USAGE CONTROL: A UNIFIED FRAMEWORK FOR NEXT GENERATION ACCESS CONTROL

by

Jaehong Park

A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
the Requirements for the Degree
of
Doctor of Philosophy
Information Technology

Committee:

_____ Dr. Ravi Sandhu, Dissertation Director

_____ Dr. Larry Kerschberg

_____ Dr. Edgar Sibley

_____ Dr. Daniel Menasce

_____ Dr. Stephen G. Nash, Associate Dean for Graduate Studies and Research

_____ Dr. Lloyd J. Griffiths, Dean, School of Information Technology and Engineering

Date: _____ Summer 2003
George Mason University
Fairfax, Virginia

Usage Control: A Unified Framework for Next Generation Access Control

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University.

By

Jaehong Park

B.B.A., Dongguk University, Seoul, Korea, February 1995
M.S., The George Washington University, Washington D.C., December 1998

Director: Dr. Ravi S. Sandhu, Professor
Department of Information and Software Engineering

Summer 2003
George Mason University
Fairfax, Virginia

DEDICATION

To my late father, Soontae Park, who has inspired and encouraged me to continue my lifelong dream but couldn't see me accomplishing it.

To my mother, Jongnam Choi, who has made many sacrifices for my academic pursuit and waited for the success of the pursuit.

My special dedication goes to my lovely wife, Hoonim Kim, who has been always with me throughout this academic journey, shared all the difficult times, and supported me with great loves and sacrifices, and to my lovely daughters, Tricia and Alyson, who have been the source of joy of my life.

I also dedicate to my hometown families, including sisters, parent-in-laws, brothers-in-law, and many others who are not mentioned here.

ACKNOWLEDGMENTS

I would like to express my sincere appreciation and gratitude to my dissertation director, Professor Ravi Sandhu who has enlightened and guided me throughout my doctoral studies. It was his invaluable advice that made me accomplish this achievement. Nothing can be said enough to express my indebtedness to him.

I am also grateful to my dissertation committee members, Professor Larry Kerschberg, Professor Edgar Sibley, and Professor Daniel Menasce for their valuable comments and suggestions.

I am especially thankful to my wife and families who have provided me a strength to come all the way to this achievement.

My appreciation also goes to many friends in George Mason University for their help.

TABLE OF CONTENTS

Page

LIST OF TABLES **viii**

LIST OF FIGURES **ix**

ABSTRACT **xi**

Chapter 1. INTRODUCTION **1**
1.1 Problem Statement 2
1.2 Summary of Contributions 5
1.3 Organization of the dissertation 6

Chapter 2. RELATED WORK AND APPROACHES **8**
2.1 Traditional Access Control Perspectives 8
2.2 Modern Access Control and Digital Rights Management Perspectives . . 10
2.3 Usage Control Approach 12
2.4 OM-AM Layered Approach 13

Chapter 3. USAGE CONTROL (UCON) **15**
3.1 Usage Control Scope 15
3.2 Usage Control Objectives and Coverage 17

Chapter 4. UCON ABC MODELS **22**
4.1 ABC Model Components 23
4.1.1 Subjects (S) and Subject Attributes (ATT(S)) 24
4.1.2 Objects (O) and Object Attributes (ATT(O)) 26
4.1.3 Rights (R) 27
4.1.4 Authorizations ($\mathcal{A}$) 28
4.1.5 oBligations ($\mathcal{B}$) 29
4.1.6 Conditions ($\mathcal{C}$) 30
4.2 The ABC Family of Core Models 32

4.2.1 $UCON_{preA}$ - pre-Authorizations Models 34
4.2.2 $UCON_{onA}$ - ongoing-Authorizations Models 40
4.2.3 $UCON_{preB}$ - pre-oBligations Models 46
4.2.4 $UCON_{onB}$ - ongoing-oBligations Models 51
4.2.5 $UCON_{preC}$ - pre-Conditions Model 54
4.2.6 $UCON_{onC}$ - ongoing-Conditions Model 56
4.2.7 Global Obligations 58
4.3 Applications in ABC Model 59
4.3.1 Mandatory Access Control 60
4.3.2 Role-based Access Control 61
4.3.3 Discretionary Access Control 64
4.3.4 Trust Management 66
4.3.5 Digital Rights Management 67
4.3.6 Modern Access Control (Healthcare Examples) 70
4.4 Related Work 76
4.5 Summary 78

Chapter 5. UCON ARCHITECTURES 80
5.1 Payment-Free vs. Payment-Based 81
5.2 Reference Monitor 83
5.2.1 Structure of Reference Monitor 83
5.2.2 Architectural Classification 85
5.2.3 Architectural Scope 88
5.3 Three factors of CRM security architectures 88
5.4 Architecture Taxonomy 89
5.4.1 Non-Encapsulated Architectures 91
5.4.2 Encapsulated Architectures 92
5.5 Related Mechanisms 99
5.5.1 Watermarking Mechanisms 99
5.5.2 Use-Control Technologies 100
5.6 Discussion 102
5.6.1 Solution Approaches 102
5.6.2 Characteristics of Security Architectures 104
5.6.3 Available COTS Solutions for the Architectures 104
5.7 Summary 106

Chapter 6. UCON APPLICATIONS **108**
6.1 UCON Management and Examples 108
6.1.1 Administrative UCON 109
6.1.2 DRM and Healthcare Applications in UCON Administration . . . 111
6.1.3 Reverse UCON 114
6.2 Originator Control in UCON 117
6.2.1 Originator Control (ORCON) 118
6.2.2 ORCON in UCON 119
6.2.3 Variations of ORCON in UCON 124
6.2.4 Discussion and Related Work 128
6.2.5 Summary for ORCON in UCON 131

Chapter 7. CONCLUSION **132**
7.1 Contributions 132
7.2 Future Research 134

BIBLIOGRAPHY **136**

LIST OF TABLES

Table **Page**

4.1 The 16 Basic ABC Models 33

5.1 Characteristics of Architectures 105
5.2 Architecture of COTS Solutions 106

LIST OF FIGURES

Figure **Page**

1.1 Traditional Access Control 3
1.2 Continuity and Mutability Properties 4

2.1 OM-AM Layered Approach 14

3.1 UCON Scope in Information Security 16
3.2 UCON Coverage 19

4.1 ABC Model Components 24
4.2 The ABC Family of Core models 35

5.1 Conceptual Structure for UCON Reference Monitor 84
5.2 Architectural Scope 88
5.3 Security Architecture Taxonomy 90
5.4 No Control Architecture with Message Push (NC1) 91
5.5 No Control Architecture with External Repository (NC2) 92
5.6 Fixed Control Architecture with Message Push (FC1) 93
5.7 Fixed Control Architecture with External Repository (FC2) 94
5.8 Embedded Control Architecture with Message Push (EC1) 95
5.9 Embedded Control Architecture with External Repository (EC2) . . . 96
5.10 External Control Architecture with Message Push (XC1) 97
5.11 External Control Architecture with External Repository (XC2) 99
5.12 Security Attacks and Protections 103

6.1 Administrative UCON Triangle 109
6.2 Two Sides of UCON Model for Privacy Non-sensitive Objects 113
6.3 Three Sides of UCON Model for Privacy Sensitive Objects 114
6.4 An Example of Reverse UCON 115
6.5 Traditional ORCON Solutions 120
6.6 An Example of ORCON with UCON 123

6.7 Legend for ORCON Variation 125
6.8 Re-dissemination without Ticket 126
6.9 Re-dissemination with LGT 127
6.10 Re-dissemination with LRT 128
6.11 Re-dissemination with LGT and LRT 129

ABSTRACT

USAGE CONTROL: A UNIFIED FRAMEWORK FOR NEXT GENERATION ACCESS CONTROL

Jaehong Park, Ph.D.

George Mason University, 2003

Dissertation Director: Dr. Ravi S. Sandhu

In this dissertation I develop the concept of Usage Control (UCON) that encompasses traditional access control, trust management, and digital rights management and goes beyond them in its definition and scope. While usage control concepts have been mentioned off and on in the security literature for some time, there has been no clear definition nor systematic treatment. By unifying these diverse disciplines in a single framework, UCON offers a promising approach for the next generation of access control.

Traditional access control has focused on a closed system where all users are known and primarily utilizes a server-side reference monitor within the system. Trust management has been introduced to cover authorization for strangers in an open environment such as the Internet. Digital rights management has mainly dealt with client-side control of digital information usage focusing on intellectual property rights protection. Each of these areas is motivated by its own specific target problem. Innovations in information technology and business models are creating new security and privacy issues which require elements of all three areas. To deal with these in a systematic unified manner I propose the new concept of usage control or UCON.

By including obligations, conditions, ongoing controls, and mutability as well as authorizations, and by relaxing closed system limits, usage control lays the foundation for the next generation access control that is required for today's highly distributed and network-connected digital environment. UCON enables finer-grained control over usage of digital resources than that of traditional access control policies and models; for example, print once as opposed to unlimited printing. Unlike traditional access control or trust management, it covers both a centrally controllable environment and one where central control authority is not available.

In this dissertation, I develop a unified framework for usage control mainly focusing on models and architectures. A family of ABC models is developed as a core model for usage control. ABC models integrate obligations, conditions as well as authorizations for usage decision making. They also cover continuity and mutability issues which have not been discussed clearly nor comprehensively in previous studies. By covering continuity and mutability properties, ABC provides finer and richer control capabilities. I further develop security architectures for usage control that utilize a client-side reference monitor (CRM) which is an important concept that enables client-side usage control. As part of usage control applications, UCON management and originator control policies in UCON are discussed based on UCON models and architectures. I also present some potential approaches that can provide valuable directions for future extensions.

Chapter 1

INTRODUCTION

Technological innovations in computers and networks have enabled pervasive availability and usability of digital information bringing us opportunities for new business models and personal life styles. Because of these innovations, digital information no longer stays in computer systems only. It is now available on many other devices such as mobile devices (PDA, cell phone, MP3 player, etc), and Internet-integrated home appliances (refrigerator, microwave machine, etc) utilizing various communication methods such as CDs, DVDs, memory cards, mobile messaging, LANs, global Internet networks (either wired or wireless), etc. This pervasive computing phenomenon has raised several new challenging issues for reliable and trusted controls on the usage of digital resources throughout their life cycle.

Traditional access control disciplines such as Mandatory Access Control (MAC), Discretionary Access Control (DAC) and Role-Based Access Control (RBAC) have difficulty in covering today's digital environment. One reason is that, since traditional access control has focused on controlling access to digital resources within closed system environments, it fails to protect digital information persistently even after the information has been accessed or disseminated to other systems. In addition, traditional access control only deals with previously known user's access which is not adequate in today's Internet world.

Trust management has relaxed this known user restriction and allowed controls on

strangers' access to digital objects based on their credentials. However, trust management has failed to control usages on already disseminated digital objects. Both traditional access control and trust management have mainly focused on sensitive information protection while ignoring modern B2C systems. Modern access control and digital rights management solutions have been studied extensively to overcome these shortcomings and to generate new business models that can cover intellectual property rights protection and even privacy issues. However, most of these studies are motivated from specific target problems and lack comprehensive and systematic treatment of fundamental issues.

This chapter discusses these issues arising in today's highly distributed computing environment and presents overall contributions of this dissertation.

1.1 Problem Statement

One commonality of traditional access controls and even trust management is its utilization of subjects' attributes and objects' attributes for authorization process. In other words, in traditional access control, authorization decision is made based on subject attributes, object attributes, and requested rights. Attributes include identities, capabilities, or properties of subjects or objects. For example, in mandatory access control, clearance labels of subjects are considered as subject attributes and objects' classification labels as object attributes. Authorization process, then, evaluates the dominance of these labels along with requested access rights (e.g., read, write) to return either 'allowed' or 'not-allowed'. Similarly, in discretionary access control, access control list (ACL) can be viewed as object attributes and capability list as subject attributes. Figure 1.1 shows this 'attribute-based' traditional access control.

Although this 'attributed-based' approach of traditional access control can cover many

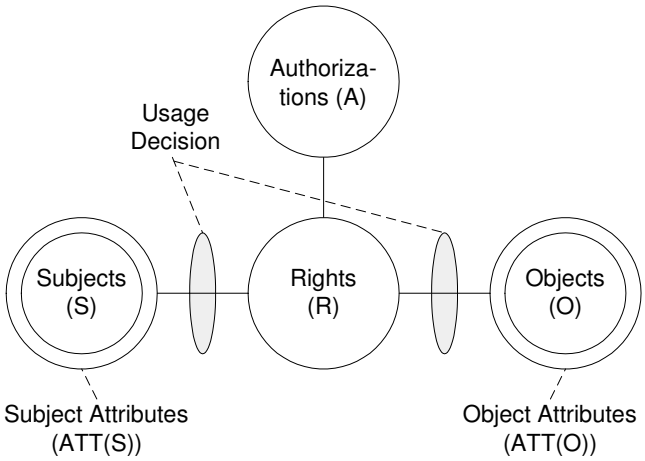


Figure 1.1: Traditional Access Control

applications, today's digital information systems require more than classical authorizations. For example, suppose Alice has to click 'yes' button for license agreement or has to fill out a certain form to download a company's whitepaper. In this case, subject's certain actions have to be performed for the allowance of a requested usage. In other words, usage decision is made based on fulfillment of required actions, not by existence of subject attributes and object attributes. This decision factor is called as "obligation" and required in addition to authorization to cover modern access control applications.

In addition to authorization and obligation, there are certain situations where access or usage needs to be limited in certain environmental or system status. For example, usages on certain digital resources can be allowed only during business hours or at certain locations. A system with heavy traffic loads may allow only premium users to be accessed. For this requirement, a system need to check current environmental or system status for usage decision. This decision factor is called as "condition" and required together with authorization and obligation for modern access control.

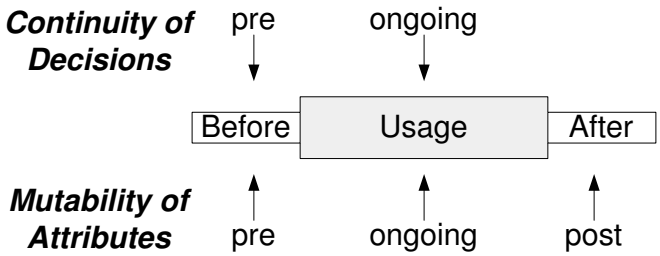


Figure 1.2: Continuity and Mutability Properties

In addition to these three decision factors, modern information system requires two other important properties called continuity and mutability as shown in Figure 1.2. In traditional access control, authorization is assumed to be done before access is allowed (pre). However, it is quite reasonable to extend this for continuous enforcement by evaluating usage requirements throughout usages (ongoing). This property is called "continuity" and has to be captured in modern access control for the control of relatively long-lived usage or for immediate revocation of usage.

In traditional access control, attributes are modifiable only by administrative actions. However, in many of modern applications such as DRM systems, these attributes have to be updated as side-effects of subjects' actions. For example, a subject's e-cash balance has to be decreased by the value of a digital object as the subject uses or accesses the object. This "mutability" property of attributes has been rarely discussed in traditional access control literature. In case attributes are mutable, updates can be done either before (pre), during (ongoing) or after (post) usages as shown in Figure 1.2. Having a mutability property allows better enforcement on various classical policies that require history-based authorizations such as dynamic Separation Of Duty or Chinese Wall policy.

Although some of these issues have been discussed in access control literature, because their focuses are limited on specific target problems, they lack comprehensiveness in

their discussion. In this dissertation, the notion of usage control (UCON) is developed to cover these diverse issues in a single framework to overcome these shortcomings. In UCON, traditional access control can be extended to include modern access control and digital rights management by integrating obligations and conditions as well as authorizations and by including continuity and mutability properties.

1.2 Summary of Contributions

The following list summarizes the contributions this research achieves.

- To develop a unified framework called usage control
  - that encompasses traditional access control, trust management and digital rights management for modern information and systems protection,
  - that enables controls on usage of digital information for confidential information protection, intellectual property rights protection, and privacy protection in a systematic manner,
  - that enables controlling usage of digital information regardless of system (computer or network) environments, and
  - that enables controlling usage of digital information even after digital information is disseminated.
- To develop a family of core models called ABC for usage control in a formal and systematic way
  - that integrates obligations and conditions as well as authorizations for finer and richer controls, and
  - that provides ongoing controls and mutability for modern information and systems protection.

- To develop comprehensive dissemination architectures for usage control
  - that utilize client-side reference monitor, and
  - that can support both controlling and tracking of digital information dissemination.
- To demonstrate usage control applications
  - that are based on UCON models and architectures, and
  - that are applicable to real world examples.

1.3 Organization of the dissertation

Chapter 2 reviews related work and its perspectives and discusses general approaches of this research. Chapter 3 identifies the scope of usage control and its relationships among other disciplines. Then, chapter 4 introduces a family model of ABC (Authorizations, oBligations, and Conditions) core models for usage control. ABC models are essence of usage control and do not cover other important issues such as administration and delegation issues hence we call these as core models. Examples for MAC, DAC, RBAC, trust management and digital rights management are presented in terms of ABC models to show how they can encompass these diverse disciplines. Chapter 5 discusses UCON security architectures that utilize only a client-side reference monitor. Traditionally server-side reference monitor has been assumed for access control architecture. Client-side reference monitor is an essential factor for modern distributed system environment. Chapter 5 provides a comprehensive architectural description in this arena. Here, I also survey COTS (Commercial Off The Shelf) solutions to map them to UCON architectures for commercial availability. This provides an opportunity to review solution approaches currently available in the market and to

identify any potential and uncovered solution approach. Chapter 6 further discusses two applications of usage control. First, management or administration issues of usage control are discussed. Also issues of re-dissemination controls on disseminated digital objects are discussed by using originator control policy. Finally the contributions of this dissertation are summarized and some future research directions are presented in chapter 7.

Chapter 2

RELATED WORK AND APPROACHES

The issue of usage control on digital resources can be approached from several viewpoints. This chapter examines each of these approaches and their characteristics by reviewing prior work related to usage control to motivate the approach of this dissertation. First I explore these issues from traditional access control viewpoint, then from modern access control and digital rights management point of view. Following this I discuss the approach of this dissertation to usage control.

2.1 Traditional Access Control Perspectives

In computer and information security history, there have been many attempts to achieve trusted controls on digital resource usage. The earliest approach was traditional access controls such as mandatory access control (MAC), discretionary access control (DAC), and role-based access control (RBAC). Access control remains a major challenge for computer and information security in modern cyberspace. Providers of services, resources and digital content need to determine selectively who can access these and exactly what access is provided. This is the central objective of access control.

Over the past thirty plus years there has been much progress in access control, but at its core the academic perspective has largely remained unchanged and centered around the access matrix model [Lam71, Lan97]. The essential concept of the access

matrix is that a right is explicitly granted to a subject to access an object in a specific mode, such as read or write. This right exists whether or not the subject is currently accessing the object. Moreover, the presumption is that the right enables repeated access until it is explicitly revoked. In practice the access matrix is never explicitly represented. Instead access control lists (ACLs), capabilities or access relations are used. Groups and roles, possibly in partially ordered seniority relationships, are used to further simplify the actual representation of rights. A variety of discretionary, mandatory and role-based access control models have emerged to accommodate a diverse range of real-world access control policies. In a sense the practice of access control has grown further and further away from the access matrix abstraction. But the core idea that access is driven by rights granted to a subject to access an object remains. On the theoretical side the seminal work of [HRU76] established Turing completeness of the access matrix. While this is a negative result for purposes of safety analysis, it formally establishes the open-ended expressive power of the access matrix.

In recent years several researchers have proposed extensions to the basic access matrix notion of subjects, rights and objects. These have typically come from specific perspectives and the extensions have tended to emphasize the particular system or application focus of the authors. In distributed system the notion of a principal's identity and the meaning of an access control list entry granting particular access to a principal was no longer as simple as in earlier timesharing systems [ABL93]. Traditionally, access control has focused on the protection of computer and information resources in a closed system environment. The enforcement of control has been primarily based on identities and attributes of known users by using a reference monitor and specified authorization rules [SS94]. In today's network-connected, highly dynamic and distributed computing environments, digital information is likely to be

used and stored at various locations, hence has to be protected regardless of user location and information location. Relaxing closed system requirement introduces the need to control access by previously unknown users. B2C mass distributions such as e-book systems or music file distributions are also examples of stranger's usages.

2.2 Modern Access Control and Digital Rights Management Perspectives

With the advent of public-key infrastructure recent research in authorizations for strangers' usages have been pursued under the name of trust management [BFL96, HMM+00, WSJ00, Wee01]. In many cases, trust management utilizes a user's capabilities or properties for authorization in the form of digital credentials or certificates. However, both traditional access controls and trust management have focused on protecting digital resources within server systems and do not deal with client-side controls for locally stored digital information. More recently by utilizing some forms of *client-side reference monitor*, and by focusing on controlling usage of already disseminated digital objects, the arena of Digital Rights Management (DRM) has brought out a significant new perspective on access control problems [SBW95, Kap96, RTM02, WLD+02]. The DRM requirement for persistent access control [Sch99] and the difficulty of achieving this on a mass-scale is a significant complication. To enable trusted client-side computing there have been industry initiatives such as Microsoft's Palladium and Intel-driven Trusted Computing Platform Alliance (TCPA) [tcp02] which were partly originated from AEGIS [Arb97]. Palladium and TCPA have gained serious attentions and concerns because of their potential impacts on security and privacy issues as well as DRM [And02]. DRM is likely to utilize this kind of trusted computing base as a critical enabling technology.

DRM technologies have emerged in mid 90's and gained notable public attention re-

cently. In Jan/Feb. 2001 issue of MIT Technology Review, DRM has been recognized as one of the top 10 emerging technologies that will change the world [mit01]. Because of DRM's potential opportunity for commercial sector, current DRM solutions have been largely driven by commercial entities and are mainly focused on intellectual property rights protection which is based on payment functions. For example, DRM has hardly recognized commercial B2B transactions in their solutions though its underlying technologies can be used for controls on this kind of sensitive information usage. For last several years, many companies have developed various technical solutions for DRM implementations. Many underlying technologies such as watermarking technologies, use-control technologies (including client-side software, server-side encapsulation software, etc.) have been studied. Some rights expression languages (e.g., XrML, ODRL, etc.) also have been developed [Con02, Ian02]. While these DRM techniques and mechanisms have dominated recent DRM studies, many researchers now believe that there is a fundamental unity between DRM and access control. That is DRM is not just a collections of enabling technologies but also about business and security-related policies and models for usage decisions and controls [LaM02]. Studies on well-defined, comprehensive models and policies for access control and DRM will provide a foundation for more trusted and secure computing environment.

On a different front several authors have realized that classic access control is inadequate for modern applications. The notion of provisional authorization [KH00, JKS01] states that authorization is not complete until the subject carries out some action to make the authorization effective. The notion of task-based authorization [TS97] treats all rights as consumable and brought into being just-in-time. In this view a right is a one-time (or k-time) permission obtained in context of enterprise activity. Exercise of a consumable right can enable other rights for different subjects and objects. Access control policy can be seen as one policy amongst many that need to be managed and

enforced in distributed system. The Ponder system [DDLS01] is a state-of-the-art example of work on policy languages that include but also transcend access control issues. A policy framework directly oriented towards access control but with a number of very useful extensions such as conditions and side-effects has been recently published [RN01, RN02]. These authors also seek to develop an API and extended ACL based implementation of a portion of their model. From an application perspective the healthcare domain continues to provide significant challenge for traditional access control because of the complexity of its policies and the number of different parties with different interests [BBB97, hhs02].

2.3 Usage Control Approach

The research cited above encompasses many significant achievements in specific target problems. At the same time the specific focus has resulted in lack of comprehensiveness and systematic treatment of fundamental issues. Studies on access controls, trust management, and DRM have followed their own tracks and have rarely influenced each other. Although traditional access controls have shown limitations to cover modern digital environments, there has been noteworthy work on security policies and models for controlling digital resources. Similarly though DRM has opened up closed system restrictions, the discipline still lacks well-defined policies and models. Also, current rights expression languages cannot express transaction-level controls including mutability and continuity aspects. Today's DRM technology requires well-defined policies and models that can express usage decisions more comprehensively and can cover sensitive information protection as well as intellectual property rights protection. Access control and trust management require enlargement of their scope to enable richer, finer and persistent controls on digital objects regardless of their locations. Furthermore, none of these approaches adequately address privacy issues.

Usage Control is a conceptual framework that covers these areas in a systematic manner to provide a general-purpose, unified framework for protecting digital resources. UCON is not a substitute for traditional access control, trust management, or digital rights management. Rather, UCON encompasses these three areas and goes beyond in its definition and scope. Also, UCON achieves fine-grained control on digital resources even after the objects have been disseminated.

2.4 OM-AM Layered Approach

Throughout the dissertation, UCON is discussed based on layered approach. Layered approaches have been used for a long time in computer science. This dissertation will follow the OM-AM framework [San00]. OM-AM framework has been used in role-based access control to describe its models, architectures, etc. OM-AM stands for Objective, Model, Architecture, and Mechanism, respectively (see Figure 2.1). Objective and Model layers articulate what the security objectives are and what should be achieved, while Architecture and Mechanism describe how to achieve these objectives and requirements. The OM-AM framework is analogous to OSI 7 layer network protocol stack. Like OSI 7 layers, each OM-AM framework layer's mapping to adjacent layers is many-to-many. In other words, a model can be supported by multiple architectures, while an architecture can support multiple models. Also, the OM-AM framework is neither a top-down nor a waterfall-style software engineering process. Each layer deals with distinct and independent functions, and at the same time these functions are tightly related to other layers in some degree. The functions in each layer require different notions and abstractions to articulate their distinct concerns. My research is performed in context of this OM-AM layered approach.

