Title

Marco Anisetti *Senior Member, IEEE,*, Claudio A. Ardagna *Senior Member, IEEE,* Alessandro Balestrucci, Chiara Braghin, Ernesto Damiani *Senior Member, IEEE,*, Antongiacomo Polimeno

Abstract—The conflict between the need of protecting and sharing data is hampering the spread of big data applications. Proper security and privacy assurance is required to protect data owners, while proper data access and sharing are fundamental to implement smart big data solutions. In this context, access control systems assume a central role for balancing the need of data protection and sharing. However, given the software and technological complexity of big data ecosystems, existing solutions are not suitable because they are neither general nor scalable, and do not support 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 with a case study in a smart city domain using an Apache-based big data engine.

Index Terms—Access Control, Big Data, Data Transformation, Data Ingestion

1 Introduction

TBW

2 Requirements and System Model

2.1 System Model

Abbiamo coalizione di servizio da un lato (ad esempio comuni che offrono servizi alla cittadinanza) e coalizione di utenti (cittadini, security operator) che accedono alla pipeline dall'altro.

Coalizione dove ogni servizio della pipeline è eseguito potenzialmente da utenti e piattaforme differenti.

Dati potenzialmente condivisi sulla piattaforma in visione Open Data.

Ad ogni passo verso un servizio in coalizione di un utente diverso ci potrebbe essere la necessità di un'ingestion MEDES-style.

NOTA: le pipeline sono completamente instanziate, la variabilità è sull'utente (soggetto) come in medes

NOTA2: perché abbiamo bisogno di AC ad ogni task?

- alcuni parametri di configurazione o task da eseguire potrebbero cambiare a seconda della specifica esecuzione, ad esempio scelgo il k attraverso elbow method, oppure ho un processo airflow che contiene alternative e variabili di decisione
- in una coalizione un task può essere fornito da provider differenti con dominio di appartenenza diverso e impatto diverso sulla pipeline e sull'AC. Ad esempio, un task fornito dalla corea del nord è a rischio

- *M. Anisetti, C.A. Ardagna, E. Damiani, are with the Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy. E. Damiani is also with.*
E-mail: {firstname.lastname}@unimi.it

2.2 2.2 Requirements

Rivisitazione dei requisiti di MEDES in ottica pipeline, coalizione etc.

2.3 Reference Scenario

IMPETUS

3 Access Control Policy Language

3.1 Access Control Policy

Il dato annotato (abbiamo applicato Medes in fase di ingestion) arriva a un servizio, il nostro access control component deve prendere una decisione rispetto alla richiesta che possiamo esprimere come: <utente, esegui servizio, dato annotato, contesto, trasformazione>

- esegui servizio è troppo specifico allora dovremmo identificare categorie di servizi a diverse criticità
- soluzione marco e chiara andare a basso livello e vedere come vengono trattati i dati (map reduce), claudio lavora ad alto livello ad esempio guardando il tipo di classificazione che si fa (M/F oppure eterosessuale/omossessuale)
- altro?
- potremmo definire come calcolare o selezionare manualmente la criticità di un servizio istanziato trasformando la politica in
- <utente, esegui servizio a criticità X, dato annotato, contesto, trasformazione>

Il dato può essere trasformato a seconda:

- tipo e annotazione
- utente
- criticità del servizio
- contesto

In certi momenti la trasformazione potrebbe essere deanonimizza, quindi dobbiamo stare attenti all'ordine di applicazione della politica (XACML -> priorità, ad esempio

politica emergenza priorità alta quindi fa overwrite delle altre)

- <utente, esegui servizio a criticità X, dato annotato, contesto, trasformazione, priorità>
- **DA DECIDERE SE SERVE LA PRIORITA' O BASTA SOLO LA POLITICA SENZA ORDINAMENTO (CHIARA/CLAUDIO SI STANNO CONVINCENDO CHE LA PRIORITA' NON SERVA). IL MOTIVO ORIGINARIO PER CONSIDERARE LA PRIORITA' ERA**
  - DA DECIDERE: se ad ogni connessione S1–>S2 applichiamo delle trasformazioni multiple o no? Facciamo prima ingestion medes e poi trasformazione sul servizio o è un passo solo? Il passo multiplo potrebbe essere utile per utenti diversi nella coalizione
    - ingestion+servizio
    - pipeline: {servizio ingestion+servizio analitica}$_n$

3.2 Annotations

Data annotation presa da MEDES

Task annotation: Come modelliamo la criticità del servizio? (dati in ingresso e tipi del servizio devono essere indipendenti)

