EFFICIENT POLICY-BASED ACCESS CONTROL IN WEB-BASED PERSONALIZATION MANAGEMENT
For Use in Convergent Applications

Heinz-Josef Eikerling
Siemens SBS, C-LAB, Fürstenallee 11, D-33 102 Paderborn, Germany

Keywords: Personalization, Distributed Data Management, Profiles, Security.

Abstract: Personalisation of web-centric user environments and applications to contextual information, as well as to the needs and preferences of the user is a contemporary concern. The involved personalisation data is usually captured by profiles and is sensitive to security concerns. Therefore mechanisms need to be devised to maintain privacy and data integrity, and to ensure proper access control. Within this paper we describe a system which handles access control by the definition of policies. The access control engine is a key component within the set of interacting services making up the profile access manager. Through this approach the according profiles can be potentially dispersed over different types of physical storage devices and access control can be featured quite independently of the actual storage type. Particularly, the integration of the different storage devices is facilitated which permits to bind locally stored profile data to data stored elsewhere (due to space or constraints with respect to accounting of profile information). We thus believe that the system is particularly applicable for convergent personalisation scenarios in which web-based personalisation data has to be combined with local information stored on mobile devices for nomadic use.

1 INTRODUCTION

With the tremendous increase of devices of different type being dynamically associated with a user, there is an increasing demand for advanced semi-automatic configuration functionalities to be applied to the user's environment. This is especially important with respect to convergent scenarios in which a nomadic user roams between located services delivered through mobile or stationary terminals. If the configuration process mainly targets the user (i.e., the physical person accessing a service), the configuration process is frequently referred to as *personalization*.

In order to facilitate this, some configuration data (configuration rules, attributes, etc.) has to be created, accessed and persistently maintained. In most practical cases, this profile data can be dispersed over different physical storages, like for instance databases, configuration files or more advanced and proprietary types of storages (like for instance tamper-resistant Smart Cards). The major reasons for the dissemination of this information over such storages are:

- Different *levels of security* with respect to accessing and modifying profiled information (e.g., identity related information) have to be supported.
- Different *types of profile information* have to be maintained and managed by different principals depending on what is inevitably required to be locally available, especially in case of temporary off-line situations.
- The *storage capabilities* for a specific device might be too limited to host the entire relevant set of profile information.

Within this paper, we describe a system permitting to handle the above aspects through the use of profiles which can be physically dispersed over networked storage resources. Basically, in our approach a profile is regarded as a repository of structured data that has an influence on sets of *configurable entities*. We propose a light-weight, platform- and site-spanning service infrastructure called Customization Management System (CMS) to handle the distribution of profile data as well as the

Eikerling H. (2006).
EFFICIENT POLICY-BASED ACCESS CONTROL IN WEB-BASED PERSONALIZATION MANAGEMENT - For Use in Convergent Applications.
In *Proceedings of WEBIST 2006 - Second International Conference on Web Information Systems and Technologies - Internet Technology / Web Interface and Applications*, pages 369-374
DOI: 10.5220/0001257003690374
Copyright © SciTePress