Based on this OM-AM four-layered framework, the engineering approaches and scopes of UCON can be divided into different layers and separately addressed. This does

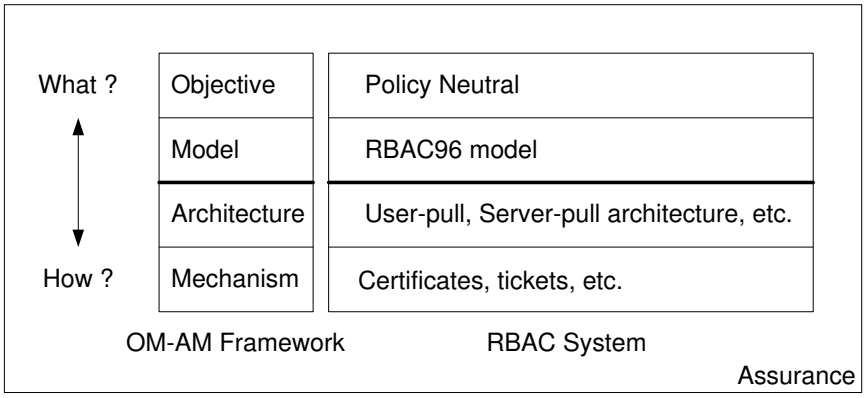


Figure 2.1: OM-AM Layered Approach

not mean that each layer is unrelated to others. Rather, they are tightly related, and instances of each layer have many-to-many relations with instances in other layers. By dividing engineering focus into several layers, the dissertation can scrutinize the unique requirements and elements of each layer. Later, the aggregated framework of each layer can be referenced and each layer's instances can be implemented as needed.

Chapter 3

USAGE CONTROL (UCON)

In previous chapter 2, I has reviewed some related work and their perspectives on usage control. This chapter discusses the main objectives of usage control and its scope in information security.

3.1 Usage Control Scope

Researchers have studied areas of information security with various approaches. They have had their own specific objectives and different targets to secure. Figure 3.1 shows the scope of usage control in information security discipline. First criteria can be objectives. The objectives of information security research can be divided into prevention, detection, and response. Traditional access control (TAC), trust management (TM), and digital rights management (DRM) solutions mainly focus on prevention.[1] Usage control also mainly focus on prevention though detection and response mechanisms (e.g., watermarking technologies) can be also utilized within UCON framework.

Another criteria can be target resources. There can be three target resources: information resources, computer systems resources and network resources. In traditional access control, computer systems resources and information resources have been re-

[1]Some DRM solutions may include detection or response mechanisms. However these are not their primary goals to achieve.

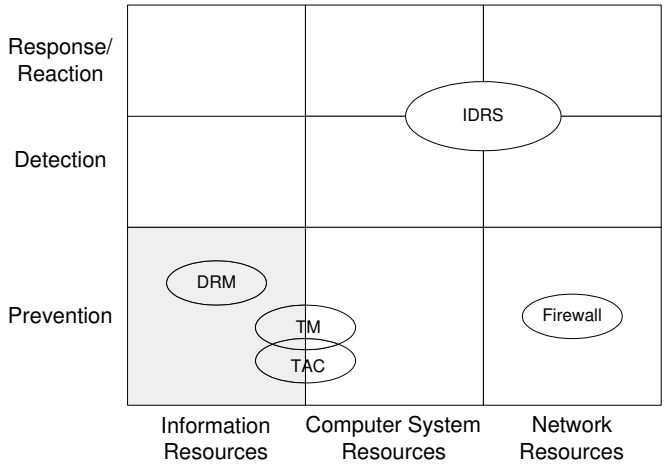


Figure 3.1: UCON Scope in Information Security

garded as target objects and only closed system environment is considered. Because of this, traditional access control has had difficulty dealing with controls on usage of network resources. This is probably one of the reason that access control has been largely ignored from network security studies (e.g., intrusion detection and response system (IDRS), firewalls, etc.). Likewise, because of the same reason, there have been difficulties in controlling information resources in distributed and heterogeneous environments. In distributed and networked environments, one of major shortcomings of traditional access control is there is no control available on already disseminated digital information.

In usage control only digital information resource is considered as a target object. This will enable continuous protection on digital information regardless of its system environments, access methods or locations. Furthermore, unlike other resources, information resources involve unique problems such as privacy issues. Healthcare information, genetic information, usage log information, and payment information are some examples of privacy related information. Because of this, treating only informa-

tion resources as target object provides a better opportunity to resolve the security and privacy issues on digital information resources. However, this does not mean computer and network resources cannot be controlled. Usage of computer systems and network resources can be also controlled while controlling information resources by utilizing them within usage decision rules. Therefore, in this research, I separate information resources from other two target resources and consider only information resources as a target object. This separation enables control of digital resources persistently regardless whether the information is in host systems or distributed network systems. In Figure 3.1, gray area indicates the scope of UCON.

3.2 Usage Control Objectives and Coverage

The goal of usage control is to provide a new intellectual foundation for access control. As discussed earlier, the thirty year old framework of the access matrix has been extended in various different directions as researchers have found it to be inadequate for their needs. The net result is a plethora of seemingly ad hoc extensions without underlying intellectual unity. In this dissertation I propose a fresh look at the fundamental nature of access control itself. Hence, the term **usage control** to convey the broader perspective I am taking. The concept of usage control is comprehensive enough to encompass traditional access control, trust management, and digital rights management. Usage control unifies these areas systematically in a single framework and goes beyond in its scope. Figure 3.2 shows UCON's coverage and its relationships to other research areas. [2] [3]

[2] In architectural point of view, Java Virtual Machine can be considered as Reference Monitor. However, the main purpose of Java Virtual Machine is to protect a user's system from malicious code while UCON's Reference Monitor focus on protecting digital information from unauthorized usages of users. Although it's functional aspects may be similar to UCON reference monitor, since it's purpose is quite opposite to UCON, I exclude Java Virtual Machine from the scope of .

[3] Please note that Figure 3.2 covers only the gray area of 3.1.

In terms of objectives, sensitive information protection has been one of the most important goals of traditional access control. Recent studies on controlling usages of digital resources have focused on other goals as well, such as Intellectual Property Rights (IPR) protection, and privacy protection. Controlling usage of sensitive information requires protection of digital information that may be critical to nations or organizations. Intelligence community and B2B transaction are good examples for this purpose. IPR protection or digital copyrights protection is relatively a new goal. Content providers' interest largely belongs here so they can realize maximum revenue. Privacy has been rarely studied in the context of controlling usage of digital information but is beginning to get more public attention. W3C's recent P3P project is one example for privacy support in Web services [p3p02]. Healthcare information system is another good example that should consider privacy as a major concern. UCON is objective-neutral and covers all these purposes in a systematic way.

In architectural perspective, there can be three possibilities based on the location of reference monitor. Reference monitor is the most crucial component of UCON architecture that facilitates control decisions and enforcements. Traditional access controls have focused on server-side controls only, with little consideration of *client-side controls*, which give an ability to control usages persistently even after the digital resources are distributed. Usage control can utilize both server-side and client-side control architectures though some functional details are likely to be different. Client-side control requires existence of client-side trusted computing base and reference monitor. The detail of reference monitor and its architectural description are presented in Chapter 5. Issues on trustworthiness of client-side reference monitor are largely dependent on requirements of business models and are not discussed in this dissertation.

The term usage control has a couple of connotations. In the DRM context it conveys

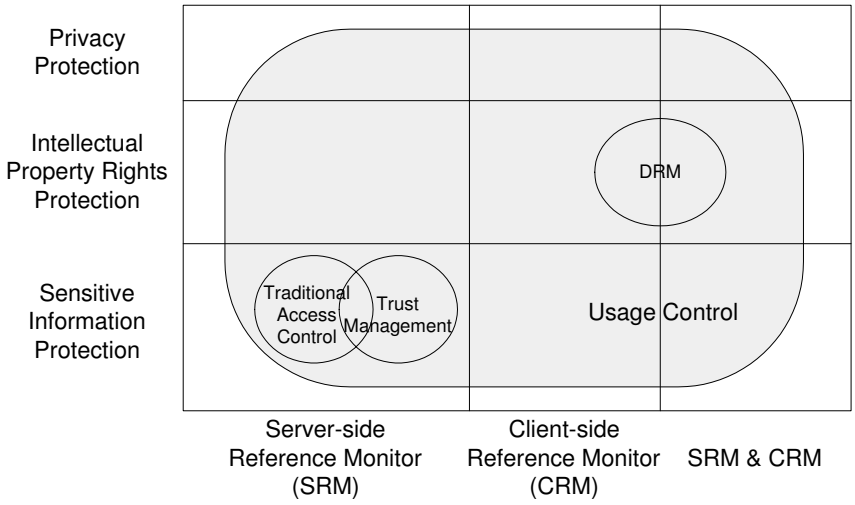


Figure 3.2: UCON Coverage

the sense that digital content is provided for use on the end-user's system, but the provider would like to retain some control over what the user does with the bits. In the privacy context the situation is reversed. It is the end-user who often provides personal information to a service provider, and would like to control how the service provider can use that information. Sometimes the personal information is provided by a third-party originator, say a health-care provider, but the individual, called 'identifiee', to whom it pertains would nevertheless like to exercise some control over its use. Usage also has a connotation of duration, so the access may continue for some time. In classic access control the usual viewpoint is that access is enforced before access is granted and then access persists for some duration without any further checks. This is appropriate for traditional access control systems but does not reflect many modern e-business cases.

In usage control, target objects have relationships with consumer, provider, and identifiee subjects. The consumer subject seeks access to a target object provided by a provider subject. The target object itself may contain privacy-related information of

subjects. These subjects are called identifiee subjects and hold certain rights on the object. Usage decision is based on relationships among these different subject parties on target resources. Ideally, these relationships may no longer be one-way control decisions which is the usual case today where provider determines consumer's access. Having multi-way control requires active involvement of each of these three parties in decision-making process. While this may be an ideal approach of usage control, this dissertation is a first step in this arena and only covers core aspects of usage control without considering relationships among different parties or multi-way controls. The core part of usage control deals with decision-making aspects of consumer subject usages. In next chapter the ABC models are introduced as a core model of usage control. The ABC model mainly discusses basic control issues of consumer subjects' usage on target objects and does not cover any issues of the relationships among different subject parties nor related administrative issues. However, in Chapter 6, some basic aspects of these relationships are discussed as part of administrative UCON.

Traditionally, access control has dealt with authorizations as the basis for its decision-making process. In the ABC model, the authorization-based decision process utilizes subject attributes and object attributes. Attributes can be identities, security labels, properties, capabilities, etc. The ABC model includes obligations and conditions as well as authorizations as part of usage decision process to provide a richer and finer decision capability. The necessity of obligations and conditions has been recognized in modern business systems such as B2C mass distribution systems as well as B2B transactions and interactions between business partners. Obligations are requirements that have to be fulfilled for usage allowance. Conditions are environmental requirements that are independent from individual subjects and objects. These decision predicates can be evaluated before or during exercise of a request. In addition, usage of target object may require certain updates on subject or object attributes before, during or

after a usage exercise (e.g., reducing a requester's account balance by the value of an e-book). UCON covers these issues within its ABC core model in a systematic manner.

Chapter 4

UCON ABC MODELS

In this chapter, I introduce the family of ABC (*Authorizations, oBligations, and Conditions*) models for *usage control (UCON)*. I call these core models because they address the essence of usage control, leaving administration, delegation and other important but second-order issues for later work. The term usage control is a generalization of access control to cover obligations, conditions, continuity (ongoing controls) and mutability. Traditionally, access control has dealt only with authorization decisions on users' access to target resources. Obligations are requirements that have to be fulfilled by obligation subjects for allowing access. Conditions are subject and object-independent environmental requirements that have to be satisfied for access. In today's highly dynamic, distributed environment, obligations and conditions are also crucial *decision factors* for richer and finer controls on usage of digital resources. Although they have been discussed occasionally in recent literature, most authors have been motivated from specific target problems and thereby limited in their approaches. The ABC model integrates these diverse concepts in a unified framework. Traditional authorization decisions are generally made at the time of requests but hardly recognize *ongoing controls* for relatively long-lived access or for immediate revocation. Moreover, *mutability* issues that deal with updates on related subject or object attributes as a consequence of access have not been systematically studied.

Unlike other studies that have targeted on specific problems or issues, the ABC model

seeks to enrich and refine the access control discipline in its definition and scope. The ABC model covers traditional access controls such as mandatory, discretionary and role-based access control. Digital rights management and other modern access controls are also covered within the model. I believe the ABC core model for UCON lays the foundation for next generation access controls that are required for real world information and systems security. This chapter articulates the core of UCON and develops several detailed models.

I first identify core components of the ABC model and develop a family of detailed models. Then, I show how traditional access control, trust management, and digital rights management can be achieved in the ABC model. I further discuss some related work and summarize this chapter.

4.1 ABC Model Components

The ABC models consist of eight core components (see Figure 4.1). They are subjects, subject attributes, objects, object attributes, rights, authorizations, obligations, and conditions. Authorizations, obligations and conditions are functional predicates that have to be evaluated for usage decision. Each predicate can be divided into detailed predicates. Subjects, objects and rights can be divided into several detailed components with different perspectives. Traditional access controls utilize only authorizations for decision process. Obligations and conditions are new concepts that have been discussed recently to resolve certain shortcomings shown in traditional access controls. These three decision factors will be used for the development of various detailed models.

A significant innovation in ABC is that subject and object attributes can be mutable. **Mutable attributes** are changed as a consequence of access, whereas immutable attributes can be changed only by administrative action. Policies requiring limits on

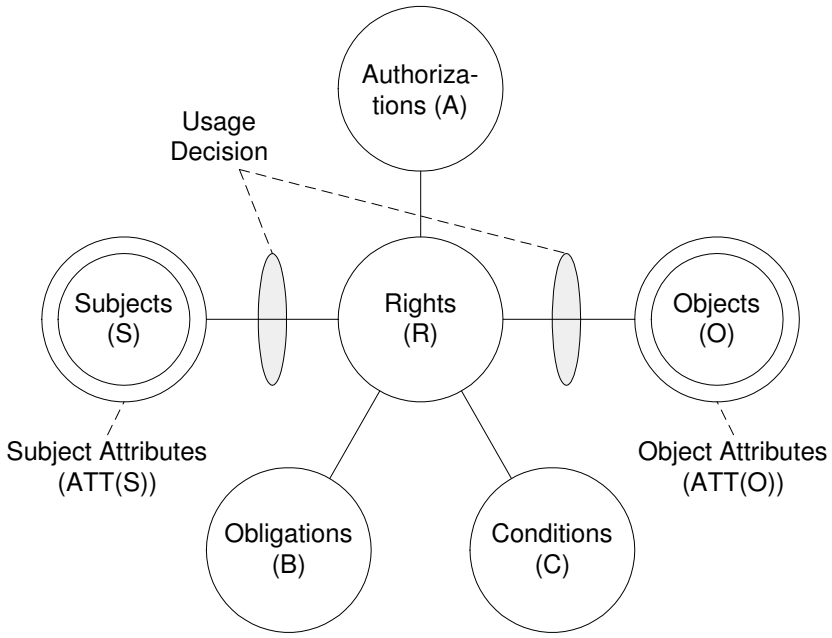


Figure 4.1: ABC Model Components

the number of accesses by a subject or reduction of account balance based on access can be easily specified using mutable attributes. More generally, various kinds of consumable authorizations can be modelled in this manner. High watermark policies on subject clearance and Chinese Walls can also be enforced in this way. The introduction of mutable attributes is a critical differentiator of ABC relative to most proposals for enhanced models for access control. Mutable attributes add further complication to the requirement for obtaining timely values of attributes from a trusted source, since now the attributes must also be modifiable in a trusted way.

4.1.1 Subjects (S) and Subject Attributes (ATT(S))

A subject is an entity associated with attributes, and holds or exercises certain rights on objects. For simplicity, a subject in usage control can be regarded as representing an individual human being. A subject is defined and represented by its attributes.

Subject attributes are properties or capabilities of a subject that can be used for the usage decision process. A subject may or may not have a unique identity. If authorization is done with a user's unique identity, accountability can be provided. If not, anonymity can be supported. Some attributes such as pre-paid credits, or usage capabilities can be used without unique identity for anonymous usages or transferable rights. Examples of subject attributes include identities, group names, roles, memberships, security clearance, etc. A group is a set of users who holds same rights as a group. A role is a named collection of users and relevant permissions [SCFY96]. Groups and roles may have hierarchical relationships. The general concept of attribute-based access control is commonplace in the access control literature and as such this aspect of ABC builds upon familiar concepts.[1]

If an attribute is *immutable*, it cannot be changed by the user's activity. Only administrative actions can change such an attribute. A *mutable* attribute can be modified as a side effect of subjects' access to objects. Many examples of trust management and DRM are likely to utilize mutable attributes. Some examples of mutable attributes are credits/capabilities (eg., $10 worth usage, five times per day, print twice), security clearance with relaxed (weak) or no tranquility, usage log (eg., already read portion cannot be read again), etc.

In usage control, the subjects can be *consumer subjects (CS)*, *provider subjects (PS)*, or *identifiee subjects (IS)*. Consumer subjects are entities who exercise the rights to access the objects. An e-book reader, MP3 music listener and even a distributor of

[1]Using attributes for access decisions requires a trusted source for the values and their timeliness. There are many challenges to achieving this. The details of how this would be accomplished is an important implementation and architectural issue. Nonetheless, in OM-AM point of view it is important to keep the model separate from these implementation concerns, however important they may be. Keeping the model separate from implementation is critical to separating concerns and reaching fundamental understanding of disparate issues [San00]. This viewpoint will be sustained throughout the dissertation.

digital objects can be a consumer subject. Provider subjects are entities who provide an object and hold certain rights on it. Examples of provider subjects include an author of an e-book, a distributor of the book, a primary physician, etc. The identifiee subjects are entities who are identified in digital objects that include their privacy-sensitive information. A patient in health care system is an example of an identifiee subject. Although the concept of identifiee subjects always exists in case of privacy-sensitive information, identifiee subjects may or may not be included within UCON systems based on the system requirements or policies.

4.1.2 Objects (O) and Object Attributes (ATT(O))

Objects are a set of entities that subjects hold rights on, whereby the subjects can access or use objects. Objects are also associated with attributes, either by themselves or together with rights. As for subjects, object attributes include certain properties that can be used for access decisions. Examples of object attributes that are associated with objects are security labels, ownerships, classes, etc. Object classes can be used to categorize objects so authorization can be done based not only on individual objects but also sets of objects that belong to same class [SS94]. Examples of attributes for objects with rights are values, role permissions, etc. The values may be used to define how many credits are required to obtain a certain right on a specific object. For example, "Harry Potter" e-book together with a 'read' right may require $10 or the book with an additional 'print' right may require $15. Object attributes also can be mutable (eg., the number of play time on each item of music).

In UCON, objects can be either privacy sensitive or privacy non-sensitive. A privacy-sensitive object includes individually identifiable information that can cause privacy problems if not used properly. An UCON object can be either original or derivative. The derivative object in UCON is different from that of other DRM literature. DRM's

derivative objects are more like "reused" or "reproduced" objects. In DRM, the term "derivative" means derived (cited, quoted, or copied) from an original work to create another digital work that includes parts of the original work. In UCON, however, the derivative object is an object that is created in consequence of obtaining or exercising rights on an original object. For example, playing MP3 music file can create usage log information. This log data file is called a derivative object in UCON. To provide mutual protection on the rights of all involved subjects (consumer, provider and/or identifiee subjects), just like the original object, these derivative objects also have to be considered as target objects and must hold UCON properties and relations with other components. Based on their format, objects can be documents (e.g., .doc, .pdf, .ps), audio (e.g., .mp3, .wav), video (e.g., JPEG, DVD, MPEG), executable files (e.g., games), etc. Each may require its own application tools to be used.

4.1.3 Rights (R)

Rights are privileges that a subject can hold and exercise on an object. Rights consist of a set of usage functions that enables a subject's access to objects. Rights may or may not have a hierarchy. Like subjects and objects, rights can also be divided into consumer rights (CR), provider rights (PR), and identifiee rights (IR). In an access control viewpoint, rights enable access of a subject to an object in a particular mode, such as read or write. In this sense the ABC concept of right is essentially similar to the familiar concept of a right in access control. However there is a subtle difference in the ABC viewpoint in that ABC does not visualize a right as existing in some access matrix independent of the activity of the subject. Rather the existence of the right is determined when the access is attempted by the subject.[2] The **usage**

[2]One could argue that the access matrix is a conceptual entity and does not exist as such. Nonetheless, the traditional position has been that the access matrix is what we are enforcing. The embodiment of the access matrix by Access Control Lists (ACLs), Capabilities or a Access

decision functions indicated in figure 4.1 make this determination based on subject attributes, object attributes, authorizations, obligations and conditions. In general, rights include rights for direct use of objects (such as read), delegation of rights and rights for administering access (such as modify subject and object attributes that in turn determine access rights). This chapter does not consider delegation rights and administrative rights. Rights can be divided into many functional categories. The two most fundamental rights categories might be view and modify, possibly augmented with creation and deletion. We can also distinguish direct access rights that are used by a subject to access an object from administrative rights that are used to administer access as well manage the object. The Digital Rights Management community has published several studies on functional rights [Con02, GWW01, Ian02], categorizing them as render rights, transport rights, derivative works rights and utility rights [RTM02]. More generally, rights can be defined by specific applications such as credit and debit in an accounting application.

4.1.4 Authorizations ($\mathcal{A}$)

Authorizations are functional predicates that have to be evaluated for usage decision and return whether the subject (requester) is allowed to perform the requested rights on the object. Authorizations evaluate subject attributes, object attributes, and requested rights together with a set of authorization rules for usage decision. Authorizations can be either pre-authorizations ($preA$) or ongoing-authorizations (onA).

Relation is a means of representing a sparse data structure efficiently. In actual practice the rights of a subject to an object are often determined when the access is attempted, e.g., the ACL may authorize a group to read an object and the subject's membership in the group is determined by subject's attributes at access time. So the ABC viewpoint is more accurate with respect to actual practice. The ABC viewpoint has consequences for access review. Predicting which rights will be available when access is attempted becomes a problem more akin to safety analysis with respect to leakage of rights [HRU76], than a simple lookup of relevant data structures. The ABC view is more accurate with respect to real-world access control systems where the actual representation of rights is rarely as straightforward as in the access matrix.

$preA$ is performed before a requested right is exercised and onA is performed while the right is exercised. onA may be performed continuously or periodically during the time span of access. In general, most traditional access control policies including MAC, DAC, RBAC and Trust Management (TM) utilize some form of pre-authorization for their decisions. Also, some DRM decision processes are pre-authorizations. Although rarely implemented, an example of ongoing authorization would be the continued checking of revocation status during the exercise of usage. Thereby usage can be immediately terminated to enforce immediate revocation.

Certain authorizations may require updates on subject attributes and/or object attributes. These updates can be either pre, ongoing, or post. Security clearance with high watermark property requires updates before usage is performed. Metered usage payment requires updates after the usage is ended to calculate current usage time. Using pre-paid credits for usage time based metered payment requires periodic updates of the credits during the access to prevent overuse.

4.1.5 oBligations ($\mathcal{B}$)

Obligations are functional predicates that verify mandatory requirements a subject has to perform before or during a usage exercise. Obligations can be either pre-obligations ($pre\mathcal{B}$) or ongoing-obligations ($on\mathcal{B}$). $pre\mathcal{B}$ is a predicate that utilizes some kind of history functions to check if certain activities have been fulfilled or not and returns either 'true' or 'false'. A user may have to fill out some personal information before reading a company's white paper. Similarly, a user may have to agree to provide usage log information before listening to music files. $on\mathcal{B}$ is a predicate that has to be satisfied continuously or periodically while the allowed rights are in use. A user may have to keep watching certain advertisements while he has logged in. Obligations may or may not utilize subject or object attributes. Attributes

can be used to determine what kind of obligations are required for usage approval. Obligations may require certain updates on subject attributes. These updates are likely to affect either current or future usage decisions. Note that attributes are not used for decision making with respect to obligations, but only for choosing what obligations apply.

One of the basic assumptions in the ABC model is that its decision-making process is transaction-based. This means that decision predicates are evaluated upon each usage request and the decision influences usages of that request. A *pre* decision predicate decides approval or denial of the request. An *ongoing* predicate may revoke or continue to allow current exercise of the requested usage. [3]

Obligations in UCON are different from duties in that duties are assigned to subjects regardless of the subjects' requests, and essential for an organization, whereas UCON obligations are requirements that have to be fulfilled and checked before or during the usage of certain rights. Traditional access control has hardly recognized the obligation concept. Some DRM solutions include obligation functions though many of them implement the obligation functions only partially or implicitly. The ABC model does not include duties. I feel that including duties within the models will distract my original purpose and cause unnecessary complexity.

4.1.6 Conditions ($\mathcal{C}$)

Conditions are environmental or system-oriented decision factors. Condition predicates evaluate current environmental or system status to check whether relevant

[3]Unlike authorizations or conditions, there can be transaction-independent, global obligations where the obligations influence only future requests and have no affects on the current request decision. For example, a user may have to fill out monthly evaluation reports for continuous subscription of a digital library, or a user may have to provide usage log information to a provider. These global obligations have to be fulfilled for future usages in timely manner (either time-based or event-based). Such global obligations are outside the scope of ABC core models.

requirements are satisfied or not and return either 'true' or 'false'. Subject attributes or object attributes can be used to select which condition requirements have to be used for a request. However no attribute is included within the requirements themselves. Unlike authorizations or obligations, condition variables cannot be mutable since conditions are not under direct control of individual subjects. Evaluation of conditions cannot update any subject or object attributes. Some examples of condition requirements include current local time for accessible time period (e.g., business hours), current location for accessible location checking (e.g., area code, device, CPU-ID), security status of the system (e.g., normal, high alert, under attack), system load, etc.

In the ABC model, one may separate a device from conditions and consider it as a different component of the model just like other components such as subjects, objects, and rights. Intuitively, since majority of current computing systems are quite mobile and network-connected, and digital information is virtually available anywhere, usage rules can specify allowed devices explicitly along with subjects, objects, and rights. While this may be true, I prefer to include device component within conditions since there are other subject and object independent factors such as time periods, and system load.

Conditions are different from authorizations in that conditions mainly focus on evaluations of environmental, system-related restrictions that have no direct relationship with subject and object attributes for usage decision (that is, subject and object attributes are not included within condition requirements hence not required for usage decision process) whereas authorizations evaluate attributes that are related to subjects (requesters) or requested objects for usage decision. [4]

[4]It should be noted there are some ambiguities as to how specific items should be treated. The IP address of a client can be viewed as a subject attribute, but can also be viewed as a condition

4.2 The ABC Family of Core Models

All of these components makes for a fairly complex model. Nonetheless, I believe, the recognition of three distinct factors, authorizations, obligations and conditions, along with mutability of attributes and continuity of enforcement is critical to supporting modern access control requirements. The resulting complexity is appropriate for modern cyberspace. Based on the eight components discussed above I develop a framework for classifying ABC models. I say these are core models because, as discussed earlier, they focus on the enforcement process for consumer-side only and do not include administrative constructs. Also they will need to be further elaborated for specific applications, as will be discussed later.

