Proceedings of the IASTED International Conference
Communication, Network, and Information Security
December 10-12, 2003, New York, USA

EC-GATE: AN INFRASTRUCTURE FOR DRM

Antonio Maña, Mariemma Yagüe, Vicente Benjumea
Computer Science Department. University of Málaga
{amg, yague, vicente}@lcc.uma.es
Spain

Abstract

DRM technologies include a range of functions to support the management of intellectual property for digital resources, such as expression of rights and obligations, description, identification, trading, protection, monitoring and tracking of digital content. This paper presents the EC-GATE system, a general framework capable of supporting very heterogeneous DRM applications and scenarios. This system enables content owners to enforce access control policies, copyright agreements, payments and other obligations, to digital objects in a distributed environment. The idea of this work is that the security requirements of all processes related to the secure transmission and commerce of digital contents can be fulfilled if we guarantee that the software running at the other side of the communication line is protected. To achieve what we call "protected software" we must ensure that it is neither possible to discover nor to alter the function that the software performs and it is also impossible to impersonate the software. The solution that we present is also based on the notion of "secure container", a protected package of data and administrative information. Our solution uses mobile software elements to convey the protected contents and force the user to fulfill the obligations previously established by the content owner before granting the rights and access to these contents. EC-GATE also includes components for authorization and management.

Key Words

Digital Rights Management, Access Control, XML metadata

1. Introduction

Usually, the content industries consider DRM to deal with unauthorized downloading of copyrighted material, a practice that makes content creators and distributors to loose a huge amount of profits. However, an important and often overlooked fact is that DRM is closely related to the general field of Access Control. In the end, rights enforcement involves an access decision about a resource subject to intellectual property rights. The problem is that current access control models are not appropriate for the DRM and other open, heterogeneous and dynamic scenarios. The main reason for this situation is that Access Control is often erroneously considered to apply to "locations" instead of "objects" or "resources". In this way, it is assumed that one or a few access control (enforcement) points are used to restrict access to a set of resources in one "location". Moreover, our diagnostic is that the main problem with current access control models is that the model is built on predefined concepts: "user", "role" and "group". The definition of roles and the grouping of users can facilitate management, especially in corporation information systems, because roles and groups are easily identifiable and fit naturally in the context of the organizational structures of the companies. However, when applied to some new and more general access control scenarios, these concepts are somewhat artificial.

We believe that a more general approach is needed for these new environments. For example, in the referred situations, groups are an artificial substitute for a more general tool: the attribute. In fact, groups are usually defined based on the values of some specific attributes (employer, position, …). Some attributes are even built into most of the current access control models. This is the case of the user element; the identity is just one of the most useful attributes, but it is not necessary in all scenarios and, therefore, it should not be a built-in component of a general model. Finally, access control models must take into account that the creation and maintenance of access control policies (or DRM policies in our case) is a difficult and error prone activity. Therefore, these models must be designed to facilitate and guarantee the correct administration of the system.

2. Description of the EC-GATE Infrastructure

Most DRM systems are very complex because the underlying framework tries to capture all possible details and features of the targeted scenarios, resulting in difficult to understand and inflexible solutions. In the development of the EC-GATE infrastructure we take into account these issues and consider as main objectives flexibility, extensibility and interoperability. For getting this, the component responsible of the access control, rights and

obligations enforcement, is based on a new access control model suitable for highly dynamic, open and heterogeneous scenarios, with very large numbers of users, resources and stakeholders. This model is the Semantic Access Control, *SAC* [1]. On the other hand, content protection mechanisms are also an essential component of DRM systems. Our proposal is also concerned with this need and we propose to use programmable tamperproof devices such as Java smart cards to achieve "persistent protection". The solution for content protection in this system is based on the *SmartProt* system [2].

2.1 Access Control

One of the building blocks of the EC-GATE system is devoted to control the access, and enforce the rights and obligations bound to the contents. This component implements the Semantic Access Control (SAC) model, which is based on the semantic properties of the contents to be controlled, properties of the clients that request access to them, semantics about the context and finally, semantics about the attribute certificates trusted by the access control system. The *SAC* model has been implemented on the basis of the *Semantic Policy Language* (SPL) to specify the access control criteria, and the semantic integration of an external authorization entity [3].

