“Share your data, Keep your secrets.”

Irini Fundulaki
Bell Labs Research
fundulaki@lucent.com

Arnaud Sahuguet
Bell Labs Research
sahuguet@lucent.com

1. MOTIVATION AND OVERVIEW

The next generation of services will not be restricted to the boundaries of a given network. This is called convergence! Land-line telephony, wireless telephony, instant messaging, Web, etc. now form a converged network where applications can be deployed.

An example of such an application is the *Selective Reach Me (SRM)* which makes it possible for me (resp. other people) to reach people (resp. me) wherever they are (resp. I am). On the caller side, instead of calling a specific device, I call a person and I let the application figure out the best way to contact her. On the callee side, I want to specify who can contact me, on which device, when and for what purpose, etc.

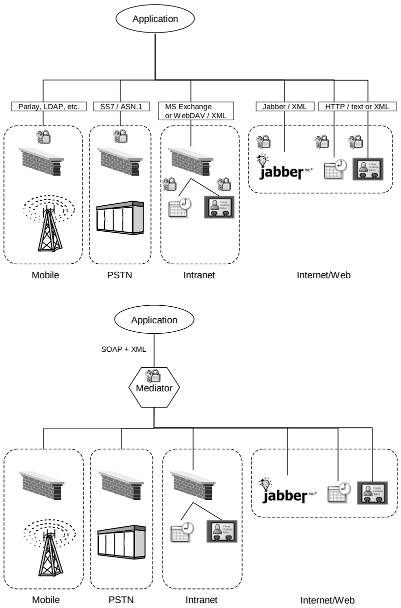

Figure 1: Before GUP$^{\mathsf{ster}}$ (top) and after.

To make the above scenario possible, it becomes critical to provide a ubiquitous access to *user profile information* which (a) is distributed across networks and devices and (b) consists of static (e.g. identity information) and dynamic data (e.g. IM and wireless presence) and cannot be “warehoused”.

Probably, even more importantly, is the critical need to control the way this information is accessed. The user is willing to disclose part of her profile information to certain users, but only if she can be sure that her information is not accessible by unauthorized parties.

The problems with the current architecture are that: (i) user profile information lives in network components with different protocols, data models, APIs. Consequently, applications need to deal with such heterogeneities which make the aggregation of information hard (if not impossible) and very expensive; (ii) access control is all-or-nothing and is spread across the various sources. The difference is illustrated by Fig. 1.

2. OUR APPROACH

Numerous industry initiatives like Microsoft Passport [4] and Liberty Alliance [3] have been started to address the issue of user profile data management. We describe here the GUP$^{\mathsf{ster}}$ system which is motivated by the 3GPP Generic User Profile (GUP) effort [1], a telecom-based initiative that aims to aggregate user profile information relevant to network operators.

GUPster in a nutshell

The main idea behind GUP$^{\mathsf{ster}}$ is to build an XML-based mediator that acts as a *centralized meta-data manager* to handle *highly distributed user profile XML data.* The mediator acts as the *single point of access* between data producers and data consumers. GUP$^{\mathsf{ster}}$ aims to be the broker for user profile components which are (a) distributed across networks and (b) their distribution varies on a per user basis. Its role is two-fold: data integration and access control. This is the major difference with traditional mediator-based systems that only address the former.

One language to rule them all

The key component behind GUPster is the XSquirrel language (see [2] for a full description of the language). The language can express (1) views over XML documents for integration (i.e. mappings between sub-documents of the user profile document and remote sources), (2) views over XML documents to describe access control (i.e. association between sub-documents and boolean predicates) and (3) queries over these documents.

SIGMOD 2004 June 13-18, 2004, Paris, France.
Copyright 2004 ACM 1-58113-859-8/04/06 . . . $5.00.