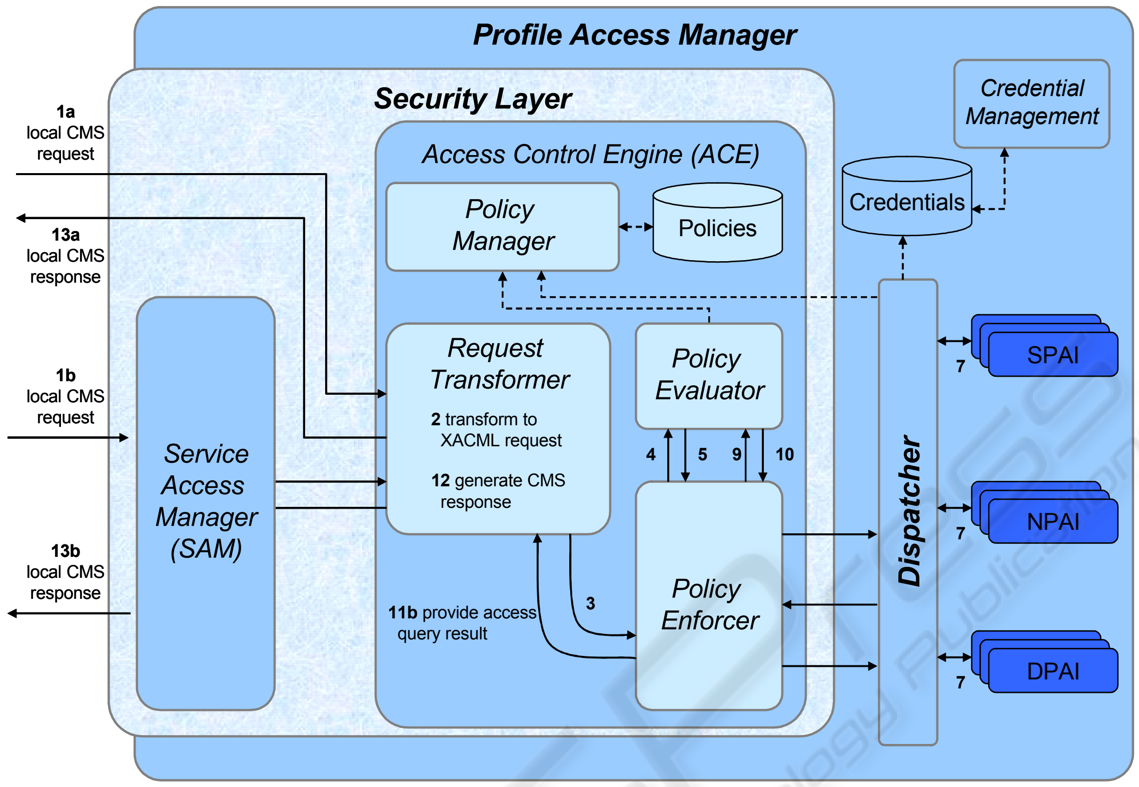

Figure 1: Service access manager and access control engine.

implications of applying that data to the configurable entities.

2 BACKGROUND

There are several formats available for the description and exchange of profiles like CC/PP, Dublin Core, PIDF, GUP, etc. However, they are all not widely employed yet. Moreover, when applied to mobile environments comprising devices like PDAs and mobile phones (or Smart Cards) offering limited processing power and low bandwidth communication capability, they seem to be not well suited as a complete profiling standard. Therefore we have defined a suitable set of different XML-based profile formats (Eikerling, 2005) taking into account the above constraints.

In general profiles can be regarded as a special type of distributed data which needs to be put under the regime of a decentralized data management (Perich, 2004). Currently, the data dissemination aspects are mostly considered in terms of persistent data being hosted by databases (Bukhres, 1997). Under these circumstances, a profile can be viewed as a collection of continuous location dependent data queries (Cherniak, 2003). In order to handle the case of information dispersal, this rather centralized model needs to be further extended.

3 REQUIREMENTS

Due to space constraints, we can only give a summary of the key requirements for the customization management system and refer to the major CMS service ingredients. The design of those takes into account the following requirements:

- *Decentralization*. As mentioned earlier, profile data might be distributed over different physical storages. Hence it should be possible to link profiles, to migrate and replicate profiled data when and where needed. This has a major impact on the data formats and the protocols to be used which both have to support extensibility and broad platform coverage.
- *Diversification*. Different storages (also referred to as profile bearers) can hold the profile information. The data management has to be able to (preferably dynamically) integrate these bearers and make them accessible through the service interface. Ideally, the profile storage devices can

be federated through the CMS service.

- *Connectivity*. With respect to nomadic scenarios, temporal disconnection from central communication services offering access to certain profile information may occur. This requires to support nomadic user mobility which has specific consequences for the enforcement of security mechanisms, i.e., access to locally available profile information should still be possible.
- *Low complexity*. This mainly addresses the service interface which should insulate from the complexity of data distribution, heterogeneity of communication and data management mechanisms.

