A Queue-based Mechanism for Unlinkability under Batched-timing Attacks

Alexander Goldberg
Carnegie Mellon University
akgoldbe@andrew.cmu.edu

Giulia Fanti
Carnegie Mellon University
gfanti@andrew.cmu.edu

Nihar B. Shah
Carnegie Mellon University
nihars@cs.cmu.edu

Abstract

In privacy-sensitive systems where participants operate under pseudonyms, timing information can be exploited to compromise anonymity. Motivated by applications in online peer review forums and cryptocurrency transactions, we consider deanonymization risk arising due to batching—multiple actions taken by a user at nearly the same time. We provide a formulation of privacy against batching attacks where an adversary has knowledge of a probabilistic model generating the data. We provide a queue-based algorithm that introduces delays to the system to prevent linking of actions to the same user and give theoretical results that demonstrate that it is possible to provide formal privacy guarantees without introducing excessive delay to the system. Then, we show that given problem constraints, it is not possible to defend against a stronger adversary modeled by standard differential privacy definitions. In particular, we prove that if an algorithm must release all the data it receives and no fake data, it is not possible for the algorithm to be differentially private in our setting.

1 Introduction

In a number of applications where anonymity is critical, users take public actions under pseudonyms to preserve their privacy. For instance, in many academic peer review conferences, reviewers make comments that are publicly viewable. They use different pseudonyms across multiple papers to remain anonymous. In cryptocurrencies like Bitcoin, users' transaction histories are recorded on a public blockchain where a person can send or receive currency to an associated public key. In order to preserve privacy, it is considered best practice to use multiple public keys to make it difficult to link transactions coming from the same person [24]. In both of these settings, anonymity is a key property of the system.

Additionally, in both settings it is common for users to engage in *batching*. Batching is the completion of several similar tasks by the same person at the same time. Batching occurs both due to natural bursts in activity (e.g., a person visits a web forum and makes many comments at once) or as a "productivity hack" used to streamline work. Indeed, both academic studies [14, 17, 1, 9, 2] and popular media [19, 20, 15] recommend performing tasks in batches in order to improve efficiency and reduce work-related stress. Batching behavior appears in many tasks from email, to data-entry to responding to mobile phone notifications. In the context of cryptocurrencies, batched transactions can also occur for different reasons. When a user has multiple public addresses and wishes to make a large transaction that exceeds the balance of any single address, they need to batch transactions by sending money from different addresses.

While batching can be useful for productivity purposes, it presents a privacy challenge, as correlated timing of actions can enable an adversary to link identities across pseudonyms and then deanonymize these linked pseudonyms. We are motivated by the following concrete scenarios where batched actions enable such attacks:

Application 1: Inferring the identity of commenters in anonymous open forums. There are online forums where participants can comment anonymously under multiple pseudonyms. For instance, many popular academic conferences use an "open review" model (like OpenReview.net) for peer review—after initial reviews are posted, reviewers anonymously post public comments on papers. In practice, a single reviewer is assigned to multiple papers within a conference and paper authors do not know the assignments. Reviewers who batch their work, that is, post comments on different papers at (nearly) the same time, may have their anonymity compromised. To see this, suppose that a person is both a meta reviewer for some papers and an author of another paper. This person then knows the identities of the reviewers of some papers. This person can then observe concurrently-arriving comments on their own paper and another paper with known reviewers. By linking a comment on their own paper to a comment on another paper, this person can identify their paper's reviewer(s).

In a similar application, Wikipedia provides public edit histories of articles. Their terms of services [21] explicitly highlight the legitimacy of maintaining a pseudonymous alternate account to edit controversial articles. However, batched edits can compromise the anonymity of a privacy-sensitive editor's pseudonymous second account.

Application 2: Linking cryptocurrency transactions on a blockchain. When a user initiates a transaction in a blockchain system, the first step is to broadcast the transaction over the underlying communication network. For example, in Bitcoin, this communication network is a peer-to-peer network of nodes. If a user wishes to make a large transaction, they may need to use multiple addresses holding cryptocurrency in order to send enough total tokens; each transfer from a single address manifests as a single transaction. Naively, a user might batch these multiple transactions by broadcasting them simultaneously. However, an adversary could use the timing of these batched transactions to link together the sending addresses. Linking pseudonyms together is a common first step in a full deanonymization attack. For instance, attacks on Bitcoin transactions [18] begin by leveraging a user's "idioms of use" to cluster together addresses likely belonging to the same person. The attacker then leverages a single known link to a real-world identity to deanonymize the entire cluster.