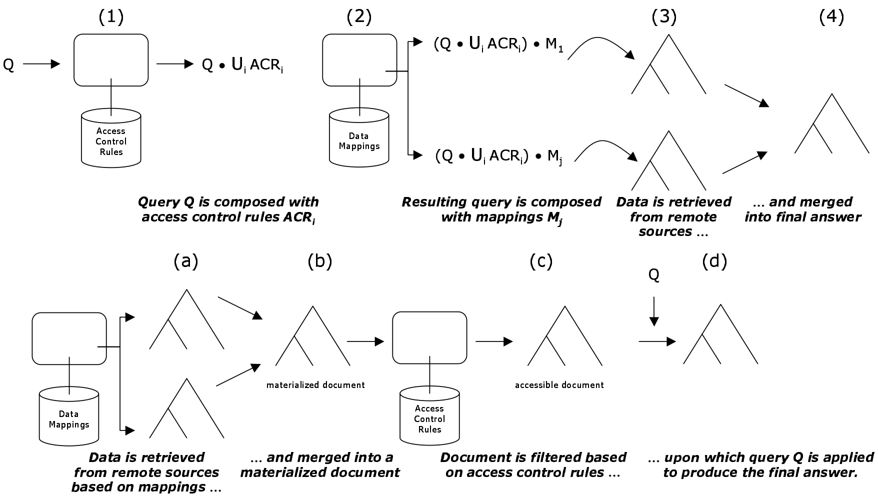
(1)
(2)
(3)
(4)
Q
Q • U$_i$ ACR$_i$
Access Control Rules
Data Mappings
(Q • U$_i$ ACR$_i$) • M$_1$
(Q • U$_i$ ACR$_i$) • M$_j$
Query Q is composed with access control rules ACR$_i$
Resulting query is composed with mappings M$_j$
Data is retrieved from remote sources ...
... and merged into final answer
(a)
(b)
(c)
(d)
Q
Data Mappings
materialized document
Access Control Rules
accessible document
Data is retrieved from remote sources based on mappings ...
... and merged into a materialized document
Document is filtered based on access control rules ...
... upon which query Q is applied to produce the final answer.

Figure 2: GUP$^{\text{ster}}$ Processing Flow vs traditional flow.

The processing of an incoming query q against a set of access control rules ACR and some mappings M (all expressed as XSquirrel expressions) can be described as:

$$\boxed{q' = (q \circ \ (\bigcup_{xsq}^{i=1..m} ACR_i)) \ \circ M_i}$$

where $\bigcup_{xsq}$ is the union and $\circ$ the composition operators for XSquirrel expressions. See [2].

3. OUR PROTOTYPE

Our current prototype is implemented in Java. It performs the rewriting for both data integration and access control, using some primitive operations for our XSquirrel language (intersection, union, and composition of trees). Mappings and access control rules are both stored in a relational engine.

The processing flow is illustrated in Fig. 2. First we identify the relevant access control rules, and then we compose the query with their union (1). We then compose the rewritten query with relevant mappings to produce a query plan (2). Individual queries are sent to the various data sources (3). If needed, components retrieved from the various data sources are merged together (4).

This is to be contrasted with the traditional approach where the virtual document is built by fetching its components (a) and merging them; then filtered based on access control rules to produce the accessible document (c), before the query is applied to it to produce the final answer (d).

Both approaches are compared in Fig. 2.

We currently support the following data sources, which we think are good representatives of different networks.

- address book and calendar information (from Microsoft Exchange, via WebDAV protocol)
- presence information (from Jabber server)
- personal information (from Lucent LDAP directory)
- presence and location (simulated from HLR data).

We have written wrappers to export the data of the above sources as XML data, compliant with the GUP$^{\text{ster}}$ schema.

4. OVERVIEW OF THE DEMO

For this demo, we will demonstrate:

- how to register user profile components (coming potentially from multiple sources) in the GUPster server.
- how to add/delete/modify access control policies (using the provisioning client).
- how the meta information dictates the result of incoming queries, based on registered user profile components, access control policies, and request context.
- how GUPster can be used by applications and services via the SOAP interfaces.

More specifically, we will present 3 possible instances of GUP data consumers that we have built:

- personal web portal implemented on the server-side, including personal information, calendar, presence, etc.
- personal web portal implemented on the client-side (making SOAP calls from Javascript using the Mozilla SOAP API).
- A device with limited capabilities (e.g. PDA or cell phone) using GUPster to synchronize some user profile information.

5. ADDITIONAL AUTHORS

Guillaume Giraud, Nicola Onose, Nicolas Pombourcq (Ecole Polytechnique)[1] and Daniel Lieuwen (Bell Labs, Lucent Technologies)[2].

6. REFERENCES

[1] 3GPP. http://www.3gpp.org.

[2] I. Fundulaki and A. Sahuguet. Privacy Conscious User Profile Data Management with GUPster. Technical report, Bell Laboratories, Lucent Technologies, 2003.

[3] Liberty Alliance. http://www.projectliberty.org.

[4] Microsoft Passport. http://www.passport.net.

[5] A. Sahuguet, R. Hull, D. Lieuwen, and M. Xiong. Enter Once, Share Everywhere : User Profile Management in Converged Networks. In *CIDR*, Asilomar,California, USA, January 2003. Online Proceedings.

[1] {pombourcq,onose,giraud}@polytechnique.org

[2] lieuwen@research.bell-labs.com