Toward anonymizing IoT data streams via partitioning

Otgonbayar, Ankhbayar; Pervez, Zeeshan; Dahal, Keshav

Published in:
2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)

DOI:
10.1109/MASS.2016.049

Published: 16/01/2017

Document Version
Peer reviewed version

Toward Anonymizing IoT Data Streams via Partitioning

Ankhbayar Otgonbayar
School of Engineering and Computing,
University of the West of Scotland,
United Kingdom
ankhbayar.otgonbayar@uws.ac.uk

Zeeshan Pervez
School of Engineering and Computing,
University of the West of Scotland,
United Kingdom
zeeshan.pervez@uws.ac.uk

Keshav Dahal
School of Engineering and Computing,
University of the West of Scotland,
United Kingdom
keshav.dahal@uws.ac.uk

***Abstract*—IoT devices are capable of capturing physiological measures, location and activity information, hence sharing sensed data can lead to privacy implications. Data anonymization provides solution to this problem; however, traditional anonymization approaches only provide privacy protection for data stream generated from a single entity. Since, a single entity can make use of multiple IoT devices at an instance, IoT data streams are not fixed in nature. As conventional data stream anonymization algorithms only work on fixed width data stream they cannot be applied to IoT. In this work, we propose an anonymization algorithm for publishing IoT data streams. Our approach anonymizes tuples with similar description in a single cluster under time based sliding window. It considers similarity of tuple when clustering, and provides solution to anonymize tuples with missing value using representative values. Our experiment on real dataset shows that the proposed algorithm publishes data with less information loss and runs faster compared to conventional anonymization approaches modified to run for IoT data streams.**

I. Introduction

The integration of cloud computing, smart devices, and data analytics has triggered a new paradigm shift referred as Internet-of-Things (IoT). This innovation created huge network of internet enabled devices referred as "IoT devices" which can be utilized to monitor and manage home, machines, buildings, factories and almost anything which can interact with sensors, actuators, and embedded computers [1]. Utilization of IoT data has great value in todays world [2], with its application within the areas of healthcare, home automation, and infrastructure monitoring to name a few.

At one side IoT devices provide valuable data to service providers to provision contextually informed services; on the other side these devices can sense confidential information that can be used to breach individuals privacy. Anonymization offers a solution to this problem by transforming data and restraining attackers from binding published data with the data source i.e., data owner. First practical attempt to provide privacy preservation for static dataset was proposed by Sweeney as k-anonymity model [3]. Static data anonymization is extended to streaming information which renders data in dynamic environment [7, 8]; data arrive continuously, and anonymized and published in a sequence of time.

Data stream anonymization quality is defined by a tradeoff between data freshness and published data quality requirements [4]. For example, some publishers want fast anonymization, and may lose much data; while other might prefer to have less information loss on the data having less restricted time requirements to publish streaming data. Most conventional data stream anonymization approaches use sliding window technique [4–12]. Published data quality depends on the sliding window capacity i.e., more number of tuple accumulated for each anonymization process guarantees less information loss for anonymized data. On the other hand, processing more number of tuples in a single iteration may lead to overflow and it may take more time to anonymize resulting in lower freshness of data quality [9].

Conventional data stream anonymization approaches are only proposed to anonymize a data stream having fixed number of attributes. Smart environments that make use of IoT devices consist of multiple internet enabled devices for different purposes. Therefore, an individual can have multiple IoT devices which can be used at any time. In this scenario, an individual can produce multiple data streams with missing values thus creating a data stream which cannot be processed by conventional data stream anonymization algorithms.

The quality of data stream anonymization is defined by three aspects; a) data freshness, b) information loss, and c) privacy level [4, 7, 9]. Conventional anonymization algorithms were not designed to handle data streams with missing values and multiple attribute combinations which significantly complicates clustering tuples with same description. To cope with these challenges, we propose IoT stream anonymization via partitioning, which anonymizes IoT data streams. It extends the k-anonymous privacy model to IoT data streams for publishing IoT streaming data. It assigns tuples into partitions with regards to their description. Partitioning limits the number of tuples for anonymization process in a single round; however, when partition does not have enough number of tuples for the anonymization it merges the partitions based on their similarity, then creates cluster using k-nearest neighbor (KNN) algorithm [13] and finally published the tuples with cluster representation.

