Digital Rights Management for Personal Networks

Imad Mahmoud Aref Abbadi

Technical Report
RHUL–MA–2008–17
4 June 2008

Department of Mathematics
Royal Holloway, University of London
Egham, Surrey TW20 0EX, England
http://www.rhul.ac.uk/mathematics/techreports

Digital Rights Management for Personal Networks

by

Imad Mahmoud Aref Abbadi

Thesis submitted to the University of London
for the degree of Doctor of Philosophy

Information Security Group
Royal Holloway, University of London

2008

Declaration

These doctoral studies were conducted under the supervision of Prof. Chris Mitchell.

The work presented in this thesis is the result of original research carried out by myself, in collaboration with others, whilst enrolled in the Information Security Group as a candidate for the degree of Doctor of Philosophy. This work has not been submitted for any other degree or award in any other university or educational establishment.

Imad Mahmoud Aref Abbadi
2 Jun 2008

Abstract

The thesis is concerned with Digital Rights Management (DRM), and in particular with DRM for networks of devices owned by a single individual. This thesis focuses on the problem of preventing illegal copying of digital assets without jeopardising the right of legitimate licence holders to transfer content between their own devices, which collectively make up what we refer to as an authorised domain.

An ideal list of DRM requirements is specified, which takes into account the points of view of users, content providers and copyright law. An approach is then developed for assessing DRM systems based on the defined DRM requirements; the most widely discussed DRM schemes are then analysed and assessed, where the main focus is on schemes which address the concept of an authorised domain. Based on this analysis we isolate the issues underlying the content piracy problem, and then provide a generic framework for a DRM system addressing the identified content piracy issues. The defined generic framework has been designed to avoid the weaknesses found in other schemes.

The main contributions of this thesis include developing four new approaches that can be used to implement the proposed generic framework for managing an authorised domain. The four novel solutions all involve secure means for creating, managing and using a secure domain, which consists of all devices owned by a single owner. The schemes allow secure content sharing between devices in a domain, and prevent the illegal copying of content to devices outside the domain. In addition, each solution incorporates a method for binding a domain to a single owner, ensuring that only a single consumer owns and manages a domain. This enables binding of content licences to a single owner, thereby limiting illicit content proliferation.

In the first solution, domain owners are authenticated using two-factor authentication, which involves "something the domain owner has", i.e. a master control device that controls and manages consumers domains, and binds devices joining a domain to itself, and "something the domain owner is or knows", i.e. a biometric or password/PIN authentication mechanism that is implemented by the master control device. In the second solution, domain owners are authenticated using their payment cards, building on existing electronic payment systems by ensuring that the name and the date of birth of a domain creator are the same for all devices joining a domain. In addition, this solution helps to protect consumers' privacy; unlike in existing electronic payment systems, payment card details are not exposed to third parties. The third solution involves the use of a domain-specific mobile phone and the mobile phone network operator to

authenticate a domain owner before devices can join a domain. The fourth solution involves the use of location-based services, ensuring that devices joining a consumer domain are located in physical proximity to the addresses registered for this domain. This restricts domain membership to devices in predefined geographical locations, helping to ensure that a single consumer owns and manages each domain.

Acknowledgements

I would like to thank my supervisor Prof. Chris Mitchell for his supervision, help, support, and encouragement. I am grateful to Prof. Ahmad-Reza Sadeghi, Dr. Jason Crampton and Prof. Peter Wild for their useful discussion and comments.

I would also like to thank Associated Newspapers for partially funding my research, and providing me with all the required resources, flexible working patterns, and time off to study that has enabled me to have a full time job and simultaneously complete a research degree.

I would like to thank my parents, who have always encouraged and supported me. Finally, I want to thank my wife for her support and patience throughout the time I have worked on this thesis, as it would have been impossible, without her help, to have a family, a full time job, and engage in full time study at the same time.

Contents

List of Figures

Acronyms

3GPP	3rd Generation Partnership Project
AIK	Attestation Identity Key
AKA	Authentication and Key Agreement protocol
BSF	Bootstrapping Server Function
CA	Certification Authority
CAS	Conditional Access System
CE	Conformance Entity
CEK	Content Encryption Key
CRTM	Core Root of Trust for Measurement
DA	Domain Authority
DAA	Direct Anonymous Attestation
DRM	Digital Rights Management
DCF	DRM Content Format
EK	Endorsement Key Pair
FAR	False Acceptance Rate
FRR	False Rejection Rate
GAA	Generic Authentication Architecture
GDRL	Global Device Revocation List
GMLC	Global Mobile Location Center

GNSS	Global Navigation Satellite System
GPS	Global Positioning Satellite receiver
HSS	Home Subscriber Server
IMSI	International Mobile Subscriber Identity
INDICARE	The INformed DIalogue about Consumer Acceptability of DRM Solutions in Europe
iTMS	iTunes Music Store
LBS	Location Based Services
LDI	Local Device Identification
LECM	Local Enforcement Copy Management
LRL	Local Revocation List
MA	Measurement Agent
MAC	Message Authentication Code
MC	Master Control device
MK	Master Key
MKL	Master Device Key List
MKB	Media Key Block
NAF	Network Application Function
NFC	Near Field Communication
OCSP	Online Certificate Status Protocol
ODRL	Open Digital Rights Language
OMA	Open Mobile Alliance
OS	Operating System
PPN	Personal Private Network
PC	Personal Computer
PCR	Platform Configuration Register
PE	Platform Entity
PKI	Public Key Infrastructure

PPR	Privacy Profile Register
REL	Rights Expression Language
RI	Rights Issuer
RTM	Root of Trust for Measurement
RTR	Root of Trust for Reporting
RTS	Root of Trust for Storage
RTT	Round-Trip Time
SML	Stored Measurement Log
SPA	MasterCard Secure Payment Application
SRK	Storage Root Key pair
TCG	Trusted Computing Group
TCPA	Trusted Computing Platform Alliance
TP	Trusted Platform
TPM	Trusted Platform Module
TPME	Trusted Platform Module Entity
TTP	Trusted Third Party
UICC	UMTS IC Card
VbV	Verified by Visa
VE	Validation Entity
VD	Visa Directory Server
xCP	eXtensible Content Protection
XrML	Extensible Rights Markup Language

Symbols

A_X	An identifier for X, as contained in Cert_{I_X}
Cert_{I_X}	A certificate signed by the CA for the public key I_X
$\text{Sign}_X(Y)$	A signature on data Y created using the private signing key of X
$E_X(Y)$	Symmetric encryption of data Y using key X, and where we assume that E provides authenticated encryption
$e_X(Y)$	Asymmetric encryption of data Y using public key X, and where we assume that the encryption primitive in use provides non-malleability
$g_k(Y)$	Symmetric encryption of data Y using key k
h	A globally agreed cryptographic hash-function
I_X	A signature verification key of X
$m_k(Y)$	MAC computed on data Y using key k
N	A fresh nonce
K_D	A domain-specific symmetric encryption key used to encrypt domain content encryption keys
K_T	A symmetric encryption key used to encrypt domain content
k_e	A session cipher key
k_i	A session integrity key
P_X/R_X	An asymmetric encryption/decryption key pair of entity X
S_X	The execution status of the DRM agent on device X
$X\|\|Y$	The result of the concatenation of data items X and Y in that order

Chapter 1

Introduction

Contents

This chapter gives an overview of the thesis. We provide the motivation for the research and describe the contributions of this thesis. We also present the overall structure of the thesis.

1.1 Motivation

Copyrighted digital asset protection is one of the most pressing current challenges in information security, as most elements of society have converted their content from physical to digital form. In the past, content piracy was limited to distribution via physical media. The recent digitisation of information, the development of communications technologies such as broadband and mobile networks, and the spread of the Internet have increased digital content piracy [19],

as content can be shared and transferred instantly with no loss of quality.

The simultaneous rapid growth of electronic commerce has opened a huge new market for digital goods, such as books, software, and music. However, content providers have concerns about the protection of their valuable content, especially since, as stated above, the Internet makes copying content very simple, resulting in increased digital content piracy [19]. Content providers therefore want to ensure that their content is protected against unauthorised use. Because of the concerns of content providers, DRM technologies have been introduced to help to prevent unauthorised use and distribution of content. DRM can be defined as the technologies that collectively support all stages in the lifecycle of digital content creation, manipulation, distribution and consumption, by preventing illegal copying and allowing the imposition of fees, processing of payments, and protection of principal rights and profits [43]. Whilst devising an effective DRM solution is clearly in the interests of the owners of content, the advantages to the consumer are much less clear. Indeed, if solutions prevent consumers using purchased copies of content in ways that they deem fair and reasonable, then the technology is likely to become very unpopular.

In this thesis we focus on ways of preventing the unauthorised copying of content without jeopardising the right of licence holders to transfer content between their own devices, which collectively make up what we refer to as an authorised domain. In addition, this thesis addresses a number of other fundamental DRM requirements, such as content backup and recovery, privacy, content mobility, and ease of use. One important issue that is not addressed in this thesis is the provision of a detailed analysis of copyright law requirements. The proposed schemes satisfy many copyright law requirements; however, some requirements have not been fully analysed, such an analysis would be an interesting

topic for future research. Most existing DRM solutions have significant security shortcomings and usability limitations in addressing the problem of illegal copying and simultaneously allowing free content sharing between devices belonging to a single user. In addition, many previously proposed DRM solutions have problems in addressing other fundamental DRM requirements, such as content backup and recovery, ease of use, and performance.

1.2 Contributions

This thesis includes the following novel contributions supporting the implementation of a successful DRM system.

- It defines an ideal list of DRM requirements from the points of view of users, content providers and copyright law.
- Using the list of DRM requirements, it analyses and assesses six of the most widely discussed DRM schemes that incorporate the authorised domain concept. We then isolate the main security issues constituting the content piracy problem.
- It defines a generic framework incorporating measures for addressing the main security elements that give rise to content piracy.
- It develops four approaches for the management of an authorised domain, based on the defined generic DRM framework. Each approach incorporates a method for binding a domain to a single owner, ensuring that a single consumer owns and manages all the devices in a domain. This enables binding of content licences to a single owner, thereby limiting illicit content proliferation. The general approach developed in the thesis

could also be useful for various other applications requiring strong user authentication, as it strongly binds consumers to their domains.

1.3 Organisation of the Thesis

The remainder of this thesis is organised as follows.

Chapter 2 and 3 are preliminary chapters. Chapter 2 introduces the security services and mechanisms used throughout this thesis. Chapter 3 provides background information about Trusted Computing technology, as necessary to understand the protocols and security analyses described in this thesis.

Chapter 4 provides an overview of DRM; it gives the main entities and workflow in a typical DRM system, and two examples of rights expression languages. Finally, it provides a comprehensive list of DRM requirements from the points of view of users, content providers and copyright law; this list is then used throughout this thesis to assess and evaluate DRM schemes.

Chapter 5 develops an approach for assessing DRM systems based on the ideal list of DRM requirements, and this approach is then used to analyse six of the most recently discussed schemes for protecting digital assets within authorised domains. Based on this assessment, we provide our own analysis of the main threats underlying the content piracy problem.

Chapter 6 provides a high level architecture for authorised domain management for a DRM system. It gives a set of requirements for devices in an authorised domain, and how such requirements could be satisfied using the Trusted

Computing Group specifications. It then describes a framework incorporating a general workflow designed to solve the content proliferation problem. Finally, it illustrates how to control the membership of a domain in the context of the defined framework.

Chapters 7–10 specify systems implementing the generic authorised domain management framework provided in chapter 6. Chapter 7 describes a means of managing an authorised domain using a master control device that authenticates a domain owner using either a password/PIN or a biometric authentication mechanism. It also presents the pros and cons of these two approaches for user authentication, and gives possible countermeasures to the issues identified. Chapter 8 describes a means of managing an authorised domain using a Trusted Authority, which authenticates a domain owner using his/her payment card details. Chapter 9 describes a means of managing an authorised domain using a mobile device and a mobile network operator, where domain owners are authenticated based on the generic authentication architecture mechanism provided by the mobile network operator. Finally, chapter 10 describes a means of managing an authorised domain using a Trusted Authority, where domain owners are authenticated using location based service mechanisms. Chapters 7–10 also provide analyses of system security requirements, threats, and services for the systems described, and how the schemes help to prevent illicit content proliferation.

