Metadata privacy and obliviousness in distributed learning via a bulletin board: a proof-of-concept

Andreis G.M. Purim, Witor M.A. Oliveira, Allan M. de Souza

University of Campinas (UNICAMP)

{andreis.purim, witor.oliveira}@students.ic.unicamp.br
allanms@ic.unicamp.br

Abstract. *While many works study payload privacy in distributed learning, metadata and communication patterns between peers can still reveal sensitive information. We treat metadata privacy as a distinct problem, and first introduces a risk and threat model for metadata leakage in distributed learning systems. Building on this analysis, we present a bulletin board-based communication architecture in which peers exchange activations, gradients, and model updates under configurable privacy and efficiency settings. We implement a proof-of-concept code of the proposed design and evaluate it on small-scale distributed learning workloads, which is available on Github*[1].

1. Introduction

Distributed learning is a set of techniques that enable multiple clients to jointly train a machine learning model without centralizing their datasets. These methods have gained substantial popularity in domains such as mobile computing, IoT, healthcare, and edge AI because they reduce the need for large-scale data collection, support heterogeneous devices, and allow local data to remain under the control of its owner. Achieving this requires the repeated exchange of model-related information across clients and servers (e.g., gradients, activations, parameters), which introduces privacy risks.

Most prior work addresses these risks by protecting the *payload* of exchanged messages using cryptographic or statistical techniques. However, protecting the payload alone is insufficient. Even when messages are fully encrypted, the *metadata* associated with communication (who communicates with whom, when messages occur, and how often) can leak sensitive information about participants, their behavior, and the structure of the learning system itself, as illustrated in Figure 1.

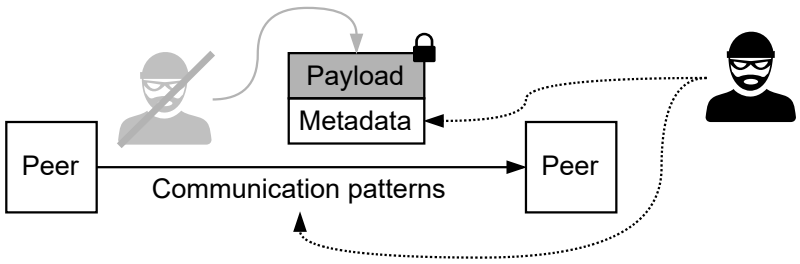

Figure 1. Illustration of the problem with metadata privacy in distributed learning

Consider, as an example, a hospital participating in a federated learning system for predicting complications in high-risk patients. Locally, the hospital trains on newly

[1] https://github.com/AndreisPurim/Bedlam

admitted cases and sends an encrypted update to the server only when a new patient evaluation is completed. Even though the model update is fully encrypted, the timestamp of the update remains visible to the server or to a network observer. Suppose a patient arrives at the emergency department at a given time. Shortly afterward, the hospital's learning client transmits an update. Because hospital admissions are partially observable (e.g., through waiting-room activity, visitor patterns, or public ambulance arrival records), an adversary can correlate patient arrival times with update transmissions. Over repeated observations, the adversary may infer that certain types of patients correspond to updates with characteristic sizes or frequencies. This form of metadata-based inference compromises patient confidentiality despite a perfect payload encryption.

This type of leakage is particularly pronounced in distributed learning settings compared to generic communication systems. Unlike traditional messaging scenarios, distributed learning involves iterative updates, structured training procedures, and communication patterns that are tightly coupled with real-world events and data generation processes. As a result, message timing, frequency, and size are not arbitrary but reflect underlying data characteristics and training dynamics. This tight coupling amplifies the risk of metadata leakage, as adversaries can exploit correlations between observed communication patterns and external events to infer sensitive information about participants or their data. Recent work has begun to show that communication structure, topology, and observable traffic patterns in federated learning may leak information even when message contents are protected [Rangwala et al. 2025, Shuvo and Hossain 2025, Shuvo et al. 2025].

Thus, we define the term *metadata-aware distributed learning* to refer to distributed learning systems in which the metadata is treated as an explicit design concern, and where mechanisms are introduced to limit, or control metadata leakage arising from message exchange. This includes approaches that modify how messages are routed, stored, retrieved, or scheduled in order to reduce linkability, activity inference, or other metadata-based attacks, independent of how payloads are protected.

To address these challenges, we propose a distributed learning architecture based on a neutral[2] bulletin board that mediates all communication between peers, removing direct linkability while remaining compatible with existing learning paradigms. We further introduce the notion of *metadata-aware distributed learning*, framing metadata privacy as a first-class design concern, and formalize it through a dedicated risk model and threat framework. We validate the feasibility of this approach via a proof-of-concept implementation using split learning, showing that metadata-aware communication can be achieved without disrupting training, while enabling configurable trade-offs between efficiency and degrees of obliviousness. The proposed architecture complements existing payload-protection mechanisms and provides a modular communication layer adaptable to different deployment requirements.

