Dynamic and Scalable Enforcement of Access Control Policies for Big Data

Marco Anisetti, Claudio A. Ardagna, Chiara Braghin, Ernesto Damiani, Antongiacomo Polimeno
firstname.lastname@unimi.it
Università degli Studi di Milano
Milan, Italy

Alessandro Balestrucci
alessandro.balestrucci@consorzio-cini.it
Consorzio CINI
Roma, Italy

ABSTRACT

The conflict between the need of protecting and sharing data is hampering the spread of big data applications. Security and privacy assurance is required to protect data owners, while data access and sharing are fundamental to implement smart big data solutions. In this context, access control systems can assume a central role in balancing data protection and data sharing. However, existing access control solutions are not general and scalable enough to address the software and technological complexity of big data ecosystems, being unable to support such a dynamic and collaborative environment. In this paper, we propose an access control system that enforces access to data in a distributed, multi-party big data environment. It is based on data annotations and secure data transformations performed at ingestion time. We show the feasibility of our approach in the smart city domain using an Apache-based big data engine.

CCS CONCEPTS

• **Security and privacy** → **Database and storage security**; • **Computer systems organization** → *Distributed architectures*; • **Information systems** → **Extraction, transformation and loading**.

KEYWORDS

Access Control, Big Data, Data Transformation, Data Ingestion

ACM Reference Format:
Marco Anisetti, Claudio A. Ardagna, Chiara Braghin, Ernesto Damiani, Antongiacomo Polimeno and Alessandro Balestrucci. 2021. Dynamic and Scalable Enforcement of Access Control Policies for Big Data. In *International Conference on Management of Digital EcoSystems (MEDES '21), November 1–3, 2021, Virtual Event, Tunisia.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3444757.3485107

1 INTRODUCTION

Today's industrial domains are based on large digital ecosystems where huge amounts of data need to be exported and shared between actors working on different projects, within and across organizational boundaries. In these distributed and data-intensive ecosystems, data protection issues are also magnified by the variety of both data and technologies, and by the fact that the type of data collected is increasingly becoming more and more sensitive. This scenario introduces a clear IT governance conflict: data should be compartmentalised to ensure strong protection, on one side, and shared to enable advanced analytics and key inter-organizational business processes, on the other side.

Historically, companies have been handling the data protection/data sharing conflict by adopting access control policies. Unfortunately, access control models developed in the 2000s for Service-Oriented Architectures fail to adequately manage the creation, use and dissemination of big data. They can be both ineffective and inefficient when applied to today's dynamic coalitions, where: *i)* partners join without necessarily integrating their cloud-based or on-premises ICT infrastructures, *ii)* collaborative processes are carried out involving multi-party data collection and analytics, *iii)* there are continuous changes in the security space structure of temporary coalitions, with many parameters for access right decisions unknown at policy writing time. The notion of Dynamic Access Control (DAC) [12, 21] has then been introduced going beyond traditional allow/deny based on users or groups and expressing access conditions that vary according to a series of parameters. More recent attempts either focus on a specific processing phase without considering data protection during the whole data life cycle (i.e., at rest, in use, and in motion), or are platform-specific and thus not applicable to a generic big data scenario [14, 22].

This paper defines a systematic approach to overcome the limits of current solutions dealing with data governance and access control in big data scenarios. The paper proposes an intelligent enforcement approach of attribute-based access control policies, which is executed at data ingestion time and builds on the concepts of *separation of duties*, *data annotation*, and *data transformation*. It performs data-model specific transformations of the requested resources according to data annotations. Data transformations are executed in different domains that depend on the state of the collaboration and the specific target where data are routed (separation of duties). Our approach makes policies verifiable and adaptable to the evolving state of resources and individuals within a specific big data context, where *i)* heterogeneous and schemaless data are collected at a fast rate and shared/analysed in a dynamic coalition, *ii)* context management drives access control enforcement, and *iii)* efficiency is a must due to the increasing cardinality of data and number of requests.

The contribution of the paper is threefold: *i)* we identify the key requirements for a big data governance solution based on access control (Section 2); *ii)* we introduce a novel attribute-based access control methodology working at data ingestion time, which is based on separation of duties and whose enforcement addresses the data protection/data sharing conflict (Section 4); *iii)* we propose a taxonomy of data transformations ensuring different security properties suitable for big data ingestion pipelines (Section 4.2). An implementation of our approach is provided using an Apache-based big data engine (Section 4.3), showing its benefits in a smart city scenario (Section 5).

2 REQUIREMENTS AND REFERENCE SCENARIO

2.1 Access Control Requirements

We propose an access control system that is at the core of a big data governance solution. The system must consider technical peculiarities of big data systems [2, 3], which points to scenarios where huge amount of data are diverse, come at high rates and must be proven to be trustworthy, as clarified by the 5V storyline [11]. On top of this, big data is made of an ecosystem of services, mainly from third parties, increasing the governance complexity under multiple perspective. Rigid control, even if possible, does not fit the need to fully exploit the big data processing capabilities, while loosely control is not acceptable in many application context. New access control requirements then emerge as follows.