Chapter 11 assesses the four proposed schemes given in chapters 7–10. This analysis is based on the ideal DRM requirements list provided in chapter 4.

Finally, chapter 12 gives the conclusions of this thesis, and outlines directions for future work.

Chapter 2

Security Services and Mechanisms

Contents

This chapter introduces the security threats, services, and mechanisms that are relevant to this thesis. This chapter is based on definitions given in a variety of sources, including [26, 46, 50, 51, 79].

2.1 Security Threats

A security threat is a potential violation of the security of a system. The following are generic security threats which could potentially affect the systems proposed in this thesis.

1. *Impersonation*, in which an attacker pretends to be an authorised entity which is entitled to participate in a transaction.
2. *Unauthorised reading*, whereby an attacker obtains sensitive information whilst it is being stored, transferred, or executed.
3. *Data manipulation*, where an attacker inserts, deletes, or reorganises the content of data whilst it is being exchanged, stored, or executed.
4. *Data replay*, where a previous message, in its entirety or in part, is re-transmitted after interception.

2.2 Security Services

To counteract the threats outlined in section 2.1, we identify the following security services.

1. *Authentication*, of which there are two types:
   (a) *Entity authentication* provides assurance in "real time" of the identity of an entity, and that it is currently active in a communication session.
   (b) *Data origin authentication* provides assurance that a given entity was the original source of a message.

2. *Confidentiality* protects a message against being read by an unauthorised entity whilst it is being transferred or stored.

3. *Data integrity* protects a message against alteration by an unauthorised entity whilst it is being transferred or stored.

4. *Message freshness* protects a message against being replayed between communicating parties.

5. *Memory protection* protects data against being read or altered by an unauthorised entity whilst it is being processed [34].

6. *Access control* provides controlled protection for a resource against unauthorised access, including: unauthorised usage, unauthorised disclosure, unauthorised alteration, unauthorised destruction, and unauthorised execution. Access control to a resource is determined by a security policy, typically defined by the resource owner.

2.3 Security Mechanisms

In this section we identify security mechanisms that are used by the schemes described in this thesis to provide the security services described above.

2.3.1 Cryptographic Hash Function

A cryptographic hash function (for example, MD5 [81] or SHA-1 [27]) maps a bit string of variable length to a bit string of fixed length [47]. It must also satisfy the following three properties.

- *Pre-image resistance*: It must be computationally infeasible to find, for a given output, an input which maps to this output.
- *Second pre-image resistance*: It must be computationally infeasible to find, for a given input, a second input which maps to the same output.
- *Collision resistance*: It must be computationally infeasible to find two different inputs which map to the same output.

2.3.2 Symmetric Cryptography

Symmetric cryptographic techniques use a key which must be kept secret. Classes of symmetric cryptographic technique include symmetric encryption algorithms and Message Authentication Code (MAC) schemes, which are briefly outlined in this section; for further details see, for example, [26, 51, 79].

2.3.2.1 Symmetric Encryption

Symmetric encryption techniques (for example, AES [31] or DES [30]) are used to provide confidentiality for data whilst it is stored in a device or whilst it is in transit. Data is encrypted using an encryption key to produce ciphertext. A decryption key is used to decrypt the ciphertext. The encryption and decryption keys are the same or can be easily computed from each other, and need to be known to all parties requiring access to the data.

2.3.2.2 Message Authentication Codes

MACs (for example, CBC-MAC [48] or HMAC [56]) are used to provide data integrity and data origin authentication services. A MAC is computed using a MAC-function that takes two inputs, namely, a secret key and an arbitrary-length message, and outputs a MAC value. A verifier can check the integrity and authenticity of a message by recomputing the MAC value using the same MAC-function and secret key, and comparing it with the value associated with the message.

2.3.2.3 Authenticated Encryption

Authenticated encryption techniques (for example, OCB 2.0 or Key Wrap [51]) can be used to provide data confidentiality, data integrity, and data origin authentication services. A mechanism of this type typically involves either a combination of a MAC algorithm and a symmetric encryption scheme, as outlined above, or uses an encryption algorithm in a special way so that it provides both integrity and confidentiality protection [51].

2.3.3 Asymmetric Cryptography

Asymmetric cryptography involves the use of pairs of related keys, made up of a public key and a private key, where it is practically impossible to deduce the private key from the public key. The private key should be known only by its owner. The public key, however, needs to be made known to all parties requiring to communicate with the holder of the private key, and the authenticity of a public key needs to be guaranteed to any user. This can be achieved, for example,

by using a Public Key Infrastructure (PKI) (see [26], Chapter 13). Asymmetric cryptographic techniques include asymmetric encryption algorithms and digital signature schemes, which are briefly outlined in this section; for further details see, for example, [26, 46, 50, 79].

2.3.3.1 Asymmetric Encryption

Asymmetric encryption techniques (see, for example, ElGamal [28] or RSA [82]) are used to provide data confidentiality. Data is encrypted using a public key to produce ciphertext, which can only be decrypted using the corresponding private key.

In the reminder of this thesis we require asymmetric encryption techniques to provide non-malleability, which means that "*it should be hard to transform a given label/ciphertext pair (L,C) encrypting a plaintext M into a different pair (L′,C′), such that the decryption of C′ with label L′ is related in some "interesting" way to M*" [50]. For the purposes of this definition, a label is "*an octet string that is input to both the encryption and decryption algorithms of an asymmetric cipher, and of a data encapsulation mechanism. A label is public information that is bound to the ciphertext in a non-malleable way*" [50].

2.3.3.2 Digital Signatures

Digital signature techniques (see, for example, DSA [29] or ElGamal [28]) can be used to provide data integrity and data origin authentication services. When using such a mechanism, a sender signs a message using his/her own signing key (the private key) to generate a signature. The sender then sends the message

with the signature to a receiver, who needs to possess a reliable copy of the corresponding verification key (the public key), e.g. as obtained via a PKI. The verification key is used by the receiver to verify the signed message.

2.3.4 Nonces

Nonces provides message freshness. A nonce must be a value that has never been used before, and will never be used again (within a specific security context).

2.3.5 Process Isolation

Process isolation provides protection for a message against being read or altered by an unauthorised entity whilst being executed [34]. This service can be provided using trusted computing technology, discussed in chapter 3.

2.4 Public Key Infrastructures

For the purposes of this thesis, a PKI is the infrastructure necessary to support the distribution of public keys to those entities which need to use them, in such a way that the user of a public key can be sure of its integrity and authenticity. A PKI can be made up of a variety of different elements, depending on the technology and organisational structure used to distribute and guarantee the correctness of public keys. However, by far the most common form of PKI is one based on the generation and distribution of public key certificates.

More specifically, most practical PKIs consist of one or more management entities responsible for generating, distributing and providing ongoing support

for public key certificates. This includes a variety of different types of entity, including, most importantly, Certification Authorities (CAs), which are responsible for generating public key certificates. A CA validates the identity of a user, validates the content of a public key certification request, digitally signs a public key certificate for a validated request, and maintains a certificate revocation mechanism. Users relying on a PKI need to have a trusted copy of the public key of the CA in order to verify public key certificates signed by that CA.

The most widely used format for public key certificates is X.509 [41], an ITU-T (ITU Telecommunication Standardisation) recommendation. An X.509 certificate is issued by a CA to bind a public key to a particular Distinguished Name, or to an Alternative Name such as an e-mail address or a DNS-entry. It is composed of various fields, including: the certified public key, the certificate serial number, the certificate issuer, the certificate validity period (start and expiry dates), the entity associated with the certified public key, and the revocation list URL [41].

A certificate might have been revoked, and hence the status of a certificate may need to be checked prior to use, e.g. by querying an Online Certificate Status Protocol (OCSP) service. An OCSP service ‘‘enables applications to determine the (revocation) state of an identified certificate. OCSP may be used to satisfy some of the operational requirements of providing a timely revocation information and may also be used to obtain additional status information. An OCSP client issues a status request to an OCSP responder and suspends acceptance of the certificate in question until the responder provides a response. This protocol specifies the data that needs to be exchanged between an application checking the status of a certificate and the server providing that status” [70].

Chapter 3

Trusted Computing and DRM

Contents

This chapter provides a very brief introduction to Trusted Computing technology. More specifically it covers the following topics: the importance of Trusted Computing; the relationship between Trusted Computing and DRM; the main building blocks of Trusted Computing as given in Trusted Computing Group specifications; and the mechanisms used for performing platform integrity measurement, reporting, and verification. Finally, this chapter briefly outlines the main challenges for applying TCG specifications to currently available devices, operating systems and applications. The material in this chapter is mainly de-

rived from [33, 84, 91, 92, 93, 94].

3.1 The Importance of Trusted Computing

Users typically have full access to their personal computers (PCs), providing them with full control and flexibility over their own environment. PC data travels between storage devices, memory, CPUs, audio chips, loudspeakers, video chips, and display devices. Data transferred within a PC is subject to potential interception or modification attacks. Also, it could be copied as it passes through the system, enabling the creation of an unlimited number of perfect copies.

When PCs and PC operating systems (OSs) were first designed, security was not a major concern. This is because PCs were designed to work in a standalone mode or within a small workgroup; not in the way they are used today. However, since the early stages of the Internet era, PCs have been connected to larger networks, such as the Internet. This has made them vulnerable to various kinds of attacks, such as viruses, worms, phishing, pharming, etc. Software-only techniques cannot provide a high degree of protection for secret keys stored in a device; for example, Apple FairPlay, which uses software-only techniques, has been hacked multiple times, as discussed in chapter 5. This and other problems raise the need for a trusted computing technology that can enforce policy-neutral access control mechanisms.

3.2 Trusted Computing and DRM

Trusted computing systems are platforms whose state can be remotely tested, and which can be trusted to store security-sensitive data in ways testable by a remote party. Trusted computing technology can enforce access control policies associated with a resource in such a way that a user cannot bypass these policies, whilst maintaining access to the resource.

A number of critiques of the technology (see, for example, [12, 14]) assert that trusted computing is designed to support DRM. However, Kuhlmann and Gehring [57] distinguish between DRM and trusted computing; the following list summarises their arguments.

- *Trusted Computing is not DRM; however, trusted computing offers functionalities that can be used to help build both DRM systems and many other applications which require the verification of the trustworthiness of running machines.*
- *Trusted Computing is policy-neutral and could be used to enforce any access control policy; however, DRM is policy-specific and enforces content owner/distributor policies.*
- *Trusted Computing is not specially protected by copyright laws; however, digital content is protected by copyright laws.*
- *Trusted Computing has a hardware-based off switch; however, DRM has no hardware-based off switch.*
- *Trusted Computing is a standardised technology; however, DRM uses a variety of systems from different vendors.*

- *Trusted Computing can be used to undermine as well as to protect users' privacy; however, DRM might undermine user privacy, as content providers could collect and process information regarding user content consumption, and thereby create profiles for their customers.*

3.3 Trusted Computing Group

The Trusted Computing Group (TCG[1]) was established in April 2003. It is the successor to the Trusted Computing Platform Alliance (TCPA), that was founded in January 1999 by a group of major technology vendors, including AMD, IBM, Intel, HP, Microsoft, and Sun Microsystems. The mission of the TCG is to develop solutions to increase the trustworthiness of computers. TCG compliant platforms are not expensive, and are currently available from a range of PC manufacturers, including Dell, Fujitsu, HP, Intel and Toshiba [35]. In addition, since early 2006, all Intel-based Apple computers are TCG compliant [96].

The TCG specifications require that Trusted Platforms (TPs) have the following functionalities.

- Confidentiality and integrity protection of the OS and underlying hardware.
- Confidentiality and integrity protection of application code and data during execution and storage.
- Confidentiality and integrity protection of data travelling to and from input/output devices.

[1] www.trustedcomputinggroup.org

- Confidentiality, integrity, and authenticity of data travelling between devices and applications.
- Platform attestation to external entities.

