Publishing Asynchronous Event Times with Pufferfish Privacy

Publishing Asynchronous Event Times with Pufferfish Privacy

Jiaxin Ding*
Shanghai Jiao Tong University

Abhirup Ghosh*
University of Cambridge

Rik Sarkar
University of Edinburgh

Jie Gao
Rutgers University

***Abstract*—Publishing data from IoT devices raises concerns of leaking sensitive information. In this paper we consider the scenario of publishing data on events with timestamps. We formulate three privacy issues, namely, whether one can tell if an event happened or not; whether one can nail down the timestamp of an event within a given time interval; and whether one can infer the relative order of any two nearby events. We show that perturbation of event timestamps or adding fake events following carefully chosen distributions can address these privacy concerns. We present a rigorous study of privately publishing discrete event timestamps with privacy guarantees under the Pufferfish privacy framework. We also conduct extensive experiments to evaluate utility of the modified time series with real world location check-in and app usage data. Our mechanisms preserve the statistical utility of event data which are suitable for aggregate queries.**

I. Introduction

With the proliferation of Internet of Things, huge volume of data is generated by densely deployed sensing devices, from fitness trackers for health care, appliances for smart homes, to traffic sensors for urban transportation, etc. [5], [12]. In the process of collecting and sharing the sensing data, protecting user sensitive information has become an important research topic. In this paper, we consider the issue of privacy in publishing timestamps of sensing events.

In the typical setting of sensing, timestamped events are sensed either by users' personal devices such as mobile phones, or by infrastructures such as RFID sensors, etc. There are two possible ways to sense and publish events (shown in Figure 1) depending on whether user information is associated. First, events are sensed and labeled with the users' identification information. Users may leave event traces directly with their own devices, e.g., tweeting check-in events. Connection events of mobile devices can be sensed and recorded by cellular towers, while users' identification information is obtained by the phone numbers of the connected devices. Second, unlabeled events are recorded and published by infrastructures, where association between users and events are unknown or hidden. For example, a Wi-Fi access point can detect connection events of mobile devices, while user IDs can be hidden by randomized MAC addresses.

In these scenarios, there are different privacy risks. In a labeled sequence of events, the events of the same user can reveal the user's private information, such as habits or daily routines. For example, many of us check the weather app before going out. In the real dataset of LiveLab [27], we observe daily periodicity for users accessing the weather app. We analyze the time series of the users opening the weather app by Fourier transformation, where the highest spectral energy corresponding to the most dominant periodicity pattern of users. In Figure 2(a) we demonstrate an example of periodicity analysis of one user, where there is a clear energy peak at 24 hours, meaning that the user has the habit of opening the weather app at the same time every day. Such information can be used to predict user behavior.

Besides, the order of the events produced by users can be used to infer sensitive information. For example, the order of the apps used can be correlated with user activities [29], e.g., opening a messaging app followed by a food review app could mean planning to dine out with friends. Moreover, suppose an adversary has accessed to the logs of website visits without knowing the user identity from compromised network routers observing that a service was accessed at exact times t_1, t_2 and t_3, and now gains information that a user requested a service shortly before each time. The adversary can then associate the user with the service indicating that the user is accessing that service. Such timing attacks can compromise the anonymity of mixing networks like Tor [20].

An unlabeled sequence can also leak privacy. For example, the adversary suspects that user u in Fig. $1(b)$ has checked in within a small time window. The presence of an event in this interval strongly suggests that the adversary's information is probably correct. Such accessory knowledge of the adversary can make privacy preserving even more challenging.

The objective in this paper is to protect users from such inferences. A perfect protection of user information can refrain from publishing data or publish irrelevant data - in either case, data becomes useless [17]. Thus our approach is statistical, where useful information is published, but the adversary cannot infer user activities with high confidence.

Our Contributions. A natural and simple way to sanitize event timestamps is to perturb the reported time by random noise. However, theoretically quantifying the privacy achieved by such a mechanism is highly non-trivial and is the main focus of this paper. Section II shows that existing differential privacy based algorithms [10], [7] operate in synchronous rounds, and over long duration of time, they accumulate excessive noise, which overwhelms the data. In distributed sensor databases, asynchronous operations are particularly

*Equal contribution. Jiaxin Ding is the corresponding author. Corresponding Email: jiaxinding@sjtu.edu.cn

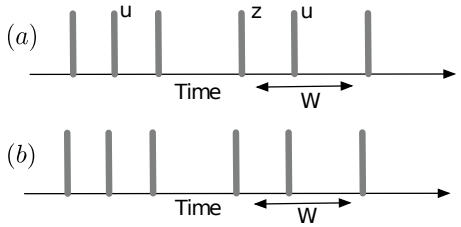
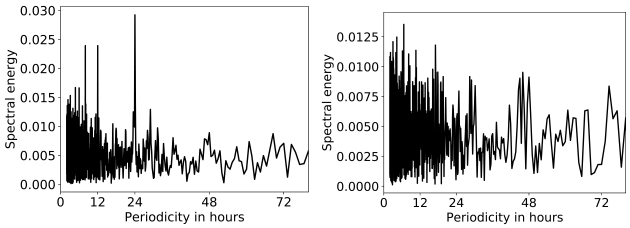


Fig. 1. Data publishing scenarios. (a) Events are labeled with users' identification information. (b) Only the presence of events are published, and events are unlabeled. Window W is adversary's hypothesis of when an event of interest may have taken place.



(a) (b)

Fig. 2. Periodicity analysis on when a user opens weather app. **(a)** Fourier periodogram of original time series shows clear periodicity of 24 hours - the user checks the app at the same time every day. **(b)** After events are sanitized using our Mechanism 2 in III-B with $\varepsilon = 1, \Delta = 24$hr, the periodic pattern vanishes.

desirable, since synchronization is non-trivial to maintain, and the frequency requirements can vary by systems.