The model classification is based on the following three criteria: *decision factors* that consist of authorizations, obligations, and conditions, *continuity* of decision being either pre or ongoing with respect to the access in question, and *mutability* that can allow updates on subject or object attributes at different times. If all attributes are immutable, no updates are possible as a consequence of the decision process. This case is denoted as '0'. With mutable attributes, updates are possible before (pre), during (ongoing), or after (post) the right is exercised, denoted as '1, 2, and 3', respectively. Based on these criteria, I enumerate the model space shown in Table 4.1.

Cases that are not likely to be useful in practice are marked as 'N'. If decision factor is 'pre', updates can occur before or after the right is exercised but there is little reason to have ongoing updates. Without ongoing decision, ongoing-update can only influence decisions on future requests and therefore the updates can be done after the

indicating the location of the client. Whether the IP address is viewed as a subject attribute or a condition element is a choice that the system architect has to make. Rather than trying to provide air-tight boundaries between these concepts, I recognize them as somewhat fuzzy. It is generally true that a rich model will accommodate multiple ways of specifying a given policy. This is no different in the case of ABC.

Table 4.1: The 16 Basic ABC Models

	0 (immutable)	1 (pre-update)	2 (ongoing-update)	3 (post-update)
preA	Y	Y	N	Y
onA	Y	Y	Y	Y
preB	Y	Y	N	Y
onB	Y	Y	Y	Y
preC	Y	N	N	N
onC	Y	N	N	N

usage is ended. For example, suppose Alice is a member of a digital music library. Suppose she has to pay $1/hour of music play. This example can be handled as a pre-authorization with post update case. Her usage time is accumulated on her usage log file as each play ends. This case does not require any updates during the playing of a music track. However, if decision factor is 'ongoing', updates can happen before, during or after the right is exercised. These updates are used for current usage decision. This explains the top four rows of Table 4.1. For the bottom two rows the only decision factor is conditions. Evaluation of conditions cannot update attributes by definition. The resulting 16 Y's in Table 4.1 define the 16 basic ABC models. The A and B models have 7 Y's each whereas the C model has only 2 Y's. In practice, many real-world systems will use some combination of these models.

Figure 4.2(a) shows possible combinations of ABC models and their relationships. It is regarded that each of A, B or C is on equal footing, hence the three base models at the bottom. At the next level up combinations of two of these exist, and further combination of all three. In this way we can succinctly represent which combination of A, B and C is being used in a given context. Each of the A, B and C models is divided into several cases as respectively shown in Figures 4.2 (b), (c) and (d). In totality these comprise the 16 Y's of Table 4.1. The pre and ongoing cases are regarded as being on equal footing. The case of mutable attributes (1, 2 and 3) always

dominates immutable attributes (0), but there is no ordering between the 1, 2 and 3 cases. Figure 4.2 graphically demonstrates the richness of the model space available in the ABC family.

The ABC model definitions do not express the mechanistic details such as how updates can be enforced. Actual pre-updates may have to be performed either before the requested rights are exercised (e.g., obtaining a lock for mutual exclusion) or right after the rights are started (e.g., e-cash decrease) [RN01, RN02]. However ABC Model does not include such implementation issues following my practice of keeping the model and architecture-mechanism distinct.

In this dissertation I am not trying to develop a logical expression language for the ABC model. Rather, while there can be numerous ways of expressing the ABC model, my focus is to develop comprehensive models for usage control that can support modern access control requirements as well as DRM in a single framework and discuss the detailed characteristics of these in a systematic manner. Many current DRM solutions utilize obligations and conditions for decision process though they may not define these factors explicitly.

4.2.1 $UCON_{preA}$ - pre-Authorizations Models

Authorizations have been considered as the core of access control and extensively discussed since the beginning of access control discipline. Traditionally, access control research has focused on pre-authorizations in which a usage decision is made before a requested right is exercised. $UCON_{preA}$ models utilize these pre-authorizations for their usage decision processes. In $UCON_{preA}$ models, an authorization decision process is done before usage is allowed. There are three detailed models based on mutability variations. $UCON_{preA_0}$ is immutable pre-authorization model that requires no update. $UCON_{preA_1}$ is pre-authorization model with an optional pre-update pro-

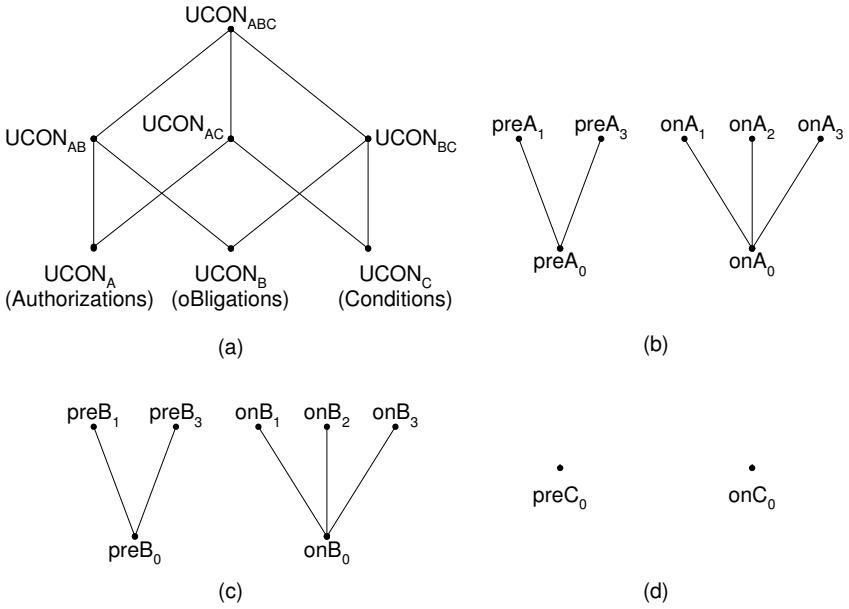


Figure 4.2: The ABC Family of Core models

cedure. A pre-update includes update functions that modify attributes before usage is started. $UCON_{preA_3}$ is pre-authorization model with an optional post-update procedure. A post-update utilizes update functions to modify certain attributes after usage is terminated.

The following definitions formalize the $UCON_{preA}$ models. Although some definitions are quite similar and the expressions can be reduced to a certain degree, I explicitly express each of the detail models for completeness of the models. This is done throughout the model definitions in this chapter.

Definition 1 The $UCON_{preA_0}$ model has the following components:

- $S, O, R, ATT(S), ATT(O)$ and $pre\mathcal{A}$ (subjects, objects, rights, subject attributes, object attributes, and pre-authorizations respectively);

- $allowed(s, o, r) \Rightarrow pre\mathcal{A}(ATT(s), ATT(o), r)$.

Definition 2 The $UCON_{preA_1}$ model is identical to $UCON_{preA_0}$ except it adds following pre-update processes:

- $preUpdate(ATT(s)), preUpdate(ATT(o))$, an optional procedure to perform update operations on $ATT(s)$ and $ATT(o)$, respectively. Note that $preUpdate$ can include non-deterministic operations.

Definition 3 The $UCON_{preA_3}$ model is identical to $UCON_{preA_0}$ except it adds following post-update processes:

- $postUpdate(ATT(s)), postUpdate(ATT(o))$, an optional procedure to perform update operations on $ATT(s)$ and $ATT(o)$, respectively. Note that $postUpdate$ can include non-deterministic operations.

$UCON_{preA_0}$ consists of subjects (S), objects (O), rights (R), subject attributes ($ATT(S)$), object attributes ($ATT(O)$), and pre-authorizations ($pre\mathcal{A}$). $pre\mathcal{A}$ is a functional predicate that utilizes $ATT(S)$, $ATT(O)$, and R for usage decision making. $pre\mathcal{A}$ examines usage requests using $ATT(S)$, $ATT(O)$, and R then decides whether the request is allowed or not. I write $allowed(s, o, r)$ to indicate that subject s is allowed right r to object o. Note that the ABC model formulates 'implies' connectives rather than 'if'. This means that righthand-side of the connective is not sufficient to allow usages but is necessary. A similar approach can be found in Bell LaPadula (BLP) model. In BLP, mandatory access control is formulated as 'necessary condition' and used together with discretionary access control to enforce additional information flow policies [BL73, San93]. Throughout the models, I formulate 'necessary condition'

rather than 'sufficient condition' so decision process can include other rules that might be necessary for finer and richer controls.

The meaning of $preUpdate(ATT(s))$ is that subject attributes are updated. Exactly what values can be used in computing the update is left unspecified in the model. These could be subject, object attributes and other variables. Similarly for the other update processes in Definition 2 and 3. Traditional access controls such as MAC, DAC, and RBAC are likely to belong to $UCON_{preA_0}$. Following Examples 1 and 2 respectively show how traditional MAC and DAC can be realized within $UCON_{preA_0}$.

Example 1 MAC policies, $UCON_{preA_0}$:

L is a lattice of security labels with dominance relation $\geq$

$clearance : S \rightarrow L$

$classification : O \rightarrow L$

$ATT(S) = \{clearance\}$

$ATT(O) = \{classification\}$

$allowed(s, o, read) \Rightarrow clearance(s) \geq classification(o)$

$allowed(s, o, write) \Rightarrow clearance(s) \leq classification(o)$

Example 2 DAC closed policies using ACL with an individual ID, $UCON_{preA_0}$:

N is a set of identity names

$id : S \rightarrow N$, one to one mapping

$ACL : O \rightarrow 2^{N \times R}$

$ATT(S) = \{id\}$

$ATT(O) = \{ACL\}$

$allowed(s, o, r) \Rightarrow (id(s), r) \in ACL(o)$

In case of MAC, security labels (clearance and classification) are used as a subject attribute ($ATT(S)$) and an object attribute ($ATT(O)$), and Bell-LaPadula's security properties (simple and star property) [BL73, San93] are utilized for pre-authorizations ($pre\mathcal{A}$). Here, if the clearance of subject s dominates the classification of object o, 'read' requests are allowed. Similarly, 'write' is allowed if clearance(s) is dominated by classification(o). In case of DAC example, individual (or group) identities and access control lists (ACL) are subject attributes and object attributes, respectively. ACL is a functional mapping of object to multiple *id*s and rights. If the subject's identity name together with the requested right exists in ACL, the request is allowed. Although it has been understood that ACL and capability list are used to achieve similar control functionalities, they can actually provide quite different results when they are used with mutable attributes. This is discussed in next section. In RBAC, a user-role and a permission-role can be considered as a $ATT(S)$ and a $ATT(O)$ respectively and compared for authorizations before allowing access. Details of MAC, DAC, RBAC, and other related areas to ABC model are discussed in next section. Also, certain authorization processes of DRM (e.g., membership-based digital library) can be expressed within $UCON_{preA_0}$ as follows. We illustrate two examples of DRM with pre-updates and post-updates respectively.

Example 3 DRM pay-per-use with a pre-paid credit, $UCON_{preA_1}$:

M is a set of money amount

$credit : S \rightarrow M$

$value : O \times R \rightarrow M$

$ATT(s) : \{credit\}$

$ATT(o, r) : \{value\}$

$allowed(s, o, r) \Rightarrow credit(s) \geq value(o, r)$

$preUpdate(credit(s)) : credit(s) = credit(s) - value(o, r)$

Example 4 DRM membership-based metered payment, $UCON_{preA_3}$:

M is a set of money amounts

ID is a set of membership identification numbers

$TIME$ is a current usage minute

$member : S \rightarrow ID$

$expense : S \rightarrow M$

$usageT : S \rightarrow TIME$

$value : O \times R \rightarrow M$ (a cost per minute of r on o)

$ATT(s) : \{member, expense, usageT\}$

$ATT(o, r) : \{valuePerMinute\}$

$allowed(s, o, r) \Rightarrow member(s) \neq \phi$

$postUpdate(expense(s)) : expense(s) = expense(s) + (value(o, r) \times usageT(s))$

In Example 3, if the credit of a subject s is not less than the value of the requested usage, the request is allowed. Once the request is allowed, the subject's credit is reduced by the value of the usage. Example 4 shows membership-based usage controls with metered payment. In this case, a request is allowed if the subject is a member. However, a total expense has to be updated at the end of each usage by using current usage time and value of the usage so it can be paid periodically. [5]

$UCON_{preA_1}$ and $UCON_{preA_3}$ are unchanged from $UCON_{preA_0}$ except the additional update procedures. $UCON_{preA_1}$ and $UCON_{preA_3}$ introduce '*preUpdate*' and '*postUpdate*' procedures to modify subject attributes and object attributes. Many B2B and B2C applications require some form of update functionalities. Recent DRM solutions have also dealt with these. For example, pre-paid credits have to be reduced at the time a user is allowed to exercise requested rights on an object ($UCON_{preA_1}$).

4.2.2 $UCON_{onA}$ - ongoing-Authorizations Models

In $UCON_{onA}$ model, usage requests are allowed without any 'pre' decision-making. However, authorization decisions are made continuously or repeatedly while usage rights are exercised. If certain requirements become dissatisfied, the currently allowed usage right is revoked and its exercise is stopped. Ongoing authorizations have been seldom discussed in access control literature. By utilizing ongoing authorizations, monitoring is actively involved in usage decisions while a requested right is exercised. This kind of continuous control is especially useful for relatively long-lived

[5] Enforcement of periodic payment of accumulated expense is not considered as part of the core ABC model since it is not related to a request-based decision process but is part of larger workflow.

usage rights. In $UCON_{onA}$, there are four detailed models. $UCON_{onA_0}$ is immutable ongoing-authorization model that has no update procedure included. $UCON_{onA_1}$ is ongoing-authorization model with pre-updates. $UCON_{onA_2}$ and $UCON_{onA_3}$ include ongoing updates and post updates, respectively.

The following definitions formalize the $UCON_{onA}$ models.

Definition 4 The $UCON_{onA_0}$ model has the following components:

- $S, O, R, ATT(S), and ATT(O)$ are not changed from $UCON_{preA}$;
- $on\mathcal{A}$ (ongoing-authorizations);
- $allowed(s, o, r) \Rightarrow true$;
- $stopped(s, o, r) \Leftarrow \neg on\mathcal{A}(ATT(s), ATT(o), r)$.

Definition 5 The $UCON_{onA_1}$ model is identical to $UCON_{onA_0}$ except it adds following pre-update processes:

- $preUpdate(ATT(s)), preUpdate(ATT(o))$, an optional procedure to perform update operations on $ATT(s)$ and $ATT(o)$, respectively.

Definition 6 The $UCON_{onA_2}$ model is identical to $UCON_{onA_0}$ except it adds following ongoing-update processes:

- $onUpdate(ATT(s)), onUpdate(ATT(o))$, an optional procedure to perform update operations on $ATT(s)$ and $ATT(o)$, respectively.

Definition 7 The $UCON_{onA_3}$ model is identical to $UCON_{onA_0}$ except it adds following post-update processes:

- $postUpdate(ATT(s)), postUpdate(ATT(o))$, an optional procedure to perform update operations on $ATT(s)$ and $ATT(o)$, respectively.

$UCON_{onA_0}$ model introduces $on\mathcal{A}$ predicate instead of $pre\mathcal{A}$. Since there is no pre-authorization, the requested access is always allowed. However, ongoing-authorizations are active throughout the usage of the requested right, and certain requirements are repeatedly checked for continuous access. Semantically these requirements have to be true all the time while the right is exercised. Technically, these checking processes are likely to be performed periodically based on time or event. ABC model does not specify these technical and implementation details. In case certain attributes are changed and requirements are no longer satisfied, '*stopped*' procedure is performed. I write '$stopped(s, o, r)$' to indicate rights r of subject s to object o is revoked. In many cases, ongoing authorizations are likely to occur only together with pre-authorizations though ABC model does not require this. For example, suppose $on\mathcal{A}$ screens certain certificate revocation lists periodically to check whether the user's identity certificate is revoked or not. While this is a case of ongoing authorizations, this makes sense only when the certificate has already been evaluated at the time of the request. This can be an example of $UCON_{onA_0}$ model. $UCON_{onA_1}$, $UCON_{onA_2}$ and $UCON_{onA_3}$ are unchanged from $UCON_{onA_0}$ but add pre-updates, ongoing-update and post-update procedures respectively. Some examples of $UCON_{onA}$ models are given below.

Example 5 A limited number of simultaneous usages, revocation using usage start time, $UCON_{onA_{13}}$:

T is an ordered set of current usage start times

UN is a set of concurrent usage numbers

N is a set of identification names

$id : S \rightarrow N$

$usageNum : O \rightarrow UN$

$startT : O \rightarrow 2^{N \times T}$

$ATT(s) : \{id\}$

$ATT(o) : \{startT, usageNum\}$

$allowed(s, o, r) \Rightarrow true$

$stopped(s, o, r) \Leftarrow (usageNum(o) > 10) \wedge$

$$(id(s), t) \in startT(o) \text{ where } t = \min\{t' | \exists s', (id(s'), t') \in startT(o)\}$$

$preUpdate(startT(o)) : startT(o) = startT(o) \cup \{(id(s), t)\}$ where s is currently requesting subject of usage

$preUpdate(usageNum(o)) : usageNum(o) = usageNum(o) + 1$

$postUpdate(startT(o)) : startT(o) = startT(o) - \{(id(s), t)\}$ where s is a subject of stopped usage

$postUpdate(usageNum(o)) : usageNum(o) = usageNum(o) - 1$

Example 6 A limited number of simultaneous usages, revocation using longest idle time, $UCON_{onA_{123}}$:

T is an ordered set of last activity times

UN is a set of concurrent usage numbers

N is a set of identification names

$id : S \rightarrow N$

$usageNum : O \rightarrow UN$

$lastActiveT : O \rightarrow 2^{N \times T}$

$ATT(s) : \{id\}$

$ATT(o) : \{lastActiveT, usageNum\}$

$allowed(s, o, r) \Rightarrow true$

$stopped(s, o, r) \Leftarrow (usageNum(o) > 10) \wedge$

$(id(s), t) \in lastActiveT(o)$ where $t = \min\{t' | \exists s', (id(s'), t') \in lastActiveT(o)\}$

$preUpdate(usageNum(o)) : usageNum(o) = usageNum(o) + 1$

$onUpdate(lastActiveT(o))$, repeated updates on $lastActiveT(o)$

$postUpdate(usageNum(o)) : usageNum(o) = usageNum(o) - 1.$

Example 7 A limited number of simultaneous usages, revocation using total usage time, $UCON_{onA_{13}}$:

T is an ordered set of current usage times

TT is an ordered set of total usage times

UN is a set of concurrent usage numbers

N is a set of identification names

$id : S \rightarrow N$

$totalT : O \rightarrow 2^{N \times TT}$, A functional mapping of object to a set of total usage times of active subjects

$usageNum : O \rightarrow UN$

$ATT(s) : \{id\}$

$ATT(o) : \{usageT, totalT, usageNum\}$

$allowed(s, o, r) \Rightarrow true$

$stopped(s, o, r) \Leftarrow (usageNum(o) > 10) \wedge$

$(id(s), tt) \in totalT(o)$ where $tt = \max\{tt' | \exists s', (id(s'), tt') \in totalT(o)\}$

$preUpdate(usageNum(o)) : usageNum(o) = usageNum(o) + 1$

$postUpdate(usageNum(o)) : usageNum(o) = usageNum(o) - 1$

$postUpdate(totalT(o)) : (id(s), tt) = (id(s), tt + t)$, where s is a subject of stopped usage and t is current usage time of the s

In Example 5, suppose only 10 users can access an object o_1 simultaneously. If a 11th user requests access, the user with the earliest time is terminated. In this case, the 11th user is allowed without any pre-authorization decision process. However, $on\mathcal{A}$ monitors the number of current usages on o_1 ($ATT(o_1)$), determines which was the earliest to start, and terminates it. Here, starting time of each request has to be added before the beginning of the requested usage and has to be taken out after the usage is stopped. Also current usage number of o_1 is increased by 1 at the time of access and decreased by 1 at the end of the access, hence a $UCON_{onA_{13}}$ model. Suppose the extra user in the above example is revoked based on longest idle time. Monitoring idle time requires ongoing updates of a last activity attribute as shown in Example 6. Further suppose that revocation of the extra user is based on total usage time in completed sessions since the start of the fiscal year. Post-updates would be needed to accumulate current usage time as shown in Example 7.

4.2.3 $UCON_{preB}$ - pre-oBligations Models

$UCON_{preB}$ introduces pre-obligations that have to be fulfilled at the time of a request and before access is allowed. $pre\mathcal{B}$ is a kind of history function that checks whether certain obligations have been fulfilled or not and return true or false for the usage decision. Suppose a user has to provide his name and email address to download a company's white papers or suppose a user has to click 'O.K' on a license agreement to access a web portal. Here, the user has to fulfill the required actions before access is allowed. $UCON_{preB}$ models consist of 2 steps. First step is to select required obligation elements for the requested usage. This selection may utilize subject and/or object attributes. Second step is to evaluate whether the selected obligation elements have been fulfilled without any error (e.g., invalid e-mail addresses). In $UCON_{preB}$ models, a request may require multiple pre-obligation elements to be fulfilled. The $pre\mathcal{B}$ predicate evaluates if all the required pre-obligation elements ($preOBL$) are fulfilled by using $preFulfilled$ and returns either true or false.

The following definitions formalize the $UCON_{preB}$ models.

Definition 8 The $UCON_{preB_0}$ model has the following components:

- $S, O, R, ATT(S)$, and $ATT(O)$ are not changed from $UCON_{preA}$;
- OBS, OBO, and OB, (obligation subjects, obligation objects, and obligation actions, respectively);
- $pre\mathcal{B}$, and $preOBL$, (pre-obligation predicates and pre-obligation elements, respectively);
- $preOBL \subseteq OBS \times OBO \times OB$;
- $preFulfilled : OBS \times OBO \times OB \rightarrow \{true, false\}$;

- $getPreOBL : S \times O \times R \rightarrow 2^{preOBL}$, a function to select pre-obligations for a requested usage;

- $pre\mathcal{B}(s, o, r) = \bigwedge_{(obs_i, obo_i, ob_i) \in getPreOBL(s,o,r)} preFulfilled(obs_i, obo_i, ob_i)$;
  $pre\mathcal{B}(s, o, r) = true$ by definition if $getPreOBL(s, o, r) = \phi$;

- $allowed(s, o, r) \Rightarrow pre\mathcal{B}(s, o, r)$.

Definition 9 The $UCON_{preB_1}$ model is identical to $UCON_{preB_0}$ except it adds following pre-update processes:

- $preUpdate(ATT(s)), preUpdate(ATT(o))$: an optional procedure to change certain attributes as a consequence of pre-obligations.

Definition 10 The $UCON_{preB_3}$ model is identical to $UCON_{preB_0}$ except it adds following post-update processes:

- $postUpdate(ATT(s)), postUpdate(ATT(o))$: an optional procedure to change certain attributes as a consequence of pre-obligations.

Decisions on what kind of obligations are required for requests ($getPreOBL$) are rather complicated and have various patterns. Subject or object attributes may or may not be used for the decisions. If no attribute is used, a user is likely to be required to fulfill same obligations at each request. For example, without a subject attribute, we cannot recognize previous users who have previously provided name and email address. We therefore have to ask for same obligations again. Selection of required

obligation elements can be based on subject attributes only, object attributes only, both subject and object attributes, rights only, or all three of subject attributes, object attributes and rights. Suppose a subject has to read a license agreement and click 'OK' button before he exercises any rights on Web Services, or suppose a subject has to provide his personal information (age, gender, organization, e-mail address, etc) to download a company's white paper. The first obligation case can be decided based on target objects regardless of subject attributes or rights. The second one can be based on object attributes and rights. In Web Services example, suppose members have to report a monthly usage history, and guests have to provide their names and e-mail addresses. These obligations are decided based on subject attributes. Again with the Web Service example, suppose guests who only want to surf (read) contents have to provide name and e-mail address, and who want to participate and write some messages on discussion board have to provide their unique ID number such as a Social Security Number for accountability. These obligations are determined based on rights.

Obligation elements consist of OBS, OBO, and OB. The entity (obs) who has to perform obligation may or may not be the same subject as the requester (s). For example, to be a member of a Web community, children may need parents' agreements. In this case, the parents (obs) are different entities from the child (s) who wants to be a member. The entity (obo) on which the obligation has to be performed can be either a constant or a function of the subject attributes, object attributes and/or rights. Suppose whoever wants to access a digital library has to provide name and address. Here, name and address (obo) is a constant regardless of subject attributes, object attributes and rights. If a non-member has to provide something different from what a member has to provide, we can say obo is different for different subject attributes. Similarly, what has to be performed (ob) can be either a constant or a

function of *obo*. For example, suppose Alice has to read and agree a license agreement for certain services by clicking 'yes' button. Here, the *ob* is always the 'click' action because the *obo* is the license agreement.

The ABC model does not specify these details. Rather, an abstract function called '$getPreOBL$' is used to obtain required obligations leaving the detail and internal decision functions as an implementation decision. After all, it can be said that obligations are decided based on requests that consist of s, o, and r. Some examples are given below.

Example 8 A license agreement obligation, every time (without attribute), $UCON_{preB_0}$:

$OBS = S$

$OBO = \{license\_agreement\}$

$OB = \{agree\}$

$getPreOBL(s, o, r) = \{(s, license\_agreement, agree)\}$

$allowed(s, o, r) \Rightarrow preFulfilled(s, license\_agreement, agree)$

Example 9 High or low license agreements for high/low objects (with object attributes), $UCON_{preB_0}$:

$OBS = S$

$OBO = \{high\_license\_agreement, low\_license\_agreement\}$

$OB = \{agree\}$

$level : O \rightarrow \{high, low\}$

$ATT(o) = \{level\}$

$$getPreOBL(s,o,r) = \begin{cases} (s, high\_license\_agreement, agree), & \text{if } level(o) = \text{'high'}; \\ (s, low\_license\_agreement, agree), & \text{if } level(o) = \text{'low'}. \end{cases}$$

$allowed(s,o,r) \Rightarrow preFulfilled(getPreOBL(s,o,r))$

Example 10 Selective license agreements for first time users only, $UCON_{preB_1}$:

$OBS = S$

$OBO = \{license\_agreement\}$

$OB = \{agree\}$

$registered : S \rightarrow \{yes, no\}$

$ATT(s) = \{registered\}$

$$getPreOBL(s,o,r) = \begin{cases} (s, license\_agreement, agree), & \text{if } registered(s) = \text{'no'}; \\ \phi, & \text{if } registered(s) = \text{'yes'}. \end{cases}$$

$allowed(s,o,r) \Rightarrow preFulfilled(getPreOBL(s,o,r))$

$preUpdate(registered(s)) : registered(s) = yes$

In Example 8, a license agreement is required from every request regardless of the user's previous agreements. This example does not require any subject and object attributes. Suppose a Web service requires a license agreement. In this case, *obs* is same as *s* and *obo* and *ob* are constants. This case can be extended to require different obligations for different object attributes or different subject attributes. Example 9 shows a case that requires two different license agreements for high and low objects. Note that these attributes are used for obtaining required obligations not for making usage decisions. Example 10 requires a license agreement only once. To do this, a

subject attribute called '*registered*' is used. Once a user has agreed on a license agreement, the user is registered (*preUpdate*) for future requests. Still, this attribute is not directly used for authorizations. To be an authorization model, this attribute has to influence usage decision results. No $UCON_{preB_3}$ example is shown in this paper. However we can easily modify Example 10 to include post updates. If a user has to agree on a licence agreement at every 5 hours of accumulated usage, the total usage time has to be updated at the end of each usage for future requests.