3.4 Trusted Platform Module

The TCG specifications [92, 93, 94] require each TP to include an additional inexpensive hardware chip to establish trust in that platform. This chip, which has protected storage and protected capabilities, is referred to as the Trusted Platform Module (TPM). In order to reduce TPM cost, the TCG specifications only require the TPM to be used for functions requiring protected storage and capabilities. Functions that do not require protected storage and capabilities can run using the platform main processor and memory space. TPMs are currently produced by a range of microelectronics manufacturers, including Atmel[2], Broadcom[3], Infineon[4], ST-Microelectronics[5] and Winbond[6].

A TPM incorporates various functional components and features including: I/O; a cryptographic co-processor that supports the following operations: asymmetric key generation, asymmetric encryption/decryption, hashing and random number generation; generation, storage and protection of cryptographic keys; an HMAC engine; a SHA-1 engine; power detection; non-volatile memory; volatile memory; platform configuration registers (PCRs), i.e. shielded locations inside the TPM used to store integrity measurements; and an opt-in component that allows the TPM to be turned on/off, enabled/disabled, activated/deactivated.

[2] www.atmel.com/dyn/resources/prod_documents/doc5010.pdf
[3] www.broadcom.com/press/release.php?id=700509
[4] www.infineon.com/tpm
[5] www.st.com/stonline/products/literature/bd/10926.pdf
[6] www.winbondusa.com/

No remote entity should be able to change the TPM status without either the knowledge of the TPM Owner or the physical presence of an Operator at the platform.

Every TPM has a statistically unique endorsement key pair (EK), which is generated and stored in the TPM at the time of production by the manufacturer. This key can be generated either internally or using an external key generator. It is used only for encryption/decryption purposes. The private decryption endorsement key is known only to the TPM and never revealed outside the TPM. The EK can only be used when assigning TPM ownership, and to support the creation and verification of Attestation Identity Keys (AIKs). AIKs (which are signature key pairs) function as aliases for the TP; they are generated by the TPM, and the public part is included in a certificate known as an Identity Credential, signed by a trusted third party called a privacy certification authority (privacy CA). The identity credential asserts that the (public part of the) AIK belongs to a TP with specified properties, without revealing which TP the key belongs to. Before generating an identity credential, the privacy CA verifies a series of signed credentials belonging to the platform, including the endorsement credential, conformance credential and platform credential. These latter credentials are used to guarantee that an EK belongs to a particular TPM, attest that a TP design meets the TCG specifications, and that a particular platform is an instantiation of a TP design as described in the conformance credentials, respectively. AIKs are used to sign data generated inside the TPM, including the values of PCRs which hold measurements of platform state. AIKs can also be used to sign other keys.

A Privacy CA generates certificates for a TPM, while preserving the privacy of the user of the platform that contains the TPM. The use of a Privacy CA

gives rise to certain operational and privacy concerns, as discussed in [18]. A Privacy CA needs to be as secure as any other certification authority, including those that normally operate off-line. In addition, if a privacy CA and a verifier collude, then they can link requests coming from the same platform. As a result of this latter privacy concern, the TCG has adopted the Direct Anonymous Attestation technique (DAA), which has the following features, as discussed in [18]: *"DAA can be seen as a group signature without the feature that a signature can be opened, i.e., the anonymity is not revocable. Moreover, DAA allows for pseudonyms, i.e., for each signature a user (in agreement with the recipient of the signature) can decide whether or not the signature should be linkable to another signature. DAA furthermore allows for detection of known keys: if the DAA secret keys are extracted from a TPM and published, a verifier can detect that a signature was produced using these secret keys. The scheme is provably secure in the random oracle model under the strong RSA and the decisional Diffie-Hellman assumption".*

Once a TPM has been assigned an owner, it generates a new Storage Root Key pair (SRK), which is used to protect all TPM keys, apart from the EK. The private part of the storage root key pair is stored inside the TPM, and is never disclosed by the TPM. Other TPM objects (key objects or data objects) are protected using keys that are ultimately protected by the SRK in a tree hierarchy.

A TPM can generate two types of keys, known as migratable and non-migratable keys. Migratable keys can be transmitted to other TPs if authorised by both a selected trusted authority and the TPM owner. A non-migratable key, on the other hand, is bound to the TP that created it, and cannot be cloned.

Each object protected by a TPM includes a secret, which is known as *AuthData*. Proving the knowledge of the value of the AuthData associated with an object grants access to that object.

3.5 Establishment of Trust

An important step in the establishment of trust in a TP is to know the origin and history of the components that make-up a TP. Such knowledge can be derived from statements made by trustworthy entities, whose authors stand to lose, for example, their reputation, brand name, revenue, etc., if they do not implement a TP according to the claimed specifications. The following entities are involved in establishing trust.

- A trusted platform module entity (TPME) attests that a particular TPM is genuine by digitally signing an endorsement credential containing the public EK belonging to a particular TPM. The TPME is likely to be the TPM manufacturer.
- A conformance entity (CE) attests that a TP design, i.e. the design of the TPM and other trusted platform building blocks, when integrated into a particular design of platform, meets the TCG specifications. This is achieved by digitally signing conformance credentials. The method of integrating a TPM into a platform must also satisfy the requirements outlined by the TCG.
- A platform entity (PE) guarantees, through the generation and signing of a platform credential, that a particular platform is an instantiation of a TP design, as described in specified conformance credentials. The PE

may be the equipment manufacturer.

- A validation entity (VE), typically a component supplier, produces validation certificates, each of which certifies a software component's integrity measurements. The integrity measurements correspond to a correctly functioning platform component (i.e. a piece of software). These validation certificates are used by a challenger wishing to evaluate the state of a challenged TP.

- A privacy CA is responsible for certifying AIKs, i.e. generating Identity Certificates confirming that a particular AIK belongs to a genuine TP.

3.6 Roots of Trust

A TP incorporates three roots of trust: a root of trust for measurement (RTM), a root of trust for storage (RTS), and a root of trust for reporting (RTR). The TCG defines roots of trust as *"something that we accept as trustworthy because we have to do so for practical purposes and because we accept that its author or creator has properly designed and implemented it. We also accept a Root of Trust because its author or creator stands to lose (e.g. brand name, revenue, legal, etc.) if they did not implement it according to its specifications"* [91]. We next outline the roles of these three roots of trust.

The RTM is a computing engine capable of making reliable integrity measurements (i.e. measurements of the code running on the TP), and is controlled by a particular instruction set, i.e. the core root of trust for measurement (CRTM), which is, ideally, stored inside the TPM. The CRTM is responsible for measuring the first piece of software to be executed during system boot, passes the

measurement result to the RTS that records the result in the TPM PCRs, and then passes control to the next piece of software to be executed, which has a measurement agent (MA) embedded within it. This MA measures the next piece of software to be executed, passes the result to the RTS that records the result in the TPM PCRs, and passes control to the next piece of software to be executed, and so on. MAs are used to build up a chain of trust in the form of a series of integrity measurements. The results of integrity measurements made by the CRTM and MAs are known as measurement events; these are made up of two classes of data: measured values, which are representations of embedded data or program code, and measurement digests, which are hashes of the measured values. The measurement digests are stored in the TPM PCRs. The measurement values are stored in the stored measurement log (SML), which is stored outside the TPM.

The RTS provides confidentiality for, and integrity protection of, keys and data that are held on external untrusted storage devices. The RTS also provides mechanisms to ensure that the release of information only occurs if the current platform state matches that are associated with the stored object. The RTR is a computing engine capable of reporting a TP integrity state, protecting reported values, and establishing a context for attesting to the reported values, as described in the next section.

3.7 Integrity Reporting and Verification

Establishing trust in a TP is based on the mechanisms used for measuring, reporting and verifying platform integrity metrics. TCG defines integrity management as *"the management of component-information throughout the supply*

chain to ensure their integrity (tamper-free state) and also to the management of the runtime integrity of the entire Trusted Platform through the correct management of its components, both at load-time and at runtime" [91]. As described in section 3.6, TP measurements are performed using the RTM, which measures software components running on a TP. The RTS stores these measurements inside TPM shielded locations. Next, the RTR mechanism allows TP measurements to be reliably communicated to an external entity in the form of an integrity report. The integrity report is signed using an AIK private key, and is sent with the appropriate identity credential. This enables a Verifier to be sure that an integrity report is bound to a genuine TPM. The term *measurement* is used in various ways, as described below.

1. *Loadtime measurements* refer to integrity measurements of TP components made whilst the platform is booting-up.

2. *Runtime measurements* refer to integrity measurements of TP components that are generated during the operation of the platform, i.e. after the end of a boot-up sequence.

3. *Reference measurements* refer to a collection of digest values of TP components, each of which must be collected from the component manufacturer. This provides an authoritative source of component integrity information, which can be read by a verifier of the state of a TP.

For example, assume a requestor TP is seeking a service from a verifier. The requestor sends a request to the verifier. The verifier then asks the requestor to perform an integrity measurement of the entire platform (using the requestor's RTM). The requestor returns a platform integrity report to the verifier (using the RTR). The verifier also gets the authoritative information, i.e. the Reference

Measurements, for each component of the requestor's platform from its manufacturer. The integrity metric provider, e.g. the hardware manufacturer or software vendor, makes these Reference Measurements accessible. In this way, the verifier knows both the current integrity-status of the component making-up the Requestor's platform, as well as the source-authenticity of those components (as coming from the manufacturer). The verifier then needs to identify each component of the requestor's platform and compare the reported measurement against the expected reference measurement value (for each component). If the result is positive, the verifier can provide the requested service.

The integrity measurements stored in a TPM's PCRs are also used in the protected storage mechanism. This is achieved by comparing the current PCR values with the intended PCR values stored with the data object. If the two values are consistent, access is then granted and data is unsealed.

3.8 Challenges in TCG Specifications

The TCG specifications, as discussed in this chapter, are based on certain assumptions. Building a system that can satisfy these assumptions using today's hardware devices and operating systems is a technical challenge, and is the subject of ongoing research. The following list, based on that given by Sadeghi [84], summarises these challenges.

- The TCG specifications assume that platform configurations cannot be manipulated after the corresponding hash values have been computed and stored in the TPM's PCRs. Satisfying this assumption requires a secure operating system that is especially designed to consider this requirement.

Currently available operating systems can easily be modified, e.g. by exploiting security bugs.

- As discussed earlier in this chapter, a verifier can determine the trustworthiness of code from hash values (binary measurements of running code). Such a binary based attestation mechanism has the following shortcomings.
  - It reveals information about the platform's hardware and software configuration to a verifier.
  - It allows remote parties to exclude certain system configurations.
  - It requires the verifier to know all possible trusted configurations of all platforms.
  - Most importantly, updates in firmware or software, or hardware migrations, result in changed hash values for the updated components. This, in turn, prevents access to data bound to the previous configuration.

  In principle attestation should only determine whether a system/component configuration has a desired property. Several methods have been proposed to meet this requirement, such as property-based attestation [1, 59, 85], anonymous property-based attestation [22], and semantic remote attestation using language-based trusted virtual machines [38].

- The TCG specifications implicitly require the establishment of secure channels between hardware components. TPM chips integrated into currently available devices are connected to the I/O board with an unprotected interface that can be eavesdropped upon and manipulated [61]. Secure channels between hardware components can be established using cryptographic mechanisms supported by an appropriate PKI.

In [58, 66] many of the above problems are addressed. However, some of the proposed solutions require changes to the TPM specifications. The challenge remains to reduce the complexity of the TPM, but still have appropriate solutions to the problems discussed in this section.

3.9 Summary

Current PCs possess major security vulnerabilities, as their original design did not take into account their connection to large networks. Therefore, there is a need to design a trusted platform for which a verifier, locally or remotely, can verify its running state. A trusted platform complying with the TCG specifications enforces a policy-neutral access control mechanism, which could be used to help build a secure application. For example, trusted platforms could be used to help build a secure DRM system, although a trusted platform is not itself a DRM system.

In this chapter we outlined the main functionality of trusted platforms based on the TCG specifications. The TPM, which is the core component of a TCG compliant platform, can be conveniently integrated into consumer devices as it is not expensive, does not result in increased device size, and does not introduce new vulnerabilities into end user computing equipment. Moreover, in this chapter we outlined the mechanisms behind platform integrity reporting and validation. This chapter also outlines the main challenges underneath the TCG specifications.

