Access Control in the Era of Big Data: State of the Art and Research Directions

Pietro Colombo
DiSTA, University of Insubria
Varese, Italy
pietro.colombo@uninsubria.it

Elena Ferrari
DiSTA, University of Insubria
Varese, Italy
elena.ferrari@uninsubria.it

ABSTRACT

Data security and privacy issues are magnified by the volume, the variety, and the velocity of Big Data and by the lack, up to now, of a standard data model and related data manipulation language. In this paper, we focus on one of the key data security services, that is, access control, by highlighting the differences with traditional data management systems and describing a set of requirements that any access control solution for Big Data platforms may fulfill. We then describe the state of the art and discuss open research issues.

CCS CONCEPTS

• **Security and privacy → Access control**;

KEYWORDS

Big Data; Access control; Privacy; NoSQL Data Management Systems

ACM Reference format:
Pietro Colombo and Elena Ferrari. 2018. Access Control in the Era of Big Data: State of the Art and Research Directions. In *Proceedings of The 23rd ACM Symposium on Access Control Models & Technologies (SACMAT), Indianapolis, IN, USA, June 13–15, 2018 (SACMAT '18),* 8 pages.
https://doi.org/10.1145/3205977.3205998

1 INTRODUCTION

The last years have seen changes related to the organization of business models and work styles, caused by the rapid evolution of data analysis and data management systems. Business strategies are more and more driven by the integrated analysis of huge volumes of heterogeneous data, coming from different sources (e.g., social media, IoT devices). Data have become a key factor for making business decisions: we have entered the Big Data era [32].

The term Big Data refers to a phenomenon characterized by "5 V": starting from datasets collecting huge Volumes of data with a high Variety of formats, Big Data analytic platforms allow one to make predictions with high Velocity, thus, in a timely manner, low Veracity, therefore with low uncertainties, and with a high Value, namely, with an expected significant gain [27]. The phenomenon has been pushed by numerous technological advancements. The most significant include: the birth of NoSQL datastores [7], that is, modern data management systems which, by means of innovative data models, provide highly efficient storage and analysis services for structured, unstructured, and semi structured data, such as transactions, electronic documents and emails; and distributed computational paradigms, like MapReduce [18], which have opened the way to the systematic analysis of semi-structured and unstructured data.

Overall, the support provided by Big Data platforms for the storage and analysis of huge and heterogeneous datasets cannot find a counterpart within traditional data management systems. In addition, the advantages of these new systems are not only related to the outstanding flexibility and efficacy of the analysis services, as Big Data platforms outperform traditional systems even with respect to performance and scalability. However, the optimization of these aspects goes to the detriment of data protection. As a matter of fact, for what data privacy and security are concerned, in contrast with traditional systems, for which a variety of data protection framework exist (e.g., see [1, 6, 10–12, 21]), the majority of Big Data platforms integrate quite basic access control enforcement mechanisms [13]. The unconstrained access to high volume of data from multiple data sources, the sensitive and private contents of some data resources, and the advanced analysis and prediction capabilities of Big Data analytic platforms, represent a serious threat. For instance, the analysis capabilities can be exploited to derive correlations between sensitive and personal data. As an example, think about the retail sector, where the analysis of the purchases associated with customer fidelity cards done for marketing purposes, may allow the identification of individuals who suffer from food intolerances. As a consequence, although the potential benefits of Big Data analytics are indisputable, the lack of standard data protection tools open these services to potential attackers.

The definition of proper data protection tools tailored for Big Data platforms is as a very ambitious research challenge. State of the art enforcement techniques proposed for traditional systems cannot be used as they are, or straightforwardly adapted to the Big Data context for manifold reasons, such as the required support for semi structured and unstructured data (Variety), the quantity of data to be protected (Volume), and the very strict performance requirements (Velocity). Therefore, the challenge is protecting privacy and confidentiality while not hindering data analytics and information sharing. Additional aspects contribute to raise the complexity of this goal, such as the variety of data models and data analysis and manipulation languages which are used by Big Data platforms. Indeed, different from RDBMSs, Big Data platforms are characterized by various data models [7], the most notable being the key-value, wide column, and document oriented ones.

SACMAT '18, June 13–15, 2018, Indianapolis, IN, USA

ACM ISBN 978-1-4503-5666-4/18/06…$15.00
https://doi.org/10.1145/3205977.3205998

In this paper, we first identify a set of requirements that any access control solution for Big Data platforms should address (cfr. Section 2). Then, we classify and analyze the related literature (Sections 3, 4, and 5), and discuss key research challenges (Section 6). Finally, we conclude the paper in Section 7.

2 REQUIREMENTS

In this section, we provide an overview of the key requirements behind the definition of an access control mechanism for Big Data platforms.

