Privacy Protection Measures and Technologies in Business Organizations:

Aspects and Standards

George O.M. Yee
Aptus Research Solutions Inc., Canada & Carleton University, Canada

Information Science
REFERENCE

Managing Director:	Lindsay Johnston
Senior Editorial Director:	Heather Probst
Book Production Manager:	Sean Woznicki
Development Manager:	Joel Gamon
Development Editor:	Myla Harty
Acquisitions Editor:	Erika Gallagher
Typesetters:	Milan Vracarich, Jr.
Print Coordinator:	Jamie Snavely
Cover Design:	Nick Newcomer, Greg Snader

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

Library of Congress Cataloging-in-Publication Data

Privacy protection measures and technologies in business organizations: aspects and standards / George O.M. Yee and Aptus Research Solutions Inc., editors.
p. cm.
Includes bibliographical references and index.
Summary: “This book is a collection of research on privacy protection technologies and their application in business organizations”--Provided by publisher.
ISBN 978-1-61350-501-4 (hbk.) -- ISBN 978-1-61350-502-1 (ebook) -- ISBN 978-1-61350-503-8 (print & perpetual access) 1. Business--Data processing--Security measures. 2. Computer security. 3. Data protection. 4. Privacy, Right of. I. Yee, George. II. Aptus Research Solutions.
HF5548.37.P755 2012
658.4’78--dc23
2011038433

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

All work contributed to this book is new, previously-unpublished material. The views expressed in this book are those of the authors, but not necessarily of the publisher.

Chapter 4
Self-Protecting Access Control:
On Mitigating Privacy Violations with Fault Tolerance

Anne V. D. M. Kayem
University of Cape Town, South Africa

Patrick Martin
Queen's University, Canada

Selim G. Akl
Queen's University, Canada

ABSTRACT

Self-protecting access control mechanisms can be described as an approach to enforcing security in a manner that automatically protects against violations of access control rules. In this chapter, we present a comparative analysis of standard Cryptographic Access Control (CAC) schemes in relation to privacy enforcement on the Web. We postulate that to mitigate privacy violations, self-protecting CAC mechanisms need to be supported by fault-tolerance. As an example of how one might to do this, we present two solutions that are inspired by the autonomic computing paradigm[1]*. Our solutions are centered on how CAC schemes can be extended to protect against privacy violations that might arise from key updates and collusion attacks.*

INTRODUCTION

The ability to execute multiple transactions across a myriad of applications has made the Internet a prime platform for building Web applications. Applications like Facebook (Facebook, 2010) and MySpace (MySpace, 2010), attest to this popularity and have been rated as being the most popular social networking applications in the English speaking world. Increasingly, business organizations are taking advantage of these social networking applications and other web applications to collect personal information about consumers and likewise consumers have shown a keenness for the web as a medium of communication because of the interactivity and fast

DOI: 10.4018/978-1-61350-501-4.ch004

response time it offers. Yet, the same qualities of flexibility and interactivity that the web is famous for have become an impediment in the face of the growing incidences of data privacy violations. For example, in October 2010 a Wall Street Journal Investigation revealed that many popular Facebook applications were transmitting consumer personal information to advertising and Internet tracking companies (Slattery, 2010), (Foremski, 2010). Cases like this have fueled growing concerns, on the part of consumers, that their data can be leaked without their consent to third parties. In this section, we discuss the context in which data privacy violations occur and why this happens in spite of the fact that access control mechanisms can be implemented to protect the information.

Context and Motivation

The Internet is built on the assumption that the users of the network can be trusted to behave honestly and so do not use the system or behave in ways that could compromise the performance and/or credibility of the system. Yet, this quality of open access makes web-applications inherently vulnerable to violations of information privacy rules (accidental or intentional) that can compromise the levels of data protection that these applications promise users (Harrison, February 2007), (Sandhu, 2005), (Tanenbaum & Steen, 2007). In many cases, privacy violations occur because consumers assume that they have “correctly” applied some access control mechanism that will prevent illegal access to their information. For instance, in social networking applications, it often is the case that a user will post “confidential” information and forget to set the parameter to prevent transitive disclosures to friends of the user’s friends.

As well, consumers have a tendency to naively assume that business organizations will do what they promise, while business organizations are sometimes unaware of the far-reaching consequences of certain management decisions. In the case of Facebook, one could imagine that a third-party made contact by indicating that they would like to test the popularity of a new application. Facebook probably agreed because usage of the application might attract new members. However, the acceptance agreement might not have indicated clearly that the application could not collect information about the users who choose to use the application and/or people whom the users know might be interested in using the application (Fung, Wang, Chen, & Yu, 2010). Data privacy leaks like the one we describe are a growing concern for organizations because they result in a loss of revue (Foremski, 2010).

Until recently, organizations simply focused on defining a security domain and security policies were used to control access to information. The assumption was that if correctly specified, failure (either deliberate or not), on the part of users, to adhere to data privacy policies would be unlikely. However, the emergence of concepts like service-oriented architectures and cloud computing have dissolved inter-organization boundaries. Consequently, web applications and/or services can interact flexibly across multiple security domains and in ways that are not easy to predict at runtime. Therefore, security policies and access control schemes need to be modeled or extended to cope with situations in which changes in security requirements result in privacy violations.

In this chapter, we discuss the growing need to extend access control models to enforce privacy in scenarios involving changing security requirements like the Web. More specifically, we consider the literature on the more popular access control models like mandatory, discretionary, and role-based access control, and discuss some of the ways in which these models have been extended to enforce data privacy requirements. In recent years, cryptographic access control (CAC) is has received increased attention as a method of enforcing data privacy on the Web. CAC schemes have the advantage of providing protection for data in untrustworthy environments like the Web.

However, CAC schemes have been criticized for being impractical in terms of performance and so have not gained wide spread popularity. We postulate that data privacy is intertwined with dependability and so, self-protecting CAC schemes can efficiently protect against data privacy violations if the CAC schemes are supported with fault tolerance solutions.

In order to extend CAC schemes to incorporate fault tolerance, we use the autonomic computing paradigm. The autonomic computing paradigm was proposed in 2001 by IBM (Corbi, 2003), (Chess, 2005) and suggests that computing systems can be modeled to be self-managing and self-configuring. Self-management implies a reduced need manual management which is time consuming and self-configuration, the ability to adapt to new scenarios (Hart, Davoudani, & McEwan, 2007), (Huebscher & McCann, 2008).

Organization

The rest of the chapter is structured as follows. In Section 2, we discuss privacy and access control in the business context and consider how access control models have been extended in recent years to enforce data privacy on the Web. Section 3 presents cryptographic access control (CAC) models in order to highlight the advantages of using these CAC schemes over standard access control schemes in privacy enforcement. Changing and conflicting security requirements on the Web make adaptability in an access control scheme a desirable quality in data privacy enforcement. In Section 4 we present two examples to show how the autonomic computing paradigm can be used as an inspiration for designing self-protecting CAC schemes. Future research directions and challenges are discussed in Section 5 and we offer concluding remarks in Section 6.

PRIVACY VIA ACCESS CONTROL IN THE WEB CONTEXT

The popularity of service oriented architectures (SOAs) and more recently, cloud computing, indicate that Mark Weiser's vision of ubiquitous computing has become a reality in the business context (Weiser, 1999). More and more, business organizations are using the Web to collect personal information from consumers in order to find adequate responses to queries and/or provide the services that a consumer requests. Inter-service interactions often result in compositions that are not easy to predict and so there is growing concern about the potential violation of consumer data privacy (Byun & Li, 2008), (Ren and Lou, 2007).

Inter-domain information exchanges are handled, in general, within a trusted security framework. In SOAs and cloud computing environments it is difficult to predict how security policies belonging to different domains will be combined to enforce access control. Additionally, verifying that the combination of security policies actually enforces the minimum access control requirements of the services and/or applications involved in accessing portions of data on the system is a challenging problem for manual security mechanisms.

In general, security models for access control on the Internet can be classified into one of three categories: discretionary, mandatory, or role-based (Tanenbaum & Steen, 2007), (Osborn, 2002), (Rjaibi, 2004). The discretionary access control approach allows a user to decide to whom they choose to authorize access and is a good access control approach for data sharing applications that users can join or leave spontaneously. In mandatory access control, access to data is regulated by a lattice that is used to monitor the flow of information among communicating users by assigning labels to files to restrict accessibility to authorized users. This is a good approach for government and military organizations with stricter requirements of access to data. Finally, role-based access control is popular in business organizations mainly because of its flexibility. Role-based access control combines the concepts of discretionary and mandatory access control, by allowing organizations to assign roles to users according to the permissions of access that a security administrator wishes to grant the user. A user can have one or more roles and these roles can be temporary or for the long term.

We use the example of a hypothetical e-business application that relies on a data sharing environment to serve as a backbone for business operations. In this example, users can join or leave the social networking environment spontaneously and use the e-business application to purchase goods. Users are categorized into groups according to interest and a user's group determines the privileges of access (read, write, modify, and/or delete) that he/she is allowed. The advantage of using a social networking environment as a backbone is that it serves platform on which the e-business application can attract a clientele without the cost of advertising on regular channels like radio and television. Examples of real world data sharing applications include Chat Systems, Shared White boards, and "social networking" environments like Facebook, (Facebook, 2010), (Nazir, 2008), and MySpace (MySpace, 2010), (Besmer, 2009). In the following, we review the three models of access control, highlighting their pros and cons in relation to enforcing self-protection against data privacy violations.

Discretionary Access Control

The concept of discretionary access control (DAC) is probably as old as the concept of network-based or distributed computing. It basically rests on the principle that each user of a system should be able to decide on the privileges that are assigned to users wishing to view files that he/she created. The DAC principle is used in many data sharing web applications like Facebook, MySpace, and Flickr because it is simple and straight-forward to implement in a distributed environment like the Internet. Using a DAC mechanism gives users control over the access rights to their files without the need to comply with a set of pre-specified rules. When these rights are managed correctly, only those users specified by the file owner may have some combination of access permissions on the file (Pfleeger & Pfleeger, 2003), (Tanenbaum & Steen, 2007) and consequently privacy violations are not possible unless the access rules are violated.

Consider the case in which a DAC model is used to enforce access control in a social networking environment that serves as a backbone for an e-business application. As shown in Figure 1, Jane can choose to create a folder containing files with photographs that she wishes to sell and make access available only to members who fulfill a certain number of criteria. For instance, Jane could decide that only the members of her photography club can have access to the files. So, as Figure 1 shows, John who belongs in Jane's class can access the photographs that Jane makes available whereas Sam, who does not belong in her class, has no access rights.

An access control matrix is a simple but effective model for expressing and enforcing simple security policies in situations like the one depicted in Figure 1 (Gollmann, 2006). With an access control matrix, access rights can be defined individually for each combination of users and files/folders. The rules of access depicted in Figure 1, can be expressed using an access control matrix like the one given in Figure 2.

Although the DAC approach is a good way of providing a standard framework for enforcing access control in a social networking environment it has certain disadvantages that limit its ability to enforce privacy. An example of such a problem is the confinement problem that Lampson (Lampson, 1973) cited, which is to determine whether there is a mechanism by which a user authorized to access a file may leak information contained in that file to users that are not authorized to ac-

Figure 1. Discretionary access control

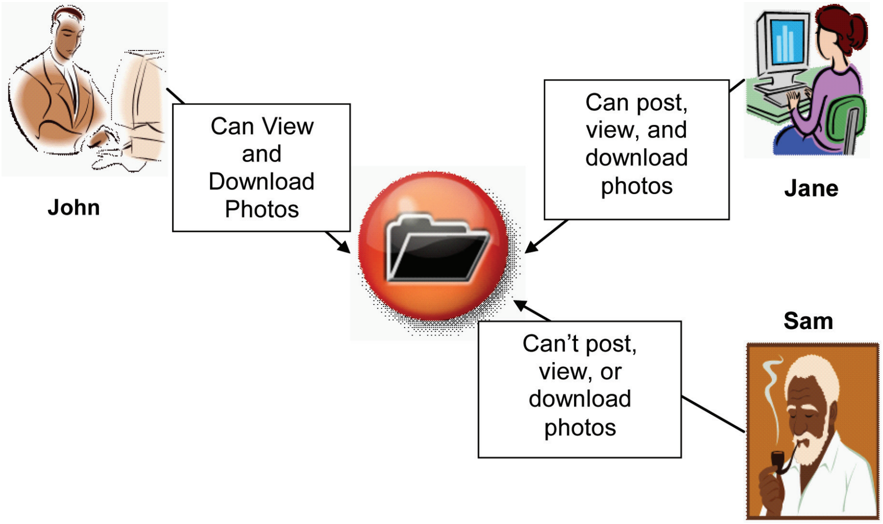

John
Can View and Download Photos
Can post, view, and download photos
Jane
Sam
Can't post, view, or download photos

Figure 2. An example of an access control matrix

	Photographs Folder
Sam	-
Jane	{View, Download, Upload}
John	{View, Download}

cess that file. Harrison et al. (Harrison, Munro, & Spiller, 2007) formally showed that the confinement problem is undecidable due to the characteristic of discretionary transfer of access rights between users in the DAC model. An added consideration is that although the DAC model is effective for specifying security requirements and is also easier to implement in practice, its inability to control information flow implies it is not well-suited to the context of Web-based collaborative applications where central control in some form is desirable (Jeong & Kim, 2004). Moreover, since users applying a DAC model do not have a global picture of the data on the system, it is difficult to take the semantics of the data into consideration in assigning access rights so, information might unknowingly be revealed to unauthorized users leading to privacy violations.