4 CMS PROFILE FORMAT

In our approach, a profile is considered to be a data container which can be split into two main parts:

- *Data payload*: this is the data uniquely characterizing the configurable entity (e.g., name, first name etc. of a user). Properties, as the most basic profile data elements, are vectors of property name and value with an associated type pairs. It has to be ensured that the name of a property is unique within a profile. The value part of a property can be anything ranging from simple data types to complex data structures.
- *Meta-data*: this is information describing the format of the data, for instance whether it can be linked to other data (e.g., profiles) or information concerning its origin, location, history etc. The meta-data is used for the processing of the profile information within the customization management system.

A configurable entity can be a user, hardware (i.e., a device or device configuration) or a software component (e.g., application or service), an organisation (e.g., a company), a network, or the environment in which such user is acting (context data). Thus, we consider different profile domains like user, device, service, application, network and context profiles.

Each type of domain profile has to support a certain set of generic operations: lifecycle operations (create, delete, replicate), editing operations (composing, linking, modifying, and moving), retrieval (lookup) and advanced operations (versioning, synching, inheritance, evolution). Additionally, there are domain-specific operations.
The profiles may reside on different types of storages reflecting the different levels of security. For example, device characteristics might be stored on a networked resource, whereas identity information as part of the user profile has to be retrieved from tamper-resistant smart cards.

A key feature of the CMS is the ability to logically link sets of profile instances to each other in order to reflect coherent relations between them. The respective information model supporting this has to be mapped to the according bearers (database schema, standard or proprietary file system on a storage device etc.) in the PAIs (Profile Access Interfaces). For the bearer-independent handling of profile information inside the CMS and in the applications accessing the CMS we have foreseen a set of XML schema definitions specifying the domain profiles. This format is also used as a bearer independent representation of results returned as a response to access queries requesting profile sets or profile fragments. Moreover, the network profile access interface (NPAI) employs this format for persistently storing profile information.

5 CMS ARCHITECTURE

The CMS constitutes a light-weight service infrastructure supporting advanced personalization through the use of distributed profiled information. The approach is based on a simple layered protocol featuring a request / response mechanism. For integrating different CMS instances, the service dispatches profile access requests either to local resources or to distant resources; this is achieved through a portable URI-based addressing scheme which is handled through the core CMS component (see Fig. 1), the Profile Access Manager (PAM).

Quite naturally, the PAM is required to feature widely accepted communication mechanisms and formats; it can be either integrated as a library or by utilizing the provided *CMS port* together with a secured transport protocol. For the latter, an accessing entity has to know the CMS address made up of the host name and the CMS port, i.e. one host can run multiple CMS instances. The SSL/TLS-based server could be addressed by

```
https://www.ubisec.org/cms?message=[b64c]
```

where `[b64c]` constitutes a base64 encoded CMS request.

The developed request format features a *query part* that is used to specify access commands. These are coded by means of a access and manipulation language. For routing requests to the according PAI, a *resource part* (containing schema definition, addressed authority, and path information) is foreseen. Both, resource and query part together form an access query:

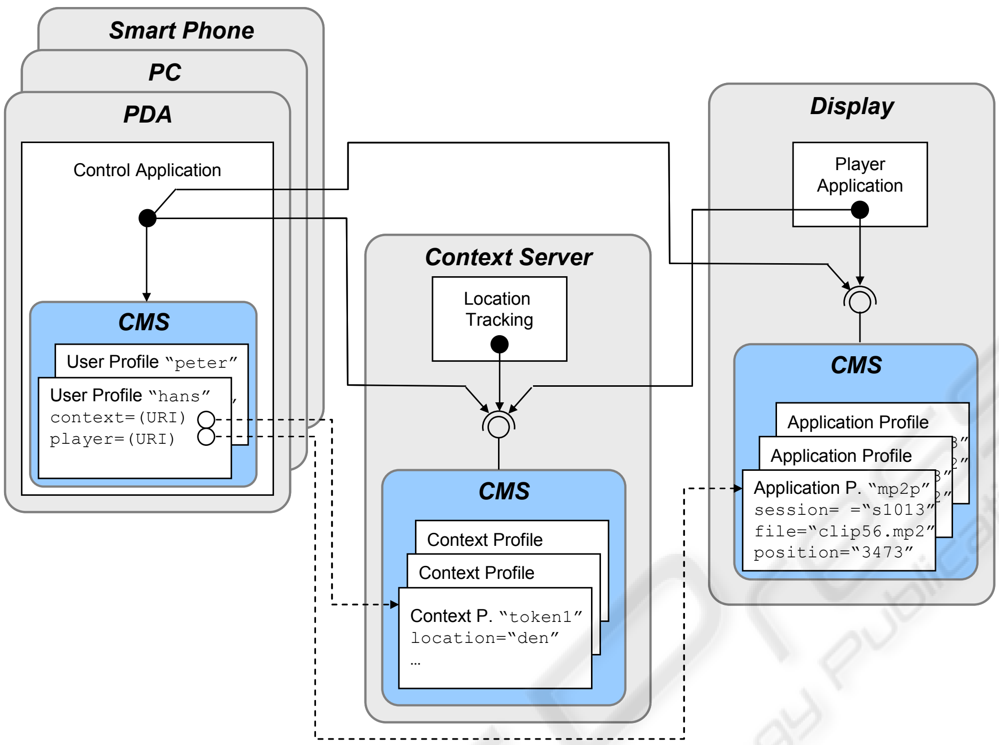

Figure 2: Mobile display scenario.

```
<scheme> ":" "//" <authority> <path>
"?" <query>
```

A sample access query to obtain certain user preferences would look as follows:

```
file:///C:/CMS/CMSProfiles/
container/ContextProfile/sampleCo
ntext.pc.xml?path=Lw==&action=cmV
hZA==&relativePath=L0NvbnRleHRQcm
9maWxlL3NhbXBsZUNvbnRleHQucGMueG1
s&subject=cm9vdA==
```

To avoid problems with masking parameters, values following the "=" symbol are in fact base64 encoded to form valid queries.

Concerning the Security Layer of the PAM, the service access manager (SAM) is responsible to ensure message integrity checking, decryption, encryption, decoding, encoding, authentication, and handling of client or server side certificates. For establishing trust relationships between CMS instances and applications accessing them, it can be bundled with an enhanced PKI (ePKI) solution which targets trust management in decentralized environments (Almenárez, 2004).

The other part of the Security Layer is constituted by the Access Control Engine (ACE). Here we refer to XACML policies (Moses, 2005) and an according implementation. The Transformer in the Security Layer parses incoming access requests and generates a common request format, i.e., a transformation from the UBISEC native CMS request format into an XACML compliant request (Step 2). This supplies the Access Controller with an access request that comes in a common, semantically well-defined format, regardless of profile type, requesting subject and accessed resource. The steps 3 – 5 and 8 – 11a are in accordance with XACML access control by means of policy evaluation.

In Step 7, the CMS Dispatcher sends a request to one of the PAIs in order to acquire profile data that are necessary for policy evaluation. The Dispatcher takes care about routing of requests to the correct PAI. The different kinds of PAIs are SPAIs for Smart Cards, DPAIs for data bases, and NPAIs for networks. Note here that a single PAM can manage a number of PAIs of the same kind. Each PAI then transforms the request into the (native) format that is understood by the resource storing the profile data (e.g., a query in case of a database or an APDU command in case of a Smart Card).