We consider the asynchronous settings and formulate the privacy concerns formally, using the Pufferfish privacy, a revised differential privacy framework, considering and incorporating the fact that the adversary may have approximate prior knowledge, e.g., an adversary may have a suspicion (based on external information) that an event has taken place at a time window W. The overall model of operation and the adversarial knowledge is considered in Section II-D. The fundamental problems we address are:

1) For unlabeled events: protect the presence of an event within an interval.
2) For labeled events: protect an event time, so that the timing cannot be inferred beyond a given precision Δ and protect relative order of nearby events.

For task (1) of protecting the presence of an event, we add fake events at a suitable rate and remove a subset of real events. Similar fake events (or messages) have previously been used to enhance privacy in anonymity networks [23], but our analysis is the first to show statistical privacy guarantee for these methods. Adding random noise to the event timestamps with a limited scale can solve both the problems in (2). An example can be found in Figure 2(b), after applying a sanitization method described in this paper, the periodic behaviour in Figure 2(a) has vanished indicating that the precise time of checking the weather app is no longer so predictable. We derive rigorously the level of noise required to achieve ε-*Pufferfish* privacy for proposed mechanisms (Section III). In experiments (Section IV), we measure the effectiveness of the sanitized data and demonstrate that the utilities are well preserved with three real world datasets.

II. Statistical Privacy and Challenges in Publishing Event Timestamps

Differential privacy is the most popular statistical privacy framework and has been used to protect event timestamps as well. Thus, this section starts with an overview of statistical and differential privacy followed by the problem of applying such framework to the problems at hand. Readers familiar with the basics may want to skip to Subsection II-B.

A. Differential Privacy

Consider a dataset, D containing sensitive attributes. Even without direct access to D, an adversary may be able to infer the sensitive information through queries. The central issue in statistical privacy is to find balance between making such inference difficult and preserving utility in the aggregate. Statistical privacy mechanisms propose randomized methods to make query answers almost equiprobable for different values of sensitive attributes and quantify privacy by the probabilistic similarity of query answers. For example, Differential Privacy protects presence of an item in D as follows.

Definition 1 (Differential Privacy)**.** *Suppose D' is a dataset that is different from D in the presence of a single element. Then randomized algorithm $\mathcal{A}$ is called ε-differentially private, if for all such pairs (D, D') and all $w \in \text{Range}(\mathcal{A})$, the algorithm $\mathcal{A}$ satisfies: $\frac{P(\mathcal{A}(D)=w)}{P(\mathcal{A}(D')=w)} \leq e^{\varepsilon}$.*

Differentially private algorithms depend on sensitivity of a query function f, defined as the maximum possible change to $f(D)$ due to presence or absence of any single element, formally denoted as $\Delta_f = \max |f(D) - f(D')|$. For a counting query, $\Delta_f = 1$, since a single element can change the count only by 1. The noise we need to add to obscure the presence of an element is proportional to the sensitivity.

A ε-differentially private *Laplace mechanism* [7], [6] answers a query by $\mathcal{A}(D) = f(D) + \xi$ where $\xi \sim \text{Lap}\,(b)$ is a random number drawn from the Laplace distribution of mean $\mu = 0$ and variance $2b^2 = 2(\frac{\Delta_f}{\varepsilon})^2$. The probability density at x as $\text{Lap}\,(x; \mu, b) = \frac{1}{2b} e^{-\frac{|x-\mu|}{b}}$.

B. Publishing events using local differential privacy

An extension of differential privacy, called local differential privacy [2] supports publishing data by adding noise to the data itself instead of the results of a query. For publishing events such framework needs synchronous rounds – at each round r, the count $c(r)$ of events is published. A particular variant of this problem was considered by [4] where in each round there is either zero or one event. A basic method described there is to publish the count with a Laplace noise: $c(r) + \xi$, where $\xi \sim \text{Lap}\left(\frac{1}{\varepsilon}\right)$. In any individual round, the presence or absence of an event is obscured (sensitivity is 1) by the noise.

The difficulty of this method is that over time, the noise builds up. Over T rounds, T random samples of ξ are added,

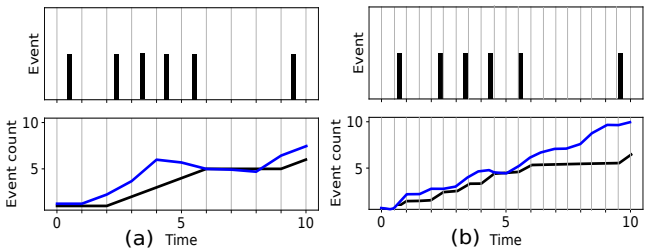


Fig. 3. The performance of synchronized privacy method [4] is sensitive to the sampling rate of discrete time rounds. The method adds $Lap(1)$ noise to the count at each time slot. The error introduced due to sanitization is the disagreement between the cumulative event counts for original (black) and sanitized (blue). **(a)** With low sampling frequency, it adds less noise (less number of $Lap(1)$) and preserves high utility. **(b)** High sampling frequency accumulates more noise from the empty rounds, making the utility worse. In both cases the top panel shows the events and the time slots.

and the number of events can deviate rapidly from the true count – see Figure 3 as an example. The problem becomes more severe when events are sparse and there are many rounds without events. In Figure 3 there are same number of events, while the series in Figure 3(b) is sampled more frequently and thus there are more empty rounds. The noise added in the empty rounds adds to inaccuracy and dominates the true total count eventually in Figure 3(b).

In practice, the event density and required temporal resolution can be unknown or vary by applications – e.g. human behavior in seconds, versus network measurements in milliseconds. Therefore, these mechanisms are unsuitable in real world applications. Thus, we would like to avoid rigid synchronous rounds, and use an asynchronous setup, where the published timestamp of an event can be a real number. For this purpose, we use a more versatile framework called Pufferfish privacy which we introduce in the following section.

C. Pufferfish Privacy

The Pufferfish Privacy framework [18] allows us to define any set of facts to be regarded as a secret, and protect them probabilistically. To define the Pufferfish privacy, we first introduce the following sets.

