Privacy-Preserving Data Analytics

Do Le Quoc, Martin Beck, Pramod Bhatotia, Ruichuan Chen, Christof Fetzer, Thorsten Strufe

Abstract Real-time processing of user data streams in online services inadvertently creates tension between the users and analysts: users are looking for stronger privacy, while analysts desire for higher utility data analytics in real time. To resolve this tension, this paper describes the design, implementation and evaluation of PRIVAPPROX, a data analytics system for privacy-preserving stream processing. PRIVAPPROX provides three important properties: *(i)* Privacy: zero-knowledge privacy guarantee for users, a privacy bound tighter than the state-of-the-art differential privacy; *(ii)* Utility: an interface for data analysts to systematically explore the trade-offs between the output accuracy (with error estimation) and the query execution budget; *(iii)* Latency: near real-time stream processing based on a scalable "synchronization-free" distributed architecture. The key idea behind PRIVAPPROX is to combine two techniques together, namely, *sampling* (used for approximate computation) and *randomized response* (used for privacy-preserving analytics). The resulting combination is complementary — it achieves stronger privacy guarantees, and also improves the performance for stream analytics.

Do Le Quoc
TU Dresden, e-mail: do.le_quoc@tu-dresden.de

Martin Beck
TU Dresden, e-mail: martin.beck1@tu-dresden.de

Pramod Bhatotia
University of Edinburgh and Alan Turing Institute, e-mail: pramod.bhatotia@ed.ac.uk

Ruichuan Chen
Nokia Bell Labs, e-mail: ruichuan.chen@nokia-bell-labs.com

Christof Fetzer
TU Dresden, e-mail: christof.fetzer@tu-dresden.de

Thorsten Strufe
TU Dresden, e-mail: thorsten.strufe@tu-dresden.de

1 Introduction

Nowadays, many modern online services continuously collect users' private data for real-time analytics. This data normally arrives as a data stream and in huge volumes, requiring real-time stream processing based on distributed systems [16, 1, 3, 2].

In the current ecosystem of data analytics, the analysts usually have direct access to users' private data, and must be trusted not to abuse it. However, this trust has been violated in the past [23, 50, 42, 58]. A pragmatic ecosystem has two desirable, but contradictory design requirements: *(i)* stronger privacy guarantees for users, and *(ii)* high-utility stream analytics in real time. Users seek stronger privacy, while analysts strive for high-utility analytics in real time.

To meet these two design requirements, there is a surge of novel computing paradigms that address these concerns, albeit *separately*. Two such paradigms are *privacy-preserving analytics* to protect user privacy and *approximate computation* for real-time analytics.

Privacy-preserving analytics. Recent privacy-preserving analytics systems favor a distributed architecture to avoid central trust (see §4 for details), where users' private data is stored locally on their respective client devices. Data analysts use a publish-subscribe mechanism to perform aggregate queries over the distributed private dataset of a large number of clients. Thereafter, such systems add noise to the aggregate output to provide useful privacy guarantees, such as differential privacy [27]. Unfortunately, these state-of-the-art systems normally deal with single-shot batch queries, and therefore, these systems cannot be used for real-time stream analytics.

Approximate computing. Approximate computation is based on the observation that many data analytics jobs are amenable to an approximate rather than the exact output (see §4 for details). Such applications include speech recognition, computer vision, machine learning, and recommender systems. For such an approximate workflow, it is possible to trade accuracy by computing over a subset (usually selected via a sampling mechanism) instead of the entire input dataset. Thereby, data analytics systems based on approximate computation can achieve low latency and efficient utilization of resources. However, the existing systems for approximate computation assume a centralized dataset, where the desired sampling mechanism can be employed. Thus, existing systems are not compatible with the distributed privacy-preserving analytics systems.

