/491/en.

[15] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," *ACM Computing Surveys*, vol. 51, no. 4, 79:1–79:35, 2018. DOI: 10.1145/3214303.

[16] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, J. Stern, Ed., ser. Lecture Notes in Computer Science, vol. 1592, Springer, 1999, pp. 223–238. DOI: 10.1007/3-540-48910-X_16.

[17] I. Damgård and M. Jurik, "A generalisation, a simplification and some applications of paillier's probabilistic public-key system," in *Public Key Cryptography, 4th International Workshop on Practice and Theory in Public Key Cryptography, PKC 2001, Cheju Island, Korea, February 13-15, 2001, Proceedings*, K. Kim, Ed., ser. Lecture Notes in Computer Science, vol. 1992, Springer, 2001, pp. 119–136. DOI: 10.1007/3-540-44586-2_9.

[18] T. Nishide and K. Sakurai, "Distributed paillier cryptosystem without trusted dealer," in *Information Security Applications - 11th International Workshop, WISA 2010, Jeju Island, Korea, August 24-26, 2010, Revised Selected Papers*, Y. Chung and M. Yung, Eds., ser. Lecture Notes in Computer Science, vol. 6513, Springer, 2010, pp. 44–60. DOI: 10.1007/978-3-642-17955-6_4.

[19] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979. DOI: 10.1145/359168.359176.

[20] A. Davidson, A. Faz-Hernández, N. Sullivan, and C. A. Wood, "Oblivious pseudorandom functions (oprfs) using prime-order groups," *RFC*, vol. 9497, pp. 1–61, 2023. DOI: 10.17487/RFC9497.

[21] M. J. Freedman, Y. Ishai, B. Pinkas, and O. Reingold, "Keyword search and oblivious pseudorandom functions," in *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, J. Kilian, Ed., ser. Lecture Notes in Computer Science, vol. 3378, Springer, 2005, pp. 303–324. DOI: 10.1007/978-3-540-30576-7_17.

[22] A. Davidson, I. Goldberg, N. Sullivan, G. Tankersley, and F. Valsorda, "Privacy pass: Bypassing internet challenges anonymously," *Proceedings Privacy Enhancing Technologies*, vol. 2018, no. 3, pp. 164–180, 2018. DOI: 10.1515/popets-2018-0026.

[23] S. Jarecki, A. Kiayias, H. Krawczyk, and J. Xu, "Highly-efficient and composable password-protected secret sharing (or: How to protect your bitcoin wallet online)," in *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*, IEEE, 2016, pp. 276–291. DOI: 10.1109/EUROSP.2016.30.

[24] S. Jarecki, A. Kiayias, and H. Krawczyk, "Round-optimal password-protected secret sharing and T-PAKE in the password-only model," in *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, P. Sarkar and T. Iwata, Eds., ser. Lecture Notes in Computer Science, vol. 8874, Springer, 2014, pp. 233–253. DOI: 10.1007/978-3-662-45608-8_13.

[25] National Institute of Standards and Technology, "Secure hash standard (SHS)," U.S. Department of Commerce, Tech. Rep. FIPS PUB 180-4, Aug. 2015. DOI: 10.6028/NIST.FIPS.180-4.

[26] B. Egressy, L. von Niederhäusern, J. Blanusa, E. R. Altman, R. Wattenhofer, and K. Atasu, "Provably powerful graph neural networks for directed multigraphs," in *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, M. J. Wooldridge, J. G. Dy, and S. Natarajan, Eds., AAAI Press, 2024, pp. 11 838–11 846. DOI: 10.1609/AAAI.V38I10.29069.

[27] D. Commey, M. Nkoom, Y. Alsenani, S. G. Hounsinou, and G. V. Crosby, "Fedgraph-vasp: Privacy-preserving federated graph learning with post-quantum security for cross-institutional anti-money laundering," *CoRR*, vol. abs/2601.17935, 2026. arXiv: 2601.17935.

