A runtime sharing mechanism for Big Data platforms

Mark Shtern, Marin Litoiu
York University, Ontario, Canada
mark, mlitoiu@yorku.ca

***Abstract*—In order to extract value from Big Data, a data source provider has to share data among many consumers. As such, data sharing becomes an important feature of Big Data platforms. However, privacy concerns are the key obstacles that prevent organizations from implementing data sharing solutions. Moreover, currently, the data owner is responsible for preparing the data before releasing it to a 3rd party. The preparation of data for release is a complex task and can become an obstacle. In this paper, we propose an ecosystem which enables data sharing responsibilities among producers and consumers and mitigates some of the obstacles.**

***Keywords*-public cloud, big data**

I. Introduction

Today, we are living in an era of Big Data [1], where 90% of the data in the world has come into existence since 2010[1]. Many big data applications are being developed through a collaboration between data providers and analytics providers. For instance, IBM reported that mortality decreased by 20% when hospital patient data was analyzed by IBM platform [2]. Another example is the Walmart product Shoppycat [3], that recommends products to Facebook users based on the hobbies and interests of their friends. All these examples require the integration between data provider and data consumer applications. To facilitate the ecosystem data provider-consumer, large data providers need to develop secure mechanisms for enabling access to their data. Sharing data is a challenging problem that is extensively explored in many research works [2], [3], [4]. One of the big obstacles in sharing data is privacy and this is the focus of this paper.

The development of mechanisms for protecting data privacy has been addressed by many researchers [5], [6], [7], [8], [9]. As a result, there are many techniques for data anonymization [6], [8], [10], [11]. Nevertheless, companies such as Netflix [12], Massachusetts Group Insurance Commission[6], failed to produce databases which were compliant with privacy policies. The privacy becomes more complex in big data contexts due to the large amount of data that is unstructured or semi-structured. Moreover, the data owner may not have sufficient knowledge about the sensitivity of data stored on its servers. Finally, big data has massive volumes and high speed and because typical analytics algorithms do not require all data [13], it means that structuring and anonymizing all existing data may lead to waste of resources and budget.

[1]http://www.viawest.com/sites/default/files/asset/document/ViaWest_IT_Infirmity_Infographic.pdf

[2]http://www-01.ibm.com/software/data/bigdata/industry.html

[3]http://www.bigdata-startups.com/BigData-startup/walmart-making-big-data-part-dna/

In this paper we propose a new method, supported by tools, for privacy management on big data platforms. The data provider will delegate the preprocessing of data, including the anonymization algorithms to the analyst. The data provider responsibility is to verify that data is pre-processed and sufficiently anonymized before the analyst get access to the result of the analytics algorithm. We assume that the data providers are more willing to share their data when the anonymization is delegated to a 3rd party because the anonimization itself is a more complex and costly problem that some sort of verification that data set meets some specific anonymization criteria. For instance, to construct a k-anonymous data set with minimum suppressing information is a NP-hard problem[5], however, to verify that a data is k-anonymous is a trivial and polinomial problem.

The main advantage of the new method is its flexibility and mining efficiency: only the analysts have sufficient information about the structure of the data they need, and they know how to anonymize data set and still get meaningful results. The data producer will verify that the pre-processing and anonymization process proposed by the analyst is compliant with the privacy or other policies. Other advantage of the proposed approach is that it avoids the construction of special anonymized data sets before allowing access. This will improve storage utilization (no need to creating storage-intensive stale data sets) and simplify the maintenance of anonymized data sets (such as synchronization with updated data and construction of anonymized data sets for unused data). The final advantage of the proposed methodology is the creation of anonymized data sets on demand and only for the required data for the specific analytic task.

The paper presents the high level architecture of the privacy aware ecosystem and the components that support the ecosystem. The remainder of the paper is organized as follows: Section II presents the overall architecture and the components of the ecosystem; conclusions are presented in Section III.

II. Methodology

In this section, we propose a methodology for developing a big data ecosystem that enforces privacy policies on data analytics workloads. We assume the following roles:

- `Analyst` is responsible for developing and submitting data mining algorithms.
- `Data Provider or Provider` is responsible for providing the big data platform and the data.

A typical big data analytics process performed by the Analyst includes a data preparation phase [14]. The objective of data preparation phase is to prepare data for data mining

algorithms. During this phase, the input data is preprocessed to extract tuples (the assumption is that the original data is unstructured), to reduce noise and handle missing values (data cleansing), then to remove the irrelevant or redundant attributes (relevance analysis) and finally to generalize or normalize data (data transformation) [14].