Combining privacy-preserving analytics and approximate computing. This paper makes the observation that the two computing paradigms, i.e., privacy-preserving analytics and approximate computation, are complementary. Both paradigms strive for an approximate instead of the exact output, but they differ in their *means* and *goals* for approximation. Privacy-preserving analytics adds explicit *noise* to the aggregate query output to protect user privacy, whereas approximate computation relies on a representative *sampling* of the entire dataset to compute over only a subset of data items to enable low-latency/efficient analytics. Therefore, this work combines these two existing paradigms together in order to leverage the benefits of both.

The high-level idea is to achieve privacy (via approximation) by directly computing over a subset of sampled data items (instead of computing over the entire dataset) and then adding an explicit noise for privacy preservation.

To realize this combination, this paper presents the design of an approximation mechanism that also achieves privacy-preserving goals for stream analytics. This design targets a distributed setting, where users' private data is stored locally on their respective personal devices, and an analyst issues a streaming query for analytics over the distributed private dataset of users. The analyst's streaming query is executed on the users' data periodically (a configurable epoch) and the query results are transmitted to a centralized aggregator via a set of proxies. The analyst interfaces with the aggregator to get the aggregate query output periodically.

Two core techniques are employed to achieve the goal. Firstly, *sampling* [48] is directly employed at the user site for approximate computation, where each user randomly decides whether to participate in answering the query in the current epoch. Since the sampling is employed at the data source, instead of sampling at a centralized infrastructure, the proposed approach can squeeze out the desired data size (by controlling the sampling parameter) from the very first stage in the analytics pipeline, which is essential in low-latency environments.

Secondly, if the user participates in the query answering process, a *randomized response* [31] mechanism is performed to add noise to the query output at the user site, again locally at the source of the data in a decentralized fashion. In particular, each user locally randomizes the truthful answer to the query to achieve the differential privacy guarantees (§3.2.2). Since the noise is added at the source of data, instead of adding the explicit noise to the aggregate output at a trusted aggregator or proxies, the proposed approach enables a truly "synchronization-free" distributed architecture, which requires *no coordination* among proxies and the aggregator for the mandated noise addition.

The last, but not the least, silver bullet of the design: it turns out that the combination of the two aforementioned techniques (i.e., sampling and randomized response) leads us to achieve zero-knowledge privacy [35], a privacy bound tighter than the state-of-the-art differential privacy [27].

To summarize, this paper presents the design of PRIVAPPROX—a practical system for privacy-preserving stream analytics in real time. In particular, the system is a novel combination of the sampling and randomized response techniques, as well as a scalable "synchronization-free" routing scheme which employs a light-weight XOR-based encryption scheme [21]. The resulting system ensures zero-knowledge privacy, anonymization, and unlinkability for users (§2.2).

2 Overview

2.1 System Architecture

PRIVAPPROX is designed for privacy-preserving stream analytics on distributed users' private dataset. This system consists of four main components: clients, proxies, aggregator, and analysts.

Clients locally store users' private data on their respective personal devices, and subscribe to queries from the system. *Analysts* publish streaming queries to the system, and also specify a query execution budget. The query execution budget can either be in the form of latency guarantees/SLAs, output quality/accuracy, or the computing resources for query processing. PRIVAPPROX ensures that the computation remains within the specified budget.

At a high-level, the system works as follows: a query published by an analyst is distributed to clients via the aggregator and proxies. Clients answer the analyst's query locally over the users' private data using a privacy-preserving mechanism. Client answers are transmitted to the aggregator via anonymizing *proxies*. The *aggregator* aggregates received answers from the clients to provide privacy-preserving stream analytics to the analyst.

2.2 System Model

Query model. PRIVAPPROX supports the SQL query language for *analysts* to formulate streaming queries, which are executed periodically at the clients as sliding window computations [9]. While queries can be complex, the results of a query are expressed as counts within histogram buckets, i.e., each bucket represents a range of the query's answer values. Specifically, each query answer is represented in the form of binary buckets, where each bucket stores a value '1' or '0' depending on whether or not the answer falls into the value range represented by that bucket. For example, an analyst can learn the driving speed distribution across all vehicles in San Francisco by formulating an SQL query "`SELECT speed FROM vehicle WHERE location='San Francisco'`". The analyst can then define 12 answer buckets on speed: '0', '1∼10', '11∼20', $\cdots$, '81∼90', '91∼100', and '$>$ 100'. If a vehicle is moving at 15 mph in San Francisco, it answers '1' for the third bucket and '0' for all others.