- **Fine-grained access control**. In terms of features the access control mechanism should support, fine grained access control (FGAC) has been widely recognized as one of the fundamental component for an effective protection of personal and sensitive data (e.g., see [1, 39]). Since data processed by Big Data analytics platforms often refer user personal characteristics, it is important that access control rules can be bound to data at the finest granularity levels. However, the related enforcement mechanisms need to be invented from scratch, as those proposed for traditional systems rely on data referring to known schema, while in the context of Big Data, data are heterogeneous and schemaless.
- **Context management**. Another key aspect that should be considered is the support for context based access constraints, as these allow highly customized access control forms. For instance, they can be used to constrain access to specific time periods or geographical locations. In case contexts are used to derive access control decisions, access authorizations are granted when conditions referring to properties of the environment within which an access request has been issued are satisfied.
- **Efficiency of access control**. The characteristics of the Big Data scenario, such as the distributed nature of the considered platforms, the complexity of the queries, and the focus on performance, require enforcement strategies that do not compromise the usability of the hosting analytic frameworks. Indeed, based on the considered queries, the number of checks to be executed during access control enforcement can match or be even greater than the number of data records, and, in the Big Data scenario, data sets can include up to hundreds of millions of such records. This requires efficient policy compliance mechanisms. FGAC has been enforced in traditional relational DBMSs according to two main approaches. The first is the view-based one, where users are only allowed to access a view of the target dataset that satisfies the specified access control restrictions, whereas the second one is based on query rewriting. Under such an approach instead of pre-computing the authorized views, the query is modified at run-time by injecting restrictions imposed by the specified access control rules. It is therefore important to determine to what extent these approaches are suitable for the Big Data scenario and how they can be possibly customized or extended.

As it should be clear from the previous discussion, one of the main difficulties in developing an access control solution for Big Data platforms is the lack of a standard model and related manipulation languages to which access control rules and the related enforcement monitor can be bound.

3 STATE OF THE ART

In the literature various proposals exist which address the issue of access control for Big Data platforms. They can be classified in two main categories:

- **Platform specific approaches**. Access control solutions under this category are designed for one system only (e.g., MongoDB, Hadoop), and possibly leverage on native access control features of the protected platform. The main advantage of this approach is that the devised access control solution can be optimized for the target system, however, its usability and interoperability is greatly limited.
- **Platform independent approaches**. The approaches falling under this category propose access control solutions which do not target a specific platform only. Existing proposals in this category mainly leverage on recent research efforts that aim at defining a unifying query language for NoSQL datastores (e.g., JSONiq [22] and SQL++ [35]). An approach designed for a unifying query language has the advantage of being more general than platform specific solutions, being applicable to the protection of data sources managed by heterogeneous platforms.

In what follows, we analyze the related literature in view of this classification, then we discuss related research challenges. A summary of the surveyed access control frameworks and of the supported requirements (cfr. Section 2) is shown in Table 1.

4 PLATFORM SPECIFIC APPROACHES

The great majority of access control frameworks targeting Big Data platforms propose enforcement approaches designed on the basis of platform specific features and which can only be used with the platform for which they have been defined.

In the remainder of this section, we analyze platform specific approaches defined for MapReduce-based analytics platforms,[1] and NoSQL datastores, which together cover the majority of existing BigData systems.

4.1 MapReduce systems

MapReduce is a distributed computational paradigm that allows analyzing very large data sets [18]. Within MapReduce systems, data resources are partitioned into multiple chunks of data and distributed in a cluster of commodity hardware nodes. Data are analyzed in parallel by means of MapReduce tasks, characterized by users defined Map and Reduce functions. These tasks operate by first extracting and then manipulating flows of key-value pairs, each modeling a portion of the target data resource. The considered computation paradigm allows processing unstructured and semi-structured data resources.

In [41], a framework denoted GuardMR has been proposed, to enforce fine grained Role-based Access Control (RBAC) [20] within

[1] MapReduce-based analytics platforms are hereafter denoted MapReduce systems for the sake of brevity.

Table 1: Summary of the surveyed access control frameworks

AC framework	Target platform	AC model	Max granularity	Context support	Efficiency
GuardMR[41]	Hadoop	RBAC	K,V pair	No	Medium/High
Vigiles[42]	Hadoop	DAC	K,V pair	No	Medium/High
HeAC[23]	Hadoop	RBAC	K,V pair	No	Not available
K-VAC[29]	Cassandra/ HBase	DAC	Cell	Yes	High
[40]	Cassandra	RBAC	Cell	No	Not available
Mem[15]	MongoDB	RBAC	Document	No	High
ConfinedMem[14]	MongoDB	DAC	Field	Yes	Medium/Low
[16]	All those supporting SQL++	ABAC	Cell/Field	Yes	Medium
[31]	Not clear	ABAC	Cell/Field	Yes	Not available

Hadoop[2], a very popular Big Data analytics platform built on top of MapReduce. GuardMR enforces data protection by filtering, and possibly altering, the key-value pairs derived from a target data resource by a MapReduce task, which are then provided as input to the Map function. Filters are used for generating views of the analyzed resources, from which all contents resulting unauthorized for the subject who requires the execution of the MapReduce task are removed or obfuscated. More precisely, filters specify: i) preconditions to the processing of any key-value pair *p* extracted from a target resource under analysis, as well as ii) the rationale for deriving from *p*, a new pair *p'*, which models the authorized content of *p*. The use of filters had previously been considered in Vigiles [42], a fine grained access control framework for Hadoop. In [42] authorization filters are handled by means of per-user assignment lists, and filters are coded in Java by security administrators. In contrast, in GuardMR, filters are assigned to subjects on the basis of the covered roles, and a formal specification approach to the definition of filters is proposed, which allows specifying selection and modification criteria at a very high level of abstraction using the Object Constraint Language (OCL)[3] [9, 43]. GuardMR relies on automatic tools[4] to generate Java bytecode from OCL-based filter specifications, as well as to integrate the generated bytecode into the bytecode of the MapReduce task to be executed. GuardMR has been used with MapReduce tasks targeting both textual and binary resources [41], showing the flexibility of the approach. GuardMR and Vigiles do not require Hadoop source code customization, however, they rely on platform specific features, such as the Hadoop APIs and the Hadoop control flow for regulating the execution of a MapReduce task. A reasonably low enforcement overhead has been observed with both Vigiles and GuardMR. Neither Vigiles nor GuardMR provide support for context aware access control policies.