- the set of secrets $\mathcal{S}$ to be protected within dataset $\mathcal{X}$,
- how the secrets shall be protected against each other: by specifying discriminative pairs $\mathcal{Q}$ of secrets,
- the prior knowledge of the adversary, written as a set Θ of prior distributions, essentially representing how the data is generated.

A discriminative pair can be written as $(s_i, s_j) \in \mathcal{Q}$, where s_i and s_j are elements of $\mathcal{S}$, and represents two facts that we do not want the adversary to be able to distinguish.

A randomized mechanism M satisfies ε-*Pufferfish* privacy if for $s_i, s_j \in \mathcal{S}$, $(s_i, s_j) \in \mathcal{Q}$, $\theta \in \Theta$ and $w \in Range(M)$ the following holds:

$$e^{-\varepsilon} \leq \frac{P(M(\mathcal{X}) = w | s_i, \theta)}{P(M(\mathcal{X}) = w | s_j, \theta)} \leq e^{\varepsilon} \tag{1}$$

assuming $P(s_i|\theta) \neq 0$ and $P(s_j|\theta) \neq 0$. Similar to differential privacy, a smaller ε implies greater privacy.

It implies that the truth of s_i versus s_j does not make a significant difference to the probability of observing any particular output w. Therefore, when w is in fact observed, the adversary does not gain any significant certainty whether s_i or s_j is true, compared to what he already knew based on prior knowledge θ. As we will see in the next section, the generic nature of secrets and discriminative pairs lets us define privacy for properties like event ordering and event times.

The prior Θ represents the underlying distributions generating the data, for example, the vehicle density at a crossing at night, morning or rush hour, or the distribution of certain apps being used on a mobile phone. In the Pufferfish framework, following usual conventions of privacy techniques, we assume that any prior knowledge such as the generating distribution is public and known to the adversary, but the precise times of actual events are not known.

D. Model

We are interested in publishing data continuously. Events can be assumed to be published in batches, for example, once in an hour or a day. The sensors send their data to an aggregator service such as a cloud server, which then publishes or uses the data. Labeled events are those that carry user IDs or other attributes that can potentially uniquely identify the events. Unlabeled events are those that carry no such attributes or carry same attributes making them indistinguishable.

All algorithms in this paper can be applied locally, i.e., the device detecting the events can sanitize or defer the publication of the event prior to sending to the aggregator. This approach has the advantage that the data transmitted is guaranteed to be private, since statistical sanitizations are known to be immune to post processing [10]. Once the data has been modified by the mechanism, it carries the privacy guarantee irrespective of how the data is used. This is useful in sensor data that may be shared with various public and private organizations.

Utility. To ensure that the sanitized data is useful, we measure its utility in terms of accuracy of range queries. Given a time interval T, the range query asks for the number of events recorded inside T. The motivation for using range queries as a test of utility is that it is akin to preserving the large scale probability distribution that generates the data. If the count of events in different ranges is preserved, it implies that the statistical properties of data are preserved. The mechanisms here are designed with range queries in mind as, in general it is impossible to preserve utility of arbitrary queries [17].

E. Adversary Model

We assume that the adversary can observe the published events. The adversary may or may not be able to compromise the communication channel or aggregator – this is irrelevant to us, since we aim to create algorithms with local privacy. We do, however, assume that the adversary cannot compromise the sensors themselves to observe events as they happen. As is standard in security and privacy, we assume that the privacy algorithms and parameters are known to the adversary. However, the random choices – such as the random numbers

generated during an execution are not known. Under these capabilities of the adversary, we consider the data protection problems stated in Section I.

III. Privacy mechanisms

In this section, we discuss how to achieve statistical privacy of publishing an event timestamp for the problems discussed in Section I. In some cases the detailed technical proofs are replaced by their sketches in the interest of space.

A. Problem 1: Protecting existence of unlabelled events

Given an unlabeled set of event timestamps, here the adversary wants to confirm the existence of an event in a Δ sized interval I. For example, the adversary has the question: *I think someone checked in at the venue between 12:00 and 12:05 (I), is this fact confirmed in the data?* In the Pufferfish model, the secret can be defined as $s := (\exists E : t \in I)$ – representing the fact that some event E has occurred in the small neighborhood I of size Δ. The discriminative pair is $(s, \neg s)$, where $\neg s$ means that no event has taken place in I. We model an unlabeled event sequence as a non-homogeneous Poisson process – an event sequence with varying rates.

Definition 2 (Non-homogeneous Poisson Process)**.** *In a non-homogeneous Poisson process defined over an interval A with intensity function $\lambda : A \to \mathbb{R}_{\geq 0}$, the number of events $X(B)$ in any subinterval B of the domain follows a Poisson distribution: $P(X(B) = K) = e^{-\Lambda(B)} \cdot \frac{\Lambda(B)^K}{K!}$, where $\Lambda(B) = \int_B \lambda(t)\, dt$ is called the mean rate in B.*

The intensity function $\lambda(t)$ can be estimated using density estimation methods. By using a function $\lambda(t)$ that may vary in time, the Poisson process can be used to model almost any desired distribution of events; see Figure 4 for an example.

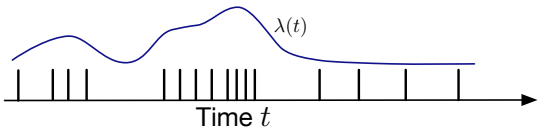


Fig. 4. Event sequence modeled as non-homogeneous Poisson process. Larger rate $\lambda(t)$ corresponds to greater density of events.