Opposed to other languages, SPL *policy* specifications do not include references to the target content. Instead, a separate specification called *Policy Applicability Specification* (PAS) is used to relate access control policies, rights and obligations to contents dynamically when a request is received. Both SPL policies and PAS use semantic information about resources included in *Secured Resource Representation* (SRRs) and other contextual information documents, which is an original contribution. SPL policies and PAS can be parameterized allowing the definition of flexible and general policies and reducing the number of different policies to manage. Parameters are instantiated dynamically from semantic and contextual information. Finally, policies can be composed importing components of other policies without ambiguity. This compositional approach allows us to define the abstract meaning of the elements of the policies, providing a mechanism to achieve abstraction, which also helps in reducing the complexity of management. Tools to graphically manage the relations among policies and DRM expressions and with other components are also essential for a simple and flexible management.

The independence of the attribute certification function is the key to the interoperability because it allows attributes to be safely communicated avoiding the necessity of being locally issued by the system administrator. Additionally, this approach avoids the registration phase of the client, and the emission of a client attribute repeatedly for each access control system. For this approach to be secure, a mechanism to establish the trust between these access control systems and the authorization entities has been developed using semantic information about the certifications issued by each authorization entity, and represented as *Source Of Authorization Description* (SOAD) documents.

One of the main characteristics of the *SAC* model is that, in contrast to traditional schemes, the attributes required to access a resource may depend on the semantic properties of the resources. The allocation of the policy corresponding to a resource is not based on the storage structure of the resources but on the semantic properties of the resources. Of course, it is also possible to consider the structure of storage.

In summary, *SAC* facilitates the management of the access control system, while guaranteeing the simplicity, correction and safety of the system. No other work provides the semantic validation of the access control criteria.

2.2 Content Protection

A DRM system must also deal with digital content protection. In EC-GATE this protection is based on the production of self-protected software objects that convey contents (software or data) and can be distributed without further security measures because they embed the access control, rights and obligations enforcement mechanisms.

Two important issues arise when considering content protection of digital objects: the secure content distribution mechanism itself and the persistent protection issue. The first one must ensure that contents are protected so that only the intended recipients can access them. In the case of DRM it also entails other requirements such as the need to bind the execution of digital rights agreements, payment or other obligations to the access to the contents. This is known as provisional authorization or provision-based access control (PBAC) [4]. The second one deals with enabling owners of the contents to retain control over them even when contents are stored in external distrusted hosts.

Our solution to the previous problems is based on the use of secure active containers. A secure active container [5] is a piece of protected mobile software that conveys the contents and forces the user to fulfill the applicable access control, rights and obligations policies before access is granted. By "protected software" we mean that it is neither possible to discover nor to alter the function that the software performs and it is also impossible to impersonate the software. In our scheme, this is achieved using a variant of the *SmartProt* system [2]. *SmartProt* partitions the software into functions that are executed by two collaborating processors. One of those processors is a trusted computing device that enforces the correct

execution of the functions and avoids that these functions are identified or reverse engineered. We are currently using smart cards for this purpose although other alternatives are possible. Our secure active containers are implemented as JavaTM applets that we call Protected Content Objects (PCOs). They include the contents to be accessed (which are encrypted), the access control, rights and obligations enforcement mechanisms, and a cryptographic link to the Mobile Policy (MP) required to gain rights and access to the contents. Each MP is specific for a smart card. We extend the concept of mobile policy described in [6] by allowing their execution in untrusted systems. Moreover, in our solution access control, rights and obligations policies are bound to the content but not integrated with. This modification makes possible that policies are dynamically changed in a transparent manner. The definition of the MP structure allows a high degree of flexibility.

The PCO generation process is independent of the customer card and will be performed just once for each piece of content. PCOs can be distributed and copied freely. One important constraint to the free distribution of protected contents in our system is that originators of those contents must be able to dynamically change the applicable access control, rights and obligations policies regardless of the storage location of the PCO. In order to fulfill this requirement, MP and PCO must be separated. In this way, the MP is retrieved from the originator server during the execution of the PCO. Requesting the MP at access time from the originator slightly reduces the performance of the system but, in return, it allows a high degree of flexibility and gives the originator more control over the application of the policies. To improve the efficiency and flexibility we have included validity constraints in MPs that can be used to control the need for an online access to the originator server. As a result, originators can define certain validity constraints for each MP (based on number of accesses, time, etc. depending on the smart card features). Hence, MPs can be cached by clients and used directly while they are still valid. As each PCO has its own key, we can manage them individually, which is not possible in other software protection proposals where all applications are protected using the same key.