2. Background and Related Work

To the best of our knowledge, this work is among the first to systematically frame metadata privacy as an explicit design concern in distributed learning. This section briefly introduces the necessary background and situates our contribution within the closest re-

[2]By neutral, we mean that the board does not enforce protocol logic, assign roles, or interpret message semantics.

lated work, focusing on the two dominant paradigms in distributed learning: data-parallel and model-parallel approaches.

Federated Learning (FL) follows a data-parallel model, where clients train local models on private data and periodically submit gradients or parameters for aggregation [McMahan et al. 2016]. In contrast, Split Learning adopts a model-parallel approach, partitioning the neural network between clients and servers. Clients compute and transmit intermediate activations, while servers complete the forward and backward passes [Vepakomma et al. 2018]. Both paradigms reduce the need for data centralization, but expose information at two distinct layers: the payload itself and the metadata associated with its transmission.

A substantial body of work has demonstrated that payloads exchanged during distributed learning can leak sensitive information. In federated learning, gradients can be inverted to reconstruct training examples, sometimes from a single update, particularly in small-batch settings [Zhu et al. 2019, Wei et al. 2020, de Retana et al. 2025]. In split learning, the intermediate activations transmitted between model partitions can similarly reveal input features or labels. Recent attacks show that an honest-but-curious server can reconstruct inputs or infer labels with high accuracy [Nguyen et al. 2023, Zhu et al. 2025]. Beyond direct reconstruction, adversaries may also infer which client produced a given update or activation, enabling user or identity inference attacks when updates are tied to explicit participant identifiers [Li et al. 2024].

In response to these payload-level attacks, numerous defenses have been proposed to protect the contents of exchanged messages. Secure aggregation protocols mask individual gradients so that only their aggregate is revealed to the server [Bonawitz et al. 2016]. Private Information Retrieval (PIR) techniques have been used to hide which parameters or updates are accessed or contributed by a client, and in some cases even obscure the server's objective function [Mozaffari et al. 2022, Egger et al. 2025]. Other approaches rely on cryptographic primitives such as homomorphic encryption, oblivious transfer, or secure multi-party computation to enable computation over encrypted data or private selection of values, including adaptations to split learning settings [Nguyen et al. 2023]. While these mechanisms significantly reduce payload leakage, they do not address the metadata exposed by communication patterns, identities, or timing.

Metadata privacy has received comparatively little attention in distributed learning. Most systems still expose communication patterns that allow participant activity, contribution, or topology to be inferred even when payloads are encrypted. Only a small number of works explicitly explore anonymity, unlinkability, or board-based coordination. AnoFel introduces a bulletin-board-based design for federated learning that enables anonymous registration and submission of model updates using commitments and zero-knowledge proofs, demonstrating that client anonymity and unlinkability are achievable in their setting [Almashaqbeh and Ghodsi 2023]. In contrast, Aero adopts a public board primarily for transparency and auditability. The server still knows which devices contribute updates, and anonymity is not a design goal [Liu and Gupta 2024].

Overall, prior work largely focuses on securing payloads rather than communication metadata. Existing approaches either hide gradient or activation values while still

revealing participant identities and activity patterns, or provide anonymity only within a constrained scope.

Unlike AnoFel, which achieves anonymity in federated learning through commitments and zero-knowledge proofs tied to a specific aggregation protocol, BEDLAM does not assume a fixed learning paradigm and instead abstracts anonymity at the level of communication itself. Unlike Aero, where the bulletin board primarily serves transparency and auditability and the server retains knowledge of participant identities, BEDLAM treats anonymity and unlinkability as first-class design goals. BEDLAM differs in that it explicitly models metadata privacy as a system-level property, independent of whether the underlying learning protocol is data-parallel or model-parallel.

More broadly, BEDLAM relates to concepts from anonymous communication systems, such as anonymous messaging, traffic analysis resistance, and private information retrieval. While these techniques are traditionally studied in the context of secure communication networks, BEDLAM applies similar principles to distributed learning protocols, where the goal is not only to protect message contents but also to conceal who communicates, when, and how often within the training process.

3. Terminology, Conceptual Scope and Threat Model

The terminology below is intentionally architecture-agnostic within the context of metadata-aware distributed learning. It applies to systems that explicitly reason about communication metadata, regardless of the specific mechanism used to mediate message exchange. We first introduce general concepts and a metadata threat model applicable to such systems, and then specialize the discussion to adversaries that arise when communication is mediated through a bulletin-board abstraction, as used in our proposal.