[R1]: Access to data must be evaluated before data analytics takes place.
[R2]: Authorization should be the primary focus. Authentication is assumed to be managed by a separate and integrated module to guarantee federations within big data ecosystem.
[R3]: Access control enforcement must protect data during their entire life cycle. Access control policies must be enforced at each phase of the analytics process, guaranteeing that data are properly protected and shared only to authorized users and for authorized operations.
[R4]: Access control policies must support the specification of context-based access conditions, that is, access rights might depend on the run-time characteristics of the big data system (e.g., in the smart city scenario, a traffic control system should implement some form of adaptive signalling to mitigate dangerous situations during emergencies such as a car accident).
[R5]: Access control enforcement should support fine-grained access control, dealing with both structured and unstructured data. When structured data are considered, policies can refer to a single cell, a column, a tuple or an entire table of structured data. When unstructured data are considered, policies can specify the portion of the unstructured file whose access need to be regulated.
[R6]: Access control enforcement should not use data ownership as the only attribute to define access rights, but rather being applied at ingestion time on the basis of the evolving state of resources and individuals within a specific big data context.
[R7]: Access control should protect the privacy of sensitive data.
[R8]: Access control enforcement should be driven by dynamic and contextual annotations on data.
[R9]: Access control enforcement should be highly efficient and scalable to accomplish the increasing cardinality of data and rate of requests.

2.2 Big Data-Enabled Smart City Scenario

Smart cities are clearly emerging as one of the key scenarios where the adoption of big data can have disruptive benefits. It is also one of the more challenging domains for data governance given *i)* the heterogeneity of actors having different data access rights and processing capabilities, *ii)* the high sensitivity of collected data. Our reference scenario is a smart city using digital technologies to develop, deploy, and promote sustainable development practices addressing growing urbanization challenges. In particular, IoT sensors, video cameras, social media, and other inputs are used to collect data. Insights gained from data are used to manage assets, resources and services efficiently and to improve the operations across the city. In a smart city scenario, all participants need to share information and combine it with contextual data analyzed in real time; at the same time, multiple domains must cooperate sharing real-time contextual information to achieve an improved and sustainable management process. This clearly points to the need of a common storage (i.e., data lake) making data available to all city services, and a multi-tenant data governance platform capable to handle different organizations belonging to the same smart city. At the same time, data protection must be properly implemented to avoid abuses based on collected data.

3 BIG DATA PROCESSING

A big data pipeline is a sequence of processing tasks that can be parallelized to exploit the big data infrastructure capabilities. Depending on the scope, pipelines can be roughly classified as *i) ingestion pipeline*, capturing and transforming data with the scope of saving it for further analysis, dealing with different data format (i.e., structured, unstructured, and semi-structured), *ii) analytics pipeline*, executing specific analysis on data, which may include data preparation tasks (e.g., normalization, cleaning, selections) to make data suitable for specific analytics, *iii) visualization pipeline*, presenting the output of an analytics to users.

Figure 1 shows a simplified view of a big data processing system, dividing the pipelines in two separate procedures: *i) ingestion procedure*, involving data sources and ingestion pipelines, and *ii) analytics procedure*, involving analytics and visualization pipelines. Solid arrows present the typical batch or analytics model generation flows. Dashed arrows present typical streaming or prediction flows. Let us first consider the ingestion procedure as follows.

- Data Sources are the source of row data. They are connected to the ingestion pipeline via specific connectors (e.g., queues, API, ingestion tools).
- Ingestion pipeline is a pipeline responsible to eventually transform data flows prior to storing them. Data can be either ingested in batches, in real-time (stream), or using a hybrid combination of the two patterns.

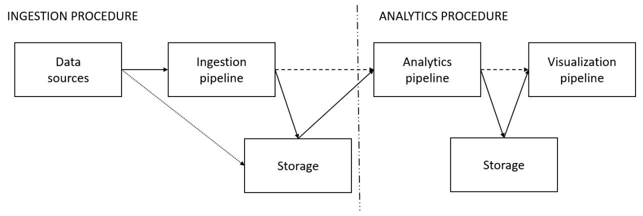


Figure 1: Simplified view of a big data processing system.

- Data storage is a distributed medium (e.g., data lake, data warehouse) where data are permanently stored. Data storage behaves differently depending on the data format (i.e., structured and unstructured data), adopting different storing solutions (e.g., tables, files) and query languages (e.g., SQL, NoSQL), and supporting different types of operations on the data.

Data ingestion procedures are composed of three steps: *i) Extract*, referring to pulling the source data from the original data source (played by the data source connectors), *ii) Transform*, the process of changing the structure of data to integrate with the target data system (ingestion pipeline in Figure 1), and *iii) Load*, referring to the process of depositing the data at rest into a data storage (storage in Figure 1). The ordering of these steps define the two main approaches: ETL (Extract, Transform, Load) and ELT (Extract, Load, Transform). ETL is typical for data warehouses and Online Analytical Processing (OLAP). In this case, data are sent in a temporary staging area prior to being transformed to a structured form. ELT is emerging as the prevalent scheme for big data analytics: data are stored immediately into a data lake storage system (dotted arrow in Figure 1), without any structural transformation. In terms of data governance, ETL is a conservative approach, supporting a priori compliance to regulations since non-compliant data are never stored in the data warehouse. On the contrary, ELT is focused on ingesting row data as they become available, transforming data only when needed. This approach brings flexibility, since with ETL ingestion pipelines may require modification in case the data structure does not allow for a new type of analysis.