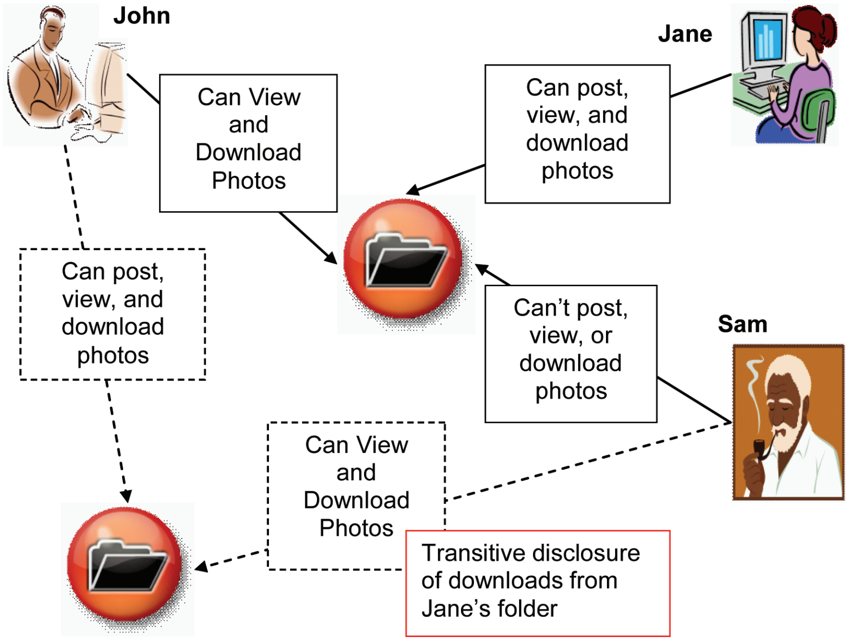
Mandatory Access Control

Apart from the confinement and information semantics problems inherent in the DAC model, a key problem that the DAC model faces is vulnerability to Trojan Horse attacks (Bell & Lapadula, 1973), (Biba, 1977), (Sandhu, R., 1993). In order to violate confidentiality and privacy, Trojan Horse attacks exploit two possibilities of access rights management:

- Changes in access rights to a file are handled by the file owner and are not centrally controlled so a malicious user can masquerade as the file owner and grant read-access to a file against the owner's desire.
- Users authorized to access a file are typically allowed to create copies of the file, so a malicious user can create a copy of the file or part of it and grant read-access to users to whom the owner has not authorized access.

For instance, as shown in Figure 3, John can download information from Jane's photographs folder and then proceed to make these pictures

Figure 3. A case of transitive disclosures in the DAC model


John
Jane
Can View and Download Photos
Can post, view, and download photos
Can post, view, and download photos
Can't post, view, or download photos
Sam
Can View and Download Photos
Transitive disclosure of downloads from Jane's folder

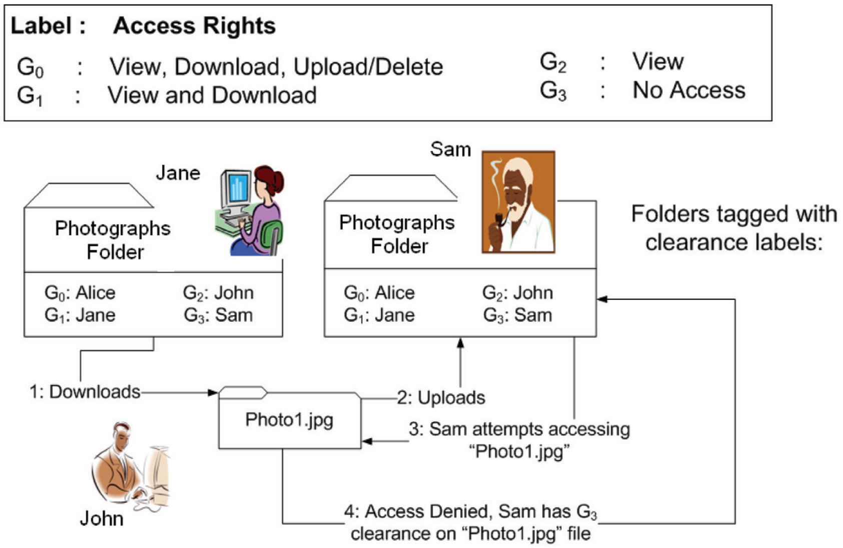
available to Sam. Sam can then proceed to sell the photographs to an unscrupulous e-business owner or modify the photographs and post false/damaging information about Jane.

The mandatory access control (MAC) model counters these threats by controlling access centrally. An ordinary user (i.e., not the central authority) cannot change the access rights a user has with respect to a file, and once a user logs on to the system the rights he/she has are always assigned to all the files he/she creates. This procedure allows the system to use the concept of information flow control to provide additional security (Gollman, 2005).

Information flow control allows the access control system to monitor the ways and types of information that are propagated from one user to another which is an advantage for privacy enforcement. A security system that implements information flow control typically classifies users into security classes and all the valid channels along which information can flow between the classes are regulated by a central authority or security administrator (Denning, 1976). Therefore, privacy violations are more difficult to perpetuate than in the DAC model because information can only be shared in ways that are authorized by the security administrator.

In the MAC model, each user is categorized into a security class and the files are tagged with security labels that are used to restrict access to authorized users (Rjaibi, 2004). The example shown in Figure 3 can be extended to handle a security scenario in which a security administrator prevents transitive disclosures, by the users accessing Jane's Photographs Folder, by using data labels to monitor information flow. Each data object is tagged with the security clearance labels of each of the users in the system. As shown in Figure 3, by extending the discretionary access example we gave in Figures 1 and 2, a transitive disclosure could occur if a user, in this case Sam, gains access to Jane's photographs folder because he belongs in John's list of "friends". The MAC model prevents such disclosures by defining a hierarchy, such as the one depicted in Figure 4,

Figure 4. MAC model – Information flow control to prevent privacy violations



to monitor information flow centrally. The users are assigned labels according to their security clearance and information flow is regulated by authenticating a user and then granting access to the file based on their privileges. Since each file is labeled with a security clearance tag, so Sam can no longer access files that John downloads from Jane's Photographs Folder because Sam does not have a security clearance that allows him access. When the access control policy of a system is based on the MAC model, the security of the system ceases to rely on voluntary user compliance but rather is centrally controlled, making it easier to monitor usage patterns and prevent privacy violations.

Multilevel Access Control

The multilevel security (MLS) model is essentially a special case of how the MAC model is implemented for different contexts or scenarios. In the MLS model, a security goal is set and information flow is regulated in a way that enforces the objectives determined by the security policy (Rjaibi, 2004). Practical implementations of access control schemes based on the MLS concept include the Bell-Lapadula (BLP), Biba Integrity Model, Chinese wall, and Clark-Wilson models (Rjaibi, 2004), (Bell & Lapadula, 1973), (Clark & Wilson, 1987), (Brewer & Nash, 1988), (Huang & Shen, 2004),(Liu & Chen, 2004). In the following, we briefly discuss each of these four MLS models but for a detailed exposition of the field one should see the works of McLean (McLean, 1990), Sandhu (Sandhu, R., 1993), Nie et al. (Mie, Feng, Che, & Wang, 2006), and Gollmann (Gollman, 2005).

- **The BLP and BIBA models:** In the BLP model (Bell & Lapadula, 1973), high level users are prevented from transmitting sensitive information to users at lower levels, by imposing conditions that allow users at higher levels to only read data at lower levels but not write to it. On the other hand, users at lower levels can modify information at higher levels but cannot read it. This method of information flow control circumvents privacy violations but allows users at lower levels to write information to files at higher levels that they cannot read. This can result in a situation where violations of data integrity are difficult to trace (Liu & Chen, 2004). So, a malicious use can modify data to provoke violations of the data privacy policy that the access control scheme is meant to enforce. Violations to data integrity are a serious problem for privacy schemes because the corrupted data not only misinforms the user but can lead to inference of unauthorized information. The Biba integrity model (Biba, 1977) addresses the problem of data integrity by checking the correctness of all write operations on a file. However, this approach opens up the possibility of privacy violations by inference of high level information from low level information.

- **The Chinese wall model:** In 1989, Brewer and Nash proposed a commercial access control model called the Chinese wall model (Brewer & Nash, 1988). The basic idea is to build a family of impenetrable walls, called Chinese walls, amongst the datasets of competing companies. So, for instance, the Chinese wall model could be used to specify access rules in consultancy businesses where analysts need to ensure that no conflicts of interest arise when they are dealing with different clients. Conflicts can arise when clients are in direct competition in the same market or because of ownerships of companies. Therefore, analysts need to adhere to an access control policy that enforces a strict privacy policy. Such a privacy policy needs to prohibit information flows that cause a conflict of interest. The access rights in this model are designed along the lines of the BLP model but with the difference that access rights are re-assigned and re-evaluated at every state transition whereas they remain static in the BLP model. Unfortunately, their mathematical model was faulty and the improvements proposed have failed to completely capture the intuitive characteristics of the Chinese wall security policy (Lin, 2000), (Lin T., 2006).
- **The Clark-Wilson (CLW) Model:** Like the BIBA model, the CLW model addresses the access control requirements of commercial applications in where data integrity is more important than data privacy and confidentiality (Clark & Wilson, 1987). The CLW model uses programs as an intermediate control level between users and data (files). Users are authorized to execute certain programs that can in turn access pre-specified files. Security policies that are modeled using the CLW model are based on five rules:
  1. All data items must be in a valid state at the time when a verification procedure is run on it.
  2. All data transformation procedures need to be set a priori and certified to be valid.
  3. All access rules must satisfy the separation of duty requirements.
  4. All transformation procedures must be stored in an append-only log.
  5. Any file that has no access control constraints must be transformed into one with one or more access control constraints before a transformation procedure is applied to it.

The CLW model is more of a security policy specification framework that extends the concepts in the BIBA model to the general case. Therefore, like the Biba model, the CLW model is vulnerable to privacy violations that are due to inference of high level information from low level information.

Role Based Access Control

Role-based access control (RBAC) is a combination of mandatory and discretionary access control.

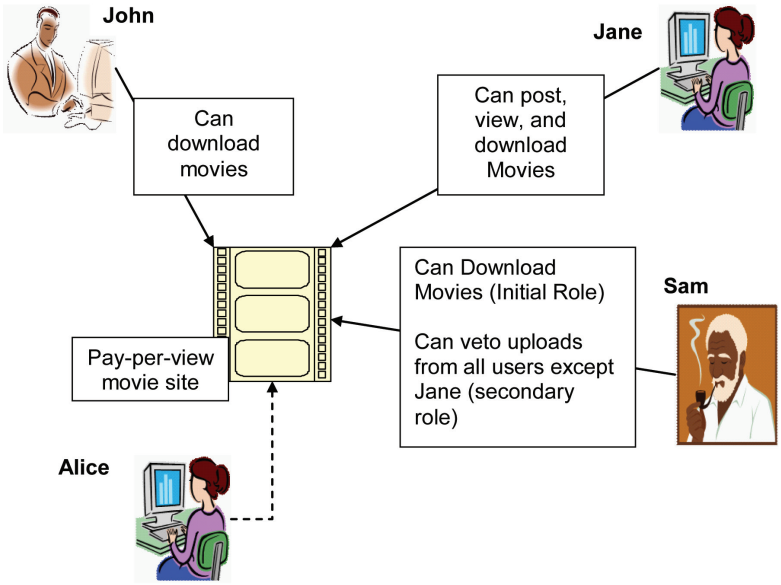
In the role-based access control model, a role is typically a job function or authorization level that gives a user certain privileges with respect to a file and these privileges can be formulated at a high level (e.g. in simple English) or at a low level (e.g. formally specified and hard coded into an application). RBAC models are more flexible than their discretionary and mandatory counterparts because a user can be assigned several roles and a role can be associated with several users. Unlike the traditional DAC approach to access control; RBAC assigns permissions to specific operations with a specific meaning within an organization, rather than to low level files. For example, a DAC mechanism could be used to grant or deny a user modification access to a particular file, but it does not specify the ways in which the file could be modified. By contrast, with the RBAC approach, access privileges are handled by assigning permissions in a way that is meaningful, because every operation has a specific pre-defined meaning within the application.

Dynamic role based access control (DRBAC) aims to extend the standard RBAC model to cope with situations that require adapting to changing security requirements. For instance, in context-aware environments, a security policy might need to adapt its settings to cope with a change in context (Zhang & Parashar, 2004). Moreover, the DRBAC approach to access control is more flexible than the one in the DAC and MAC models in the sense that roles can have overlapping responsibilities and privileges, so users belonging to different roles may need to perform common operations.

RBAC assumes that all permission needed to perform a job can be neatly encapsulated so that the role in which a user gained membership is not mutually exclusive of others roles that the user already has. The operations and roles can be subject to organizational policies or constraints and, when operations overlap, hierarchies of roles are established. Instead of instituting costly auditing to monitor access, organizations can put constraints on access through DRBAC. For example, it may seem sufficient to allow all the users on the system (Jane, John and Sam) to have 'view' and 'download' access to the Photographs Folder, if their accesses are monitored carefully to prevent violations of privacy. By using DRBAC, constraints can be placed on user access and context so that they do not tamper with contents of the Photographs Folder. However, role engineering is a challenging problem because guaranteeing data privacy requires a model that ensures data security and makes security administration less cumbersome than it currently is (Gollman, 2005). On the one hand, for stronger security, it is better for roles to be more granular, thus having multiple roles per user. On the other hand, for easier administration, it is better to have fewer roles to manage. Organizations need to comply with privacy and other regulatory mandates and to improve enforcement of security policies while lowering overall risk and administrative costs. Meanwhile, web-based and other types of new applications are proliferating, and the Web services application model promises to add to the complexity by weaving separate components together over the Internet to deliver application services.