To the best of our knowledge this paper contributes first ever formal definition of anonymization problem for IoT data streams, and presented an enhanced k-anonymity privacy algorithm to handle missing values in IoT streams.

Comprehensive experiments on real datasets demonstrate the efficiency and effectiveness of the proposed algorithm. The rest of this paper is organized as follows: Section 2 reviews the related work. Section 3 introduces the basic concept of IoT data stream anonymity and defines an anonymization model. Section 4 presents the proposed algorithm. Section 5 reports the experimental results. The paper is concluded in Section 6.

II. Related work

With the advancement of data sensing technologies, data stream anonymization has gain lot of attention. It publishes streaming data with less information loss in short time [8]. Sliding window is a popular anonymization technique for data stream anonymization, which anonymizes most recent tuples of data and publishes freshly received data. Mainly two types of sliding window techniques are used "time based sliding window" [7, 9] and "count based sliding window" [4, 8, 12]. Cao et. al., proposed first real-time anonymization scheme with a count-based sliding window called CASTLE, providing k-anonymity and l-diversity on data stream using generalization [7]. For expiring tuples CASTLE releases them immediately; however, if expiring tuple is not assigned to a k-anonymous cluster, it performs merge and split operations to create a k-anonymous cluster. Furthermore, to minimize information loss, CASTLE adopts cluster re-using strategy to anonymize newly arriving tuples using generalization information of recently published cluster.

Hessam and Sylvia introduced FAANST, a count based sliding window anonymization algorithm for numerical data stream [9]. The main purpose of FAANST is to enhance data quality. To achieve this, authors proposed information loss constraint on each cluster. It outputs k-anonymized clusters having less than δ information loss. Since, it uses count based sliding window, tuples are only published when window is filled with certain number of tuples. FAANST outperforms CASTLE in terms of data quality and time complexity.

Zhou et al., developed a three phase method for generalizing streaming data [10]. In the first stage, their algorithm makes decision about data publishing based on the information loss of clusters. In the second step, the distribution of the data stream is incorporated in the decision-making process of cluster anonymization. In the third step, the effect of cluster anonymization on future tuples is considered. The authors considered that data publishing based on uncertainty may not be effective because it does not consider the distribution of tuples in a stream. They added new features to data stream anonymization by considering the distribution of tuples. For instance, if two tuples are expiring at the same time it publishes tuple from dense area before publishing tuple from the crowded area.

Guo and Zhang proposed data stream anonymization with time constraint called FADS [8]. It resolved the problem of cluster overload in CASTLE for homogeneous data stream having non-negligible difference between the arriving tuples. FADS considered time delay as the main constraint, and set a time constraint on sliding window, and cluster set. By this, the longest time for a tuple to stay in memory is δ, and re-usable k-anonymized cluster set can hold clusters for certain amount of time. The authors noted that the complicated merge and split operations of CASTLE are unnecessary since the cluster size is already constrained by k. FADS is easily adapted to anonymize under L-diversity privacy model, by employing hash data structure to group tuple by their attribute values.

Wang et al., found that CASTLE [7] generates few huge clusters when applied on some data stream resulting in frequent split function to create many k-anonymized small clusters [14]. The split and merge processes in CASTLE were time consuming and resulted in higher information loss during re-clustering and publishing. To deal with the flaw B-CASTLE was proposed. It set a threshold α as a cluster size by changing the best selection and merging features of CASTLE. B-CASTLE demonstrated higher efficiency and lower complexity as compared to CASTLE.