- **Peers:** A peer is any participant in a distributed learning process. In split learning, peers may hold different model segments (e.g., M1, M2, M3), in federated learning, peers may correspond to local models or aggregators. A peer may act as a sender or receiver of messages depending on the training step.
- **Sender and Receiver (Communication roles):** Sender and receiver refer purely to message direction. A sender is any peer that emits an encrypted activation, gradient, or update, while a receiver is any peer that retrieves and consumes it. These roles are transient and may change across training steps. The board is never considered a sender or receiver, but only an intermediary. We therefore prefer the terms "peer/board" and "sender/receiver" over "client/server."
- **Source and Compute (Training roles):** Orthogonal to message direction, we distinguish the peer that initiates a training interaction from the peer that performs the requested computation. The *source* peer posts an activation, gradient, or update to begin a training step, while the *compute* peer retrieves this message, executes the corresponding forward or backward computation, and returns the result. In federated learning, the source is typically a client producing a local update and the compute peer is an aggregator that processes or combines updates. In split learning, the source may be a peer holding an upstream model segment (e.g., M1) that emits activations, while the compute peer holds a downstream segment (e.g., M2 or M3) that continues the forward or backward pass. These roles describe interaction and do not imply hierarchy or identity.

- **Payload:** The payload is the data content of a message that directly contributes to learning, such as gradients, activations, model parameters, or update vectors. Payloads are typically protected using cryptographic or statistical mechanisms (e.g., encryption, secure aggregation, or differential privacy).
- **Metadata:** Metadata refers to non-payload characteristics of communication, including message timing, frequency, size, ordering, channel use, sender–receiver linkage, and participation patterns. Even when payloads are encrypted end-to-end, metadata may leak sensitive behavioral or structural information.
- **Anonymity and Obliviousness:** Anonymity denotes the inability of an adversary (including the board or other peers) to link a message to a specific peer, encompassing sender anonymity, receiver anonymity, and sender–receiver unlinkability. Obliviousness refers more broadly to limiting what participants or intermediaries learn about communication patterns.
- **Bulletin board (or *board*):** A bulletin board is a neutral intermediary communication abstraction that stores encrypted messages posted by peers and makes them retrievable by other peers without direct addressing. The board is modeled as an honest-but-curious service whose only required property is availability. It is not trusted with confidentiality and does not interpret message semantics. In our proposed architecture, the board serves as the communication layer for metadata-aware distributed learning. Other metadata-aware designs may employ different communication abstractions and do not require the use of a bulletin board.

3.1. Metadata Aware DL Threat Model

In metadata-aware distributed learning, peers exchange protected payloads through observable communication channels. Even when payload contents are encrypted or otherwise protected, metadata associated with message exchange may enable adversaries to infer sensitive information about peers, their behavior, or the structure of the learning process. We identify the following metadata-related risks that arise across distributed learning architectures, independent of the specific communication mechanism employed:

- **(R1) Message Linkability:** The ability to associate multiple messages with the same peer over time based on identifiers, timing regularities, or message size patterns, even when peer identities are not explicitly revealed.
- **(R2) Sender–Receiver Association:** The ability to infer which sender peer communicates with which receiver peer, due to explicit addressing, routing metadata, or tightly coupled request–response patterns.
- **(R3) Peer Activity Inference:** The ability to infer when a peer is active, idle, or participating in training from message frequency, synchronization behavior, or temporal patterns.
- **(R4) Contribution Inference:** The ability to infer whether, when, or how much a peer contributes to training based on the presence, size, or timing of messages associated with that peer.
- **(R5) Topology and Role Inference:** The ability to infer system structure, including persistent roles (e.g., source or compute), hierarchy, or coordination patterns, from asymmetric, repeated, or directional communication flows.
- **(R6) External Temporal Correlation:** The ability to correlate learning activity with external events or data changes by observing burstiness, adaptive training behavior, or time-aligned message exchanges.

We consider adversaries whose objective is to infer sensitive information from communication metadata rather than from payload contents. Payloads (e.g., activations, gradients, or model updates) are assumed to be protected by mechanisms such as encryption, secure aggregation, or differential privacy, and are therefore out of scope. Adversaries are assumed to be passive with respect to protocol execution. In this study, they observe metadata but do not modify, inject, delay, or drop messages. Availability, integrity, and denial-of-service attacks are not considered yet. Under these assumptions, we consider the following general classes of adversaries:

- **(A1) Communication Observer:** A passive adversary capable of observing communication metadata at one or more points in the system (e.g., at a relay, server, or aggregation point), including message timing, size, and addressing information.
- **(A2) Network Observer:** A passive adversary that can observe traffic entering or leaving peers over the network (e.g., an ISP-level or infrastructure-level observer), but cannot inspect or modify payload contents.
- **(A3) Curious Peer:** A legitimate peer that follows the training protocol but records metadata from its own interactions in an attempt to infer information about other peers, their activity, or their roles.
- **(A4) Colluding Adversaries:** Any combination of the above adversaries that share observations in order to increase inference capability.

Across all adversary models, we assume that cryptographic primitives are not broken and that peers are not fully compromised. Attacks targeting payload confidentiality, endpoint compromise, or denial of service are orthogonal and out of scope. Our focus is exclusively on inference risks arising from communication metadata. This threat model is therefore not intended to be a complete security model, but rather a focused analysis of metadata-related risks in distributed learning systems.

4. BEDLAM: A Bulletin Board approach

This section presents our proof-of-concept design for addressing the metadata privacy risks identified earlier. We introduce *Board-Enabled Distributed Learning via Anonymous Messaging* (BEDLAM), an architecture for metadata-aware distributed learning in which all communication between participants is mediated by a neutral and oblivious bulletin board rather than through direct, identifiable communication channels. Figure 2 illustrates the proposal using a three-party split learning example[3].