4.2.4 $UCON_{onB}$ - ongoing-oBligations Models

$UCON_{onB}$ models are similar to $UCON_{preB}$ models except that obligations have to be fulfilled while rights are exercised. Ongoing-obligations may have to be fulfilled periodically or continuously. For this, a time parameter T is introduced as part of obligation elements $onOBL$. Here, T is likely to define certain time intervals that are either time-based or event-based. For example, a user may have to click an advertisement within every 30 minute interval or within every 20 Web pages accessed, or a user may have to keep an advertisement window active all the time. Note that our concern is about when users have to fulfill obligations, not when a system actually checks the fulfillments. The model assumes that $on\mathcal{B}$ has to be true all the time though actual obligation verification intervals can vary. In $UCON_{onB}$ models, there are four detailed models based on mutability issues. $UCON_{onB_0}$ includes ongoing-obligations predicate instead of pre-obligations predicate. $UCON_{onB_1}$, $UCON_{onB_2}$ and $UCON_{onB_3}$ are same as $UCON_{onB_0}$ except that they add pre-updates, ongoing-updates and post-updates, respectively.

The following definitions formalize the $UCON_{onB}$ models.

Definition 11 The $UCON_{onB_0}$ model has the following components:

- $S, O, R, ATT(S), ATT(O), OBS, OBO$, and OB are not changed from $UCON_{preB}$;
- T, a set of time or event elements;
- $on\mathcal{B}$ and $onOBL$, (ongoing-obligations predicates and ongoing-obligation elements, respectively);
- $onOBL \subseteq OBS \times OBO \times OB \times T$;
- $getOnOBL : S \times O \times R \rightarrow 2^{onOBL}$, a function to select ongoing-obligations for a requested usage;
- $onFulfilled : OBS \times OBO \times OB \times T \rightarrow \{true, false\}$;
- $on\mathcal{B}(s, o, r) = \bigwedge_{(obs_i, obo_i, ob_i, t_i) \in getOnOBL(s,o,r)} onFulfilled(obs_i, obo_i, ob_i, t_i)$;
- $on\mathcal{B}(s, o, r) = true$ by definition if $getOnOBL(s, o, r) = \phi$;

- $allowed(s, o, r) \Rightarrow true$;
- $stopped(s, o, r) \Leftarrow \neg on\mathcal{B}(s, o, r)$.

Definition 12 The $UCON_{onB_1}$ model is identical to $UCON_{onB_0}$ except it adds following pre-update processes:

- $preUpdate(ATT(s)), preUpdate(ATT(o))$: an optional procedure to change certain attributes as a consequence of pre-obligations.

Definition 13 The $UCON_{onB_2}$ model is identical to $UCON_{onB_0}$ except it adds following ongoing-update processes:

- $onUpdate(ATT(s)), onUpdate(ATT(o))$: an optional procedure to change certain attributes as a consequence of pre-obligations.

Definition 14 The $UCON_{onB_3}$ model is identical to $UCON_{onB_0}$ except it adds following post-update processes:

- $postUpdate(ATT(s)), postUpdate(ATT(o))$: an optional procedure to change certain attributes as a consequence of pre-obligations.

The following Example 11 shows a simple example for $UCON_{onB_0}$.

Example 11 Watch advertisement windows while s exercise r, $UCON_{onB_0}$:

$OBS = S$

$OBO = \{ad\_window\}$

$OB = \{keep\_active\}$

$T = \{always\}$

$getOnOBL(s, o, r) = \{(s, ad\_window, keep\_active, always)\}$

$allowed(s, o, r) \Rightarrow true$

$stopped(s, o, r) \Leftarrow \neg onFulfilled(s, ad\_window, keep\_active, always)$

Here, there is only one ongoing obligation is required. Suppose a free Internet service provider requires users to watch an advertisement while they are connected to the

server. In this case, there is no requirement that has to be completed before using the service. As long as the advertisement window is active, the usage is allowed. Again how frequently the system checks the status of the advertisement window is not considered. Although examples for $UCON_{onB_1}$, $UCON_{onB_2}$ and $UCON_{onB_3}$ are not shown here, they can be developed without much effort. For example, consider free Internet Services. Suppose every user has to watch ad for first 10 minutes of connection to Internet, but every 10th user has to watch ad for 20 minutes. Here every time a user connect Internet, usage number has to be increased or reset to 0 at the beginning of access to decide which ongoing obligation has to be fulfilled. This is an example of $UCON_{onB_1}$. Suppose a user has to click an advertisement within every 30 minute window. Here a last click time has to be updated throughout usage. This is an example of $UCON_{onB_2}$. Further suppose a user has to watch advertisement after first 10 hours every month. Here, current connection time has to be accumulated at the end of each connection. This is an example of $UCON_{onB_3}$.

4.2.5 $UCON_{preC}$ - pre-Conditions Model

As described earlier, conditions define certain environmental restrictions that have to be satisfied for usages. In general, *preCON* includes certain environmental restrictions that are not directly related to subjects and objects. Current environmental or system-oriented status is retrieved each time a condition is evaluated. By utilizing conditions in usage decision process, $UCON_C$ can provide finer-grained controls on usages. Unlike authorization and obligation models, condition models cannot be mutable. Note that this is different from the fact that the value of conditional status can be changed as the environmental situation is being changed (e.g., current time is changed as time goes, or a wireless access point is changed as a user moves around

a building). Although subject or object attributes are not used for usage decision process, they can be used to decide what kind of condition elements ($preCON$) have to be used for usage decision. $UCON_{preC}$ introduces pre-conditions ($pre\mathcal{C}$) that has to be satisfied before requested rights are exercised.

The following definitions formalize the $UCON_{preC}$ model.

Definition 15 The $UCON_{preC_0}$ model has the following components:

- $S, O, R, ATT(S)$, and $ATT(O)$ are not changed from $UCON_{preA}$;
- $preCON$ (a set of pre-conditions elements);
- $getPreCON : S \times O \times R \rightarrow 2^{preCON}$;
- $preConChecked : preCON \rightarrow \{true, false\}$;
- $pre\mathcal{C}(s, o, r) = \bigwedge_{preCon_i \in getPreCON(s,o,r)} preConChecked(preCon_i)$
- $allowed(s, o, r) \Rightarrow pre\mathcal{C}(s, o, r)$.

In $UCON_{preC_0}$, $pre\mathcal{C}$ is utilized in usage decision process along with S, O and R. A set of relevant condition elements $preCON$ is selected based on a request possibly using subject or object attributes. To allow a request, all of the selected condition restrictions have to be evaluated ($pre\mathcal{C}$). For example, suppose there are requirements to restrict locations where usages can be exercised. Checking a CPU-id or an IP address before a usage allowance is an example of $UCON_{preC_0}$. Example 12 checks the current location of a user at the time of a request. Allowed locations for student and faculty

can be different and have to be selected accordingly. This example assumes either there is no location change while the request is exercised or there is no restriction of location changes during the usages once the original location has been approved.

Example 12 Location limitation, $UCON_{preC_0}$:

$studentAREA, facultyAREA$ (allowed area codes for student and faculty)

$curArea$ is a current rendering device's area code

$ATT(s) = \{member\}$

$preCON = \{(curArea \in studentAREA), (curArea \in facultyAREA)\}$

$$getPreCON(s,o,r) = \begin{cases} (curArea \in studentAREA), & \text{if } member(s) = \text{'student'}; \\ (curArea \in facultyAREA), & \text{if } member(s) = \text{'faculty'}. \end{cases}$$

$allowed(s,o,r) \Rightarrow preConChecked(getPreCON(s,o,r))$

4.2.6 $UCON_{onC}$ - ongoing-Conditions Model

In many cases, environmental restrictions have to be satisfied while rights are in active use. This could be supported within $UCON_{onC}$ model. In $UCON_{onC}$, usages are allowed without any decision process at the time of requests. However, there is an ongoing conditions predicate to check certain environmental status repeatedly throughout the usages. As mentioned earlier, the $UCON_{onC_0}$ model is intrinsically immutable.

The following definitions formalize the $UCON_{onC}$ model.

Definition 16 The $UCON_{onC_0}$ model has the following components:

- $S, O, R, ATT(S)$, and $ATT(O)$ are not changed from $UCON_{preA}$;
- $onCON$ (a set of ongoing-conditions elements);
- $getOnCON : S \times O \times R \rightarrow 2^{onCON}$;
- $onConChecked : onCON \rightarrow \{true, false\}$;
- $on\mathcal{C}(s, o, r) = \bigwedge_{onCon_i \in getOnCON(s,o,r)} onConChecked(onCon_i)$

- $allowed(s, o, r) \Rightarrow true$;
- $stopped(s, o, r) \Leftarrow \neg on\mathcal{C}(s, o, r)$.

The $UCON_{onC_0}$ model introduces an ongoing conditions predicate ($on\mathcal{C}$) for monitoring selected condition elements ($getOnCON(s, o, r)$). If any current environmental status violates any of the restrictions, the allowed right is revoked and the exercise is stopped. In Example 13 below, allowed time period limitation is required throughout usage exercises. For example, suppose a day-time user can access objects during day time (say 8am to 4pm), and a night-shift user can access objects during night time (say 4pm to 12pm). Note that, *currentT* is a current status (time) of local time, not an attribute of subject or object. Here, *currentT* is evaluated throughout the usage and if its value is no longer within the allowed period, the usage is stopped. This example is likely to use both pre-condition and ongoing-condition since current time is also likely to be checked at the time of request, hence a combined model $UCON_{preC_0onC_0}$.

Example 13 Time limitation, $UCON_{preC_0 onC_0}$:

$dayH, nightH$ (day-shift and night-shift office hours, mutually exclusive)

$currentT$ is current time

$preCON : \{(currentT \in dayH), (currentT \in nightH)\}$

$onCON : \{(currentT \in dayH), (currentT \in nightH)\}$

$$getPreCON(s, o, r) = \begin{cases} (currentT \in dayH), & \text{if } shift(s) = \text{'day'}; \\ (currentT \in nightH), & \text{if } shift(s) = \text{'night'}. \end{cases}$$

$$getOnCON(s, o, r) = \begin{cases} (currentT \in dayH), & \text{if } shift(s) = \text{'day'}; \\ (currentT \in nightH), & \text{if } shift(s) = \text{'night'}. \end{cases}$$

$allowed(s, o, r) \Rightarrow preConChecked(getPreCON(s, o, r))$

$stopped(s, o, r) \Leftarrow \neg onConChecked(getOnCON(s, o, r))$

4.2.7 Global Obligations

In ABC model, obligations are used for usage decision of current request. However, there are other cases where the evaluation of obligations' fulfillment is postponed for some time and used for future usage decision rather than usage decision of current request. Suppose a member has to pay monthly metered payment for continuous music services. Though total usage time has to be updated at each service, this does not influence usage decision of service requests until the payment due.

This kind of obligation is called global obligation and does not fit into the ABC model structure though it is also an important aspect in usage control. UCON considers

these global obligations as an exceptional case of UCON obligation models. Global obligations are unique in their characteristics. Like other obligations, they have to be fulfilled by obligation subjects. However, they do not affect any usage decisions on the originated requests. Rather, what they can do is to influence future requests. Although certain updates can be required for global obligations, these updates do not influence any decision for current usages. Since there is no influence on decision-making for current usages, this feature can't belong to $UCON_{B_0}$, hence can't belong to $UCON_{B_1}, UCON_{B_2}$, or $UCON_{B_3}$. Note that the actual updates still can happen either before, during, or after the usages. In case of an update after usage, unlike post-update, it does not have to be done right after the end of usage. For example, a user may have to fulfill usage log report at the end of each usage, each day, or each month. Monthly metered payment, or monthly subscription payment are also examples of global obligations. The actual influence on decision making is postponed to certain point. These global obligations are independent from the originated usage transactions and have impact only on future decisions.

4.3 Applications in ABC Model

Usage control encompasses traditional access controls, trust management, and digital rights management and goes further in its definition and the scope. This section presents how MAC, DAC, RBAC, trust management, and DRM can be realized within the ABC model. It further discusses some possible extensions of these policies for better understanding of their characteristics and richer controls. Most traditional access controls and trust management can be realized by using $UCON_{preA_0}$ model. Some extensions may require $UCON_{preA_1}$ model. Ongoing decisions are rarely discussed in literature. Mutability issues are not common in previous work. Most of the research that deals with continuity or mutability issues still lack systematic perspective and

comprehensiveness, being narrowly focused. Discussions on some of this prior work in terms of the ABC model provides solid evidence of comprehensiveness and richness of the model. We also show some healthcare system examples that require obligations, conditions as well as authorizations.

4.3.1 Mandatory Access Control

In mandatory access control, security labels are used for usage decisions. In UCON point of view, clearance is a subject attribute and classification is an object attribute. These security labels of subjects and objects are compared to enforce simple security property (no read up) and star property (no write down). Example 1 in previous section shows this MAC policy using the $UCON_{preA_0}$ model.

Traditional access controls have rarely supported an update property. In MAC, strong tranquility property which belongs to immutable authorizations is normally assumed. In other words, security labels of subjects and objects cannot be changed by users' actions. Only administrative actions can change the labels. With a weak tranquility property, security labels can be changed by users' autonomous actions but only without violating defined security policies. This has been known as a high watermark property. Example 14 shows this high watermark property of BLP as an example of $UCON_{preA_1}$. Here, a subject always start with the lowest possible clearance label. The clearance of the subject is raised to higher labels until it reaches its maximum label as the subject accesses higher objects. LUB denotes least upper bound.

Example 14 MAC policies with high watermark property, $UCON_{preA_1}$:

L is a lattice of security labels with dominance relation $\geq$

$clearance : S \rightarrow L$

$maxClearance : S \rightarrow L$

$classification : O \rightarrow L$

$ATT(S) = \{clearance, maxClearance\}$

$ATT(O) = \{classification\}$

$allowed(s, o, read) \Rightarrow maxClearance(s) \geq classification(o)$

$preUpdate(clearance(s)) : clearance(s) = LUB(clearance(s), classification(o))$

4.3.2 Role-based Access Control

The $UCON_{preA_0}$ model also can support RBAC in its authorization process. In RBAC96 model [SCFY96], a role is a collection of users and a collection of permissions. The permission is a collection of object-right pairs. In $UCON_{preA_0}$, user-role assignment can be viewed as subject attributes and permission-role assignment as attributes of object and rights. Example 15 shows how RBAC_1 can be viewed in our ABC models. RBAC_1 includes hierarchy in its definition. Only activated roles are used for authorization decision. Here, if there exists an active role ($actRole(s)$) that dominates any of the permission roles ($Prole(o, r)$), a request is allowed. Example 16 and 17 shows examples that include high watermark property. Example 16 does not have a role hierarchy while Example 17 does. In both cases, active roles are updated if other than currently active roles are required for a request. Since the role hierarchy is not a lattice (so LUB is not always defined), Example 17 also has a selection issue. Having an automated high watermark property in RBAC eliminates the least privilege principle that is supported in original models.[6] It also causes a selection problem in case there are multiple roles available for a request. In Example 17, 'UB' denotes

[6]Least Privilege can be supported if role activation is done manually by users.

upper bounds.

Example 15 RBAC$_1$ with activation, $UCON_{preA_0}$:

$P = \{(o, r)\}$

$ROLE$ is a partially ordered set of roles with dominance relation $\geq$

$actRole : S \rightarrow 2^{ROLE}$

$Prole : P \rightarrow 2^{ROLE}$

$ATT(S) = \{actRole\}$

$ATT(O) = \{Prole\}$

$allowed(s, o, r) \Rightarrow \exists role \in actRole(s), \exists role' \in Prole(o, r), role \geq role'$

Example 16 RBAC$_0$ with high watermark property, $UCON_{preA_1}$:

$P = \{(o, r)\}$

$ROLE$ is an unordered set of roles

$srole : S \rightarrow 2^{ROLE}$

$prole : P \rightarrow 2^{ROLE}$

$actRole : S \rightarrow 2^{ROLE}$

$ATT(S) = \{srole, actRole\}$

$ATT(O) = \{prole\}$

$allowed(s, o, r) \Rightarrow srole(s) \cap prole(o, r) \neq \phi$

$preUpdate(actRole(s)) : actRole(s) =$

$$\begin{cases} actRole(s), & \text{if } actRole(s) \cap Prole(o, r) \neq \phi; \\ actRole(s) \cup \lambda(srole(s) \cap prole(o, r)), & \text{if } actRole(s) \cap Prole(o, r) = \phi, \end{cases}$$

where λ is a nondeterministic selection function of an element from a set.

Example 17 RBAC_1 with high watermark property, $UCON_{preA_1}$:

$P = \{(o, r)\}$

$ROLE$ is a partially ordered set of roles with dominance relation $\geq$

$srole : S \rightarrow 2^{ROLE}$

$prole : P \rightarrow 2^{ROLE}$

$actRole : S \rightarrow 2^{ROLE}$

$ATT(S) = \{srole, actRole\}$

$ATT(O) = \{prole\}$

$\overline{ROLE} = \{role \mid \exists role' \in srole(s), \exists role'' \in prole(o, r), role' \geq role \geq role''\}$

$\widetilde{ROLE} = \{role \mid \exists role' \in actRole(s), \exists role'' \in prole(o, r), role' \geq role \geq role''\}$

$\widehat{ROLE} = \{role \mid \exists role' \in actRole(s), \exists role'' \in prole(o, r), role \in \overline{ROLE},$

$role \in UB(role', role'')\}$

$allowed(s, o, r) \Rightarrow \overline{ROLE} \neq \phi$

$preUpdate(actRole(s)) :$

$$actRole(s) = \begin{cases} actRole(s), & \text{if } \widetilde{ROLE} \neq \phi; \\ actRole(s) \cup \lambda(\widehat{ROLE}), & \text{if } \widetilde{ROLE} = \phi, \end{cases}$$

where λ is a nondeterministic selection function of an element from a set.

4.3.3 Discretionary Access Control

Discretionary access control also can be supported in $UCON_{preA}$. DAC policies govern the access of users to an object based on individual or group identities of users and objects. The access modes such as read, write, or execute are granted to a user if the user has privilege to use a specific access mode on an object. Traditionally access matrix has been realized by using either access control list (ACL), or capability list. In $UCON_{preA}$, DAC can be expressed by using either ACLs or capability lists as shown below. In many examples, ACLs and capability lists can be used to achieve same control functions. However in certain cases such as group-based usage number restrictions, ACLs and capability lists have different control functionalities. This distinction has been rarely discussed in previous literature.

Following Examples 18 and 19 utilize ACL and capability list respectively. In Example 18, if any of the subject's group names is listed in a requested object's ACL, the request is allowed. On the other hand, in Example 19, capability list is used to check whether a subject holds any dominant group ID for the requested right. Although these two cases seem to accomplish similar functionality, the functional distinction is much clearer when an update procedure is required. Suppose each group of a subject has a limited number of usage times. In this case, the available number of a subject's usage has to be reduced. To do this we have to select one (or some) of the subject's group(s) and update the current usage number(s) of the selected subject group(s). Here, the number of allowed usage count is assigned to each group of subjects so that the usage can be controlled on a subject group basis. On the other hand, if capability lists are used, the allowed usage count number is assigned to each group of objects and the update of the number is controlled on an object group basis.

Example 18 utilizes ACL and allows multiple group IDs of a subject (assuming no group hierarchy). If one of the subject's group IDs exists in ACL(o), the request is

allowed. Example 19 utilizes capability lists and includes a hierarchy of object group IDs. Here, if there exists a object group IDs that is lower than or equal to a group ID of $CL(s)$, the request is allowed. Example 20 utilizes capability lists and includes usage count k to limit the number of usages for each object group. Here, for the sake of simplicity, we assume there is only one k for each (g, r). In this example, since an object can have multiple group IDs, an update requires a selection of one group ID among the object's group IDs.

Example 18 DAC using ACL w/ multiple group ID, $UCON_{preA_0}$:

G is a set of groups of subject s

$groupId : S \rightarrow 2^G$, many to many mapping

$ACL : O \rightarrow 2^{G \times R}, g$ is authorized to do r to o

$ATT(S) : \{groupId\}$

$ATT(O) : \{ACL\}$

$allowed(s, o, r) \Rightarrow \{(g, r) \mid g \in groupId(s)\} \cap ACL(o) \neq \phi$

Example 19 DAC using Capability List w/ group hierarchy, $UCON_{preA_0}$:

G is a partially ordered set of groups of o

$groupId : O \rightarrow 2^G$

$CL : S \rightarrow 2^{G \times R}, s$ is authorized to do r to g or lower g's

$ATT(S) = \{CL\}$

$ATT(O) = \{groupId\}$

$allowed(s, o, r) \Rightarrow \exists g \in groupId(o), \exists (g', r) \in CL(s), g \leq g'$

Example 20 DAC using Capability List w/ multiple group IDs and usage count, $UCON_{preA_1}$:

G is a set of group names

$groupId : O \rightarrow 2^G$, many to many mapping

$CL : S \rightarrow 2^{G \times R \times K}$, s is authorized to do r to g for k times

$ATT(S) = \{CL\}$

$ATT(O) = \{groupId\}$

$allowed(s, o, r) \Rightarrow \overline{GR} \neq \phi$,

$\overline{GR} = \{(g, r) \mid g \in groupId(o)\} \cap \{(g', r) \mid (g', r, k) \in CL(s), k \geq 1\}$

$\lambda : \overline{GR} \rightarrow G$, a functional mapping to select a group for update

$preUpdate(CL(s)) : k = k - 1, (\lambda(\overline{GR}), r, k) \in CL(s)$

4.3.4 Trust Management

Trust management has mainly focused on authorization decisions of previously known users. Most of the related research has discussed architectural and mechanistic aspects of authorizations. Although the ABC model also covers authorizations of strangers, its

focus is not how to get a credential to authorize a stranger's usage request. Rather, it focuses on usage decision policies and models. In Example 21, a doctor can be mapped to multiple specialities. If a requester holds any speciality, then he can read the object. However if a requester want to write on an object, one of his specialities should be same as the object's speciality.

Example 21 Hospital information usages of doctor by specialty, $UCON_{preA_0}$:

$SPECIALITY$ is a set of medical specialty names

$cert : S \rightarrow 2^{SPECIALITY}$

$groupID : O \rightarrow SPECIALITY$

$ATT(s) : \{cert\}$

$ATT(o) : \{groupID\}$

$allowed(s, o, read) \Rightarrow (cert(s) \neq \phi)$

$allowed(s, o, write) \Rightarrow (cert(s) \neq \phi) \wedge groupID(o) \in cert(s)$

4.3.5 Digital Rights Management

Usage decisions in commercial DRM solutions usually utilize user-defined, application-level, payment-based security policies, and do not include traditional access control policies. Typical examples include pay-per-view, metered payment, membership-based monthly subscriptions, etc. These DRM examples can be realized within our ABC model. In Example 22, a usage request is allowed if a subject holds enough pre-paid credits to use certain rights on specific objects. In this case, the credit is

considered as a subject attribute and the value of the requested usage as an attribute of the object and right.

Example 22 DRM pay-per-use, $UCON_{preA_1}$:

M is a set of money amount

$credit : S \rightarrow M$

$value : O \times R \rightarrow M$

$ATT(s) : \{credit\}$

$ATT(o, r) : \{value\}$

$allowed(s, o, r) \Rightarrow credit(s) \geq value(o, r)$

$preUpdate(credit(s)) : credit(s) = credit(s) - value(o, r)$

In general, payment based authorization requires certain update procedures to resolve usage expenses. In Example 22, a user's credit has been reduced by the value of usages at the time of a request approval. In case of metered payment, post-updates are likely to be required. In Example 23, since a usage on an object holds more than one value, a system has to select a value for update. The system may have to select a value based on subject's membership ranks, sale period, or multiple purchases. Because the selection policies can vary, Example 23 simply utilizes a non-deterministic selection function to describe this functionality.

Example 23 DRM pay-per-use, one credit, multiple values, $UCON_{preA_1}$:

M is an ordered set of money amount

$credit : S \rightarrow M$

$value : O \times R \rightarrow 2^M$

$ATT(s) : \{credit\}$

$ATT(o, r) : \{value\}$

$\overline{M} = \{m \mid m \in value(o, r), m \leq credit(s)\}$, a set of available values for selection.

$allowed(s, o, r) \Rightarrow \exists m \in value(o, r), m \leq credit(s)$

$preUpdate(credit(s)) : credit(s) = credit(s) - \lambda(\overline{M})$,

where λ is a selection function to select a value for update.

In case a user holds more than one credit account, if the sum of credits is more than a value, the request is allowed. Here, some or all of the credit accounts have to be reduced in total by the usage value (Example 24).

Example 24 DRM pay-per-use, multiple credits, one value, $UCON_{preA_1}$:

M is an ordered set of money amount

$credit : S \rightarrow 2^M$

$value : O \times R \rightarrow M$

$ATT(s) : \{credit\}$

$ATT(o, r) : \{value\}$

$\overline{M} = \{\overline{m} \mid \overline{m} \in credit(s)\}$

$\widehat{M} = \{\widehat{m} \mid \sum \widehat{m} = value(o)\}$

$\lambda : \overline{M} \rightarrow \widehat{M}, \overline{m} \geq \widehat{m}$

$allowed(s, o, r) \Rightarrow \sum credit(s) \geq value(o, r)$

$preUpdate(credit(s)) : \forall \overline{m}, \overline{m} = \overline{m} - \lambda(\overline{m})$

Many DRM solutions and studies have included some form of obligations and conditions because of DRM's distributed and payment-based nature. Some objects can only be used at certain locations or time durations. A user may have to provide certain personal information or usage log information for further use. Some DRM related examples for obligations and conditions are presented in previous sections.