An added drawback RBAC faces in privacy enforcement is that roles can be assigned such that conflicts are created which can open up loopholes in the access control policy. For example in the scenario in Figure 5, we can assume that Jane is the security administrator for the pay-per-view Movies Folder, and that she chooses to assign roles to users in a way that allows the users to either download or upload movies but not both. Now suppose that at a future date Jane decides to assign a third role that grants a user, say Sam, the right to veto an existing user's (e.g. Alice's) uploads. In order to veto Alice's uploads, Sam needs to be able to download as well as temporarily delete questionable uploads, verify the movies and, if satisfied, reload the movies to the site. So, essentially Sam has the right to both download and upload movies to Movies Folder, a role assignation that conflicts with the initial security

Figure 5. A case of conflicting access assignments


John
Can download movies
Jane
Can post, view, and download Movies
Can Download Movies (Initial Role)
Can veto uploads from all users except Jane (secondary role)
Sam
Pay-per-view movie site
Alice

policy specification that Jane made. Security policy combinations or extensions like the one have just described need to be handled with care to prevent violations of privacy. Therefore extensions RBAC model to enforce privacy and incorporate adaptability to cope with scenarios of changing security requirements, need to be evaluated and/or implemented with care.

Extensible Access Control Markup Language: A Privacy Discussion

In the previous sections, we presented and discussed some of the standard access control models highlighting the challenges that they face in handling scenarios of privacy violations on the Web. Although RBAC schemes offer a number of advantages over the DAC and MAC models in terms of security management, they too are not designed to prevent violations of privacy. The objective of this section therefore, is to explore extensions to the RBAC model as well as other access control paradigms for privacy enforcement on the Web.

The principal paradigm in distributed systems before the emergence of the World Wide Web had been the client-server architecture (Tanenbaum & Steen, 2007), (Gollman, 2005). The latter, in its simplest form, allows the server to protect information by authenticating a client requesting access. Kerberos is an example of an authentication service designed for such an environment (Tanenbaum & Steen, 2007). This client-server architecture has however changed in many aspects. For instance, when a client looks at a web page, the client's browser will run programs embedded in the page. So, instead of handling simple accesses either to an operating system or a database, programs are being sent from the server to be executed at the client side. Clients receive programs from servers and can store the session states in "cookies". The Internet has also created a new avenue for software distribution via downloads that can sometimes result in privacy violations and so organizations have learnt, sometimes through the hard way, to restrict the kinds of programs that they allow their employees to download. As well, while the Internet has not created fundamentally new problems data privacy violations, it has changed the context in which privacy needs to be enforced. Consequently, the design of access control paradigms is currently going through a transitory phase in which standard paradigms are being re-thought and evolved to cope with the scenarios that arise on the Internet.

Inherent in the current paradigm shift in designing access control schemes, is the desire to allow users more control over how their personal information is managed (Ardanga et al., 2010). The main focus in terms of extending access control schemes for privacy enforcement has been on the Extensible Access Control Markup Language (XACML) and IBMs XML Access Control Language (XACL). XACML is based on the eXtensible Markup Language (XML) (Chadramouli, 2003). The Extensible Access Control Markup Language (XACML) is a general-purpose language for specifying access control policies (Hu, Martin, Hwang, & Xie, 2007). In XML terms, it defines a core schema with a namespace that can be used to express access control and authorization policies for XML objects. Since it is based on XML, it is, as its name suggests, easily extensible. XACML supports a broad range of security policies (Chadramouli, 2003), (Hu, Martin, Hwang, & Xie, 2007), and uses a standardized syntax for formatting requests so that any one of the following responses to an access request will be valid:

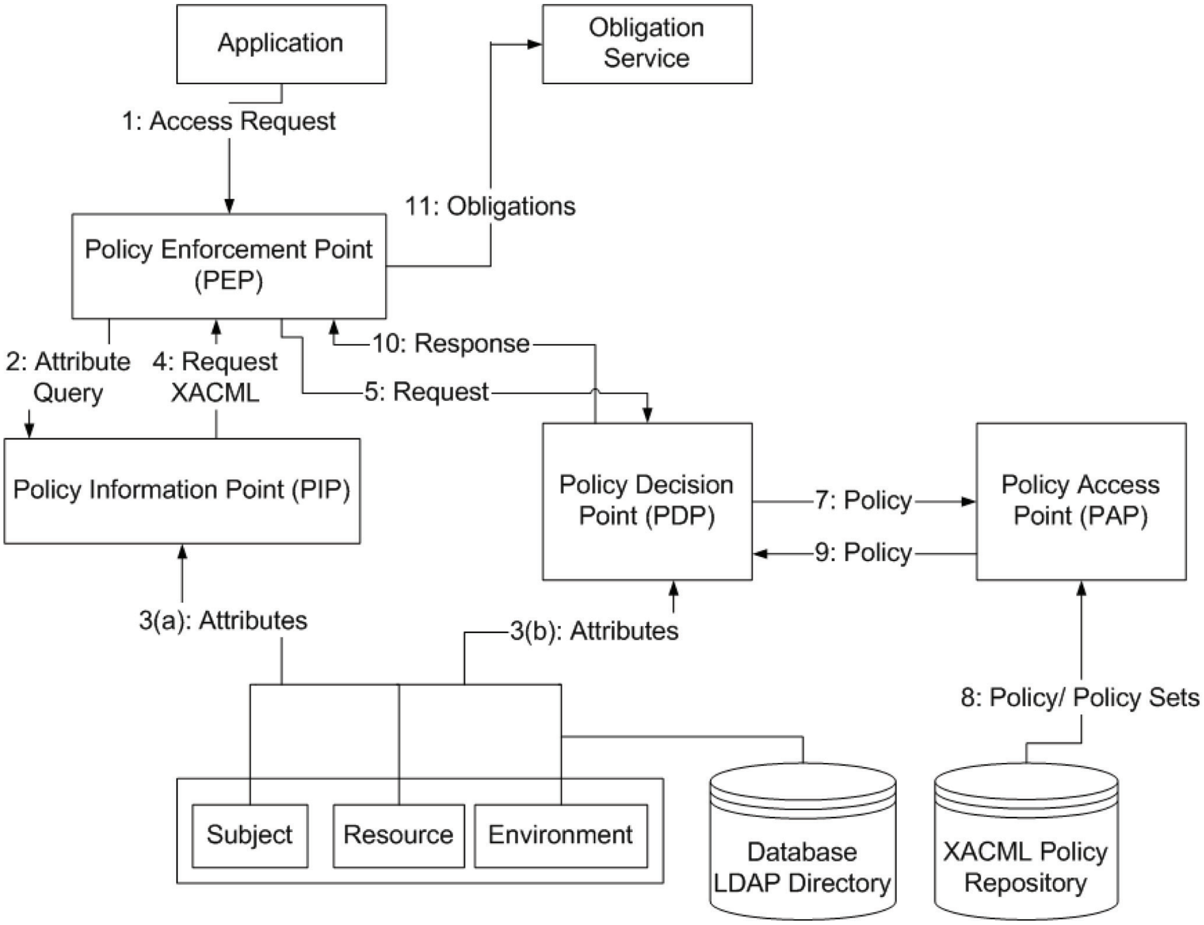
- Permit: action allowed
- Deny: action disallowed
- Indeterminate: error or incorrect/missing value prevents a decision
- Not Applicable: request cannot be processed

As shown in Figure 6, XACML's standardized architecture for this decision-making uses two primary components: the Policy Enforcement Point (PEP) and the Policy Decision Point (PDP). The PEP constructs the request based on the user's attributes or credentials, the resource requested, the action specified, and other situation-dependent information provided through a Policy Information Point (PIP). The PDP receives the constructed request, compares it with the applicable policy and system state through the Policy Access Point (PAP), and then returns one of the four replies specified above to the PEP. The PEP then allows or denies access to the resource. The PEP and PDP components may be embedded within a single application or may be distributed across a network. This is an advantage for incorporating extensions for privacy enforcement because privacy enforcing security policies can be verified at the PEP. While user credentials can be checked at the PDP. The combination of both mechanisms makes privacy enforcement easier than in standard DAC, MAC, or RBAC mechanisms and also allows for the establishment of a trust infrastructure where users can manage their identities and personal information.

In order to make the PEP and PDP work, XACML provides a policy set, which is a container that holds either a policy or other policy sets, plus links to other policies. Each individual policy is stated using a set of rules. Conflicts are resolved through policy-combining algorithms which are good for handling cases of potential privacy violations that could result from combining security policies belonging to different domains. XACML also includes methods of combining these policies and policy sets, allowing some to override others. This is necessary because the policies may overlap or conflict. For example, a simple policy-combining algorithm is "Deny Overwrites", which causes the final decision to be "Deny" if any policy results in an "Overwrite". Conversely, other rules could be established to allow an action if any of a set of policies results in "Allow".

Determining what policy or policy set to apply is accomplished using the "Target" component. A target is a set of rules or conditions applied to each subject, object, and operation. When a rule's conditions are met for a user (subject), object, operation combination, its associated policy or policy set is applied using the process described

Figure 6. XACML access control model (Verma, 2004)



above. The associated access control data for a given enterprise domain can then be encoded in an XML document, and the conformance of data to the enterprise access control model can be obtained by validating the XML document against the XML schema that represents the enterprise access control model using XML parsers. These XML parsers are based on standard application programming interfaces such as the Document Object Model (DOM) (Wikipedia DOM, 2010), and the parser libraries are implemented in various procedural languages to enable an application program to create, maintain, and retrieve XML-encoded data.

XML-based, and other, access control languages provide capabilities for composing policies from scratch, by allowing users to specify access control policies, together with the authorizations through the programming of the language. They however lack a formal specification language for access control constraints (like historical-based and domain constraints) that prevent assigning overlapping privileges. As an example, consider the case of constraints that require the manipulation and recording of access states (such as granted privileges). This is to avoid creating situations that result in users who were previously denied access to certain files being unknowingly granted access in a future state. Like most access control languages, XACML does not provide tools for the expression of historical constraints for historical-based access control policies, thus leaving the completeness of the constraint logics to the policy writer. This case is similar to the one that was evoked in Section 2.2 where Sam unintentionally gets a combination of "view" and "download" rights with respect to photographs belonging to Jane that John downloads to his *Photographs Folder*.

Domain constraints are based on the semantic information pertaining to an enterprise context. Therefore a grammar-based language cannot deal with content-based constraints. Consequently, an XML schema is insufficient for a complete specification of the RBAC model for an enterprise since the latter contains content-based domain constraints. An example is not allowing more than one user to be assigned to the role of security administrator (role cardinality constraint) and not allowing the roles "viewer" and "uploader" to be assigned to the same user (separation-of-duty constraint).

XML schema provides a very extensible means for specifying document structures through a comprehensive type definition language. So, advocates for XML access control hold that XML is a good candidate for a linguistic framework that is needed to express an access control model that embodies multiple policy requirements.

Considerable effort has gone into extending XML-based security frameworks to express privacy policies based on the credentials users provide (Ardanga et al., 2010). For instance, recent extensions to XACML incorporate RBAC support for privacy enforcement (Ardagna, De Capitani di Vimercati, Paraboschi, Pedrini, & Samarati, 2009). Other extensions include credential-based access control extensions to XACML to ensure that there is a framework to correctly authenticate user access to data, but also enforce privacy (Ardanga et al., 2010).

Examples of proposals on extending XACML for privacy include the XACML-based privacy centered access control system that provides credential based management and privacy support as well as credential-based access control extensions to XACML (Ardanga, De Capitani di Vimercati, Paraboschi, Pedrini, & Samarati, 2009) (Ardanga et al., 2010). The Ardanga et al. proposal combines XACML with PRIME (Privacy and Identity Management for Europe) to produce an infrastructure that handles access control in a way the enforces privacy policies flexibly. The PRIME system handles five aspects, namely, resource representation, subject identity, secondary use, context representation, and ontology integration. All of these aspects are used to specify the access control requirements and conditions that are used to release data to a user based on their role, or context. The ontology integration aspect allows the system to apply access control rules by using concepts defined in the ontology. This approach provides a first step in integrating privacy contraints into access control mechanisms while taking into account the context and role of the user requesting the access. In credential-based access control, the idea is to build a trust framework in which service providers use a user's credentials to detremine what data to release to the user. Ardangaet al. suggest that the specification of how these credentials are authenticated be based on some formalisation that determines which attributes of the information have to be disclosed and to whom. Therefore, a key advantage of credential-based mechanisms is that they allow the user more control over their data and consequently gives the users more privacy.

However, it is worth noting that both credential and privacy based systems need some form of record of a user in order to decide whether or not to grant access to that data. This implies that a user needs to provide some information about themselves in order to access the data which in certain cases may expose them. For instance, in the pay-per-view movies scenario that we described in Section 2.2 (see Figure 5), a user may not be comfortable with allowing Jane to know that they like watching "Horror" movies. So, if Jane bases access to the movies on a credentials system she may not attract as many clients as

Table 1. Comparison: DAC, MAC, RBAC, and XACML

	DAC	MAC	RBAC	XACML
Control Point	User	Server	Server	Server/User
Authentication (Control Point)	User	Server	Server	Server/User
Review of Access Rights	User	Server	Server	Server/User
Access Right Propagation	User	Server	Server	Server/User
Access Right Revocation	User	Server	Server	Server/User
Information Flow Control	None	Yes	Yes	None unless security policy specified
User-reliant Security Policy	Yes	No	No	No unless authorized in security policy
Extension for Privacy	No	To some extent via information flow control	Possible through specialized role definition	Yes – Privacy policy specification and enforcement via the policy enforcement point and the policy decision point

she would have if she required less information to grant access to the movies. It is also worth noting that, as with the previous approaches that we discussed, the specification languages and/or frameworks assume a static environment where changes in access control policies are generally effected manually by a security administrator. When security policy combinations involve different domains handling the conflicts that arise might require dynamic adjustments to the combined security policy. Resolving these conflicts to establish a global security policy that satisfies the minimum requirements of the security domains involved, requires an access control scheme that is able to redefine the constraint rules adaptively. While XACML provides features to specify a broad range of policies, a formal specification is still needed to define constraint rules adaptively in order to enforce privacy on the Web.