Our contributions: The scenarios discussed above motivate the need for defenses against linkage attacks using batched timing, and in this work we identify and formulate this problem. First, we provide a definition of privacy against batching where an adversary has knowledge of a probabilistic model generating the data and wants to distinguish between a generating process where batching occurred and one without batching. We give a queue-based mechanism that delays batched actions to combat timing attacks in this setting and provide theoretical results on the privacy and utility (as measured by delay introduced by the mechanism) of our algorithm. Then, we show that it is not possible to defend against a more knowledgeable adversary by providing differential privacy guarantees. In particular, we prove that under the constraint that an algorithm must release all the data it receives and no fake data, it is not possible to meet standard differential privacy guarantees in our setting.

2 Related Work

There is a substantial body of work studying anonymity when sending packets over a network, however, the techniques developed there are inapplicable to our setting. Prior work has described de-anonymization attacks which leverage correlated timing of packet arrivals and defenses against such attacks [23, 16, 12, 10]. In anonymous networking, the goal is to prevent an adversary from inferring the sender and recipient of a given message. Packets are routed through a sequence of "mix nodes" to obscure the path taken. However, the highly correlated arrival times of packets on the first mix node and the last mix node in one path can enable inferences that a specific sender and recipient are communicating with one another. Prior work in [23, 16] demonstrates the practical viability of de-anonymization attacks that take advantage of batching in anonymous networks.

The defenses proposed in these papers rely on introduction of *dummy packets* or "cover traffic" to a network, obscuring any instance of batching amidst many instances of spurious batching. In contrast, a critical constraint in the settings we consider is the *infeasibility of generating fake data* as a means of preserving

privacy. In the commenting setting, it is undesirable to generate fake comments as this would require giving made-up feedback to paper authors. For cryptocurrencies, introducing dummy transactions would require additional transaction fees representing undesirable overhead. Furthermore, transactions include the amount of currency sent, so dummy transactions would require a sender to transfer actual funds just to preserve privacy. Therefore, our work will consider mechanisms that delay batched arrivals in order to preserve anonymity, trading off delay for privacy without introducing any synthetic data.

Our running application in this paper is that of peer review. The papers [4, 11] address issues of privacy when releasing certain types of peer review data (e.g., for academic research), and [3] addresses privacy compromise when a conference corrects for miscalibration of reviewers. See [22] for an overview of research on peer review.

3 Problem Setup

We now describe the problem formulation. For concreteness, we state the setting in terms of the peer review application.

Papers and Reviewers. Let $\mathcal{P}$ denote a set of papers at a conference and let $\mathcal{R}$ denote the set of reviewers. Each reviewer is assigned to a small subset of papers and each paper is assigned to a small subset of reviewers, but we make no additional assumptions about the assignment. The assignment of reviewers to papers is private information.

Comment Arrivals. We consider peer review on an open forum like OpenReview.net. After the initial review submission phase, reviewers can make publicly visible comments on papers. Each comment is associated with three pieces of metadata $\langle t, p, r\rangle$: the timestamp t when it arrived, the paper $p \in \mathcal{P}$ to which it corresponds, and the reviewer $r \in \mathcal{R}$ who made the comment. The reviewer identity is obscured by a per-paper pseudonym (e.g., "Anonymous Reviewer 2"). However, the timestamp is publicly observable. Assume that comments arrive over an *infinite time horizon* in *discrete time*. Time starts at time-step 1. We call the event when a comment is made by a reviewer a "comment arrival." For all $t \geq 1$, let A_t denote the set of comments that arrive at time-step t and $A = A_1, A_2, \ldots,$ be the infinite sequence of sets of comment arrivals.