In our proposal, communication between peers is mediated by the board: a message is a unit posted to the board by a sender and later retrieved by a receiver peer. It is composed of three parts: a public header, a private header, and a private payload. The public header contains only non-sensitive fields (e.g., a nonce) that do not compromise privacy. The private header is an encrypted component indicating the type of payload, while the payload itself carries the actual learning data, such as activations, gradients, or model updates. Both the private header and private payload are treated as opaque ciphertext blobs, ensuring that the board cannot interpret their contents or link them to specific peers.

[3]For clarity of exposition, we describe BEDLAM using a split learning configuration, while emphasizing that the same communication layer and design principles apply to other distributed learning settings.

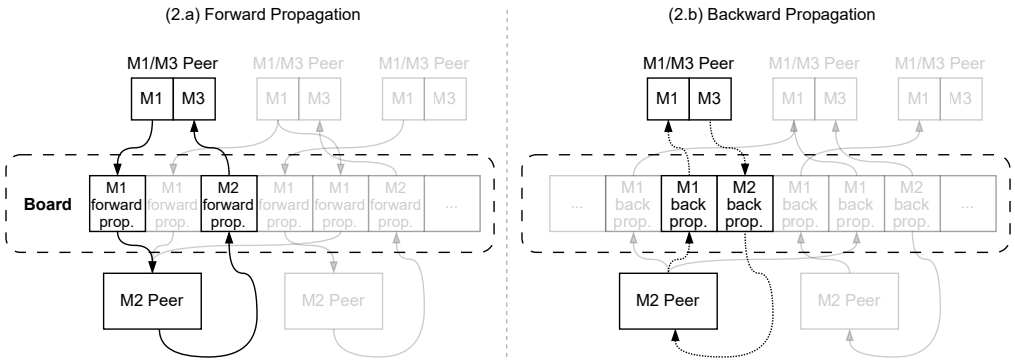

Figure 2. Illustration of the forward and backpropagation steps in BEDLAM.

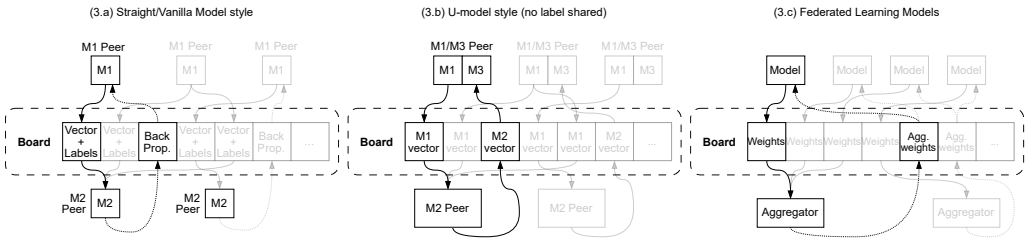

Figure 3. Example model configurations supported by the bulletin board.

In the depicted scenario, the model is partitioned into three segments (M1, M2, and M3). One peer holds M1 and M3, while another holds M2. A training iteration begins when the M1/M3 peer adds an encrypted M1 activation vector to the board. The M2 peer periodically scans the board, detects the relevant message, computes the forward pass for its segment, and adds the resulting M2 activation back onto the board for M3 to consume. Backpropagation proceeds in a similar manner: the M1/M3 peer posts the backward activation after computing gradients for M3, and M2 retrieves it to perform its own backward computation. After completing its update, M2 again posts its backward activation to the board for M1. Communication may continue by appending new messages to the board.

Because the board is role-agnostic, it does not assume or encode which peers perform which actions or assume which roles. As shown in Figure 3, the same architecture supports multiple configurations: a standard two-way split (M1/M2), the M1/M3-M2 split used in our example, and federated learning models exchanging entire parameter vectors. Importantly, a single board instance can support multiple model types simultaneously, and peers may change communication partners at any point during training.

The board-based communication model naturally supports asynchronous interaction, allowing participants to disconnect, delay responses, or operate on different time scales without disrupting the protocol. Because messages persist on the board until retrieved, temporary unavailability of a peer does not result in lost updates or coordination failures. This makes the architecture well suited for settings where responsiveness is secondary to privacy and robustness. For example, a network of hospitals may collaborate via a shared board to train or exchange multiple distributed learning models concurrently, with different subsets of hospitals participating in different tasks. In such a scenario, institutions may go offline or respond days later without compromising correctness or privacy, as the board preserves all necessary state until it is consumed.

The properties described in this section should be interpreted as design goals of the proposed architecture rather than unconditional guarantees. Their realization depends on underlying assumptions about the system model, the behavior of participating peers, and the correct use of supporting mechanisms (e.g., cryptographic primitives or retrieval protocols). Some properties may be achieved fully under ideal conditions, while others represent optional or configurable trade-offs that depend on deployment choices and efficiency constraints.

