Adaptive Buffer Resizing for Efficient Anonymization of Streaming Data with Minimal Information Loss

Aderonke Busayo Sakpere and Anne V. D. M. Kayem
Department of Computer Science, University of Cape Town, Cape Town, South Africa

Keywords: Data Anonymity, Streaming Data, Crime Reporting, Privacy Enhancing Model, k-Anonymity, Information Loss.

Abstract: Mobile crime reporting systems have emerged as an effective and efficient approach to crime data collection in developing countries. The collection of this data has raised the need to analyse or mine the data to deduce patterns that are helpful in addressing crime. Since data analytic expertises are limited in developing nations, outsourcing the data to a third-party service provider is a cost effective management strategy. However, crime data is inherently privacy sensitive and must be protected from "honest-but-curious" service providers. In order to speed up real time analysis of the data, streaming data can be used instead of static data. Streaming data anonymity schemes based on k-anonymity offer fast privacy preservation and query processing but are reliant on buffering schemes that incur high information loss rates on intermittent data streams. In this paper, we propose a scheme for adjusting the size of the buffer based on data arrival rates and use k-anonymity to enforce data privacy. Furthermore, in order to handle buffered records that are unanonymizable, we use a heuristic that works by either delaying the unanonymized record(s) to the next buffering cycle or incorporating the record(s) into a cluster of anonymized records with similar privacy constraints. The advantage of this approach to streaming-data anonymization is two-fold. First, we ensure privacy of the data through k-anonymization, and second, we ensure minimal information loss from the unanonymized records thereby, offering the opportunity for high query result accuracy on the anonymized data. Results from our prototype implementation demonstrate that our proposed scheme enhances privacy for data analytics. With varied data privacy requirement levels, we incur an average information loss in delay of 1.95% compared to other solutions that average a loss of 12.7%.

1 INTRODUCTION

Streaming data are real-time and continuous data flows that are ordered implicitly by arrival time or explicitly by timestamps. Examples include phone-calls and network monitoring. Mining continuous data streams is useful because it enables data holders or organizations to learn hidden knowledge and patterns through analyzing the data. For instance, in newly industrialized countries law enforcement agencies are encouraging users to report crime covertly via electronic crime reporting systems based on mobile phone technology (Mark-John and Kayem, 2014; Jensen et al., 2012; CryHelp-App, 2014). Real-time data analysis is important in enabling these agencies address reported crime more effectively and efficiently. However, often times these law enforcement agencies are not equipped with the on-site expertise required to analyze the data efficiently in real-time. It is therefore a cost-effective strategy to transfer streaming crime data to a third party service provider (Qiu et al., 2008).

Since crime data is inherently privacy sensitive, it makes sense to ensure that the outsourced data is protected from all unauthorized access including that of an "honest-but-curious" data mining service provider. Cryptographic techniques have been studied for protecting outsourced data from unauthorized access but have been shown to create a high overhead in terms of querying and updates, making analyzing large volumes of data in real-time is a time consuming process (Vimercati et al., 2010; Kayem et al., 2011). Other privacy preserving techniques for big data include those based on differential privacy. However, differential privacy techniques are better suited to static repositories as opposed to smaller sizes of streaming data (Dwork, 2006). Anonmyization schemes are a better alternative than cryptographic and differential privacy approaches to protecting the privacy of streaming data because of the time sensitivity of the data (Guo and Zhang, 2013). Most existing streaming data anonymization schemes are based

Busayo Sakpere A. and V. D. M. Kayem A..
Adaptive Buffer Resizing for Efficient Anonymization of Streaming Data with Minimal Information Loss.
DOI: 10.5220/0005288901910201
In *Proceedings of the 1st International Conference on Information Systems Security and Privacy* (ICISSP-2015), pages 191-201
ISBN: 978-989-758-081-9

on the k-anonymity technique for privacy preservation. This is because k-anonymity techniques offer a simple and effective approach to producing data with integrity (Bayardo and Agrawal, 2002). K-anonymity achieves privacy preservation by using generalization and suppression to ensure record indistinguishability (Sweeney, 2002).

As a result of rapid change in streaming data, there is a need for anonymization to happen fast with minimal delay. Failure to keep up with the changes in data stream during anonymization may lead into information loss (Guo and Zhang, 2013).

1.1 Motivation and Problem Statement

Streaming data anonymization algorithms rely on buffering mechanisms to hold the data temporarily while it is anonymized (Guo and Zhang, 2013; Cao et al., 2008; Zhang et al., 2010). Typically, anonymization of data streams require an optimal buffer size in order to enhance privacy preservation. However, intermittent data streams make determining an adequate buffer size to guarantee effective anonymization a challenge.

To determine the buffer size for effective anonymization, existing k-anonymization schemes arbitrarily choose an integer number to represent the number of records needed for effective anonymization. For instance, if the buffer size is set to 20 records, this implies that anonymization will only begin when there are 20 records in the buffer. This approach delays anonymization and so results in a high degree of information loss in scenarios involving delay-sensitive data especially if the data stream is slow as may be the case in crime-reporting.

Zakerzadeh and Osborn, 2013 have shown that count-based buffering approaches (like the one we have just described) can incur record expiry rates of as high as 61.3%. This is not desirable for delay sensitive scenarios where both the privacy of the anonymized data as well as accuracy in query results are important. Furthermore, in crime reporting scenarios, the time-sensitivity of the records requires that the data analytics service provider is provided with a comprehensive privacy preserving dataset that can be analysed efficiently in real-time in order to ensure query result accuracy.

The problem we seek to address therefore, is that of coming up with an approach to resizing the buffer to ensure efficient streaming data anonymization (for privacy preservation) with minimal information loss (for query accuracy) in a delay or time-sensitive context such as one involving reported crime data. In the next sub-section, we briefly present our approach to addressing this problem.