Assume now that the policy evaluation terminated with a result, such that we have an access response generated by the Access Controller (Step 11b). If the XACML evaluation result (Step 10) concludes that access is denied, the Access Controller returns an according 'deny' response. If access is permitted, this in turn may require the

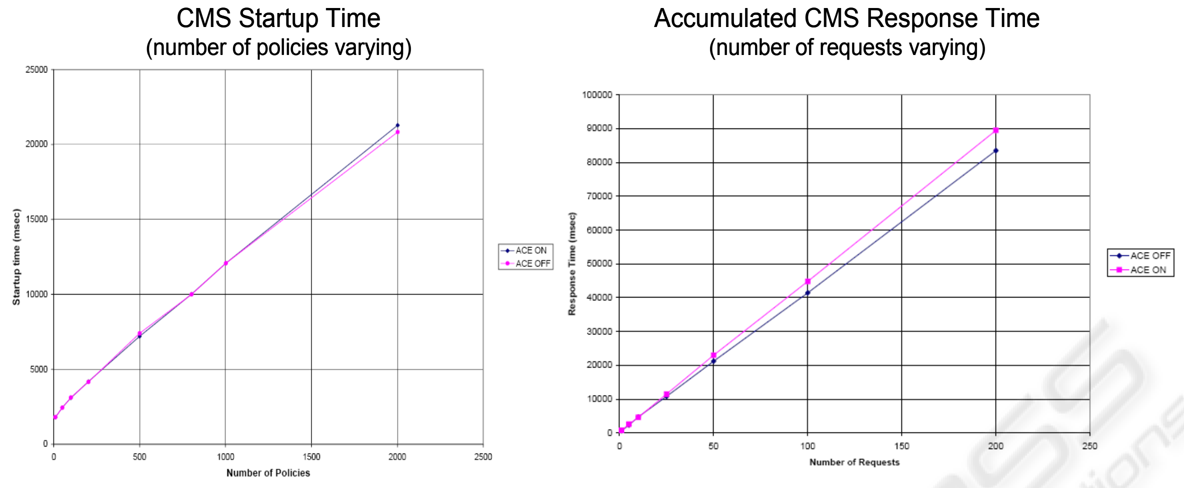

Figure 3: CMS ACE performance.

execution of obligations that have to be fulfilled before access is granted. A possible obligation could be that an additional notification has to be sent to some other entity (e.g., a profile administrator) or that some security obligations have to be enforced. The Policy Enforcer takes care for these obligations. Moreover, the access response may include a data payload, in particular when a 'read' action was specified in the CMS request.

6 APPLICATION SAMPLE

The CMS is a basic means for profile management and various actual application scenarios can be thought of. A specific class of scenarios that is frequently examined is the context-aware (location-dependent) delivery of digital content to nomadic users. Within our setup, we have focused on users roaming between different discrete, distinct sites / locations while accessing streamed media. In the particular prototype system, the transmission of MPEG-2 streams is examined. Depending on the context of the users, the media is delivered to them. For accessing the media and for controlling the environment, either a mobile or a stationary access device can be used. During a session, the handoff / handover among the access devices can be featured.

Pertaining to the personalisation of the environment of a nomadic user and according to the scenario described above, the following functions and data sets (see Fig. 2) have to be supported:

- *User management*: profile data concerning the user (e.g., identity related information like user name "hans") and references to other types of profiled information (e.g., to some context "token1" and application data "player") relevant to the user have to be maintained.
- *Context management*: the location contexts have to be managed, i.e., the user has to be tracked. The tracking system managing the symbolic user locations (e.g., "den") is physically detached from the mobile access device.
- *Application / service management*: the replay of media streams via the delivery service might be interrupted (for instance, if the user's context changes). For this, the status of the playing session (i.e., file name and current positioning within file) should be maintained, so that the settings can be restored once the user roams into the range of the service again.

The above functional needs clearly motivate a demand for distributing the customization data as shown in Fig. 2. The advantages of this approach over more traditional (e.g., centralized) ones are:

1. *Autonomous acting*: by partitioning and distributing the functionalities and the required customization data, the interference and dependability between the involved peers is minimized.
2. *Privacy / anonymity*: the location tracking only deals with abstract position names. The mapping of the symbolic names to the identities of the users being mapped to these names can be only done via the user profile mastered by the user.
3. *Linking*: linking to profile data from within a profile is essential to enable such features as anonymity. Through a URI-based request format, different domain profiles can be easily linked which facilitates complex personalization tasks.