We propose to extend the data preparation phase by including an anonymization step. In this step, the Analysts will provide the anonymization method suitable for their analytics workload. To prevent breaches, the Provider will monitor whether the Analyst complies with its privacy policies. To enable the Provider to monitor the anonymization process, the Analyst provides the preparation algorithm as a separate process/job in a domain specific language (DSL). The DSL helps to reduce the complexity of privacy compliance verification process. When the Analyst defines the data preparation process using the DSL, it also specifies a schema of extracted facts. In other words, for each attribute it will specify its semantic, such as city, name, SIN etc. The schema definition is similar to relational database schema and is defined for the output of data cleansing phase. The data preparation job expressed in DSL can be checked for compliance without actually running the job, by performing a static analysis. Only in the case where the static analysis does not detect breaches, the Provider will run the DSL transformation on the actual data to detect if it causes a violation of privacy policies. The Provider is also responsible to verify that schema aligns with underline data. The key properties of DSL will be discussed in Section II-B.

To reduce the risk that the automatic private policy verification process fails to catch leakage of private information, the data preparation process will run first on a subset of data (which we call test dataset) that contains all previously identified private information. In case the system detects that the anonymization process fails on the test dataset, it stops the execution of the data mining process.

Since the verification of privacy compliance can be done in parallel with the execution of data mining algorithm and because the big data jobs usually run for long time, the verification process should not introduce a significant delay in the overall process.

Often, data mining jobs require mixing data from different sources. In such cases, several data preparation jobs need to be created. The system will validate each data preparation process in sequence. This strategy will protect against dataset linkage attacks [15] even if it increases complexity. In [16] researchers demonstrate the feasibility to verify data anonymization between several views, which is similar to our case.

Next, in Section II-A, we present the architecture for the proposed ecosystem.The main components are discussed in Sections II-B and II-C.

A. High-level Architecture

The privacy ecosystem high-level architecture consists of a set of components that include a privacy aware big data platform and privacy protection mechanisms.

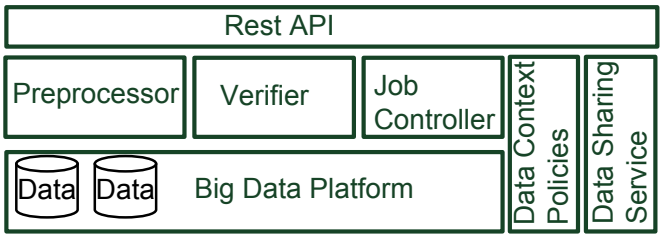

Fig. 1: The logical components of the privacy preserved ecosystem architecture.

Figure 1 shows an overview of the high-level architecture. The components of the architecture are `Rest API`, `Preprocessor`, `Verifier`, `Job Controller`, `Big Data Platform`, `Data Context Policies` and `Data Sharing Service`.

- `Rest API` is a restful API that allows the Analyst to submit analytic jobs together with a corresponding data preparation job. The Analyst can track the job progress and get the result of the data mining algorithm using this API. The system should ensure that there is no other access point to data or big data platform other than this API.
- `Preprocessor` is responsible for transforming the original data into anonymized data using the transformation defined in the DSL language program. It is invoked after Verifier validates the DSL using static analysis and augments the transformation to include supplementary information (see Verifier). During the transformation process, the Preprocessor sends the produced dataset (including supplementary data) to Verifier and then to the data mining algorithm.
- `Verifier` performs the static analysis of the DSL program to verify that DSL transformation produces a data set aligned with data context policies. Depending on the underline policies, it may modify the DSL program to attach additional transformations to comply with the polices. Verifier is also responsible for validating that DSL correctly defines extracted facts from input dataset. The Verifier runs in either streaming and batch data processing style and can run in parallel with the data mining algorithm.
- `Job Controller` is responsible for coordinating different components of the system. It is also responsible for monitoring the job execution, scheduling execution of data processing tasks on Preprocessor and scheduling the verification tasks upon the completion of data preparation process. It also feeds output data from Preprocessor to corresponding data mining algorithm. In addition, the Job Controller is responsible to schedule data preparation process on the test dataset for verification of privacy policies. To achieve this, the Job Controller should have a tied integration with Data Sharing Service. We have verified the feasibility of such integration on DaaSPatcher

system [2].