Let us then consider the analytics procedure as follows.

- Analytics pipeline is responsible to retrieve some value out of ingested data (e.g., build a ML model, apply a model for prediction).
- Visualization pipeline graphically visualizes the outcome of an analytics or any aggregations on the data in the data lake. For this purposes, it usually includes some processing tasks suitable to lower the plotting dimensions or aggregate data in specific manner.

It is important to note that the ingestion pipeline acts as the middleware between data sources and analytics pipeline. It is therefore the most suitable candidate for implementing and integrating our access control approach, as discussed in the remaining of this paper.

4 INGESTION-TIME ACCESS CONTROL

We propose a novel ingestion procedure enriched with access control capabilities. Our ingestion procedure performs data-model transformations of the ingested resources that depend on the state of the collaboration and the specific target where data are routed (separation of duties). Our approach makes policies verifiable and adaptable to the evolving state of resources and individuals within a specific big data context. It is based on an advanced ETL schema where *i)* data transformation is the result of a policy enforcement and *ii)* the target of phase load of the data ingestion procedure is the data lake infrastructure. The scope of our advanced ETL is to enforce access control at ingestion time before an access request is received. The access control policies are designed and enforced to guarantee that the data stored in the data lake are compliant with the policies for specific services/actors and directly accessible with no delay when requested. Our approach balances the flexibility of ELT with the control capability of traditional ETL. We note that implementing access control at ingestion time can lead to data duplication in case a specific data source is ingested by **or routed to** two different actors having different access control policies to comply with. However, in a big data context, such duplication can be considered negligible compared to the advantages obtained in terms of data governance and analytics performance.

Our methodology builds on and extends the policy structure of traditional attributed-based access control systems, whose policies rules are expressed as a tuple of the form
$\langle \texttt{subject}, \texttt{action}, \texttt{object}, [\texttt{context}], [\texttt{obligation}], \texttt{deny/allow} \rangle$
where an optional obligation specifies an action that must be performed before the policy is enforced. Obligations are in most of the cases actions that are not directly associated with the controlled data. In collaborative big data scenarios, however, data are shared among large coalitions of different organizations but it is still important to ensure that sensitive data are properly protected and only exposed to the level required for specific business needs. In this context, deny access to data is the very last decision to take. Deny means no analytics, and then no value. In this paper, we put forward the idea that, in the big data context, preemptive data transformations are more suitable that denying data access. Indeed, we propose a platform-independent methodology still based on the common XACML architecture, where policy rules are of the form: $\langle \texttt{subject}, \texttt{action}, \texttt{object}, \texttt{context}, \texttt{data transformation} \rangle$, where a data transformation is a type of obligation that sanitizes data resources to guarantee a minimum level of access to data. Deny access to data can happen in extreme cases when data transformations delete all data resources.

Figure 2 shows the big data processing system in Figure 1 extended with our policy-aware ingestion pipeline that builds on the following building blocks.

(1) Data annotation is the process where data sources are enriched with contextual information and metadata on data sensitivity/peculiarities. Such a process indicates to the ingestion pipeline the proper way of dealing with the data and drives data transformation.
(2) Taxonomy of data transformations as a library of transformations that can be executed on data resources. It includes pruning, reshaping or encrypting/decrypting the different

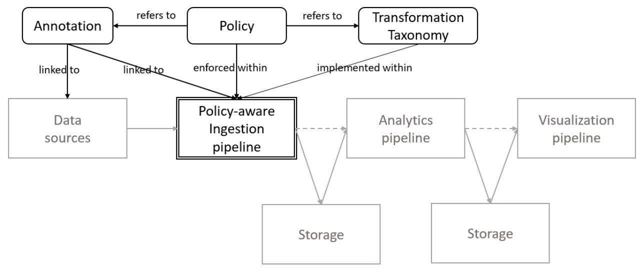


Figure 2: The ingestion-time access control methodology.

resource parts.Each entry in the taxonomy is mapped to a transformation function at ingestion layer.

(3) Policies expressing the need to enforce certain transformations based on attributes and actors executing the ingestion process.

The policy-aware ingestion pipeline extends the ingestion pipeline with the ability to enforce access control policies and transforming data. In the rest of the section, we describe in detail the building blocks of our methodology, and show how we implement our enforcement approach using an Apache-based big data engine.