1.2 Contribution

We propose an adaptive buffer resizing scheme to minimize record suppression and information loss due to delay during anonymization of intermittent streaming data.

Firstly, we model our buffering mechanism as a time-based tumbling sliding window because of the time-sensitivity of crime data. The buffer size and rate of arrival of the streaming crime data affect the rate of information loss and the levels of privacy offered by the anonymization scheme.

Secondly, we develop a solution to adaptively readjust the size of the sliding window based on the arrival rate of data that follows a Poisson process.

As a further step, we employ a time-based metric in evaluating the data records to prioritize processing (anonymizing) records that are nearing expiry. We handle this by either including the selected record(s) in a subsequent sliding window (buffer) or including the record(s) into a reusable anonymity cluster.

Results from our prototype implementation demonstrate that in addition to enhancing privacy of the data, our proposed scheme outperforms previous schemes with an average information loss of 1.95%.

1.3 Outline

The rest of the paper is structured as follows. In Section 2, we present related work highlighting the weaknesses of existing data stream anonymization schemes. Section 3, presents our proposed dynamic buffer sizing solution using the Poisson probability distribution and the time-based tumbling sliding window. The arrival rate of data that follows a Poisson process influences the the size of the the time-based tumbling sliding window. In Section 4, we present results from our implementation and conclude in Section 5.

Other domain where the application of our Poisson Model concept to k-anonymity can be applied include stock companies and hospitals. For example, a stock company needs to investigate its sales daily in order to adjust stock or marketing strategy promptly and a hospital needs to release its daily medical records for research purpose.

2 RELATED WORK

Proposed k-anonymity schemes for handling streaming data use the concept of a sliding window or buffer

to temporarily store data based on a pre-defined processing delay constraints such as time or record-count (Patroumpas and Sellis, 2006; Li et al., 2008; Guo and Zhang, 2013; Zhang et al., 2010; Zakerzadeh and Osborn, 2011; Zakerzadeh and Osborn, 2013; Cao et al., 2008). Processing delay constraints ensure that information loss (delay) is minimized while the buffer holds the portion of the streaming data to be anonymized.

The first reported algorithm that considers k-anonymity on streaming data for privacy protection is Stream K-anonYmity (SKY) (Li et al., 2008). The algorithm searches the specialisation tree to find the most specific node that generalises a new record. SKY needs a specialization tree even for anonymizing numerical values which makes anonymization process more tedious because of the difficulty in finding a suitable hierarchy on the tree (Zakerzadeh and Osborn, 2013).

Continuously Anonymizing STreaming data via adaptive cLustEring (CASTLE) solves the aforementioned deficiency of SKY which emerges as a result of using specialization tree for its anonymization through the use of a clustering process. CASTLE relies on the count-based delay constraint for imposing constraints on the size of the buffer (Cao et al., 2008). However, one of the key challenges that CASTLE faces is that of determining an optimal bound on the number of records to which the buffer needs to be constrained. Furthermore, since the buffer size is fixed at runtime, CASTLE fails to handle changing speeds of streaming data flows effectively.

Other data stream anonymization techniques/algorithms that use a similar delay-constraint approach to that of CASTLE include K-anonymIzation Data Stream based on sliding window (KIDS) (Zhang et al., 2010), Fast clustering-based k-Anonymization approach for Data Streams (FADS) (Guo and Zhang, 2013) and B-CASTLE (Wang et al., 2010).

The Fast Anonymizing Algorithm for Numerical STreaming data (FAANST) addresses the challenge inherent in CASTLE inspired approaches by delaying the start of the anonymization process until the buffer is full (Zakerzadeh and Osborn, 2011). This allows for batching in terms of outputing results and recycling of records that the scheme was unable to anonymize during a given batch of data. A major drawback of FAANST is that time-sensitive records that are withheld and recycled may expire. The consequence of this is that such expired records lead to high information loss.

The delay-sensitive FAANST scheme addresses the issue in FAANST with a user-defined soft deadline for processing each record in the buffer (Zakerzadeh & Osborn, 2013). A major drawback of the delay-sensitive FAANST scheme is that there is no way of deciding whether or not unanonymizable records would be anonymizable during the next sliding window. So a record can get repeatedly recycled until it actually expires. Another drawback of the delay-sensitive FAANST is that the verification of record expiration generates additional performance overhead (Zakerzadeh & Osborn, 2013).

A detailed survey of existing data stream anonymization algorithms in relation to reported crime streaming data is given in (Sakpere and Kayem, 2014). It is clear from current literature in data stream anonymization that the issue of adaptive buffer resizing in order to minimize information loss in terms of delay and to avoid expiration of records still needs to be addressed. Minimizing information loss in terms of delay is important in generating anonymized reported crime data that is shared with third party service providers. It is important to anonymize data because it protects users' data (Sweeney, 2002). The next section describes our proposed solution.

3 ADAPTIVE BUFFER RE-SIZING SCHEME

In this section, we present our proposed adaptive buffer re-sizing approach. The buffer size and rate of arrival of the streaming data affect the rate of information loss and the levels of privacy offered by the anonymization scheme. In order to minimize information loss we use a time-based tumbling sliding window to adjust the size of the buffer with respect to the arrival rate of the data.

3.1 Buffer Streaming Data

This section explains the concept of sliding window as illustrated in Figure 1.

A Data Streams, DS, is defined as a real-time and continuous data flow ordered implicitly by arrival time or explicitly by timestamps.

Definition 1: A sliding window, say sw_i, is a subset of the data stream, DS where DS = $\{sw_1, sw_2, sw_3, ..., sw_m\}$ implies that DS consists of a set of m sliding windows.