The common pattern inherent in all the approaches discussed above is the inability to predict privacy violation scenarios mainly because these approaches need to be extended to handle situations of changing security requirements adaptively. For instance, Norton's Symantec Antivirus software is taking steps towards building pre-emptive anti-virus software that incorporates adaptability by using machine learning and data mining techniques. This is an indication that professional organizations also recognize the need for an evolution towards adaptive security mechanisms (Harrison, Munro, & Spiller, 2007), (Chess, 2005). Adaptive intrusion detection algorithms are also still at a budding stage but the idea of moving towards schemes that can adjust to changing security requirements and enforce privacy is inherent in all these approaches.

Table 1 summarizes our discussion of the DAC, MAC, RBAC, and XACML approaches to access control in relation to privacy enforcement. From the table below, it can be noted that one of the key reasons that these models fail to enforce privacy effectively is that access control is typically handled from the server's end. Therefore, users are not rights to allow them to manage their identities and privacy. The DAC model offers users some autonomy but as discussed earlier (see Figure 1, Section 2.1), poses a problem of information flow control.

This discussion illustrates that although no single access control scheme can be designed to handle every possible security scenario, web-based security scenarios are increasingly difficult to predict and control manually. Privacy violations typically arise because of mismanaged access control constraints and so the dynamic nature of the Web adds a further complication to the problem. In these cases, therefore, the needs for good security and consequently privacy enforcement are strongly intertwined with performance. This is because the delays created in trying to address new situations manually can be exploited maliciously and so a lead to privacy violations. In the next section we consider cryptographic access control schemes as a method of enforcing privacy and consider some extensions to allow for adaptability in situations of changing security requirements. Cryptographic access control schemes offer the advantage of being simpler to model mathematically and so lessen the security administrator's burden of security policy specification. In the next section we briefly explain how hierarchical cryptographic access control schemes are designed to work and proceed in Section 4 to discuss extensions for privacy enforcement and adaptability.

CRYPTOGRAPHIC ACCESS CONTROL FOR PRIVACY ENFORCEMENT

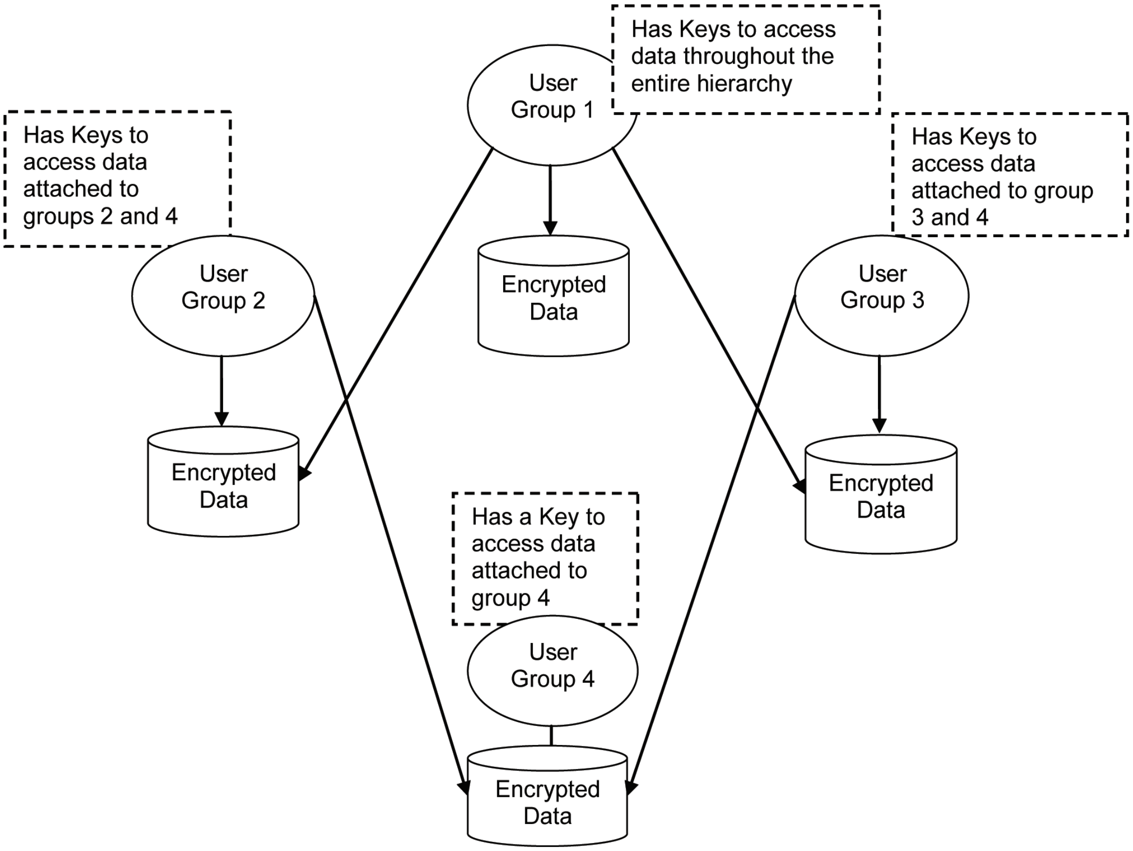
Hierarchical cryptographic access control (CAC) schemes emerged as an attempt to design MLS models that are more general and capable of providing security in different contexts without requiring extensive changes to the fundamental architecture (Akl & Taylor, 1983), (Mackinnon, Taylor, Meijer, & Akl, 1985), (De Capitani Di Vimercati, Foresti, Jajodia, Paraboschi, & Samarati, 2007). For instance, in situations that require data outsourcing, CAC schemes are useful because the data can be double encrypted to prevent a service provider from viewing the information yet be able to run queries or other operations on the data and return a result to a user who can decrypt the data using the keys in their possession (De Capitani Di Vimercati, Foresti, Jajodia, Paraboschi, & Samarati, 2007). In this case, CAC schemes are a good way of enforcing privacy and improving the performance of the access control scheme. The performance improvements come from the fact the encryption ensures data privacy and saves the data owner from having to dedicate management resources to looking after the data.

CAC schemes are typically modeled in the form of a partially ordered set (poset) of security classes that each represents a group of users requesting access to a portion of the data on the system. Cryptographic keys for the various user groups requiring access to part of the shared data in the system are defined by classifying users into n disjoint security classes U_i, represented by a poset $(S, \prec=)$, where $S=\{U_0,U_1,\ldots,U_{n-1}\}$ (Akl & Taylor, 1983). By definition, in the poset, $U_i \prec= U_j$ implies that users in class U_j can have access to information destined for users in U_i but not the reverse. In the following paragraphs we present the two models on which CAC schemes are designed and discuss some approaches for performance improvements.

Hierarchical Cryptographic Access Control Models

Hierarchical Cryptographic Access Control (CAC) models are typically designed on the basis of the concept of posets and are generally divided into two main categories: independent and dependent CAC schemes. Independent CAC schemes originate from the multicast community where the concern is securing intra-group communications efficiently. In these protocols, the focus is on how to manage keys within a group in a way that minimizes the cost of key distribution when the membership of the group changes (Yu, Sun, & Liu, 2007). The reason for updating the keys is to prevent users who have left the group from continuing to access information available to group members. This aligns itself with our theme of privacy enforcement because in essence, we

Figure 7. An example of an independent CAC model


Has Keys to access data throughout the entire hierarchy
User Group 1
Encrypted Data
Has Keys to access data attached to groups 2 and 4
User Group 2
Encrypted Data
Has Keys to access data attached to group 3 and 4
User Group 3
Encrypted Data
Has a Key to access data attached to group 4
User Group 4
Encrypted Data

want CAC schemes to control access to the data by preventing all users who are not authorized to view information from accessing it.

Independent CAC schemes approach hierarchical access control by assigning each security class all the keys they need to access information both at their level and below. Accesses are granted only if the user requesting access holds the correct key (Akl & Taylor, 1983), (Atallah & Frikken, 2006). While this method of CAC is easier to implement in practical systems because of its flexibility, the cost of key distribution as well as the possibility of violations, both security and privacy, due to mismanaged or intercepted keys, is higher than that in dependent CAC schemes (Hassen, Bouabaallah, Bettahar, & Challal, 2007). In fact, in the worst case scenario where all the keys in the hierarchy are updated, $2n+1$ keys are redistributed (where n represents the maximum number of security classes in the hierarchy), making key re-distribution more costly than in dependent CAC schemes where only n keys are redistributed (Hassen, Bouabaallah, Bettahar, & Challal, 2007), (Yu, Sun, & Liu, 2007).

As shown in Figure 7 the data is encrypted to ensure that only the users in possession of the correct keys are allowed access. In order to access the encrypted data a user belonging say to Group 1 will have to use the required key to download and decrypt the data to which access is sought. Since these keys might be available to several users at a time, each time the group membership changes, the keys affected by the change are replaced and the data reencrypted. This is to prevent the departed

Table 2. Key management models: Comparison (Kayem, 2008)

	Dependent Model	**Independent Model**
Security	Fewer keys distributed	More keys distributed
Encryption Cost	More re-encryption	Less re-encryption
Effect of Rekeying	Changing one key implies updating the whole hierarchy	Change only affected keys, and distribute to users requiring the keys
Key Distribution Cost (Number of keys transmitted)	n keys	2n + 1 keys

user from continuing to access the date. When a user who holds many of the keys departs this is a problem because it requires replacing and encrypting several portions of the data which is time-consuming. As well, as mentioned before, delays in encrypting the data can be exploited maliciously to provoke privacy violations.

A good way to alleviate these problems is to design the CAC scheme in a way that minimizes the number of keys distributed to any security class in the hierarchy. This model, typically referred to as the dependent key management (DKM) scheme, defines a precedence relationship between the keys assigned to the security classes in the hierarchy whereby keys belonging to security classes situated at higher levels in the hierarchy can be used to mathematically derive lower level keys. Access is not possible if the derivation function fails to yield a valid key. So for instance, in Figure 7, the data associated with Group 2 would be inaccessible to users in Group 1 if the assigned key does not allow them to mathematically derive the key with which the data at Group 2 was encrypted. This minimizes the cost of key assignment and distribution because a user only needs to hold one key from which all the other required keys can be derived. However, the problem of costly encryptions to cope with key updates remains (Hassen, Bouabaallah, Bettahar, & Challal, 2007), (Yu, Sun, & Liu, 2007), (Yu, Sun, & Liu, 2007). In Table 2 we summarize the differences between the independent and dependent CAC models in relation to the implications to privacy enforcement.

From Table 2, we note that the independent CAC model's main drawback lies in the cost of key distribution while the dependent CAC model's main drawback lies in the cost of encryption to cope with key updates. We also note, that either drawback does not help the case for privacy enforcement because encrypting large volumes of data can be quite time consuming and so create a wide window of vulnerability during which the data is unprotected. A solution would be to withdraw the data and handle encryption offline but this creates a problem of data availability that might affect the system's ability to meet its service level agreements (Meziane & Benbornou, 2010).. On the other hand with the independent CAC model, key distribution and the potential for interception also poses a privacy risk. When highly sensitive data is concerned this can become a serious problem because tracing illegal key usage is a challenging problem. For instance, if our hypothetical scenario of an e-business application is extended to include a health insurance service, it would be unwise to implement a CAC scheme that increases the risk of exposure of patient data. In Section 3.2 we discuss some approaches that have been proposed to alleviate both the problems of key distribution and costly encryptions to handle updates in group membership to ensure data privacy.

Other CKM Schemes

In order to minimize the amount of information distributed during key replacements variants of independent CAC model that appear in the literature (Shen & Chen, 2002), (Kuo, Shen, Chen, & Lai, 1999) propose ways of making key updates (distributions) easier and more secure by encrypting the keys that are to be distributed with a public key. The encrypted keys are then placed in some public location and a secret key is transmitted to each group. Access to a particular set of keys is only allowed if a user is in possession of the correct secret key. This makes it easier to exclude users that are compromised and reduces the number of keys distributed but the advantage comes at the cost of added public key information that increases the chances of an adversary correctly guessing at the secret keys being used (Crampton, Martin, & Wild, 2006).

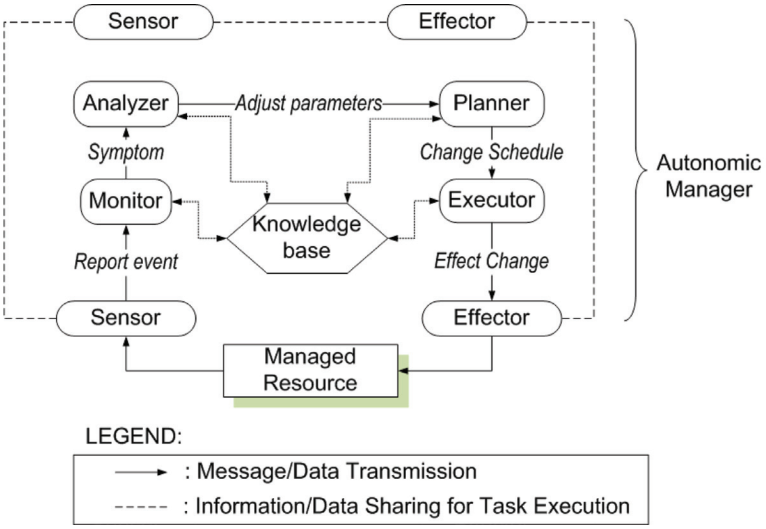
Other approaches in the area of secure group communications have proposed batching key update requests to minimize the long term cost of rekeying (Li, Yang, Gouda, & Lam, 1999). Batching operates by accumulating requests for key updates during a preset interval at the end of which the keys are then replaced. Although this improves on the cost of rekeying, it widens the vulnerability window of the key management scheme.