4.1 Data Annotation

We implement a data annotation process at ingestion time to support our dynamic access control enforcement. A tag is a label consisting of a key and an optional value that is assigned to a resource. In our case, tags are used to specify attributes of resources or contextual information, supporting the specification of tag-based access control policies. Using data annotation to enforce data protection policies has three major advantages:

(1) Metadata tagging permits to identify, organise and extract value out of (possibly unstructured) ingested raw data. For example, it is possible to capture document semantics through tags. Thus, tagging helps in defining policies at different levels of granularity, also in case of unstructured data.
(2) Tagging enables to categorize resources in different ways (e.g, by purpose, owner, environment, or other criteria), encompassing conventional database schema information. Moreover, relationships between attributes of different data sets can be indicated as tags, without the need of, computationally expensive, data normalization.
(3) Tagging permits the definition of access control policies on categories of resources and not on a specific resource, which is more efficient in a distributed and highly dynamic environment (i.e., it is not necessary to update the policy each time a new resource is added).
(4) Tagging supports dynamic policies that change according to changes in the data annotation process. Tags can in fact be modified or added during all the data processing lifecycle, permitting a fine-grained data governance.

When using tagging for access control, it is important to devise a consistent set of tag keys, that is, a common vocabulary. However, which tags to apply to which resources differs depending on the specific use case, working environment, or context. For example, tags can be used to express operational requirements, ownership information, a resource workload or data sensitivity. Tags can then be used to express multi-level security policies by means of owner-defined security levels or role-based policies, to limit access to specific environments (e,g., development, test, quality assurance or production), to add information to data (e.g., creation date, expiring date, details on the content).

Table 1 provides an example of a common vocabulary at the basis of tag-based access control policies. For instance, use case *privacy classification* should be used every time data privacy must be protected, that is, every time personal data are involved. Data privacy generally refers to the ability of a person to determine when, how, and to what extent her personal information can be shared with or communicated to others. According to the General Data Protection Regulation (GDPR) [1], personal data are any information relating to an identified or identifiable natural person (called data subject). Based on this definition and on the technological advances that have improved data collection and analytics, personal data (Personally Identifiable Information - PII) fall into three categories: *i)* explicit identifiers, any information that directly identifies individuals with certainty, such as national ID, assurance number, phone number, passport number; *ii)* quasi identifiers, a set of data attributes that could jointly or uniquely identify a subject when combined with publicly available data, such as ZIP code, date of birth, and address; *iii)* sensitive information, data related to a person without directly identifying her but that, if linked to an individual, reveals sensitive aspects of her private life (e.g., religion belief, health information, legal issues).

Example 4.1 (Data annotation). Let us consider a traffic monitoring service regulating accesses to a Low Emission Zone in the smart city scenario in Section 2.2 . Data sources are videos of city traffic. At ingestion time the ingestion pipeline enriches videos (unstructured data) with vehicles' information (structured data). Our data annotation refers to both data sources and the data generated by the ingestion pipeline, with the aim to preserve users' privacy. More specifically, the tagging vocabulary contains:

- a set of tags to be associated with the snapshots metadata taken by the video camera, to add information such as *ViolationTime* indicating the time the picture was taken, *CameraID* to retrieve the street controlled by the camera, *Type*=`JPG` and *Content*=`PlateID` specifying that the JPG file contains a picture with a given plate number;
- the privacy classification tags introduced in Table 1, to be assigned both to the *Content* of the snapshot file and to the vehicles' information that includes owner's personal attributes (e.g., license plate number, owner and home address).

Table 2 shows the annotation tags of vehicles' information.

4.2 A Taxonomy of Data Transformations

A taxonomy of data transformations collects all functions used to sanitize the ingested data according to access control policies. Data transformations are categorised depending on the security property they aim to guarantee.

4.2.1 Data transformations for confidentiality (or integrity). The conventional security mechanisms used to protect data are encryption or hashing. With the advent of multi-tenant distributed clouds

Table 1: Tagging use cases

Use case	Description	Tag key and possible values
Data classification	User-defined sensitivity of data	*DataClassification*: `Public, Secret, Top Secret`
Protected health info	Health data created, received, stored or transmitted by HIPAA-covered entities (e.g., demographic data, medical histories, test results, insurance information, etc.)	*PHI*
Privacy classification	Any information that permits the identity of an individual to be directly or indirectly inferred	*PrivacyClassification*: `PII-id, PII-quasi-id, PII-Sensitive`
Disaster recovery	Business criticality of the application, workload, or service	*DR*: `Mission-critical, Critical, Essential`
Business criticality	Business impact of the resource	*Criticality*: `Low, Medium, High`
Environment	Deployment environment of the workload or service	*Env*: `Prod, Dev, QA, Stage, Test`
Roles	Roles of employees in a hospital	*Role*: `IT admin, Doctor, Nurse, Radiologist, Cardiologist, etc.`
Owner name	Owner of the workload or service	*Owner*: `mrossi@companyA.com`
Type	Type of document	*Type*: `JPG,PDF,TXT,XML,HTML, ...`
Content	Content of document	*Content*: `FacePic,EnvPic, ...`
Creation date	Creation date of the resource	*CreationDate*: `2023-10-15`
Source	Origin of data	*SensorID*: `Camera123`

Table 2: Result of data annotation process in Example 4.1.

`PII-id`	`PII-semi-id`	`PII-Sensitive`	`PII-id`	`PII-semi-id`
plate_id	**owner_LN**	**owner_FN**	**owner_BD**	**address**
M-X2495	Brown	Alice	1971-01-01	11, Grant Avenue
B-AR667	Green	Bob	1968-08-16	345, Market Street
B-A7813	White	Charlie	1950-03-03	2, Van Ness Avenue
ALZ-1234	Black	Trudy	2001-01-02	23, Alemany Boulevard

and collaborative machine learning environments, ad hoc encryption schemes have been proposed as follows.