III. Preliminaries

For this work we define IoT data stream as:

Definition 1 (IoT data stream) *Let* $A_s = \{a_1, a_2, , a_m, q_1, q_2, , q_n\}$ *be a main attribute set,* $q_1, q_2, , q_n$ *are quasi-identifier attributes which will be anonymized before publishing, and* $a_1, a_2, , a_m$ *are other attributes. An IoT data stream* (IS) *is defined as* $IS = (pid, A_t, ts)$ *, where* pid *is the personal identity,* $A_t \subseteq A_s$ *and* ts *is the arrival time of a tuple.*

$k-$anonymity for IoT data stream can be defined as:

Definition 2 (k-Anonymity of IoT data stream) *Let* $IS(pid, A_t, ts)$ *be an IoT data stream with the main attribute set* $A_s = \{a_1, a_2, , a_m, q_1, q_2, , q_n\}$ *where* pid *is the personal identity,* ts *is the arrival time of tuple in IS. Let* IS_{out} *be the anonymized stream generated from IS,* $f : IS \rightarrow IS_{out}$ *if* IS_{out} *satisfies:*

a) For $\forall t \in IS, \exists t' \in IS_{out}$ *corresponds to t.*

b) For $\forall t' \in IS_{out}, DP(EQ(t')) \geq k, EQ(t') = \{t \mid t \in IS \cap f(t).q_i = t' \cdot q_i, q_i \in A_t\}$ *is the equivalence class of the tuple* t'*. DP is the number of distinct values if* pid *of tuple in* $EQ(t')$*.*

Definition 3 (Cluster generalization) *Let* $G_c^*(G_1, G_2, ..., G_m)$ *be a generalization of cluster C, then*

a) $g_i = [r_{i.min}, r_{i.max}]$ *where* $r_{i.min}(r_{i.max})$ *is the minimum(maximum) of the values of all tuple in C that have attribute value on* q_i*.If* q_i *is numeric attribute.*

b) $g_i = H_{i.lowest}$*, where* $H_{i.lowest}$ *is the lowest common ancestor of the* v_{qi} *values of the tuple in cluster C that have attribute value on* q_i*.If* q_i *is categorical attribute.*

Definition 4 (k-anonymous cluster) *A cluster C built from a data stream IS satisfies k-anonymity if:*

1) $DP(C) \geq k$, *where DP(C) is number of distinct values of pid of the tuple in C*;
2) *The tuples in C published with C''s generalization, For* $\forall t \in C, \exists t' = g_c$ *where* $t' \in IS_{out}$ *is the anonymized tuple corresponding to t,* g_c *is the generalization of C that contains t, then cluster C called k-anonymous.*

Definition 5 (Information loss of tuple) *The information loss caused by generalizing tuple* $t(pid, A_t)$ *to generalization* $G_t(g_1, g_2, ..., g_m)$ *is*:

$$InfoLoss(t, G_t) = \frac{1}{|G_t|}\Big(\sum_{q_i \in A_t} Loss(v_{qi}) + |G_t| - |A_t^q|\Big) \tag{1}$$

where, $|A_t^q|$ *is the number of quasi-identifier attributes in* A_t*, and* $Loss(v_{qi})$ *is the information loss of t on quasi-attribute* q_i *caused by generalization, which is defined as:*

$$InfoLoss(t, G_t) = \begin{cases} \frac{r_{i.u} - r_{i.l}}{R_{i.u} - R_{i.l}} & g_i \in [r_{i.l}, r_{i.u}] \\ \frac{|leaves(H_i)|-1}{|leaves(DGH_i)|-1} & g_i = H_i \end{cases} \tag{2}$$

where $[r_{i.l}, r_{i.u}]$ *is the range value for a numeric attribute* q_i*,* DGH_i *is the DGH of corresponding categorical attribute* q_i *,* $|leaves(H_i)|$ *and* $|leaves(DGH_i)|$ *is the number of nodes of* H_i *and* DGH_i *trees.*