The sliding windows obey a total ordering such that for every $i < j$, sw_i precedes sw_j. Each sliding window, sw_i, only exists for a specific period of time T and consists of a finite and varying number of records, n, such that $sw_i = R_0, ..., R_{n-1}$.

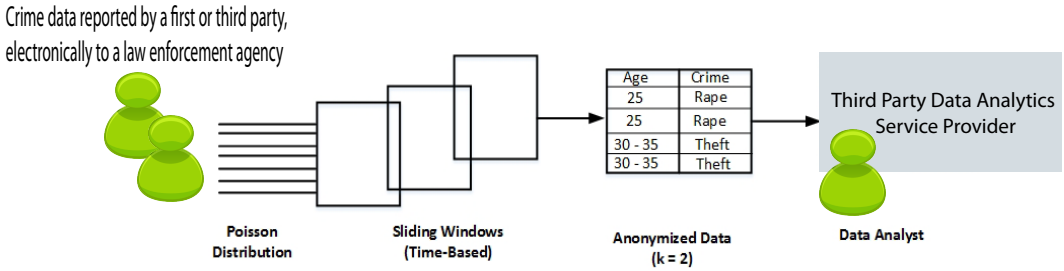

Figure 1: Overview of Buffer Resizing Process.

We use Poisson probability model to predict the rate of data flow in the next sliding window, sw_{i+1}, based on the rate of flow in a previous sliding window, sw_i. We opted to use a Poisson model because the Poisson distribution is concerned with the number of success that an event occurs in a given unit of time. This property of the Poisson model makes viewing the arrival rate of the reported crime data as a series of events occurring within a fixed time interval at an average rate that is independent of occurrence of the time of the last event (Li, 2006). Only one parameter needs to be known: the rate at which the events occur, which in our case is the rate at which crime reporting occurs.

3.2 Preliminaries

In this section, we present our proposed approach to addressing the adaptive buffer re-sizing problem. The buffer size and rate of arrival of the streaming crime data affect the rate of information loss and the levels of privacy offered by the anonymization scheme.

To better understand how our scheme works, we divide our adaptive buffer sizing scheme into six phases namely: Initial Buffer Size, Reduction of Information Loss, Inclusion of Suppressed Records into the Next Sliding Window, Determination of Arrival Rate, Possible Optimal Sizes for the Next Sliding Window using Poisson Probability Distribution and Final Decision on the Size of the Next Sliding Window.

3.2.1 Phase 1: Initial Buffer Size

Let T be the time for which a sliding window, sw_i, exists, where T is a time value that is bounded by a lower bound value, t_l, and an upper bound value, t_u, then:

1 k-anonymization algorithm is applied to the data that was collected in the sliding window, sw_i, during the period T

2 Essentially $sw_i = T$

3 All records that are not anonymizable from the data collected in sw_i are suppressed or excluded from the dataset released for publication

We begin by setting the size of the buffer to some initial threshold value, T. For example, in previous work (Zakerzadeh and Osborn, 2013), values between 2000ms and 5000ms have been used as the time interval in which a record can stay in the buffer. In line with our threshold value, t_l = 2000ms and t_u = 5000ms.

Example 1: Consider the dataset provided in Table 1 that has a time defined size of 5000ms for a sliding window, sw_i. This implies that the k-anonymization algorithm is applied to the data that was collected in the sliding window, sw_i, during the period $T = 5000ms$. The anonymization process was handled with a k-anonymity scheme in which we used $k = 3$ as the anonymization metric. We chose k = 3 because of the small data set which consists of only 10 records. A higher value of k will lead to higher information loss. All records that are not anonymizable from the data collected in sw_i are suppressed (excluded) from the dataset released for publication.

To achieve anonymization on Table 1 we used the crime taxonomy tree in Figure 2 by clustering records that belong to the same parent node and this results in Table 2.

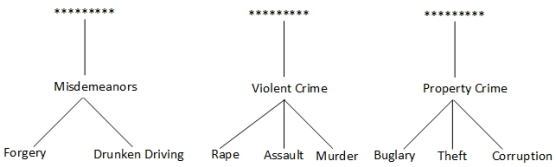

Figure 2: Crime Taxonomy Tree.

3.2.2 Phase 2: Reduction of Information Loss

Let C be a set of anonymized clusters where C = $\{c_1, c_2, c_3, ..., c_m\}$. A cluster is anonymized if it satisfies the k-anonymity requirements. k-anonymization algorithm requires that records be classified into clus-

Table 1: Data for Sliding Window, $T = \text{sw}_1 = 5000\text{ms}$; $T_A = 80\text{ms}$.

Record ID	Reported Crime	Age	WaitingTime $= T_S$
1	Vandalism	60	4782
2	Murder	20	4017
3	Theft	50	3361
4	Corruption	60	2566
5	Rape	30	2118
6	Burglary	70	2069
7	Forgery	35	1492
8	Arson	40	1214
9	Drunken Driving	50	417
10	Robbery	40	100

Table 2: Results for k-anonymization of $\text{sw}_1, k = 3$ and $T = 5000\text{ms}$.

Cluster 1	Cluster 2	Cluster 3
(Violent Crime, 20-40)	(Misdemeanors, 35 - 50)	(Property Crime, 40-70)
2	7	1
5	9	3
10		4
		6
		8

ters of at least size k, such that each record in the cluster is indistinguishable from at least *k-1* records. A record, R_i, is unanonymizable or suppressible if it does not fit into any of the cluster in set C.

This second phase attempts to reduce information loss that is likely to occur as a result of suppressed/unanonymizable records in Phase 1. In order to minimize the rate of information loss due to the unanonymizable records, we either include these unanoymizable records in a subsequent sliding window, say sw_{i+1}, or incorporate them into already anonymized clusters (reusable cluster) of data that are similar in terms of content. We describe a reusable anonymity cluster as one that has successfully published a set of anonymized records whose privacy and information loss levels are not negatively impacted by the inclusion of the suppressed record(s).