Attribute based encryption (ABE) [13, 19]. It is a public-key encryption scheme where the key pair of a user and a ciphertext are dependent on a set of attributes. The decryption of a ciphertext is possible only if the set of attributes of the user wanting to decrypt the ciphertext matches the attributes of the ciphertext.

Identity based encryption (IBE) [7]. It is an alternative to public key encryption and aims to simplify key management in a certificate-based public key infrastructure (PKI), by using human identities such as email or IP addresses as public keys.

Homomorphic encryption [5]. It is a form of encryption that permits performing computations on encrypted data without first decrypting it. The resulting computations are left in an encrypted form that, when decrypted, results in an identical output to the one obtained when performing computations on unencrypted data.

4.2.2 Data transformations for anonymity. Data anonymization is the general term used to describe the process of hiding identifiable information that may lead to personal identification, so that users described by such data remain anonymous. Main anonymization techniques are described as follows.

Suppression. It removes an entire part of data, thus affecting their usability, for example by replacing original values with a symbol that represents a null character. In case of structured data, a column or an entire tuple can be removed, or substituted with a null character. For example, a licence plate number can be substituted by the ###### sequence.

Generalization. It replaces the value with a less specific but semantically consistent value. Generalization is typically applied to structured data, however not all attributes are suitable for it. For example, numerical values can be easily generalized into a fixed range of values (e.g., age), whereas a user ID can hardly be generalized.

Masking and encryption. They change characters to different characters, making the value inconceivable. It is a general case of truncation and of pre-fix preserving. Masking and encryption techniques make data useless for analysis. In addition, they are computational inefficient: in the first case, values must be checked before changing them; in the second case, encryption has a cost.

Swapping. It rearranges variables randomly within a specific column in a relational database.

Distortion. It maps the value to a new/different value, obtained by using a hash function or a cryptographic primitive.

Data anonymization is frequently used to preserve privacy. In this case, it is important to identify which technique should be applied to each PII identifier, quasi-identifier and sensitive data. Significant data should in fact be retained for analysis, but not at the cost of data leakage. The *k-anonymity* method [20] uses a combination of generalization, masking and suppression over an original data set to achieve k-anonymity, that is, to obtain an anonymized data set where each tuple is similar to at least other $k - 1$ tuples on the potentially identifying attributes that have been obfuscated.

Example 4.2 (Data Transformation). Let us consider the tags used in Example 4.1 and highlighted in red in Table 2. Table 3 shows how the personal attributes of the owner labeled as `PII-id` or `PII-quasi-id` during the annotation phase are transformed by

Table 3: Vehicle owner data of Example 4.1 after the transformation (i.e., anonymization) described in Example 4.2.

plate_id	owner_FN	owner_BD	address
M-xxxxx	Alice	1971-01-01	Grant Avenue
B-xxxxx	Bob	1968-01-01	Market Street
B-xxxxx	Charlie	1950-01-01	2, Van Ness Avenue
ALZ-xxxxx	Trudy	2001-01-01	Alemany Boulevard

the ingestion pipeline to preserve privacy. More in detail, different obfuscation techniques have been applied to the different columns:

- The first column records the plate number. In the example, we followed the German plates' format, where the first 1-3 letters are for the Region and City (e.g., M stands for Munich, B for Berlin and ALZ for Alzenau in Bavaria), followed by the state/city seals and random letters or numbers. Plate ids have been only partially masked by substituting the free sequence of letters and numbers, to maintain the information of the possible origin of the vehicle.
- The column with the last name of the owner has been suppressed. On the contrary, in the example, we maintained the first name of the vehicle owner (for possible gender statistics).
- Generalization has been applied to the column with the birth date of the vehicle owner, where only the year value corresponds to the original one.
- The last column shows another example of generalization applied to the street address, where the street number has been removed.

4.3 An Apache-based Big Data Engine Architecture.

Our methodology is implemented within a big data engine based on the Apache ecosystem. Figure 3 shows our architecture, which is detailed in the following.

- **Hadoop:** It is a framework designed for storage and processing of large amount of data on clusters made of commodity hardware. It constitutes the lowest layer of our architecture in Figure 3 providing storage (i.e., HDFS) and resource management (i.e., YARN). It also provides Map Reduce processing model suitable for processing analytics that does not suffer from data intensive activities. It is used to support the storage functionality in Figure 2.
- **Hive:** It is a structured data warehouse based on Hadoop. It manages large data sets residing in a distributed storage using a SQL-like language. It is part of layer Data Management and leverages on layer resource management. To query data, a Hive query is first converted into Map Reduce and then processed by Hadoop. It is used to implement the storage functionality in Figure 2.
- **Spark:** It is an advanced unified analytics engine for large-scale data processing. It leverages on Hadoop, but compared to the Hadoop Map Reduce it *i)* focuses on in-memory processing of cached data, *ii)* offers a more expressive computing model not limited to Map and Reduce instructions, and *iii)* achieves fault tolerance via re-execution and lineage instead of replication. It is part of layer processing in Figure 3 and can leverage on YARN for processing resource management. It also provides the support for tools like Spark SQL, Mlib and structured streaming with Spark stream. It is used to provide processing capabilities for pipelines ingestion and analytics in Figure 2.