The query model supports not only numeric queries as described above, but also non-numeric queries. For non-numeric queries, each bucket is specified by a matching rule or a regular expression. Note that, at first glance, the query model may appear simple; however, it has been shown to be effective for a wide-range of analytics algorithms [14, 15].

Threat model. *Analysts* are potentially malicious. They may try to violate the PRIVAPPROX's privacy model (described later), i.e., de-anonymize clients, build pro-

files through the linkage of queries and answers, or remove the added noise from answers.

Clients are potentially malicious. They could generate false or invalid responses to distort the query result for the analyst. However, the proposed system does not defend against the Sybil attack [26], which is beyond the scope of this work [62].

Proxies are also potentially malicious. They may transmit messages between clients and the aggregator in contravention of the system protocols. PRIVAPPROX includes at least two proxies, and there are at least two proxies which do not collude with each other. The *aggregator* is assumed to be honest-but-curious. The aggregator faithfully conforms to the system protocols, but may try to exploit the information about clients. The aggregator does not collude with any proxy nor the analyst.

Privacy model. PRIVAPPROX provides three important privacy properties: *(i)* zero-knowledge privacy, *(ii)* anonymity, and *(iii)* unlinkability.

All aggregate query results in the system are independently produced under the *zero-knowledge privacy* guarantees [35]. The zero-knowledge privacy metric builds upon differential privacy [27], and provides a tighter bound on privacy guarantees compared to differential privacy. Informally, zero-knowledge privacy states that essentially everything that an adversary can learn from the output of an zero-knowledge private mechanism could also be learned using the aggregate information. *Anonymity* means that no system component can associate query answers or query requests with a specific client. Finally, *unlinkability* means that no system component can join any pair of query requests or answers to the same client, even to the same anonymous client. The formal definition, analysis, and proof are described in the technical report [52].

3 Design

PRIVAPPROX consists of two main phases: *submitting queries* and *answering queries*. In the first phase, an analyst submits a query (along with the execution budget) to clients via the aggregator and proxies. In the second phase, the query is answered by the clients in the reverse direction.

3.1 Submitting Queries

To perform statistical analysis over users' private data streams, an analyst creates a query using the query model described in §2.2. In particular, each query consists of the following fields, and is signed by the analyst for non-repudiation:

$$Query := \langle Q_{ID}, SQL, A[n], f, w, \delta \rangle \tag{1}$$

- Q_{ID} denotes a unique identifier of the query. This can be generated by concatenating the identifier of the analyst with a serial number unique to the analyst.
- *SQL* denotes the actual SQL query, which is passed on to clients and executed on their respective personal data.
- $A[n]$ denotes the format of a client's answer to the query. The answer is an n-bit vector where each bit associates with a possible answer value in the form of a "0" or "1" per index (or answer value range).
- f denotes the answer frequency, i.e., how often the query needs to be executed at clients.
- w denotes the window length for sliding window computations [8]. For example, an analyst may only want to aggregate query results for the last ten minutes, which means the window length is ten minutes.
- δ denotes the sliding interval for sliding window computations. For example, an analyst may want to update the query results every one minute, and so the sliding interval is set to one minute.

After forming the query, the analyst sends the query, along with the query execution budget, to the aggregator. Once receiving the pair of the query and query budget from the analyst, the aggregator first converts the query budget into system parameters for sampling and randomization. The system operator can set the sampling fraction using resource prediction model [65, 64, 66] for any given SLA. Hereafter, the aggregator forwards the query and the converted system parameters to clients via proxies.