In a Poisson process, roughly speaking, the expected interval between events is $1/\lambda(t)$. If $\Delta \gg 1/\lambda(t)$, then there are likely to be multiple events in I, and presence of an event is not informative for the adversary. However, when $\Delta \leq 1/\lambda(t)$, the adversary has fairly accurate information of the event's possible time compared to the local density of events, and other events are unlikely in the interval. Thus, a strong adversary with small Δ can *isolate* the event – and have reasonable confidence that any event in the interval is the one of interest. Therefore, we assume that the lengths of the intervals of the adversary's interests are reasonable with respective to the event rates, with bounded numbers of events in expectation. We consider the interval I whose event density is bounded by a lower bound c and an upper bound c', i.e., $\lambda = \int_{t\in I} \lambda(t)dt \in [c, c']$. Note that a sensor can estimate the $\lambda(t)$ by counting the number of events at the latest unit interval and we assume this to be global knowledge.

Mechanism 1**:** Given a real event sequence, remove each event with probability $p = \frac{1}{c'} \ln(e^{-\varepsilon}(e^{c'} - 1) + 1)$ to obtain a sequence S_R. Generate a fake event sequence, S_F with Poisson event rate $\Lambda(t) = \frac{\lambda(t)}{c} \ln(1 + e^{-\varepsilon})$. The published events are time ordered union of S_R and S_F.

The greater the rate of this Poisson process is, the greater the possibility becomes, that one or more fake events will be present in the interval I, and thus the adversary cannot be certain of the presence of a true event (See Figure 5). Further, stochastic deletion of real events stops from inferring whether an event exists after observing a published event or not within a very small interval. However, a legitimate user knowing the exact time of a real event can find it with probability $1 - p$, which is useful in provenance applications [26]. Also note that if the deletion probability, $p \to 1$, stronger privacy ($\varepsilon \to 0$) is guaranteed, but, the utility gets significantly reduced. The deletion probability reflects the tradeoff between the privacy protection and utility of the event sequences.

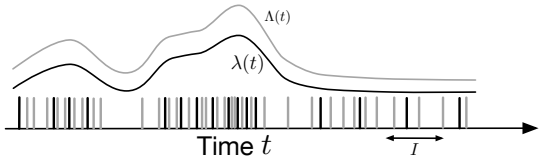


Fig. 5. A set of fake events (shown in grey) added at rate $\Lambda(t)$. Interval I now contains fake events, making the adversary uncertain about presence of true events.

To infer the existence of an event in interval I, the adversary's prior knowledge, θ_λ contains the real event intensity $\lambda = \int_{t\in I} \lambda(t)dt \in [c, c']$, and correspondingly fake event rate $\Lambda = \int_{t\in I} \Lambda(t)dt$, $\Lambda \geq \frac{\lambda}{c} \ln(1 + e^{-\varepsilon}) \geq \ln(1 + e^{-\varepsilon})$.

Theorem 1. *With fake event rate $\Lambda(t) \geq \frac{\lambda(t)}{c} \ln(1 + e^{-\varepsilon})$ and removing probability $p = \frac{1}{c'} \ln(e^{-\varepsilon}(e^{c'} - 1) + 1)$, the Mechanism 1 provides ε-Pufferfish privacy for existence of events in any interval I, such that $\lambda = \int_{t\in I} \lambda(t)dt \in [c, c']$.*

Proof. Let η be the situation that at least an event is seen in a interval I in the published stream. The probability that no fake event is generated in the interval I is $e^{-\Lambda}$, the probability that no real events is published is $e^{-\lambda(1-p)}$, and the probability that there is no real event in I is $e^{-\lambda}$. Therefore, the probability that no real event is published when there is at least one real event is $e^{-\lambda(1-p)} - e^{-\lambda}$. Then

$$P(\neg\eta|s) = \frac{(e^{-\lambda(1-p)} - e^{-\lambda})e^{-\Lambda}}{1 - e^{-\lambda}} = \frac{(e^{\lambda p} - 1)e^{-\Lambda}}{e^{\lambda} - 1}$$

and $P(\neg\eta|\neg s) = e^{-\Lambda}$. Since $f(x) = \frac{1}{x} \ln(e^{-\varepsilon}(e^x - 1) + 1)$ is a monotonic increasing function when $x \in (0, c']$, $p \geq \frac{1}{\lambda} \ln(e^{-\varepsilon}(e^{\lambda} - 1) + 1)$. We have

$$\frac{P(\neg\eta|s)}{P(\neg\eta|\neg s)} = \frac{(e^{\lambda p} - 1)}{e^{\lambda} - 1} \geq \frac{(e^{\frac{\lambda}{\lambda} \ln(e^{-\varepsilon}(e^{\lambda} - 1) + 1)} - 1)}{e^{\lambda} - 1} \geq e^{-\varepsilon}.$$

Moreover, $\frac{P(\neg\eta|s)}{P(\neg\eta|\neg s)} \leq 1 \leq e^{\varepsilon}$. Now we analyze the case when at least one event is published. Since $\Lambda \geq \ln(1+e^{-\varepsilon})$, $e^{-\Lambda} \leq \frac{e^{\varepsilon}-1}{e^{\varepsilon}-e^{-\varepsilon}}$. Combining with $\frac{(e^{\lambda_p}-1)}{e^{\lambda}-1} \geq e^{-\varepsilon}$, we have

$$\frac{P(\eta|s)}{P(\eta|\neg s)} = \frac{1-\frac{(e^{\lambda_p}-1)}{e^{\lambda}-1}e^{-\Lambda}}{1-e^{-\Lambda}} \leq \frac{1-e^{-\varepsilon}e^{-\Lambda}}{1-e^{-\Lambda}} = \frac{e^{\varepsilon}-1}{1-e^{-\varepsilon}} \leq e^{\varepsilon}.$$

Besides, $\frac{(e^{\lambda_p}-1)}{e^{\lambda}-1} \geq 1 \geq e^{-\varepsilon}$. We have the theorem. □

Utility analysis: range counting of events. The addition of fake events and probabilistic removal of real events changes the number of events in an interval. However, since the events are added from a known Poisson model and the probability of removing a real event is known, it is possible to simply deduct the expected number of fake events and adjust to the number of real events retained in expectation in the interval to obtain a good estimation. Thus, given a query range $[x, y]$, the algorithm returns event count as:

1) Compute n: The total number of events in $[x, y]$
2) Estimate the number of fake events: $\int_x^y \Lambda(t)\,dt$
3) Return $\frac{n-\int_x^y \Lambda(t)\,dt}{1-p}$

Here, the error in range count is introduced in two steps. While the error in step 3 follows standard analysis of binomial distribution with success probability $\frac{1}{1-p}$, the error in step 2 is more subtle and is analysed below. The following Lemma from [13] is useful to us:

Lemma 1. *If a random variable Y follows a Poisson distribution with mean Λ_Y, $P(|Y-\Lambda_Y| \geq \alpha) \leq 2\exp\left(-\frac{\alpha^2}{2(\Lambda_Y+\alpha)}\right)$.*

In our case, this leads to the following theorem:

Theorem 2. *If $\Lambda(t) = \frac{\lambda(t)}{c}\ln(1+e^{-\varepsilon})$, in step 2, we get an estimation of the count of real events not removed by the mechanism in the interval $[x, y]$ with error bounded by $O\left(\sqrt{\Lambda \log\frac{1}{\beta}}\right)$, with probability $1-\beta$, where $\Lambda = \int_x^y \Lambda(t)dt$.*

Proof. Given a interval $[x, y]$, denote the count of real events as n^*. Denote $\mathcal{A}(x, y)$ as the count of events after applying our mechanism. $\mathcal{A}(x, y) - n^* = n - n^* - \Lambda$. Notice that $n - n^*$ is the count of fake events, following the Poisson with mean Λ. Applying the Lemma 1, for any β, we set $\alpha = \log\frac{2}{\beta} + \sqrt{\log^2\frac{2}{\beta} + 2\Lambda\log\frac{2}{\beta}}$, and we have

$$P\left(|\mathcal{A}(x, y) - n^*| \leq \log\frac{2}{\beta} + \sqrt{\log^2\frac{2}{\beta} + 2\Lambda\log\frac{2}{\beta}}\right) \geq 1-\beta.$$

Rearranging, we get the theorem. □

From the above theorem, we can see that when the privacy protection requirement is stronger (ε decreases), the rate of fake events increases to achieve the level of protection, which results in the decrease of the utility of the range counting of events. This mechanism has the nice property that if each distributed device j applies the mechanism with respect to its local Poisson rate λ_j, then the aggregated global event stream has the same privacy guarantee with respect to the global Poisson event rate. This result follows from the Superposition Theorem [19], [14] – that the union of Poisson processes is still a Poisson process.

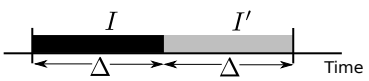


Fig. 6. Schematic for the setting in Problem 2. The adversary knows that an event occurs either in I or in I' and wants to confirm if it has occurred in a I. Both I and I' have size Δ.

B. Problem 2: Protecting Event Times of Labeled Events

In this problem, we consider the events to be labeled and the adversary knows the approximate time of an event and wants to find a more precise time. In most cases, the adversary does not need to know the precise timestamp, and just need to narrow the time down to a small interval around the true time. For example, for human activities like checking into a venue, accuracy within a few minutes of the true time can be considered accurate enough. Thus, for an event E occurred at time t, we want to prevent an adversary from inferring whether E had happened within a Δ sized interval I (secret s) versus within a neighboring interval I' (secret s') as shown in Figure 6. The objective is to obtain ε-*Pufferfish* privacy for the discriminative pair (s, s').

Note that the Mechanism 1 works on unlabeled event sequence and does not work here as it is not trivial to construct the labels for the fake events. Further, this is harder than Problem 1 as we intend to hide the real timestamp of an event, instead of its presence only. Also we relax the assumption on the distribution of the events.

Mechanism 2: $M(E) = t + \xi$, where $\xi \sim \text{Lap}(b)$. That is, a Laplace noise similar to the Laplace mechanism as described in Section II.

Prior distribution. For an event E we consider the adversary's prior distribution, θ as uniform distribution over $W = I \cup I'$, where I and I' are neighboring time interval with size Δ. In practice, an adversary often have a bounded guess of when an event has occurred, for example, the adversary knows that the target person went to a restaurant for lunch in a 2 hour window and wants to find out if the check-in happened in the first hour or in the next. Further the uniform distribution means that the adversary does not know anything more about the timing of the event. We would like to guarantee that the adversary can not discriminate whether E happens in the interval I and I'.

While the Mechanism 2 can put a sanitized event, $M(E)$ outside W with non-zero probability, the actual event time, t will always be inside W. Below we analyze the privacy guarantees for a uniform prior, followed by a utility analysis.

Theorem 3. *Given that the adversary's prior (θ) is a uniform distribution over an interval $W = I \cup I'$, Mechanism 2 with noise distribution $\text{Lap}\left(\frac{2\Delta}{\varepsilon}\right)$ guarantees to protect ε-Pufferfish privacy of whether an event E has occurred in a Δ sized interval I or in a neighboring interval I' of the same size.*

Proof sketch. The proof considers three cases (i) $M(E) \in I$ and (ii) $M(E) \in I'$, and $iii)$ $M(E) \in W'$ where W' denotes

the region outside $I \cup I'$. For each case we calculate the ratio of the probabilities of the output given the secret values and combining them we get the theorem.

Utility Analysis. The utility of the mechanism is considered with respect to range query over an interval Q. For a real event E (at time t) and its sanitized version $M(E)$, one of the following would happen, i) both are inside Q (true positive case), ii) $t \in Q, M(E) \notin Q$ (false negative), iii) $M(E) \in Q, t \notin Q$ (false positive), and iv) both outside Q (true negative). We denote that the probability of an event being true positive is $P_{TP} = P(M(E) \in Q | t \in Q)$, false negative is $P_{FN} = P(M(E) \notin Q | t \in Q)$, false positive is $P_{FP} = P(M(E) \in Q | t \notin Q)$, and true negative is $P_{TN} = P(M(E) \notin Q | t \notin Q)$.