- `Big Data Platform` provides both access to stored data and to distributed processing. For instance, Hadoop ecosystem is a popular example of big data platform [17].
- `Data Context Policies` is a service that manages privacy and access policies on specific data types (e.g. SIN, names, Age etc.) and is Analyst's group or Analyst' attribute specific. For instance, the access policies may require that a user may have access only to cities and movies. Or that data mining algorithm should comply with 10-anonymity. XCAML [4] is a flexible approach for defining such data context polices. The data Provider may want to configure additional access control policies using data sharing facilities. Enforcement of such data sharing policies is outside the scope of this paper.
- `Data Sharing Service` is responsible for enabling fine-grained control over what data is shared. It enables analytics tasks to run on the infrastructure near the actual data. The data sharing service also provides services for authorization and authentication of users. DaaSPatcher [2] is an example of such data sharing service.

The system automatically stores all submitted DSL transformations for future auditing. In addition, approved DSL transformations can be used for constructing and improving test datasets due to the fact that DSL transformations contain information about the type of extracted data needed by Analysts. In the section II-D we discuss the details about constructing test datasets.

To prevent unauthorized access to sensitive data, the ecosystem should deploy a safeguard which prevents 3rd party code such as data mining job or data preparation process to send data directly to Analyst using network communication channels.

In next section, we discuss the Preprocessor and the DSL.

B. Preprocessor and DSL

Preprocessor is a data parser and filtering component. The input for Preprocessor is a stream of unstructured data and a transformation specified using DSL. The output is a stream of tuples. When one pass on data is sufficient for implementing the privacy protection, then Preprocessor could follow a streaming paradigm. When streaming is used, the typical Preprocessor data flow is to read one input record, parse it, transform it and in parallel send Verifier all intermediate and final records. Sometimes, this process may be insufficient to meet privacy goals and in such cases a second pass over data is required.

Preprocessor ability to satisfy Analyst's data preparation needs depends on the flexibility and expressivity of DSL. At the same time, in order for Verifier to effectively evaluate the correctness of data transformation and to limit the vector of possible attacks (such as encrypting data or sending it over network), the language should be simple and limited. We have identified the following requirements for DSL language:

- the ability to specify the beginning and end of every phase of the transformations such as data parsing, anonymization, etc.
- the ability to specify the schema of extracted tuples and to specify how tuples will be anonymized.
- the ability to specify additional information required by Verifier in a programmatic way.
- include high-level abstraction for simplification of the anonymization process.

We envision DSL language as mix declarative style for defining schema and procedural style for specifying how and what information to extract from the unstructured data.

C. Verifier

Verifier is a key component of the ecosystem. It is responsible for validating the compliance of both DSL and dataset with the Provider policies. The Provider has two ways to address a violation of policies. The first one is to cancel a job when the first violation is discovered. Such approach may not be practical in all cases due to large volume of data and because not all policies require cancelling. An alternative approach to filter data which violates the policies might be more practical in some cases. The proposed system can accommodate both approaches for general policy violation.

Verifier consists of several independent components such as `DSL verifier and enhancer`, `Schema verifier` and `Anonymization verifier`.

`DSL verifier and enhancer` is a static analyzer that attempts to discover non-compliance with provider polices. In addition, this component is responsible for modifying the transformation script to include additional information and steps to allow verification of privacy policies.

`Schema verifier` component validates data compliance with schema on each step (such as parsing, filtering, generalization) of transformation. It may be part of Verifier or part of Preprocessor process (In such scenario, verification happens immediate after data cleaning step). There is a deacrease of network traffic when schema verifier is included into data preparation process. This also allows the filtering of data fields that are not compliant with schema. Since schema verifier checks whether actual data complies with specific required data type, the Provider should develop rules to verify this. Many verification rules can be developed using open source database such as WorDnet [5], Freebase [6]. Since schema verifier may require a significant time for verification between data and schema, to avoid delays, schema verifier can run outside of the Preprocessor process.

The final component is the `Anonymization verifier` that can be deployed as separate process or part of the final step of the Preprocessor process. Anonymization verifier performs the following actions:

- ensure that data parsing step (extraction of tuples from unstructured/semi-structured data) from data preparation

[4]https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

[5]http://wordnet.princeton.edu

[6]http://en.wikipedia.org/wiki/Freebase

process does not modify the original data. This test mitigates some sort of remapping/encoding attacks, where private data will be encoded using non-private data.

- verify whether the constructed dataset meets privacy policies. The test is dependent on the required anonymization methodology. In case of k-anonymity, for example, verify that tuples for each person contained in the anonymized dataset cannot be distinguished from at least k-1 individuals whose tuples also appear in the anonymized dataset. When data-mining algorithm consumes data from different data sources then the Verifier will verify the anonimization based on the composition of the extracted information from different sources. Therefore, this ecosystem can be used in federation with other similar ecosystems.

An additional step to protect against the leakage of private information is the assessment of data preparation process on test dataset. During such assessment, Verifier will check if any part of private information appears in the elements of constructed tuples. In the next section we will discuss the test dataset.