Example 2: Searching the output of the k-anonymization process in phase 1 i.e. Table 2 for unanonymizable/suppressed records, we note that records with ID 7 & 9, i.e. R_7 and R_9 are not anonymizable with the dataset in the current sliding window sw_1 because the group of records they are categorized into does not contain sufficient records to meet the k-anonymity requirement of $k = 3$. Therefore, we need to decide whether to process the records R_7 and R_9 in the next sliding window sw_2 or whether to find an appropriate reusable cluster into which to incorporate the records instead.

3.2.3 Phase 3: Inclusion of Suppressed Records into the Next Sliding Window

Let sw_i be the time-size of the previous sliding window, let T_S be the time for which a suppressed Record, R_i was stored in a previous sliding window, sw_i, and T_A is the time it took to carry out anonymization in the previous window, sw_i. We therefore compute the expiry time of R_i as follows:

$$T_E = \text{sw}_i - T_S - T_A \quad ...(1)$$

In order to determine whether or not a suppressed record can be included in a subsequent sliding window, say sw_{i+1}, we compute its expiry time T_E using equation 1 and compare the value of T_E to the bounds for acceptable sliding window sizes $[t_l, t_u]$.

Example 3: From Table 2, records R_7 and R_9 are unaonymizable. In order to determine whether or not to include these records into the next sliding window, sw_2, we compute the remaining time $T_E(R_i)$ of both records and compare both values to the bounds for acceptable sliding window sizes. From Table 1, $T_S = 5000$ and $T_A = 80$. We therefore compute $T_E(R_i)$ using equation 1 by subtracting T_S and T_A from $\text{sw}_1 = T$ which in this case gives $T_E(R_7) = \text{sw}_1 - T_{S_7} - T_A = 5000 - 1492 - 80 = 3428\text{ms}$ and $T_E(R_9) = \text{sw}_1 - T_{S_9} - T_A = 5000 - 417 - 80 = 4503\text{ms}$. Given that $t_l = 2000\text{ms}$ and $t_u = 5000\text{ms}$, it follows that $t_l \leq T(R_7), T(R_9) \leq t_l$ and we can conclude that it makes sense to incorporate R_7 and R_9 into sliding window sw_2.

3.2.4 Phase 4: Determination of Arrival Rate

Let U be a set of unanonymized clusters of an anonymization process where $\text{U} = \{\text{u}_1, \text{u}_2, \text{u}_3, ..., \text{u}_n\}$. A cluster is unanonymized if it does not satisfy k-anonymity requirement.

Starting with the unanonymizable cluster that has the suppressed record, R_i, with the lowest T_E and whose value falls within the acceptable sliding window bound, $[t_l, t_u]$, the algorithm checks for other suppressed records that belong to the same unanonymized cluster, u_i, as R_i. We then proceed to find the rate of arrival, λ, of data in that unanonymized cluster u_i, within the time interval, sw_i and compute the expected arrival rate of records required to

anonymize R_i within its expiry time, T_E using equation 2.

$$\lambda = \frac{+|u_i|}{\mathrm{sw}_i} \times \mathrm{T}_E \quad ...(2)$$

Example 4: In order to decide on what the optimum size of sw_2 should be set to, we consider the expiry time, T_E, of the suppressed records in sw_1. Since $T_E = 3428ms$ for R_7 and 4508ms for R_9, $\mathrm{sw}_1 = 5000ms$ and $k = 3$ is being used as the k-anonymization metric and both records (R_7 and R_9) fall under the generalization attributes of (Crime = "misdemeanors") and (Age = "35 - 50"), therefore we require that at least 1 similar record arrive during sw_2 in order to ensure that anonymization succeeds and thereby avoiding information loss from record expiry due to failure to anonymize the records. Starting with the least T_E, 3428, we compute $\lambda_{i+1} = \lambda_2$ for R_7 as follows:

$$\lambda_2 = \frac{\text{Number of Records}}{\mathrm{sw}_1} \times \mathrm{T}_E = \frac{2}{5000} \times 3428$$

R_7 gives $\lambda_2 = 1.37$.

3.2.5 Phase 5: Optimal Size for the Next Sliding Window using Poisson Probability

Let λ be the expected arrival rate of data in an unanonymized cluster, u_i, in a sliding window, sw_i and n is the number of records u_i required to undergo proper anonymization. Then, the probability that an unanonymizable/suppressed record R_i in u_i would be anonymized in the next sliding window, sw_{i+1}, can be calculated using equation 3

$$f(\mathrm{sw}_{i+1}, \lambda) = \Pr(\mathrm{i} = 0 \ldots. \mathrm{n}) = \frac{\lambda^i e^{-\lambda}}{\mathrm{i}!} \quad ...(3)$$

where λ is the expected data arrival rate, e is the base of the natural logarithm (i.e. $e = 2.71828$), n is the total number of observation and i is the number of records under observations. Therefore the probability of having n or greater than n records arrive in the stream within time T_E is

$$1 - \sum_{i=0}^{n-1} Pr \quad ...(4)$$

where Pr is the probability outcome of equation 3.

The expected arrival rate, λ, from phase 4 is then used to determine the probability of arrival of the minimal number of records, n, we require in order to guarantee that delaying the anonymization of the suppressed record, R_i, to the sliding window sw_{i+1} will not adversely increase information loss. We achieve this by finding the probability that n records will actually arrive in the data stream within time, T_E, in order to anonymize the suppressed record, R_i. We use the expression in equation 3 to compute the probability of having i = 0...n records arrive in the stream within the period T_E and equation 4 to find out the probability that n or more than n records will arrive in the stream within T_E.