Definition 6 (Distance between 2 tuples) *The distance between 2 tuples* $t_1(pid, A_1)$ *and* $t_2(pid, A_2)$ *is calculated on attributes those have received value in both tuples.*

$$Distance(t_1, t_2) = \frac{1}{|A_1^q \cap A_2^q|} \sum_{q_i \in A_1 \cap A_2} d_i(q_i) \tag{3}$$

$$d_i(q_i) = \begin{cases} \frac{|r_{i.1} - r_{i.2}|}{R_{i.u} - R_{i.l}} & q_i \text{ is numerical} \\ \frac{|leaves(H_i)|-1}{|leaves(DGH_i)|-1} & q_i \text{ is categorical} \end{cases} \tag{4}$$

where $|A_1^q|(|A_2^q|)$ *is the number of quasi-identifier attributes in* $A_1(A_2)$*,* $r_{i.1}(r_{i.2})$ *is the value of* $t_1.q_i(t_2.q_i)$ *if* q_i *is numeric attribute,* H_i *is the lowest common ancestor of* $t_1.q_i(t_2.q_i)$ *if* q_i *is categorical attribute. If* DGH_i *has only one leaf node* d_i*=0 to avoid division by zero.*

The quality of anonymization algorithm is measured by average information loss:

Definition 7 (Average information loss) *The average information loss of first N tuples from IoT data stream is:*

$$AvgInfoLoss = \frac{1}{N}\sum_{i=1}^{N} InfoLoss(t_i, G_i) \tag{5}$$

Where G_i *is generalization of tuple* t_i*.*

Definition 8 (Similar tuple) *Tuples* $t_i(pid, A_i)$ *and* $t_j(pid, A_j)$ *is called similar tuples when quasi-identifier of both tuples are same.* $\forall q_i \in A_i \Rightarrow q_i \in A_j$ *and vice versa.*

Definition 9 (Partition on attribute set A_k**)** *Let partition created on tuples* $P = t_i, t_j$*. If all tuples in partition defined by same quasi-attribute set we call P is partition on* A_i*.*

We use Jaccard's similarity coefficient to measure similarity of partitions[15].

Definition 10 (Similarity between 2 partitions) *Let* P_i *and* P_i *be a similar tuple partitions defined on* A_i *and* A_j *attribute sets respectively.*

$$Similarity(P_i, P_j) = \frac{|q_i \in A_i \cap A_j|}{|q_i \in A_i \cup A_j|} \tag{6}$$

Definition 11 (Delay constraint) *Let F be an anonymization scheme for varied data stream IS, if the anonymized data stream* IS' *published by F satisfies* $\forall t' \in IS', t'.ts - t.ts < \delta$ *,where t is corresponding tuple to* t' *in IS,* δ *is predefined positive real number. Then we call F satisfies delay constraint* δ*.*

IV. Anonymizing IoT data streams

The main strategy of the proposed algorithm is to anonymize IoT data streams using partitions (Definition 9) in sliding window. The idea behind partitioning is to limit the number of tuples which may be involved in single anonymization round over one cluster. The proposed algorithm assigns receiving tuple to partitions according to their attribute description. It then checks the size of the partition containing expiring tuple. If it is possible to form cluster from that partition, it uses KNN algorithm to find cluster generalization which satisfies k-anonymity (Definition 4) for IoT data stream. However, if the size of the partition is less than k, it expands the anonymization area using partition merging operation based on similarity measurement (Definition 10). To avoid overusing partition merging it implies re-using strategy.

Algorithm 1 IoT anonymization(IS,K,δ, T_{ck})

```
Let be Set_p is set of partition which will act as buffer, initialized empty.
Let be Set_kc is set of K-anonymous cluster, initialized empty.
while IS ≠ NULL do
    Read tuple t_i from IS and assign partition of Set_p or create new partition on it.
    Update range of all numeric attributes according to t_c.
    if Tuple is going to expire then
        DelayConstraint()
    end if
end while
while |IS| > 0 do
    DelayConstraint()
end while
```

The details of the proposed algorithm are presented in Algorithm-1 with four parameters: the IoT data stream IS, the k-anonymity (K), the delay constraint (δ), and the time

constraint (T_{ck}) for reusing k-anonymous clusters. The algorithm continuously read tuple and assign them to a partition set (denoted as Set_p hereafter) which plays a role of buffer. For expiring tuple, the proposed algorithm calls $DelayConstraint()$ presented in Algorithm-2 .