[28] A. Bay, Z. Erkin, M. Alishahi, and J. Vos, "Multi-party private set intersection protocols for practical applications," in *Proceedings of the 18th International Conference on Security and Cryptography, SECRYPT 2021, July 6-8, 2021*, S. D. C. di Vimercati and P. Samarati, Eds., SCITEPRESS, 2021, pp. 515–522. DOI: 10.5220/0010547605150522.

[29] K. Zhang et al., "Experimental secure multiparty computation from quantum oblivious transfer with bit commitment," *CoRR*, vol. abs/2411.04558, 2024. arXiv: 2411.04558.

[30] Financial Action Task Force, "International standards on combating money laundering and the financing of terrorism & proliferation: The FATF recommendations," FATF, Paris, France, Standard, Oct. 2025. Accessed: Jun. 21, 2026. [Online]. Available: https://www.fatf-gafi.org/en/publications/Fatfrecommendations/Fatf-recommendations.html.

[31] O. Goldreich, *The Foundations of Cryptography - Volume 2: Basic Applications*. Cambridge University Press, 2004. DOI: 10.1017/CBO9780511721656.

[32] K. Soramäki, M. L. Bech, J. Arnold, R. J. Glass, and W. E. Beyeler, "The topology of interbank payment flows," Federal Reserve Bank of New York, Staff Report 243, Mar. 2006. Accessed: Jun. 21, 2026. [Online]. Available: https://www.newyorkfed.org/medialibrary/media/research/staff_reports/sr243.pdf.

[33] M. Boss, H. Elsinger, M. Summer, and S. Thurner, "The network topology of the interbank market," *CoRR*, vol. abs/cond-mat/0309582, 2003. arXiv: cond-mat/0309582.

[34] A. Saxena, Y. Pei, J. Veldsink, W. van Ipenburg, G. Fletcher, and M. Pechenizkiy, "The banking transactions dataset and its comparative analysis with scale-free networks," in *ASONAM '21: International Conference on Advances in Social Networks Analysis and Mining, Virtual Event, The Netherlands, November 8 - 11, 2021*, M. Coscia, A. Cuzzocrea, K. Shu, R. Klamma, S. O'Halloran, and J. G. Rokne, Eds., ACM, 2021, pp. 283–296. DOI: 10.1145/3487351.3488339.

[35] Amazon Web Services. "Amazon EC2 C7a instances," Amazon.com, Inc., Accessed: Jun. 21, 2026. [Online]. Available: https://aws.amazon.com/ec2/instance-types/c7a/.

[36] P. Lagerwaard and M. de Goede, "In trust we share: The politics of financial intelligence sharing," *Economy and Society*, vol. 52, no. 2, pp. 202–226, 2023. DOI: 10.1080/03085147.2023.2175451.

[37] Autoriteit Consument & Markt, "Concurrentie op de Nederlandse spaarmarkt," ACM, Report ACM/UIT/622124, Jul. 16, 2024. Accessed: Jun. 21, 2026. [Online]. Available: https://www.acm.nl/system/files/documents/rapport-concurrentie-op-de-nederlandse-spaarmarkt.pdf.

A Symbols

Table 4: SMURF Framework Symbols and Descriptions

Symbol / Variable	Description
$G = (\mathcal{V}, \mathcal{E})$	Global distributed transaction network graph
$\mathcal{B} = \{B_1, B_2, \ldots, B_n\}$	Finite set of n distinct financial institutions
$\mathcal{V}_i$	Private local vertex set owned by bank B_i
$N^-(v)$	In-neighborhood of vertex v
$N^+(v)$	Out-neighborhood of vertex v
$I(s,t)$	Global count for pair (s,t)
$I_i(s,t)$	Local count for pair (s,t) at bank B_i
$\mathcal{V}_{smurf}$	Set of local candidate intermediary vertices
α	Pruning bound for path combinations
β	Pruning bound for maximum in- or out-degree
θ	Global filtering threshold parameter
$\mathbb{G}$	Elliptic curve group of prime order p
H_1, H_2	Hash-to-curve and base hash functions
k	Central Coordinator's secret master key
r_i	Bank B_i's private random blinding scalar
τ_{st}	Globally aligned pseudorandom identifier
N	Uniform maximum global padding capacity
$\mathsf{Enc}_{\mathsf{pk}}(\cdot)$	Paillier encryption function under public key pk
sk_i	Distributed private key share assigned to bank B_i
m	Minimum threshold of banks for decryption
h_i	Partial decryption share generated by bank B_i
Q	Total number tokens queried for decryption
$\mathsf{DecShare}_{\mathsf{sk}_i}(\cdot)$	Generates partial decryption share using key sk_i
$\mathsf{CombineThreshold}(\cdot)$	Interpolates m shares to reconstruct final plaintext