A recent work targeting access control enforcement within MapReduce systems is described in [23]. More precisely, [23] introduces the foundations of an access control model, called HeAC, which formalizes the authorization model of Apache Ranger[5] and Apache Sentry[6], as well as the native access control features of Hadoop.[7] Authorization assignments are specified for operations and objects, possibly on the basis of object tags, namely attributes specifying properties, like sensitivity, content or expiration date. Moreover, [23] introduces the foundation of Object Tagged RBAC, an RBAC model which, while preserving RBAC role based permission assignments, introduces support for object attributes. A prototypical implementation of the model has been defined by introducing role support into Apache Ranger. The proposed enforcement approach is again platform specific as it has been designed on top of Hadoop specific features. No support is given to context related properties, and no performance evaluation is presented.

4.2 NoSQL datastores

NoSQL datastores represent highly flexible, scalable, and efficient data management systems for Big Data, based on different data models. Cattell [7] classifies NoSQL systems on the basis of the adopted data model into three classes, namely key value, wide column, and document-oriented datastores, each suited to specific application scenarios. Key-value datastores (e.g., Redis[8]) can be seen as big hash tables with persistent storage services. Data are modeled by means of key-value pairs, where values of primitive or complex type are directly addressed by means of a key. Wide column stores (e.g., Cassandra[9]) model data as records with variable structures, which are then grouped into tables with flexible schema. Document-oriented datastores (e.g., MongoDB [10]) model data as hierarchical records, denoted documents, whose fields either specify a primitive value, or are in turn records composed of multiple fields. Documents are partitioned into collections, which in turn are grouped in a database.

Fine grained access control within NoSQL datastore management systems is still in the very early stage, and only few access control frameworks have been proposed so far for wide column and document oriented datastores.

K-VAC [29] is among the earliest fine grained access control frameworks targeting wide-column NoSQL datastores which have been proposed in the literature. K-VAC supports the enforcement of content-based, and context-based access control policies possibly specified at different levels of the data model hierarchy (e.g., for a column or for a row). Two prototypical versions of K-VAC have been released. One has been specifically designed as an internal module of Cassandra, a popular wide-column datastore whose

[2] http://hadoop.apache.org/
[3] https://www.omg.org/spec/OCL
[4] Dresden OCL Toolkit, http://st.inf.tu-dresden.de/oclportal
[5] https://ranger.apache.org/
[6] https://sentry.apache.org/
[7] Apache Ranger and Apache Sentry represent state of the art technologies for the enforcement of fine grained access control in Hadoop ecosystems.
[8] https://redis.io/
[9] http://cassandra.apache.org/
[10] https://www.mongodb.com

source code has been modified to host K-VAC's enforcement monitor. In contrast, the latter version has been released as an external library, with the aim to enforce access control on multiple datastores. However, the use of the proposed library still requires *ad-hoc* implementation of binding criteria, which so far have been only defined for Cassandra and HBase[11]. Overall the integration of K-VAC requires deep customizations of the hosting platform. Empirical performance evaluations show the efficiency of both the proposed prototypes, with a lower overhead measured with the customized version of Cassandra.

Another work targeting Cassandra has been proposed in [40], where an approach to the cryptographic enforcement of RBAC policies has been defined. Predicate encryption [28] and second level encryption [33] are used for the definition of an efficient scheme for RBAC enforcement which operates within Cassandra distributed architecture. The proposed approach is an example of platform specific solution designed on top of platform specific features, such as the distributed architecture of Cassandra. Also in this case no support is given for context-aware policies, and, unfortunately, the enforcement overhead is not discussed.

For what document-oriented datastores is concerned, efficient solutions to the integration of fine-grained purpose-based access control into MongoDB have been proposed in [14] and [15]. In [15] the RBAC model natively integrated in MongoDB has been enhanced with the support for the specification and enforcement of purpose-based policies [6] regulating the access up to document level. The proposed approach refines the granularity level at which the native MongoDB RBAC model operates. An enforcement monitor, called Mem (MongoDB enforcement monitor), has been designed, which monitors and possibly manipulates the flow of messages exchanged by MongoDB clients and the MongoDB server, thus acting like a proxy. Once Mem intercepts a message m issued by a MongoDB client on behalf of a subject s, it forwards m to the server, or it temporary blocks m, and issues additional messages finalized at profiling s. If m models a query q, Mem rewrites m as m' in such a way that m' encodes a query q' that only accesses those documents accessed by q which result authorized by the applicable access control policies. Mem's proxy based architecture allows the straightforward integration of the enforcement monitor into existing MongoDB deployments with basic configuration tasks. Experimental evaluations show the efficiency of the proposed approach, however also in this case no support is given for context-aware policies.

In [14], the framework presented in [15] has been significantly extended, introducing support for access control policies regulating the access up to field level, and providing support to specification and enforcement of content and context based policies. The proposed enforcement monitor, denoted ConfinedMem, applies the same logic as Mem, but it operates according to a two-steps process, which consists of: 1) the derivation of the authorized views of all documents to be accessed by a submitted query q included in a message m requiring the access to data resources, 2) the rewriting of m as m' in such a way that m' specifies a query q' which can only access the authorized views of the documents to be accessed by q. Different implementation techniques have been considered for queries specifying different operations (e.g., selection and aggregations) with the aim to minimize the overhead. Experimental evaluations show that overall, the enforcement overhead which has been observed with access control policies specified at field level is significantly higher than the one measured for document level policies.