3.2 Answering Queries

After receiving the query and system parameters, the query is answered by clients and processed by the system to produce the result for the analyst. The query answering process involves four steps including *(i)* sampling at clients for low-latency approximation; *(ii)* randomizing answers for privacy preservation; *(iii)* transmitting answers via proxies for anonymization and unlinkability; and finally, *(iv)* aggregating answers with error estimation to give a confidence level on the approximate result. (The detailed algorithms are covered in [52, 53]).

3.2.1 Step I: Sampling at Clients

PRIVAPPROX makes use of approximate computation to achieve low-latency execution by computing over a subset of data items instead of the entire input dataset. Specifically, the system builds on sampling-based techniques [4, 36, 44, 55, 54]. To keep the private data stored at individual clients, PRIVAPPROX applies an input data sampling mechanism locally at the clients. In particular, the system uses *Simple Random Sampling* (SRS) [48].

Note that, this work assumes that all clients produce the input stream with data items following the same distribution, i.e., all clients' data streams belong to the same stratum. The sampling mechanism can be further extended with the *stratified sampling* technique [44, 55, 54] to deal with varying distributions of data streams. The algorithm and evaluation of stratified sampling are covered in the technical report [52].

3.2.2 Step II: Answering Queries at Clients

Clients that participate in the query answering process make use of the *randomized response* technique [31] to preserve answer privacy, with *no* synchronization among clients. Randomized response works as follows: suppose an analyst sends a query to individuals to obtain the statistical result about a sensitive property. To answer the query, a client locally randomizes its answer to the query [31]. Specifically, the client flips a coin, if it comes up heads, then the client responds its truthful answer; otherwise, the client flips a second coin and responds "Yes" if it comes up heads or "No" if it comes up tails. The privacy is preserved via the ability to refuse responding truthful answers (see details in [53, 52]).

It is worth mentioning that, combining the randomized response with the sampling technique described in Step I, PRIVAPPROX achieves not only differential privacy but also zero-knowledge privacy [35] which is a privacy bound tighter than differential privacy. The detailed proof is described in the technical report [52].

3.2.3 Step III: Transmitting Answers via Proxies

After producing randomized responses, clients transmit them to the aggregator via the proxies. To achieve anonymity and unlinkability of the clients against the aggregator and analysts, PRIVAPPROX utilizes the XOR-based encryption together with source rewriting, which has been used for anonymous communications [22, 21, 57, 25]. The underlying idea of this encryption is simple: if Alice wants to send a message M of length l to Bob, then Alice and Bob share a secret M_K (in the form of a random bit-string of length l). To transmit the message M privately, Alice sends an encrypted message '$M_E = M \oplus M_K$' to Bob, where '$\oplus$' denotes the bit-wise XOR operation. To decrypt the message, Bob again uses the bit-wise XOR operation: $M = M_E \oplus M_K$ (see details in [53, 52]).

3.2.4 Step IV: Generating Result at the Aggregator

At the aggregator, all data streams ($\langle M_{ID}, M_E \rangle$ and $\langle M_{ID}, M_{K_i} \rangle$) are received, and can be joined together to obtain a unified data stream. Specifically, the associated M_E and M_{K_i} are paired by using the message identifier M_{ID}. To decrypt the original randomized message M from the client, the XOR operation is performed over M_E

and M_K: $M = M_E \oplus M_K$ with M_K being the XOR of all M_{K_i}: $M_K = \bigoplus_{i=2}^{n} M_{K_i}$. As the aggregator cannot identify which of the received messages is M_E, it just XORs all the n received messages to decrypt M.

Note that an adversarial client might answer a query many times in an attempt to distort the query result. However, this problem can be handled, for example, by applying the *triple splitting* technique [21].