Chapter 4

Fundamentals of Digital Rights Management

Contents

This chapter provides an overview of Digital Rights Management (DRM). We introduce the main entities and workflow in a typical DRM system; this model underlies the discussions throughout the remainder of the thesis. We then give some examples of languages that are used to construct licence files. Finally, we develop a comprehensive list of DRM requirements from the points of view of users, content providers and copyright law.

4.1 What is DRM?

DRM can be defined as the technologies that collectively support all stages in the lifecycle of digital content creation, manipulation, distribution and consumption, by preventing illegal copying and allowing the imposition of fees, processing of payments, and protection of principal rights and profits [43].

The problem of piracy of digital content is expected to escalate with time, so finding a solution in the near future is essential [16, 37]. However, for a DRM solution to succeed it should satisfy consumer expectations, and it should not prevent users from freely using content that they have legitimately obtained rights to access. In addition, a DRM system should satisfy requirements imposed by copyright law.

The INformed DIalogue about Consumer Acceptability of DRM Solutions in Europe (INDICARE) project [40] has found that applying 100% protection is not always the objective of content providers, for two main reasons. First, security and convenience are often opposing concepts, and many consumers choose convenience over security. Second, in many cases there are sound business reasons for allowing some users to use pirated copies of content. For example, in the case of software products, businesses may believe that some classes of user may not be able to afford to pay for their product; in such a case not preventing such users from using pirated versions will not significantly affect sales revenue. Indeed, through use of the product the pirate users may become accustomed to it, and, as a result, are both more likely to pay for a legitimate copy when they are able, and less likely to use a competitor product.

4.2 DRM Model

There is no common standard for DRM based solutions, as different DRM vendors employ different systems (see, for example, [69, 62, 74, 88]). However, most DRM systems incorporate five main types of entity: content owners, content distributors, rights issuers, electronic payment processors, and consumers, who are the ultimate users of content. In this section we briefly outline the role of each of these entities.

A **content owner** is the entity which holds the rights to the content, and is the party most interested in protecting and enforcing copyright law. The content owner may be the content creator, or may have purchased rights to the content from the creator. A content owner will typically need to negotiate an agreement with a content distributor and rights issuer in order to establish content usage rules and charges. This enables a content distributor to sell content on behalf of the content owner with some means of obtaining a return.

A **content distributor** is typically responsible for the following functions; this list is based on that given in [88].

- Generating packaged content in a form that is suitable for consumption.
- Uniquely identifying content by associating it with a unique identifier.
- Facilitating the delivery of content offline, e.g. using CDs and DVDs, or online, e.g. using the Internet.
- Supporting an appropriate e-payment method for content usage, e.g. using different tariffs for different types of content consumption and a variety of means for pricing and payments (e.g., including micropayments).

- Facilitation of the customisation of content to the preferences of the consumer, to enable interoperation, and to support multiple content formats in a transparent manner.
- Acting as an intermediary in communications between consumers, content owners, rights issuers and electronic payment processors.

A **rights issuer** is responsible for protecting content from misuse by associating each copy of the content with a licence file, also known as a rights object. This rights object specifies the access rights available to the consumer for the associated content. A rights object syntax and semantics for specifying rights objects needs to be chosen, in the form of a Rights Expression Language (REL), as discussed in section 4.4.

A content distributor and a rights issuer might be a single entity or two separate entities. In the latter case, they need to have an agreement that covers the terms and conditions for issuing content rights objects.

An **Electronic Payment Processor** arranges the transfer of an agreed fee from the consumer to the content distributor. The payment processor software agent is typically either integrated into a content distributor website, or a content distributor redirects consumer browsers to a third party payment processor to handle the required financial transactions.

4.3 DRM Workflow

In this section we describe the workflow in a typical DRM system; we give the description in the context of a scenario for selling content, as outlined in

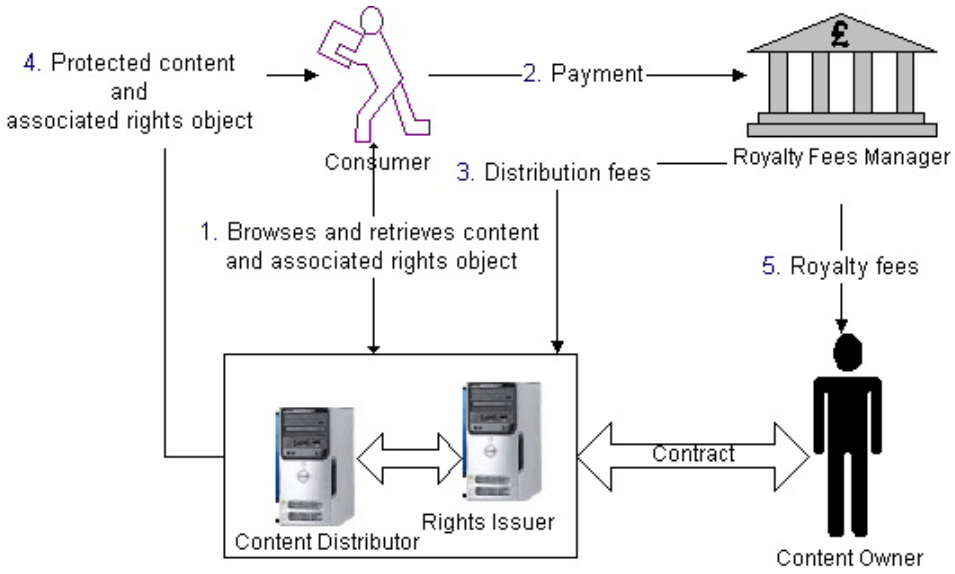


Figure 4.1: A DRM System Workflow

Figure 4.1. A content owner creates content C. If C is not in digital form, then the content owner needs to digitise it. This can be done either by the content owner itself, or, alternatively, by a content distributor (possibly for an agreed fee). The content owner transfers C to the content distributor. The content distributor generates a unique identifier, say id, and binds it to C. In order to track unlawful use, the content distributor may also add a label to the content itself by some means, so that the content is indelibly associated with the licence holder, e.g. using watermarking and/or fingerprinting (see, for example, [77]).

Next, the content distributor protects C, for example by symmetrically encrypting C using a content-specific secret key, which is added to a "trial" rights object R. R is created by the rights issuer, and incorporates: a content identifier id, the content-specific encryption key, and content consumption rules. The content-specific encryption key is protected inside R, for example by encrypting it with a key that is protected with either software or hardware mechanisms at the destination device. Content consumption rules defined in R should be enforced

by a trusted agent running on a consumer device. These rules enable consumers to use C for a trial period, e.g. a consumer can play a song three times and/or cannot forward it to others, the rights expire after one month of first use, or printing is disabled. The content distributor publishes a protected version of C, typically via a web server. A consumer browses content by contacting the content distributor web server, and then sends a request to download a trial version of C, as identified by *id*. The content distributor checks the consumer device to ensure it is trusted enough to enforce the usage rules of the rights object. If the validation succeeds, the content distributor authorises the consumer to download C associated with R. The consumer can then download C and R.

Later on, if the consumer decides to buy a usage licence for C, he/she contacts the content distributor and request a licence for C, as identified by *id*. The content distributor collects payment from the consumer, as outlined in section 4.2, either by redirecting the customer browser to a third party payment processor, or by using a payment service integrated into its web server. At some later time (or perhaps even in advance) a financial transfer is made from the distributor to the owner for the sale of some number of copies of the content. Once payment is collected from the consumer, the content distributor instructs the rights issuer to generate a rights object for this consumer. The rights issuer creates a rights object containing appropriate rights, and then authorises the consumer to download it (note that the consumer does not need to re-download C). Ideally, the consumer should now be able to use C on all devices he/she owns.

However, only encrypted content can be transferred to devices owned by someone else. In this case, the receiver must contact the corresponding rights issuer to either download a trial rights object or buy a usage licence, as described above. This concept is known as **Superdistribution**, as proposed by the Open

Mobile Alliance (OMA) [74]. OMA is a standards body which develops open standards for the mobile phone industry.

OMA defines Superdisribution as *"a mechanism that (1) allows a User to distribute Protected Content to other Devices through potentially insecure channels and (2) enables the User of that Device to obtain a rights object for the superdistributed Protected Content"*. The separation of content from rights objects allows the content to be distributed or downloaded freely. However content cannot be consumed without a valid rights object. The rights object specifies the ways in which the associated content is permitted to be accessed.

4.4 Rights Expression Languages

A Rights Expression Language (REL) is used to specify the syntax and semantics of rights objects, which contain rules governing the use and distribution of associated content. Two commonly discussed RELs are the Open Digital Rights Language (ODRL[1]), and the Extensible Rights Markup Language (XrML[2]). A rights object file consists of content consumption rules written using such a REL. Examples of REL elements are as follows.

- *Agreement* expresses the permissions granted over content.
- *Permission* specifies the rights over a piece of content, such as display, print, play, execute, lend, modify, or delete.
- *Constraints* set restrictions on the permissions by providing fine-grained consumption control for content, e.g. by specifying a count, specifying the

[1] http://odrl.net/
[2] http://www.xrml.org/

number of times a permission is granted, and/or date/time, specifying a time limit for permission. For example, the REL may specify that a piece of music can be played a maximum of 10 times (i.e. a count constraint) for a period of one month (i.e. a time constraint).

4.5 DRM System Requirements

In order for a DRM system to succeed it should be acceptable from the points of view of consumers, content distributors and copyright law. This section presents an ideal list of DRM requirements, which are derived from those given in [16, 69, 40, 80, 86].

1. *Minimum cost*, especially on low-cost devices. Digital rights management is not a feature required by consumers. When purchasing a digital rendering device, a consumer is not interested in paying more for DRM technology. Consequently, the cost of this functionality should be covered by the main beneficiaries of the DRM systems, namely digital content distributors.

2. *Ease of use.* A DRM system should be designed to be transparent from the user's point of view; consumers are likely to reject systems that add extra complexity.

3. *Performance.* The system should work reasonably quickly; an excellent but slow DRM system is likely to be rejected by end users.

4. *Content mobility.* It should be possible to move content between consumer devices without requiring new licences for every device.

5. *Lack of dependence on network infrastructure.* A DRM solution should not require continuous network connectivity across domain devices. Typically, it will not be the case that all user devices are interconnected. For example, car CD players, MP3 players and other mobile devices are unlikely to be permanently interconnected.

6. *Lack of dependence on a secure clock.* A DRM system should not require the presence of secure clocks in devices. Some devices, such as CD players, do not always have an integrated clock. In addition, adding clocks to all devices increases system cost.

7. *Cryptographic robustness.* The cryptographic algorithms used in the system should be cryptographically strong, to provide robust protection against attacks such as brute force, dictionary analysis, reverse engineering, etc.

8. *Ease of recovery.* Recovering after a DRM system failure should be convenient and incur minimal cost. System failures could arise both accidentally and as a result of malicious software (e.g. viruses).

9. *Robust content protection.* Content owners need to be assured that their digital assets are protected, in order to be convinced to release their content to end user devices. Satisfying this requirement automatically implies satisfying requirement (7).

10. *Flexible rights structure.* Content providers need to be able to control content consumption based on licence files provided by a rights issuer. The licence file format should support flexible content management; for example a rights issuer should be able to specify content expiry time, how frequently and how long content can be used, and if content can be lent or re-sold, etc.

11. *Simple key management.* A DRM solution will typically use cryptographic keys to protect content while it is stored and transferred. Key management — including creation, update, validation and revocation of keys — should be easy to use and almost transparent to consumers, who generally have very limited knowledge of the subject.

12. *Flexible Revocation Mechanism.* If required, a key revocation mechanism needs to be designed in such a way that it does not require all devices to be connected to the network. Most devices are unlikely to be connected to the network all the time. A key revocation mechanism should be capable of adapting to varying device characteristics and network connectivity. In addition, a revocation list should be accessible, and should not require the download of large volume of data into consumer devices.

In addition to the above requirements, additional requirements can be derived from copyright law; we summarise those given in [40].