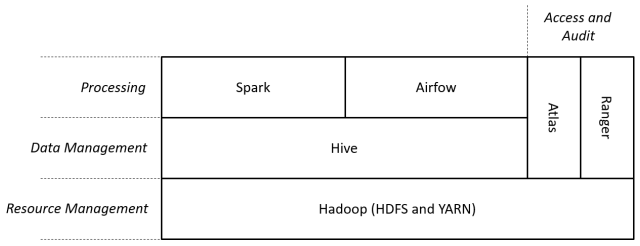


Figure 3: An Apache-based Big Data Engine Architecture

- **Airflow:** It is an orchestrator of processing pipelines aimed to provide scheduling and monitoring capabilities. Differently from other orchestrators designed with a metalanguage (e.g., XML), Airflow pipelines are built in Python, supporting better dynamic pipeline generation. The pipeline is modeled as a DAG of tasks. Tasks can be Spark jobs and Airflow can easily manage failures or complex and dynamic workflows of Spark jobs. It is part of layer processing in Figure 3. It is used to implement to create the flows between/within pipelines of Figure 2.
- **Ranger:** It offers centralized authorization and auditing across Hadoop components via specific plugins. It permits to design attribute-based access control policies using resource classifications or tags, and to specify data transformations. It is part of layer access and audit in Figure 3. It also offers audit capabilities of policy enforcement. It implements the policies and provides enforcement capabilities to policy-aware ingestion pipeline in Figure 2. It is also used to define the transformation procedures as defined in the Transformation Taxonomy in Table 1.
- **Atlas:** It provides a metadata repository with a flexible type system to capture schema and metadata of multiple components (e.g, Hive and HDFS). It supports the data annotation process, tag policy specification in Ranger, and data lineage and provenance. It is part of layer access and audit in Figure 3. It provides also search functionalities based on attributes. It is used to implement the annotation functionality in Figure 2.

5 CASE STUDY

Let us consider an extended version of the smart city scenario in Example 4.1, where *Smog Analysis Service* is considered in addition to the *Traffic Monitoring Service* regulating accesses to Low Emission Zones. Both services are based on the same data sources and ingestion pipeline.

5.1 Walkthrough

We first provide a description of the ingestion pipeline and how our methodology applies to *Smog Analysis Service* and *Traffic Monitoring Service*.

Ingestion pipeline. The *Smog Analysis* and *Traffic Monitoring* services share the same ingestion pipeline that includes video processing and metadata enrichment discussed in Example 4.1. Data sources and ingested data are annotated as in Example 4.1. The outcome of this ingestion procedure is stored in the city data lake to be used by the two services independently. We note that, though the pipeline is the same, it is executed by the two different services independently thus providing the separation of duties characterizing our approach. Each service implements its own set of access control policies with different data transformations, leading to different enforcement results that are at the basis of our data governance approach.

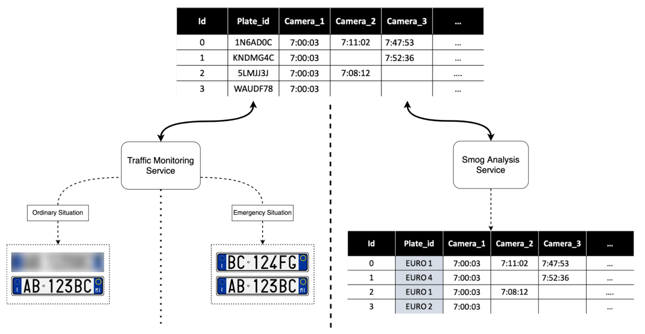
Traffic Monitoring Service. This service monitors accesses to low emission zones and issues a fine when an unauthorized access is registered by a vehicle not belonging to the set of authorized license plates (e.g., ambulances, residents, public transport). The service implements a simple visualization pipeline loading data after ingestion from the lake and showing vehicles metadata details. Such details are obtained using the plate number extracted from the video with a relative set of snapshots, with the scope of showing them to the service user (i.e., a member of the local police department). For this service we consider two scenarios: *i)* ordinary and *ii)* emergency, having different policy sets. Ordinary is the typical execution scenario where the authorized accesses are anonymized as in Example 4.2, while unauthorized access are identified and plate number kept in clear to later issue a fine. In an emergency scenario, restrictions have to be relaxed: a different policy applies without the application of the anonymization transformation, allowing service users to get all the details of vehicles that entered the Low Emission Zones during emergency. We note that the data lake contains plain data just for the emergency period, when the emergency is over the ordinary policies are restored. In the following we present the ordinary and emergency policies, respectively, as described by Apache Ranger with the form ⟨`subject`, `action`, `object`, `context`, `data transformation`⟩:

⟨`LocPolDept`, `READ`, `Pii-id`, {`LEZ_Ordinary`}, `OK_PlateAnonym`⟩
⟨`LocPolDept`, `READ`, `Pii-id`, { `LEZ_Emergency` }, ∅ ⟩