Error bound estimation. PRIVAPPROX provides an error bound estimation for the aggregate query results. The accuracy loss in PRIVAPPROX is caused by two processes: *(i)* sampling and *(ii)* randomized response. Since the accuracy loss of these two processes is statistically independent (see details in [53, 52]), PRIVAPPROX estimates the accuracy loss of each process separately. In addition, independent of the error induced by randomized response, the error coming from sampling is simply being added upon. Following this, the system sums up both independently estimated errors to provide the total error bound of the query results. To estimate the accuracy loss of the randomized response process, PRIVAPPROX makes use of an experimental method. It performs several micro-benchmarks at the beginning of the query answering process (without performing the sampling process) to estimate the accuracy loss caused by randomized response. On the other hand, to estimate the accuracy loss of the sampling process, PRIVAPPROX applies the statistical theory of the sampling techniques (see details in [53, 52]).

4 Related Work

Privacy-preserving analytics. Since the notion of differential privacy [27, 29], a plethora of systems have been proposed to provide differential privacy with centralized databases [46, 51, 47, 40]. In practice, however, such central trust can be abused, leaked, or subpoenaed [23, 50, 42, 58].

To overcome the limitations of the centralized database schemes, recently a flurry of systems have been proposed with a focus on preserving user privacy (mostly, differential privacy) in a distributed setting where the private data is kept locally [37, 22, 49, 21, 45, 61, 38, 60, 28, 41, 5]. However, these systems are designed to deal with the "one-shot" batch queries only, whereby the data is assumed to be static.

To overcome the limitations of the aforementioned systems, several differentially private stream analytics systems have been proposed [30, 18, 17, 59, 56, 32, 39]. Unfortunately, these systems still contain several technical shortcomings that limit their practicality. One of the first systems [30] updates the query result only if the user's private data changes significantly, and does not support stream analytics over an unlimited time period. Subsequent systems [18, 39] remove the limit on the time period, but introduce extra system overheads. Some systems [59, 56] leverage expensive secret sharing cryptographic operations to produce noisy aggregate query results. These systems, however, cannot work at large scale under churn; moreover, in these systems, even a single malicious user can substantially distort the aggregate results without detection. Recently, some other privacy-preserving distributed

stream monitoring systems have been proposed [32, 17]. However, they all require some form of synchronization, and are tailored for heavy-hitter monitoring only. Streaming data publishing systems like [63] use a stream-privacy metric at the cost of relying on a trusted party to add noise. In contrast, PRIVAPPROX does not require a trusted proxy or aggregator to add noise. Furthermore, PRIVAPPROX provides stronger privacy properties (i.e., zero-knowledge privacy).

Sampling and randomized response. Sampling and randomized response, also known as input perturbation techniques, are being studied in the context of privacy-preserving analytics, albeit they are explored separately. For instance, the relationship between sampling and privacy is being investigated to provide k-anonymity [19], differential privacy [47], and crowd-blending privacy [34]. In contrast, this paper shows that sampling combined with randomized response achieves the zero-knowledge privacy, a privacy bound strictly stronger than the state-of-the-art differential privacy.

Approximate computation. Approximation techniques such as sampling [6, 33, 20], sketches [24], and online aggregation [43] have been well-studied over the decades in the databases community. Recently, sampling-based systems [36, 4, 55, 54, 44] have also been shown effective for "Big Data" analytics. In particular, our work builds on IncApprox [44], a data analytics system that combines incremental computation [13, 10, 11, 12, 7] and approximate computation. However, PRIVAPPROX differs in two main aspects. First, the system performs sampling in a distributed way as opposed to sampling in a centralized dataset. Second, PRIVAPPROX extends sampling with randomized response for privacy-preserving analytics.

5 Conclusion

This paper presents PRIVAPPROX, a privacy-preserving stream analytics system. The approach in PRIVAPPROX builds on the observation that both computing paradigms — privacy-preserving data analytics and approximate computation — strive for approximation, and can be combined together to leverage the benefits of both.