Still along this line of batching key update requests, Crampton has suggested using lazy re-encryption to minimize the cost of data re-encryption (Crampton, 2007). Lazy re-encryption operates by using correlations in data updates to decide whether or not to update a key and re-encrypt the old data when group membership changes. In this way, since data re-encryption accounts for the larger part of the cost of key replacement, re-encryption is only performed when the data changes significantly, after a user's departure, or if the data is highly sensitive and requires immediate re-encryption to prevent the user from accessing it. The problem of having to re-encrypt the data after a user's departure still remains. Moreover, if the file is a sensitive file that does not change frequently, lazy re-encryption can allow a malicious user time to copy off information from the file into another file and leave the system without ever being detected.

More recently, Ateniese et al. (Ateniese, Fu, Green, & Hohenberger, 2006) have proposed an improvement on the variant of IKM schemes that Blaze et al. (Blaze, Bleumer, & Strauss, 1998) proposed in 1998 whereby proxy-reencryption is used to assign users access to particular files associated with another user or group. Basically, each group or user in the hierarchy is assigned two keys (a master key and a secondary key). The secondary key is used to encrypt files and load them into a block store where they are made accessible to users outside of the group. In order to access encrypted data from the block store a user must retrieve the encrypted data and present both the encrypted data and their public key to an access control server. The access control server re-encrypts the data in a format that is decryptable with the user's secret key, only if the presented secondary public key authorizes them access. The problem of having to re-encrypt, update and distribute new keys when group membership changes remains.

Therefore irrespective of how a CAC scheme is designed, rekeying is handled by replacing the affected key and re-encrypting the associated data. Rekeying ensures that data privacy is always enforced but the rekeying process is time consuming which increases the vulnerability window. As mentioned before, an increased vulnerability window size makes a CAC scheme susceptible to two issues: delayed response time in handling key updates and an increased possibility of privacy. In Section 4, we present two approaches to ensuring privacy under changing security conditions without impeding performance. The first approach alleviates the cost of encryption by using data replication as a fault tolerance mechanism. Data replication is handled by predicting encryption and key update requirement. This indicates that by extending a CAC scheme to allow adaptability to a situation of changing security requirements, an access control scheme can meet its goals of

Figure 8. The autonomic computing feedback control loop (Kayem, 2008)



privacy. The second approach evokes the problem of collusion which basically occurs when two or more users gain access to unauthorized data by performing illegal key combinations. Our second proposition shows how one might prevent such violations of privacy by monitoring key assignments to prevent collusion susceptible keys from being assigned to users.

SELF-PROTECTING CRYPTOGRAPHIC ACCESS CONTROL

Our self-protecting cryptographic access control scheme is based on the paradigm of autonomic computing. This paradigm emerged in a bid to design applications with the ability to adaptively handle scenarios of varying complexity (Kephart & Chess, 2005). From our discussions in the previous sections, it is safe to say that there is a growing need for access control mechanisms with the ability to adjust to new scenarios adaptively.

Autonomic Computing and Privacy

Security via the autonomic computing paradigm was first proposed by Chess et al. in 2003 (Chess, 2005). In order to address the challenge of handling complex situations for which security needs to be ensured, they suggest using the paradigm of autonomic computing that IBM proposed in 2001 (Kephart & Chess, 2003), (Kephart & Chess, 2005). The paradigm of autonomic computing supposes that a system can be designed to self-regulate by using automatic reactions to defend, optimize and heal. The functions of an autonomic system are modeled using a feedback control loop that has two major components: the autonomic manager and the managed resource. The autonomic manager adjusts the behavior of the managed resource on the basis of recorded observations.

The autonomic model shown in Figure 8 is comprised of six basic functions: the sensor, monitor, analyzer, planner, executor, and effector. The sensor captures information relating to the behavior of the managed component and transmits this information to the monitor.

The monitor determines whether or not an event is abnormal by comparing observed values to threshold values in the knowledge base. The analyzer, on reception of a message from the monitor, performs a detailed analysis to decide what parameters need to be adjusted and by how much, and transmits this information to the planner where a decision is made on the action to be taken. The executor inserts the task into a scheduling queue and calls the effector to enforce the changes on the managed resource in the order indicated by the planner.

Autonomic computing aims to provide survivability and fault-tolerance for security schemes by allowing access control schemes to self-manage and self-configure to minimize security violations (Chess, 2005), (Johnson, Sterritt, Hanna, & O'Hagan, 2007). Johnston et al. (Johnson, Sterritt, Hanna, & O'Hagan, 2007) propose a preliminary approach that uses reflex autonomic computing in the development of a multi-agent security system. This is an interesting approach to self-protecting security, but the authors indicate that real-world implementations of their prototype system would require additional security controls and the prototype do not allow a security class to operate independently. As Moreno et al. (Moreno, Sanchez, & Isern, 2003) have pointed out, the connection to the rest of the system is lost. We note also that this work on autonomic access control focuses mainly on security policy definitions and restrictions on the messages sent and received by entities (users and/or agents) in the system as opposed to handling cases requiring some form of adaptability in the access control policy. Other approaches include trust and access control negotiation frameworks that are aimed at enforcing privacy in conflict situations in collaboration environments (Ryutov, Zhou, Neuman, Leithead, & Seamons, 2005), (Smari, Zhu, & Clemente, 2009), (Kuang & Ibrahim, 2009). The problem of designing adaptive access control schemes to support security policy definitions in ways that allow an access control mechanism to cope with changing scenarios still needs to be addressed.

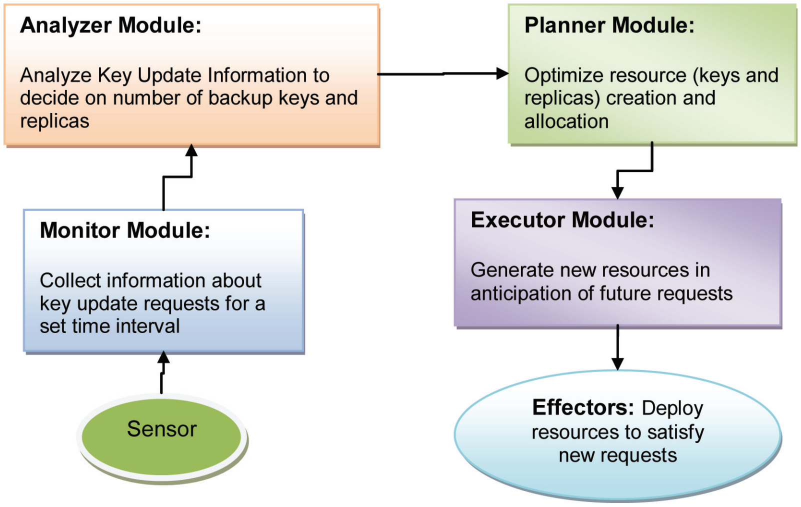
Data Replication for Self-Protecting Cryptographic Access Control

We use an example of a simple read-intensive scenario to explain how data replication enhances the performance of CAC schemes in scenarios requiring key updates (Kayem, Martin, Akl, & Powley, 2008). Our CAC scheme is supported by a feedback control loop that is inspired by the autonomic computing paradigm (Chess, 2005). As shown in Figure 9, the CAC scheme can be supported by a framework that is structured in the form of an autonomic computing feedback control loop (see Figure 8). We note that this framework is formed simply by restructuring the model given in Figure 8 to suit the specific case of key updates.

The monitor module observes the behavior of the users on the system via a sensor that captures requests for key updates that are emitted either by a user or the security administrator. The monitor studies the rate at which key update request arrive over a given period and then transmits the information to the Analyzer module. At the Analyzer module, a computation to predict future the rate of future key update requests is made and this prediction information is transmitted to the Planner module. The Planner module computes the optimal number of resources (keys and encrypted replicas) that the security system needs to generate in order to cope with future key update requests. Once this is done a message is sent to the Executor module to generate the keys and replicas in anticipation of the key update requests. The Effector takes care of distributing new keys, making available new encrypted replicas, and deleting the old keys as well as the associated data.

An example of how this works is given in Figure 10 where we have a situation in which the feedback control loop is embedded in the key server and the Monitor module observes user behavior during a

Figure 9. An autonomic computing framework for handling key update requests


Analyzer Module:
Analyze Key Update Information to decide on number of backup keys and replicas
Planner Module:
Optimize resource (keys and replicas) creation and allocation
Monitor Module:
Collect information about key update requests for a set time interval
Executor Module:
Generate new resources in anticipation of future requests
Sensor
Effectors: Deploy resources to satisfy new requests

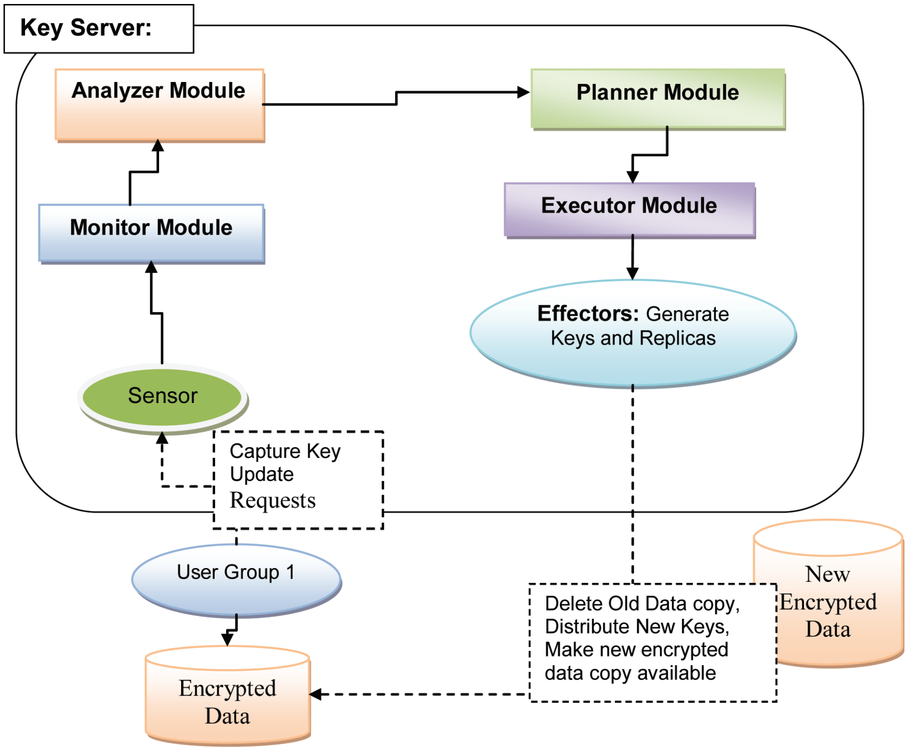
period W_1. The information is transmitted to the Analyzer module after the observation period is over and it is determined that one rekey request is likely to arrive from User Group 1 during a future monitoring period W_x. This information is transmitted to the Planner module where it is determined that in order to handle the rekey request one backup key and replica need to be generated in anticipation of this request. As depicted in Figure 10 the key server, via the Executor module, creates a new backup key, for the User Group 1, and transmits this key to the Effector where it is kept in a secret registry. The Effector then generates a new copy of the data and encrypts it with the new key. When a request for a key update arrives from User Group 1, the Effector will proceed to destroy the old data and replace it with the new encrypted version. The users remaining in the group will then be sent a copy of the key needed to decrypt the new data. Copy consistency is not a real concern in this case because our example is one of a read-intensive scenario. Updates are only authorized by the security administrator. So by anticipating key update requests we can alleviate the delays caused by key update requests and consequently avoid privacy violations that occur because of malicious exploitations of a wide vulnerability window.

Collusion Resolution in an Access Control Hierarchy

A common problem that leads to privacy violations in CAC schemes is the one that occurs when two or more users illegally compute a key to access information that they are not authorized to access. This problem also known as the collusion attack is one that all key management algorithms supporting CAC schemes seek to avoid. Checking assigned key to ensure that collusion is avoided, is a challenging problem and particularly so under changing security conditions. For instance, it is difficult to check if a new key that is assigned in response to a key update request cannot be used to provoke a collusion attack. Therefore, guaranteeing privacy under these conditions is also a challenging problem. We propose solving this problem with a self-protecting scheme that is inspired by the autonomic computing paradigm.