We consider the following model of comment arrivals. First, we present the arrival process if no batching occurs. In the absence of batching, a single comment arrives at every unit of time. We make an *IID assumption* on arrivals. At each time-step, the paper-reviewer pair associated with the comment is drawn independently from a probability distribution $\mathcal{D}$ over $\mathcal{P} \times \mathcal{R}$. For instance, $\mathcal{D}$ could be a uniform distribution over $\mathcal{P} \times \mathcal{R}$ although it need not be uniform or even known to the algorithm. We say $A \leftarrow \mathcal{A}^{(0)}$ if the arrivals are drawn from this no batching process.

Batching. An instance of *potential batching* consists of multiple comments. The batch arrives at a single time-step, but the adversary is uncertain as to which papers and reviewers are in the batch. Thus, when potential batching occurs, the arrival process remains the same except for one modification - batches consisting of more than one comment arrive at some time-steps. Formally, let $B = (S, m) : S \subseteq \mathbb{N}, m : S \to \mathbb{N}$ be a multi-set of time-steps at which batching occurs. Then, for each $t \in S$ a batch of size $m(t) + 1$ arrives. For instance, if $B = \{10, 10, 15\}$ then a batch of size 3 arrives at time 10 and a batch of size 2 arrives at time-step 15. The paper-reviewer pairs associated with the batched comments are drawn independently with replacement from distribution $\mathcal{D}$. We say that $A \leftarrow \mathcal{A}^{(B)}$ if the arrivals are drawn from this process with batchings occurring at time-steps in $\mathcal{A}^{(B)}$. We allow comments to arrive according to $\mathcal{A}^{(B)}$ for any finite multi-set of time-steps B where B is not known to the algorithm a priori.

Batched comment arrivals present a deanonymization risk. In particular, if an observer sees multiple comments arrive at the same time on different papers, they may infer that the comments came from the same reviewer. This linking of anonymous reviewers can then be used to compromise anonymity as we described in Section 1.

Comment Posting Mechanism. Currently, comments are posted to the forum as soon as they arrive. In order to obfuscate timing information, we propose using an intermediary comment-posting mechanism that

receives all comments on papers as they arrive and can delay their post times. In the case of peer review, the mechanism could be implemented by the online forum hosting reviews and comments. An observer of the online forum only sees the times at which comments are posted after (potential) delay by the mechanism, but not the time at which the comment actually arrives. In our setting, we consider any *valid* comment-posting mechanism to be a mechanism that must release all comments that arrive without any fake comments introduced:

Definition 3.1 (Valid Comment Posting Mechanism)**.** A *valid comment posting mechanism* receives an infinite sequence of comment arrivals as input and outputs a set of comments respecting the following properties:

1. (*Delay-Only*) If a comment arrives at time t it must be outputted at time $t' \geq t$.
2. (*No Fake Data*) Any comment posted through time t must have arrived at or before time t.
3. (*No Withholding Data*) Let d denote the potentially randomized delay introduced to a comment by the mechanism. For any comment, it must be that $\lim_{m\to\infty} \Pr[d \leq m] = 1$.

Privacy. In order to prevent an adversary from linking batched comments, we want to design a privacy-preserving comment-posting mechanism that delays batched comments in order to obfuscate whether any batching occurred. Our definition of privacy takes inspiration from the popular notion of differential privacy [6]. Roughly, a mechanism is differentially private if the distribution of mechanism outputs does not change much on "neighboring" inputs. In our case, an adversary observing potentially delayed comments posted by the mechanism C_T over any time horizon T should not be able to distinguish whether the arrivals A were drawn according to the *no batching* process or the *potential batching* process. Thus:

Definition 3.2 (Batching Privacy)**.** A comment-posting mechanism $\mathcal{M}$ is ϵ-*batching private* with respect to arrival processes $(\mathcal{A}^{(0)}, \mathcal{A}^{(B)})$ if for all time horizons $T \geq 1$, all finite batching multi-sets B, and any output of the mechanism between time 1 and T, C_T:

$$e^{-\epsilon} \leq \frac{\Pr[\mathcal{M}(A) = C_T; A \leftarrow \mathcal{A}^{(B)}]}{\Pr[\mathcal{M}(A) = C_T; A \leftarrow \mathcal{A}^{(0)}]} \leq e^{\epsilon}$$

Note that unlike typical differential privacy formulations, our notion of privacy requires distributional assumptions on the data-generating process as we assume that comments are generated by an IID arrival model. In Section 5, we motivate the need for stronger assumptions in our setting by demonstrating that standard guarantees of differential privacy are not possible to provide for any *valid* comment-posting mechanism.

