PrivComp: A Privacy-aware Data Service Composition System

Mahmoud Barhamgi
Université de Lyon. CNRS
Université de Lyon 1. LIRIS. UMR5205
69622 Villeurbanne, France
mahmoud.barhamgi@univ-lyon1.fr

Djamal Benslimane
Université de Lyon. CNRS
Université de Lyon 1. LIRIS. UMR5205
69622 Villeurbanne, France
djamal.benslimane@univ-lyon1.fr

Youssef Amghar
Université de Lyon. CNRS
INSA-Lyon. LIRIS. UMR5205,
F-69621. France
youssef.amghar@insa-lyon.fr

Nora Cuppens-Boulahia
Mines-Telecom/Telecom-Bretagne
2 Rue de la Chataigneraie
35576 Cesson Sevigne, France
nora.cuppens@telecom-bretagne.eu

Frederic Cuppens
Mines-Telecom/Telecom-Bretagne
2 Rue de la Chataigneraie
35576 Cesson Sevigne, France
frederic.cuppens@telecom-bretagne.eu

ABSTRACT

In this demo paper, we present a new privacy preserving composition execution system. Our system allows to execute queries over multiple data services without revealing any extra information to any of the involved services. None of involved services (and their providers) is able to infer any information about the data the other services provide beyond what is permitted

Categories and Subject Descriptors

K.2.0 [**COMPUTERS AND SOCIETY**]: Public Policy Issues—*privacy*; H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Theory

Keywords

Privacy, Data services, composition

1. INTRODUCTION

Recent years have witnessed a growing interest in using Web services as a reliable means for data publishing and sharing among enterprises [2]. This new type of Web services is known as *Data Services*, where services correspond to calls (i.e., parameterized queries) over *business objects* (e.g. *Customer*, *Product*, etc) in underlying data sources. Most of the time user's queries require the composition of multiple autonomous data services. For example, in the healthcare domain, conducting research studies involves the analysis of a huge amount of data collected from various data services provided by healthcare providers, pharmacies, research institutions, etc. Given the sensitive nature of the accessed information and the social and legal implications of its disclosure [4], privacy becomes a major concern for data service composition.

In this demo paper, we propose a novel privacy preserving data service composition scheme and an implementation thereof. Our composition system allows to execute queries over multiple data services without revealing any extra information to any of the involved services (i.e., none of involved services (and their providers) should be able to learn/infer any information about the data the other services provide beyond what is permitted). Our system is not intrusive, it assumes that services implement locally their privacy policies before integrating their data.

1.1 Running Example

Consider the following scenario from the healthcare domain. Assume that a pharmaceutical researcher, *Alice*, wants to investigate the connection between a chemical component *ABC* present in HIV medicines and the development of severe psychiatric disorders at HIV female patients. *Alice* needs to combine information from autonomous sources including HIV healthcare centers, psychiatric hospitals, pharmacies and pharmaceutical labs. For the sake of clarity, assume that the data services in Table-1 are available to *Alice*.

Obviously, *Alice* can answer her research questions by composing these services as follows (refer to Fig. 1). She invokes S_1 with the desired city to get the identifiers of HIV patients. Then for each obtained *ssn*, she verifies whether the patient has psychiatric disorders by invoking S_2; then

EDBT/ICDT '13 Mar 18-22 2013, Genoa, Italy

Table 1: Available Data Services

Service	Semantics
$S_1(\$city, ?ssn)$	*Returns the SSN of HIV patients in a given city.*
$S_2(\$ssn, ?description)$	*Returns a description of the psychiatric disorder of a given patient if she/he has any.*
$S_3(\$ssn, ?age, ?sex)$	*Returns the age and sex of a given patient.*
$S_4(\$ssn, \$type, ?medication)$	*Returns the medications of a given type taken by a given patient*
$S_5(\$medication, \$ingredient, ?quantity)$	*Returns the quantity of a given ingredient in a given medication.*

for each of these patients she retrieves the age and sex by invoking S_3 and the HIV medications by invoking S_4. For each of the obtained HIV medications, she retrieves its ABC content by invoking S_5. Then she joins the outputs of S_3 and S_5 to link the medical and the personal information to the same patient.