13. *Access to and Usage of Content.* A DRM system should respect the usage expectations that consumers have, or that are given by copyright laws, such as "fair use". Fair use is a doctrine in United States copyright law[3], which means[4] *Notwithstanding the provisions of sections 106 and 106A, the fair use of a copyrighted work, including such use by reproduction in copies or phonorecords or by any other means specified by that section, for purposes such as criticism, comment, news reporting, teaching (including multiple copies for classroom use), scholarship, or research, is not an infringement of copyright. In determining whether the use made of a work in*

[3] http://www.copyright.gov/title17/92chap1.html

[4] The fair use definition is extracted from United States copyright law, which can be found at http://www.copyright.gov/title17/92chap1.html

any particular case is a fair use the factors to be considered shall include:

(1) the purpose and character of the use, including whether such use is of a commercial nature or is for nonprofit educational purposes;

(2) the nature of the copyrighted work;

(3) the amount and substantiality of the portion used in relation to the copyrighted work as a whole; and

(4) the effect of the use upon the potential market for or value of the copyrighted work. The fact that a work is unpublished shall not itself bar a finding of fair use if such finding is made upon consideration of all the above factors.

14. *Consumer Privacy.* A DRM system may enable content providers to monitor private consumption of content, create reports of consumption, and profile users. In addition, some revocation mechanisms only work if intensive monitoring of consumer use of devices and content is possible. As a result, privacy protection for end users is a serious issue. Implementing anonymous access can reduce the impact of this problem. In addition, DRM systems should comply with data protection legislation in the relevant countries (e.g. the UK Data Protection Act[5]).

15. *Interoperability.* Content should be accessible on different platforms and devices belonging to the same owner, and a DRM system should avoid platform lock-in.

16. *Security and Hardware Issues.* DRM system software should not limit the use of other protection software on consumer computers. In general, consumers should not be required to use immature DRM technology. Also, a DRM system should not introduce new vulnerabilities into end user

[5]http://www.opsi.gov.uk/ACTS/acts1998/19980029.htm

computing equipment. In addition, a DRM system must enable consumers to set their own policy and level of security for their machines.

4.6 Summary

Currently there are many DRM systems, which do not adhere to any single standard. Nevertheless, these systems have certain common components, and in most cases have much the same workflow. In this chapter we have described the basic components of a typical DRM system. We started by defining what we mean by DRM. We then defined the main roles in a DRM system, and described the interactions between them. Next, we provided two examples of rights expression languages, used to express rules governing content use and management.

For a DRM-based solution to have a future depends not only on the acceptance of the system by content owners, but also, and probably most importantly, its acceptance by consumers. Hence, in this chapter we have proposed a list of requirements for an 'ideal' DRM system from the points of view of users, content owners, and copyright law. This list covers the fundamental requirements that need to be met if a DRM system is to be successful. In the remainder of this thesis we use this list to assess both existing and novel DRM schemes.

Chapter 5

Authorised Domain Content Protection Systems

Contents

This chapter assesses DRM systems using the ideal list of DRM requirements given in chapter 4; we analyse six of the most widely discussed schemes for protecting digital assets within authorised domains. Based on this analysis, we provide our own definition of content piracy, and conclude that all the analysed schemes have problems in addressing the fundamental DRM requirements. In addition, all the analysed schemes have major security and usability limitations in addressing the authorised domain concept. Most of the material in this chapter was previously published in [8].

5.1 Analysis of Existing Schemes

A number of schemes have been proposed to try to solve the problem of protecting proprietary content in personal networks. In this chapter we summarise and analyse six of the most widely discussed such schemes. The primary criterion that was used to select these schemes was whether or not they implement the concept of an authorised domain. There are a large number of other DRM schemes; however, many such schemes do not address the authorised domain concept, and only focus on binding a licence to a single device. Such schemes are not considered here, as they do not address the core theme of this thesis. Nevertheless, such schemes (including some of the schemes discussed in this chapter) could be integrated with the systems proposed in later chapters, to provide means for downloading content from content distributors to an authorised domain.

5.1.1 OMA DRM

The OMA DRM v2 system was proposed by the Open Mobile Alliance (OMA) [74]. It provides mechanisms for secure authentication of trusted DRM agents running on consumer devices; these agents are trusted to enforce the DRM policies associated with the content they render. In addition, it provides secure packaging of content, and transfer of content and usage rights to the trusted DRM agent.

5.1.1.1 System Overview

A part of the OMA specifications that relates to this thesis is that concerned with the "Domain Concept". In each domain, copyrighted content can be rendered on any device in that domain, and only one licence per domain is required.

Domain owners must register all their devices with all the rights issuers (RIs) whose content they wish to access. The device may join multiple domains managed by one or more RIs. A domain is associated with a unique domain identifier and a domain generation number.

Each RI is in charge of creating domains, managing domain keys, and controlling which and how many devices are included in a domain. Once a device has successfully joined a domain, the RI sends to the device the domain key, the domain identifier, and the domain expiry time. If a domain key has been compromised or a device is revoked, then the RI increments the relevant domain generation number and changes the domain key.

The system has the following operational requirements:

1. Each device has a public/private key pair certified by a certification authority (CA), and which is installed at the time of manufacture. In addition, the integrity of the key pair and the confidentiality of the private key must be protected when stored on the device.
2. Each RI has a public/private key pair certified by the CA.
3. The CA must provides an online certificate status protocol (OCSP) service, [70].

Content is transferred from distributors to a domain as follows:

1. A content distributor packages content in a secure content container using the DRM Content Format (DCF). Each item of content is encrypted using a secret content encryption key (CEK).

2. The RI generates a rights object that contain the permissions, constraints and CEK associated with the corresponding item of content. The rights object is cryptographically bound to the target domain by encrypting the CEK using the domain secret key, and then signing the rights object using the RI signing key.

3. The rights object and the DCF can be delivered to the target device either together or separately.

5.1.1.2 Analysis

We now analyse the OMA DRM scheme using the list of requirements given in section 4.5.

1. *Minimum cost.* Each device needs to securely store the domain keys, domain identifiers and domain expiry time provided by the RI for each domain that it joins. Extra secure storage is needed to store these keys, which potentially increases the cost and complexity of a device.

   Each RI is required to define domains, manage domain keys, and control which and how many devices are included in a domain. This in turn increases the overall complexity and maintenance costs imposed on RIs.

2. *Ease of use.* In order for domain devices to use all content in a domain,

they must join all the relevant RIs. This is not user friendly, and makes administration complex from the user's perspective.

If a device in a domain is hacked, new domain keys are generated by all RIs from whom content has been obtained. These keys must be transferred to all registered domain devices (except, of course, for revoked devices). This means that the domain owner must update the keys on all devices for all RIs, which is not user friendly.

3. *Performance.* Content is encrypted using a secret key, which is itself encrypted using the domain key associated with the corresponding RI. This means that the bulk of encryption is done using a symmetric algorithm, that is typically much faster than an asymmetric algorithm.

4. *Content mobility.* Each item of content is encrypted using a key stored inside an associated rights object, which is itself encrypted using the domain key associated with the corresponding RI. Therefore, content can be freely moved between domain devices, as long as they are registered with the RIs who have provided the content.

5. *Lack of dependence on network infrastructure.* Network key distribution requires all devices to be connected to all the RIs which have provided content, but not simultaneously.

6. *Lack of dependence on a secure clock.* Nonces are used to provide message timeliness.

7. *Cryptographic robustness.* OMA uses the RSA-KEM-KWS asymmetric encryption scheme, as defined in ISO/IEC 18033-2 [50].

8. *Ease of recovery.* The case of hacking of a device is addressed in point (2). Backup and recovery of domain keys does not need to be done by the

consumer, as all domain keys are stored by RIs. However, recovery does require connecting all domain devices to all the RIs from which content has been obtained.

9. *Robust content protection.* There is no mechanism to control who owns the devices assigned to a domain. For example, a set of devices belonging to a number of different users might join domain A, and the same devices could also join domain B. This means that a device owner could access protected content in unauthorised ways by adding the device to multiple domains.

10. *Flexible rights structure.* A rights object containing content usage rules is associated with each piece of content.

11. *Simple key management.* This requirement is addressed in points (1) and (2).

12. *Flexible Revocation Mechanism.* An OCSP server can be queried to check whether device certificates have been revoked.

13. *Access to and Usage of Content.* This requirement is not addressed.

14. *Consumer Privacy.* All RIs store sensitive information about consumer domains, such as device public keys and digital content usage. This enables RIs to track the usage patterns of domain owners, which potentially raises major privacy concerns.

15. *Interoperability.* The scheme can run on a variety of platforms.

16. *Security and Hardware Issues.* The scheme does not interfere with device security settings.

5.1.2 eXtensible Content Protection (xCP)

The eXtensible Content Protection (xCP) scheme was proposed by IBM [44]. This scheme is designed to enable proprietary digital content to move freely around a home network, without the need to purchase separate licences for each device.

5.1.2.1 System Overview

The xCP content protection mechanism uses a technique known as broadcast encryption, proposed by Fiat and Naor in 1994 [32]. Broadcast encryption allows a set of devices to share a common secret sent by a broadcast centre to a group of privileged devices, using a one-way communication method. Broadcast encryption is based on symmetric encryption, and thus may be up to 1000 times less resource expensive than public key encryption techniques; however, it requires more storage at both the transmission centre and on user devices [64].

In xCP, only compliant devices (e.g. players, recorders) can join a consumer domain, where compliant devices are those that always enforce the DRM policies associated with the content they render. Each piece of content is encrypted with a unique key called a title key. Title keys are encrypted using a domain Master Key (MK). The MK is calculated as a cryptographic hash of the media key, network binding ID, and network authorisation table.

The network binding ID is generated by the first device that establishes a consumer domain. The network authorisation table is a file designed to protect against man-in-the-middle attacks [44]. Adding a new device to a domain

changes the network authorisation table, which in turns changes the MK.

A Media Key Block (MKB) is a table of encrypted values that is provided by licensed media manufacturers. A compliant device is given a unique set of keys by licensed media manufacturers, which enables it to calculate the media key from the MKB by performing a complex sequence of mathematical operations. Whenever a device is revoked (i.e. becomes non-compliant), licensed media manufacturers distribute a new MKB to exclude the revoked device's set of keys. As a result, a non-compliant device will be unable to calculate the media key. Whenever a new MKB is introduced into a personal domain, the media key changes, which in turns changes the MK.

When a device is instructed to join a consumer domain, it indicates its type, calculates the media key, and then computes a Message Authentication Code (MAC) using the media key as the MAC algorithm secret key. The domain authoriser checks the MAC, checks the network authorisation table, and checks whether the maximum number of allowed devices has been reached. If the device is accepted, it is sent the MK. As a result, at this point all the content title keys need to be re-encrypted.

5.1.2.2 Analysis

We now analyse the xCP scheme using the list of requirements given in section 4.5.

1. *Minimum cost.* The broadcast encryption protocol requires a licensing agency to produce the MKB and assign device keys [64]. Moreover, devices must be capable of processing the MKB to produce the MK. Each device

in the domain requires tamper-resistant storage to store the MK and the set of keys assigned by licensed media manufacturers. These requirements have the effect of increasing the overall cost of the system.

2. *Ease of use.* Limiting the number of devices that can be added to a domain makes the system less flexible for the consumer, who may not be able to add as many devices as he/she might wish. More specifically, it is potentially very inconvenient for a consumer who already has the maximum number of devices in his/her domain, if he/she is required to remove a device from the domain before a new device can be added.

3. *Performance.* The MKB is a large data structure, and hence moving it between devices and using it to generate the media key imposes a significant overhead, especially on devices that have limited capabilities. Every time the domain membership changes, or a new MKB is released, the MK must be changed. Consequently, the content title keys will need to be re-encrypted with the new MK, which means that all content and associated title keys must be tracked. One possibility is to have a database on each device showing the location of all encrypted content. This imposes additional administrative, storage and processing costs. This could result in a significant overhead if the number of items of content is large.

4. *Content mobility.* Each piece of content is encrypted with a content title key, which is encrypted with the MK that is shared between domain devices. Hence, content can be rendered on all domain devices.