4.3.6 Modern Access Control (Healthcare Examples)

Generally, modern access control requires more than authorizations for usage decision. In this section, we show several healthcare information system examples that require authorizations, obligations and conditions. We also show how one example can be expressed in various ways using different decision predicates. Example 25 shows an example that requires an authorization for usage decision. Here, the number of doctor's previous operations is considered as a subject's attribute and used for authorization.

Example 25 A medical doctor (s) can perform (r) an operation (o) only if he has performed operations more than 3 times, $UCON_{preA_1}$:

$ROLE$ is an unordered set of roles

$SPECIALITY$ is a set of medical speciality names

N is a set of subject's total operation numbers

$exp : S \rightarrow N$

$sRole : S \rightarrow 2^{ROLE}$

$sArea : S \rightarrow 2^{SPECIALITY}$

$oArea : O \rightarrow 2^{SPECIALITY}$

$ATT(s) : \{sRole, sArea, exp\}$

$ATT(o) : \{oArea\}$

$allowed(s, o, operate) \Rightarrow$ 'doc' $\in sRole(s), sArea(s) \cap oArea(o) \neq \phi, exp(s) \geq 3$

$preUpdate(exp(s)) : exp(s) = exp(s) + 1$

In Example 26, suppose medical operations can be allowed only when patients agree on a consent form. This requires additional obligation predicate. *obs* is selected based on an object' attribute, *opid*. In this example, obligation actions (*ob*) have to be performed by patient of the requested operation. Authorization is also used for medical speciality verification.

Example 26 A medical doctor can perform an operation only if patients agree on consent form, $UCON_{preA_0preB_0}$:

$ROLE$ is an unordered set of roles

$SPECIALITY$ is a set of medical speciality names

$PATIENTid$ is a set of patients' identification numbers

$sRole : S \rightarrow 2^{ROLE}$

$sArea : S \rightarrow 2^{SPECIALITY}$

$oArea : O \rightarrow 2^{SPECIALITY}$

$spid : S \rightarrow PATIENTid$

$opid : O \rightarrow PATIENTid$

$ATT(s) : \{sArea, spid\}$

$ATT(o) : \{oArea, opid\}$

$OBS = \{s' | \text{`PATIENT'} \in sRole(s')\}$

$OBO = \{consent\}$

$OB = \{agree\}$

$getPreOBL(s, o, operate)$

$= \{(s', consent, agree)\}$ where $s' \in OBS, spid(s') = opid(o)$

$allowed(s, o, operate) \Rightarrow$ 'doctor' $\in sRole(s), sArea(s) \cap oArea(o) \neq \phi$

$allowed(s, o, operate) \Rightarrow preFulfilled(getPreOBL(s, o, operate))$

Suppose there are junior doctors and senior doctors ($sRole(o)$). In case a junior doctor wants to perform operations, the operation is allowed only with the presence of any of his senior doctors. This can be realized by using either authorization, obligation, or condition predicates. Example 27 utilizes condition predicate that checks current

local time and decides whether any of the senior doctors is on-duty at the time of operation request. Authorization is used together with condition predicate to check doctor's speciality. In Example 28, obligation predicate is used to check whether any of senior doctor has agreed to be available. Alternatively, as shown in Example 29, same example can be also realized by using authorization predicate. Here, senior doctor's presence is treated as an attribute of the subject. In Example 27, 28, and 29, 'sDoc' and 'jDoc' are labels for senior doctor and junior doctor, respectively.

Example 27 A junior medical doctor can perform an operation only with the presence of a senior doctor, $UCON_{preA_0 preC_0}$:

$ROLE$ is an unordered set of roles

$SPECIALITY$ is a set of medical speciality names

$DOCid$ is a set of doctors' identification numbers

$curT$ is a current local time, T is a set of time

$sRole : S \rightarrow 2^{ROLE}$

$sArea : S \rightarrow 2^{SPECIALITY}$

$dId : S \rightarrow DOCid$, a functional mapping of subject to a doctor's ID number

$sdId : S \rightarrow 2^{DOCid}$, a functional mapping of subject to a set of senior doctors

$dutyS : S \rightarrow T$, a functional mapping of subject to duty start time

$dutyE : S \rightarrow T$, a functional mapping of subject to duty end time

$oArea : O \rightarrow 2^{SPECIALITY}$

$ATT(s) : \{sRole, sArea, dId, sdId, dutyS, dutyE\}$

$ATT(o) : \{oArea\}$

$getPreCON(s, o, operate)$

$$= \begin{cases} (\exists s', dId(s') \in sdId(s), dutyS(s') \leq curT \leq dutyE(s')), & \text{if } sRole(s) = \text{`jDoc'}; \\ \phi, & \text{if } sRole(s) = \text{`sDoc'}. \end{cases}$$

$allowed(s, o, operate) \Rightarrow$ 'doctor' $\in sRole(s), sArea(s) \cap oArea(o) \neq \phi$

$allowed(s, o, operate) \Rightarrow preCondChecked(getPreCON(s, o, operate))$

Example 28 A junior medical doctor can perform an operation only with the presence of a senior doctor, $UCON_{preA_0preB_0}$:

$ROLE$ is an unordered set of roles

$SPECIALITY$ is a set of medical speciality names

$DOCid$ is a set of doctors' identification numbers

$sRole : S \rightarrow 2^{ROLE}$

$sArea : S \rightarrow 2^{SPECIALITY}$

$dId : S \rightarrow DOCid$, a functional mapping of subject to a doctor's ID number

$sdId : S \rightarrow 2^{DOCid}$, a functional mapping of subject to a set of senior doctors

$oArea : O \rightarrow 2^{SPECIALITY}$

$ATT(s) : \{sRole, sArea, dId, sdId\}$

$ATT(o) : \{oArea\}$

$OBS = \{s' | \text{`sDoc'} \in sRole(s')\}$

$OBO = \{presence\}$

$OB = \{agree\}$

$getPreOBL(s, o, operate)$

$$= \begin{cases} ((s', presence, agree) \text{ where } s' \in OBS, dId(s') \in sdId(s)), & \text{if } sRole(s) = \text{`jDoc'}; \\ \phi, & \text{if } sRole(s) = \text{`sDoc'}. \end{cases}$$

$allowed(s, o, operate) \Rightarrow$ 'doctor' $\in sRole(s), sArea(s) \cap oArea(o) \neq \phi$

$allowed(s, o, operate) \Rightarrow preFulfilled(getPreOBL(s, o, operate))$

Example 29 A junior medical doctor can perform an operation only with the presence of a senior doctor, $UCON_{preA_0}$:

$ROLE$ is an unordered set of roles

$SPECIALITY$ is a set of medical speciality names

$DOCid$ is a set of doctors' identification numbers

$sRole : S \rightarrow 2^{ROLE}$

$sArea : S \rightarrow 2^{SPECIALITY}$

$sdId : S \rightarrow 2^{DOCid}$, a mapping of subject to a set of **on-duty** senior doctors.

$oArea : O \rightarrow 2^{SPECIALITY}$

$ATT(s) : \{sRole, sArea, sdId\}$

$ATT(o) : \{oArea\}$

$allowed(s, o, operate) \Rightarrow$ 'jDoc' $\in sRole(s), sArea(s) \cap oArea(o) \neq \phi, sdId(s) \neq \phi$

ABC model is comprehensive enough to include various access control policies in a single framework. The goal of ABC family model is not to make an air-tight distinction among the detailed models. In ABC model, policies or requirements can be

resolved in multiple ways. Although we have shown many examples, how one actually implements access (or usage) control policies and requirements is not the issue of ABC model. These issues are to be considered at the architecture and mechanism layers, not at the model layer. ABC model provides possible ways to realize or express various policies and requirements in a formal framework. It is this richness and robustness of the expressive power that makes ABC model significant.

4.4 Related Work

Several lines of related research were discussed in the introduction chapter. Of these the policy-based authorization representation and enforcement model of [RN01, RN02] is possibly the closest to the ABC models. This model builds authorizations from objects, rights and conditions. A possible implementation by means of extended access control lists (EACLs) is outlined. Subjects are not explicitly recognized in EACLs but are rather embedded into conditions. Similar to ABC, this model also recognize pre- and mid-conditions. I feel that ABC pre and ongoing decisions are more precisely and systematically defined. In addition to pre- and mid-conditions, the model also identifies "post-conditions" which may have "side-effects" as part of the model to resolve update timing issue. However its definition of post-condition is closer to an implementation level aspect. In ABC model point of view, it may be viewed as a special case of pre-authorization with pre-update model ($UCON_{preA_1}$) implementation. This is largely because of its lack of systematic treatment on mutability issue. At this point the ABC model is more mature and comprehensive.

The term 'obligation' has been used with different meanings in the literature. Damianou et al. introduced the Ponder policy specification language [DDLS01]. Ponder policies consists of authorization, obligation, refrain, and delegation policies. Schaad and Moffett have discussed further on the obligation part of Ponder [SM02]. In both

case, obligations are duties that have to be done independently from users' access requests. For example, software developer of a project A in an organization may have duties to provide weekly progress report to project manager. These duties are given to him not because he has requested certain accesses, but because he has assigned to a software developer role in the organization. Schaad and Moffett argue that obligations require authorizations so the required actions can be performed. By definition, UCON obligation is different from Ponder obligations (or duties). In ABC model, obligations are what subjects have to perform before or during (or even after in case of global obligation) obtaining or exercising usages. If an obligation is required, it just has to be done and does not require any authorization process for obligation fulfillment. Fundamentally, obligations in ABC model are different from duties. Also, ABC model does not include any concept of duties in its model.

The notion of 'provisional authorization' has been introduced in recent literature [KH00, JKS01]. In a narrow definition, provision is what has to be performed prior to the authorization of usage requests. Provision is similar to UCON pre-obligations. Bettini et al. have discussed the notion of 'obligation' [BJWW02]. Here, obligation is what has to be performed after authorization decisions. This is similar to UCON global obligations. Neither has defined a notion for ongoing-obligations which have to be fulfilled continuously or regularly while the requested action is being performed. In the ABC model, obligations are defined and discussed in systematic manner so that they can be used for various situations with finer-grained controls. What really sets ABC apart from other research efforts is its systematic and comprehensive effort to provide a new intellectual foundation for access control. No prior effort has this reach and scope.

In terms of industry trends two ongoing efforts are worth mentioning. ContentGuard's eXtensible rights Markup Language (XrML), evolved from Xerox PARC's DPRL,

has emerged as an OASIS based standard for rights expression languages [Con02, WLD+02]. As defined in its XML schema-based specification version 2, 'grant' consists of four entities called principal, rights, resource, and condition [Con02, WLD+02]. XrML conditions include terms, conditions and obligations. However their definition of terms, conditions and obligations are different from our conditions and obligations and not as precise as ours. Furthermore, while XrML may express various rules and policies for rights, it fails to resolve a transaction-based decision-making process. For example, XrML can express 'student can play a MP3 file 5 times' but assumes usage history of 'play' rights is supported by applications. Hence, it fails to resolve mutable cases such as 'after being played 2 times, now the MP3 file can be played only 3 more times'. Also there is no attempt to express ongoing decision-making. Similar shortcomings can be found also in OASIS eXtensible Access Control Markup Language (XACML) [GM02]. Although authorization in XACML is based on transaction or request, it fails to cover mutable cases and ongoing cases.

4.5 Summary

In this chapter I have introduced the ABC model as a core model for usage control. Usage control encompasses traditional access control, trust management, and digital rights management and goes beyond them in its scope. By unifying these diverse areas in a systematic manner, the ABC model offers a promising approach for the next generation of access control.

I have given a description of the ABC family of models for usage control. The models, and their relationships, are summarized in Figure 4.2. I emphasize that I have only described the "pure" models corresponding to individual points in these figures. In practice I would expect real systems to use composite models which combine several of these together. Space does not permit us to explore the expressive power of combined

models. Nonetheless, I have shown by example that a wide range of policies can be easily expressed in these models.

Chapter 5

UCON ARCHITECTURES

There can be numerous architectural variations for usage control. One way of viewing these variations is based on the location of reference monitor. Reference monitor is one of the most crucial components for usage control architecture that enables decision and enforcement functions. Another way can be based on the existence of payment function. Because of its commercial potential, most commercial DRM solutions mainly focus on payment-based architecture while ignoring situations where payment is not required. Rather than covering all of these variations, this chapter narrows down its scope on usage control architectures for payment-free, client-side reference monitor environment. However this does not mean that every detail of the architectures developed here can be covered by ABC model. In other words, ABC model is the very core of usage control and leaves uncovered many other important and practical aspects such as a composite object that consists of multiple sub-objects. For example, suppose a subject tries to access a composite object that requires different usage rules. If only parts of the requested object are allowed, there should be a customization procedure that generates a 'view' file. This is accommodated in UCON reference monitor though not covered in ABC model.

This chapter starts with detailed discussions of these architectural variations to narrow down the scope. First, payment-based and payment-free type environments are discussed. Next, discussions on reference monitor and its variations are presented.

Then three major factors for usage control architecture are discussed. Based on these three factors I develop eight security architectures for usage control. I also discuss some related mechanisms. In addition, some current COTS solutions are discussed in terms of UCON security architectures to show the commercial availability of UCON security solution architectures.

5.1 Payment-Free vs. Payment-Based

Controlling usage of digital resources can be divided into two types based on payment function: Payment-Based Type (PBT) and Payment-Free Type (PFT). In PBT, a payment function is required in order to access digital information. In PFT, dissemination of digital information does not require payment, but must be controlled nonetheless to satisfy confidentiality or other security requirements. This chapter mainly focuses on security architectures for PFT dissemination. These architectures for PFT dissemination do not necessarily exclude support of payment functions. It may be possible to overlap payment functions onto PFT security solution architectures.

Unlike the Commercial mass-distribution environment, there are situations in which payment function is not required and higher distribution security is the primary concern. For example, in the Intelligence community digital information is often disseminated to organizations in various countries. The White House may wish to distribute a document in digital form to the South Korean government in such a manner that the received digital information is not revealed either intentionally or accidentally, to the North Korean government. Similar situations can exist in the commercial sector. In recent business-to-business (B2B) e-commerce, it is common for a hub organization to distribute information digitally to its several smaller partners. The challenge is to prevent further distribution of the digital information by the small

partners to others. For instance, General Motors could disseminate several different technical descriptions in digital form to different suppliers who provide the specific parts of GM motor vehicles. However, GM would like to prevent the leakage of digital information amongst suppliers regardless of their intention or possession of the digital information.

The characteristics of digital information of the PFT environment differ significantly from characteristics of digital information of the PBT environment. In the latter environment, a small amount of information leakage is acceptable and even desired [Cox96], while this may not be acceptable for the PFT environment. The number of legitimate copies of a single digital item in PBT is typically greater than that of PFT copies. In general, the objective in the PBT environment is to distribute as many copies as possible and to extract payment for each copy. In the PFT environment, it is the distribution itself which needs to be limited. Therefore, solutions and research for Payment-Based mass distribution purposes may not be directly applicable to the PFT environment, i.e. in Intelligence community or B2B Transactions.

In PBT, security breaches of digital assets result directly in financial loss. Redistribution of illegally obtained digital information does not reduce its quality or worth to the consumer. Consequently, digital content providers have put much effort into protecting digital information from unauthorized distribution. However, no systematic study has been done for controlling usage of digital information.

Hence, studies for more generalized security architectures that can provide secure environments for payment-free type usage of digital information should be considered. Identifying generalized security architectures for controlling usage of digital resources is important in order to provide a cornerstone for developing proper usage control solutions that satisfy an organization's requirements for secure and controlled usage

of digital resource, as well as for better understanding the current DRM solutions. Therefore, this chapter focuses on UCON architectures for payment-free type usages.

5.2 Reference Monitor

In architectural point of view, one of the most critical issues in enforcing UCON is the reference monitor. The reference monitor has been discussed extensively in access control community and is a core concept that provides control mechanisms on access to or usage of digital information. Reference monitor associates decision policies and rules for control of access to digital objects. It is always running and tamper resistant. Subjects can access digital objects only through the reference monitor. In this section, I discuss a conceptual structure of UCON's reference monitor and compare the differences from traditional reference monitor. Also, I discuss some architectural variations of UCON systems based on the utilization of reference monitors.

5.2.1 Structure of Reference Monitor

ISO has published a standard for access control framework [ISO/IEC 10181-3] that defines reference monitor and trusted computing base [iso96]. According to the standard, reference monitor consists of two facilities; access control enforcement facility (AEF) and access control decision facility (ADF). Every request is intercepted by AEF that asks an ADF for a decision of the request approval. ADF returns either 'yes' or 'no' as appropriate. Reference monitor is a part of trusted computing base, always running, temper-resistant, and cannot be bypassed.

UCON reference monitor is similar but different in detail from traditional reference monitor of ISO's access control framework. Figure 5.1 shows the conceptual structure of UCON reference monitors. UCON reference monitor consists of *Usage Decision Facility (UDF)* and *Usage Enforcement Facility (UEF)*. Each facility includes several

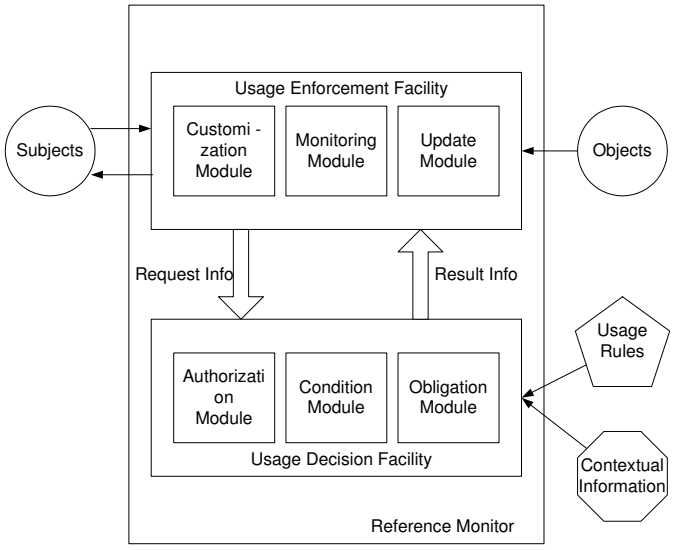


Figure 5.1: Conceptual Structure for UCON Reference Monitor

functional modules. UDF includes conditions and obligations decision modules as well as authorization module. Authorization module takes care of a process similar to traditional authorization process. It utilizes subject and object information (attributes) and usage rules to check whether the request is allowed or not. It may return yes or no. It may return metadata information of authorized portion of requested digital objects along with allowed rights. Then, this metadata information is used for customization of requested digital objects by customization module of UEF. Condition module decides whether the conditional requirements for the authorized requests are satisfied or not by using usage rules and contextual information (e.g., current time, IP address, etc). It may limit rendering devices (e.g., CPU-ID, IP address), rendering time (e.g., business hour, on-duty), etc. Obligation module decides whether certain obligations have to be performed or not before or during the requested usage has been performed. If there exists any obligation that has to be performed, this must be monitored by monitoring module and the result has to be

resolved by update module in UEF. Note that usage decision rules may or may not be hardwired into decision facility. Those rules can come along with related digital information or independently [PSS00]. Utilization of these modules largely rely on the target application systems' requirements.

5.2.2 Architectural Classification

Based on the location of reference monitor, there can be *Server-side Reference Monitor (SRM)*, and *Client-side Reference Monitor (CRM)*. Here, server is an entity that provides a digital object and client is an entity that receives and uses the digital object. Like a traditional reference monitor, a SRM resides within server system environment and mediates all access to digital objects. On the other hand, a CRM resides in the client system environment and controls access to and usage of digital objects on behalf of a server system. SRM and CRM can coexist within a system. The trustworthiness of CRM is considered relatively lower than that of SRM. Therefore, the main concern here is how reliable and trustworthy the CRM is. In fact, if the client-side computing device is fully functional and general-purpose, CRM is likely to be manipulated with relatively less effort. Therefore, CRM is more suitable to applications with less assurance requirements. This may be improved by using tamper-resistant add-on hardware devices such as dongles, smartcards, etc. On the other hand, if the client device is limited in its functionality and dedicated to specific purposes such as e-book reader or DVD player, CRM is relatively secure from unauthorized manipulations so applications with relatively high assurance requirements are more suitable. After all, the implementation of reference monitors largely depends on business models and their application requirements. For real world implementations, the chances are that both CRM and SRM are likely to be used for better functionality and security. In the following subsections these SRM-only, CRM-only,

and SRM & CRM architectures are briefly discussed.

SRM-Only Architecture

A system with SRM-only facilitates a central means to control subjects' access to and usage of digital information objects. A subject can be either within same organization/network area or outside this area. In this environment a digital object may or may not be stored in client-side non-volatile storage. If the digital object is allowed to reside in client-side non-volatile storage, it means the saved client copy of the digital object is no longer UCON's target object and doesn't have to be controlled. It can be used and changed freely at client-side. For example, an on-line bank statement can be saved at a customer's local machine for his records and the server system (bank) doesn't care about customer's copy as long as the bank keeps original account information safe. However if the content of digital information itself has to be protected and controlled centrally, the digital information must remain at server-side storage and never be allowed to be stored in cleartext on client-side non-volatile storage. Traditional access control and trust management mainly utilize this kind of system.

CRM-Only Architecture

In a system with CRM-only environment, no reference monitor exists on server-side system. Rather, a reference monitor exists at the client system for controlling usage of disseminated digital information. In this environment digital objects can be stored either centrally or locally. The usage of digital objects saved at the client-side is still under the control of CRM in lieu of the server. Since there exists no SRM, a digital object cannot be customized for specific users for distribution. Hence, this system is likely to be suitable for B2C mass distribution environments such as a e-book systems or MP3 music file distributions. However this doesn't mean that every user will have same usage rights. Distributed digital objects are associated with certain usage rules

and users have to prove they have sufficient credentials to exercise certain rights on the objects. At this point users may be limited to perform certain rights on the object under certain conditions such as a specific device identity.

Digital rights management solutions mainly utilize CRM in their systems. In real world implementation, CRM is likely to be embedded within application software where digital objects can be rendered. One example is Acrobat Reader with "Webbuy" plug-in. Webbuy functions as a CRM. Digitally encapsulated PDF files can be viewed through Acrobat Reader with Webbuy. Webbuy controls access to the contents based on a valid license called Voucher. A Voucher may include a specific CPU-ID to restrict rendering devices.

SRM & CRM Architecture

By having SRM in addition to CRM, this architecture can provide two-tier control. SRM may be used for distribution related control while CRM can be used for a finer-grained control on usages. For instance, in SRM, digital objects can be pre-customized for distribution and the distributed, pre-customized digital objects can be further controlled and customized for clients' usages by CRM. As a result, server can reduce or eliminate unnecessary exposure of digital objects that do not have to be distributed. Suppose we have an intelligence system with this architecture. If an unclassified user requests certain digital information that includes some secret information as well, SRM can pre-customize the requested objects before distribution so the distributed version of the objects don't include any secret information. Any finer-control on the distributed objects can be done by CRM at client side. In real world applications, functional specifications of UCON reference monitor can be divided into SRM and CRM in various ways based on the system's functional and security requirements.

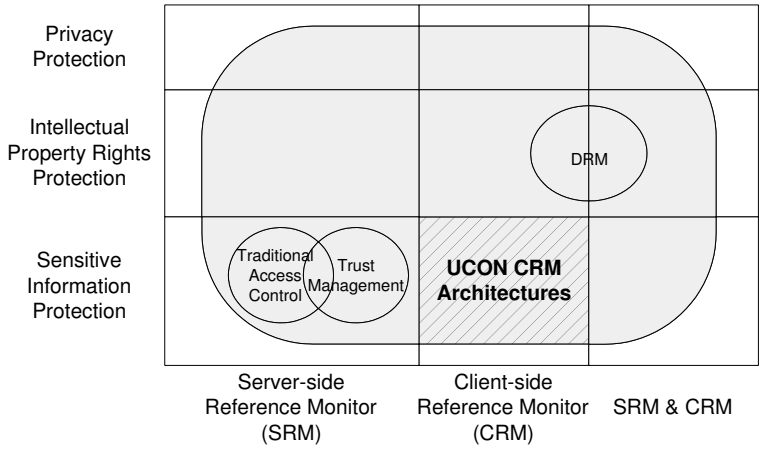


Figure 5.2: Architectural Scope

5.2.3 Architectural Scope

In this chapter, I narrow down my focus on CRM only architectures so I can articulate in detail how CRM-based UCON architectures can be realized. CRM only architecture is chosen since it is one of the most significant architectural aspects of today's DRM system. Figure 5.2 shows this area of coverage as diagonally shaded.

Next, three major factors that are crucial for CRM-based architectures are discussed. Then several security architectures are identified based on those factors. I further discuss detailed characteristics and show how COTS solutions can be viewed in this context.

5.3 Three factors of CRM security architectures

There are three major factors that distinguish CRM security architectures. They are virtual machine (VM), control set (CS), and distribution style. The combination of use of each factor results in different security architectures. For better understanding of these architectures, first we have to understand these three factors.

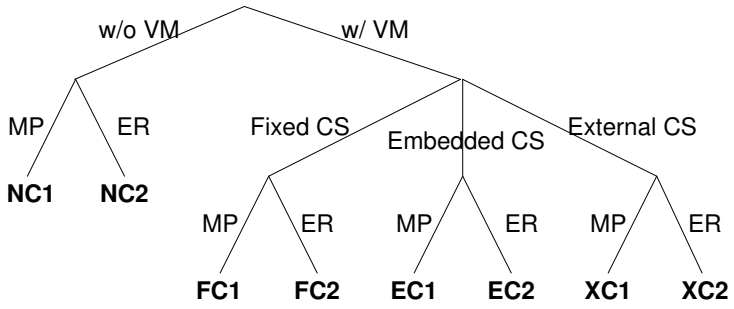
The virtual machine is software that runs on top of vulnerable computing environment and utilizes reference monitor to provide the means to control and manage access and usage of digital information. For instance, Adobe Acrobat Reader and Web Buy plug-in can be considered as a virtual machine though Acrobat Reader alone has very limited control functions to control access to digital content. The existence of a virtual machine on the client side is one of the most significant factors of the architecture, and it provides the foundation for client-side control. It also implies the need for specialized client software.