Example 5: From example 4, the number of unanonymizable records in the unaonymizable cluster ("misdemeanors", "35 - 50") is 2 i.e. R_7 and R_9. Substituting $\lambda_2 = 1.37$ into equation 3 and subsequently into equation 4, we find the probability Pr ($\geq$ 1 record belonging to group 2 arrive in the next 3428 seconds) = 1 - Pr (0) = 1 - 0.25 = 0.75.

3.2.6 Phase 6: Final Decision on the Size of the Next Sliding Window

Let δ be a pre-set probability threshold and Pr be the result of equation 4. If $Pr \geq \delta$ then the size of the next sliding window, sw_{i+1}, is set to the expiry time of the suppressed record under consideration in equation 4.

If the result of equation 4 from phase 5 is greater than a pre-set probability threshold, δ, we set the size of the subsequent sliding window, sw_{i+1}, to the expiry time of the suppressed record under consideration. We then mark the suppressed record for inclusion in sw_{i+1} along with other suppressed records that have their T_E within bounds for acceptable sliding window sizes $[t_l, t_u]$. If the probability is less than the pre-set probability threshold, δ, we anonymize the suppressed records using a reusable cluster and calculate the size of sw_{i+1} using the next suppressed record whose T_E lies within the bounds $[t_l, t_u]$. In the event that the probability of all suppressed records is less than δ, we set the size of sw_{i+1} to a random number or some initial threshold value within the time bound, $[t_l, t_u]$. Finally, in order to decide into which reusable data cluster to include a suppressed record, R_i, our model searches for the cluster that covers the record and has the least information loss.

Example 6: The output of example 5 is 0.75. This implies that there is a high likelihood of having one or more records belonging to group 2 (where records R_7 and R_9 belong) arrive within the next 3428ms. Therefore the existence time (size) of the next sliding window, $\mathrm{sw}_2 = 3428$ms.

Algorithm 1: SWET (i,K).

1: **for** each sliding window sw_i, i:1 ...m **do**
2: **if** $((sw_i == 1)||(SuppRec == \phi))$ **then**
3: $sw_{iExistTime} \leftarrow T$
4: **else**
5: $sw_{iExistTime} \leftarrow RSWET(T_R, T_A, i, SuppRec)$
6: **end if**
7: $T_A \leftarrow$ Anonymization Processing Time
8: $SuppRec \leftarrow$ Suppressed Records
9: $T_R \leftarrow$ Remaining Time of Suppressed Records
10: Update Reusable Cluster (RC)
11: **end for**

Algorithm 2: RSWET($T_R, T_A, i, SuppRec$).

1: **Sort**: Sort T_R in ascending order and group by unanonymizable cluster
2: **for** j:1 ...$|SuppRec|$ **do**
3: **if** T_{R_j} - $T_A < T_l$ **then**
4: Anonymize $SuppRec_j$ using RC
5: Delete $SuppRec_j$
6: **else**
7: Calculate arrival rate, λ, of $SuppRec_j$ in the sliding window, sw_i
8: Find the Probability, P, of successful anonymization in sw_i
9: **end if**
10: **if** P or $\lambda > \delta$ **then**
11: $ExistTime_i \leftarrow T_{R_j} - T_A$
12: Add $SuppRec$ to sw_i
13: **break**
14: **else**
15: anonymize $SuppRec_j$ using RC
16: delete $SuppRec_j$ from $SuppRec$
17: **end if**
18: **end for**
19: **if** P or λ for all suppressed records $< \delta$ **then**
20: $ExistTime_i \leftarrow T$
21: **end if**
22: return ExistTime$_i$

3.3 Buffer Resizing: Algorithm

From the discussions in subsection 3.2, our framework for the Buffer Re-sizing anonymization of data streams can be summarized as follows:

Procedure Sliding Window Existence Time (SWET) has two parameters: i which is the ith sliding window under consideration and k is the k-anonymity requirement. Step 3 determines when to launch the first sliding window, sw_i, by randomly selecting its existence time, T, within the time bound $[t_l, t_u]$ i.e. $t_l \leq T \leq t_u$. Apply k-anonymization algorithm to the data collected in the sliding window during the period T. Step 5 call on procedure RSWET to determine when to launch a sliding window, sw$_i$, where $i \geq 2$. Step 7 computes the processing time used for carrying out k-anonymization. Step 8 search for unanonymizable/suppressed records sorted by their remaining time, T_R, and group by their unaonymized cluster. If no suppressed records exist, then randomly select existence time, T, for the next sliding window from $[t_l, t_u]$.

Procedure Reset Sliding Window Existence Time (RSWET) has four parameters: T_R which is a set that contains Remaining Time of all Suppressed Records, T_A is the time required to carry out anonymization process, i is the ith sliding window under consideration and SuppRec is a set that contains Suppressed Records. RSWET starts by sorting T_R of each suppressed records in ascending order. If there exists suppressed records/an unanonymized cluster whose $T_R - T_A \leq T_l$, then the reusable cluster will be used for its anonymization. Reusable cluster is a data structure of anonymized records whose privacy and information loss levels are not negatively impacted by the inclusion of the suppressed record. Otherwise, start with the suppressed record/group that has the least T_R. Then find the probability, P, that if such record(s) is/are included in the sliding window, sw_i, under consideration, it will be successfully anonymized before it expires.

If the λ or P result is greater than a threshold, δ, the sliding window size will be set to $T_{R_j} - T_A$ where T_{R_j} is the remaining time of the suppressed record under consideration. Otherwise, the algorithm fetches the next suppressed records. In the event that the value of λ or P for all suppressed records under consideration is less than the threshold, δ, the algorithm randomly select its existence time, T, within the time bound $[t_l, t_u]$ i.e. $t_l \leq T \leq t_u$.