[11] HBase is a popular wide-column store, https://hbase.apache.org/

5 PLATFORM INDEPENDENT APPROACHES

The great majority of the research contributions in the field of access control for Big Data analytics platforms propose a platform specific solution. This is probably due to the high heterogeneity of the considered application scenarios. Indeed, the lack of a reference standard for query languages and data models has caused the birth of a variety of proprietary solutions. As a matter of fact, numerous NoSQL datastores exist, most of which operate with a platform specific query language (e.g., the query language of MongoDB can only be used with that platform), and adopt a different data model. Even different datastores that nominally refer to the same data model can use different data organization and terminology. For instance, both MongoDB and CouchDB[12] use the document oriented data model, however the concept of collection is not supported by CouchDB, whereas collections are basic data organization features of MongoDB. The great heterogeneity of the scenario has significantly raised the complexity of devising enforcement solutions that can work with multiple platforms. Overall, the definition of a general access control enforcement approach represents a very ambitious task.

In the recent years, academia and industry started collaborating to the definition of unifying query languages for NoSQL datastores. To the best of our knowledge, JSONiq [22] and SQL++ [35] represent the most relevant results that have been so far achieved towards the fulfillment of this goal. JSONiq is an Xquery[8] based language that has been defined with the aim to analyze data handled by NoSQL datastores adopting a JSON-based data model. Unfortunately, at present JSONiq is only supported by Zorba,[13] and Sparksoniq,[14] which allow processing data serialized in JSON format, and by a platform denoted 28msec, [15] which supports the execution of JSONiq queries on MongoDB databases.

SQL++ [35] is a recent proposal of unifying query language that allows analysing semi-structured data handled by NoSQL datastores as well as structured data of traditional DBMSs. SQL++ has been recently adopted by Couchbase[16] and AsterixDB,[17][3] whereas Apache Drill [18] is in the process of aligning with SQL++. The diffusion of this language is thus growing, and the adopted SQL based syntax and the backward compatibility with relational DBMSs promise to further increase the popularity and diffusion of this language.

In [16] an SQL++ based Attribute-based Access Control (ABAC) [25, 26] framework for NoSQL datastores has been proposed. The choice to base the framework on SQL++ allows protecting any

[12] http://couchdb.apache.org/
[13] http://www.zorba.io
[14] http://sparksoniq.org/
[15] https://www.28msec.com/
[16] https://www.couchbase.com/
[17] https://asterixdb.apache.org/
[18] https://drill.apache.org/

NoSQL datastore which provides support to this language. Therefore, the proposal distinguishes from all other work introduced in Section 4 for higher generality and applicability, which may even grow with a future potential wider diffusion of SQL++. The framework operates at a very fine grained level, in that it allows regulating the access up to single data fields.[19] Enforcement is based on query rewriting and operates with heterogeneous data with no assumption on data schema, thus overcoming state of the art query rewriting techniques proposed for RDBMSs [30, 39]. Query rewriting techniques finalized at enforcing cell-level access control within traditional DBMSs operate by projecting or nullifying the value of each cell to be accessed by a query q on the basis of the compliance of the access performed by q with the applicable access control policies [30]. More precisely, a query q submitted for execution is rewritten in such a way to: i) include a subquery s for each table t accessed by q, which, cell by cell, generates an authorized view of t, and ii) perform the same analysis tasks as q on the result set of s. The subquery s specifies projection criteria conditioned by the compliance of the accesses operated by q with the cell level access control policies that have been specified for t's cells. A similar approach can only be used if the scheme of any accessed table is *a priori* known, as the projection criteria of the subqueries need to refer to table columns. The schemaless and highly heterogeneous nature of the data within Big Data platforms does not allow to use similar techniques. In [15] this issue has been handled by means of SQL++ operators that allow achieving the projection without knowing in advance the accessed fields. The approach operates by visiting, field by field, the data unit[20] *du* of an analyzed resource, and adding a visited field f to the authorized view *du'* of *du* only if the access to f complies with the ABAC policies specified for f. The proposed approach allows deriving in-memory authorized views of the data resources to be analyzed, and executing the analysis tasks of the original queries on such derived views. The ABAC framework proposed in [15] supports the specification and enforcement of context-aware access control policies. Empirical performance assessments show an enforcement overhead that varies with the characteristics of the specified policies and the number of fields of the analyzed documents. The overhead is high when field level policies cover high percentage of data units fields.

Another language-based ABAC approach has been proposed in [31], with the goal to be usable with traditional data management systems, Mapreduce systems, as well as NoSQL datastores. The work proposes a query rewriting approach that targets user transactions specified with an SQL-like language. Unfortunately, a detailed description of the adopted query language and data model is missing, which makes unclear how the approach could be used with different platforms, and how the heterogeneity of schemaless data can be handled by means of an SQL-like language.

6 RESEARCH ISSUES

In what follows we discuss some open research issues in the field of Big Data access control.

[19] The supported granularity is thus equivalent to cell level within relational DBMSs.

[20] SQL++ can be used with datastores adopting different data models, thus, the term data unit is used to denote a table row, or a document.