B Dataset Dimensions

Table 5 outlines the exact structural dimensions of the synthetic datasets generated via AMLWorld [7]. The metrics distinguish between transactions and edges. Specifically, **Transactions** represent individual historical fund transfers between accounts. Since an account can transfer money to the same recipient multiple times, the underlying network is a directed multigraph [26]. **Unique Edges** represent the collapsed, simple directed graph of the network. By treating repetitive transfers between a sender and a receiver as a single directed link and omitting self-transactions, this metric describes the real structural topology between accounts.

Table 5: Structural Dimensions of the Evaluated AMLWorld Datasets

Scale	**Intensity**	**# Banks**	**# Accounts**	**# Transactions**	**# Unique edges**	**# Scatter-Gather Accounts**
Small	Low (LI)	32,173	576,777	6,924,049	880,800	110
	High (HI)	23,378	422,726	5,078,345	647,939	369
Medium	Low (LI)	91,919	1,719,984	31,251,483	2,918,161	551
	High (HI)	94,462	1,758,573	31,898,238	2,998,917	2291

C Bank Sharding

The raw transaction ledgers in AMLWorld [7] present a highly fragmented financial ecosystem containing tens of thousands of sparse banking entities as listed in Table 5 and visualised in Figure 4. Evaluating these many independent banks fails to reflect a realistic deployment scenario for a domestic anti-money laundering framework. In practice, regulatory jurisdictions and financial intelligence units operate at a domestic scale [36]. For example, a mid-sized country like the Netherlands contains fewer than 100 operating credit institutions, where the top four banks (ING, Rabobank, ABN AMRO, and Volksbank) have a total share of around 90%-95% of the Dutch savings market in the years 2014 to 2023 [37].

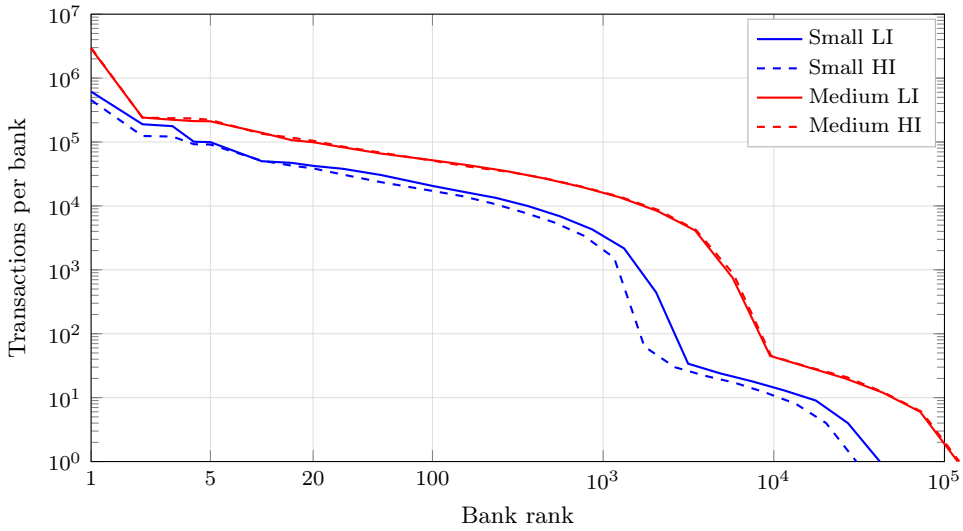

Figure 4: Log-log distribution of original transaction count across banks for Small and Medium datasets (LI and HI).