Smog Analysis Service. The Smog Analysis Service combines information about air quality and traffic counting to create deeper insights about the connection between traffic regulation and air pollution. It uses the same ingestion pipeline, but the policy associated with the smog service is more restrictive in terms of privacy (e.g., video completely removed). In addition, the transformation applied to the plate number is not a suppression like in the case of the Traffic Monitoring Service, but a replacement with the corresponding pollution class of the vehicle. In this case, the policy is of the form:

⟨`SmogAnalyst`, `READ`, `Pii-id`, - , `Plate2EmissionClass`⟩

Figure 4 shows our case study with the two services in action and an excerpt of the ingested data. It is clear that both services can be implemented using an ad hoc end-to-end analytic approach; however, the level of data governance, as well as the flexibility provided, are not be comparable. With our solution, there is a common data governance based on an access control system, while in an ad hoc solution each service should be monitored for data governance compliance. In addition, a city-level data governance strategy can be applied to all the services or to the whole smart city ecosystem by simply changing or adding new policies, while in case of an ad hoc solution it requires to change every single service impacted by the new data governance strategy.



Id	Plate_id	Camera_1	Camera_2	Camera_3	...
0	1N6AD0C	7:00:03	7:11:02	7:47:53	...
1	KNDMG4C	7:00:03		7:52:36	...
2	5LMJJ3J	7:00:03	7:08:12		....
3	WAUDF78	7:00:03			...

Id	Plate_id	Camera_1	Camera_2	Camera_3	...
0	EURO 1	7:00:03	7:11:02	7:47:53	...
1	EURO 4	7:00:03		7:52:36	...
2	EURO 1	7:00:03	7:08:12		....
3	EURO 2	7:00:03			...

Figure 4: Data transformations for Traffic Monitoring and Smog Analysis services.

5.2 Performance analysis

We also present some preliminary performance results. We deploy the above system in our big data engine having one single node made of a Ubuntu 20.04.2 LTS machine with Intel Xeon E5-2620 - 2.095GHZ CPU and 32 GB of RAM. To evaluate the performance impact of our methodology, we used the Mock Traffic dataset (https://www.kaggle.com/mathfour/mock-traffic-data) and extended it generating other entries to reach 20000 records. We used the policies and services defined in this section and executed the ingestion procedure measuring the time needed to ingest data with and without policy enforcement. Figure 5 shows the performance results. We note that in the experiment without policies Atlas and Ranger were not fully disabled, therefore communication and notifications of incoming data between them were considered. The performance difference observed in Figure 5 represents just the policy enforcement time. The latter was not impacted by an increasing the number of records due to the scaling capabilities of our architecture. We note that the enforcement time is reasonably negligible given the reduced set of policies and the limited effort requested by the transformations specified in the policies.

6 RELATED WORK

As organizations see practical results and significant value in the usage of big data, they also recognize the limits of current big data ecosystems with respect to data governance and data protection. Recently, both industry and academic community started to investigate the issue, both from a data governance perspective [2, 3] or recognizing the need of new security requirements [10]. Although Attribute Based Access Control (ABAC) [17] is currently adopted in big data projects as a common underlying model given its ability to support highly flexible and dynamic forms of data protection to

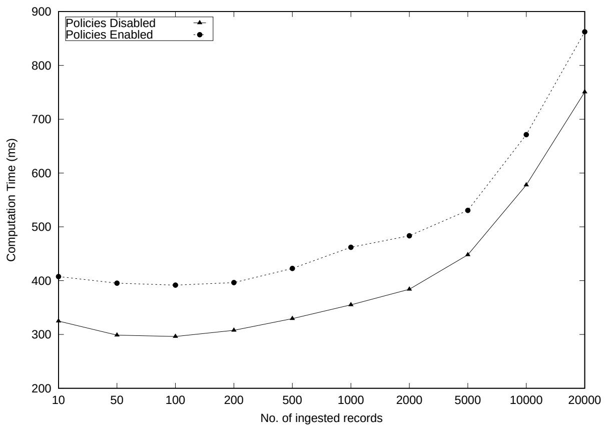


Figure 5: Preliminary performance evaluation.

business-critical data, the dynamic and decentralized nature of big data asks for new solutions. Actual solutions are neither general nor scalable, since they are either platform-dependent or coarse-grained [10].

Platform-specific approaches are designed for one system only (e.g., MongoDB, Hadoop) and leverage on native access control features of the platform [4, 18]. However, Hadoop access control presents some limitations, such as the complexity of deployment and the consumption of resources. Some recent proposals, like Federated Access Control Reference Model (FACRM) [6] or [14, 15], are specifically tailored to the Apache Hadoop stack. On the other hand, platform-independent approaches have the advantage of being more general than platform-specific solutions. However, the currently available platforms either model resources to be accessed as monolithic files (e.g., Microsoft DAC) or lack scalability. In addition, existing database-oriented approaches mainly leverage on recent research efforts for defining a unifying query language for NoSQL datastores (e.g., JSONiq and SQL++) [9], thus only focusing on a particular type of analytic pipelines.