Theorem 4. *If an event E is sanitized using the mechanism 2 to follow ε-Pufferfish according to Theorem 3, then it has the following utility results with respect to range query on Q of size $R\Delta$: i) $\frac{1}{2}(1-e^{-R\varepsilon/2}) \leq P_{TP} \leq 1-e^{-R\varepsilon/4}$, ii) $e^{-\frac{R\varepsilon}{4}} \leq P_{FN} \leq \frac{1}{2}(1+e^{-R\varepsilon/2})$, iii) $0 \leq P_{FP} \leq \frac{1}{2}(1-e^{-R\varepsilon/2})$, and iv) $\frac{1}{2}(1+e^{-R\varepsilon/2}) \leq P_{TN} \leq 1$.*

Proof sketch. The upper bound for the true positive probability occurs when the event E is centered at Q. Then the probability that $M(E) \in Q$ is the Laplace probability mass in Q. W.l.o.g. consider $Q = [-R\Delta/2, R\Delta/2]$. Then $P(M(E) \in Q | t = R/2) = \int_{-R\Delta/2}^{R\Delta/2} \text{Lap}\,(x; 0, 2\Delta/\varepsilon)\, dx = 1-e^{-R\varepsilon/2}$. The lower bound occurs when t is at the boundary of Q. Similar consideration produces the bounds for other probabilities. Note that t can be arbitrarily far away from Q, and this produces the lower bound for P_{FP} and upper bound for P_{TN}.

From the above analysis, we observe that when the privacy guarantee increases (ε decreases), the probability of counting from E correctly decreases. While the Theorem 3 and 4 focus on a single event, these results remain the valid for multi-event scenario by applying the same claim individually to each event.

C. Problem 3: Protecting Orders of Labeled Events

In this problem, we consider protecting the relative order of two nearby labeled events. We assume that the adversary knows (θ_o) that the events e_i and e_j occurred within a Δ sized interval, i.e., $|t_i - t_j| \leq \Delta$ and wants to find out the order of the events. Thus the discriminative pair of secrets are $s_{i<j}$ and $\neg s_{i<j}$ denoting $t_i < t_j$ and $t_j < t_i$ respectively. Below we show that with suitable noise, Mechanism 2 achieves privacy guarantees for event ordering.

Theorem 5. *Mechanism 2 with $\text{Lap}\left(\frac{2\Delta}{\varepsilon}\right)$ preserves ε-Pufferfish privacy for event ordering for event-pairs within Δ time interval.*

Proof sketch. According to the Mechanism 2, both $M(e_i)$ and $M(e_j)$ are Laplace random variables distributed with mean values at t_i and t_j. Observe that the cumulative distribution of the random variable $Z = (M(e_i) - M(e_j))$ at 0 produces the probability of $M(e_i) < M(e_j)$. To prove the above result, we need to consider two cases when $M(e_i) \leq M(e_j)$, i.e., $Z \leq 0$ and $M(e_i) \geq M(e_j)$, i.e., $Z \geq 0$. We skip the calculations here for the interest of space.

This result implies that our mechanism provides a guarantee against inference of precise ordering of nearby events. The utility analysis for this problem is the same as Problem 2, where the same mechanism is applied. Note that events that are far apart in timings are not covered under this model, since any modified assignment of timestamps that modifies the ordering of distant events will randomize the the data completely.

IV. Experiments

This section evaluates the utility for both proposed sanitization mechanisms on three real world datasets.

Datasets. The datasets we use include location check-in data (i) and mobile phone usage data $(ii), (iii)$ to reflect different patterns of event distributions and user behaviors. (i) Check-in dataset: $12,372$ timestamped check-ins at Akihabara train station in Tokyo by $1k$ users from Foursquare [31]. The events are aggregated in a single day to increase the density of the events. (ii) App-usage dataset: events occur in the app-usage dataset [27] when a user opens one of his 10 most frequent apps, e.g., SMS, mail, Facebook, etc. (iii) Phone-unlock dataset: events occur when a user unlocks the phone. The first two datasets contain data for about an year, and the third has data for a week. The datasets we consider have diverse densities (Figure 7(a)) with average inter-event duration, from seconds for dataset (i) to minutes for dataset $(ii), (iii)$ to test our methods in varied scenarios.

Evaluating utility. The utility of the methods is measured in line with our theoretical analyses. The utility of the Mechanism 1 is measured using range counts. Given the counts at a query range in the original and sanitized stream are n and $\tilde{n}$ respectively, the relative error measured as $\xi = \frac{|n-\tilde{n}|}{n}$ is used for the utility. The utility for Mechanism 2 is measured with the true positive, false negative, false positive, and true negative probabilities in the theory, and in the experiment, we use F_1 score to measure the four. F_1 score is the harmonic mean of the precision and recall, where precision is the ratio of true positive over true positive plus false positive, and recall is the ratio of true positive over true positive plus false negative. The value of ε is application specific, and therefore we show how the utility varies for different privacy parameters.

A. Evaluating Mechanism 1 for unlabeled events

The event rate $\lambda(t)$ is estimated by the average rates of the latest 100 events and we produce sanitizied events using Mechanism 1. The range counts reported here are for each such intervals. We consider the resolution of the event times in seconds. c and c' are fixed to 1 and 2 respectively. Figure 8 evaluate the mechanism 1 and demonstrate that the relative error of Mechanism 1 is bounded and the error increases with the decrease of ε (stronger privacy guarantee).