Therefore, we introduce a consolidation method to reduce the total number of banking entities while preserving original topological patterns, as illustrated in Figure 5. First, we isolate the top $N = 20$ primary banking institutions by total transactional count (the blue shaded region). For all remaining institutions, we apply a deterministic mapping function to

partition them into a fixed set of virtual proxy banks. For any given bank identifier $\mathrm{ID}_{\mathrm{bank}}$ outside the designated Top-N set, a sharded proxy ID ($\mathrm{ID}_{\mathrm{sharded}}$) is assigned via:

$$\mathrm{ID}_{\mathrm{sharded}} = -1000 - |\mathrm{ID}_{\mathrm{bank}} \pmod{S}|$$

where $S = 80$ represents the designated sharding size. This mathematical projection folds the massive tail distribution of sparse peripheral banks into a strictly bounded pool of 80 virtual banks.

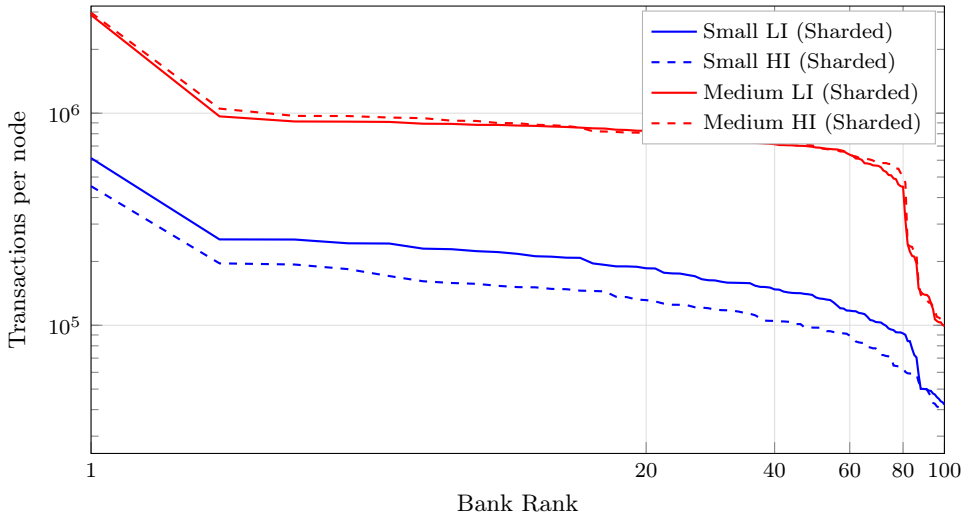

Figure 5: Log-log distribution of sharded transaction counts across banks for Small and Medium datasets (LI and HI).

D Hyperparameter Sensitivity Analysis

This section evaluates the impact of the pruning bounds α and β, on the downstream classification performance of the system. As formalised in (6), these hyperparameters are introduced to prevent combinatorial path explosions on high-degree hub nodes during local triplet enumeration.

Because the underlying financial transaction networks exhibit highly skewed, heavy-tailed degree distributions [32, 33, 34], a tiny fraction of accounts account for a massive proportion of potential edge pairings. Setting restrictive upper bounds for α (the max edge combination product) and β (the max degree) drops these hub nodes from processing, which dramatically reduces computational overhead.

To ensure that pruning these high-degree nodes does not lose considerable information, we execute a grid sensitivity sweep across both variables simultaneously. Figure 6 reports the resulting macro F_1 scores obtained across different threshold combinations.

The empirical results show that varying α and β across multiple orders of magnitude yields a negligible impact on overall detection accuracy. For instance, transitioning from highly aggressive filtering ($\alpha = 10^3, \beta = 100$) to an unconstrained graph processing run ($\alpha = \infty, \beta = \infty$) does not affect the F_1-score at all in the Small (HI) dataset.