1.2 Challenges

If the data returned by individual services were completely privacy-sanitized (by removing identifiers and anonymizing sensitive information) then the composition could not be executed, as the input parameter *ssn* required to invoke S_2, S_3 and S_4 will no longer be provided to these services. On the other hand, if returned data were not protected, then participant services and the query issuer (i.e. *Alice*) will learn sensitive information that they must not know. For instance, if the provider of S_2 knows that its input tuples are coming form S_1, and assuming that the data accessed by our services are given in Fig. 2, then he will learn who of his patients have been tested positive for HIV (i.e, P_{15}, P_{201} and P_{512}). Similarly, the providers of S_3 and S_4 will learn who of their patients are receiving treatments for psychiatric disorders and are HIV patients. *Alice* and the entity responsible for executing the composition (which we call the *composition execution engine* in the rest of the paper) will learn sensitive information about patients including their ssn, ages, medications, etc. Based on this observation, the main challenge we address is how to enable services involved in a composition to enforce locally their privacy policies while at the same time keeping it possible to answer queries that require linking data subjects[1] (e.g., patients) across autonomous services. This challenge implies the two following requirements: (i) the knowledge leaked to a service S_i (denoted by $\Re(S_i)$) about the data held by another service S_j in the composition must be less than a threshold defined by S_j; (ii) the composition execution engine as well as the final data recipient must not have access to any individually identifiable information.

2. A PRIVACY-PRESERVING COMPOSITION EXECUTION MODEL FOR HONEST-BUT-CURIOUS DATA SERVICES

2.1 Context and Assumptions

In this work, we made the following assumptions.

We consider a honest-but-curious environment. An honest-but-curious environment (a.k.a. semi-honest environment [3]) is one where the parties involved in the query processing (i.e., composed data services and the composition execution engine) follow correctly the given protocol, but may keep any result or information they obtain during the course of the protocol. We assume that the services, the composition execution engine and the recipient of the final results are three independent entities.

[1]We use the term data subject to mean the individual whose private information is stored and managed by data services

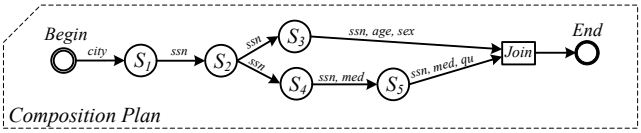

Figure 1: The Composition Plan

We consider that the attributes of a dataset can be divided into: *identifier attributes* and *non-identifier attributes*. The integration of the data subjects across the different data services is carried out using the identifier attributes. We assume the existence of universal identifiers in each application domain (e.g., the social security number in the healthcare domain).

2.2 Preliminaries

A Composition of Data Services $\mathcal{H}$: A composition of n data services is represented as a directed acyclic graph (DAG) $\mathcal{H}$ in which there is a node corresponding to each data service, and there is a directed edge e_{ij} from S_i to S_j if there is a precedence constraint $S_i \prec S_j$ (i.e., S_j is preceded by S_i when one of its inputs is an output of S_i), and where each service S_i 1≤ i≤ n has a set of inputs and outputs that could be privacy-sensitive or identifier attributes. Edges e_{ij} may be associated with constraints to filter relayed tuples.

Service Selectivity $Se(S_i, R_j)$: Given a data service S_i, and a range of input values R_j, the selectivity of S_i relative to R_j is the number of outputted tuples when S_i is invoked with R_j. For example, assuming that the *ssn* values in Fig. 2 are ordered, then $Se(S_3, [P_0, P_{10}]) = 2$, $Se(S_3, [P_5, P_{20}]) = 3$, and $Se(S_3, [P_0, P_{1000}]) = 13$ are the selectivities of S_3 relative to the ranges $[P_0, P_{10}]$, $[P_5, P_{20}]$ and $[P_0, P_{1000}]$. We assume that data services can provide operations (i.e., functions) to provide statistical information about their managed data (including the selectivity of a service.).