Figure 1 shows the execution of the PCO. When the client requests some content from a server, it receives the PCO containing it. Before the PCO can execute the protected sections of its code it has to retrieve the corresponding MP by sending a request containing the certificate of the public key of the client smart card. In case the server from where the PCO was retrieved is the originator of the PCO, it produces the MP for that PCO. When the MP is received by the client smart card, it is decrypted, verified and stored inside the card until it expires or the user explicitly decides to extract it.

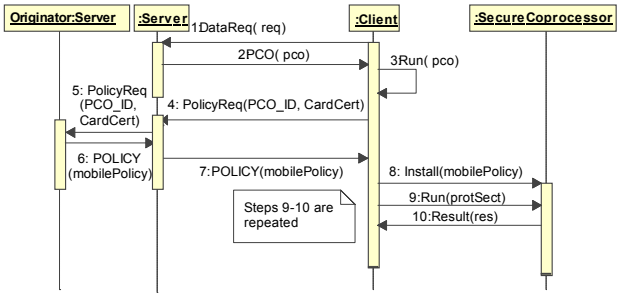

Fig.1 A scenario for content access

Once the MP is correctly installed in the card the protected sections of the PCO can be executed, which requires the cooperation of the card containing the MP. The protected sections of the software do not reside in the cards. Instead, during the execution of the PCO, these sections are transmitted dynamically as necessary to the card, where they are decrypted using the installed MP and executed. When finished, the card may send back some results. Some other partial results will be kept in the card in order to obtain a better protection against function analysis and other attacks. A more detailed description of SmartProt can be found in [7].

A very important advantage of this scheme is that it can be extended to work in other scenarios such as audio and video reproduction in independent player devices where software containers are not appropriate.

2.3 Fulfillment of Obligations

The main problem when trying to use an obligation-enforcement mechanism is that the fulfillment of obligations must be an indivisible part of the transaction. In the case of payment, this results in the impossibility to use existing e-purse designs. In our system, the payment mechanism is integrated within the process of sale of information. The payment mechanism implemented guarantees that the obligations are fulfilled if the user accesses the content. It also guarantees that the customer can avoid the obligations (for instance, refuse to pay) in case the content has not been accessed. Additionally, it should provide proofs for the customer in case the contents received do not suit the request [5].

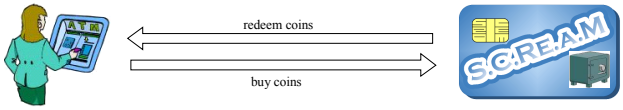

Fig. 2a. Coin buying scenario

CoinID	Secret	Value	Status	To/From	Date
#1	******	4.25	Spent	WebMerchant1.com	01/01/01
#3	******	14.11	Spent	WebMerchant2.com	02/02/01
#5	******	6.2	Spent	WebMerchant3.com	07/10/01
#12	******	1.02	Ready	WebMerchant4.com	10/11/01
#15	******	51.25	Paid	User89	02/01/01
#0	******	474.42	Unused	Bank1	-
#12	******	57.00	Unused	Bank2	

Fig. 2b. Structure of EC-GATE cards e-purse

Figure 2a shows how customers may buy card money in ATM machines from their banks or by Internet. This

money is received as a single "unused" coin worth for the amount requested. Unused coins are encrypted for a specific smart card. Therefore, they can not be used by any other card. Figure 2b shows the structure of the e-purse implemented in the EC-GATE cards. To make a purchase one of the unused coins in the card is split producing three new coins. The first one is a "paid" coin prepared for the merchant (worth the amount of the purchase). The second coin is a copy of the first one that is marked as "ready" and kept inside the customer card. The third one is a new "unused" coin (worth the rest of the value of the original coin). The merchant coin is sent to the merchant in the content request. Once the merchant receives the "paid" coin it produces a MP for that smart card and sends it to the customer. When the customer receives and executes the PCO containing the requested content, the coin state is changed from "ready" to "spent". In case the user finally decides not to access the content of the PCO the coin remains in the "ready" state which allows the user to cancel the payment to the merchant.

"Paid" coins are sent by merchants to the bank as payment orders. Each "paid" coin has a validity interval. Upon reception of the "paid" coin the bank will transfer its value from the customer account to the merchant account. In case the bank also receives the matching "ready" coin the transfer is cancelled. In case the customer has not received the license or decides not to execute the PCO, the "ready" coin can be used to cancel the payment to the merchant.

This payment mechanism guarantees that the merchant gets the payment if the user executes the PCO. It also guarantees that the customer can refuse to pay in case the information has not been accessed. Additionally, it provides proofs for the customer in case the information contained by the PCO does not suit the request. Redemption always takes place before the user of the card (in this case the user may be a merchant) buys new money. Before users can buy new money, their "spent" coins and their received "paid" coins are sent to the bank. Optionally, users can also send the "ready" coins that they do not wish to use.