6.1 Unifying access control model and mechanism

State of the art review done in Sections 4 and 5 has highlighted that, although research in the area of access control for Big Data platforms is progressing, no solution has been proposed so far for a unifying access control framework which can combine generality and efficiency of access control. The heterogeneous schemaless nature of the managed data significantly complicates the definition of this framework, and so far this has lead mainly to ad-hoc platform specific solutions (see Section 4). In contrast, language centric approaches still suffer of limited applicability (see Section 5).[21]

One key element that may be instrumental to fill this void is the definition of a unifying data model capable of representing data resources of the different data models currently adopted by Big Data platforms. The ability to represent data resources is a fundamental requirements for binding access control policies to the protected data, as well as for the specification of policies regulating the access on the basis of the protected objects' attributes. Indeed, in the literature on access control, multiple models allow enforcing content-based access constraints (e.g., [14, 29]), as well as access control rules that refer to various security meta-data related to the protected data resources (e.g., [12]).

The key-value, wide column, and document-oriented models adopt different data modeling criteria, however, in all these models data are hierarchically organized as tree structures, where nodes at different height of the tree represent resources at different granularity levels of the related data model (e.g., database, table, row and cell). Data models differ among them for the height of the tree with which data resources can be represented. This may range from 2, within key-value datastores[22] to a height of variable length n (n > 2) for document-oriented datastores.[23] A data resource of a data model corresponds to a node n of the tree representing all the resources handled by a platform, and it can be accessed traversing the path from the root of the tree to n. Therefore, we believe that a unifying representation of data resources of multiple models should take into account the identification of proper modeling strategies for the nodes of the above mentioned resource tree. In particular, nodes should be specified in such a way to keep track of: i) any structural property related to the modeled resource, ii) hierarchical relations with other nodes (e.g., a parent of relationship), iii) possible meta-data, and iv) access control policies specified for the modeled resource. The considered policies may refer to different access control models, specifying context aware access control rules as well as content-based constraints.

Going one step further, the specified unifying model could also be used for enforcement purposes. We believe that enforcement mechanisms should be achieved by means of bidirectional mappings between resources represented with a platform specific native data model and the unifying data model. Overall, the analysis of related work has revealed that fine grained access control with schemaless

[21] Although the popularity of the SQL++[35] initiative is growing, the support provided to this language is still limited to a small number of platforms.

[22] All key-value pairs (leaf nodes), belong to a key-space (root node).

[23] A database (root node), groups a variable number of collections (level 2 nodes), which in turn include a variable number of documents (level 3 nodes), each composed of a variable number of fields, which in turn are possibly hierarchically organized into a tree structure (level 4 to n).

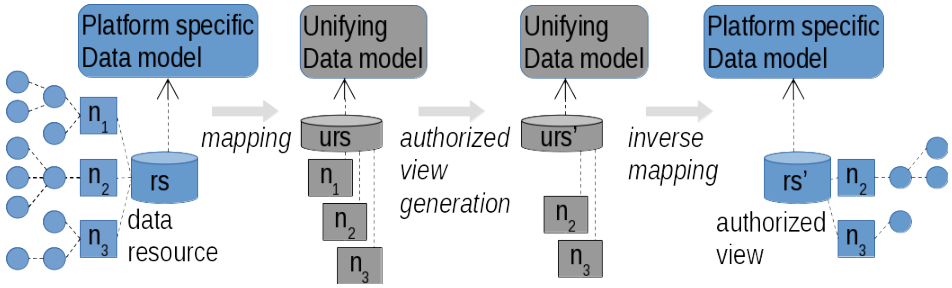

Figure 1: The pipeline of operations at the basis of the enforcement mechanism

data is usually enforced executing the submitted analysis tasks on authorized views of the accessed resources (e.g., see [16]). Therefore, a platform independent strategy to handle fine grained access control enforcement may consist of a pipeline of operations supported by any platform, which, by means of the unifying data model, handle the generation of authorized views. The generated view can then be analyzed by the originally submitted query without additional platform specific rewriting activities. The above mentioned pipeline is illustrated in Figure 1. For each accessed resource *rs*, represented as a tree characterized by different nodes n_i, the process: i) derives a unifying model-based representation *urs* of *rs*, ii) derives the authorized view *urs'* of *urs*, where the unauthorized contents have been removed, and, finally, iii) maps back the authorized view *urs'* to the native data model, so that the generated view *rs'* can be analyzed by the originally submitted analysis task. In order to support such approach within multiple platforms, the above mentioned mapping and view generation mechanisms should be defined in such a way that any platform, independently from the supported query language and data model, could handle the execution of this process. To the best of our knowledge, the majority of today Big Data platforms provide support for MapReduce computational paradigm, independently from the adopted data model and query language. Therefore, a promising approach could be that of specifying mapping and view generation mechanisms by means of MapReduce operations.

The enforcement overhead of the above discussed technique is expected to depend on the platform hosting the data to be protected, as different behaviors are expected to be observed. For instance, Apache Spark,[24] integrates a highly efficient computation engines, which promises to be significantly faster than Hadoop[2] (up to 100 times faster.[25]) The overhead is expected to be reasonably contained in all those platforms supporting in-memory MapReduce computations, as well as data streams.