5. *Lack of dependence on network infrastructure.* Every time the domain membership changes, or a new MKB is released, the MK must be changed. All devices must be online to obtain the new MK. It is potentially difficult for some devices to be online (e.g., a car CD Player).

6. *Lack of dependence on a secure clock.* The xCP scheme does not require a secure clock.

7. *Cryptographic robustness.* The published description of the scheme [44] does not provide enough information to evaluate this requirement. For example, the encryption algorithm and the MK and content title key lengths are not specified.

8. *Ease of recovery.* If the system is hacked, a new MKB is released that excludes keys for non-compliant devices. However, a method for recovery of domain keys and content in the event of a system failure has not been proposed.

9. *Robust content protection.* Devices in a domain exchange content by encrypting it with the content title key, which is then encrypted with the MK. There is no binding between the MK and a domain owner. This means that a device not owned by the domain owner could be added to the domain, as long as this device has not been revoked and the maximum number of devices in the domain has not exceeded a pre-specified limit.

   It is not clear from the description of the scheme how domain keys are stored on domain devices. If the domain keys are not protected, then this could enable the MK to be revealed, which would enable unauthorised access to unprotected content.

10. *Flexible rights structure.* This requirement is not addressed.

11. *Simple key management.* Key management is transparent to users.

12. *Flexible Revocation Mechanism.* This requirement is discussed in point (8).

13. *Access to and Usage of Content.* This requirement is not addressed.

14. *Consumer Privacy.* It is not clear whether this requirement is addressed.

15. *Interoperability.* The scheme can run on a variety of platforms.

16. *Security and Hardware Issues.* The scheme does not interfere with device security settings.

5.1.3 SmartRight

The SmartRight scheme [89] is a copy protection system for digital home networks that is combined with a Conditional Access System (CAS), i.e. a DRM system. SmartRight security involves use of a smart card that is attached to presentation functionality, and that enables or disables de-scrambling of digital content transferred from the producer to the consumer. This presentation functionality is used to access the received digital content.

5.1.3.1 System Overview

The SmartRight system has two protection domains: the first domain protects content while it is being delivered to the Personal Private Network (PPN), while the second domain protects content after it enters the PPN. The operation of the first domain is outside the scope of this thesis. The second domain has three main functionalities: protection of content while it is being accessed, protection of content while it is being presented and exported to other systems in the PPN, and protection of content while it is stored in the PPN devices. The three sets of functionality could coexist on a single device or be implemented separately on two or more device(s). The SmartRight system uses smart cards that hold the necessary secret keys. Two types of smart cards are available: Converter

cards for the content access role, and Terminal cards for the presentation role.

Each Terminal card holds a card-specific unique public/private key pair, the SmartRight certification authority public key, and a secret network key. The network key is generated using a random number generator by the first device joining the PPN that possesses presentation functionality. The network key is securely transmitted to each Terminal card joining the PPN, in such a way that the network key is transmitted only by the most recently installed Terminal card. It is not clear from [89] whether the network key is fixed, or is updated every time a Terminal card is added to, or removed from, the PPN; and thus, in section 5.1.3.2 we analyse the pros and cons of the scheme both if the network key is fixed and if the network key is changeable. Content cannot be shared between different PPNs, as each PPN has a different network key.

SmartRight domain control involves limiting the number of Terminal cards joining a PPN. This is controlled using a counter that is stored in each Terminal card and that is initialised to the default maximum value. Every time a new Terminal card joins the PPN, the value of the counter is decremented by one, and the new counter value is transferred to the joining Terminal card. When the value of the counter is zero, no more Terminal cards can join the PPN.

A data structure called the Local Enforcement Copy Management (LECM), carries information relevant to content protection and usage.

For each item of locally held content, the LECM holds a distinct key (the LECM key), generated using a random number generator. The LECM key is encrypted with the network key, and is used to encrypt content-specific de-scrambling information used to access content, and, in addition, to encrypt a specific infor-

mation used to handle view-only content. Whenever a new Terminal card joins the PPN, the LECM key is changed.

A LECM contains the following information for each item of content: version information, content type (video, audio), SmartRight usage state (copy-free, private-copy or view-only), content protection rules, the encrypted value of the LECM key, content de-scrambling information, and specific information used to handle view-only content.

The presentation functionality extracts the LECM from the stream and passes it to the Terminal card. The Terminal card extracts the encrypted LECM key from the LECM and decrypts it using the PPN network key. The LECM key is used to decrypt the content de-scrambling information. To render content, the Terminal card securely transmits the de-scrambling information to the presentation functionality.

5.1.3.2 Analysis

We now analyse the SmartRight scheme using the list of requirements given in section 4.5.

1. *Minimum cost.* In order for a device to join a PPN, it must be SmartRight device compatible (in particular it must possess a smart card reader) and needs to be equipped with a Terminal module smart card and/or a Converter module smart card. These requirements increase the total system cost, especially the costs of using and maintaining smart cards. Moreover, it may be inconvenient to add smart card readers to small devices. If the system is hacked, the current smart cards must be replaced [89], which is

a potentially expensive and time-consuming process.

2. *Ease of use.* A user must have converter smart card(s) in order to add new content to the PPN. In addition, he/she needs to have multiple Terminal smart card(s) to render digital content on a SmartRight device. It may not be convenient for a user to manage a smart card for each device he/she owns.

   Limiting the number of Terminal cards that can be added to a domain using the PPN counter makes the system inflexible for consumers, especially those having many devices. In addition, once the value of the PPN counter reaches zero, it cannot be increased, and the only means of adding devices to the PPN is by reinitialising the domain. This would mean that all currently held content, encrypted with the old network key, would need to be re-downloaded, and a new network key generated.

   The network key is stored on the Terminal card. If all the Terminal cards are lost or fail, then all existing content will be unusable, and will need to be downloaded again from content distributor sites.

   The network key must be transferred by the Terminal card that most recently joined the network. If the end user forgets which Terminal card most recently joined the domain, then he/she must try all Terminal cards. This could pose a serious usability issue in networks with a large number of Terminal cards.

3. *Performance.* All the existing LECM keys are changed every time a Terminal card joins or leaves the network. Therefore, the greater the number of items of content in a domain, the longer it will take to encrypt all the new LECM keys with the network key.

   Because all LECM keys must be changed, this also means that all content

in a domain must be tracked, as a LECM key is stored encrypted inside a content-specific LECM data structure stored with the content. One possibility is to have a database on each device showing the location of all encrypted content. This imposes extra administrative, storage and processing costs.

4. *Content mobility.* Each piece of content is protected with an LECM key. The LECM key is encrypted with the network key that is shared by domain devices. Hence, content can be rendered on all domain devices.

5. *Lack of dependence on network infrastructure.* Network key distribution requires all Terminal cards to be connected to a local network, but not simultaneously.

6. *Lack of dependence on a secure clock.* This requirement is not discussed in the SmartRight specifications.

7. *Cryptographic robustness.* SmartRight uses RSA with a 1024-bit key.

8. *Ease of recovery.* A backup and recovery policy for network keys and content has not been proposed. If all Terminal cards are lost or fail, then all existing content will be unusable, and will need to be downloaded again from content distributor sites. Moreover, if the system is hacked, the current smart cards must be replaced [89], which is a potentially expensive and time-consuming process.

9. *Robust content protection.* There is no binding between the network key and the domain owner. This means that any device can be added to a domain, as long as the device has not been revoked and the value of the PPN counter is not zero. It is not clear from the SmartRight documentation whether or not the network key is changed when a device is added or removed. Assuming that a network key is fixed for the life of a domain:

(a) If a Terminal card is revoked, then it will still be able to access newly delivered content by eavesdropping on the network connection.

(b) It is suggested in [89] that the LECM key can be changed whenever a device is added to, or removed from, a domain. The network key is used to encrypt the LECM key, and so, given the assumption that the network key is fixed, devices removed from a domain can still decrypt the updated LECM key and access content. Therefore there is no point in changing the LECM keys.

10. *Flexible rights structure.* The LECM contains content protection rules. However, it is not clear from the description of the scheme how flexible these rules are.

11. *Simple key management.* Although the PPN network key management system is transparent to an end user, the key can only be transmitted by the Terminal card most recently added to the network. If this Terminal card is lost or stolen, then no devices can be added to the network.

12. *Flexible Revocation Mechanism.* It is claimed that the scheme supports the revocation of terminal cards, presentation functionality, and the entire PPN. However, [89] does not described how this can be achieved.

13. *Access to and Usage of Content.* This requirement is not addressed.

14. *Consumer Privacy.* It is not clear from the description of the scheme whether consumer privacy is protected.

15. *Interoperability.* The scheme can run on any platform with a smart card reader.

16. *Security and Hardware Issues.* The scheme does not interfere with device security settings.

5.1.4 Apple Fairplay

The Apple Fairplay[1] system was proposed by Apple Inc. [13]. It provides a means to protect digital music and video files bought from Apple's iTunes Music Store using software only mechanisms. It also allows a consumer to define a domain consisting of five computers and an unlimited number of iPods[2], within which the consumer's content can be freely used.

5.1.4.1 System Overview

The Apple Fairplay system is only concerned with digital music and video files, which must be bought from Apple's iTunes Music Store (iTMS), and can only run on iPods and on platforms running one of two operating systems, i.e. Mac OS X and Microsoft Windows.

The process of adding the first device to a consumer domain is performed using a copy of the iTunes application running on the joining device (iTunes is an apple proprietary digital media player application, which plays and manages digital music and video files). In order to add a device to a domain iTunes sends a request to the Apple server, which includes a unique identifier for the joining device. The Apple server then issues an Apple-ID/password that can be used to authenticate the user to the server, enabling the user to manage domain membership. Adding more devices to a domain follows the same procedure; however, the user is authenticated using the provided Apple-ID/password and, in addition, Apple restricts the total number of devices that can simultaneously be in a domain to five computers and an unlimited number of iPods.

[1] http://en.wikipedia.org/wiki/FairPlay
[2] http://en.wikipedia.org/wiki/IPod

Removing a device from a domain requires the iTunes application to contact the Apple server. It requests the removal of a device by specifying the device unique identifier that was used in the joining request. Once the Apple server has removed the unique identifier from its list, iTunes removes the content-protection keys from the device.

When a consumer buy an audio stream from the iTMS, the audio stream is protected using a content-specific Master Key, which is generated by the iTMS. iTunes then generates a Master Key specific user key, which is used to encrypt the content-specific Master Key. The encrypted Master Key is then associated with the encrypted content. The user key is protected using a key hardcoded inside the iTunes application. The user key is also transferred to the Apple server, and, when a device joins a domain, the Apple server transfers all the user keys to that device. Every time a consumer downloads a protected audio stream, iTunes generates a new user key to protect the audio stream Master Key. Content can be backed up into a standard Audio CD in unencrypted form.

5.1.4.2 Analysis

We now analyse the Apple Fairplay scheme using the list of requirements given in section 4.5.

1. *Minimum cost.* The proposed scheme protects domain secrets using a software only technique, which is cheaper than using hardware mechanisms.

2. *Ease of use.* The domain size is restricted to five computers; this is potentially very inconvenient for a consumer who already has the maximum number of computers in his/her domain, since he/she will be required to

remove a computer from the domain before a new one can be added.

3. *Performance.* Symmetric encryption is used to protect audio streams on consumer devices, which is much faster than using asymmetric encryption. This should help increase system performance. However, each protected audio stream is encrypted with a content-specific Master Key, which is itself encrypted with a user key. The user key is protected with a key hard coded inside iTunes; i.e. to use an audio stream, the user key needs to be decrypted, which is itself used to decrypt the Master Key that is in turn used to decrypt the content.

4. *Content mobility.* User keys, which are used to protect domain audio streams, are stored inside the server. These keys are transferred from the Apple server to devices joining a domain, which ensures content mobility within each domain. However, content can only be accessed via iPods or platforms running either Mac OS X or Microsoft Windows.

5. *Lack of dependence on network infrastructure.* Every time a new audio stream is bought, a new user key is generated by iTunes and transferred to the Apple server. Therefore, in order for all other devices to use that content, they must be connected online to obtain the updated user key list, which may be inconvenient for some devices.