References

1. Apache S4. http://incubator.apache.org/s4. Accessed: Nov, 2017.
2. Apache Spark Streaming. http://spark.apache.org/streaming. Accessed: Nov, 2017.
3. Apache Storm. http://storm-project.net/. Accessed: Nov, 2017.
4. S. Agarwal, B. Mozafari, A. Panda, H. Milner, S. Madden, and I. Stoica. BlinkDB: Queries with Bounded Errors and Bounded Response Times on Very Large Data. In *Proceedings of the ACM European Conference on Computer Systems (EuroSys)*, 2013.

5. I. E. Akkus, R. Chen, M. Hardt, P. Francis, and J. Gehrke. Non-tracking web analytics. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2012.
6. M. Al-Kateb and B. S. Lee. Stratified Reservoir Sampling over Heterogeneous Data Streams. In *Proceedings of the 22nd International Conference on Scientific and Statistical Database Management (SSDBM)*, 2010.
7. P. Bhatotia. *Incremental Parallel and Distributed Systems*. PhD thesis, Max Planck Institute for Software Systems (MPI-SWS), 2015.
8. P. Bhatotia, U. A. Acar, F. P. Junqueira, and R. Rodrigues. Slider: Incremental Sliding Window Analytics. In *Proceedings of the 15th International Middleware Conference (Middleware)*, 2014.
9. P. Bhatotia, M. Dischinger, R. Rodrigues, and U. A. Acar. Slider: Incremental Sliding-Window Computations for Large-Scale Data Analysis. Technical Report MPI-SWS-2012-004, MPI-SWS, 2012. http://www.mpi-sws.org/tr/2012-004.pdf.
10. P. Bhatotia, P. Fonseca, U. A. Acar, B. Brandenburg, and R. Rodrigues. iThreads: A Threading Library for Parallel Incremental Computation. In *Proceedings of the 20th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2015.
11. P. Bhatotia, R. Rodrigues, and A. Verma. Shredder: GPU-Accelerated Incremental Storage and Computation. In *Proceedings of USENIX Conference on File and Storage Technologies (FAST)*, 2012.
12. P. Bhatotia, A. Wieder, I. E. Akkus, R. Rodrigues, and U. A. Acar. Large-scale incremental data processing with change propagation. In *Proceedings of the Conference on Hot Topics in Cloud Computing (HotCloud)*, 2011.
13. P. Bhatotia, A. Wieder, R. Rodrigues, U. A. Acar, and R. Pasquini. Incoop: MapReduce for Incremental Computations. In *Proceedings of the ACM Symposium on Cloud Computing (SoCC)*, 2011.
14. A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: the SuLQ framework. In *Proceedings of the ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, 2005.
15. A. Blum, K. Ligett, and A. Roth. A Learning Theory Approach to Non-interactive Database Privacy. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, 2008.
16. P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, and K. Tzoumas. Apache flink: Stream and batch processing in a single engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 2015.
17. T.-H. H. Chan, M. Li, E. Shi, and W. Xu. Differentially Private Continual Monitoring of Heavy Hitters from Distributed Streams. In *Proceedings of the 12th International Conference on Privacy Enhancing Technologies (PETS)*, 2012.
18. T.-H. H. Chan, E. Shi, and D. Song. Private and Continual Release of Statistics. *ACM Trans. Inf. Syst. Secur.*, 2011.
19. K. Chaudhuri and N. Mishra. When Random Sampling Preserves Privacy. In *Proceedings of the 26th Annual International Conference on Advances in Cryptology (CRYPTO)*, 2006.
20. S. Chaudhuri, G. Das, and V. Narasayya. Optimized Stratified Sampling for Approximate Query Processing. *Proceedings of ACM Transaction of Database Systems (TODS)*, 2007.
21. R. Chen, I. E. Akkus, and P. Francis. SplitX: High-performance Private Analytics. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, 2013.
22. R. Chen, A. Reznichenko, P. Francis, and J. Gehrke. Towards Statistical Queries over Distributed Private User Data. In *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2012.
23. ComScore Reaches $14 Million Settlement in Electronic Privacy Class Action. http://www.alstonprivacy.com/comscore-reaches-14-million-settlement-in-electronic-privacy-class-action/. Accessed: Nov, 2017.
24. G. Cormode, M. Garofalakis, P. J. Haas, and C. Jermaine. Synopses for Massive Data: Samples, Histograms, Wavelets, Sketches. *Found. Trends databases*, 2012.