4.1. Properties of the "ideal" BEDLAM system

We now describe the properties of an idealized BEDLAM system. Practical deployments may implement a subset of these properties or trade some of them for efficiency, depending on threat model and system constraints. Table 1 maps the relationship between metadata risks, adversary capabilities, and the properties that mitigate each risk.

- **(P1) Payload Opacity:** The board cannot inspect, interpret, or modify the contents of messages. All payloads and private headers are treated as opaque ciphertext blobs protected by cryptographic mechanisms external to the board.
- **(P2) Role Agnosticism:** The communication layer does not encode or enforce persistent roles (e.g., client, server, source, compute). Any peer may assume different roles across training steps without revealing persistent role assignments through communication metadata
- **(P3) Protocol Independence:** The communication layer is agnostic to the learning protocol, model partitioning, and optimization method. The same message format and board abstraction can support split learning, federated learning, or hybrid configurations concurrently.
- **(P4) Session Unlinkability (Key Rotation):** Peers are not required to maintain long-lived identifiers or cryptographic keys across training interactions. By supporting key rotation and session renewal, BEDLAM prevents adversaries from linking multiple messages or training steps to the same long-term peer or training process. In particular, a source peer may intentionally terminate an ongoing session and initiate a new one (for example, to avoid revealing how many epochs it has already trained). By issuing a fresh request on the agreement sub-board, the peer effectively appears as a new participant, enabling session renewal without exposing training progress or persistent identity.
- **(P5) Board Obliviousness:** The board cannot link messages to specific peers, roles (source or compute), or training steps (beyond what is trivially observable from message existence). In particular, the board does not learn sender–receiver pairs or model structure.
- **(P6) Sender–Receiver Decoupling:** Communication between peers does not rely on direct addressing. Messages are posted to and retrieved from the board without explicit sender–receiver identifiers, preventing direct association between communicating peers. Depending on design choices, this property may yield the following sub-properties:
  - **(P6.a) Source Unobservability at Compute ("single-blind"):** A compute peer cannot identify which source peer initiated a training interaction. For example, all M2 peers may publish their public keys. An M1/M3 peer encrypts the payload using the chosen M2's public key and includes

a pseudonym and an ephemeral return key. M2 can decrypt the message and use the embedded pseudonym or return key to send a response back through the board, without learning the true identity of the originating M1/M3 peer.
    - **(P6.b) Mutual Peer Unobservability ("double-blind"):** Neither source nor compute peers can identify each other, and no direct sender–receiver linkage is revealed during training. The exact mechanism implemented to achieve this will is explained later.
- **(P7) Asynchronous Persistence:** Messages persist on the board, enabling asynchronous communication. Peers may disconnect, delay responses, or operate on different time scales without causing message loss or protocol failure.
- **(P8): Board Memory:** The board may be append-only (ideal case) or may implement memory optimization mechanisms at the cost of additional metadata leakage. That is, removing or modifying past entries may leak timing information, reveal delivery events, or introduce synchronization inconsistencies if expected messages disappear.
    - **(P8.a) Append-Only Boards:** Messages, once posted, are not removed or modified. Append-only semantics prevent delivery events, retrieval timing, or message disappearance from leaking metadata.
    - **(P8.b) Non-append-only Boards:** Boards that implement memory management mechanisms such as garbage collection, message expiration, or slot reuse. Such mechanisms must be carefully designed, as they may reintroduce timing or access-pattern leakage if not metadata-safe.
- **(P9): Optimized Retrieval:** The board may support efficient, privacy-preserving message discovery or retrieval mechanisms (e.g., digest-based scanning or PIR-style access), allowing peers to identify relevant messages without revealing access patterns to the board or external observers.

4.2. Append-only boards, efficiency, and practical limitations

Scalability is a central challenge of board-based communication, particularly in systems that rely on append-only data structures. As the volume of messages grows over time, both storage and retrieval costs can increase significantly, impacting the practicality of such designs.

The bulletin board abstraction was initially conceived as an append-only ledger, similar to blockchain-based systems, where immutability naturally enforces strong auditability and prevents retrospective manipulation. In practice, however, a bulletin board need not be implemented as a blockchain. It may instead be realized as an in-memory service, a database-backed log, or a distributed key-value store, depending on deployment constraints.

A direct consequence of the append-only design is that, in the naïve case, receiver peers must download and locally inspect all board entries in order to determine whether a given ciphertext is intended for them. As training progresses and the board accumulates messages, this approach leads to unbounded growth in bandwidth and storage costs. Even though BEDLAM explicitly targets settings where peers are asynchronous and low latency is not a primary concern, repeatedly downloading and scanning an ever-growing board quickly becomes impractical for most realistic distributed learning deployments.

Table 1. Relationship map between risks, adversaries and properties.