D. Test dataset

Analysts are obligated to specify all personal information that they extract. To verify this and ensure that the transformation process was correct, the ecosystem will run the data preparation process together with the Verification process on a test dataset, which is a subset of original dataset. For each test dataset, there is a meta-data that includes information about personal identification fields and known attributes and their types. When Verifier has both meta-data and dataset constructed after preprocessing, it can better validate the anonymization and whether the Analyst correctly specifies identifiable information and correlation between schema and the dataset.

III. Conclusions and Further Work

In this work, we have proposed a big data ecosystem that facilitates privacy aware data mining algorithms. We proposed a platform where both the Provider and the Analyst share the responsibility of addressing pre-processing and privacy concerns. In addition, the proposed ecosystem helps improve the results of data mining algorithms by ensuring that the input data is properly aligned with the assumptions of the algorithms.

Acknowledgment

This research was supported by the SAVI Strategic Research Network (Smart Applications on Virtual Infrastructure), funded by NSERC (The Natural Sciences and Engineering Research Council of Canada) and by Connected Vehicles and Smart Transportation(CVST) funded Ontario Research Fund.

References

[1] A. Labrinidis and H. Jagadish, "Challenges and opportunities with big data," *Proceedings of the VLDB Endowment*, vol. 5, no. 12, pp. 2032–2033, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2367572

[2] M. Shtern, B. Simmons, M. Smit, and M. Litoiu, "Toward an ecosystem for precision sharing of segmented big data," in *Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference on*. IEEE, 2013, pp. 335–342. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6676712

[3] I. Roy, S. T. Setty, A. Kilzer, V. Shmatikov, and E. Witchel, "Airavat: Security and privacy for mapreduce." in *NSDI*, vol. 10, 2010, pp. 297–312.

[4] S. Saklikar, "Embedding security and trust primitives within map reduce," http://www.emc-china.com/rsaconference/2012/en/download.php?pdf_file=TC-2003_EN.pdf, 2012.

[5] A. Meyerson and R. Williams, "On the complexity of optimal k-anonymity," in *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2004, pp. 223–228. [Online]. Available: http://dl.acm.org/citation.cfm?id=1055591

[6] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S0218488502001648

[7] C. Dwork, "Differential privacy: A survey of results," in *Theory and Applications of Models of Computation*. Springer, 2008, pp. 1–19. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-540-79228-4_1

[8] ——, "Differential privacy," in *Automata, languages and programming*. Springer, 2006, pp. 1–12. [Online]. Available: http://link.springer.com/chapter/10.1007/11787006_1

[9] B. Fung, K. Wang, R. Chen, and P. S. Yu, "Privacy-preserving data publishing: A survey of recent developments," *ACM Computing Surveys (CSUR)*, vol. 42, no. 4, p. 14, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1749605

[10] M. E. Nergiz, C. Clifton, and A. E. Nergiz, "Multirelational k-anonymity," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 21, no. 8, pp. 1104–1117, 2009. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4653492

[11] A. Friedman, R. Wolff, and A. Schuster, "Providing k-anonymity in data mining," *The VLDB Journal*, vol. 17, no. 4, pp. 789–804, 2008. [Online]. Available: http://link.springer.com/article/10.1007/s00778-006-0039-5

[12] F. McSherry and I. Mironov, "Differentially private recommender systems: building privacy into the netflix prize contenders," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 627–636. [Online]. Available: http://dl.acm.org/citation.cfm?id=1557090

[13] R. Appuswamy, C. Gkantsidis, D. Narayanan, O. Hodson, and A. Rowstron, "Scale-up vs scale-out for hadoop: Time to rethink?" in *Proceedings of the 4th annual Symposium on Cloud Computing*. ACM, 2013, p. 20. [Online]. Available: http://dl.acm.org/citation.cfm?id=2523629

[14] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*. Morgan kaufmann, 2006.

[15] M. M. Merener, "Theoretical results on de-anonymization via linkage attacks," *Transactions on Data Privacy*, vol. 5, no. 2, pp. 377–402, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2423652

[16] C. Yao, X. S. Wang, and S. Jajodia, "Checking for k-anonymity violation by views," in *Proceedings of the 31st international conference on Very large data bases*. VLDB Endowment, 2005, pp. 910–921. [Online]. Available: http://dl.acm.org/citation.cfm?id=1083697

[17] J. Y. Monteith, J. D. McGregor, and J. Ingram, "Hadoop and its evolving ecosystem," in *Proceedings of the Fifth International Workshop on Software Ecosystems*, 2013.