Comparison with existing methods. The p-sum mechanism described in [4] and briefed in Section II-B is applied to check-in event series with different resolutions prepared by suitable

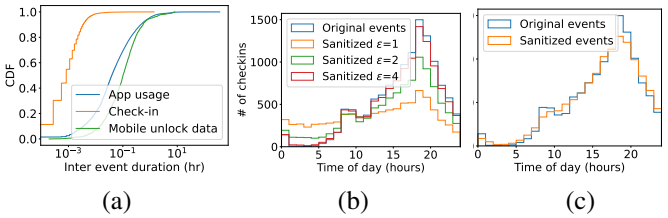


Fig. 7. **(a)** Inter-event duration distribution differs for the considered dataset showing diversity in the settings. Temporal distribution of original and sanitized events after applying Mechanism 1 (**b**) and Mechanism 2 (**c**) using $\varepsilon = 1$ and $\Delta = 1$hr. Here we use check-in data. Note that Figure 8 considers the range counting method described in Section III-A and thus is more accurate than **(b)**.

binning (Figure 9 (a)). The error for the p-sum mechanism increases with finer resolution, from the resolution of hours to seconds. With the same level of privacy guarantee, our Mechanism 1 produces events in second resolution and has more accurate range count query results compared to the best performance of p-sum method with hour resolution with the least noise added.

Processing time. Both Mechanism 1 and 2 achieve constant computation per event, the total computation cost is linear with respective to the size of dataset as shown in Figure 9 (b), and therefore the mechanisms scale well for large datasets.

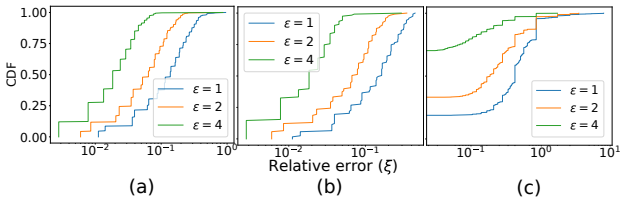


Fig. 8. Evaluating Mechanism 1. **(a, b, c)** vary ε for app-usage dataset, check-in dataset, and phone-unlock dataset respectively. The errors remain low for all values of ε. Naturally error increases with decreasing ε.

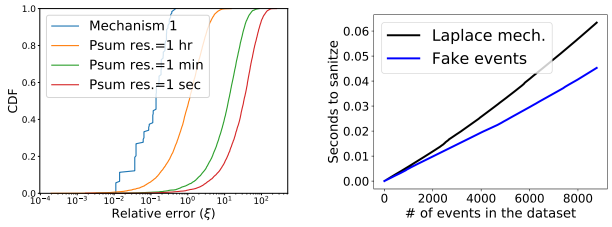


Fig. 9. **(a)** Range count queries are more accurate for our Mechanism 1 (resolution in seconds) than p-sum mechanism with even hour resolution. We take $\varepsilon = 1$ for both cases. **(b)** Both the mechanisms are efficient for large dataset. Sampling from Laplace distribution is slightly slower than sampling from Poisson and Uniform distribution, thus the slight difference.

B. Evaluating Mechanism 2 for labeled events

Figure 7(c) demonstrates that the distribution of events is well preserved by the Mechanism 2, which guarantees the utility of events after sanitization. Figure 10 evaluates the privacy-utility trade-off by varying Δ and ε of Mechanism 2. The smaller Δ and larger ε require less privacy guarantee, less noise is added, and therefore more original information is preserved. The utility is measured using the F_1 score, which

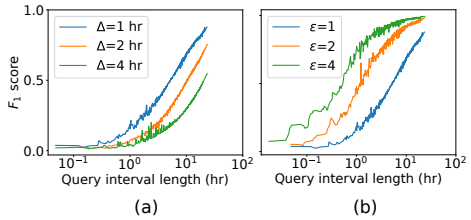


Fig. 10. Evaluating the utility for Mechanism 2 with 2000 randomly sampled intervals in the check-in dataset. While varying Δ, ε is set to 1 and while varying ε, Δ is set to 1. The utility (F_1 score) increases with larger ε and smaller Δ.

is defined as the harmonic average of the precision and recall. A larger F_1 score corresponds to higher utility.

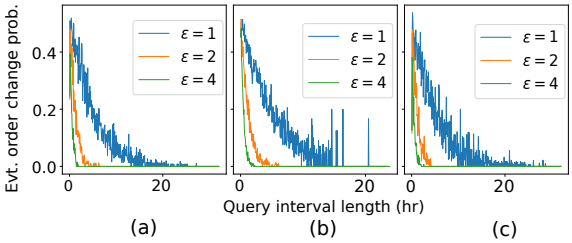


Fig. 11. Evaluating event ordering for Mechanism 2. Δ is set to 1 hour for all datasets. (a), (b), and (c) are showing results for app-usage, check-in, and phone-unlock datasets respectively.

We evaluate the probability that the order of randomly chosen event pairs with different intervals is changed after sanitization with mechanism 2, demonstrated in Figure 11. With a larger ε, less noise is added to the event time and the event orders are better preserved, where the order change are mostly limited to the events within smaller time intervals.

V. Related Work

Releasing continual counts for temporal event streams with differential privacy guarantees have been studied in [4], [9], [8], [30], obtaining similar results as we discussed in Section II-B. More data-dependent works on continual aggregate statistics release of stream data can be found in [5], [16]. All these works assume a suitable sampling frequency for obtaining the event stream in the first place, and therefore, they are not suitable for asynchronous event publication.

[24] achieves ε-Differential privacy by applying sanitization in the frequency domain to the k Fourier coefficients with highest spectral energy. However, this is not applicable in our setting as it releases data only once. A data-dependent heuristic method is proposed in [11] to release real time aggregate traffic statistics under differential privacy using sampling and filtering. [12], [1] have proposed to generate fake time series following the same distribution as the original data. The objective is to protect against re-identification of a time series, but do not hide individual events in series. [28] has discussed privacy preservation in presence of correlation in the data, e.g., activities of a user over time using Pufferfish privacy [18] and models the data generation process as a Markov chain. [22] has designed a time-series data trading system with Pufferfish privacy under temporal correlations. However, it

does not trivially translate to privacy of individual event timings. Blowfish [15] is another instantiation of the Pufferfish framework for use on continuous variables, which however does not apply to discrete abstractions like event ordering. Differential privacy is also being explored for correlation time series data [3].