Risk	Adversaries	Properties
R1 (Message linkability over time)	A1 (Communication Observer), A2 (Network Observer), A4 (Colluding Adversaries)	P4 (Session Unlinkability), P6 (Sender–Receiver Decoupling), P8.a (Append-only Boards)
R2 (Sender-receiver association)	A1 (Communication Observer), A3 (Curious Peer), A4 (Colluding Adversaries)	P6 (Sender-Receiver Decoupling), P6.a / P6.b (Peer Unobservability), P5 (Board Obliviousness)
R3 (Peer activity inference)	A1 (Communication Observer), A2 (Network Observer), A4 (Colluding Adversaries)	P7 (Asynchronous Persistence), P4 (Session Unlinkability), P8.a (Append-only Boards)
R4 (Contribution inference)	A1 (Communication Observer), A3 (Curious Peer), A4 (Colluding Adversaries)	P4 (Session Unlinkability), P6 (Sender–Receiver Decoupling), P7 (Asynchronous Persistence)
R5 (Topology and role inference)	A1 (Communication Observer), A3 (Curious Peer), A4 (Colluding Adversaries)	P2 (Role Agnosticism), P3 (Protocol Independence), P5 (Board Obliviousness)
R6 (External temporal correlation)	A2 (Network Observer), A4 (Colluding Adversaries)	P7 (Asynchronous Persistence), P4 (Session Unlinkability), P8 (Board State Management)

As a result, efficiency emerges as one of the central practical challenges of board-based communication. Broadly speaking, two classes of approaches exist. The first is to relax the append-only assumption (P8.b), allowing the board to support message deletion, batching, expiration, or windowed storage. While such mechanisms can significantly improve efficiency, they may reintroduce metadata leakage by exposing delivery events, access timing, or synchronization artifacts. The second is to retain the append-only property (P8.a) and address efficiency at the level of message discovery and retrieval.

When using an append-only board, the most straightforward approach is for peers to periodically download the entire board and perform local filtering or cryptographic checks to identify relevant messages. While conceptually simple and privacy-preserving, this strategy scales poorly as board size increases and is unsuitable for long-running or large-scale training processes.

An alternative is to equip the board with an efficient, privacy-preserving message retrieval mechanism (P9). One possible implementation is to combine the board with Private Information Retrieval (PIR). In this setting, senders publish, alongside the board, compact encrypted *clues* associated with each message. A receiver can locally test these clues to identify the index of the message intended for it, and then retrieve only that specific entry via a PIR query. Crucially, this allows the receiver to access the desired message without revealing which board entry was retrieved, thereby preserving receiver anonymity and preventing the board from learning access patterns.

While PIR-based retrieval introduces additional computational and communication overhead compared to direct access, this cost is explicit, bounded, and independent of the total board size. In practice, it is substantially lower than the cost of repeatedly downloading and scanning the entire board as it grows. As such, PIR-style retrieval can offer

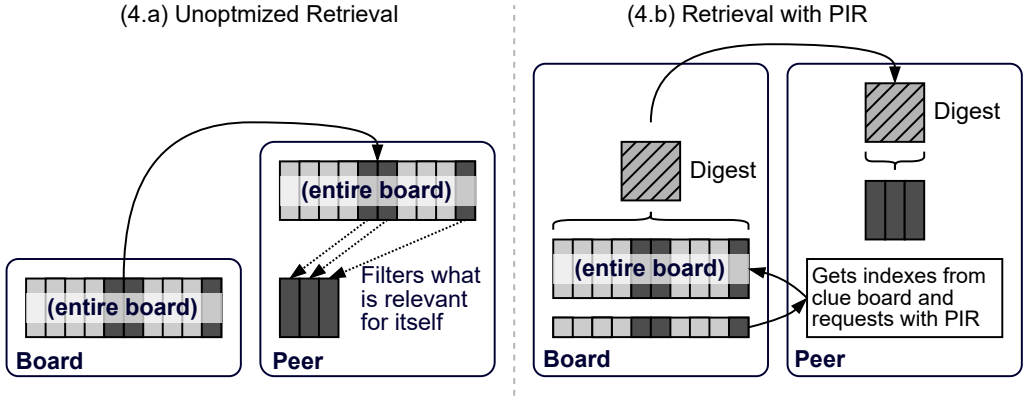

Figure 4. Illustration of the difference between retrieval optimization.

a compromise between efficiency and metadata privacy, enabling append-only boards to remain viable without sacrificing the anonymity guarantees that motivate their use.

5. BEDLAM proof-of-concept implementation and experiments

This experimental evaluation is intended as a small-scale proof-of-concept and is not representative of real-world or large-scale deployments. The goal is not to provide a comprehensive performance evaluation, but to demonstrate the feasibility of metadata-aware communication and to illustrate the qualitative overhead introduced by progressively stronger privacy properties. To this end, we implemented a proof-of-concept version of BEDLAM using simple convolutional neural networks trained under both federated and split learning paradigms on MNIST and Fashion-MNIST. These models are trained using simplified implementations of the bulletin board abstraction described earlier and are not optimized for performance.