Order Preserving Encryption Scheme OPES. An OPES [1] allows to encrypt numeric data values while preserving the order relation between them. This allows to apply equally and range queries as well as the MAX, MIN and COUNT queries on encrypted data, without decrypting the operands.

2.3 Privacy-preserving Composition Execution Model

Our model relies on two key ideas. First, we use a combination of OPES for identifier attributes and anonymization techniques for non-identifier attributes. Composed services could apply the desired anonymization algorithms on non-identifier attributes, but they must all use the same OPES for identifier attributes. This way the composition execution engine has only access to anonymized data and can link the anonymized information of the same data subject across the different services using the encrypted identifier attributes (recall that the OPES allows for applying equality queries on encrypted data). It cannot decrypt the encrypted identi-

The data accessed by S_1

city	ssn
villeurbanne	P_{15}
villeurbanne	P_{201}
villeurbanne	P_{512}

The data accessed by S_2

ssn	description
P_3	...
P_{11}	...
P_{15}	...
P_{16}	...
P_{21}	...
P_{32}	...
P_{201}	...
P_{199}	...
P_{300}	...
P_{510}	...
P_{512}	...
P_{571}	...
P_{575}	...

The data accessed by S_3

	ssn	age	sex
	P_0	[0-10]	m
	P_8	[0-10]	m
t_1	P_{15}	[0-20]	null
	P_{20}	[0-20]	f
	P_{23}	[0-20]	f
	P_{188}	[0-20]	m
t_2	P_{201}	[10-15]	null
	P_{204}	[10-15]	m
	P_{209}	[10-15]	m
	P_{411}	[0-10]	m
t_3	P_{512}	[15-20]	f
	P_{513}	[0-30]	f
	P_{514}	[10-15]	f

The data accessed by S_4

ssn	medicine	type
P_5	dox	cardiac
P_{15}	alpha1	HIV
P_{17}	dox	cardiac
P_{227}	drab	cardiac
P_{201}	alpha 2	HIV
P_{242}	drab	cardiac
P_{411}	dox	cardiac
P_{512}	alpha 3	HIV
P_{711}	dox	cardiac

The data accessed by S_5

	medicine	quantity
l_1	alpha1	[0-30] mg
l_2	alpha2	5 mg
l_3	alpha3	[10-100]mg

Figure 2: Sample of the data accessed by the data services

fier attribute values, as it does not have the encryption key. By the end of the composition's execution, it removes from the final results the encrypted identifier attributes before returning them to the recipient, who will thus get only the anonymized data.

Second, our model implements the *K-protection* notion that we introduce below, and which limits the knowledge leaked to participant services during the execution of $\mathcal{H}$.

***K*-protection**: *Given a vector $K = (k_1, k_2, ..., k_n)$, where k_i is an integer representing the protection degree the service S_i must provide for its outputted tuples. For each edge e_{ij} in $\mathcal{H}$, the knowledge leaked to S_j during the execution of $\mathcal{H}$ (denoted by $\Re(S_j)$) must be $\leq min(1/k_l)$, where k_l is associated with S_l, which denotes the (direct or indirect) parents of S_j in $\mathcal{H}$. Note that S_j has at least one parent in $\mathcal{H}$, which is S_i.*

The above definition can be interpreted as follows: when a service S_j is invoked, it must not be able to determine precisely its input value between k input values for which it has outputs; i.e., it must not be able to determine precisely the tuple t in which the invoker is interested between k tuples of its own data. This can be realized by invoking S_j by a range of values R instead of a precise value v, where $Se(S_j, R) = K$.