6. *Lack of dependence on a secure clock.* This requirement is not discussed in the Apple Fairplay specifications.

7. *Cryptographic robustness.* This requirement is not discussed in the Apple Fairplay specifications.

8. *Ease of recovery.* Content can be backed up unencrypted to a CD, as described by Carden [20].

9. *Robust content protection.* This scheme fails to bind devices in a domain to the domain owner. Adding a device to a domain simply requires the domain owner to be authenticated using an Apple-ID and password that can be shared with others (there is no binding between the domain key and the domain owner).

   The domain key is protected using software techniques only, unlike other schemes that use hardware measures, and the software protection has been hacked many times; see, for example, the Hymn project[3]. In addition, content can be backed up to a standard CD in unencrypted form, or transported elsewhere via email or FTP, enabling content proliferation, as described by Carden [20].

10. *Flexible rights structure.* This requirement is not addressed in this scheme; a consumer device either has full access to content or no access at all.

11. *Simple key management.* Key management is based on sharing a list of secret keys between all devices in a domain. Every time a consumer buys new content, a user key is generated to protect the added content. This key is also transferred to the Apple server so that it can be used by other devices in the same domain. This requires all devices to be online to retrieve the new key. In addition, if the number of pieces of content is large, this might cause a key management problem.

12. *Flexible Revocation Mechanism.* This requirement is not addressed in this scheme.

13. *Access to and Usage of Content.* This requirement is not addressed.

14. *Consumer Privacy.* The Apple server and iTMS could jointly track consumer content usage, and this raises a potentially major privacy issue.

[3] http://hymn-project.org

15. *Interoperability.* Only iPod devices and platforms running Mac OS X or Microsoft Windows can join a domain, limiting interoperability. As stated by Rowell [83], *"Apple Fairplay is allowing Apple to lock iPod owners into its proprietary store"*.

16. *Security and Hardware Issues.* The scheme does not interfere with device security settings.

5.1.5 DRM in a 3G Mobile Phone and Beyond

A DRM system for a 3G mobile phone was proposed by Dabbish and Messerges [69]. The proposed scheme requires the mobile phones to possess the features of a trusted system, as defined by Stefik [87].

5.1.5.1 System Overview

The part of the Dabbish and Messerges [69] scheme that relates to this thesis is the "Family Domain" concept, which allows a consumer to transfer copyrighted digital content between his/her own devices without requiring further licences. The system requires the establishment of a trusted Domain Authority (DA) that has the following functionality.

1. It registers consumer devices, to allow protected digital content to be moved freely between registered devices in the same domain (owned by the same consumer).

2. It creates and installs a domain unique public/private key pair in a device immediately after registration. This enables registered devices to have

access to all content in the family domain.

3. In order to control family domain membership, DA enforces registration policies that limit the number of devices in the domain and the number of times a device can join and leave a domain.

4. It detects fraud by tracking devices that join or leave registered domains. For example, if a device joins and leaves multiple domains frequently, then this gives an indication of possible system abuse.

5. Devices are removed from a domain by creating a new domain public/private key pair, and then installing this new key pair on all the remaining devices in that domain.

Each item of content is encrypted using a unique secret key, and is associated with a licence file that holds both the associated secret key and the consumption rules for this content. Licence files are encrypted using the domain public key, which is installed by the DA on all registered devices, as described above.

Dabbish and Messerges [69] suggest using a password shared between the domain owner and the DA in order to control domain membership. Users could potentially share domain passwords to add non-family members to a domain; to reduce the scope of this problem it is suggested that each domain password is bound to user private information, or an ability to spend money (thereby providing a disincentive to password sharing).

5.1.5.2 Analysis

We now analyse the DRM in a 3G Mobile Phone and Beyond scheme using the list of requirements given in section 4.5.

1. *Minimum cost.* The use of a DA requires the establishment and maintenance of a complex infrastructure that increases the overall cost and complexity of the system.

2. *Ease of use.* Every time a device is revoked or removed from a domain, the domain key pair has to be changed, which requires the user to connect all devices to the Internet in order to install the new domain key. For devices that do not have direct access to the Internet, Dabbish and Messerges [69] suggest using other devices to act as a proxy while registering the new device. Moreover, all domain content licence keys must be re-encrypted with the new domain key. Users are likely to find such a procedure inconvenient.

3. *Performance.* As discussed immediately above, every time a device is revoked or removed from a domain, a new key pair is generated, which necessitates re-encryption of all content encryption keys. These keys are stored within domain devices, and must therefore be tracked. One possibility is to have a database on each device showing the location of all encrypted content. This requires extra administration, in addition to requiring more storage and processing. This results in a significant overhead if a large quantity of content is present.

   A Family Domain is protected against abuse in two ways:

   (a) The frequency with which a device is added to or removed from domains is monitored. This requires a potentially complex and sophisticated infrastructure, that processes and records potentially huge numbers of events. Moreover, before authorising a device to join a domain, the DA must parse the history log files to ensure the device is not abusing the system. This is likely to adversely affect system

performance.

(b) Domain creation is bound to owner private information or the ability to spend money. One major problem with this approach is user privacy, as the DA must hold confidential user information. In addition, it does not prevent abuse of the system, as the owner can add devices that he/she does not own to his/her domain without giving the password to other entities.

4. *Content mobility.* This requirement is met by the use of a unique public/private key pair shared by domain devices, which is used to encrypt content encryption keys.

5. *Lack of dependence on network infrastructure.* Network key distribution requires all devices to be connected to the Internet, but not simultaneously.

6. *Lack of dependence on a secure clock.* It is not clear whether or not the system requires a clock.

7. *Cryptographic robustness.* Content is encrypted using a symmetric algorithm, such as AES. The secret key used is stored in a licence file, which is itself encrypted using a public-key scheme, such as RSA.

8. *Ease of recovery.* Part of this requirement is discussed in point (2). In addition, key backup and recovery are handled by the DA.

9. *Robust content protection.* There is no binding between a domain public/private key and the domain owner. This means that any device could be added to the domain, regardless of ownership, as long as this device has not been revoked and the DA authorises it.

10. *Flexible rights structure.* The scheme associates a licence file with every item of content. This file can be customised to hold content usage rules.

11. *Simple key management.* The DA is in charge of creating, installing and managing the public/private key pair in all domain registered devices. This creates a significant key management overhead, including the need for a secure infrastructure for creating, storing, archiving and transferring domain keys. Moreover, key revocation requires domain devices to be connected online, as described in point (2).

12. *Flexible Revocation Mechanism.* This point is addressed in point (2).

13. *Access to and Usage of Content.* This requirement is not addressed.

14. *Consumer Privacy.* This requirement is discussed in point (3-b).

15. *Interoperability.* A platform can join a domain as long as it possesses the features of a trusted system, as defined by Stefik [87].

16. *Security and Hardware Issues* The proposed scheme requires devices to have trusted system features, as defined by Stefik [87].

5.1.6 DRM Security Architecture for Home Networks

The DRM Security Architecture for Home Networks was proposed by Popescu, Kamperman, Crispo, and Tanenbaum [80]. This scheme includes a technique for managing DRM in a home network. The main idea is to allow devices to establish dynamic groups called authorised domains (ADs), where copyrighted content can be freely copied between devices in a single AD.

5.1.6.1 System Overview

The system model involves the following main entities: content providers, consumer electronics manufacturers, and a licensing organisation that certifies con-

sumer electronics manufacturers and issues device revocation information. Further details of the definition and functionality of the system components are given in [80].

An AD consists of the following main components:

- Domain Manager: An AD must have a device with domain manager functionality that is in charge of keeping track of all devices in the AD, adding devices to the AD, and removing devices from the AD.
- Content Manager: An AD can have more than one device with content manager functionality, as used to download content into the AD by interacting with content providers. Different providers may choose different types of device to supply their content.
- Compliant Devices: are devices that are certified by the consumer electronics manufacturers, which are trusted to enforce proprietary content usage rules.

During the manufacturing process, each compliant device is given a public/private key pair, certified by the consumer electronics manufacturer. Each device certificate include a unique Global Device ID (GDI), which is equal to the manufacturer prefix concatenated with the manufacturer serial number. Creating an AD involves a compliant device with a domain manager functionality creating a master device key list (MKL), containing a list of newly generated keys of length equal to the maximum number of devices allowed to join an AD. This limit is set by consumer electronic manufacturers and is difficult to change, as all devices must be equipped in advance with the keys for all devices expected to join an AD. The larger this limit, the more protected storage is required in

each device. Once the MKL has been created, the domain manager creates a domain identification (ID_{Domain}), composed of the domain manager GDI and the domain version number.

Adding a device to an AD involves two steps: first, compliance checking of the device, and second, authorising the device to communicate with other devices in the AD. The protocol used to add devices to an AD is described in [80]; it assumes that devices know the domain manager's GDI prior to joining an AD. The domain manager selects the next unused key in its MKL to be used as the device master key. The index of this master key in the MKL is referred to as the local device identification (LDI).

In the authorisation procedure, the domain manager issues to the joining device an authentication credential set consisting of a number of (authentication key, authentication ticket) pairs. The credential set is sent to the joining device together with the LDI and the master key. Authentication keys in the credential set are symmetric keys shared between pairs of devices in the AD. Each device is given authentication keys and corresponding authentication tickets for all devices already part of the domain, as well as for all potential devices that may join the AD in the future. The (authentication key, authentication ticket) pair, allowing device A to authenticate itself to device B, has the following form: $(K_{AB}, W_{K_B}(K_{AB}||ID_{Domain}||GDI_A||LDI_A||LDI_B))$, where K_{AB} is the authentication key for device A to communicate with device B, K_B is the master key for B, $W_{K_B}(Y)$ denotes the symmetric encryption of data Y using key K_B, and $X||Y$ is the result of the concatenation of data items X and Y in that order. The authentication key and corresponding ticket can be used by device A to prove to device B that it is a compliant device and that it is part of the same AD. Device B is assured that the domain manager has created the ticket,

because the ticket is encrypted with K_B, that is known only to device B and the domain manager.

Before exchanging content, the two devices must authenticate each other in order to prove that they are part of the same domain. The authentication protocol is described in [80]. Subsequently, both devices must generate a shared key that is used to establish a secure channel to transfer the content encryption key. The shared key is calculated as: $h(K_{AB}||K_{BA}||N_A||N_B)$, where h denotes a globally agreed cryptographic hash-function, and N_A and N_B are random nonces generated by devices A and B, respectively. The content itself is sent encrypted using the content encryption key chosen by the content distributor.

Revocation information is associated with each item of content, in the form of a list of GDIs called the Global Device Revocation List (GDRL). The content manager downloads a copy of the GDRL with every piece of content from the content provider. Subsequently, the content manager attempts to connect to the domain manager to convert the GDRL into a local list contains LDIs for the revoked devices in this AD. If the domain manager is reachable, the content manager forwards it the GDRL; the domain manager processes the GDRL and returns a Local Revocation List (LRL). An LRL is associated with each item of content in the AD (this is referred to as lightweight content). The GDRL contains a list of all revoked devices worldwide, and is expected to become relatively large. One estimate for the potential size of the GDRL is around 40M bytes [80]. If the domain manager is not reachable, the content manager keeps the original GDRL attached to each item of content (this is referred to as heavyweight content).

5.1.6.2 Analysis

We now analyse the DRM Security Architecture for Home Network scheme using the list of requirements given in section 4.5.

1. *Minimum cost.* The scheme requires devices to possess a processor, memory, and protected storage to store secret domain information. This increases the total costs. In addition, before devices can join a domain they must know the domain manager's GDI. This means joining devices must possess an I/O component to insert the manager GDI value.

2. *Ease of use.* The scheme is not flexible in that the AD size is fixed. The maximum number of devices that can join an AD depends on hardware factors. These factors include the storage available for storing secure domain information, i.e. the credential set and the device master key on the least capable device that is expected to join the AD. This is potentially inconvenient for users, especially corporate users.

3. *Performance.* The global revocation list, GDRL, is associated with every downloaded item of content, which has a serious impact on the overall system performance. The size of the GDRL increases as more devices are revoked. This causes the time to download content to increase. In addition, the local revocation list, LRL, is very difficult to maintain, because it is associated with every item of content inside the AD rather than being stored in a central location.