4 IMPLEMENTATION AND RESULTS

The proposed framework was implemented on an Intel Core i5-3210 2.50 GHz machine with 4GB of random access memory (RAM). The operating system used was Ubuntu 12.10 and the CSE 467 k-anonymization implementation[1] was integrated into our adaptive buffering scheme using JAVA NetBeans IDE 7.0.1.

In order to simulate streaming data, we used the

[1]http://code.google.com/p/cse467phase3/source%20/browse/trunk/src/Samarati.java?r=64

file input stream functions in java that enabled data to be read in real-time from an external source data file into sliding window at random time interval of between 1 and 800 milliseconds. We randomized the time between 1 and 800ms in order to simulate a realistic crime report data stream with varying flow rates noting that this implies some slower report arrival rates (to mimic peaceful days when crime reports are few) and faster report arrival rates (to mimic disaster scenarios when reporting traffic is more bursty). A MySQL database was used as storage for the sliding window (buffer) and we assumed that data is read sequentially from the external file into the buffer.

Due to the large data set of crime data needed for this experiment, we synthetically generated a realistic crime data set that follows the structure of the CryHelp App using a random generator software[2]. The CryHelp App is a simple crime reporting application developed for mobile phones running the Android Operating System (CryHelp-App, 2014). Figure 3 shows some screenshots from the CryHelp App. The app was developed in conjunction with the University of Cape Town Campus Protection Service (CPS). The app enables users to send crime reports[3]. The synthetically generated crime dataset contains 1000 records and nine attributes that define the reporter's or victim's identity and the reported crime. The attributes of the dataset are divided into explicit, quasi and sensitive identifiers. In order to decide, if a tuple has exceeded its time-delay constraint, additional attributes such as arrival time, expected waiting time and entry time were included in the sliding window.

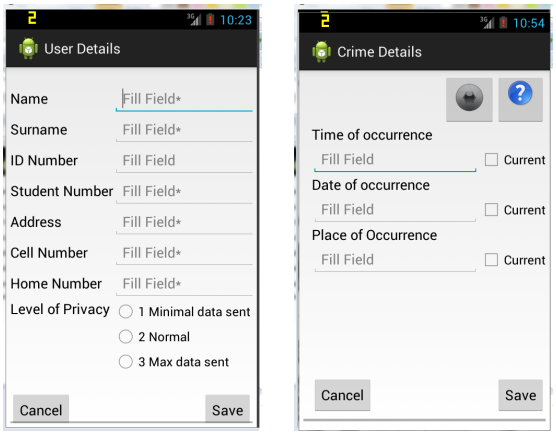

Figure 3: Screenshots from CryHelp App.

As a baseline case, for evaluating our proposed adaptive buffering scheme we implemented the proactive-FAANST and passive-FAANST. These algorithms are a good comparison benchmark because they are the current state-of-the-art streaming data anonymization that reduce information loss with minimum delay (Zakerzadeh and Osborn, 2013). The proactive-FAANST decides if an unanonymizable record will expire if included in the next sliding window while passive-FAANST searches for unanonymizable records that have expired. A major drawback of these two variants is that there is no way of deciding whether or not unanonymizable records would be anonymizable during the next sliding window. In our experiment, the proactive-FAANST and passive-FAANST solutions also use the reusable cluster concept as well but do not allow for overlapping of sliding windows, which our implementation does, nor do they model the flow rate of reported crime data as a Poisson process.

Our experiments were conducted to measure the following: information loss in terms of delay, information loss in terms of records, gains obtained from modelling the flow rate of the data as a Poisson process and using reusable anonymization clusters to reduce the number of unanonymizable/suppressed records. We ran the experiment ten different times and took the average of the results. The entire dataset size that was used included 1000 tuples with varying sliding window sizes.

4.1 Effect of Privacy Levels (k-anonymity Value) on Information Loss (Delay)

Figure 4 shows the effect of k-anonymity level on information loss with respect to delay (the number of expired records). For our experiment, the value of k-anonymity was varied from the values of 2 to 4. Our rationale for the choice of these k-values is that Zakarzadeh and Osborn (2013) use a k-value of 100 for 2000 records, so by analogy in a sliding window of 20 records a minimum k-value of 2 would suffice. We also ensured that no more than 5 records were suppressed per sliding window in order to achieve privacy preserving k-anonymization.

As a heuristic, the choice of $t_l = 2000$ms and $t_u = 5000$ms, is guided by values of delay that are used in published experimentation results (Zakerzadeh and Osborn, 2013). The sliding window size for our Poisson solution varies between t_l and t_u. The window size for passive and proactive solution in our experiment was chosen to be 8 records. The choice of this value was based on the number of records that arrive in our slow data stream within 5000ms. Within 5000ms, as low as 6-8 records and as high as 20 records were ob-

[2]http://www.mockaroo.com

[3]Further details about the app can be found in http://cryhelp.cs.uct.ac.za/download

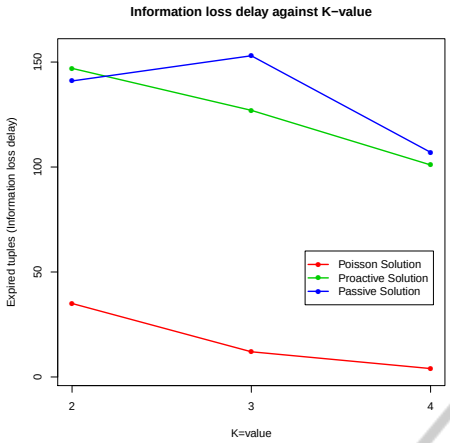

Figure 4: Performance comparison: Information loss with respect to Privacy Levels (expressed by the K-value).

served. We therefore, chose 8 records to minimize expired tuples.