Algorithm 2 DelayConstraint ()

```
Delete expiring clusters from Set_kc
Let P_i be a partition that containing expiring tuple t_i
if |P_i| ≥ K − 1 then
    ClusterInPartition(t_i,P_i)
else if |Set_p| ≥ K − 1 then
    ClusterWithMerge() (t_i,P_i)
else
    OutputOrSuppress()
end if
```

At first, method $ClusterInPartition()$ presented in Algorithm-3, finds $k-1$ nearest neighbors of expiring tuple t using equation Eq. (3) and creates cluster over it. It then anonymizes expiring tuple using reusable k-anonymous cluster defined over the same partition which covers t. k-anonymous is based on information loss calculated by Eq. (2).

Algorithm 3 ClusterInPartition (t, P)

```
Find K − 1 nearest neighbors of expiring tuple t from partition P abd create cluster C_t on t and its K − 1 neighbors
Find K-anonymous cluster C_min from Set_kc which covers t, and defined from P has minimum information loss
if C_min found then
    if InfoLoss (C_min) > InfoLoss(C_t) then
        Use generalization information of C_min to anonymize t and RETURN
    end if
end if
Anonymize publish all tuple of C_t using its generalization and remove published tuples from P
Insert C_t into Set_ck
```

Method $ClusterWithMerge(P,t)$ from Algorithm-4, is proposed to perform clustering on multiple partitions while merging them via similarity (Definition 10). This procedure is called when we have more than k number of tuple in Set_p and former partition of expiring tuple t has less than $k-1$ number of tuples. It anonymizes expiring tuple t using reusable $k-$anonymous cluster which covers t. If no appropriate k-anonymous cluster is found it merges most similar partitions using Eq. (6). The merge operation continues until number of tuples in merged partition is greater or equal to $k-1$. Then it finds $k-1$ nearest neighbor of t from the merged partition, and creates cluster for neighbor and expiring tuples.

Procedure OutputOrSuppress() presented in Algorithm-5, tries to publish tuple with k-anonymous cluster that covers it with minimum information loss. If such cluster exist it anonymizes and outputs with the generalization information of that cluster otherwise tuple is suppressed and published.

Algorithm 4 ClusterWithMerge (t, P)

```
Find K-anonymous cluster C_min from Set_kc which covers t, and defined from P has minimum information loss
if C_min found then
    Use generalization information of C_min to anonymize t and RETURN
end if
while |P| < K − 1 do
    Find partition P_sim from Set_p which most similar to P that has maximum number of tuple
    Merge P_sim into P and remove P_sim from Set_p
end while
Find K − 1 nearest neighbors of expiring tuple t from partition P and create cluster C_new on t and its K − 1 neighbors
Adjust attaching representative values according to the tuples of C_new
Anonymize and publish all tuple of C_t using its generalization and remove published tuples from P
Re-assign remaining tuples into Set_p
```

Algorithm 5 OutputOrSuppress ()

```
while Set_p is not empty do
    t is a first tuple according to arrival time
    Find K-anonymous cluster C_min from Set_kc which covers t, and defined from P has minimum information loss
    if C_min found then
        Use generalization information of C_min to anonymize t
    else
        Suppress and output
    end if
    Remove t from Set_p
end while
```

V. Experimental evaluation

In order to estimate the performance of the proposed algorithm we compared it with FAANST [9] on adult dataset from UCI [1]. We removed all missing records from the original dataset and the total number of records decreased to 30162. For anonymization we selected four categorical attributes: age, $fnlweight$, $education-number$, $capital-gain$, and six numerical attributes: $capital-loss$, $hours-per-week$, $education$, $marital-status$, $work-class$, and $country$. In order to create dataset with repeating $pids$ 10 percent of record was randomly selected and added back into the dataset, similar