Basically, as in the framework we described earlier to handle replication, we will create a framework that is structured in the form of the autonomic computing feedback control loop. In the

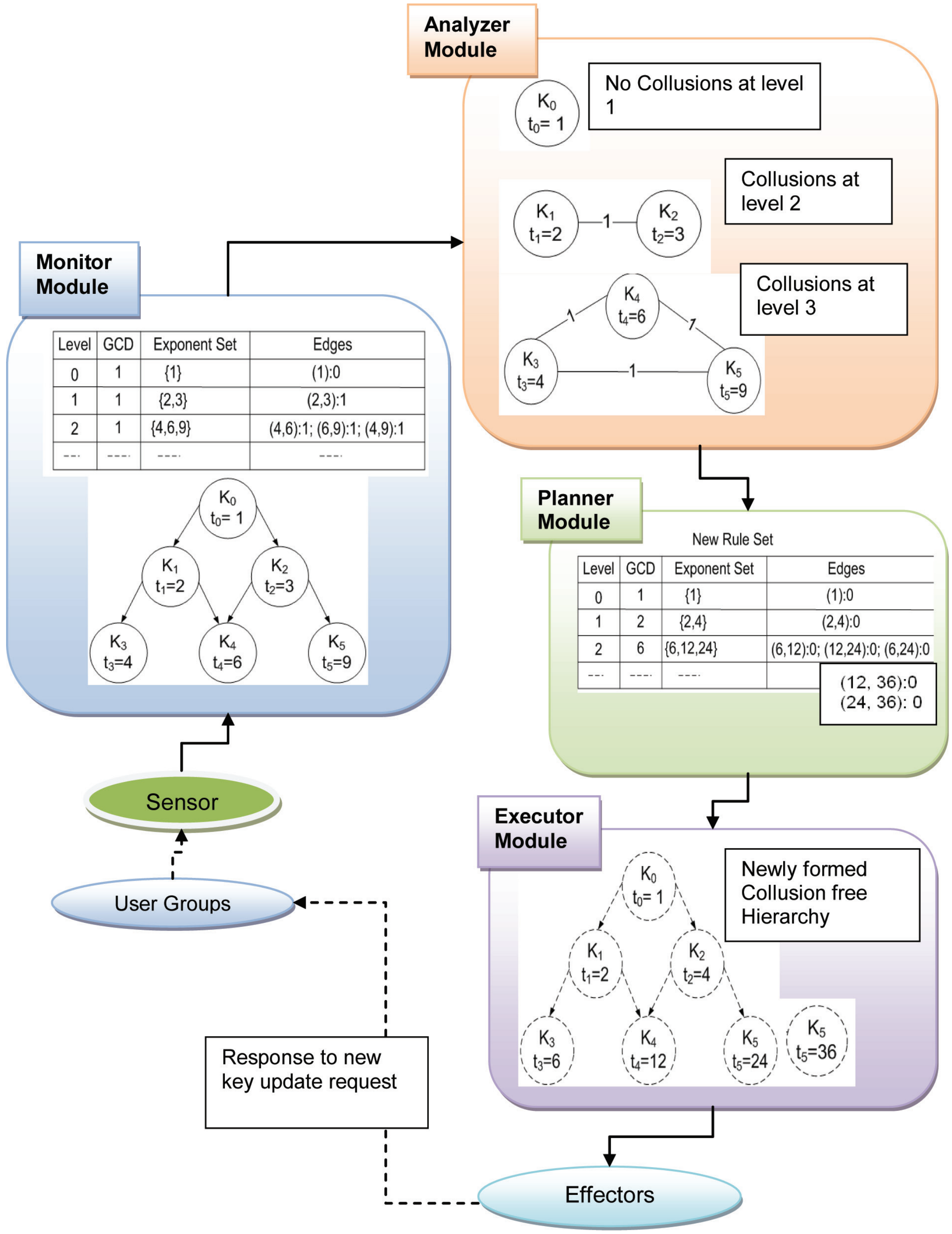
Figure 10. Data replication and rekeying to handle update requests (an example)


Key Server:
Analyzer Module
Planner Module
Executor Module
Monitor Module
Effectors: Generate Keys and Replicas
Sensor
Capture Key Update Requests
User Group 1
New Encrypted Data
Delete Old Data copy, Distribute New Keys, Make new encrypted data copy available
Encrypted Data

framework key updates are handled preemptively by monitoring, via a sensor, the rate at which key update requests occur. This information is then used by the Analyzer module to determine which parts of the CAC hierarchy need new keys and new keys are selected in a way that prevents collusions from occurring. The Analyzer module then calls the Planner module to generate a rule set that represents the new hierarchy for the new keys. This information is shared with the Executor module that keeps track of the old hierarchy and the new hierarchy. When a key update request needs to be handled the Effector assigns the affected user group a new key that is provided by the Executor module.

An example of how this can be done occurs when a hierarchy of keys is created using the key generation function that Akl and Taylor proposed (i.e. $K_i = K_0^{t_i} \bmod M$, where t_i is an integer value that is assigned to a security class and $M$the product of two large primes). In this case, the exponent set is V = {1, 2, 3, 4, 6, 9}. According to the Akl and Taylor scheme (Akl & Taylor, 1983), the greatest common divisor (GCD) at level 1 is 1, level 2, 1 and level 3, 1; and since the GCDs at all three levels are the same, it implies that the keys at levels 1 and 2 will be collusion liable. We note also that in each of the cases collusion is possible either because the combined GCD at a given level yields a value that is a divisor of some or all of the exponents at the higher levels (here levels 0 and/or 1) or is a divisor of the combined GCD of the exponents at levels 0 and/or 1. As shown in Figure 11 the monitor maintains a graph of the current key assignments and notes in this case that the potential assignment of keys K_2 and K_9 are likely to provoke collusions. This is indicated by the table (see Figure 11 – Monitor Module) that shows that collusions be-

Figure 11. A feedback control loop framework for collusion resolution



Level	GCD	Exponent Set	Edges
0	1	{1}	(1):0
1	1	{2,3}	(2,3):1
2	1	{4,6,9}	(4,6):1; (6,9):1; (4,9):1
---	----	----	----

New Rule Set

Level	GCD	Exponent Set	Edges
0	1	{1}	(1):0
1	2	{2,4}	(2,4):0
2	6	{6,12,24}	(6,12):0; (12,24):0; (6,24):0
---	----	----	(12, 36):0 (24, 36): 0

tween keys K_3, K_4, and K_9 are possible because of the exponent choice.

In order to remove the collusion-liable keys, the collusion resolution algorithm, that is located in the Analyzer module, operates as follows. In the first step, the algorithm is executed for level 0.

There is nothing higher than t_0, t_0 retains the value of 1. At level 1, as shown in Figure 11 (see Analyzer Module), the GCD {2, 3} = 1 which is equal to t_0, this indicates that there is a possibility that the keys K_1 and K_2 that are formed from t_1 = 2 and t_2 = 3 can be used to provoke a collusion attack. The collusion prevention algorithm, at the Planner Module, prevents this occurrence, by selecting a random value for t_2 such that GCD $\{t_1,t_2\}$ 6= 1 and both t_1 and t_2 remain multiples of t_0. A randomly chosen value of 4 is selected and since GCD{2, 4} = 2 which is not a divisor of the GCD at level 0, and both are multiples of t_0, t_2 retains the randomly assigned value of 4.

Likewise at level 2, as shown in Figure 11(see Analyzer Module), first t_3 = 4 which has been assigned to t_2, and GCD{4, 9} = 1 = t_0, GCD{4, 6} = 2 = t_1 and t_5 = 9 is not a multiple of t_2. Hence, the collusion prevention algorithm needs to re-assign integer values to t_3, t_4, and t_5 such that GCDs of the pairs of t_i at level 2 are not a factor of any GCDs at levels 0 and 1 and that additionally, the divisibility condition continues to hold. The random assignments in Figure 11(see Analyzer Module) present two possibilities of assignments, t_3 = 12, t_4 = 24, t_5 = 36 and t_3 = 6, t_4 = 12, t_5 = 24. So one of the sets is chosen and finally in the new assignment of exponents is such that collusion is prevented (see Planner Module – Figure 11). The Executor module is then called to construct the key graph (see Figure 11 – Executor Module) and make the new keys available to the Effector when the need arises.

It is obvious from this illustration that several different combinations would generate correct and valid independent sets. We could use a heuristic to control the size of the GCD of the exponent pairs at all of the levels in the hierarchy. However, we do not consider having different exponent sets to be a disadvantage but rather an advantage because different sets could be generated off line and attributed when there is a demand for a new set of keys. This would be the case when a user joins or leaves the system and the original structure is maintained. In this way the method can in the best case contribute to an improvement in the efficiency of the key generation scheme (Mackinnon, Taylor, Meijer, & Akl, 1985).

The approach we adopt for identifying collusion liable keys is to map the keys on to a graph where the vertices represent the keys and the edges the probability of their being combined to provoke collusions. Adjacent keys indicate a higher likelihood of attack than non-adjacent keys. A collusion removal algorithm that is based on the principle of computing an independent set from the vertices in a graph is used to remove collusion liable keys from a key set.

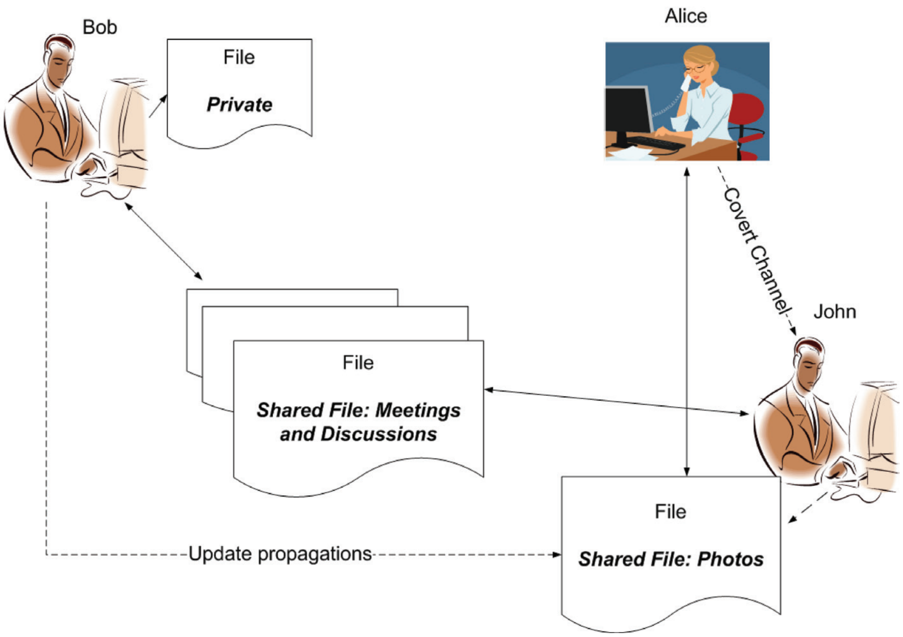
Using the independent set approach to resolve this problem shows that the problem of removing the collusion liabilities in a key set is similar to the classic graph theory problem of computing a largest independent set. As the problem is NP-hard (Levitin, 2003), a heuristic was used to achieve an efficient (but perhaps suboptimal) solution in polynomial time. Nevertheless, as illustrated, the solution is feasible. It should be noted also that it is a good idea to have this work with the data replication approach that we described in Section 4.2.

FUTURE RESEARCH DIRECTIONS

The open problems that emerge from our discussions in the preceding sections include the prevention of inference and covert channels and so we discuss each in a little more detail below.

Inference and Covert Channels: Inference and covert channels can be categorized as problems of internal and privacy violations. These problems

Figure 12. Indirect information access via covert channels (Kayem, 2008)



occur when a valid (authentic) user abuses his/her privileges either to transfer information to a user that is not authorized to view the information or to illegally gain access to data. Basically inference occurs when users are able to piece together information through legal access channels, in order to deduce information at another security level (Brodsky, Farkas, & Jajodia, 2000), (Farkas & Jajodia, 2002). Covert channels represent another manifestation of indirect violations of security policies in the context of hierarchical access control (Rjaibi, 2004). A covert channel refers to a transfer of information, from one level in the hierarchy to another, which violates the partial order between the security classes. This occurs when, for example, a higher level user employs their legal key to access information at their level and then deposits this information in a memory or storage location that can be accessed by the user with the lower security clearance.

Figure 12 depicts a scenario based on our hypothetical example of an e-business application that uses a social networking environment (e.g. Facebook) as a backbone. In this case, we consider two examples where indirect access to information is achieved via inference channels and covert channels. A user, say, Alice, happens to be on the "friends" profile of several users. Imagine for instance that she seeks to obtain information on another user, say Bob, whose "friends" profile she does not belong to. She however is on John's "friends" profile and John in turn is on Bob's "friends" profile, so when Bob posts information to his "wall" (environment accessible by all his friends), John, can read the information. Now suppose that John comments on a message Bob has posted, Alice can, from reading information on John's public space, infer information related to Bob. For instance, she may infer that if Bob is watching certain movies that he purchased from Jane's pay-per-view service (see Figure 5) then Bob will probably be going out to watch a certain other similar movie at a certain time. Furthermore with a little cooperation from John, a covert channel can be opened between herself and John that allows her to directly receive all the updates that Bob propagates. These weaknesses are not easy to handle with standard CAC schemes and also usually occur because of weakness in functional, multivalued and join dependencies in the databases that support these systems (Brodsky, Farkas, & Jajodia, 2000), (Pfleeger & Pfleeger, 2003). Moreover, these weaknesses result in very subtle privacy violations that are difficult to detect. Handling problems like these manually is also challenging and time-consuming because it is hard to determine what kinds of inferences can be made from the information that is available publicly. We however, believe that embedding functions that allow access control schemes to adjust to new scenarios adaptively is a good starting point for addressing these problems.

CONCLUSION

The discussion in this chapter has been centered on the extensions that need to be made to standard access control schemes in order to enforce privacy on the Web. We highlighted the pros and cons of each one discussing also some of the extensions that have been made to some to cater to privacy needs on the Web. The most extensive work has been on the Extensible Access Control Markup Language (XACML) with proposals to incorporate privacy checking mechanisms. We noted that one of the challenges in enforcing privacy is the fact that the Web environment is inherently dynamic and so security requirements can change on-the-fly. This can affect privacy enforcing schemes negatively leading to vulnerabilities that can be exploited maliciously. We noted also that some of the key problems that result in privacy violations include: inefficiencies in management of access control schemes and a lack of inbuilt mechanism to handle dynamic scenarios adaptively. Moreover, until recently, the assumption was that a well specified security solution will implicitly enforce privacy.

In this chapter we have shown, using two examples from cryptographic access control (CAC) schemes that the assumption that a good security scheme is privacy enforcing, is not always true. We proposed extending CAC schemes with an autonomic computing framework that is structured in the form of a feedback control loop to ensure self-protection, in a system, against privacy violations. Specifically we considered the problem of efficient key updates and looked at how to assign keys in a way that prevents collusion and also makes for less expensive data encryptions.