Example: Examples of privacy breaches that could happen if the composition in Fig. 1 was executed without ensuring the k-protection requirement include: S_2 will know that its patients P_{15}, P_{201} and P_{512} have AIDS; S_3 will know that these same patients have AIDS and suffer from severe psychiatric disorders, etc. Now, assume that $K_1 = 3$, the k-protection requirement implies that S_2 must not be able to distinguish each of its input values (e.g., P_{15}) from at least 3 values for which it has answers. Fig. 3 shows how the k-protection is enforced on the edge e_{12}. The value P_{15} is *k-generalized* into a range of values V which has at least three values (e.g., P_{11}, P_{15} and P_{16}) for which S_2 has corresponding tuples. After the invocation of S_2, the extraneous tuples are filtered out.

The Model Description: Fig. 4 gives an overview of our proposed composition system. The figure is self-describing, we therefore focus only on its main components. The recipient specifies an encryption key, submits it directly to participant services in $\mathcal{H}$, and launches the execution of $\mathcal{H}$. When participant services are invoked, they anonymize their sensitive data and encrypt the identifiers with the supplied

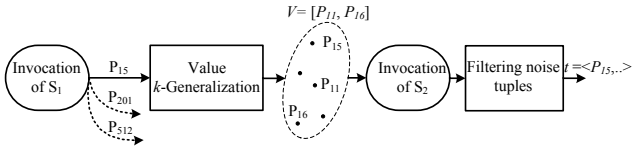

Figure 3: Ensuring the *k*-protection on the edge e_{12} (k=3)

key. The composition execution engine implements (in the *Value K-Generation module*) an algorithm to ensure the k-protection requirement when it invokes participant services. Specifically, for each invoked service S_i, it determines the protection factor k that must be ensured: $k = \text{MAX}(S_j.k_j)$, where S_j denotes the parents of S_i in $\mathcal{H}$. Then, for each input tuple t, the algorithm determines the minimum range of value $R[a, b]$ that should be used to invoke S_i instead of t. For this purpose the execution engine requests the selectivity of S_i with respect to a wide range of identifier values R (we use the range $]-\infty, +\infty[$ to denote the range covering the whole tuples set managed by S_i) along with a value v occurring in the middle of the ordered value sets held by S_i. Then if the returned selectivity is greater than k, the execution engine compares the identifier attribute (denoted by x) of t to v to determine the half of R covering t, which becomes the new range R. This step is repeated with the new R until there is no R with a selectivity greater than k. Then, S_i is invoked[2] with the obtained range, and the execution engine retains only the output related to t.

Example: Fig. 5 shows how the k-protection requirement is enforced on the edge e_{23}. Assume that S_1 and S_2 require a protection factor $k = 3$. The invocation of S_2 returns the tuples corresponding to c_{15}, c_{201} and c_{512} (i.e., the encrypted values of P_{15}, P_{201} and P_{512}). Instead of invoking S_3 directly with the tuple c_{15}, the execution engine *k-generalizes* c_{15} as follows. It requests the selectivity of S_3 with respect to $R =]-\infty, +\infty[$; S_3 acknowledges it has 13 distinct values and that the value ($v = c_{199}$) occurs in the middle of these ordered values set. The execution engine compares c_{15} to c_{199}, and determines the new range $R =]-\infty, c_{199}]$. It then requests

[2]We assume that data services provide different operations to query the underlying data sets by precise values or by ranges of values.