[1]Adult Data Set from UC Irvine Machine Learning Repository is widely used in the area of data publishing; accessible at https://archive.ics.uci.edu/ml/datasets/Adult

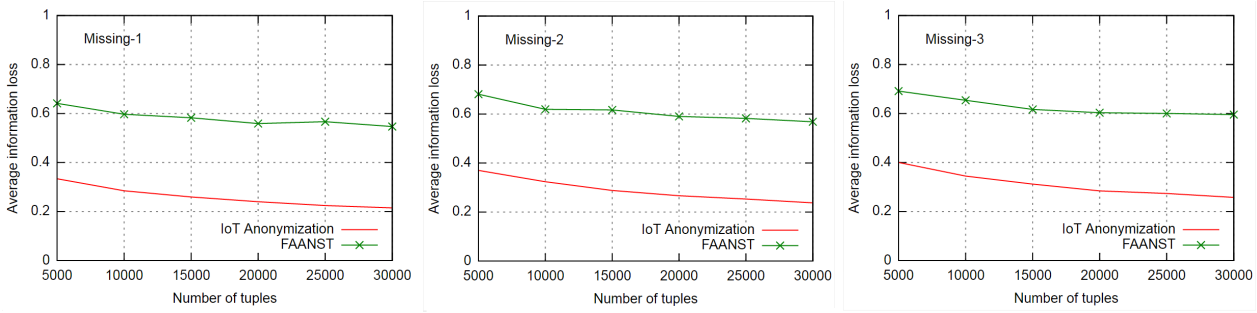

Fig. 1: Average information loss of FAANST and proposed IoT anonymization

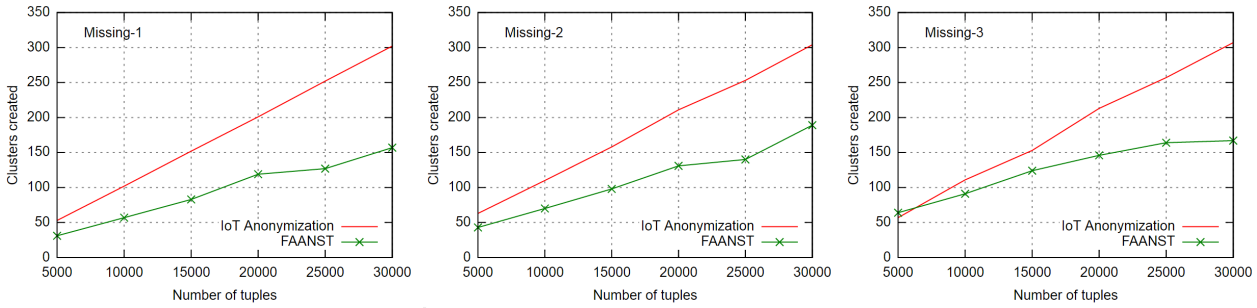

Fig. 2: Number of cluster created on FAANST and the proposed IoT anonymization

to [8]. Domain hierarchical graphs of categorical attributes are same as those in [10]. In order to create IoT data stream we set arrival order for each row of the dataset and removed up to three attributes from the tuples. The average information loss is calculated according to Definition 7.

TABLE I: Parameters of algorithms

Algorithm	**Parameters**
FAANST	K=100, Attributes=6, δ=2000, τ=0.5
IoT anonymization	K=100, Attributes=6, δ=2000, T_{ck}=200

FAANST [9] requires buffer to have certain amount of tuples in order to perform anonymization, therefore we can run FAANST on every partitions created by IoT data stream. Since this algorithm only works with numeric dataset we used six numerical attributes of the adult dataset. Description of the experiment parameter is shown in Table I. The average information loss of the proposed algorithm and FAANST is illustrated in Fig. 1; this result shows the proposed algorithm anonymizes data more efficiently than FAANST. The main reason that it out-performance FAANST is the time based sliding window which helps to accumulate more tuples in the current window. On the other hand the total number of tuples for the each partition is relatively less. Since FAANST does not employ merging technique, total tuples involved in FAANST anonymization of single partition is relatively less compared to the total data size. Fig. 2. illustrates the total number of clusters created by FAANST and the proposed algorithm. We considered every new tuple generalization as a cluster, therefore suppression of tuples are counted as newly created cluster. As shown in Fig. 2, total number of created cluster is increased when more attributes are missing. Also, FAANST published less number of clusters than the proposed algorithm. FAANST anonymization process starts when partition accumulated enough number of tuples; however, the proposed algorithm merges the partitions instead of waiting to accumulate enough tuples in partition. By this, it guarantees freshness of the arriving tuples and publishes more number of clusters.