Utility (Delay). The cost of anonymity is additional delay, as some comments may be delayed in order to obscure batching. We capture the delay cost as the *worst-case expected delay of a comment*. That is, letting d_c denote the delay on any comment c, we want to minimize $\sup_{c \in \text{comments}} \mathbb{E}[d_c]$. We consider the worst-case over individual comments rather than overall expected delay in order to guarantee that no single comment is delayed for an excessively long time.

4 Algorithm

We propose a mechanism that uses a queue to delay comments and make the no-batching process indistinguishable from any process where batching occurs. In this section, we describe the mechanism, prove that it guarantees privacy, and show that utility is optimal among algorithms preserving batching privacy. The algorithm is described below as Algorithm 1.

Theorem 4.1 (Analysis of Algorithm 1)**.** *When Algorithm 1 is applied to any comment sequence drawn from $\mathcal{A}^{(0)}$ or $\mathcal{A}^{(B)}$:*

(1) (Privacy) The algorithm guarantees perfect batching privacy ($\epsilon = 0$).

(2) (Delay) The worst-case delay of any comment is $|B|$.

Algorithm 1

Initialize empty queue $Q = \emptyset$
for t= $1, 2, \ldots$ **do**
  if set of batched comments A arrives **then**
    if $Q \neq \emptyset$ **then**
      Dequeue comment c' from Q and post it.
      Enqueue all comments in A to Q in a random order.
    else
      Choose $c \in A$ uniformly at random to post.
      Enqueue all comments in $A \setminus \{c\}$ to Q in a random order.
      Post comment c immediately.
    end if
  else if a single comment c arrives **then**
    if Q$\neq \emptyset$ **then**
      Dequeue comment c' from Q and post it.
      Enqueue comment c to Q.
    else
      Post comment c.
    end if
  end if
end for

Proof. First, we prove privacy of the algorithm. Fix a time horizon T and multi-set of batching times B. We let $\mathcal{D}(C)$ denote the probability of observing the comment c under distribution $\mathcal{D}$. When the algorithm is applied to comments drawn according to the no batching process, one comment arrives at each time-step and all comments are posted immediately so by the IID assumption, $\Pr[\mathcal{M}(A) = C_{1:T}; A \leftarrow \mathcal{A}^{(0)}] = \mathcal{D}(C_t)^T$.

If comments were drawn according to the process where batching occurred at times B, then at any time-step before the first instance of batching occurs the mechanism posts the single comment that arrives so the probability of observing output $\{c\}$ is $\mathcal{D}(c)$ independent of other-timesteps. On the first instance of batching, the mechanism posts one of the batched comments chosen uniformly at random from the batch, so due to the iid arrivals of the batch the probability of observing this output $\{c\}$ at this time-step is also $\mathcal{D}(c)$. At any later time-step, the algorithm posts the comment at the top of the queue, which consists of previous comments that arrived iid drawn from $\mathcal{D}$. Therefore, the probability of observing any output is still $\Pr[\mathcal{M}(A) = C_{1:T}; A \leftarrow \mathcal{A}^{(B)}] = \mathcal{D}(C_t)^T$.

Now, we argue that delay is $|B|$ for all comments arriving after the last instance of batching in B. In particular, at this time $T + |B|$ comments have arrived in total. The mechanism posts the earliest-arriving comment at each time-step and delays the incoming comment so the queue has length $|B|$ and any single incoming comments are delayed for $|B|$ timesteps before being posted.

□

Additionally, the mechanism is optimal in preserving indistinguishability of $\mathcal{A}^{(0)}$ and $\mathcal{A}^{(B)}$ in that any mechanism that preserves batching-privacy introduces the same worst-case delay.