7 RESULTS

With respect to the above scenario, the efficient handling of context information is paramount to the overall performance of the system. We particularly examined the dependency of the request processing time on the authorization (which is handled by the ACE) and authentication (handled by the SAM) mechanisms. We found that:

- There is virtually no dependency on the authentication handled by the SAM.
- When considering profiles delivered as complete XML files through the NPAI, with respect to security a certain tribute has to be paid to authorization through the ACE.
- The accumulated response time is pretty independent of the number of policies and the number of requests issued over a certain time period as can be seen from Fig. 3. The overhead of ACE request validation amounts to approx. 7% of the overall request processing time.
- When having different runs of the same access pattern (forming an iteration) we found that just after one iteration the ACE gets ‘used’ to this pattern, i.e., within forthcoming repetitions the of the particular pattern, the accumulated response is constant (within certain limits).

Through its ease of use, the CMS features some interesting characteristics. For instance, aside from user mobility as described in the application scenario described above, it is also possible to realise terminal mobility (or access device mobility) as shown in Fig. 2. The playing of a clip can be stopped on one device and can be resumed on another device with virtually no user intervention through the use of personalisation features offered by the CMS.

8 CONCLUSION

We have presented an approach to the secured data management and dissemination of profile data over different types of physical storages. This particularly includes the transparent maintenance of a user’s context depending on device, application and environmental settings as well as user preferences and capabilities. The described concepts are especially applicable to decentralised environments as opposed to pure centralised environments with unlimited connectivity to a profile server (e.g., data base) in which authorization decisions can be handled more efficiently though by the penalty of centralisation.

We believe that the distribution of profile data is an inevitable requirement for convergent scenarios which has to be adequately dealt with. So far, rather centralized approaches and proprietary solutions only targeting specific aspects of personalization / customization (i.e., tied to a specific operating system) exist. The design of the CMS is a step to approach the distribution of profile data from the perspective of nomadic users. For a wider dissemination and proliferation of the proposed mechanisms, a standardization of the concepts, protocols and formats is quite essential and efforts to promote this have been launched.

ACKNOWLEDGEMENTS

The work described herein is funded by the FP6-2002-IST-1 project UBISEC (Ubiquitous Networks with a Secure Provision of Services, Access, and Content Delivery), contract number 506926.

REFERENCES

Eikerling, H-J. et al. (2005, June). Customization of Secured Ubiquitous Environments via Advanced Profile Management, *IST Mobile & Wireless Communications Summit, Dresden.*

Perich, F. et al. (2004, May). On Data Management in Pervasive Computing Environments. *IEEE Transactions on Knowledge and Data Engineering*, 16(5):621-634.

Bukhres, O. et al. (1997). A Proposed Mobile Architecture for Distributed Database Environment. Technical report, Indiana University, Purdue University.

Q. Ren and M. Dunham (2000, August). Using Semantic Caching to Manage Location Dependent Data in Mobile Computing. *6th Annual International Conference on Mobile Computing and Networking (MobiCom’00)*, Boston, MA, USA, pp. 210-221.

M. Cherniack, E. Galvez, D. Brooks, M. Franklin, and S. Zdonik (2003, March). Profile-Driven Cache Management. *Proceedings of the 19th International Conference on Data Engineering (ICDE)*, Bangalore, India.

F. Almenárez, A. Marín, C. Campo, C. García-Rubio. PTM (2004). A Pervasive Trust Management Model for Dynamic Open Environments. *Workshop on Pervasive Security, Privacy and Trust (PSPT 2004)*, Boston, USA.

T. Moses (ed.) (2005). OASIS eXtensible Access Control Markup Language (XACML) Version 2.0, OA-SIS Standard.