VI. Conclusion

In this paper we presented IoT anonymization a novel algorithm that $k-$anonymizes IoT data streams generated from multiple IoT devices. It used time based sliding window technique to manipulate IoT streams by partitioning tuples based on their description. This preliminary operation helped to form cluster faster by localizing tuples and supports the partitions merging when needed. It was necessary to merge similar partitions to anonymize tuples with less uncertainty. The proposed algorithm outperformed conventional data stream anonymization approach which was modified to run in IoT data streams. In experiment we demonstrated the conventional stream anonymization can not be directly applied to IoT streaming data and require significant research and development. Further to this, we would like to investigate new privacy model for anonymizing cluster with missing data.

References

[1] S. Dey, A. Mukherjee, H. S. Paul, and A. Pal, "Challenges of using edge devices in iot computation grids," in *Parallel and Distributed Systems (ICPADS), 2013 International Conference on*. IEEE, 2013, pp. 564–569.

[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[3] L. Sweeney, "K-anonymity: A model for protecting privacy," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002. [Online]. Available: http://dx.doi.org/10.1142/S0218488502001648

[4] E. Mohammadian, M. Noferesti, and R. Jalili, "Fast: fast anonymization of big data streams," in *Proceedings of the 2014 International Conference on Big Data Science and Computing*. ACM, 2014, p. 23.

[5] K. Wang, Y. Xu, R. C.-W. Wong, and A. W.-C. Fu, "Anonymizing temporal data," in *2010 IEEE International Conference on Data Mining*. IEEE, 2010, pp. 1109–1114.

[6] P. Harbin, "Anonymizing streaming data for privacy protection," *Age*, vol. 25, no. 29, p. 35, 2008.

[7] J. Cao, B. Carminati, E. Ferrari, and K.-L. Tan, "Castle: Continuously anonymizing data streams," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 3, pp. 337–352, 2011.

[8] K. Guo and Q. Zhang, "Fast clustering-based anonymization approaches with time constraints for data streams," *Knowledge-Based Systems*, vol. 46, pp. 95–108, 2013.

[9] H. Zakerzadeh and S. L. Osborn, "Faanst: fast anonymizing algorithm for numerical streaming data," in *Data Privacy Management and Autonomous Spontaneous Security*. Springer, 2011, pp. 36–50.

[10] B. Zhou, Y. Han, J. Pei, B. Jiang, Y. Tao, and Y. Jia, "Continuous privacy preserving publishing of data streams," in *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*. ACM, 2009, pp. 648–659.

[11] J. Zhang, J. Yang, J. Zhang, and Y. Yuan, "Kids: k-anonymization data stream base on sliding window," in *Future Computer and Communication (ICFCC), 2010 2nd International Conference on*, vol. 2. IEEE, 2010, pp. V2–311.

[12] W. Wang, J. Li, C. Ai, and Y. Li, "Privacy protection on sliding window of data streams," in *Collaborative Computing: Networking, Applications and Worksharing, 2007. CollaborateCom 2007. International Conference on*. IEEE, 2007, pp. 213–221.

[13] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.

[14] P. Wang, J. Lu, L. Zhao, and J. Yang, "B-castle: An efficient publishing algorithm for k-anonymizing data streams," in *2010 Second WRI Global Congress on Intelligent Systems*, vol. 2. IEEE, 2010, pp. 132–136.

[15] P. Jaccard, *Nouvelles recherches sur la distribution florale*, 1908.