Theorem 4.2 (Lower Bound)**.** *Any comment-delaying mechanism guaranteeing ϵ-batching privacy with $\epsilon < \infty$ for comments arriving according to $\mathcal{A}^{(0)}$ and $\mathcal{A}^{(B)}$ must introduce delay of at least $|B|$ to at least one comment when applied to comments arriving according to $\mathcal{A}^{(B)}$.*

Proof. Let $T' = \max\{B\}$ be the latest time-step when batching occurs and $T = T' + |B|$. Then, if comments arrive according to $\mathcal{A}^{(B)}$, $T' + |B| + 1 = T + 1$ comments arrive up until time $T' + 1$. Assume for the sake of contradiction that all of the comments arriving before time $T' + 1$ are posted with delay strictly less than $|B|$. Then, when acting on comments arriving according to $\mathcal{A}^{(B)}$, the mechanism must post at least $T + 1$ comments within time horizon T (with probability 1). However, under arrival process $\mathcal{A}^{(0)}$, only T comments

have arrived up until T, so no mechanism can ever output $T+1$ comments up until time T. Hence, any output of the mechanism up until time T on comments arriving per $\mathcal{A}^{(B)}$ contains $T+1$ comments with probability 1, while for comments arriving per $\mathcal{A}^{(0)}$ any output up until time T contains $T+1$ comments with probability 0. □

Remark 4.3. One attempt to circumvent the lower bound is to define privacy between processes where the total number of batched arrivals is fixed, but the timing of the batchings is unknown. For instance, a privacy guarantee could be defined with respect to all pairs of arrival processes $\mathcal{A}^{(B_1)}$ and $\mathcal{A}^{(B_2)}$ corresponding to multi-sets B_1 and B_2 where $|B_1| = |B_2|$. However, the lower bound on utility would still apply in this case. Consider two arrival processes with the same total number of batchings but in one process all of the batched comments arrive much earlier than in the other: $\min\{B_2\} - \max\{B_1\} > |B_1|$. Then, $\mathcal{A}^{(B_2)}$ looks like a no batching process up until time $\max\{B_1\} + |B_1| + 1$ hence the lower bound on delay still applies.

5 Impossibility of Differential Privacy

In our privacy formulation, an adversary knows that comments arrived according to one of two probabilistic models and tries to infer from which model the comment arrived given the post times of comments. A natural question to ask is - *can we guarantee privacy against a stronger adversary who knows the arrival times of all but one potentially batched comment as in standard formulations of differential privacy?*

Differential privacy guarantees that the distribution of mechanism outputs does not change very much on "neighboring" inputs differing in the timing of one comment. There are two natural notions of neighbors in the batching setting - either batching could lead to the introduction of an additional comment or the total number of comments could be fixed and batching leads to a change in the timing of a single comment. These are analogous to two different DP definitions, sometimes called "unbounded" and "bounded" differential privacy [13]:

Definition 5.1 (Unbounded Differential Privacy, [5])**.** A mechanism $\mathcal{M}$ satisfies (ϵ, δ)-*unbounded DP* if for any finite time horizon T and any set of outputs during this time period S_T,

$$\Pr[\mathcal{M}(A) \in S_T] \leq e^{\epsilon}\Pr[\mathcal{M}(A') \in S_T] + \delta$$

where A and A' are two "neighboring" comment arrival sequences such that one can be obtained from the other by *adding or removing a batched comment at some time-step.*

Definition 5.2 (Bounded Differential Privacy, [7])**.** A mechanism $\mathcal{M}$ satisfies (ϵ, δ)-*bounded DP* if for any finite time horizon T and any set of outputs during this time period S_T,

$$\Pr[\mathcal{M}(A) \in S_T] \leq e^{\epsilon}\Pr[\mathcal{M}(A') \in S_T] + \delta$$

where A and A' are two "neighboring" comment arrival sequences such that one can be obtained from the other by *moving a batched comment at one time-step to another time-step where it is no longer batched.*

Remark 5.3. Our formulation in Section 3 aligns with the "unbounded" DP model in that additional comments arrive due to batching. In this section, we give results in both settings as they pose distinctive privacy challenges. We leave open the problem of formulating a relaxed version of bounded differential privacy for future work.

Unfortunately, it is impossible to simultaneously satisfy the constraints of a *valid* comment-posting mechanism (Definition 3.1) that releases all comments without generating fake data, while preserving differential privacy for any $\epsilon < \infty$ or $\delta < 1$. Intuitively, it is impossible to guarantee unbounded differential privacy because if an extra comment arrives in a batch then this comment must be withheld forever since it does not appear at all in some adjacent database and we cannot generate fake comments. It is impossible to preserve bounded differential privacy, because a comment that arrived in a batch may arrive at an arbitrarily later time in an adjacent database and must be withheld for at least this time.