The control set is a list of access rights and usage rules that is used by the virtual machine to control a recipient's access and usage of a digital object. There can be three styles of control sets. A fixed control set is hardwired into the virtual machine and applies uniformly to all digital objects and all users. An embedded control set is inextricably bound to each digital object and is carried along with it. An external control set is separate and independent from the digital object (and can be transported separately or together with the object). Embedded and external control sets can apply different controls to each object and each user.

Message push (MP) and external repository (ER) are two possible distribution styles. In message push style, digital information is sent to each recipient. In external repository style, each recipient obtains the digital information from a dissemination server on the network.

5.4 Architecture Taxonomy

In this section eight different security architectures for PFT usage control are identified, based on previously identified three factors. Each architecture has different security implications. The classification of these architectures has been done somewhat exhaustively to cover all possibilities. Each architecture provides different advantages



VM: Virtual Machine
MP: Message Push
ER: External Repository
CS: Control Set

NC1: No control architecture w/ MP
NC2: No control architecture w/ ER
FC1: Fixed control architecture w/ MP
FC2: Fixed control architecture w/ ER
EC1: Embedded control architecture w/ MP
EC2: Embedded control architecture w/ ER
XC1: External control architecture w/ MP
XC2: External control architecture w/ ER

Figure 5.3: Security Architecture Taxonomy

and disadvantages. This section defines these architectures based on the three factors and discusses their merits and demerits.

Figure 5.3 illustrates the taxonomy of these eight architectures. The term "no control" is used to mean the lack of a virtual machine. The term "fixed control" means that the only control is that which is fixed in the virtual machine. The terms "embedded" and "external" control mean variable control as discussed above. These may coexist with fixed controls in the virtual machine.

Each of these architectures is described in the following sub-sections. Non-Encapsulated digital information dissemination architectures are described here as basic architectures for comparison purpose. Encapsulated digital information dissemination archi-

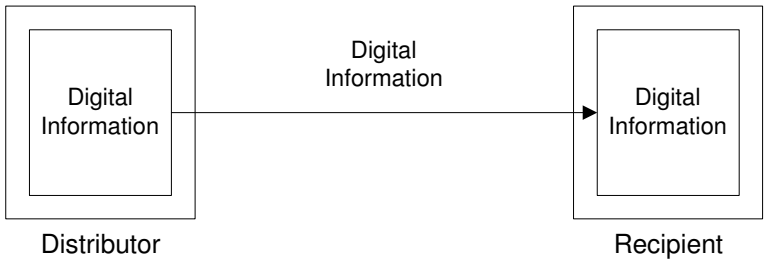


Figure 5.4: No Control Architecture with Message Push (NC1)

tectures are our main concern. Even though I have distinguished and defined these architectures, there can be real world security solutions that combine more than one security architecture. The diagrams for each of the architectures do not explicitly show encryption mechanisms or watermarking mechanisms.

5.4.1 Non-Encapsulated Architectures

No Control Architecture w/ Message Push (NC1)

No control architecture with message push (NC1) is a classic architecture for digital information dissemination. In this architecture, the distributor of digital information directly sends a copy of the digital content to each recipient. Each recipient stores the copy at his/or her storage device.

After distribution is done, the distributor has no direct means to control the distributed digital information, so the likelihood of deliberate re-dissemination or theft is increased. The recipient can either keep the digital information or delete it from his or her storage device. After the digital information is deleted, there is no way for recipient to access the digital information. To access the saved information from multiple computers, the recipient needs to transport the information.

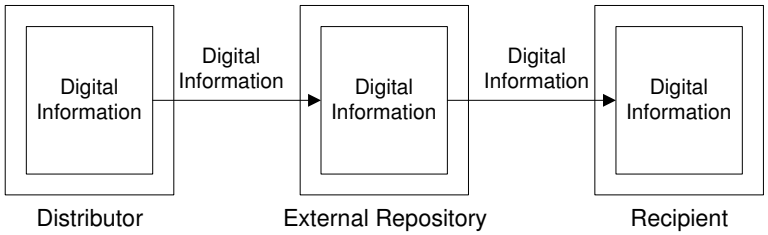


Figure 5.5: No Control Architecture with External Repository (NC2)

No Control Architecture w/ External Repository (NC2)

No control architecture with external repository (NC2) has similar features to NC1, except that in NC2 digital information is sent to an external repository server for distribution. A recipient must connect to the external repository to access and retrieve the digital information. Once a recipient has received the digital information, the distributor has no means to control or manage access rights or usage rights. Since this architecture does not have a virtual machine, there is no control set.

5.4.2 Encapsulated Architectures

Fixed Control Architecture w/ Message Push (FC1)

In the fixed control architecture with message-push (FC1), the control set is included in virtual machine. Since the control set is encoded into a virtual machine, the control set cannot be changed after the distribution of the virtual machine. Digital information is encapsulated in a digital container that does not allow the recipient to access digital information without using the virtual machine. Access is based on the control set encoded inside the virtual machine. This control set will contain rules which the virtual machine enforces, such as preventing storage of the cleartext digital information on the recipient's non-volatile storage. Re-dissemination of the digital container in this case would be accessible only by someone who has the virtual

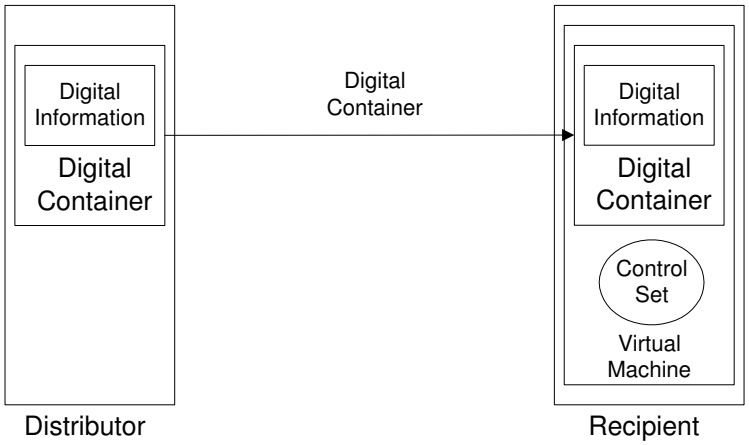


Figure 5.6: Fixed Control Architecture with Message Push (FC1)

machine and meets control set rules.

This architecture has the message-push distribution style. Each recipient must maintain the received digital container on his or her storage device for further access to it.

Fixed Control Architecture w/ External Repository (FC2)

The Fixed control architecture with external repository (FC2) has basically same characteristics as FC1 except for the distribution style. In this architecture, digital information encapsulated within a digital container is sent to external repository for distribution. A recipient must connect to the external repository to access or download the digital container. The recipient can access digital information encapsulated within the digital container through a virtual machine using the control set encoded in the virtual machine. In general, architectures based on external repositories facilitate access to the information by a single recipient from multiple computers.

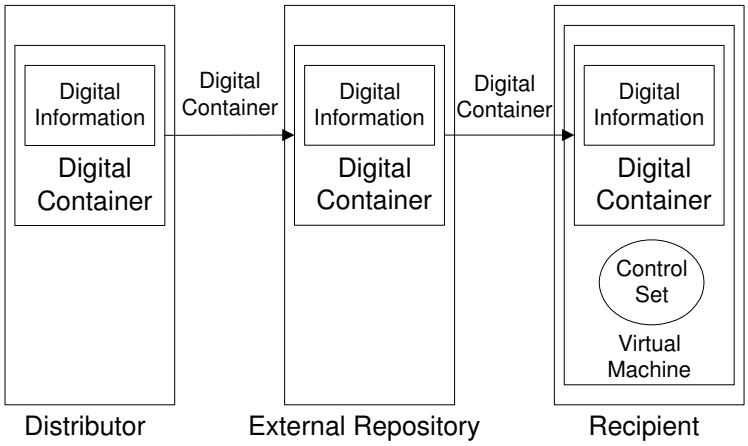


Figure 5.7: Fixed Control Architecture with External Repository (FC2)

Embedded Control Architecture w/ Message Push (EC1)

In the embedded control architecture with message-push (EC1), the control set is embedded in the digital information and always comes with the digital information within its digital container. The distributed digital information will be controlled based only on the pre-set access rights and usage rules on the digital information. Because there is no external control center function, the distributor cannot change the control set of the distributed digital information. In this and all subsequent architectures, the control set applied to the digital container may be in addition to a fixed control set in the virtual machine.

After a recipient has received a digital container, he or she can access the digital information without any network connection, if he or she has proper access rights. This means that there is no additional access control (i.e. changing access rights after dissemination) for the distributed digital information. In addition, there can be only pre-set revocation. In other words, there is no revocation function available, which can be applied after distribution of the digital container.

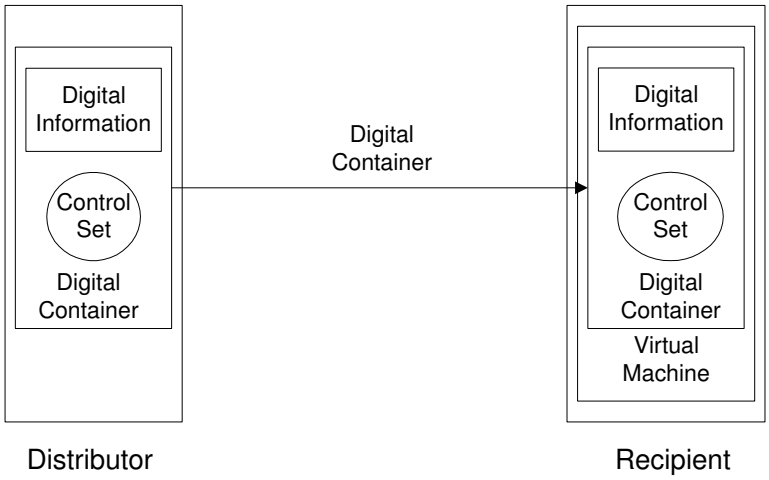


Figure 5.8: Embedded Control Architecture with Message Push (EC1)

In this architecture the control set can prevent storage of cleartext digital information on the recipient's non-volatile storage. Storage of the digital container by the recipient would be required for future access. However, the control set can prevent someone else from opening the digital container if it is re-disseminated to them.

Embedded Control Architecture w/ External Repository (EC2)

The embedded control architecture with external repository (EC2) also has fundamentally the same features as EC1, except for its distribution style. In this architecture, digital information is encapsulated within a digital container and sent to the external repository server. In addition to the controls that can be imposed in EC1, in this case the control set can further prevent the recipient from storing the digital container on the recipient's non-volatile storage. If the encapsulated digital container cannot be locally stored then this architecture enables the distributors to revoke a previously granted access.

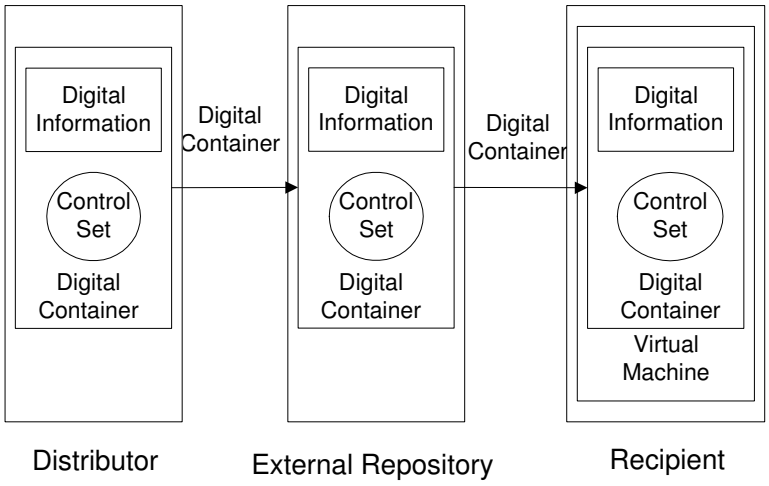


Figure 5.9: Embedded Control Architecture with External Repository (EC2)

External Control Architecture w/ Message Push (XC1)

In the external control architecture with message push (XC1), digital information is freely available in the form of digital container, but only those who have valid access rights can open the digital information in it. The recipient gets access rights by connecting to the control center. These access rights can be encapsulated in a digital container with or without the original digital information. This means that access rights can be distributed independently and that access rights can be encapsulated in a digital container with other digital contents that are not related to the access rights. In this architecture, distributors can control and manage recipients' access rights on the digital information, including causing the revocation of previously granted rights. Both senders and recipients must trust the control center.

There are two options based on the usage rule information. In first case, every time a recipient wants to open an item of distributed information, he or she must access the control center. In second case, the recipient does not have to access the control center every time, but he or she should access the control center from time to time to

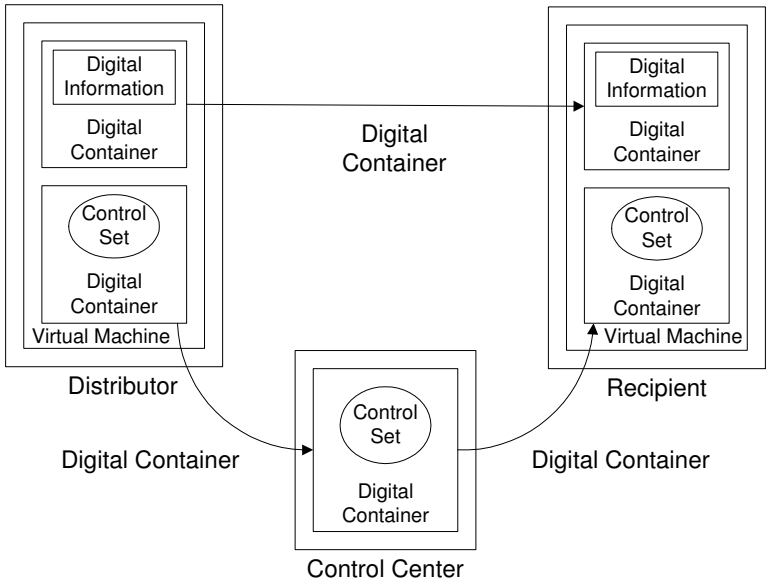


Figure 5.10: External Control Architecture with Message Push (XC1)

get the proper usage rights, based on the usage rights policy. The recipient may have to access the control center based on the usage time, the number of access, or fixed time period. In latter case, there can be a one-time only connection to the control center if the recipient receives an unlimited (no expiration) set of access rights that do not require any further connection. The distributors should decide very carefully before distributing any control set that does not require any further connection, lest they forfeit their power to revoke access.

External Control Architecture w/ External Repository (XC2)

The external control architecture with external repository (XC2) has primarily the same characteristics as XC1 except that it includes an external repository. Encapsulated digital information is stored at the external repository for distribution. The information may or may not be freely available. This architecture can provide separation of content and access rights. This architecture may have four possible options

based on the usage rule information. Note that there are two digital containers in this architecture. In reference to Figure 5.11, the digital container at the top carries the digital information, whereas the one at the bottom only carries a control set. Each of these digital containers may or may not be storable by the recipient. This gives us four combinations as follows.

In first case, both the encapsulated digital information and the encapsulated control set can be stored on a recipient's local storage device. In this case, a recipient does not have to connect to either an external repository or a control center every time he or she wants to access the digital information. The recipient may have to connect to the control center from time to time to renew the control set (as explained in XC1). Alternately, only a one-time connection to the control center is required for the recipient to access the digital information thereafter.

In the second case, a digital container that includes digital information is freely available, but the control set digital container cannot be locally stored. In this case, a recipient can save the encapsulated digital information in local storage and does not have to download it every time he or she wants to access it. However, the recipient must always connect to the control center to get the control set that is required to access the information. This case can be very useful when the size of the digital information is large.

In the third case, the encapsulated digital information cannot be locally stored, but the encapsulated control set can be stored. Finally, in the fourth case neither digital container is locally storable. These last two cases allow greater control over information dissemination. For example, the digital information can be completely withdrawn.

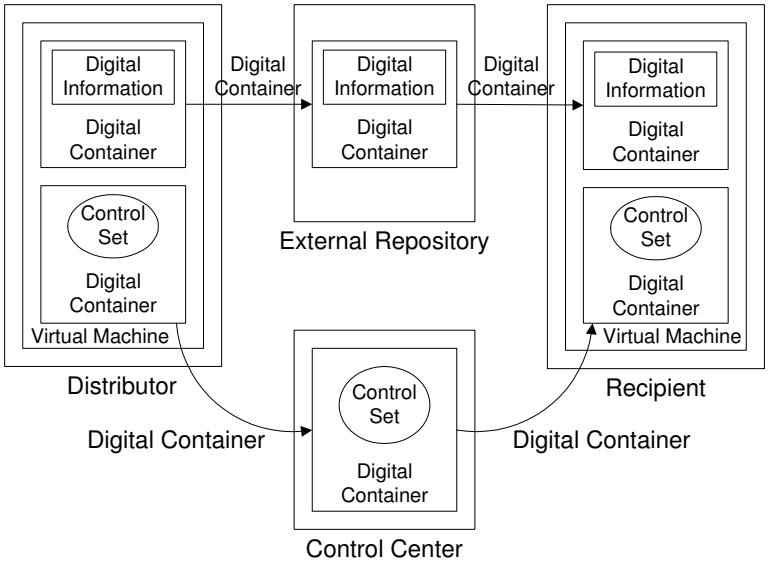


Figure 5.11: External Control Architecture with External Repository (XC2)

5.5 Related Mechanisms

Mechanisms are sets of technologies that support the solution architectures. Implementation of each of the architectures may require a different mix of mechanisms. In this section I present a list of mechanisms that are potentially useful in achieving the security objectives and goals. Use-control mechanisms cannot be applied to NC1 and NC2 architectures. Cryptographic techniques for secure transmission and integrity protection are not included here, but would be used as appropriate.

5.5.1 Watermarking Mechanisms

Digital watermark, or fingerprint is used to mark the identity of the objects (digital information) with information such as the author's name, recipient's name, distribution date, or usage rights. This is done to identify, rather than to protect the digital information from unauthorized access. Digital watermarking can thus provide a tracking capability to illicit distribution of digital information [KK00, Zha97]. In

usage control, digital watermarking technologies are required to enable tracking of disseminated digital information. The detailed description of the characteristics of digital watermarking technologies and linguistic techniques are out of the scope of this thesis. Watermarking mechanisms can be implemented into all of the security solution architectures presented in this chapter.

Digital information can be in several formats such as text, image, audio, and video format. Watermarking technologies are dependent mainly on the type of digital information where the watermarks are to be stored. Each of image, video, audio, and text content needs different watermarking technologies.

The size range of digital information can vary widely, but the size needs to be large enough to facilitate watermarking. If this cannot be guaranteed, padding technologies may be needed. This issue is important because if the typical size of a type of digital information is too small (for example, small text email messages), there might not be any means to store watermarks in it, and then we cannot implement watermarking technology in our security solutions. The size of digital information also influences the security architectures. If the size of digital information is too big, downloading may not be a good way to access the digital information.

For tracking purpose especially, each copy of the originally disseminated digital information needs unique watermarking information (a "fingerprint") so as to identify the sender's and receiver's identities. Embedding different watermarking information in each copy of the originally disseminated digital content, however, is not yet realistic for cases of mass dissemination [Dwo99].

5.5.2 Use-Control Technologies

The Use-Control mechanism is originally based on the superdistribution concept. Superdistribution is a concept that electronic information is available freely, but access

to the information is controlled. In the use-control mechanism, digital information is encapsulated into a cryptographically protected electronic container called a Digital Container. This encapsulated digital information is only accessible by using special application software called a Virtual Machine, with approved access rights that are stored in a Control Set. This mechanism can be applied to all architectures except NC1, NC2.

Virtual Machines on recipients' computers

In the DVD industry, to prevent illicit copying and distributions, several security features have been developed (e.g. regional restriction, copy control restriction). These features are embedded in DVD players. Each DVD title is burned with one or more security configurations based on these features. Because of these security features, a DVD title can only be played or copied within its allowed restriction boundary.

Similarly, we can use a secure and tamper-resistant virtual machine on top of a vulnerable computing environment such as PCs. So, digital information can be only accessible within the virtual machine. By using a virtual machine, we can restrict the access privileges. For instance, we can disable the print function, save function, and save-as function within the virtual machine. Virtual machine mechanisms that reside on recipient's computer can be implemented in all architectures except NC1, NC2.

Digital Container

The digital container [SBW95, Kap96] is a key feature of use-control technologies. A digital container is a tamper resistant electronic envelope that is designed to protect digital information and to control usage by wrapping it up with cryptographic mechanisms. A digital container can contain digital information and control sets. A control set is a collection of usage rules and rights information. Control sets can be

encapsulated in a digital container with or without digital information. When a recipient tries to access digital information, a virtual machine in the recipient's system will check the control set to verify that the recipient has the adequate usage rights.

Control Center

In general, a control center exists for controlling and managing the access rights, usage rules, and even usage history. A control center holds security policies (control sets) that govern usage of digital information and a database of senders and recipients. Generally, the purpose of the control center is to provide access rights on digital container to authorized users, so users can access the digital information. To achieve this, client application software (the virtual machine) will check the control set in a digital container or virtual machine, and if necessary, it will communicate with the control center for additional information such as granting access rights to certain digital information. In the commercial world, the control center can be also responsible for payment functions, access to the digital information can be granted/revoked based on payment.

5.6 Discussion

In this chapter I have identified several possible security architectures for controlling the dissemination of digital information and tracking its re-dissemination, along with some required or related mechanisms to enforce the security architectures. In this section I analyze these architectures and give the findings of the study.

5.6.1 Solution Approaches

This chapter has focused on two major security objectives: controlling dissemination of digital information and tracking its re-dissemination. The fundamental ideas of

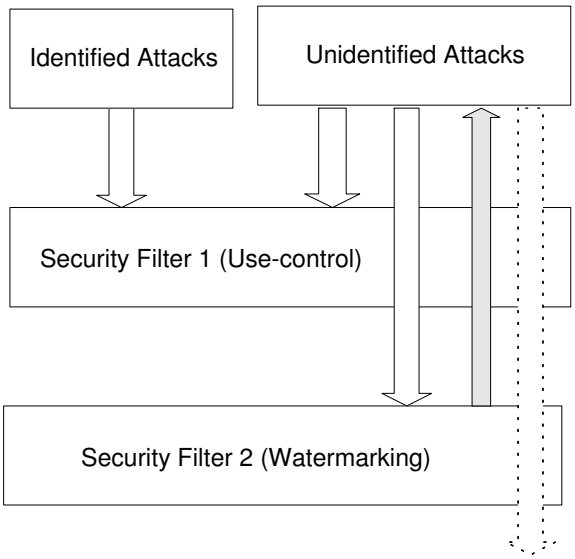


Figure 5.12: Security Attacks and Protections

these security objectives are as follows.

In a security perspective, there can be two types of attacks: identified attacks (known) and unidentified attacks (unknown). Figure 5.12 shows a logical diagram for security attacks and protections. By implementing use-control mechanisms in our security architectures, UCON architectures can protect digital information from these attacks. these solution architectures with use-control mechanisms can protect known attacks and some unknown attacks. However, as shown in Figure 5.12, there can still be some other unknown attacks. These attacks are likely to break through the architectures and thus access the digital information. If this cannot be avoided, there should be well-defined methods to trace the attackers (and hence the watermarking techniques).

Security architectures with use-control mechanisms do not have any tracking features per se. However, in addition to use-control, by implementing watermarking mechanisms into the security architectures, we can achieve reasonable tracking methods. In Figure 5.12, the gray arrow shows tracking action using watermarking technolo-

gies. However, current watermarking technologies are still premature to guarantee the tracking of dissemination and re-dissemination of digital information. If the attackers have tampered the watermarked digital documents, the likelihood of successful tracking of watermarked information will be significantly reduced. The dashed line in Figure 5.12 shows those attacks that subvert watermark tracking.

In this chapter, I have defined security architectures, which can include use-control mechanisms for protection of disseminated digital information (security filter 1), and also which can include watermarking mechanisms for tracking methods (security filter 2). Also, I have identified related mechanisms that can be implemented into the security architectures.

5.6.2 Characteristics of Security Architectures

I have proposed eight security architectures for UCON dissemination architecture. These security architectures have different characteristics. These characteristics are important features for choice of a security solution in a particular context. Table 5.1 shows security and functional characteristics of the security architectures. These characteristics can be merits, demerits, limitations, or requirements of the architectures, depending on the environment in which the architectures are deployed.

5.6.3 Available COTS Solutions for the Architectures

Table 5.2 shows the currently available COTS solutions which belong to one of the UCON security architectures.

For Example, Adobe PDF Merchant and Acrobat Reader (v4.05) with Web Buy plug-in belong to the first case of XC2. PDF Merchant generates a cryptographically encapsulated PDF file and a Voucher file. The encapsulated PDF only can be accessed through Web Buy plug-in. Acrobat Reader with Web Buy is Virtual Machine (VM) in

Table 5.1: Characteristics of Architectures

	Characteristics	NC1	NC2	FC1	FC2	EC1	EC2	XC1	XC2
C1	Disseminator can control access and usage of disseminated digital information			Y	Y	Y	Y	Y	Y
C2	Disseminator can change recipients' access rights after dissemination						Y	Y	Y
C3	Re-disseminated digit al information can be protected			Y	Y	Y	Y	Y	Y
C4	Special client software (virtual machine) is vulnerable to attacks			Y	Y	Y	Y	Y	Y
C5	Tracking re -disseminated digital information is possible	Y	Y	Y	Y	Y	Y	Y	Y
C6	Disseminated digital container is reusable f or other recipients by re-dissemination							Y	Y
C7	Digital information does not have to be on recipient's storage		Y		Y		Y		Y
C8	Digital information can be accessible from any machine if it is connected to network		Y		Y		Y		Y
C9	Recipient should ca rry digital information to access it from multiple machines	Y		Y		Y		Y	
C10	Special client software (virtual machine) is required			Y	Y	Y	Y	Y	Y
C11	In case of large digital information, download time can be significantly costly		Y		Y		Y		Y
C12	Every acc ess to digital information requires network connection.								
C13	The architecture can be supported without network connection	Y		Y		Y			
C14	Control center trusted by both distributors and recipients is mandatory							Y	Y

Note: C1 ~ C5: Security characteristics, C6 ~ C14: Functional characteristics

Table 5.2: Architecture of COTS Solutions

Solution	Organization	NC1	NC2	FC1	FC2	EC1	EC2	XC1	XC2
PDF Merchant & WebBuy	Adobe								X
PageVault	Authentica							X	
SoftSEAL	Breaker Technologies								X
Confidential Courier	Digital Delivery, Inc.					X			
DocSPACE	DocSPACE Co.		X						
CIPRESS	Fraunhofer Institute for Computer Graphics & Mitsubishi Co.								X
Cryptolope	IBM							X	
InTether	Infraworks Co.					X			
InterTrust	InterTrust Technologies Co.							X	
RightMarket	RightMarket.com Inc.							X	

UCON security architectures. Voucher file is Control Set (CS) in the architectures. It grants the right to access the PDF. Both files can be stored at local storage. Therefore there is no network connection required every time recipients want to access digital content. It also provides an option for binding content to CPU ID, storage device ID, network ID, e-mail address, or time, so the recipients only can access within certain environments such as a specific hardware or time period.