25. R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. Technical report, DTIC Document, 2004.
26. J. R. Douceur. The Sybil Attack. In *Proceedings of 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
27. C. Dwork. Differential privacy. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming, part II (ICALP)*, 2006.
28. C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In *Proceedings of the 24th Annual International Conference on The Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, 2006.
29. C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating Noise to Sensitivity in Private Data Analysis. In *Proceedings of the Third conference on Theory of Cryptography (TCC)*, 2006.
30. C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, 2010.
31. J. A. Fox and P. E. Tracy. *Randomized response: a method for sensitive surveys*. Beverly Hills California Sage Publications, 1986.
32. A. Friedman, I. Sharfman, D. Keren, and A. Schuster. Privacy-Preserving Distributed Stream Monitoring. In *Proceedings of the Symposium on Network and Distributed System Security (NDSS)*, 2014.
33. M. N. Garofalakis and P. B. Gibbon. Approximate Query Processing: Taming the TeraBytes. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 2001.
34. J. Gehrke, M. Hay, E. Lui, and R. Pass. Crowd-blending privacy. In *Proceedings of the 32nd Annual International Conference on Advances in Cryptology (CRYPTO)*, 2012.
35. J. Gehrke, E. Lui, and R. Pass. Towards Privacy for Social Networks: A Zero-Knowledge Based Definition of Privacy. In *Theory of Cryptography*, 2011.
36. I. Goiri, R. Bianchini, S. Nagarakatte, and T. D. Nguyen. ApproxHadoop: Bringing Approximations to MapReduce Frameworks. In *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2015.
37. S. Guha, B. Cheng, and P. Francis. Privad: Practical Privacy in Online Advertising. In *Proceedings of the 8th Symposium on Networked Systems Design and Implementation (NSDI)*, 2011.
38. S. Guha, M. Jain, and V. N. Padmanabhan. Koi: A location-privacy platform for smartphone apps. In *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2012.
39. T. h. Hubert Chan, E. Shi, and D. Song. Privacy-preserving stream aggregation with fault tolerance. In *Proceedings of 16th International Conference on Financial Cryptography and Data Security (FC)*, 2012.
40. A. Haeberlen, B. C. Pierce, and A. Narayan. Differential Privacy Under Fire. In *Proceedings of the 20th USENIX Security Symposium (USENIX Security)*, 2011.
41. M. Hardt and S. Nath. Privacy-aware personalization for mobile advertising. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS)*, 2012.
42. HealthCare.gov Sends Personal Data to Dozens of Tracking Websites. https://www.eff.org/deeplinks/2015/01/healthcare.gov-sends-personal-data. Accessed: Nov, 2017.
43. J. M. Hellerstein, P. J. Haas, and H. J. Wang. Online Aggregation. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1997.
44. D. R. Krishnan, D. L. Quoc, P. Bhatotia, C. Fetzer, and R. Rodrigues. IncApprox: A Data Analytics System for Incremental Approximate Computing. In *Proceedings of the 25th International Conference on World Wide Web (WWW)*, 2016.
45. S. Lee, E. L. Wong, D. Goel, M. Dahlin, and V. Shmatikov. πBox: A Platform for Privacy-Preserving Apps. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2013.