6.2 Policy analysis tools

The availability of a unifying data model on which access control policies can be specified would also allow to support policy analysis and reasoning at an abstract layer independent from any specific platform. As a matter of fact, the variety of data models, access control models, and related configuration options, such as policy propagation and conflict resolution criteria, adopted by Big Data platforms, can make really hard for security administrators to understand the effect of a set of access control policies on the data resources which are managed by their systems, as well as assessing the quality of the specified policies. Most of the research efforts in this field have been devoted to correctness verification, detection of inconsistencies and redundancies, and policy sets completeness analysis (e.g., see [2, 4, 38]). A more recent work [5] has proposed the use of provenance techniques to check the quality of the specified access control policies for a scenario where collaborations are carried out by autonomous cognitive devices. However, to the best of our knowledge, so far no proposal has yet targeted Big Data platforms. The model centric approach previously discussed may be exploited as a basis for the definition of such policy analysis framework. For instance, it may be used to generate views of the protected resources that show the authorized and unauthorized contents when different policies and configuration options are used, as well as to quantify policy coverage for a requesting subject with respect to an execution context.

The definition of a policy reasoning tool is also instrumental to fulfill the new EU General Data Protection Regulation, (GDPR)[26] which is intended to strengthen data protection for all individuals within the European Union. GDPR applies regardless of where a company is located, provided that the company manages data of EU residents. GDPR introduces a set of very important principles for Big Data management, such as privacy by-design and by-default. The new regulation also emphasizes accountability for data controllers to demonstrate compliance to GDPR, whereas article 35 requires controllers to carry out Data Protection Impact Assessments in cases of potentially high-risk processing activities. All such principles require tools to clearly assess the effect of access control policies on the managed data.

Finally, a policy analysis framework is also required for community centered collaborative systems, such as online social networks and collaborative editing platforms, which may be seen as federated applications that handle Big Data. Recent surveys pointed out that these systems typically provide rudimentary forms of access control [37] A key requirement for access control models tailored for collaborative systems is to allow users to understand collaborative decisions, as well as to inspect users access preferences, and to evaluate their effects [37]. Paci et al. [37] claim that, although a few work exist which explain the effect of access decisions[24], and the

[24] https://spark.apache.org/
[25] https://www.datamation.com/data-center/hadoop-vs.-spark-the-new-age-of-big-data.html
[26] https://www.eugdpr.org/

reasons for which certain decisions have been taken [19], the above mentioned requirements are still largely understudied. Therefore, the definition of a reasoning framework capable of operating within such federated environments with multiparty access control models appears as a research challenge of paramount importance.

6.3 Access control for Big Data streaming analytics

In recent years, the number of Big Data platforms that provide support to data stream management is growing. Apache Spark[27] is probably the most popular open source framework which supports the analysis of continuous streams of data. Apache Storm[28] is another open source distributed real-time computation system which can also be used for real-time analytics and continuous computation. In addition, several commercial solutions exist, such as, for instance, Amazon Kinesis[29], which is a service for real-time processing of streaming data on the cloud, and IBM Streaming analytics,[30] a platform supporting risk analysis and decision making in real-time. Due to the growing emphasis to real-time analysis of data flows, a future research goal is the support of enforcement mechanisms targeting continuous flows of data. Some results have been presented in the past years in the field of Data Stream Management Systems. For instance, in [34], a framework called FENCE has been proposed, which supports continuous access control enforcement. Data and query security restrictions are modeled as meta-data, denoted security punctuations, which are embedded into the data streams. Different enforcement mechanisms have been proposed, which operate by analyzing security punctuations, such as special physical operators which are integrated within query execution plans with the aim to filter the tuples which can be analyzed, and rewriting mechanisms targeting continuous queries. The definition of similar approaches for the Big Data scenario is an open issue. A possible system centric strategy may consist in designing the mechanism on top of one of the existing streaming framework. However, similar to the platform specific approaches presented in Section 4, such a solution would suffer from a limited applicability. Moreover, existing solutions (e.g., FENCE [34]) operate at tuple level, whereas cell/field level granularity may be necessary in the Big Data scenario (see Sections 4 and 5), requiring a data filtering approach that operates at a finer granularity level. The development of an enforcement mechanisms based on language centric approaches seems still impracticable, as no standard continuous query language exists. In contrast, since some of these platforms can implement MapReduce tasks (e.g., Apache Spark, Apache Storm), a model centric approach may be a possible strategy, however, thorough investigations are required to support this intuition.

6.4 Access control for IoT ecosystems

Internet of Things (IoT) ecosystems are representative cases of Big Data applications. IoT applications are rapidly getting popularity in a variety of domains for the indisputable improvements of people life style. For instance, a growing number of users cannot do without wearable devices that continuously track their sport activities and health conditions. Due to the personal and sensitive nature of the handled information, security and privacy of these systems have become a major concern. As a consequence, in the recent years, several research efforts have been devoted to security and privacy of IoT applications, and a variety of access control models have been proposed (see, for instance, [36] for a compendium). These initial efforts have mainly produced models adopting centralized enforcement mechanisms (e.g., see [17]). However, multiple IoT ecosystems may be connected to each other exchanging data. As a matter of fact, federated systems where multiple IoT applications cooperate cannot be handled with centralized enforcement mechanisms. Multiparty access control solutions for IoT ecosystems are thus needed, which are suited to operate at Big Data scale. The definition of such access control frameworks still represent a big open research challenge.