To implement mutual peer unobservability (P6.b), we created the *blind-agreement mechanism* using a dedicated agreement sub-board. Source peers anonymously post training requests to this sub-board. Compute peers periodically scan it, select a request, and respond. During this interaction, the two peers exchange public keys and derive a shared symmetric session key, which is then used to protect subsequent message exchanges without revealing peer identities. Figure 5 illustrates the information exchanged between peers through both the agreement sub-board and the encrypted messages posted to the main board. Because message formats are agnostic to model structure and agreement mode, the same communication primitives are reused across training configurations and property combinations. To mitigate leakage through message sizes, messages in the current implementation are padded to a fixed multiple of 1024.

Table 2 reports baseline performance measurements for split learning on the Fashion-MNIST dataset under four communication configurations mediated by a bulletin board. All configurations share a minimal architectural baseline that enforces role agnosticism, protocol independence, asynchronous persistence, and a non-append-only board with bounded memory (P2, P3, P7, P8.b). The baseline configuration provides no metadata privacy guarantees beyond these architectural properties. The remaining configurations progressively enforce additional metadata-privacy properties at the communication layer.

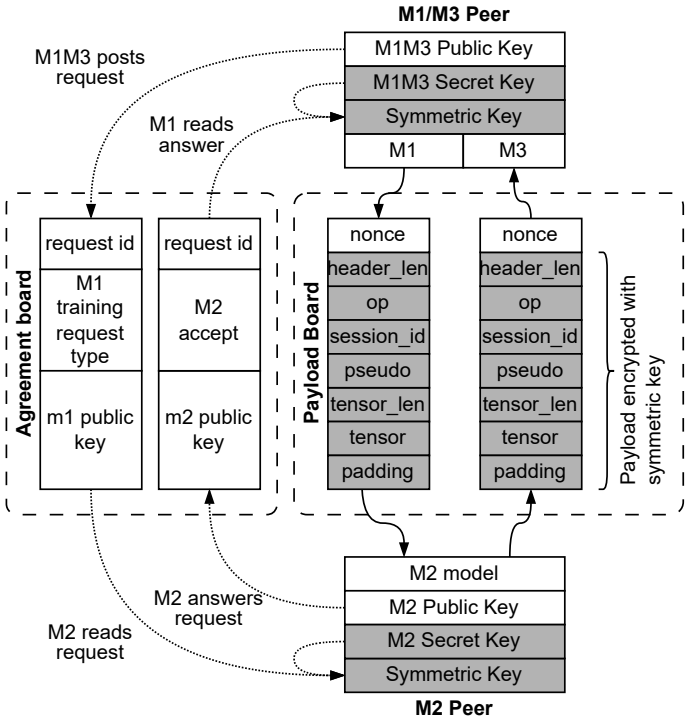

Figure 5. Message content exchanged between peers in the proof-of-concept.

Specifically, we evaluate: a plaintext baseline with no metadata protection (**Baseline**), a board-oblivious configuration enforcing payload opacity and board obliviousness (**"board-blind"**, base+P1+P5), a source-unobservable configuration enforcing source unobservability at the compute peer (**"single-blind"**, base+P1+P5+P6.a), and a mutually unobservable configuration enforcing full peer anonymity (**"double-blind"**, base+P1+P5+P6.a+P6.b). All experiments use the same convolutional neural network architecture, batch size 128, and Adam optimizer with learning rate 10^{-3}.

Table 2 shows the runtime and communication overhead incurred by progressively stronger metadata-privacy properties. We report per-step averages for runtime (in milliseconds)[4] and communication cost (in bytes)[5]. As expected, the baseline configuration is the fastest, as it avoids cryptographic processing and additional protocol logic. This, of course, is still slower than any protocol without a board or metadata protection.

Enforcing payload encryption and board obliviousness (Board-blind) introduces only modest overhead, mainly due to serialization and lightweight cryptographic encapsulation. Adding source unobservability at the compute peer (Single-blind, P6.a) further increases cryptographic and protocol overhead, but has limited impact on overall runtime.

The largest overhead appears in the Double-blind configuration, where mutual peer unobservability (P6.b) requires the blind-agreement mechanism and more frequent key rotation. This leads to higher cryptographic processing, serialization cost, and proto-

[4]Per-step runtime is decomposed into model computation (*Comp.*), message serialization/deserialization (*Ser.*), cryptographic processing (*Crypto*), and waiting time (*Wait*) spent polling the board or awaiting peer responses

[5]Communication cost is decomposed into tensor payloads (*Payload*), padding for obfuscation (*Padding*), and protocol overhead (*Overh.*), including headers and cryptographic metadata.

Table 2. Split-learning performance and communication breakdown in BEDLAM.

Properties	Time (ms / step)				Bytes (per step)		
	Comp.	**Ser.**	**Crypto**	**Wait**	**Payl.**	**Pad**	**Overh.**
Baseline (P2, P3, P7, P8.b)	68.10	0.66	-	37.85	-	-	-
Board-blind (base+P1+P5)	72.93	1.41	0.56	39.39	197K	2824.8	855.2
Single-blind (base+P1+P5+P6.a)	71.92	1.58	1.56	41.97	197K	2824.8	897.2
Double-blind (base+P1+P5+P6.a+P6.b)	73.55	**3.41**	**15.72**	42.01	197K	2824.8	**1300.4**

col metadata. Overall, the overhead increases proportionally with the privacy guarantees enforced, while model computation remains the dominant cost.