46. F. McSherry and R. Mahajan. Differentially-private Network Trace Analysis. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, 2010.
47. P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler. GUPT: Privacy Preserving Data Analysis Made Easy. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 2012.
48. D. S. Moore. *The Basic Practice of Statistics*. W. H. Freeman & Co., 2nd edition, 1999.
49. A. Narayan and A. Haeberlen. DJoin: Differentially Private Join Queries over Distributed Databases. In *Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation (OSDI)*, 2012.
50. Privacy Lawsuit Targets Net Giants Over 'Zombie' Cookies. http://www.wired.com/2010/07/zombie-cookies-lawsuit. Accessed: Nov, 2017.
51. D. Proserpio, S. Goldberg, and F. McSherry. Calibrating Data to Sensitivity in Private Data Analysis: A Platform for Differentially-private Analysis of Weighted Datasets. *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 2014.
52. D. L. Quoc, M. Beck, P. Bhatotia, R. Chen, C. Fetzer, and T. Strufe. Privacy preserving stream analytics: The marriage of randomized response and approximate computing. https://arxiv.org/abs/1701.05403, 2017.
53. D. L. Quoc, M. Beck, P. Bhatotia, R. Chen, C. Fetzer, and T. Strufe. PrivApprox: Privacy-Preserving Stream Analytics. In *Proceedings of the 2017 USENIX Conference on USENIX Annual Technical Conference (USENIX ATC)*, 2017.
54. D. L. Quoc, R. Chen, P. Bhatotia, C. Fetzer, V. Hilt, and T. Strufe. Approximate Stream Analytics in Apache Flink and Apache Spark Streaming. *CoRR*, abs/1709.02946, 2017.
55. D. L. Quoc, R. Chen, P. Bhatotia, C. Fetzer, V. Hilt, and T. Strufe. StreamApprox: Approximate Computing for Stream Analytics. In *Proceedings of the International Middleware Conference (Middleware)*, 2017.
56. V. Rastogi and S. Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *Proceedings of the International Conference on Management of Data (SIGMOD)*, 2010.
57. M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, 1998.
58. SEC Charges Two Employees of a Credit Card Company with Insider Trading. http://www.sec.gov/litigation/litreleases/2015/lr23179.htm. Accessed: Nov, 2017.
59. E. Shi, T. H. Chan, E. G. Rieffel, R. Chow, and D. Song. Privacy-Preserving Aggregation of Time-Series Data. In *Proceedings of the Symposium on Network and Distributed System Security (NDSS)*, 2011.
60. K. Singh, S. Bhola, and W. Lee. xbook: Redesigning privacy control in social networking platforms. In *Proceedings of the 18th Conference on USENIX Security Symposium (USENIX Security)*, 2009.
61. B. Viswanath, E. Kiciman, and S. Saroiu. Keeping Information Safe from Social Networking Apps. In *Proceedings of the ACM SIGCOMM Workshop on Social Networks (WOSN'12)*, 2012.
62. G. Wang, B. Wang, T. Wang, A. Nika, H. Zheng, and B. Y. Zhao. Defending against sybil devices in crowdsourced mapping services. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2016.
63. Q. Wang, Y. Zhang, X. Lu, Z. Wang, Z. Qin, and K. Ren. RescueDP: Real-time Spatio-temporal Crowd-sourced Data Publishing with Differential Privacy. In *Proceedings of the 35th Annual IEEE International Conference on Computer Communications (INFOCOM)*, 2016.
64. A. Wieder, P. Bhatotia, A. Post, and R. Rodrigues. Brief Announcement: Modelling MapReduce for Optimal Execution in the Cloud. In *Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of Distributed Computing (PODC)*, 2010.
65. A. Wieder, P. Bhatotia, A. Post, and R. Rodrigues. Conductor: Orchestrating the Clouds. In *Proceedings of the 4th international workshop on Large Scale Distributed Systems and Middleware (LADIS)*, 2010.

66. A. Wieder, P. Bhatotia, A. Post, and R. Rodrigues. Orchestrating the Deployment of Computations in the Cloud with Conductor. In *Proceedings of the 9th USENIX symposium on Networked Systems Design and Implementation (NSDI)*, 2012.