[27] https://spark.apache.org/
[28] http://storm.apache.org/
[29] https://aws.amazon.com/kinesis/
[30] https://www.ibm.com/cloud/streaming-analytics

7 CONCLUSIONS

Security services for Big Data represent a key feature instrumental to foster trust on how data are managed and analyzed by Big Data platforms. This paper has focused on one of the key security service, that is, access control, by discussing the requirements that an access control solution for Big Data platforms should address. Moreover, the paper has provided a review of the state of the art in view of the devised requirements, and it has also discussed future research challenges in the area.

REFERENCES

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. 2002. Hippocratic databases. In *28th International Conference on Very Large Data Bases (VLDB)*.
[2] Gail-Joon Ahn, Hongxin Hu, Joohyung Lee, and Yunsong Meng. 2010. Representing and reasoning about web access control policies. In *Computer Software and Applications Conference (COMPSAC), 2010 IEEE 34th Annual*. IEEE, 137–146.
[3] Sattam Alsubaiee, Yasser Altowim, Hotham Altwaijry, Alexander Behm, Vinayak Borkar, Yingyi Bu, Michael Carey, Inci Cetindil, Madhusudan Cheelangi, Khurram Faraaz, et al. 2014. AsterixDB: A scalable, open source BDMS. *Proceedings of the VLDB Endowment* 7, 14 (2014), 1905–1916.
[4] Lujo Bauer, Scott Garriss, and Michael K Reiter. 2011. Detecting and resolving policy misconfigurations in access-control systems. *ACM Transactions on Information and System Security (TISSEC)* 14, 1 (2011), 2.
[5] E. Bertino, A. A. Jabal, S. B. Calo, C. Makaya, M. Touma, D. C. Verma, and C. Williams. 2017. Provenance-Based Analytics Services for Access Control Policies. In *2017 IEEE World Congress on Services, SERVICES 2017, Honolulu, HI, USA, June 25-30, 2017*. 94–101.
[6] J.W. Byun and N. Li. 2008. Purpose based access control for privacy protection in relational database systems. *The VLDB Journal* 17, 4 (2008).
[7] Rick Cattell. 2011. Scalable SQL and NoSQL Data Stores. *SIGMOD Rec.* 39, 4 (May 2011), 12–27. https://doi.org/10.1145/1978915.1978919
[8] Don Chamberlin, Daniela Florescu, Jonathan Robie, Jerome Simeon, and Mugur Stefanescu. 2003. XQuery: A query language for XML. In *SIGMOD Conference*, Vol. 682.
[9] Tony Clark and Jos Warmer. 2002. *Object Modeling With the OCL: The Rationale Behind the Object Constraint Language*. Vol. 2263. Springer.
[10] P. Colombo and E. Ferrari. 2014. Enforcement of Purpose Based Access Control within Relational Database Management Systems. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 26, 11 (2014). https://doi.org/10.1109/TKDE.2014.2312112
[11] P. Colombo and E. Ferrari. 2014. Enforcing Obligations within Relational Database Management Systems. *IEEE Transactions on Dependable and Secure Computing (TDSC)* 11, 4 (2014). https://doi.org/10.1109/TDSC.2013.48
[12] P. Colombo and E. Ferrari. 2015. Efficient Enforcement of Action aware Purpose Based Access Control within Relational Database Management Systems. *Knowledge and Data Engineering, IEEE Transactions on* (2015). in press.
[13] Pietro Colombo and Elena Ferrari. 2015. Privacy Aware Access Control for Big Data: A Research Roadmap. *Big Data Research* 2, 4 (2015), 145 – 154. https://doi.org/10.1016/j.bdr.2015.08.001