In general, our approach shows that there are fewer expired tuples when compared to passive-FAANST and proactive-FAANST solutions. This is because before our Poisson prediction transfers suppressed records to another sliding window, it checks for possibility of its anonymization. In other solutions, there is no mechanism in place to check the likelihood of the anonymizability of a suppressed record before allowing it to go to the next sliding window/round. As a result, such tuples get sent to the next sliding window and have high tendency to eventually expire.

Our solution also shows that the lower a k-value, the higher the number of expired tuples. This is because the outcome of Poisson prediction is lower for higher k-values. As a result, there are fewer changes of sliding windows as k-value increases and this means there are fewer possibility of expired tuples.

The main goal of our solution is to reduce information loss in delay (i.e. to lower the number of expired tuples). Figure 4 depicts that our solution is successful in achieving its main goal, and the information loss (delay) in our solution is lower than passive and proactive solutions. In order to determine the total number of records that expired, a simple query was executed to retrieve all records that have stayed in the buffer longer than the upper limit threshold, t_u. To get the average expired records, we sum up the expired records in all the experiments and divide by the total number of experiments.

4.2 Information Loss (Records)

In order to measure the effect of the anonymity degree and Time-Based Sliding Window on information loss, we have set k-value to values between 2 and 4, δ i.e. the Poisson probability threshold to 0.4, and Time-Based Sliding Window to values between 2000ms and 5000ms. The choice of $t_l = 2000$ms and $t_u = 5000$ms, is guided by values of delay that are used in published experimentation results (Zakerzadeh and Osborn, 2013). The choice of δ = 0.4 is based on the various experiments we ran. We varied our δ from 0.4 to 0.6 and had the best output at 0.4.

To calculate information loss with respect to the number of records i.e. deviation of anonymized data from its initial form, we used the formula in equation 5 as it is in (Iyengar, 2002). We adopted this metric because it is a benchmark in many data stream anonymization schemes (Cao et al., 2008; Guo and Zhang, 2013; Zakerzadeh and Osborn, 2013).

$$\text{InfoLoss} = \frac{M_P - 1}{M - 1} \ldots (5)$$

M_p is number of leaf nodes in the subtree at node P and M is the total number of leaf nodes in the generalization tree. We calculate the information loss of a Sliding Window, $SW_i = \{R_1, R_2, R_3, \ldots, R_n\}$ as follows:

$$\frac{1}{n} \sum_{i=1}^{n} \text{InfoLoss}(R_i) \ldots (6)$$

The total information loss of a data stream is simply calculated by averaging the information loss of all sliding windows in it.

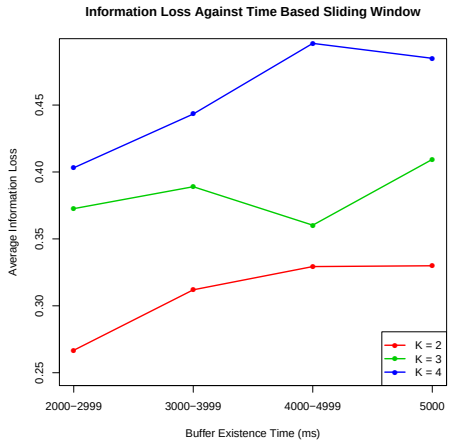

Figure 5: Effect of Sliding Window Size and Privacy Level Variation (expressed in terms of k-value) on Information Loss.

Figure 5 shows the effect of applying the time-based sliding window buffering mechanism and Poisson Probability distribution model on information loss. Here we observe that for smaller sliding window sizes information loss is lower in comparison to larger window sizes. One of the reasons for this is because the Poisson distribution considers unanonymizable records in a sliding window with higher size for consideration in a sliding window with lower size and the reusable cluster is more active at the lower sliding windows. This helps to reduce information loss.

We also observe that as the anonymity degree increases, privacy is enhanced and anonymization quality or output drops. It therefore implies that an increase in privacy level, k, also leads to increase in information loss.

4.3 Record Suppression

One of the goals of a good anonymization scheme is to ensure that information loss is minimal. Records suppression usually leads to a high information loss. The combination of the reusable cluster and the Poisson distribution helped to minimize the total number of suppressed records and as a result reduced information loss. However, our approach was unable to effectively recover some of the suppressed records because their deadlines were already exceeded or the sliding window size prediction for recovering those records was low and a suitable reusable cluster could not be constructed before the record expired.

As shown in Figure 6, a higher privacy level of k-value leads to the recovery of more suppressed records by the reusable cluster. This is because as the privacy level (i.e. k-value) increases, it becomes more difficult to achieve k-anonymization which leads to increase in suppressed records.

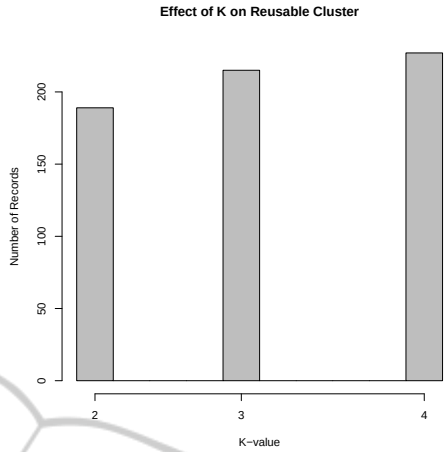

Figure 6: Impact of the Reusable Cluster on Minimizing Number of Suppressed Records.

5 CONCLUSIONS

In this paper, we used an adaptive buffer resizing solution to aid in supporting a privacy preserving streaming data k-anonymity algorithm by minimizing the rate of information loss from delay and unanonymized crime data reports. We began with an overview of the problem scenario which emerges in developing nations where the lack of data analytics expertise within a law enforcement agency makes the need to have a third party data analytics provider intervene to aid in fast crime report analysis. In addition, we highlighted the fact that the growing need to make the processed information available to field officers requires a mechanism for capturing crime reports in real-time and transferring these reports to the third-party service provider. While solutions in the literature that are hinged on cryptography have been shown to be successful in protecting data in outsourced scenarios from unauthorized access including that of "honest-but-curious" service providers, we note that querying encrypted streaming data is a time consuming process and that anonymization is a more practical approach to data privacy preservation in this case.