Theorem 5.4 (Impossibility of Unbounded Differential Privacy)**.** *There is no valid comment-posting mechanism acting on an infinite stream of comment arrivals that is* (ϵ, δ)*-unbounded DP for* $\epsilon < \infty, \delta < 1$*.*

Proof. Consider any input A where an instance of batching occurs at some time-step t. Let A' be identical to A, except some comment c that arrived in a batch at time t does not arrive at all in A'. Then on input A', since any valid comment delaying mechanism cannot generate fake data, for any $d > 0$ and time $T = t + d$, the mechanism outputs c at time T with probability 0. Thus, if the mechanism is (ϵ, δ)-unbounded DP with $\epsilon < \infty$, then for any d the mechanism outputs C within that delay with probability at most δ and so the mechanism violates the no-withholding property. □

Theorem 5.5 (Impossibility of Bounded Differential Privacy)**.** *There is no valid comment-posting mechanism acting on an infinite stream of comment arrivals that is ϵ-bounded DP for $\epsilon < \infty, \delta < 1$.*

Proof. Consider any input A with an instance of batching that occurs at some time-step t. Fix any time horizon $T = t + d$ where $d > 0$. Define A' to be an identical sequence with one comment moved from time t to time T: $A'_T = A'_T \cup \{c\}$, $A'_t = A_t \setminus \{c\}$,and $A_i = A'_i \; \forall i \notin \{t, T\}$. Since a valid comment-posting mechanism must delay comments and cannot generate fake data, the mechanism outputs comment c at time T or later on input A' with probability 1. However, if the mechanism is (ϵ, δ)-bounded DP with $\epsilon < \infty$ then it must delay comment c until at least time $T = t + d$ with probability at least $1 - \delta 1$. Taking d to be arbitrarily large, the mechanism violates the no-withholding data property for any $\delta < 1$. □

Remark 5.6. If we considered mechanisms acting on a finite sequence of comment arrivals, then the proof above would suggest that the only valid mechanism is one that delays all comments until the end of the finite time horizon and then releases all comments at that time. This is both intuitively and formally good for preserving privacy from timing attacks since it eliminates all timing information, but is expensive in terms of delay. In particular, in the peer review setting, releasing all comments simultaneously at the end of the review period eliminates potential for replies and ongoing discussion.

Now, in our proof above, the impossibility of bounded differential privacy arose due to the possibility that a comment moved from arriving in a batch to arriving at an (arbitrarily larger) later time. A natural assumption to try to resolve this issue is to limit the time-gap over which a comment can move on neighboring inputs:

Definition 5.7 (g-Bounded Differential Privacy)**.** A mechanism $\mathcal{M}$ satisfies (ϵ, δ, g)-*bounded DP* if for any finite time horizon T and any set of outputs during this time period S_T,

$$\Pr[\mathcal{M}(A) \in S_T] \leq e^{\epsilon} \Pr[\mathcal{M}(A') \in S_T] + \delta$$

where A and A' are two "neighboring" comment arrival sequences such that one can be obtained from the other by *moving a batched comment by at most g time-steps to another time-step where it is no longer batched.*

However, it is still impossible for meaningful values of δ to preserve privacy with a reasonable bound on delay in this formulation because the mechanism can only delay comments (not move release times earlier) while the DP guarantee is symmetric. Therefore, considering a sequence of adjacent inputs to the mechanism where a comment c is arrives slightly later in time as the sequence progresses, on a later database in the sequence c must be posted at a late time-step with probability 1, so an earlier database c is posted with a high delay with probability close to 1 in order to meet the symmetric privacy guarantee between each pair of adjacent databases.