6. Limitations, Conclusion and Future Work

This work examined metadata privacy as a separate problem in distributed learning. We identified metadata-related risks that remain even when message contents are protected, and defined a threat model in which an adversary can infer sensitive information from communication patterns, timing, and system structure. To address this problem, we proposed *Board-Enabled Distributed Learning via Anonymous Messaging* (BEDLAM). BEDLAM adds a communication layer to distributed learning systems instead of changing the learning protocol itself. This layer can provide properties such as role agnosticism, unlinkability, and peer obliviousness, while remaining compatible with existing federated and split learning methods. These properties can be enabled selectively depending on the threat model and the acceptable cost.

We implemented a toy version of BEDLAM and evaluated it on small-scale split learning workloads. The experiments show that progressively stronger metadata-privacy properties can be enforced without disrupting training correctness. As expected, these properties introduce additional overhead due to cryptographic processing and protocol coordination. However, the overhead is explicit and proportional to the privacy guarantees provided, and does not dominate overall runtime in the evaluated settings, demonstrating the feasibility of metadata-aware communication in controlled settings.

This study, however, has clear limitations. The implementation is a proof of concept rather than a production system, and the evaluation is restricted to lightweight models and small datasets, without validation under large-scale or realistic deployment conditions. Moreover, the privacy guarantees are assessed through threat modeling and architectural analysis rather than formal proofs, and no formal privacy guarantees are established (yet). System parameters such as padding, polling frequency, and board memory management are fixed rather than optimized. Finally, from a practical perspective, BEDLAM is most beneficial in settings where metadata privacy is a primary concern, such as sensitive domains involving personal, medical, or confidential data, where the additional overhead introduced by metadata protection may be justified by the reduction in inference risks. In less sensitive or performance-critical environments, simpler communication mechanisms may remain preferable.

Acknowledgements

The authors thank Paulo Pacitti and Prof. Hilder Vitor Lima Pereira for their suggestions and comments during the development of this work. The authors also acknowledge the

use of Artificial Intelligence tools (ChatGPT and Grammarly for textual revision of the manuscript, and Codex and Claude for the development of the implementation) - however, all scientific decisions, experiments, interpretations, and conclusions were made by the authors, who remain fully responsible for the content of this work.

References

Almashaqbeh, G. and Ghodsi, Z. (2023). Anofel: Supporting anonymity for privacy-preserving federated learning.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. (2016). Practical secure aggregation for federated learning on user-held data.

de Retana, M. F., Zulaika, U., Sánchez-Corcuera, R., and Almeida, A. (2025). Differential privacy: Gradient leakage attacks in federated learning environments.

Egger, M., Urbanke, R., and Bitar, R. (2025). Federated one-shot learning with data privacy and objective-hiding.

Li, Z., Lowy, A., Liu, J., Koike-Akino, T., Parsons, K., Malin, B., and Wang, Y. (2024). Analyzing inference privacy risks through gradients in machine learning.

Liu, K. and Gupta, T. (2024). Federated learning with differential privacy and an untrusted aggregator. In *Proceedings of the 10th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP*, pages 379–389. INSTICC, SciTePress.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2016). Communication-efficient learning of deep networks from decentralized data.

Mozaffari, H., Marathe, V., and Dice, D. (2022). Private and robust federated learning using private information retrieval and norm bounding. In *Workshop on Federated Learning: Recent Advances and New Challenges (in Conjunction with NeurIPS 2022)*.

Nguyen, K., Khan, T., and Michalas, A. (2023). Split without a leak: Reducing privacy leakage in split learning.

Rangwala, M., Sinnott, R. O., and Buyya, R. (2025). Network structures as an attack surface: Topology-based privacy leakage in federated learning.

Shuvo, M. N. H. and Hossain, M. (2025). Fingerprinting deep learning models via network traffic patterns in federated learning. In *Proceedings of the 2025 ACM Workshop on Wireless Security and Machine Learning*, page 32–37. ACM.

Shuvo, M. N. H., Hossain, M., Mallik, A., Twigg, J., and Dagefu, F. (2025). Flare: A wireless side-channel fingerprinting attack on federated learning.

Vepakomma, P., Gupta, O., Swedish, T., and Raskar, R. (2018). Split learning for health: Distributed deep learning without sharing raw patient data.

Wei, W., Liu, L., Loper, M., Chow, K.-H., Gursoy, M. E., Truex, S., and Wu, Y. (2020). A framework for evaluating gradient leakage attacks in federated learning.

Zhu, L., Liu, Z., and Han, S. (2019). Deep leakage from gradients.

Zhu, X., Luo, X., Wu, Y., Jiang, Y., Xiao, X., and Ooi, B. C. (2025). Passive inference attacks on split learning via adversarial regularization.