We believe that the closest approach to the one in this paper is the work of Hu et al. [16], introducing a generalised access control model for big data processing frameworks, which can be extended to the Hadoop environment. However, the paper discusses the issues only from a high-level architectural point of view, without discussing a tangible solution. Another relevant work is by Xue et al. [22]. They propose a solution based on the notion of purpose-aware access control [8] that, although focusing only on Apache Spark, recognises the need of a generalised approach to deal with access control in analytics pipelines.

7 CONCLUSIONS

Big data platforms have been designed mainly with performance and scalability requirements in mind and no proper considerations have been given to data governance issues. In this paper, we tackled the data governance problem in big data scenarios focusing on the data protection problem. Following a data-centric perspective, we first discussed the complex structure of big data pipelines that should be secured as a whole, and identified the key requirements of an access control system dealing with the peculiarities of big data scenarios. Then, we proposed an access control methodology based on an advanced ETL schema where data transformations are performed at ingestion time to comply with security policies. We showed the feasibility of the approach with a case study in a smart city scenario that uses an Apache-based big data engine.

ACKNOWLEDGMENTS

Some of the work reported here is supported by funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 883286, project "IMPETUS".

REFERENCES

[1] The general data protection regulation (gdpr) - regulation (eu) 2016/679.
[2] M. M. B. Aissa, L. Sfaxi, and R. Robbana. DECIDE: A new decisional big data methodology for a better data governance. In *Proc. of EMCIS*, Dubai, EAU, November 2020.
[3] A. Al-Badi, A. Tarhini, and A. I. Khan. Exploring big data governance frameworks. *Procedia Computer Science*, 141:271–277, 2018.
[4] M. Anisetti, C. Ardagna, V. Bellandi, M. Cremonini, F. Frati, and E. Damiani. Privacy-aware big data analytics as a service for public health policies in smart cities. *Sustainable cities and society*, 39:68–77, 2018.
[5] F. Armknecht, C. Boyd, C. Carr, K. Gjøsteen, A. Jäschke, C. A. Reuter, and M. Strand. A guide to fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2015:1192, 2015.
[6] F. M. Awaysheh, M. Alazab, M. Gupta, T. F. Pena, and J. C. Cabaleiro. Next-generation big data federation access control: A reference model. *Future Generation Computer Systems*, 108:726–741, 2020.
[7] X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In *Proc. of CRYPTO*, Santa Barbara, USA, August 2006.
[8] J.-W. Byun and N. Li. Purpose based access control for privacy protection in relational database systems. *The VLDB Journal*, 17(4):603–619, 2008.
[9] P. Colombo and E. Ferrari. Towards a unifying attribute based access control approach for nosql datastores. In *Proc. of IEEE ICDE*, San Diego, USA, April 2017.
[10] P. Colombo and E. Ferrari. Access control technologies for big data management systems: literature review and future trends. *Cybersecurity*, 2(1):3, 2019.
[11] Y. Demchenko, P. Grosso, C. de Laat, and P. Membrey. Addressing big data issues in scientific data infrastructure. In *Proc. of IEEE CTS*, Brisbane, Australia, April 2013.
[12] Y. Demchenko, F. Turkmen, C. de Laat, C.-H. Hsu, C. Blanchet, and C. Loomis. Chapter 2 - cloud computing infrastructure for data intensive applications. In *Big Data Analytics for Sensor-Network Collected Intelligence*, Intelligent Data-Centric Systems, pages 21–62. Academic Press, 2017.
[13] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proc. of ACM CCS*, Alexandria, USA, November 2006.
[14] M. Gupta, F. Patwa, and R. Sandhu. An attribute-based access control model for secure big data processing in hadoop ecosystem. In *Proc. of ACM ABAC*, 2018.
[15] M. Gupta, F. Patwa, and R. S. Sandhu. Object-tagged rbac model for the hadoop ecosystem. In *Proc. of. DBSec*, Philadelphia,USA, July 2017.
[16] V. Hu, T. Grance, D. Ferraiolo, and D. Kuhn. An access control scheme for big data processing. In *Proc. of IEEE CollaborateCom*, Miami, USA, October 2014.
[17] V. C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone. Guide to Attribute Based Access Control (ABAC) Definition and Considerations. *NIST special publication*, 800(162):1–54, 2014.
[18] M. M. Rathore, A. Paul, A. Ahmad, M. Anisetti, and G. Jeon. Hadoop-based intelligent care system (hics) analytical approach for big data in iot. *ACM Transactions on Internet Technology*, 18(1):8:1–8:24, 2017.
[19] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Proc. of EUROCRYPT*, Aarhus, Denmark, May 2005.
[20] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.
[21] T. W. Shinder, Y. Diogenes, and D. Littlejohn Shinder. *Chapter 7 - Controlling Access to Your Environment with Authentication and Authorization.* Syngress, 2013.
[22] T. Xue, Y. Wen, B. Luo, B. Zhang, Y. Zheng, e. Hu, Y. Li, G. Li, and D. Meng. Guardspark++: Fine-grained purpose-aware access control for secure data sharing and analysis in spark. In *Proc. of ACM ACSAC*, page 582âĂŞ596, Online, December 2020.