- Dato annotato in modo indipendente
- Servizio ha criticità indipendente dal dato. Possiamo pensare che la criticità dipenda da delle proprietà del servizio che evidenziano relazioni tra input e output esempio Trasparenza e interpretability
- Bisogna differenziare tra predizione (classificazione binaria o più il tipo di classificazione va considerato) e modellazione
- task provider

3.3 Data Transformation

Presa da MEDES e raffinata. Possibili trasformazioni

In certi momenti la trasformazione potrebbe essere deanonimizza, quindi dobbiamo stare attenti all'ordine di applicazione della politica (XACML -> priorità, ad esempio politica emergenza priorità alta quindi fa overwrite delle altre)

4 Quality vs Privacy

Definire e modellare metriche quantitative che indichino quanto dato è rimasto rispetto all'originale ingestion time

- Sono metriche sui dati manipolati. Ad esempio, numero di feature soppresse, numero di datapoint soppressi. Queste due potrebbero essere pesate su feature ranking.

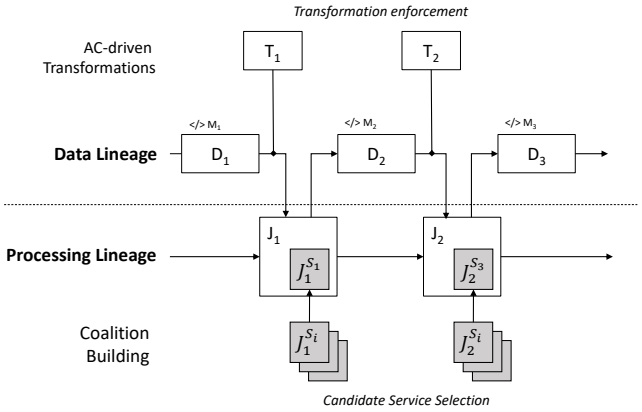

Fig. 1. Methodology.

5 Coalition Building

Data lineage is a set of complex relationships between datasets an jobs in a pipeline. The scope of this paper is to add control on the data lineage differentiating data flow from processing pipeline and controlling them separately: i) data controlled using AC and transformations prior to be feed into a processing jobs, ii) processing controlled selecting the suitable processing jobs among a set of available and compatible alternatives offered by different service provider S.

In this paper, for the sake of simplicity we assume that the transformation on the data made by the processing jobs are known and not influenced by the service provider behaviour, trusting the service provider to produce the correct transformation requested by the given job. With this assumption the coalition selection is assumed to be equivalent to select different service provides for the different jobs and thus different transformation policies to be applied on the data.

Example 5.1. *Let us consider the following example where we have a pipeline made of just one ingestion job that can be offered by service provider s_1 or by the service provider s_2. In case the s_1 is selected the transformation T_1 is triggered according to the authorization s_1 has on the data, in this example s_1 has full control meaning that transformation t_1 is empty. In case the s_2 is selected the transformation T_2 is triggered according to the authorization s_2 has on the data and in this example data labelled as PII are removed.*

Considering the two data lineage generated by the two different coalition in Example5.1 the one involving s_2 produce a significant changes to data compared to the other one. This data changes can have direct impact on the quality of the analytics outcomes, therefore our goal is to build coalitions ensuring specific data quality. This coalition building problem can be assimilated to xxx showing an exponential complexity ... In the following we fist introduce our data quality metrics and then our euristics to solve the problem of coalition building

5.1 Data Quality metrics

5.2 Coalition Heuristics

Execute a template where for each task a set of functionally equivalent alternatives provided by different service are available.

Con la coalizione, vogliamo mettere per ogni nodo della pipeline servizi equivalenti forniti da utenti diversi e tra cui scegliere? Se sì, definire le euristiche per la selezione con identificazione del servizio a qualità maggiore.

- come calcolare la qualità al punto 1? Vedi sezione 4.
- in ogni caso l'indicatore di qualità serve anche se non vogliamo scegliere il servizio ottimo e non abbiamo alternative. Modella un parametro importante del nostro AC. Ad esempio se abbiamo trasformazioni alternative

6 Architectural Deployment

Parliamo dei due tipi di deployment

COMMENTO (architectural deployment): noi stiamo ragionando usando una coreografia, autocrmap ragionava usando una orchestrazione con un'entità centrale che forniva sandboxing, come se noi avessimo il nostro approccio centralizzato e i servizi vengono di volta in volta a prendersi il dato

7 Experiments

8 Related Work

9 Conclusions

DISCUTERE CON MARCO POSSIBILI ANALOGIE SECURESCM: ad esempio i servizi non sono alternativi ma sono delle piccole coalizioni a loro volta.

(ri)Vedere anche Autocrmap per vedere analogie.

References