Some of the architectures have not been used in any COTS solution because of their different security characteristics and functional characteristics.

5.7 Summary

In this chapter, I have first identified the scope of UCON architectures that is covered here. Then I have identified eight security architectures for usage control. Each architecture's main characteristics, merits, and demerits also have been discussed.

I also described some related mechanisms such as watermarking technologies, and use-control technologies that can be implemented in these security architectures. In addition, I have related these security architectures to COTS solutions to show commercial availability of the security architectures.

The study performed in this chapter is the first systematic study of this topic. In particular, the architectures I have identified have not been previously defined in this manner in the literature. Nevertheless, this architectural approach is fundamentally a starting point for the study of UCON architectures. It provides the basis for future research and development for UCON architecture. Further research on the architectures and mechanisms will lead to practical solutions for usage control.

Chapter 6

UCON APPLICATIONS

So far, I have discussed UCON core models and security architectures. This chapter discusses two applications based on these core models and architectures. However the case studies discussed in this chapter would require further studies for practical applications. This chapter only presents some potential approaches that are likely to provide valuable directions for further extensions. One that will be crucial for the success of usage control is the management issue of UCON that deals with administrative aspects of provider and identifiee subject parties as well as consumer subject and their relationships. Another important issue is how to control re-dissemination of disseminated digital objects. To address this issue, I borrow one of traditional access control policy called originator control also known as ORCON and show how ORCON can be used in UCON for re-dissemination control.

6.1 UCON Management and Examples

This section consists of three subsections. First administrative UCON is discussed that includes provider subject and identifiee subject parties as well as consumer subject party. Then two examples for privacy sensitive and privacy non-sensitive cases are presented. Privacy sensitive example includes all three subject sides while privacy non-sensitive case includes only provider and consumer subject parties.

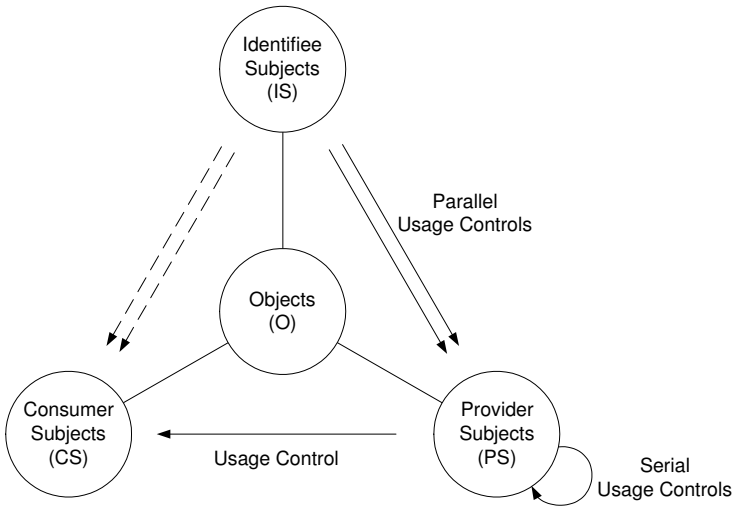


Figure 6.1: Administrative UCON Triangle

6.1.1 Administrative UCON

In the ABC model, I have mainly focused on usages of consumer subjects on objects while ignoring provider and identifiee subjects. In this section I assume that usage rules such as authorization, obligation, and condition rules are already provided. As mentioned earlier, in UCON, subjects can be either consumer, provider, or identifiee. Each subject party has close relationships with other parties. One party may influence other parties' usage decisions. Each party holds its own rights on objects. Exercising rights on an object may require certain obligations that have to be fulfilled before, during or after the rights are exercised. Fulfillment of these obligations may create other objects (called derivative objects) that have to be protected and controlled from usages. This series of relationships has to be resolved seamlessly in UCON as an administrative issue. Here, I discuss some fundamental issues of UCON administration briefly. I believe that the further work on administrative UCON is crucial for the success of usage control.

Figure 6.1 shows an administrative triangle for usage control decisions. Here, a consumer subject is an end-user who is the last beneficiary of an object content in a supply chain. If allowed by a provider, a consumer can hand over the object to another consumer and can control usage of the new consumer. In this case, the original consumer becomes a provider of the object to new consumer. Note that this is different from that a consumer passes an object or a copy of an object to another consumer on behalf of the previous provider of the object while the previous provider controls usage of the disseminated object. Normally, a consumer's usage on an object is likely to be controlled by a single provider. Although there can be multiple providers who actually provide same object copies to a consumer, these copies are considered as separate objects and may have different control policies. If a provider is not an originator of an object, the provider's ability to control consumers's usages on the object is likely to be limited by another provider. If an object o_1 includes other objects o_2 and o_3, a provider subject s_1 of o_1 is considered as a consumer of the included objects o_2 and o_3 and the o_1 as a separate object. In this case, s_1's ability on usages of o_1 is also limited by the providers of o_2 and o_3. In Figure 6.1, this chain of usage controls is denoted as 'serial usage control'. Unlike provider subjects, there exists no control chain of identifiee subjects. Identifiee subjects are subjects whose individually identifiable information is included within an object, therefore hold certain rights to control usages on the object. Credit card information or DNA information are some of the examples of individually identifiable information. The usages of an object that includes these privacy-related information of multiple subjects are controlled by the identifiee subjects. Such multiple controls on usages are denoted as 'parallel usage control'. In general, identifiee subjects are likely to limit provider's usages on the object to control consumer's usages (dotted arrows) on their privacy-related information.

As a summary, UCON has to be viewed as a comprehensive approach to protecting

and controlling usages of three subject parties and their relationships and influences on each other. In today's dynamic, distributed digital environment, traditional one-way control no longer provides adequate trustworthiness. Eventually, unlike previous one-way (from provider to consumer) approaches, control decision of UCON has to be multi-directional for mutual controls and privacy protections. I believe these issues are no longer just technical matters. Business commitment and legal and social support are also crucial for the success of usage control.

6.1.2 DRM and Healthcare Applications in UCON Administration

By distinguishing subject parties, UCON emphasizes relationships between subjects and objects and between subjects themselves. This distinction is shown in figure 6.2 and 6.3. Figure 6.2 is a UCON diagram for privacy non-sensitive objects and Figure 6.3 is for privacy sensitive objects. The UCON model for privacy sensitive objects includes an additional subject called identifiee and relevant rights. Figure 6.2 and 6.3 are based on the following legend.

PNO: Privacy Non-sensitive Object

PSO: Privacy Sensitive Object

Cx: Consumer x

Px: Provider x

Ix: Identifiee x

yR: y Rights

yA: y Authorization

yC: y Condition

yB: y oBligation

where $x = \{x|R, A, C, B\}, y = \{y|C, P, I\}$

I will use two examples and demonstrate how UCON models can be applied for privacy non-sensitive and privacy sensitive digital information. One simple example is a popular MP3 music file distribution. This example can be explained with Figure 6.2 that has provider and consumer subjects sides. Suppose a music composer (say Bob) wants to sell his new song through a distributor, and a buyer (say Alice) wants to buy the song from the distributor. In case of the relations between Bob and the distributor, Bob will be a provider subject (PS) and the distributor will be a consumer subject (CS). Bob will have certain provider rights (PR) that are agreed at the time of a contract with the distributor. The distributor will have rights (CR) to distribute the MP3 song (PNO) and get certain profits from the sales. Likewise, in case of Alice and the distributor, Alice will be a consumer subject and the distributor will be a provider subject. Then Alice has rights (CR) such as play right for the song and the distributor will have rights (PR) such as copy and disseminate rights on the object. In this case, Alice may be required to pay ahead (CA) to obtain a play right but only on a specific player (CC) which is selected by her. In addition, she may have to agree on submission of her usage log report to the provider (CB). On the other hand, the distributor can have rights to collect consumers' usage log information. This shows that in UCON system, a consumer's obligation is likely to be a provider's right and vice versa.

One good example for the control of privacy sensitive objects might be a healthcare system. We consider a healthcare system called PCASSO to demonstrate the UCON model for privacy sensitive objects. The PCASSO project was developed by UC San Diego and SAIC under the support of NIH [BBB97]. The main purpose of the project is to develop a healthcare system that provides secure access to highly

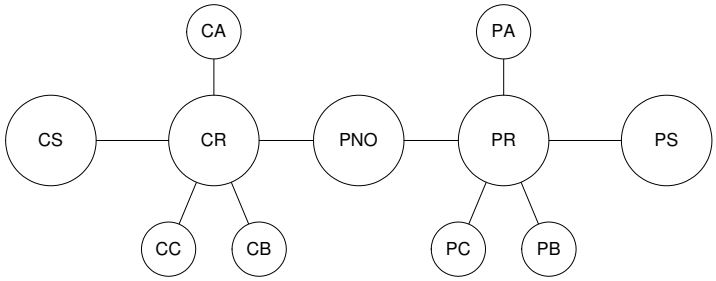


Figure 6.2: Two Sides of UCON Model for Privacy Non-sensitive Objects

sensitive patient information over Internet. Access control of PCASSO mainly utilizes labels and roles. Patient records are labeled with one of 5 security levels including Low, Standard, Deniable, Guardian Deniable, and Patient Deniable. As a provider subject, the primary care provider provides patient medical record (PSO). In addition, the primary care provider decides security level of patient medical information. Care providers (primary, emergency or others), guardians, researchers, and even patients can be consumer subjects. In PCASSO, the patient role can be either a consumer subject or an identifiee subject. As a consumer subject, a patient can read his medical record if it is not patient deniable. As an identifiee subject, the patient can review (IR) access log information on his record. Note that the patient doesn't have rights to decide use and disclosure of his medical information in PCASSO.

According to recent regulation called the Privacy Rule from the US Department of Health and Human Services (HHS), healthcare providers such as doctors and hospitals are required to obtain a patient's written consent before using or disclosing the patient's personal healthcare information to carry out treatment, payment, or healthcare operations (TPO) [hhs02]. To use or disclose the patient's medical information for other reasons than TPO, healthcare providers are required to obtain written authorization documents. In Privacy Rule, authorization is more detailed and specific than consent. In PCASSO, neither consent nor authorization is included in the sys-

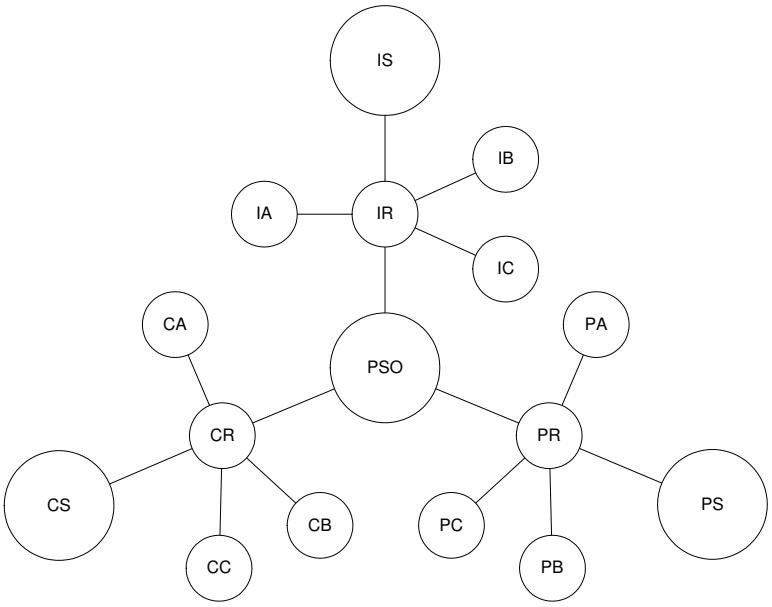


Figure 6.3: Three Sides of UCON Model for Privacy Sensitive Objects

tem. Moreover, usage and disclosure of patient medical information is entirely up to a primary care provider. For better control of all parties on patients' healthcare information and for better privacy protection, these consent and authorization should be part of identifiee rights in UCON model. Also, it should be the patient who holds those identifiee rights.

6.1.3 Reverse UCON

As mentioned above, obtaining or exercising usage rights on a digital object may create another digital information object (derivative object) which also needs controls for its access and usage. Some examples are payment information, usage log, etc. The usage control on these derivative objects is reversed in its control direction in such a way that the provider subject becomes the consumer subject and vice versa. This reversed usage control is called reverse UCON and the rights are called reverse rights. Furthermore, obtaining or exercising the reverse rights on these derivative objects

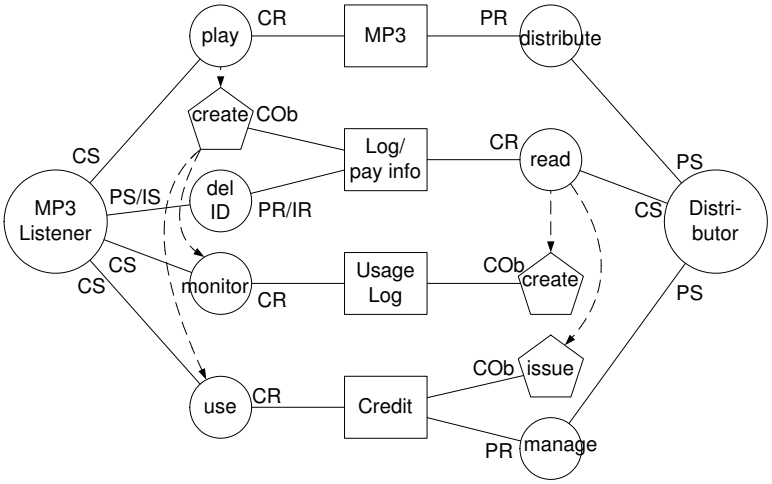


Figure 6.4: An Example of Reverse UCON

may also create other derivative objects and reverse (more correctly inverse) rights on it.

Figure 6.4 shows an example of reverse UCON. Some components are omitted in this diagram for the sake of simplicity. Suppose Alice wants to listen to a MP3 music file. To obtain play rights, she as a consumer subject (CS) may have to agree on payment-per-play (OB: obligation) and provide credit card information. Upon her exercise of the play rights, she has to report her usage log on the MP3 file (OB). In UCON, this payment information and log information are also considered as objects (derivative objects) and as part of UCON model. Now Alice becomes both a provider subject (PS) and an identifiee subject (IS) of the log/payment information and may hold certain rights (PR and IR) on them such as the right that she can delete her ID from log information. The distributor may have rights to collect log information either by putting an obligation on consumer rights or by giving consumer rights to get some store credits on log reports. If Alice has rights to get some store credit based on her play time, then it is now distributor's obligation as a provider subject to issue

certain credit to Alice.

Control and protection of rights and usage of rights on the derivative objects have been hardly recognized or discussed in information security literature. In UCON, reverse UCON can be viewed as part of the UCON model and is not different from ordinary UCON in its model specifications. In general, derivative objects are likely to include privacy-related information. Adequate controls on derivative objects will be crucial for better privacy treatment. By handling derivative objects in UCON system, at least security and privacy issues can be discussed systematically within a common framework.

UCON systems are likely to be implemented and managed under the control of one of three subject sides: consumer, provider or identifiee. This implies it's hard to guarantee availability of adequate control mechanisms implemented for the other two sides on the rights and usage of rights. There can be also a third party who develops/manages UCON system on behalf of all of PS, CS and IS sides. Therefore, to make a sound reverse UCON system available, there should be either a voluntary commitment from a development/management group or legal enforcement. In its implementation, UCON system may have to include following mechanisms for reverse UCON.

- To provide ability to review detail of derivative objects which are going to be created.
- To provide ability to refuse creation of derivative objects (the consumer may have to give up or reduce exercising original rights).
- To provide ability to restrict reverse usage by blocking certain part of derivative objects (i.e., identity) or by allowing only aggregated information of individual objects.

- To provide ability to monitor reverse usage on derivative objects (this may cause another round of reverse UCON).

6.2 Originator Control in UCON

One of the key concerns of UCON is how to control re-dissemination of disseminated digital objects. Originator Control (ORCON) is an access control policy that requires recipients to gain originator's approval for re-dissemination of an originally disseminated digital object or a new digital object that includes originally distributed digital objects.

UCON is a relatively new approach for next generation information security solutions, while ORCON has been discussed for more than a decade. Nevertheless, ORCON and UCON are alike in many aspects. In ORCON's perspective, by using UCON technologies, ORCON policies can be enforced in more versatile and flexible ways compared to the traditional ORCON solutions because blending ORCON with UCON enables control of dissemination and re-dissemination outside of a closed system environment where central control authority such as a reference monitor is not available. In UCON's perspective, ORCON is a "must have" policy because ORCON policy is one of the generic access control policies that are applicable to UCON solutions. Unlike other access control policies like RBAC, MAC and DAC, ORCON is naturally applicable for both payment-free and payment-based dissemination control.

Regardless of this tight relationship, ORCON has not been examined carefully in current DRM solutions because of the lack of immediate commercial interest. I believe investigating UCON with ORCON policy in mind can provide a promising way to control and manage digital information dissemination not only for non-commercial, payment-free environment such as the intelligence community or the commercial B2B environment, but also for commercial, payment-based dissemination.

Next two sections explain UCON and ORCON technologies briefly. Then, I define license and ticket concepts which are key elements in the UCON solution to implement ORCON policies. Then, I demonstrate how ORCON in UCON solutions can extend traditional ORCON to control even outside of the local control domain area and identify variations of ORCON policy enforcement in UCON solutions by using license and ticket. Finally, I relate some recent DRM works of other authors to our ORCON in UCON solutions.

6.2.1 Originator Control (ORCON)

In the spectrum of traditional access control policies, Mandatory Access control (MAC) and Discretionary Access Control (DAC) are at opposite extremes. Between MAC and DAC ends of the spectrum, there are areas where neither MAC nor DAC are applicable. ORCON is one of the access control policies that belong to this middle ground. ORCON is similar to MAC in that access restrictions on original objects are propagated to derived objects. However, ORCON is different from MAC in that policies are modifiable on a subject/object basis, while in MAC policies are uniform across all subjects and objects. Also, ORCON is similar to DAC in that policies are changeable by the original owner or originator of the object. However, ORCON is different from DAC in that control privileges on an object can be modifiable only by the originator of the object, while in DAC the owner (recipient) of a derived object can often also change control privileges on the object or on copies of the object. In some sense, DAC can be viewed a special case of ORCON where the originator delegates all the rights to recipients.

In the paper world, ORCON is one of the control markings for restriction of document distribution defined by the Director of Central Intelligence Directive (DCID) 1/7 [dci81]. A document marked ORCON can only be distributed with the approval

of the originator of the document. Traditional ORCON solutions [AHK+91, Gra89, MMN90, San92] try to automate the paper world's originator controlled dissemination policies. In the proposed solutions, ORCON policies typically utilize some form of non-discretionary access control list [Abr93]. The implementation of this non-discretionary access control list, however, limits the ability to enforce ORCON policies to a closed control environment.

Traditional ORCON solutions are focused on the enforcement of ORCON access control policies within a control domain. A control domain implies a system environment that facilitates a central means to control access of any subject within the domain to digital information objects. These solutions have tried to enforce access control policies in a centrally controlled manner. They normally set centrally controlled policies for a whole domain and all of the users have to behave within the boundaries of the policies. These solutions may run on either mainframe systems or client-server systems.

Figure 6.5 illustrates the structure of a traditional ORCON solution. In this schematic, the originator creates a digital object marked with "ORCON" and makes it available to subject A by setting appropriate access control policies that are tied to a subject and object relationship. If subject A wants to allow subject B to access the received digital object, the control authority (which is effectively a reference monitor) must check if subject B's access to the object is allowed or not by the non-discretionary access control lists. In this way, the originator can always control recipient access to the distributed digital object.

6.2.2 ORCON in UCON

UCON is different from access control policies. The usage rights of UCON are more versatile and finer-grained than privileges of traditional access control policies. UCON

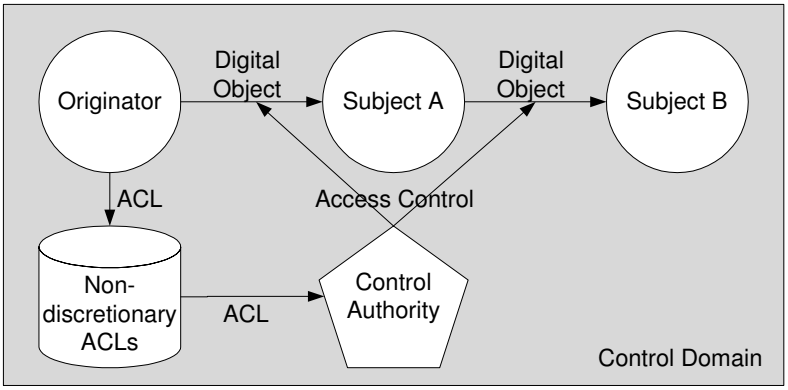


Figure 6.5: Traditional ORCON Solutions

can be viewed as a broader concept than access control. UCON solution includes various kinds of access control policies. ORCON is one of the most tightly related access control policies to UCON solutions. ORCON and UCON are similar in their goals. Both focus on original distributor's controls on the usage of distributed digital objects. While UCON deals with delegation of control privilege and controlling re-dissemination of digital objects, ORCON considers only re-dissemination of distributed digital information objects. In UCON, we can implement other access control policies such as MAC, DAC and RBAC. In practice, the re-dissemination control of ORCON policies can be enforced by allowing access to the digital object with the originator's direct or indirect approval. In UCON, ORCON policies can be achieved in different ways by using licenses and tickets. In the next subsections, I define license and ticket concepts and demonstrate how ORCON in UCON can support control of digital information re-disseminations.

License and Ticket

License and ticket are used for propagation of usage rights such as read, print, dissemination/re-dissemination rights, etc. They are key concepts needed to im-

plement ORCON policies in the UCON solution. Commercial DRM solutions also use some form of license or even tickets. However, most of these solutions have used them for payment-based dissemination and usage. There are few, if any, solutions for payment-free dissemination where payment does not matter and access control policies are resolved. In payment-free dissemination, authorization requires certain access control policies such as MAC, DAC, RBAC or ORCON. By using license and ticket, we can enforce ORCON policies for digital information dissemination.

A license is a digitally signed certificate that includes all the usage rights information of qualified recipients on specified digital objects and allows the user access to the digital objects through a Virtual Machine. Only users with a qualified license are allowed to access digital objects. With ORCON in UCON, a license has to be issued by either the originator of a digital object or third parties approved by the originator. A license may or may not include a license-issuing privilege (LIP). If LIP is included in a license issued by the originator, the recipient of this primary license can issue a secondary license to a license requester without consulting the originator about the request.

A ticket is a specialized license that is used either to delegate LIP or to provide the information from where the requester can obtain a license. A ticket may be used only for a limited number of times or for a limited time period and marked to be void after use. In this section I define two different kinds of tickets: License-Granting Ticket (LGT) and License-Requesting Ticket (LRT). The issuer of the LGT is the originator of the requested digital object (or recipients who are qualified as issuers of the LGT by the originator), while the issuer of the LRT is the original requestee (that is, the subject to whom the license requestor makes its original request). Both the LGT and the LRT may include ticket issuer, ticket recipient, license issuer and license recipient information.

A LGT always include LIP. A LGT may also include usage rights information, so it can be used when a license is issued to original requesters. By issuing a LGT, the issuer (including the originator) can allow third parties to issue licenses to further availability of the digital object on behalf of the originator or the previous issuer. LIP can optionally exist in a license. If a license includes LIP, re-dissemination is possible only to pre-defined qualifiers. However, with a LGT, approval can be reviewed case-by-case upon each license request and the distribution can be monitored. A license with LIP and a LGT can coexist, but cannot be used for the same request instance together.

LRT is a ticket that is issued by an original requestee other than originator and is used by the requester to request a license from the originator to access a digital object. The original requestee is the primary recipient from whom the requester gets the digital object information and is the subject who is asked for the license. Suppose the original requestee does not have an appropriate LGT to fulfill the request. In this case the original requestee can return a LRT to the requestor, and this LRT can be used to request a license from the originator. The originator will then decide to issue the license by checking whether the license requester is a qualified user and has a valid LRT.

Binding ORCON and UCON

While traditional ORCON solutions try to control access to disseminated digital objects centrally, the solutions of ORCON in UCON try to enforce access control policies for both centralized and de-centralized cases. As mentioned previously, traditional ORCON solutions can enforce access control policies within the closed control domain environment. However, by using the concept of license and ticket, ORCON in UCON can go further and enforce access control policies outside the control domain area.

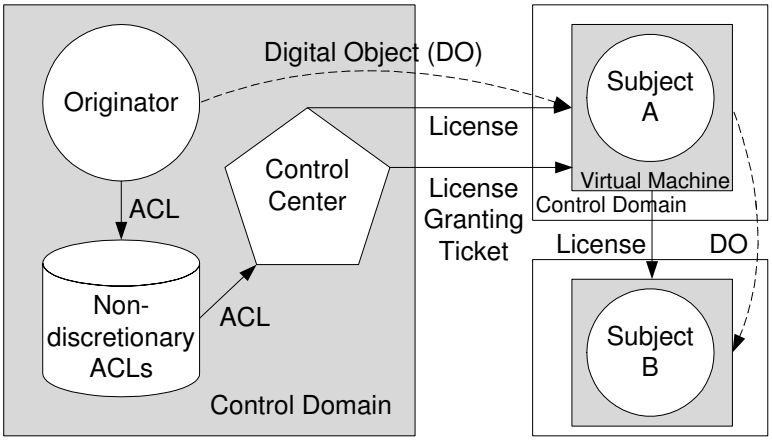


Figure 6.6: An Example of ORCON with UCON

Figure 6.6 shows an example of ORCON in UCON and demonstrates how ORCON in UCON solutions can enforce access control policies outside of the local control domain. Here, each recipient belongs to different control domains. The gray areas (originator's control domain and recipients' virtual machine) indicate the control areas of an originator on disseminated digital objects. Control mechanisms for dissemination and re-dissemination of the originator's digital object within the originator's control domain are exactly the same as those for the open control environment. In ORCON in UCON solution (for both the closed and open control environments), access to the disseminated digital objects can be done only through the virtual machine. In Figure 6.6, the digital object is available to Subject A either directly or indirectly. The key is that Subject A needs a license to access the received digital information object. In this particular example, Subject A gets a LGT from the originator so she can issue a license to Subject B.