2.4 System Overview

A general overview of the main components of the system and their relation is depicted in figure 3. The first component is the *SmartProt* protection system. It relies on the use of egate USB smart cards. This component transforms unprotected content objects in the originator server into PCOs as described in section 2.2.

The second component is the *SAC* system, designed to help security administrators in the specification, management and validation of access control policies and rights/obligations expressions. This component uses the SOADs documents as a basis for the specification of SPL policies and PAS [3]. It is also responsible for the automated validation of access control and rights/obligations policies at different levels. SPL policies are validated syntactically and, additionally, a semantic validation is made possible by the use of a specific Semantic Policy Validator, included in the *Policy Assistant*, which parses the SPL specification validating it. Finally, as an extension of the semantic validation, policies can also be validated in the context where they will be applied. Policy context validation uses the context information for the detection of possible inconsistencies in the SPL policies. Therefore, the *Policy Assistant* integrates all the tools to facilitate the administration of the access control system.

The enforcement mechanisms of the SPL policies, along with the rights/obligations expressions, are provided by the first component based on *SmartProt*.

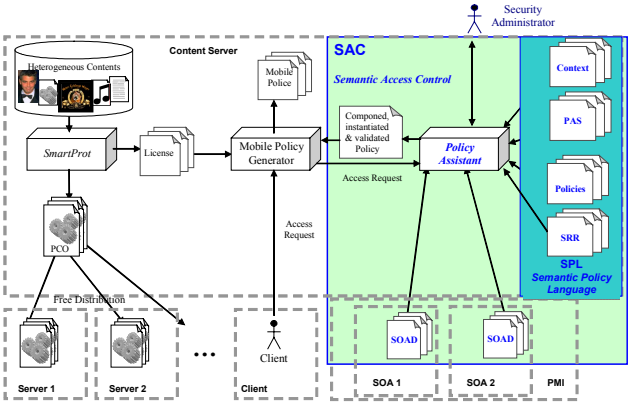

Fig. 3. EC-GATE Infrastructure

The third component, called *Mobile Policy Generator*, attends requests from end users producing MPs dynamically. When a new access request is received, the *Mobile Policy Generator* submits it to the *Policy Assistant*, part of the *SAC* component. It uses different sources of metadata (SRR, Context,…) to determine the set of applicable policies and rights/obligations expressions for a given PCO. After receiving a request the *Mobile Policy Generator* analyses the semantic metadata available for the target PCO, which is contained in SRRs, finds the appropriate PAS and retrieves the necessary SOADs. Using this information, the *Mobile Policy Generator* is able to find the applicable SPL policies with the corresponding rights and obligations expressions. These policies are then analyzed and instantiated using the metadata about the resource (SRRs) and the context. Finally, these policies are combined. The combination of policies helps reducing the complexity of administration while enabling more expressive and flexible policies to be considered in the access decision.

2.5 Security and Implementation Issues

Today, smart card technology offers features that not so many years ago corresponded to personal computers. However, compared to the processing power of today's host computers, each access to the smart card introduces important delays. As our scheme requires the transmission

of a considerable amount of code and data to and from the card, it is important to take into consideration the efficiency of the protection scheme. The amount of data and code transmitted determines the magnitude of the delay introduced. On the other side, since the main attack line to the protection scheme is based on the analysis of the functions performed by the card, the protection scheme will be more secure as the functions grow in number, size and complexity. Consequently, it is necessary to find a balance between security and speed. Fortunately, in this case, this equilibrium is possible and it is not difficult to obtain security and speed measures that satisfy both the software producer and the client.

It has been proved that the main bottleneck in the performance of smart card applications is the communication between the card and the host [8], therefore the introduction of the new USB smart cards with a bandwidth of up to 448Kbits/sec. represents a significant advance towards the solution of this problem.
The structure of the contents of the card is depicted in figure 4.