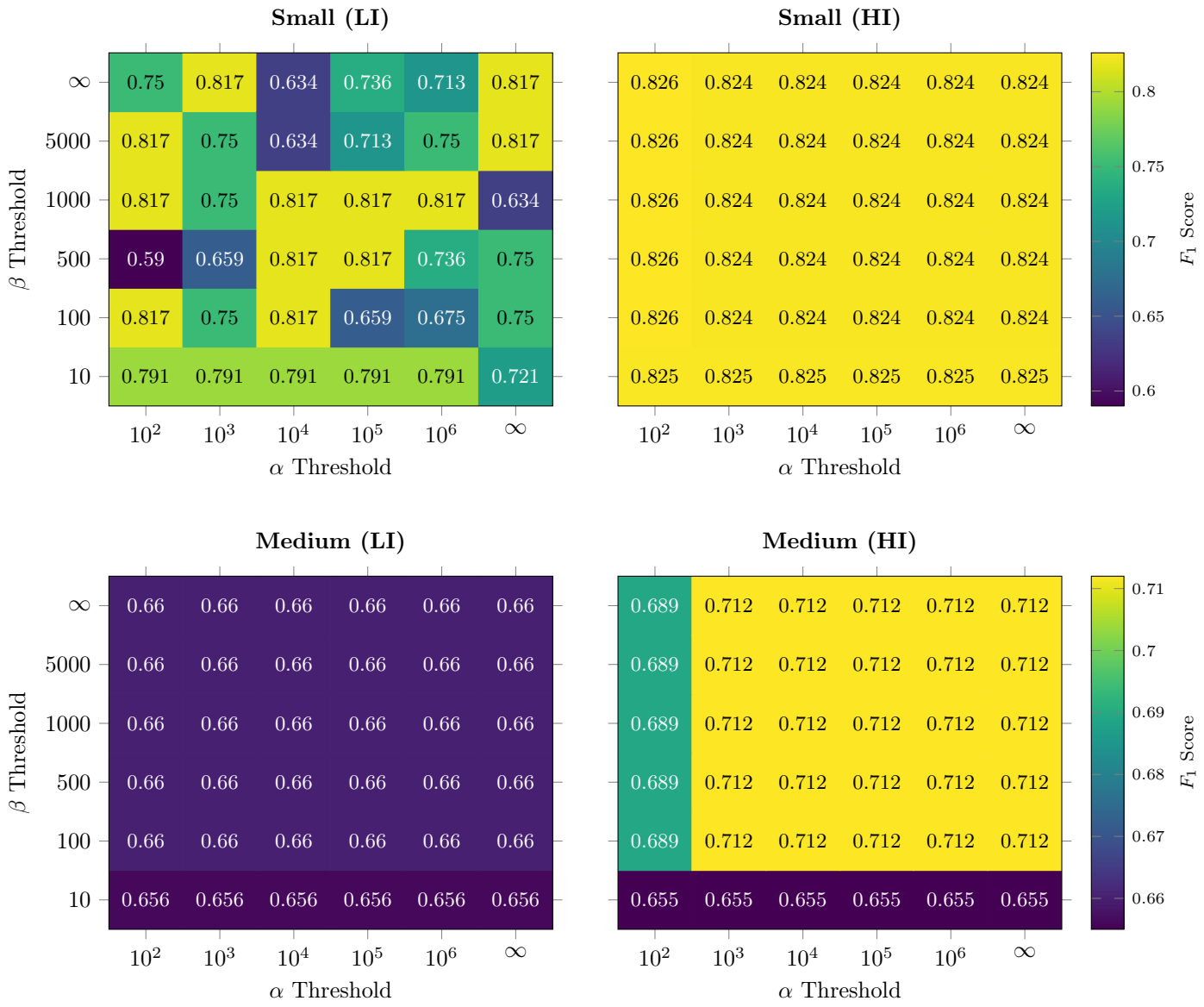

Figure 6: Sensitivity Analysis of F_1 Scores Under Varying α and β Pruning Thresholds.