[14] Pietro Colombo and Elena Ferrari. 2016. Towards Virtual Private NoSQL datastores. In *32nd IEEE International Conference on Data Engineering, ICDE 2016, Helsinki, Finland, May 16-20, 2016*. 193–204.
[15] Pietro Colombo and Elena Ferrari. 2017. Enhancing MongoDB with Purpose-Based Access Control. *IEEE Trans. Dependable Sec. Comput.* 14, 6 (2017), 591–604. https://doi.org/10.1109/TDSC.2015.2497680
[16] Pietro Colombo and Elena Ferrari. 2017. Towards a Unifying Attribute Based Access Control Approach for NoSQL Datastores. In *33rd IEEE International Conference on Data Engineering, ICDE 2017, San Diego, CA, USA, April 19-22, 2017*. 709–720.
[17] Pietro Colombo and Elena Ferrari. 2018. Access Control Enforcement within MQTT-based Internet of Things Ecosystems. In *Proceedings of the ACM Symposium on Access Control Models and Technologies (SACMAT)*. ACM.
[18] Jeffrey Dean and Sanjay Ghemawat. 2004. MapReduce: Simplified Data Processing on Large Clusters. In *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6 (OSDI'04)*. USENIX Association, Berkeley, CA, USA, 10–10. http://dl.acm.org/citation.cfm?id=1251254.1251264
[19] Jerry den Hartog and Nicola Zannone. 2016. A Policy Framework for Data Fusion and Derived Data Control. In *Proceedings of the 2016 ACM International Workshop on Attribute Based Access Control (ABAC '16)*. ACM, New York, NY, USA, 47–57. https://doi.org/10.1145/2875491.2875492
[20] D.F. Ferraiolo, R. Sandhu, S. Gavrila, D.R. Kuhn, and R. Chandramouli. 2001. Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security (TISSEC)* 4, 3 (2001).
[21] Elena Ferrari. 2010. *Access Control in Data Management Systems*. Morgan & Claypool Publishers.
[22] D. Florescu and G. Fourny. 2013. JSONiq: The History of a Query Language. *IEEE Internet Computing* 17, 5 (Sept 2013), 86–90. https://doi.org/10.1109/MIC.2013.97
[23] Maanak Gupta, Farhan Patwa, and Ravi Sandhu. 2017. Object-Tagged RBAC Model for the Hadoop Ecosystem. In *Data and Applications Security and Privacy XXXI*, Giovanni Livraga and Sencun Zhu (Eds.). Springer International Publishing, Cham, 63–81.
[24] H. Hu, G. J. Ahn, and J. Jorgensen. 2013. Multiparty Access Control for Online Social Networks: Model and Mechanisms. *IEEE Transactions on Knowledge and Data Engineering* 25, 7 (July 2013), 1614–1627. https://doi.org/10.1109/TKDE.2012.97
[25] Vincent C Hu, David Ferraiolo, Rick Kuhn, Arthur R Friedman, Alan J Lang, Margaret M Cogdell, Adam Schnitzer, Kenneth Sandlin, Robert Miller, Karen Scarfone, et al. 2013. Guide to attribute based access control (ABAC) definition and considerations (draft). *NIST special publication* 800, 162 (2013).
[26] V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo. 2015. Attribute-Based Access Control. *Computer* 48, 2 (Feb 2015), 85–88. https://doi.org/10.1109/MC.2015.33
[27] Xiaolong Jin, Benjamin W. Wah, Xueqi Cheng, and Yuanzhuo Wang. 2015. Significance and Challenges of Big Data Research. *Big Data Research* (2015). https://doi.org/10.1016/j.bdr.2015.01.006
[28] Jonathan Katz, Amit Sahai, and Brent Waters. 2013. Predicate encryption supporting disjunctions, polynomial equations, and inner products. *Journal of cryptology* 26, 2 (2013), 191–224.
[29] Devdatta Kulkarni. 2013. A fine-grained access control model for key-value systems. In *Proceedings of the third ACM conference on Data and application security and privacy*. ACM, 161–164.
[30] Kristen LeFevre, Rakesh Agrawal, Vuk Ercegovac, Raghu Ramakrishnan, Yirong Xu, and David DeWitt. 2004. Limiting disclosure in hippocratic databases. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment, 108–119.
[31] Jim J. Longstaff and Joanne Noble. 2016. Attribute Based Access Control for Big Data Applications by Query Modification. In *Second IEEE International Conference on Big Data Computing Service and Applications, BigDataService 2016, Oxford, United Kingdom, March 29 - April 1, 2016*. 58–65.
[32] Viktor Mayer-Schönberger and Kenneth Cukier. 2013. *Big data: A revolution that will transform how we live, work, and think*. Houghton Mifflin Harcourt.
[33] Mohamed Nabeel and Elisa Bertino. 2014. Privacy preserving delegated access control in public clouds. *IEEE Transactions on Knowledge and Data Engineering* 26, 9 (2014), 2268–2280.
[34] R. V. Nehme, H. S. Lim, and E. Bertino. 2010. FENCE: Continuous access control enforcement in dynamic data stream environments. In *2010 IEEE 26th International Conference on Data Engineering (ICDE 2010)*. 940–943. https://doi.org/10.1109/ICDE.2010.5447899
[35] Kian Win Ong, Yannis Papakonstantinou, and Romain Vernoux. 2014. The SQL++ unifying semi-structured query language, and an expressiveness benchmark of SQL-on-Hadoop, NoSQL and NewSQL databases. *CoRR, abs/1405.3631* (2014).
[36] Aafaf Ouaddah, Hajar Mousannif, Anas Abou Elkalam, and Abdellah Ait Ouahman. 2017. Access control in the Internet of Things: Big challenges and new opportunities. *Computer Networks* 112 (2017), 237 – 262. https://doi.org/10.1016/j.comnet.2016.11.007
[37] Federica Paci, Anna Squicciarini, and Nicola Zannone. 2018. Survey on Access Control for Community-Centered Collaborative Systems. *ACM Comput. Surv.* 51, 1, Article 6 (Jan. 2018), 38 pages. https://doi.org/10.1145/3146025
[38] Edelmira Pasarella and Jorge Lobo. 2017. A Datalog Framework for Modeling Relationship-based Access Control Policies. In *Proceedings of the 22nd ACM on Symposium on Access Control Models and Technologies*. ACM, 91–102.
[39] S. Rizvi, A. Mendelzon, S. Sudarshan, and P. Roy. 2004. Extending query rewriting techniques for fine-grained access control. In *ACM SIGMOD 2004*. 551–562.
[40] Yossif Shalabi and Ehud Gudes. 2017. Cryptographically Enforced Role-Based Access Control for NoSQL Distributed Databases. In *Data and Applications Security and Privacy XXXI*, Giovanni Livraga and Sencun Zhu (Eds.). Springer International Publishing, Cham, 3–19.
[41] H. Ulusoy, P. Colombo, E. Ferrari, M. Kantarcioglu, and E. Pattuk. 2015. GuardMR: Fine-grained Security Policy Enforcement for MapReduce Systems. In *ACM ASIACCS 2015*.
[42] Huseyin Ulusoy, Murat Kantarcioglu, Kevin Hamlen, and Erman Pattuk. 2014. Vigiles: Fine-grained Access Control for MapReduce Systems. In *IEEE BigData*.
[43] Jos B Warmer and Anneke G Kleppe. 1998. The object constraint language: Precise modeling with uml (addison-wesley object technology series). (1998).