In order to address these concerns, we proposed embedding in the autonomic framework fault tolerance mechanisms like data replication and backup keys to anticipate key update requests. Replication is used to overcome the cost of encryption and consequently overcome the associated risk of privacy violations due to delays in re-encryptions of the data with the newly generated keys. Backup keys are created and checked preemptively to prevent assignments that can lead to privacy violations through collusions between assigned keys. The collusion resolution strategy is to map the keys onto a key graph and then compute and independent set of the graph using a heuristic. The autonomic computing framework allows the CAC scheme to enforce self-protection by simply monitoring the rate at which key updates are required and generating keys as well as encrypted replicas to respond to the requests preemptively as opposed to on demand. The advantage of this approach is that it allows the access control scheme to adjust its behavior based on the scenario with which it is faced making privacy enforcement somewhat simpler.

REFERENCES

Ahmed, Q., & Vrbsky, S. V. (2002). Maintaining security and timeliness in real-time database system. *Journal of Systems and Software*, *61*, 15–29. doi:10.1016/S0164-1212(01)00111-X

Akl, S. G., & Taylor, P. D. (1983). Cryptographic solution to a problem of access control in a hierarchy. *ACM Transactions on Computer Systems*, *1*(3), 239–248. doi:10.1145/357369.357372

Ardagna, C. A., De Capitani di Vimercati, S., Paraboschi, S., Pedrini, E., & Samarati, P. (2009). An XACML-based privacy-centered access control system. In *Proceedings of the First ACM Workshop on information Security Governance* (pp. 49-58). Chicago, IL: ACM.

Ardanga, C. A., De Capitani di Vimercati, S., Paraboschi, S., Pedrini, E., & Samarati, P. (2010). Enabling privacy preserving credential-based access control with XACML and SAML. *Proceedings of the 10th IEEE International Conference on Computer and Information Technology, CIT 2010,* (pp. 1090-1095). Bradford, West Yorkshire, UK.

Atallah, M., Frikken, K. B., & Blanton, M. (2009). Dynamic and efficient key management for access hierarchies. *ACM Transactions on Information and System Security*, *12*(3), 190–202. doi:10.1145/1455526.1455531

Atallah, M. J., & Frikken, K. (2006). Key management for non-tree hierarchies. *ACM Symposium on Access Control Models and Technologies* (pp. 11-18). Lake Tahoe, CA: ACM.

Ateniese, G., De Santis, A., Ferrara, A. L., & Masucci, B. (2006). Provably-secure time-bound hierarchical key assignment schemes. In *Proceedings of the 13th ACM Conference on Computer and Communications Security* (pp. 288-297). Alexandria, Virginia, USA, October 30 - November 03, 2006: CCS ‘06. New York, NY: ACM.

Ateniese, G., Fu, K., Green, M., & Hohenberger, S. (2006). Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security*, *9*(1), 1–30. doi:10.1145/1127345.1127346

Bell, D., & Lapadula, L. (1973). *Secure computer systems: Mathematical foundtaions and model*, (p. 2). MITRE report, MTR2547.

Bertino, E., & Sandhu, R. (2005). Database security - Concepts, approaches, and challenges. *IEEE Transactions on Dependable and Secure Computing*, *2*(1), 2–19. doi:10.1109/TDSC.2005.9

Biba, K. (1977). *Integrity considerations for secure computer systems.* Bedford, MA, April 1977: Technical Report ESD-TR-76-372 ESD/AFSC, Hanscom AFB.

Blaze, M., Bleumer, G., & Strauss, M. (1998). Divertible protocols and atomic proxy cryptography. *In Proceedings of EUROCRYPT'98*, (pp. 1403:127-144).

Brewer, D. D., & Nash, M. (1988). The Chinese wall security policy. *IEEE Symposium on Security and Privacy* (pp. 206-214). Oakland, CA: IEEE.

Brodsky, A., Farkas, C., & Jajodia, S. (2000). Secure databases: Constraints, inference channels and monitoring disclosures. *IEEE Transactions on Knowledge and Data Engineering*, *12*(6), 900–919. doi:10.1109/69.895801

Byun, J.-W., & Ninghui, L. (2008). Purpose based access control for privacy protection in relational database systems. *The VLDB Journal*, *17*, 603–619. doi:10.1007/s00778-006-0023-0

Chadramouli, R. (2003). A policy validation framework for enterprise authorization specification. *ACSAC: In Proceedings of the 19th Annual Computer Security Applications Conference* (pp. 319-328). Washington, DC: IEEE Computer Society.

Chess, D. M. (2005). Security in autonomic computing. *SIGARCH Comput. Archit. News*, *33*(1), 2–5. doi:10.1145/1055626.1055628

Chien, H.-Y. (2004). Efficient time-bound hierarchical key assignment scheme. *IEEE Transactions on Knowledge and Data Engineering*, *16*(10), 1301–1304. doi:10.1109/TKDE.2004.59

Clark, D. R., & Wilson, D. (1987). A comparison of commercial and military computer security policies. In *Proceedings 1987 IEEE Symposium on Security and Privacy* (pp. 184-194). Oakland, CA: IEEE.

Corbi, A. G. (2003). The dawning of the autonomic computing era. *IBM Systems Journal*, *42*(1), 5–18. doi:10.1147/sj.421.0005

Crampton, J. (2007). Cryptographically-enforced hierarchical access control with multiple keys. *In Proceedings 12th Nordic Workshop on Secure IT Systems (NordSec 2007)*, (pp. 49-60).

Crampton, J., Martin, K., & Wild, P. (2006). On key assignment for hierarchical access control. *In Proceedings 19th IEEE Workshop on Computer Security Foundations* (pp. 98-111). S. Servolo Island, Italy: IEEE.

Dai, Y.-S. (2005). Autonomic computing and reliability improvement. *In Proceedings 8th IEEE Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'05)*, (pp. 204-206).

Das, M. L., Saxena, A., Gulati, V. P., & Phutak, D. B. (2005). Hierarchical key management scheme using polynomial interpolation. *SIGOPS Oper. Syst. Rev.*, *39*(1), 40–47. doi:10.1145/1044552.1044556

De Capitani Di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., & Samarati, P. (2007). Over-encryption: Management of access control evolution on outsourced data. *VLDB '07: Proceedings of the 33rd International Conference on Very Large Databases* (pp. 123-134). Vienna, Austria: VLDB Endowment.

De Santis, A., Ferrara, A. L., & Masucci, B. (2008). New constructions for provably-secure time-bound hierarchical key assignment schemes. *Theoretical Computer Science*, *407*(1-3), 213–230. doi:10.1016/j.tcs.2008.05.021

Denning, D. E. (1976). A lattice model of secure information flow. *Communications of the ACM*, *19*(5), 236–243. doi:10.1145/360051.360056

Du, S., & Joshi, J. (2006). Supporting authorization query and inter-domain role mapping in presence of hybrid role hierarchy. *In Proceedings of the 11th ACM Symposium on Access Control Models and Technologies* (pp. 228-236). ACM.

Farkas, C., & Jajodia, S. (2002). The inference problem: A survey. *ACM SUGKDD Explorations Newsletter*, *42*(1), 5–18.

Fernandez-Medina, E., & Piattini, M. (2005). Designing secure databases. *Information and Software Technology*, *4*, 6–11.

Foremski, T. (2010, October 18). *TRUSTe responds to Facebook privacy leaks*. Retrieved October 28, 2010, from http://www.siliconvalleywatcher.com/mt/ archives/2010/10/truste_responds.php

Fujii, K., & Suda, T. (2009). Semantics-based context-aware dynamic service composition. *ACM Trans. Auton. Adapt. Syst.*, *4*(2), 1–31. doi:10.1145/1516533.1516536

Fung, B. C., Wang, K., Chen, R., & Yu, P. S. (2010). Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys*, *42*(4), 1–53. doi:10.1145/1749603.1749605

Ganek, A., & Corbi, T. (2003). The dawning of the autonomic computing era. *IBM Systems Journal*, *42*(1), 5–18. doi:10.1147/sj.421.0005

Gollman, D. (2005). *Computer security*. John Wiley and Sons, Ltd.

Harn, L., & Lin, H. (1990). A cryptographic keys generation scheme for multilevel data security. *Computers & Security*, *9*, 539–546. doi:10.1016/0167-4048(90)90132-D

Harrison, K., Munro, B., & Spiller, T. (2007). Security through uncertainty. *Elsevier -. Network Security*, 4–7. doi:10.1016/S1353-4858(07)70016-8

Hart, E., Davoudani, D., & McEwan, C. (2007). Immunological inspiration for building a new generation of autonomic systems. In *Proceedings of the 1st International Conference on Autonomic Computing and Communication Systems (Autonomics '07). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST,* (pp. 1-9). Brussels, Belgium.

Hassen, R. H., Bouabaallah, A., Bettahar, H., & Challal, Y. (2007). Key management for content acess control in a hierarchy. *Computer Networks*, *51*, 3197–3219. doi:10.1016/j.comnet.2006.12.011

Hengartner, U. (2008). Location privacy based on trusted computing and secure logging. In *Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks* (pp. 1-8). Istanbul, Turkey, September 22 - 25, 2008: SecureComm '08. New York, NY: ACM.

Hsu, C. L., & Wu, T. S. (2003). Cryptanalyses and improvements of two cryptographic key assignment schemes for dynamic access control in a user hierarchy. *Computers & Security*, *22*(5), 453–456. doi:10.1016/S0167-4048(03)00514-5

Hu, V. C., Martin, E., Hwang, J., & Xie, T. (2007). Conformance checking of access control policies specified in XACML. In *Proceedings 31st Annual International Computer Software and Applications Conference, COMPSAC 2007*, (pp. 275-280).

Huang, Q., & Shen, C. (2004). A new MLS mandatory policy combining secrecy and integrity implemented in highly classified secure OS. In *Proceedings 7th International Conference on Signal Processing (ICSP '04)*, (pp. 2409-2412).

Huebscher, M. C., & McCann, J. A. (2008). A survey of autonomic computing - Degrees, models, and applications. *ACM Computing Surveys*, Article 7 (August 2008), 28 pages. http://doi.acm.org/10.1145/1380584.1380585

J-W., B., & Li, N. (2008). Purpose based access control for privacy protection in relational database systems. *The VLDB Journal*, *17*, 603–619. doi:10.1007/s00778-006-0023-0

Jeong, M., & Kim, J. a. (2004). A flexible database security system using multiple access control policies. *IEEE International Conference on Systsem, Man, and Cybernetics* (pp. 5013-5018). IEEE.

Johnson, S. E., Sterritt, R., Hanna, E., & O'Hagan, P. (2007). Reflex autonomicity in an agent-based security system: The autonomic access control system. *4th IEEE International Workshop on Engineering Autonomic and Autonomous Systems (EASe'07)*, (pp. 68-78).

Jonker, C. M., Robu, V., & Treur, J. (2007). An agent architecture for multi-attribute negotiation using incomplete preference information. *Autonomous Agents and Multi-Agent Systems*, 221–252. doi:10.1007/s10458-006-9009-y

Kayem, A. (2008). *Adaptive cryptographic access control for dynamic data sharing environments*. Kingston, Canada: Queen's University.

Kayem, A., Martin, P., & Akl, S. (2010). Enhancing identity trust in cryptographic key management systems for dynamic environments. *Security and Communication Networks, 4*. Retrieved from http://onlinelibrary.wiley.com/doi/10.1002 / sec.164/abstract

Kayem, A., Martin, P., Akl, S., & Powley, W. (2008). A framework for self-protecting cryptographic key management. In *Proceedings of the 2nd IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, (pp. 191-200). Venice, Italy.

Kephart, J. O., & Chess, D. M. (2003). The vision of autonomic computing. *IEEE Computer*, *36*(1), 41–50. doi:10.1109/MC.2003.1160055

Kephart, J. O., & Chess, D. M. (2005). Research challenges of autonomic computing. In *Proceedings 27th International Conference on Software Engineering*, (pp. 15-22). St. Louis, MO, USA.

Knuth, D. E. (1981). *The art of computer programming* (seminumerical algorithms, 2nd ed., vol. 2). Reading, MA: Addison Wesley.

Kuang, T. P., & Ibrahim, H. (2009). Security privacy access control policy integration and conflict reconciliation in health care organizations collaborations. *International Conference on Information Integration and Web-based Applications & Services* (pp. 750-754). Kuala Lumpur, Malaysia: ACM.

Kuo, F., Shen, V., Chen, T., & Lai, F. (1999). Cryptographic key assignment scheme for dynamic access control in a user hierarchy. *IEEE Proceedings. Computers and Digital Techniques*, *146*(5), 235–240. doi:10.1049/ip-cdt:19990311

Lampson, B. W. (1973). A not on the confinement problem. *Communications of the ACM*, *16*(10), 613–615. doi:10.1145/362375.362389

Lecue, F., Delteil, A., & Leger, A. (2008). Towards the composition of stateful and independent Semantic Web services. *SAC '08: In Proceedings of the 2008 ACM Symposium on Applied Computing* (pp. 2279-2285). Fortaleza, Brazil: ACM.

Levitin, A. (2003). *Introduction to the design and analysis of algorithms. Pearson*. Addison-Wesley.

Li, H., Ahn, D., & Hung, P. C. (2004). Algorithms for automated negotiations and their applications in information privacy. *In Proceedings IEEE International Conference* (pp. 255- 262). e-Commerce Technology, 2004. CEC 2004.

Li, X., Yang, Y., Gouda, M., & Lam, S. (1999). Batch rekeying for secure group communications. *WWW10, 99*(7), 525-534.

Lin, T. (2006). Managing information flows on discretionary access control models. *2006 IEEE International Conference on Systems, Man., and Cybernetics*, (pp. 4759-4762).

Lin, T. Y. (2000). Chinese Wall security model and conflict analysis. *24th IEEE Computer Society International Computer Software and Applications Conference (COMPSAC 2000)*, (pp. 122-127). Taipei, Taiwan.