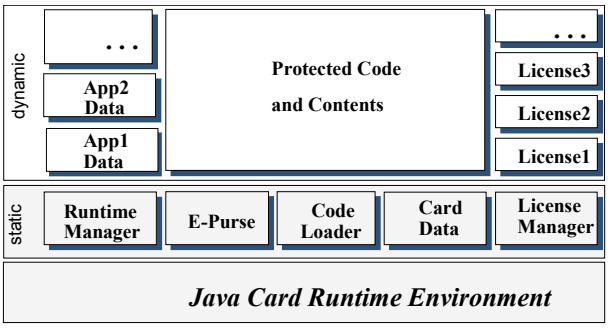

Fig. 4. Internal Structure of EC-GATE cards

In our current implementation, the components of the *SmartProt* system (code loader, license manager and runtime manager) and the payment system (e-Purse) are implemented by a JavaCard cardlet. However, applications at this level have limitations (for example, they are isolated by firewalls) and performance constraints. Therefore, our aim is to propose the implementation of those functions at a lower level in order to obtain better performance and to enable the deployment of other applications that can take advantage of the software protection infrastructure.

The code of the PCO contains encrypted sections that must be executed within the smart card. When one of these sections is downloaded into the card, the *SmartProt* cardlet must locate the corresponding license and execute this section. The code of protected sections must be translated to be executed in the smart cards. The basic goal was to achieve a compact, powerful and flexible card code. As the main performance bottleneck is actually the communication with the card, we have defined a compact format that is later translated into an internal format. This internal format was defined in order to overcome the problems associated to the lack of file management functionality in JavaCard. Therefore we have defined the Instruction class in the JavaCard language in order for the instructions to be self-contained and to achieve easy referencing between instructions. In this way we do not need to put a standard interpreter in the card (the interpreter is the Instruction class itself), which results in greater flexibility.

The code is loaded in the card and converted into an Instruction objects array, afterwards the execution of the code as simple as calling the Execute method of the first object of the array. The execution of each instruction ends with the execution of the rest of the code. This process continues until there are no more instructions in the branch. At this moment, the execution of each calling (previous) instruction terminates. If the branch was started in any Loop instruction then the condition is evaluated and the execution continues appropriately.

3. Conclusion

The EC-GATE system combines a software protection mechanism (*SmartProt*), a new access control model (*SAC*) for DRM scenarios and an external PMI in order to provide distributed access control, rights, obligations management and enforcement and secure content distribution in information commerce scenarios.

More specifically, EC-GATE represents a solution applicable in different distributed scenarios, is flexible, solves the persistent protection problem, can be applied regardless of the attribute certification scheme, implements distributed access control and rights management and enforcement mechanisms, integrates obligations (payment) and access to the contents, incorporates secure content distribution and allows the dynamic modification of access control, rights and obligations policies transparently and efficiently. The system implemented demonstrates the feasibility of the proposed approach, using e-gate CyberflexTM USB Java smart cards as secure coprocessors. The high capacity and the transfer speed of these cards makes possible that the performance of the PCO is good. As mentioned, the implementation of the *SmartProt* functionality at the card operating system level would allow the scheme to be more efficient.
This DRM system has been applied to a Digital Library [3] and electronic newspaper [9] scenarios, showing the suitability of this approach to the pay per use and pay per view business models.

References

[1] Yagüe, M.I., Maña, A., López, J., Troya, J.M. Applying the Semantic Web Layers to Access Control. *Proc. Workshop on Web Semantics, DEXA 2003 Conference*. IEEE Computer Society Press, 2003.
[2] Maña, A., Pimentel, E. An Efficient Software Protection Scheme. *Proc. IFIP SEC'01*. Kluwer. 2001.

[3] Yagüe, M.I., Maña, A., López, J., Pimentel, E., Troya, J.M. A Secure Solution for Commercial Digital Libraries. Online *Information Review journal, 27*(3), 2003, 147-159.
[4] Kudo, M., Hada, S. XML Document Security based on Provisional Authorization. *Proc. 7th ACM Conference on Computer and Communications Security*,2000.
[5] López, J., Maña, A., Pimentel, E., Troya, J.M., Yagüe, M. I. An Infrastructure for Secure Content Distribution. *Proc. of ICICS'02*. Springer-Verlag, 2002.
[6] Fayad, A., Jajodia, S. Going Beyond MAC and DAC Using Mobile Policies. *Proc. IFIP SEC'01*. Kluwer, 2001.
[7] Maña, A., López, J., Ortega, J., Pimentel, E., Troya, J. M. Smart Card-based Practical Software Protection. To appear in *International Journal of Information Security*.
[8] Markantonakis, C. Is the Performance of Smart Card Cryptographic Functions the Real Bottleneck? *Proceedings of IFIP SEC'01*. Kluwer, 2001.
[9] A. Maña, M.I.Yagüe, V. Benjumea. EC-GATE: Electronic Commerce based on e-gate technology. *Golden Award of the E-gate Open Contest 2002*, París, 2002. http://www.axalto.com/press/news.asp?id=41