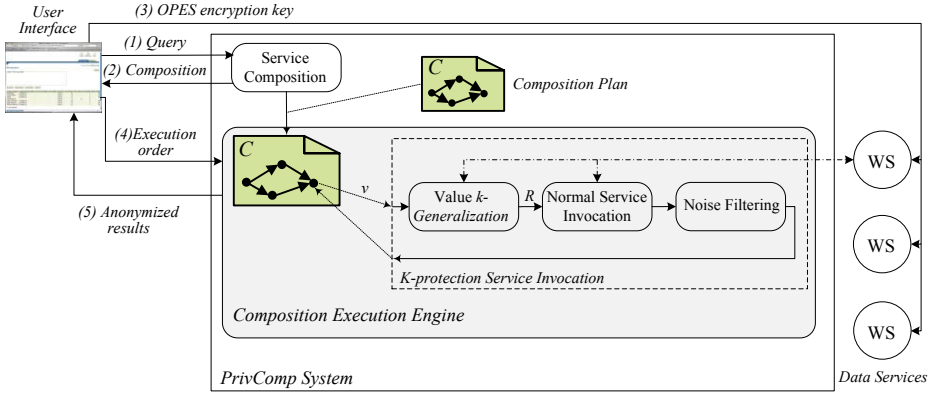

Figure 4: The architecture of the PrivComp system

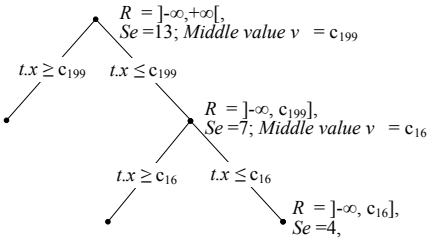

Figure 5: Finding the minimum range for invoking S_3

the selectivity of the new R along with the new v; the new values of Se and v are 7 and c_{16}. It determines again the new range by comparing c_{15} to c_{16}. The new range is R=]-∞, c_{16}] and its selectivity is 4. The algorithm stops here as if the new range was divided then Se will be less than k.

3. DEMONSTRATION OUTLINE

Our PrivComp system (Fig. 4) is composed of two main modules: the *Service Composition Module* which generates the composition execution plan and the *Composition Execution Module* which executes the composition in a privacy-preserving manner. We implemented the system in Java and evaluated thoroughly its performance on a set of 400 medical data services, managing the medical information of more than 30.000 patients. We highlight below along with the demonstration description our obtained findings. Based on our experimental evaluations, in our demonstration we illustrate the following processes:

A. Composition Plan Generation: We describe our demonstration scenario as follows:

1. The user is *interactively* assisted by the system to formulate his query over a domain ontology. The query is expressed in SPARQL.
2. The system rewrites the query (by the *Service Composition Module*) in terms of calls to relevant data services, generates the composition plan and displays it.

Evaluation Findings: The rewriting module scales very well, it can rewrites complex queries (containing 10 ontological concepts) in the presence of 400 data services in less than 1 second.

B. Privacy Preserving Composition Execution:

1. The system provides the user with an interface to the composition.
2. The user supplies, through his interface, an encryption key that will be relayed directly to participant services without passing by the execution engine. He triggers also the execution of the composition.
3. The execution engine executes the k-protection algorithm when it proceeds with services invocations.

Evaluation Findings: For all of the tests conducted, the time required to execute the composition with privacy preservation is at most three orders of magnitude of the time required without privacy preservation (K_i was set to 4 in all tests). We cut down further that cost to two orders of magnitude by *reusing* the selectivities and ranges computed in past invocations of the same services (and during the same composition execution).

4. ACKNOWLEDGMENTS

This research work is funded by the French National Research Agency under the grant number ANR-09-SEGI-008.

5. REFERENCES

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order-preserving encryption for numeric data. In *SIGMOD Conference*, pages 563–574, 2004.
[2] M. J. Carey, N. Onose, and M. Petropoulos. Data services. *Commun. ACM*, 55(6):86–97, 2012.
[3] F. Emekçi, D. Agrawal, A. E. Abbadi, and A. Gulbeden. Privacy preserving query processing using third parties. In *ICDE*, page 27, 2006.
[4] T. C. Rindfleisch. Privacy, information technology, and health care. *Commun. ACM*, 40(8):92–100, 1997.