6.2.3 Variations of ORCON in UCON

In UCON, ORCON can be accomplished in various ways. In the following subsections, I identify different kinds of architectural approaches by incorporating the concepts of license and ticket in different ways. However, I am not trying to analyze every aspect of each variation. Rather, I am trying to demonstrate different approaches by introducing different uses of licenses and tickets. The purpose or benefit of this distinction is not to show which one is better than others but to demonstrate possible approaches based on message (e.g., usage rights requests, rights approval and rights delegation information) flows, so that the approaches can be considered and included in UCON solutions to deal with various situations. In some cases, for example, an originator may want to delegate license-issuing privilege to other recipients so further requests can be handled by authorized recipients without the originator's involvement. In other cases, if requester does not have originator's contact information, she may first have to contact the provider of digital information object. Unlike traditional ORCON solutions, possession of both the digital information object and the relevant license with qualified usage rights is required to access the object.

In the following figures, note that though the virtual machine is omitted for the sake of convenience, it is required for every recipient and should be used to handle all license/ticket requests of and license/ticket issuance to subjects other than the originator. In addition, the digital object is assumed available to requesters (the digital object may or may not be received directly from the requestee) and is not explicitly shown. Also, detailed configurations of the originator site are omitted since there can be many possibilities. Although LRT can be used for license/LGT requests, we consider this as a request. The Legend shown in Figure 6.7 pertains to other figures in this section.

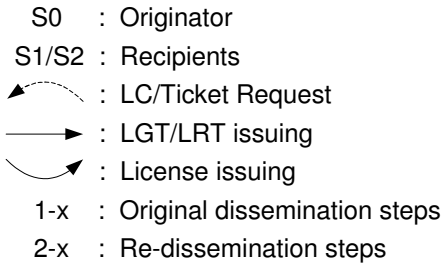


Figure 6.7: Legend for ORCON Variation

Re-dissemination without Ticket

The originator can approve recipient re-dissemination of the distributed digital object without implementing a ticket either by issuing a license directly to the original requester or by issuing a license that includes LIP. The originator receives requests either directly or indirectly. In Figure 6.8(a), S0 issues a license with LIP to S1 (1-2). If S2 requests a license from S1 (2-1), S1 issues a license with tacit permission from S0 (2-2) because S1 has LIP. This is the only approach in which a license should include LIP among our ORCON in UCON solutions. This means that the originator delegates license-issuing privilege to primary recipients so the recipients can issue licenses to third parties without asking originator's authorizations. This may be useful when the originator wants to distribute its license issuing tasks to increase performance or availability.

In Figure 6.8(b), S1 requests a license from the originator S0 (1-1) to access the digital object that is originally released by S0 and S0 issues a license for the digital object (1-2). If S2 gets the digital object from S1 and wants to access it, S2 requests a license from S1 (2-1) and S1 requests a license for S2 from S0 (2-2). If qualified, S0 issues a license directly to S2. This approach can be used in case the requestee cannot or doesn't want to issue a license or a ticket to the requester. Figure 6.8(c)

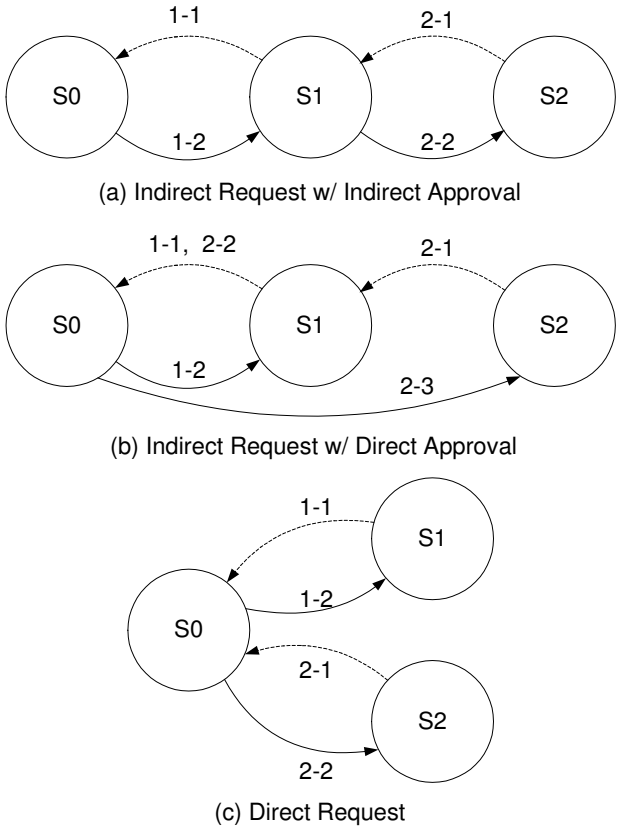


Figure 6.8: Re-dissemination without Ticket

is the same as Figure 6.8(b) except that the license request is submitted directly to the originator, rather than through S1. The originator responds to every request directly without any involvement of requesters. In 6.8(b) and 6.8(c), there is no authorization activity of recipients. Rather the originator authorizes usage rights by issuing the license directly to the original requester. However, these latter two cases provide the same functional effect as in re-dissemination under the originator's control.

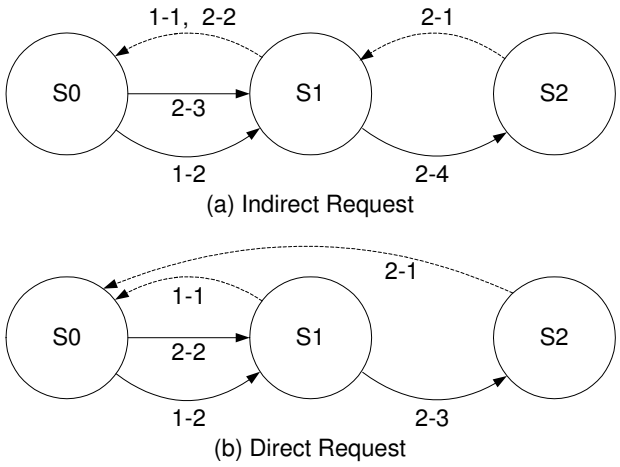


Figure 6.9: Re-dissemination with LGT

Re-dissemination with LGT

A license-granting ticket (LGT) can be used to delegate the license-issuing privilege to recipients. In an indirect request configuration (Figure 6.9(a)), S2 requests a license from S1 and S1 requests a LGT from S0. If qualified, S0 issues a LGT to S1 so S1 can issue a license to S2. The direct request approach (Figure 6.9(b)) is same as the indirect request approach except that S2 requests a license (LGT) directly from S0. As mentioned previously, LGT is different from a license with LIP in that a LGT is issued upon requests and can be customized to each request while a license is pre-issued for future requests.

Re-dissemination with LRT

Re-dissemination with LRT, illustrated in Figure 6.10, is similar to re-dissemination without a ticket with direct request (Figure 6.8(c)) except that it requires a LRT from the previous recipient (2-1, 2-2) so the LRT can be submitted to S0 with a direct request for a license from S0 (2-3). For simplicity, It is assumed that the original recipient does not have to present a LRT to the originator to receive a license

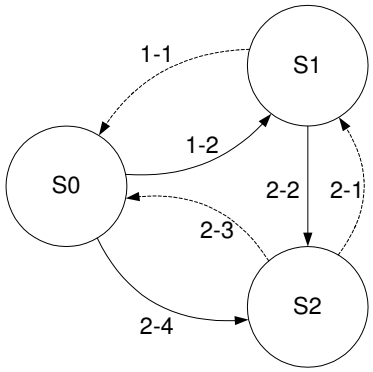


Figure 6.10: Re-dissemination with LRT

for his or her own usage on the digital object released by originator. Approaches using the LRT (Figure 6.10, 6.11) require two requests from the requester S2. In this case, S2's requests for a license include LRT so S0 may verify S1's agreement on the re-dissemination.

Re-dissemination with LGT and LRT

Re-dissemination with LGT and LRT, illustrated in Figure 6.11, is like re-dissemination with LGT with a direct request (Figure 6.9(b)) except that it requires LRT from the previous recipient. Thus the LRT can be submitted directly to the originator for a license issuing. Unlike other previous approaches, this approach requires both LGT and LRT to implement ORCON policies. Note that in step 2-2, a LRT is issued to S2 and in step 2-4, a LGT is issued to S1 so that S1 can issue a license to S2 (2-5). In both Figure 6.10 and 6.11, S0 can verify S1's agreement on re-dissemination to S2. Also S2 has to place two requests to get a license.

6.2.4 Discussion and Related Work

Currently, literature available on models and languages for DRM or UCON is scarce. Most of the work is done in the commercial sector. Some commercial efforts intend

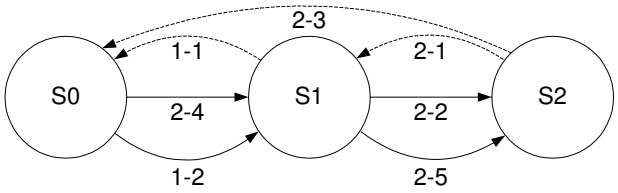


Figure 6.11: Re-dissemination with LGT and LRT

to be proprietary and others to be open standards. These models and languages for DRM have mainly focused on payment-enabled usage control systems. None of the solutions has carefully considered access control policies for the environment where payment is irrelevant. In this section, I relate some of the published works available to the ORCON in UCON solutions and to the license and ticket concepts demonstrated above.

DRM languages by InterTrust

Gunter et al have developed a very simple mathematical model and language to describe licenses for DRM solutions based on InterTrust's DRM solutions [GWW01]. This mathematical model was developed to define license precisely in its semantic meaning. In their models and languages, Gunter et al focused on simplified payment-based control. Their models and languages fail to describe re-dissemination and delegation control functions. Specifically they do not have any kind of ticket concept. The only way to enforce ORCON policies is to include a license-issuing privilege within a license. This means that no instant and temporary delegation of re-dissemination privilege is possible upon request for recipients to re-disseminate the received digital information object but only pre-defined (and delivered) re-dissemination rules can be used. I believe using the ticket concept will improve Gunter et al's model.

XrML

ContentGuard has defined XrML as "a language in XML for describing specifications of rights, fees and conditions for using digital contents, together with message integrity and entity authentication within these specifications" [Con02]. Historically, XrML is an extension of the Xerox "Digital Property Rights Language version 2.0 (DPRL)" and has been developed as an open specification licensed on a royalty-free basis by ContentGuard. ContentGuard claims the purpose of XrML, for the commercial sector, is to support commerce in digital contents (i.e. e-book, digital movie, games, software, etc.) and for Intelligence Community, the purpose is to support specification of access and use controls for secure digital documents. However, XrML still lacks well-defined enforcement of access control policies.

We can build a license in the form of XML by using XrML specification. The license can be used as a "description part" of XrML solutions. XrML defines digital license and use "certificate" element under "aPrincipal" entity which is used for identification of principal. This certificate is different from the license because it is used for authentication just like an identity certificate. They also include digital ticket concept using "ticket" element. However, this ticket element is one of two sub-elements of a "fee" element and is used only for the evidence of payment, just like a ticket at a movie theater. By considering this ORCON in UCON approaches, I believe that XrML language specifications and solutions can be improved.

ODRM

ODRM stands for Open Digital Rights Management. It is developed by IPR Systems Pty Ltd. and has been submitted as a position paper for the W3C DRM workshop [Ian02]. ODRM is based mainly on a DRM model and DRM language (ODRL). Like XrML, ODRL uses XML for model expression. Since ODRM is still preliminary

and focused on the semantics of expressing rights languages, it has not yet evolved any mechanisms or expressions for delegation of dissemination or re-dissemination privileges.

6.2.5 Summary for ORCON in UCON

In this section, I reviewed UCON and ORCON in general and introduced two most important elements: license and ticket. Then, I discussed the differences between traditional ORCON solutions and ORCON in UCON solutions to demonstrate extended control on the usage of disseminated digital information and proposed seven approaches that implements license and ticket concepts in various ways. Then I compared some characteristics of each approach. Finally, I briefly discussed currently available DRM solutions in terms of UCON solutions and provided some suggestions.

The study performed in this section is the first systematic study of this topic. In particular, the solutions I have proposed have not been previously defined in this manner in the literature. Also I am first to suggest license and ticket concepts to enforce ORCON policies in UCON solutions. Nevertheless, this section does not provide comprehensive solutions. It provides the basis for future research and development for usage control solutions that enforce access control policies for dissemination and re-dissemination of digital information objects. Many aspects should be considered for better understanding of this subject. One crucial aspect is how to revoke the authorized usage rights. Although revocation is not discussed in this section, since ORCON in UCON deals with delegations of usage rights, careful studies on revocation of these rights should be performed in further research. In addition, a solid understanding of the models and languages is essential for the development of practical usage control solutions. Further research on these aspects will lead to comprehensive and more practical solutions for digital information dissemination controls.

Chapter 7

CONCLUSION

The following sections summarize contributions of this dissertation and discuss some future research directions that have to be further studied to enrich the are of usage control.

7.1 Contributions

It has been my explicit goal to accommodate all the ideas we have seen in the access control literature in the past decade in a single unified framework. In particular I have looked to the Digital Rights Management community for inspiration to take me beyond the usual bounds of access control. Over the last 30 plus years, traditional access control policies and models have dealt with authorization only. Modern information systems require more than authorization process to protect digital resources from unwanted usages.

As a result, I have defined a new area of usage control and developed a novel framework for information and systems protection. Usage control explicitly defines obligations and conditions as well as traditional authorizations within the framework for finer and richer controls on usage of digital resources. Usage control unifies traditional access control, trust management, and digital rights management, and goes beyond in its scope.

Specifically, I have developed a family of ABC core models for UCON and have shown

how ABC models can be utilized for MAC, DAC, RBAC, trust management, and DRM. ABC models not only unify these diverse disciplines but also cover many other important issues such as continuity (ongoing control) and mutability (attribute updates) properties systematically and comprehensively. Ongoing control and attribute updates are two important aspects that must be resolved in modern information systems security.

In architectural perspective, I have defined the structure of a new reference monitor that is crucial for usage control and have shown three different variations of it. By utilizing a client-side reference monitor, I have identified eight different dissemination architectures for usage control in a systematic manner and further discussed how commercial products can fit into the architectures to show commercial availability of the architectures.

Also, as an application of the ABC models, UCON administration issues have been discussed and DRM and Healthcare examples in UCON administration are shown. In addition, originator control policy has been discussed to show how it can be recaptured to support usage control.

There is increasing realization that traditional access control is not adequate for modern application needs. Many researchers have published possible extensions to the basic access control concepts. This research is the first effort to overhaul the underlying foundation of access control itself. It provides a robust and integrated framework for access control models and systems of the future. I believe this UCON approach will contribute to re-unify a discipline that is starting to get fragmented at a time when the importance of access control is being increasingly appreciated.

7.2 Future Research

I believe usage control achieves the desired unification at an appropriate level of abstraction and provides a solid foundation for further research. There are several areas that must be continued for more comprehensive solutions in usage control.

In this dissertation, ABC models leave certain administration issues incomplete, including controls on provider and identifiee subjects parties. Although ABC core models can be also applicable to the parties other than consumer subjects, UCON requires more than a simple combination of controls on the usage of each subject parties. Administrative issues of usage control deal with administration or management of the relationships among these different parties. I believe further studies on administrative UCON will provide significant enhancements to usage control.

Within the ABC model, a detailed model for update procedure has not been developed for simplicity in taking the first step. Regarding that there is no systematic treatment for attribute updates in previous literature, and that current rights expression languages such as XrML and XACML lack update concept, further development of update procedure will be a promising evolution for ABC model and UCON in general.

Throughout the research, delegation of usage rights has not been discussed. Further research on delegation of usage rights will enrich usage control framework. This is one of the important challenges as we look ahead.

In architectural point of view, only CRM based architectures for payment-free type dissemination have been discussed in this dissertation. In future, this should be extended to cover payment-based dissemination architectures mainly for commercial B2C applications. Also, studies on architectures for both CRM and SRM along with details of reference monitor itself have to be done. In addition, studies on security

architectures for B2B systems where multiple organizations are involved will also enrich UCON architectures.

After all, the main purpose of usage control is to provide a foundational framework that can be referenced for analysis of existing systems and for development of new systems. To make usage control more useful in real world applications, studies on usage control engineering have to be done. Specifically, analysis of usage decision policy has to be conducted for well-designed decision rules and attributes. With a well-engineered usage control system, the usability of usage control can be increased. Therefore, further studies on UCON engineering is one of the success factor of usage control.

I believe further studies on these issues will provide more comprehensive solution approaches for the area of usage control.

BIBLIOGRAPHY

BIBLIOGRAPHY

[ABL93] M. Abadi, M. Burrows, and B. Lampson. A calculus for access control in distributed systems. In *ACM Transactions on Programming Languages and Systems*, pages 706–734, 1993.

[Abr93] Marchall Abrams. Renewed understanding of access control policies. In *Proceedings of 16th NIST-NCSC National Computer Security Conference*, pages 87–96, 1993.

[AHK+91] M. Abrams, J. Heaney, O. King, L. LaPadula, M. Lazear, and Ingrid. Olson. Generalized framework for access control: Toward prototyping the orgcon policy. In *Proceedings of 14th NIST-NCSC National Computer Security Conference*, pages 257–266, 1991.

[And02] Ross Anderson. TCPA / Palladium frequently asked questions. *Online Available:http://www.cl.cam.ac.uk/˜rja14/tcpa-faq.html*, 2002.

[Arb97] William Arbaugh. A secure and reliable bootstrap architecture. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 65–71, 1997.

[BBB97] Dixie Baker, Robert Barnhart, and Teresa Buss. PCASSO: applying and extending state-of-the-art security in the healthcare domain. In *Proceedings of 13th Annual Computer Security Application Conference*, 1997.

[BFL96] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized trust management. In *Proceedings of IEEE Symposium on Security and Privacy*, 1996.

[BJWW02] C. Bettini, S. Jajodia, X. Wang, and D. Wijesekera. Obligation monitoring in policy management. In *Proceedings of 3rd International Workshop on Policies for Distributed Systems and Networks*, 2002.

[BL73] D. Bell and L. LaPadula. Secure computer systems: Mathematical foundations and model. *MITRE Report*, 2(2547), November 1973.

[Con02] ContentGuard. XrML: Extensible rights Markup Language Core 2.1 Specification. *Online, Available: http://www.xrml.org*, 2002.

[Cox96] BRAD Cox. *Superdistribution*. Addison Wesley, 1996.

[dci81] Control of dissemination of intelligence information. Directive No. 1/7, Director of Central Intelligence, May 1981.

[DDLS01] N. Damianou, N. Dulay, E. Lupu, and M. Sloman. The ponder policy specification language. In *Proceedings of 2nd International Workshop on Policies for Distributed Systems and Networks*, 2001.

[Dwo99] Cynthia Dwork. Copyright? protection? *The Mathematics of Information Coding, Extraction, and Distribution, The IMA Volumes in Mathematics and its Applications, Springer-Verlag*, 107, 1999.

[GM02] Simon Godik and Tim Moses. OASIS eXtensible Access Control Markup Language (XACML) Specification 1.0. *Online, Available: http://www.oasis-open.org/committees/xacml/docs/*, 2002.

[Gra89] Richard Graubart. On the need for a third form of access control. pages 296–303, 1989.

[GWW01] Carl Gunter, Stephen Weeks, and Andrew Wright. Models and languages for digital rights. In *Proceedings of the Hawaii International Conference on System Sciences*, 2001.

[hhs02] Standards for privacy of individually identifiable health information. *Online, Available: http://www.hhs.gov/ocr/hipaa/finalreg.html*, 2002.

[HMM+00] A. Herzberg, Y. Mass, J. Mihaeli, D. Naor, and Y. Ravid. Access control meets public key infrastructure, or: Assigning roles to strangers. In *Proceedings of IEEE Symposium on Security and Privacy*, 2000.

[HRU76] M.H. Harrison, W.L. Ruzzo, and J.D. Ullman. Protection in operating systems. *Communications of the ACM*, 19(8):461–471, 1976.

[Ian02] Renato Iannella. Open digital rights language v1.1. *Online, Available: http://odrl.net*, 2002.

[iso96] Security frameworks for open systems: Access control framework. Technical Report ISO/IEC 10181-3, ISO, 1996.

[JKS01] S. Jajodia, M. Kudo, and V.S. Subrahmanian. Provisional authorizations. In Anup Gosh, editor, *E-Commerce Security and Privacy*. Kluwer Academic Press, 2001.

[Kap96] Marc Kaplan. IBM cryptolopes, superdistribution and digital right management. *Online, Available:http://www.research.ibm.com/people/k/kaplan/cryptolope-docs/crypap.html*, 1996.

[KH00] Michiharu Kudo and Satoshi Hada. XML document security based on provisional authorization. In *Proceedings of ACM Conference on Computer and Communications Security*, 2000.

[KK00] Jens Koblin and Michael Kockelkorn. The IMPRIMATUR Multimedia IPR Management System. *Online, Available: http://www.imprimatur.alcs.co.uk/newstore.htm*, 2000.

[Lam71] B.W. Lampson. Protection. In *5th Princeton Symposium on Information Science and Systems*, pages 437–443, 1971. Reprinted in *ACM Operating Systems Review* 8(1):18–24, 1974.

[LaM02] B. LaMacchia. Key challenges in DRM: An industry perspective. In *Proceedings of 2nd ACM DRM Workshop*, November 2002. in conjunction with ACM CCS Conference.

[Lan97] Carl Landwehr. Protection (security) models and policy. In Allen B. Tucker, editor, *The Computer Science and Engineering Handbook*, pages 1914–1928. CRC Press, 1997.

[mit01] Ten emerging technologies that will change the world. *MIT Technology Review*, Jan/Feb 2001.

[MMN90] Catherine J. McCollum, Judith R. Messing, and LouAnna Notargiacomo. Beyond the pale of mac and dac - defining new form of access control. In *Proceedings of IEEE Symposium on Research in Security and Privacy*, 1990.

[p3p02] The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. Technical report, W3C, Online, Available: http://www.w3.org/P3P/, 2002.

[PSS00] Jaehong Park, Ravi Sandhu, and James Schifalacqua. Security architectures for controlled digital information dissemination. In *Proceedings of 16th Annual Computer Security Application Conference*, 2000.

[RN01] Tatyana Ryutov and Clifford Neuman. The set and function approach to modeling authorization in distributed systems. In *Proceedings of the Workshop on Mathematical Methods and Models and Architecture for Computer Networks Security*, 2001.

[RN02] Tatyana Ryutov and Clifford Neuman. The specification and enforcement of advanced security policies. In *Proceedings of 3rd International Workshop on Policies for Distributed Systems and Networks*, 2002.

[RTM02] B. Rosenblatt, B. Trippe, and S. Mooney. *Digital Rights Management: Business and Technology*. M&T Books, 2002.

[San92] Ravi Sandhu. The typed access matrix model. In *Proceedings of IEEE Symposium on Research in Security and Privacy*, pages 122–136, 1992.

[San93] Ravi Sandhu. Lattice-based access control models. *IEEE Computer*, pages 9–19, November 1993.

[San00] Ravi Sandhu. Engineering authority and trust in cyberspace: The oam and rbac way. In *Proceedings of 5th ACM Workshop on Role-Based Access Control*, 2000.

[SBW95] O. Sibert, D. Bernstein, and D.V. Wie. The digibox: A self-protecting container for information commerce. In *Proceedings of USENIX Workshop on Electronic Commerce*, 1995.

[SCFY96] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman. Role-based access control models. *IEEE Computer*, pages 38–47, February 1996.

[Sch99] Paul Schneck. Persistent access control to prevent piracy of digital information. In *Proceedings of the IEEE*, volume 87, July 1999.

[SM02] A. Schaad and J. Moffett. Delegation of obligation. In *Proceedings of 3rd International Workshop on Policies for Distributed Systems and Networks*, 2002.

[SS94] Ravi Sandhu and Pierangela Samarati. Access control: Principles and practice. *IEEE Communications*, pages 40–48, September 1994.

[tcp02] Trusted Computing Paltform Alliance, Main Specification V1.1b. Technical report, TCPA, Online Available:http://www.trustedcomputing.org/docs, 2002.

[TS97] Roshan Thomas and Ravi Sandhu. Task-based authorization controls (TBAC): Models for active and enterprise-oriented authorization management. *Database Security XI: Status and Prospects*, 1997.

[Wee01] Stephen Weeks. Understanding trust management systems. In *Proceedings of IEEE Symposium on Security and Privacy*, 2001.

[WLD+02] Wang, Lao, DeMartini, Reddy, Nguyen, and Valenzuela. XrML - eXtensible rights Markup Language. In *Proceedings of ACM Workshop on XML Security*, 2002.

[WSJ00] W. Winsborough, K. Seamons, and V. Jones. Automated trust negotiation. In *Proceedings of the DARPA Information Survivability Conference and Exposition*, 2000.

[Zha97] Jian Zhao. Applying digital watermarking techniques to online multimedia commerce. In *In Proc. of the International Conference on Imaging Science, Systems, and Applications*, 1997.

VITA

Jaehong Park was born on February 12, 1969, in Korea and is a citizen of Korea. He received the B.B.A in Management Information System from Dongguk University, Seoul, Korea in 1995 and M.S. in Information Systems from the George Washington University, Washington D.C., USA in 1998. During 1995 - 1996, he was a software engineer of POSDATA Co., Seoul, Korea. Currently he is a member of Laboratory for Information Security Technology at George Mason University.

This dissertation was typeset with LaTeX[‡] by the author.

[‡]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.