Theorem 5.8 (Necessary Delay of g-Bounded Differential Privacy)**.** *For any valid comment-posting mechanism that satisfies (ϵ, δ, g)-unbounded DP for $\epsilon < \infty, \delta < 1$ and $g \geq 1$, there is an infinite stream of comment arrivals on which the mechanism delays a comment by at least m with probability at least $1 - 2\delta(m + 1)$.*

Proof. Let $A^{(1)}$ be an input where a single comment arrives at each time-step, so $A_1^{(1)} = \{c_1\}, A_2^{(1)} = \{c_2\}$ and so on. Then, define $A^{(1)'}$ to be a neighboring input to $A^{(1)}$ where c_2 arrives in a batch with c_1 at time-step 1. Define $A^{(2)}$ to be a neighboring input to $A^{(1)'}$ where c_1 and c_2 arrive separately with c_1 at time 2 and c_2 at time 1 and so on:

$$A^{(1)} = \{c_1\}, \{c_2\}, \{c_3\}, \cdots$$
$$A^{(1)'} = \{c_1, c_2\}, \emptyset, \{c_3\}, \cdots$$
$$A^{(2)} = \{c_2\}, \{c_1\}, \{c_3\}, \cdots$$
$$A^{(2)'} = \{c_2\}, \{c_1, c_3\}, \emptyset, \cdots$$

Now, for any j: $A^{(j)}$ and $A^{(j)'}$ are neighbors and $A^{(j)}$ and $A'^{(j-1)}$ are neighbors as defined in g-Bounded Differential Privacy for $g = 1$. On input $A^{(j)}$, comment c_1 arrives at time j and so any valid comment-posting therefore posts c_1 at time j or later with probability 1 since it can only delay comments. Likewise, because $A^{(j-1)'}$ neighbors $A^{(j)}$ and the mechanism cannot generate fake data, any (ϵ, δ)-DP mechanism releases c_1 at a time earlier than j with probability at most δ on input $A^{(j-1)'}$. Since $A^{(j-1)}$ neighbors $A^{(j-1)'}$ the mechanism releases c_1 at a time earlier than j with probability at most 2δ on this input. Thus, on input $A^{(1)}$ comment c_1 gets posted before time j with probability less than $2j\delta$. This suggests, that the comment gets delayed by at least m with probability at least $1 - 2\delta(m + 1)$. □

Remark 5.9. It follows from Theorem 5.8 that there is no valid comment delay mechanism satisfying g-Bounded Differential Privacy with $\delta = 0$, since any private mechanism would violate the non-withholding property of valid comment-delay mechanisms. Additionally, even for $\delta > 0$, the probability of experiencing a delay longer than m only decreases *linearly* in δ and m. Considering a finite time horizon T, so the arrival set with one comment per time-step contains T comments in total. Typically, δ should be taken to be much smaller than the inverse of the size of the database ($1/T$ here) [8]. So, even for large but reasonable δ (e.g. $\frac{1}{T^2}$) any mechanism needs to delay comments until the end of the time-period T with high probability.

6 Discussion

Our work is a starting point in guaranteeing anonymity in systems where the timing of batched actions can compromise privacy. We provide initial results in a simplified setting where time is discrete and comments arrive according to an IID model, with additional comments arriving if batching occurs. We also demonstrate the difficulty of preserving privacy in our setting where all inputs must be outputted without synthetic data introduced. There are a number of open challenges remaining in making our algorithm deployable:

- Most critically, the comment arrival model will not adhere exactly to the stylized IID model. It is important to study the properties of privacy and utility under further relaxations to the model or under mismatch between model assumptions and real-world arrivals.
- Not all participants may be willing to experience increased delay to preserve privacy. This raises the question: is it possible to design an algorithm that allows some users to opt out of anonymity-preserving delays while still preserving anonymity for those who opt in?
- An active adversary can make comments of their own. They can then observe delay added to their own comments by the privacy-preserving mechanism in order to preserve privacy of some other comments. By observing delays, the adversary learns about the batching of other comments.
- Our algorithm protects against batching when additional comments arrive due to batching. In some scenarios, the set of comments may be fixed, while batching varies the timing of a comment. It is worth considering privacy-preserving algorithms for this case and whether it is possible to protect against both types of batching simultaneously.
- Delaying a comment may actually change the arrival process of subsequent comments, and the mechanism and its guarantees should be robust to this.

In conclusion, the problem of preserving anonymity against batched-timing attacks raises many exciting new theoretical questions that can move us closer to practical solutions in impactful domains like peer review and cryptocurrency.