Various methods have been proposed to protect user level and activity level privacy. Differential privacy frameworks for protecting specific inference from the data are investigated [25] using dynamic Bayesian networks to capture the adversary's knowledge. The proposed method first infers latent states and protects them. Continual adaptive release of histogram statistics is considered [21] to protect users to be re-identified. Unlike these works, the goal of the current paper is to protect individual event timings.

VI. Conclusion

In this paper, we proposed multiple privacy preserving mechanisms for protecting individual event timings and hiding occurrence of an event. The proposed framework uses Pufferfish privacy – a generalization of differential privacy – and supports asynchronous event timings. The mechanisms add considerably less noise than existing algorithms [4] to achieve similar privacy guarantees. In sensor data streams of multiple variables, privacy of complex relations between variables may be of interest. Rigorous privacy definitions of such relations through Pufferfish style privacy approach is a promising research area.

Acknowledgment

J. Ding would like to acknowledge supports from Shanghai Sailing Program 20YF1421300, Natural Science Foundation of Shanghai No. 22ZR1429100. A. Ghosh would like to acknowledge supports from Welcome Trust (Grant No. 213939). J. Gao would like to acknowledge supports from NSF CCF-2118953 and OAC-1939459.

References

[1] H. Ahmadi, N. T. Pham, R. K. Ganti, T. F. Abdelzaher, S. Nath, and J. Han. Privacy-aware regression modeling of participatory sensing data. In *SenSys*, 2010.

[2] B. Bebensee. Local differential privacy: a tutorial. *arXiv preprint arXiv:1907.11908*, 2019.

[3] Y. Cao, M. Yoshikawa, Y. Xiao, and L. Xiong. Quantifying differential privacy in continuous data release under temporal correlations. *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[4] T.-H. H. Chan, E. Shi, and D. Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26:1–26:24, Nov. 2011.

[5] Y. Chen, A. Machanavajjhala, M. Hay, and G. Miklau. Pegasus: Data-adaptive differentially private stream processing. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1375–1388. ACM, 2017.

[6] C. Dwork. Differential privacy. In *Proceedings of the 33rd International Conference on Automata, Languages and Programming - Volume Part II*, ICALP'06, pages 1–12. Springer-Verlag, 2006.

[7] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[8] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 715–724. ACM, 2010.

[9] C. Dwork, M. Naor, O. Reingold, and G. N. Rothblum. Pure differential privacy for rectangle queries via private partitions. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 735–751. Springer, 2015.

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

[11] L. Fan, L. Xiong, and V. Sunderam. Differentially private multi-dimensional time series release for traffic monitoring. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 33–48. Springer, 2013.

[12] R. K. Ganti, N. Pham, Y.-E. Tsai, and T. F. Abdelzaher. Poolview: stream privacy for grassroots participatory sensing. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 281–294. ACM, 2008.

[13] O. Goldreich. *Introduction to property testing*. Cambridge University Press, 2017.

[14] G. Grimmett and D. Stirzaker. *Probability and random processes*. Oxford university press, 2001.

[15] X. He, A. Machanavajjhala, and B. Ding. Blowfish privacy: tuning privacy-utility trade-offs using policies. In *SIGMOD Conference*, 2014.

[16] G. Kellaris, S. Papadopoulos, X. Xiao, and D. Papadias. Differentially private event sequences over infinite streams. *Proceedings of the VLDB Endowment*, 7(12):1155–1166, 2014.

[17] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 193–204, 2011.

[18] D. Kifer and A. Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Transactions on Database Systems (TODS)*, 39(1):3, 2014.

[19] J. F. C. Kingman. *Poisson processes*, volume 3. Clarendon Press, 1992.

[20] B. N. Levine, M. K. Reiter, C. Wang, and M. Wright. Timing attacks in low-latency mix systems. In *International Conference on Financial Cryptography*, pages 251–265. Springer, 2004.

[21] H. Li, L. Xiong, X. Jiang, and J. Liu. Differentially private histogram publication for dynamic datasets: an adaptive sampling approach. *Proceedings of the ACM International Conference on Information and Knowledge Management.*, 2015:1001–1010, 2015.

[22] C. Niu, Z. Zheng, S. Tang, X. Gao, and F. Wu. Making big money from small sensors: Trading time-series data under pufferfish privacy. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 568–576. IEEE, 2019.

[23] A. M. Piotrowska, J. Hayes, T. Elahi, S. Meiser, and G. Danezis. The loopix anonymity system. In *26th USENIX Security Symposium, USENIX Security*, pages 16–18, 2017.

[24] V. Rastogi and S. Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 735–746. ACM, 2010.

[25] N. Saleheen, S. Chakraborty, N. Ali, M. M. Rahman, S. M. Hossain, R. Bari, E. H. Buder, M. B. Srivastava, and S. Kumar. msieve: differential behavioral privacy in time series of mobile sensor data. *Proceedings of the ACM International Conference on Ubiquitous Computing (UbiComp)*, 2016:706–717, 2016.

[26] B. Schatz, G. Mohay, and A. Clark. A correlation method for establishing provenance of timestamps in digital evidence. *digital investigation*, 3:98–107, 2006.

[27] C. Shepard, A. Rahmati, C. Tossell, L. Zhong, and P. Kortum. Livelab: measuring wireless networks and smartphone users in the field. *ACM SIGMETRICS Performance Evaluation Review*, 38(3):15–20, 2011.

[28] S. Song, Y. Wang, and K. Chaudhuri. Pufferfish privacy mechanisms for correlated data. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1291–1306. ACM, 2017.

[29] Y. Tian, K. Zhou, M. Lalmas, and D. Pelleg. Identifying tasks from mobile app usage patterns. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2357–2366, 2020.

[30] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. *IEEE Transactions on knowledge and data engineering*, 23(8):1200–1214, 2011.

[31] D. Yang, B. Qu, J. Yang, and P. Cudre-Mauroux. Revisiting user mobility and social relationships in lbsns: a hypergraph embedding approach. In *The World Wide Web Conference*, pages 2147–2157, 2019.