Anonymizing streaming data in a crime reporting context however, can have strong real-time requirements and therefore information loss can lead to faulty or misguided conclusions on the part of the data analytics service provider. Therefore, streaming data anonymization algorithms (schemes) need to be supported by good buffering mechanisms.

Our proposed approach uses the concept of modelling the flow rate of reported crime streaming data as a Poisson process that guides the sizing of a time-based sliding window buffer. The data collected in the buffer is subjected to k-anonymization to ensure privacy of the data. Results from our prototype implementation demonstrate that in addition to ensuring privacy of the data, our proposed scheme outperforms other with an information loss rate of 1.95% in comparison to 12.7% on varying the privacy level of crime report data records.

As future work, we will be extending this work to design an anonymization algorithm, which is efficient for processing reported crime data or streaming data that is highly categorical in nature. As well, in our adaptive buffering algorithm, we did not consider cases when anonymization might not be possible as a result of no records or few records in the stream as may often be the case in a crime data stream, we could

look at applying a perturbative method to anonymizing the data in this case (Aggarwal and Philip, 2008). In our experiment, the choice of the threshold for the probability of having enough requests within a specified time frame is set to an extrema of the presented benchmark (Zakerzadeh and Osborn, 2013). For future work, further benchmark could be considered in order to determine if a lower threshold performs better. In the future we will also make some inclusion for plans to work on real datasets. We can achieve this by carrying out some usability study to collect real data with the CRY-HELP App.

REFERENCES

Aggarwal, C. and Philip, S. (2008). A general survey of privacy-preserving data mining models and algorithms.

Bayardo, R. J. and Agrawal, R. (2002). Data privacy through optimal k-anonymization. *In Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on (pp. 217-228).*

Cao, J., Carminati, B., Ferrari, E., and Tan, K. L. (2008). Castle: Continuously Anonymizing Data Streams. *Dependable and Secure Computing, IEEE Transactions on, 8(3), 337-352.*

CryHelp-App (2014). http://people.cs.uct.ac.za/ tndlovu /(accessed, may 2014).

Dwork, C. (2006). Differential privacy. *In Automata, languages and programming (pp. 1-12).*

Guo, K. and Zhang, Q. (2013). Fast clustering-based anonymization approaches with time constraints for data streams. *Knowledge-Based Systems, Elsevier.*

Iyengar, V. S. (2002). Transforming data to satisfy privacy constraints. *In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 279-288).*

Jensen, K. L., Iipito, H. N., Onwordi, M. U., and Mukumbira, S. (2012). Toward an mpolicing solution for namibia: leveraging emerging mobile platforms and crime mapping. *In Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference (pp. 196-205).*

Kayem, A. V. D. M., Martin, P., and Akl, S. G. (2011). Effective cryptographic key management for outsourced dynamic data sharing environments. *In Proc. of the 10th Annual Information Security Conference (ISSA 2011), Johannesburg, South Africa, pp.1-8.*

Li, J., Ooi, B. C., and Wang, W. (2008). Anonymizing streaming data for privacy protection. *In Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on (pp. 1367-1369).*

Li, S. (2006). Poisson process with fuzzy rates. *In Fuzzy Optimization and Decision Making, 9(3), pp. 289-305.*

Mark-John, B. and Kayem, A. V. D. M. (2014). K-anonymity for privacy preserving crime data publishing in resource constrained environments. *In the 8th International Symposium on Security and Multinodality in Pervasive Environments, (SMPE 2014), Victoria, Canada - May 13-16, 2014.*

Patroumpas, K. and Sellis, T. (2006). Window specification over data streams. *In Current Trends in Database TechnologyEDBT 2006 (pp. 445-464).*

Qiu, L., Li, Y., and Wu, X. (2008). Protecting business intelligence and customer privacy while outsourcing data mining tasks. In *TEMPLATE'06, 1st International Conference on Template Production*. Knowledge and information systems, 17(1), pp. 99-120.

Sakpere, A. B. and Kayem, A. V. D. M. (2014). *A state of the art review of data stream anonymisation schemes.* Information Security in Diverse Computing Enviroments, 24. IGI Global, PA, USA., USA.

Sweeney, L. (2002). k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10 (05), 557-570.*

Vimercati, S. D. C. D., Foresti, S., Jajodia, S., Paraboschi, S., and Samarati, P. (2010). Encryption policies for regulating access to outsourced data. *ACM Trans. Database Syst., 35(2), pp. 12:1-12:46.*

Wang, P., Lu, J., Zhao, L., and Yang, J. (2010). B-castle: an efficient publishing algorithm for k-anonymizing data streams. *Proceedings of the 2010 Second WRI Global Congress on Intelligent Systems, Wuhan, China, 2010, pp. 132136.*

Zakerzadeh, H. and Osborn, S. L. . (2011). Faanst: Fast anonymizing algorithm for numerical streaming data. *In Data Privacy Management and Autonomous Spontaneous Security (pp. 36-50).*

Zakerzadeh, H. and Osborn, S. L. (2013). Delay-sensitive approaches for anonymizing numerical streaming data. *International Journal of Information Security, 1-15.*

Zhang, J., Yang, J., Zhang, J., and Yuan, Y. (2010). KIDS: K-anonymization data stream base on sliding window. *In Future Computer and Communication (ICFCC), 2010 2nd International Conference on (Vol. 2, pp. V2-311).*