References

[1] C. Blank, S. Zaman, A. Wesley, P. Tsiamyrtzis, D. R. Da Cunha Silva, R. Gutierrez-Osuna, G. Mark, and I. Pavlidis. *Emotional Footprints of Email Interruptions*, page 1–12. Association for Computing Machinery, New York, NY, USA, 2020. 1

[2] J. W. Borghouts, D. P. Brumby, and A. L. Cox. Batching, error checking and data collecting: Understanding data entry in a financial office. In *ECSCW Exploratory Papers*, 2017. 1

[3] W. Ding, G. Kamath, W. Wang, and N. B. Shah. Calibration with privacy in peer review. In *ISIT*, 2022. 2

[4] W. Ding, N. B. Shah, and W. Wang. On the privacy-utility tradeoff in peer-review data analysis. In *AAAI Privacy-Preserving Artificial Intelligence (PPAI-21) workshop*, 2020. 2

[5] C. Dwork. Differential privacy. In M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, editors, *Automata, Languages and Programming*, pages 1–12, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. 5.1

[6] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, page 265–284, Berlin, Heidelberg, 2006. Springer-Verlag. 3

[7] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In S. Halevi and T. Rabin, editors, *Theory of Cryptography*, pages 265–284, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. 5.2

[8] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, aug 2014. 5.9

[9] N. Fitz, K. Kushlev, R. Jagannathan, T. Lewis, D. Paliwal, and D. Ariely. Batching smartphone notifications can improve well-being. *Computers in Human Behavior*, 101:84–94, 2019. 1

[10] O. Javidbakht and P. Venkitasubramaniam. Delay anonymity tradeoff in mix networks: Optimal routing. *IEEE/ACM Transactions on Networking*, 25(2):1162–1175, 2017. 2

[11] S. Jecmen, H. Zhang, R. Liu, N. B. Shah, V. Conitzer, and F. Fang. Mitigating manipulation in peer review via randomized reviewer assignments. In *NeurIPS*, 2020. 2

[12] S. Kadloor, P. Venkitasubramaniam, and N. Kiyavash. Preventing timing analysis in networks: A statistical inference perspective. *IEEE Signal Processing Magazine*, 30(5):76–85, 2013. 2

[13] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, SIGMOD '11, page 193–204, New York, NY, USA, 2011. Association for Computing Machinery. 5

[14] K. Kushlev and E. W. Dunn. Checking email less frequently reduces stress. *Computers in Human Behavior*, 43:220–228, 2015. 1

[15] K. Kushlev and E. W. Dunn. Stop checking email so often, Jan 2015. 1

[16] B. N. Levine, M. K. Reiter, C. Wang, and M. Wright. Timing attacks in low-latency mix systems. In A. Juels, editor, *Financial Cryptography*, pages 251–265, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. 2

[17] G. Mark, S. T. Iqbal, M. Czerwinski, P. Johns, A. Sano, and Y. Lutchyn. Email duration, batching and self-interruption: Patterns of email use on productivity and stress. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 1717–1728, 2016. 1

[18] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage. A fistful of bitcoins: Characterizing payments among men with no names. *Commun. ACM*, 59(4):86–93, mar 2016. 1

[19] K. Moore. How to improve productivity with time batching. Monday blog, Nov 2021. Accessed on April 25, 2022. 1

[20] M. Murphy. If you haven't tried time batching, you'll be shocked at how quickly it improves your productivity and happiness. Forbes Magazine, 2021. Accessed on April 25, 2022. 1

[21] W. T. of Service. Sockpuppetry - alternative accounts. Wikipedia TOS, 2022. Accessed on April 28,2022. 1

[22] N. B. Shah. An overview of challenges, experiments, and computational solutions in peer review. Communications of the ACM (to appear). Preprint available at http://bit.ly/PeerReviewOverview, June 2022. 2

[23] V. Shmatikov and M.-H. Wang. Timing analysis in low-latency mix networks: Attacks and defenses. In D. Gollmann, J. Meier, and A. Sabelfeld, editors, *Computer Security – ESORICS 2006*, pages 18–33, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. 2

[24] B. Wiki. Address reuse. https://en.bitcoin.it/wiki/Address_reuse, 2021. Accessed on April 21, 2021. 1