Liu, Y., & Chen, X. (2004). A new information security model based on BLP Model and BiBA Model. *In Proceedings 7th International Conference on Signal Processing (ICSP'04)*, (pp. 2643-2646).

Mackinnon, S. J., Taylor, P. D., Meijer, H., & Akl, S. G. (1985). An optimal algorithm for assigning cryptographic keys to control access in a hierarchy. *IEEE Transactions on Computers*, *34*(9), 797–802. doi:10.1109/TC.1985.1676635

McLean, J. (1990). The specification and modeling of computer security. *IEEE Computer*, *23*(1), 9–16. doi:10.1109/2.48795

Meziane, H., & Benbernou, S. (2010). A dynamic privacy model for Web Services. *Computer Standards & Interfaces*, *32*, 288–304. doi:10.1016/j.csi.2010.02.001

Mie, X.-W., Feng, D.-G., Che, J.-J., & Wang, X.-P. (2006). Design and implementation of security operating system based on trusted computing. *International Conference on Machine Learning and Cybernetics*, (pp. 2776-2781).

Moreno, A., Sanchez, D., & Isern, D. (2003). Security measures in a medical multi-agent system. *Frontiers in Artificial Intelligence and Applications*, *100*, 244–255.

Nazir, A., Raza, S., & Chuah, C.-N. (2008). Unveiling Facebook: A measurement study of social network based applications. *IMC '08: Proceedings of the 8th ACM SIGCOMM conference on Internet measurement* (pp. 43-56). Vouliagmeni, Greece: ACM.

Osborn, S. (2002). Integrating role graphs: A tool for security integration. *Data & Knowledge Engineering*, *43*, 317–333. doi:10.1016/S0169-023X(02)00130-1

Pauley, W. A. (2010). Cloud provider transparency – An empirical evaluation. *IEEE Security and Privacy*, November-December 2010, 32-38.

Pfleeger, C. P., & Pfleeger, S. L. (2003). *Security in computing*. New Jersey: Pearson Education, Prentice Hall.

Ren, K., & Lou, W. (2007). Privacy-enchanced, attack-resilient access control in pervasive computing environments with optional context authentication capability. *Mobile Networks and Applications*, *12*, 79–92. doi:10.1007/s11036-006-0008-7

Rjaibi, W. (2004). A multi-purpose implementation of mandatory access control in relational database management systems. *In Proceedings 30th VLDB Conference,* (pp. 1010-1020). Toronto, Canada.

Ryutov, T., Zhou, L., Neuman, C., Leithead, T., & Seamons, K. E. (2005). Adaptive trust negotiation and access control. *2005 Symposium on Access Control Models and Technologies* (pp. 139-146). Stockkholm, Sweden: ACM.

Sandhu, R. (1988). Cryptographic implementation of a tree hierarchy for access control. *Information Processing Letters*, *27*, 95–98. doi:10.1016/0020-0190(88)90099-3

Sandhu, R. (1993). Lattice-based access control models. *IEEE Computer*, *26*(11), 9–19. doi:10.1109/2.241422

Shen, V., & Chen, T. (2002). A novel key management scheme based on discrete logarithms and polynomial interpolations. *Computers & Security*, *21*(2), 164–171. doi:10.1016/S0167-4048(02)00211-0

Slattery, B. (2010, March 31). *Facebook flub leaks private e-mail addresses*. Retrieved October 28, 2010, from http://www.pcworld.com/article/193009/ facebook_flub_leaks_private_email_addresses.html

Smari, W. W., Zhu, J., & Clemente, P. (2009). Trust and privacy in attribute based access control for collaboration environments. *International Conference on Information Integration and Web-based Applications & Services* (pp. 49-55). Kuala Lumpur, Malaysia: ACM.

Tanenbaum, A. S., & Steen, V. (2007). *Distributed systems: Principles and paradigms*. Upper Saddle River, NJ: Prentice Hall.

Tzeng, W.-G. (2002). A time-bound cryptographic key assignment scheme for access control in a hierarchy. *IEEE Transactions on Knowledge and Data Engineering*, *14*(1), 182–188. doi:10.1109/69.979981

Verma, M. (2004, October). *XML security: Control information access with XACML*. Retrieved from http://www.ibm.com/developerworks/xml/library/x-xacml/

Wang, S.-Y., & Laih, C.-S. (2006). Merging: An efficient solution for time-bound hierarchical key assignment scheme. *IEEE Transactions on Dependable and Secure Computing*, *3*(1), 91–100. doi:10.1109/TDSC.2006.15

Weiser, M. (1998). The future of ubiquitous computing on campus. *Communications of the ACM*, *41*(1), 41–42. doi:10.1145/268092.268108

Weiser, M. (1999). The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.*, *3*(1), 3–11. doi:10.1145/329124.329126

Wikipedia. (2010, October 27). *Facebook*. Retrieved October 28, 2010, from http://en.wikipedia.org/wiki/Facebook

Wikipedia. (2010, October 28). *MySpace*. Retrieved October 28, 2010, from http://en.wikipedia.org/wiki/MySpace

Wikipedia. (2010, November 20). *Domain model*. Retrieved November 25, 2010, from http://en.wikipedia.org/wiki/Domain_model

Yang, C., & Li, C. (2004). Access control in a hierarchy using one-way functions. *Elsevier: Computers and Security*, *23*, 659–664.

Yao, D., Frikken, K., Atallah, M., & Tamassia, R. (2008). Private information: To reveal or not to reveal. *ACM Transactions on Information and System Security*, *12*(1), 1–27. doi:10.1145/1410234.1410240

Yi, X. (2005). Security of Chien's efficient time-bound hierarchical key assignment scheme. *IEEE Transactions on Knowledge and Data Engineering*, *17*(9), 1298–1299. doi:10.1109/TKDE.2005.152

Yi, X., & Ye, Y. (2003). Security of Tzeng's time-bound key assignment scheme for access control in a hierarchy. *IEEE Transactions on Knowledge and Data Engineering*, *15*(4), 1054–1055. doi:10.1109/TKDE.2003.1209023

Yu, W., Sun, Y., & Liu, R. (2007). Optimizing the rekeying cost for contributory group key agreement schemes. *IEEE Transactions on Dependable and Secure Computing*, *4*(3), 228–242. doi:10.1109/TDSC.2007.1006

Zhang, G., & Parashar, M. (2004). Context-aware dynamic access control for pervasive application. *In Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation Conference* (pp. 21-30). Society for Modeling and Simulation International.

Zou, X., & Ramamurthy, B. (2004). A GCD attack resistant CRTHACS for secure group communications. *In Proceedings International Conference on Information Technology: Coding and Computing (ITCC'04)*, (pp. 153-154).

ADDITIONAL READING

Besmer, A., Watson, J., & Lipford, H. R. (2010). The impact of social navigation on privacy policy configuration. In *Proceedings of the Sixth Symposium on Usable Privacy and Security* (Redmond, Washington, July 14 - 16, 2010). SOUPS '10, vol. 485. ACM, New York, NY, 1-10. DOI= http://doi.acm.org/10.1145/1837110.1837120

Bishop, M. (2008). *Computer Security: Art and Science*. Addison-Wesley.

Carminati, B., Ferrari, E., & Perego, A. (2009, Oct.). Enforcing access control in Web-based social networks. *ACM Transactions on Information and System Security*, *13*(1), 1–38. doi:10.1145/1609956.1609962

Dandalis, A., & Prasanna, V. K. 2004. An adaptive cryptographic engine for internet protocol security architectures. *ACM Trans. Des. Autom. Electron. Syst.* 9, 3 (Jul. 2004), 333-353. http://doi.acm.org/10.1145/1013948.1013952

Fayssal, S., Alnashif, Y., Kim, B., & Hariri, S. 2008. A proactive wireless self-protection system. In *Proceedings of the 5th international Conference on Pervasive Services* (Sorrento, Italy, July 06 - 10, 2008). ICPS '08. ACM, New York, NY, 11-20. DOI= http://doi.acm.org/10.1145/1387269.1387272

Ferraiolo, D. F., Kuhn, D. R., & Chandramouli, R. (2003). *Role-Based Access Control, Computer Security Series*. Boston, London: Artech House.

Ganek, A., & Corbi, T. (2003). The Dawning of the Autonomic Computing Era. *IBM Systems Journal*, *42*(1), 5–18. doi:10.1147/sj.421.0005

Gollman, D. (2005). *Computer Security*. John Wiley and Sons, Ltd.

Hafner, M., & Breu, R. (2009). *Security Engineering for service Oriented Architectures*. Berlin, Heidelberg: Springer-Verlag.

Hart, E., Davoudani, D., & McEwan, C. 2007. Immunological inspiration for building a new generation of autonomic systems. In *Proceedings of the 1st international Conference on Autonomic Computing and Communication Systems* (Rome, Italy, October 28 - 30, 2007). Autonomics, vol. 302. ICST (Institute for Computer Sciences Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, 1-10.

Hellerstein, J. L. 2009. Engineering autonomic systems. In *Proceedings of the 6th international Conference on Autonomic Computing* (Barcelona, Spain, June 15 - 19, 2009). ICAC '09. ACM, New York, NY, 75-76. DOI= http://doi.acm.org/10.1145/1555228.1555254

Kayem, A., Akl, S., & Martin, P. (2010). *Adaptive Cryptographic Access Control. Advances in Information Security, 48*. Springer.

Kayem, A., Martin, P., & Akl, S. (2010). Enhancing Identity Trust In Cryptographic Key Management Systems for Dynamic Environments. *Security and Communication Networks*, http://onlinelibrary.wiley.com/doi/10.1002 /sec.164/abstract.

Kayem, A., Martin, P., Akl, S., & Powley, W. (2008). A framework for self-protecting cryptographic key management. *In Proceedings, 2nd IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, (pp. 191--200). Venice, Italy.

Kephart, J. O., & Chess, D. M. (2003). The Vision of Autonomic Computing. *IEEE Computer*, *36*(1), 41–50. doi:10.1109/MC.2003.1160055

Kephart, J. O., & Chess, D. M. (2005). Research Challenges of Autonomic Computing. *In Proceedings. 27th International Conference on Software Engineering*, (pp. 15-22). St. Louis, MO, USA.

Lee, J., Jeong, K., & Lee, H. 2010. Detecting metamorphic malwares using code graphs. In *Proceedings of the 2010 ACM Symposium on Applied Computing* (Sierre, Switzerland, March 22 - 26, 2010). SAC '10. ACM, New York, NY, 1970-1977. DOI= http://doi.acm.org/10.1145/1774088.1774505

Osborn, S. (2002). Integrating Role Graps: A tool for Security Integration. *Data & Knowledge Engineering*, *43*, 317–333. doi:10.1016/S0169-023X(02)00130-1

Pfleeger, C. P., & Pfleeger, S. L. (2003). *Security in Computing*. New Jersey: Pearson Education, Prentice Hall.

Rosen, K. (2003). *Discrete mathematics and Its Applications* (5th ed.). McGraw Hill.

Ryutov, T., Zhou, L., Neuman, C., Leithead, T., & Seamons, K. E. 2005. Adaptive trust negotiation and access control. In *Proceedings of the Tenth ACM Symposium on Access Control Models and Technologies* (Stockholm, Sweden, June 01 - 03, 2005). SACMAT '05. ACM, New York, NY, 139-146. DOI= http://doi.acm.org/10.1145/1063979.1064004

Squicciarini, A. C., Shehab, M., & Paci, F. 2009. Collective privacy management in social networks. In *Proceedings of the 18th international Conference on World Wide Web* (Madrid, Spain, April 20 - 24, 2009). WWW '09. ACM, New York, NY, 521-530. DOI= http://doi.acm.org/10.1145/1526709.1526780

Tanenbaum, A. S., & Steen, V. (2007). *Distributed Systems: Principles and Paradigms.* Upper Saddle River, NJ 07458: Prentice Hall.

Vincenzo Taddeo, A., Marcon, P., & Ferrante, A. 2009. Negotiation of security services: a multi-criteria decision approach. In *Proceedings of the 4th Workshop on Embedded Systems Security* (Grenoble, France, October 15 - 15, 2009). WESS '09. ACM, New York, NY, 1-9. DOI= http://doi.acm.org/10.1145/1631716.1631720

Watson, J., Whitney, M., & Lipford, H. R. 2009. Configuring audience-oriented privacy policies. In *Proceedings of the 2nd ACM Workshop on Assurable and Usable Security Configuration* (Chicago, Illinois, USA, November 09 - 09, 2009). SafeConfig '09. ACM, New York, NY, 71-78. DOI= http://doi.acm.org/10.1145/1655062.1655076

KEY TERMS AND DEFINITIONS

Access Control: Method of protecting data from access by unauthorized users usually done by classifying users into groups according to privilege of access.

Autonomic Management: This a method of system management whereby a system is designed and/or implemented to be self-managing and self-configuring to reduce the delays and risks of failure that are inherent in manually managed systems.

Cloud Computing: This is an Internet-based concept that is an extension of the service oriented architecture concept whereby resources like memory, software and information are made available to users on demand.

Cryptographic Key Management: Access control model based on the assignment of cryptographic keys that users can use to encrypt or decrypt data.

Data Privacy Protection: This is the concept of designing access control mechanisms to ensure that the mechanisms protect data from being accessed by unauthorized users and also from being used in ways that violate the rules of access to the data.

Fault Tolerance: Ability of a system to cope with failure usually with some form of backup solution that ensures that the system is available to perform the tasks required of it on-demand.

Keywords: Data Privacy Protection, Access Control, Cryptographic Key Management, Fault Tolerance, Autonomic Management, Self-Protection, Cloud Computing.

Self-Protection: This is the ability of any system, and in particular an access control mechanism, to protect itself from failure by using autonomic management to reduce human error and to provide efficient management.

ENDNOTES

[1] The autonomic computing paradigm aims at creating systems that are self managing and self-configuring to reduce human error and make systems more efficient time wise.