4. *Content mobility.* Each item of content is encrypted with a unique key chosen by the content provider. The published description [80] does not specify how the confidentiality of this key is assured. This key is trans-

ferred via a secure channel from the sender to the receiver, also used to exchange content between domain devices. This ensures content mobility.

5. *Lack of dependence on network infrastructure.* A key list is generated in advance for all devices expected to join a domain, as described above. This does not require all devices to be connected to the network simultaneously.

6. *Lack of dependence on a secure clock.* Nonces are used to provide message timeliness.

7. *Cryptographic robustness.* The proposed scheme uses 128-bit AES keys.

8. *Ease of recovery.* If the system is hacked, the AD must be re-initialised and all domain content re-encrypted. In addition, a backup and recovery policy has not been proposed for domain keys and content. If an insecure backup method is implemented, content and keys could be restored to a hacker device. This could then mean that domain keys and credential sets are revealed to the hacker. Alternatively, if backup is not allowed, then end users are likely to reject the system.

9. *Robust content protection.* This scheme fails to bind devices in an AD to the AD owner. Adding a device to the AD depends solely on the device being compliant, and the number of devices in the AD not exceeding the AD-specific limit. Moreover, the scheme does not specify where each content encryption key is stored.

10. *Flexible rights structure.* This requirement is not addressed in this scheme.

11. *Simple key management.* This scheme uses secret keys shared between each pair of devices in an AD. These keys are created and managed by the AD manager, and are given to devices when they join the AD.

12. *Flexible Revocation Mechanism.* This requirement is discussed in point (3).

13. *Access to and Usage of Content.* This requirement is not addressed.

14. *Consumer Privacy.* This requirement is not discussed in [80].

15. *Interoperability.* Devices must have a processor and protected storage in order to join an AD.

16. *Security and Hardware Issues.* The scheme does not interfere with device security settings.

5.2 Content Piracy Problem Definition

In this section we attempt to identify the main issues that underlie content piracy. Most current DRM systems recognise that consumers may have more than one device, which they would like to use to access their content without requiring multiple licences. As a result, many DRM system providers have incorporated the concept of an *authorised domain* into their content protection solutions. Such a domain is a collection of devices belonging to a single owner, within which digital assets can be freely moved.

Devices in a domain can be divided into two categories, as illustrated in Figure 5.1, namely *roots* and *leaves*. The domain root (unique per domain) represents a licensed content holder. The leaves in a domain receive content (and means to access the content) from the domain root.

The content piracy problem can then be divided into two sub-problems. The first is **Root Distribution**, where the root of the domain illegally distributes

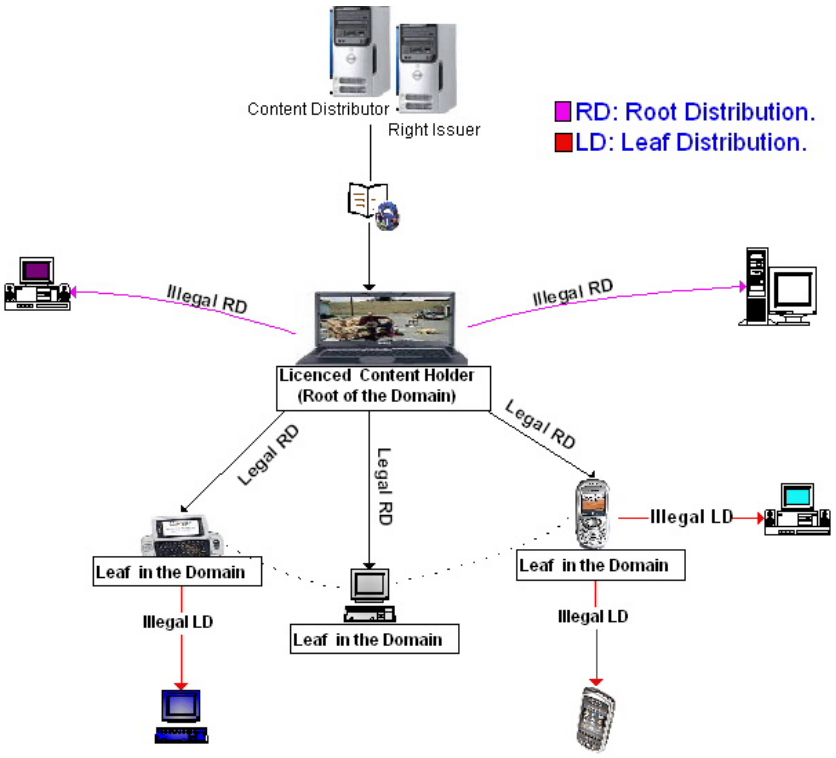


Figure 5.1: Illicit Content Proliferation

content and any associated passwords/keys to an unlimited number of users. For example, the content holder could illegally distribute content and an associated password to devices outside the domain, i.e. devices which are not leaves in this domain. The second is **Leaf Distribution**, where an individual leaf in a domain illegally redistributes content to devices outside the domain, as if it is the licensed content holder. For example, after receiving content and any associated passwords/keys, a leaf device could illegally re-distribute the content and passwords/keys to user_1, which in turn re-distributes them to user_2, and so on (where user_1 and user_2 are not leaves in this domain).

A fundamental authorised domain requirement for DRM is to restrict Root Distribution to legitimate devices owned by the domain owner, and to completely

prevent Leaf Distribution. In addition, an authorised domain system for DRM should satisfy other requirements, as discussed in chapter 4, such as: Ease of Use, Content Mobility, Performance, Ease of Recovery, Privacy, Interoperability and Robust Content Protection.

Other authorised domain management solutions for DRM, as discussed in this chapter, typically attempt to address these problems by using a counter to control the number of devices that can simultaneously access domain content. However such counter-based mechanisms have significant security and usability limitations. For example, in many schemes, devices can abuse the system by joining and then leaving multiple domains to illegally use their content. Moreover, in many schemes, increasing the domain size limit requires re-initialising and reconfiguring the domain, and, in some schemes, domains cannot be expanded.

In addition, there is no binding between the domain key (content protection key) and the domain owner. As a result, a leaf in a domain can redistribute content associated with a password/PIN; i.e. the Leaf Distribution problem arises. Also, all these schemes have additional problems in addressing other fundamental DRM requirements, such as content backup and recovery, ease of use, performance, etc. These issues are summarised in Figure 5.2.

5.3 Summary

This chapter contains an analysis and assessment of six of the most widely discussed DRM schemes that incorporate the authorised domain concept. The analyses and assessments were based on the 16 DRM requirements defined in

DRM Requirements	OMA DRM	xCP	Smart Right	Apple Fairplay	DRM for 3 G	DRM for Home Network
Minimum cost						
Ease of use						
Performance						
Content mobility.						
Lack of dependence on network infrastructure.						
Lack of dependence on a secure clock.						
Cryptographic robustness						
Ease of recovery						
Robust content protection						
Flexible rights structure						
Simple key management						
Flexible Revocation Mechanism						
Access to and Usage of Content						
Consumer Privacy						
Interoperability						
Security and Hardware Issues						

Addressed Not Addressed It is not clear if this requirement is addressed

Figure 5.2: Authorised Domain Content Protection Systems

chapter 4. The main conclusion of these analyses is that current schemes for protecting digital assets in personal networks do not satisfy many user, content owner, and/or copyright law acceptance criteria for digital content protection and consumption, as shown in Figure 5.2. For example, OMA-DRM satisfies the largest number of requirements, but even this scheme only meets ten of the 16 requirements.

Studying the shortcomings in the existing schemes enables us to isolate the issues underlying content piracy, namely Root Distribution and Leaf Distribution. This in turns helps us to devise new schemes which meet the requirements given in chapter 4.

Chapter 6

Authorised Domain Management Framework

Contents

This chapter presents a high level architecture for an authorised domain DRM system. After introducing the notion of an authorised domain, we provide a set of requirements for authorised domain devices. We then describe how devices conforming to the Trusted Computing Group specifications satisfy these requirements. Next, we specify a general workflow designed to address the two

elements underlying content piracy, i.e. Root Distribution and Leaf Distribution, as discussed in section 5.2. We then discuss and analyse the provided general framework. Subsequent chapters describe specific implementations of the general framework outlined in this chapter. Most of the material in this chapter has previously been published in [8, 9, 10, 11].

6.1 Authorised Domain Concept

Most current DRM systems recognise that consumers may have more than one device, which they would like to use to access their content without requiring multiple licences. As a result, many DRM system providers (as described in chapter 5) have incorporated the concept of an *authorised domain* into their content protection solutions; see, for example, [13, 44, 74, 89]. Such a domain is a collection of devices belonging to a single owner, within which digital assets can be freely moved.

Chapter 5 provided an analysis of the sources of threats to content within an authorised domain, and divides the devices in a domain into two categories: *roots* and *leaves*. The domain root (unique per domain) represents a licensed content holder. The leaves in a domain receive content (and means to access the content) from the domain root. Content piracy can be divided into two sub-problems, *Root Distribution* and *Leaf Distribution*. A fundamental DRM requirement is to restrict Root Distribution to legitimate devices owned by the domain owner, and to completely prevent Leaf Distribution. In addition, a DRM system should satisfy other requirements, as discussed in chapter 4, such as: Flexible Rights Structure, Ease of Use, Content Mobility, Performance, Ease of Recovery, and Interoperability.

6.2 Domain Device Requirements

Software-only techniques cannot provide a high degree of protection for domain credentials; for example, Apple Fairplay, which uses software-only techniques, has been hacked many times, as discussed in section 5.1.4. Also, all schemes analysed in chapter 5, apart from Apple Fairplay, require devices to incorporate trusted hardware components. These components are used to securely store domain credentials and/or to enforce content usage rules. For example, the SmartRight scheme requires the presence of a smart card reader on every device, and the OMA-DRM scheme requires each device to possess secure storage for domain credentials, and to incorporate a DRM agent trusted to enforce content usage rules. As a result, many currently used devices do not satisfy all the security requirements for existing domain-based DRM schemes.

Nevertheless, a DRM system must be designed in such a way that it imposes the minimum cost on the device manufacturers and (hence) the end users. Thus, the necessary hardware components should be simple and convenient to integrate into consumer devices without resulting in increased consumer device size, and they should not introduce new vulnerabilities into end user computing equipment. These considerations motivate our requirement here that domain devices must be trusted platforms (TPs), since TPs of this type appear likely to become very common. TPs are assumed to possess the following properties.

1. Each TP must have a tamper-resistant module that is bound physically and cryptographically to the TP.

2. The TP protects all the secret keys required by a domain device, typically by storing them inside the tamper-resistant module. Secrets can be sent

to a platform after the platform's software state has been measured and reported. Stored secrets are only released after the platform's software state has been measured and checked. Therefore, if a process relies on the use of secrets, it cannot operate unless it and its software environment are correct.

3. The TP must provide a protected execution environment, in which applications run in isolation, free from being observed or compromised by other processes running in the same protected partition, or by software running in any insecure partition [34].

In theory a TCG[1] compliant platform possesses all of these properties, as discussed in section 6.3. TCG compliant platforms are not expensive, and are currently available from a range of PC manufacturers, including Dell, Fujitsu, HP, Intel and Toshiba [35]. In addition, since early 2006, all Intel-based Apple computers are TCG compliant [96]. However, in practice, as discussed in section 3.8, many challenges remain to be solved in the TCG specifications, and these issues are the subject of ongoing research. Also, it will take some time until TCG functionality is integrated into all types of device, e.g. car CD players, TVs, etc. We do not expect the above properties to be available in all types of devices in the near future. This is because current operating system designs are insecure, and they possess many vulnerabilities; without a secure operating system, trusted applications can be subject to various attacks [84].

Nevertheless, this is not to say that the proposed DRM schemes are not practical for today's devices and operating systems. The primary purpose of requiring devices to possess the above TP properties was to protect the means for accessing

[1] www.trustedcomputinggroup.org

content, which is not main thrust of the work described in this thesis. The main novel contribution of this thesis is the specification of the four different schemes for strongly authenticating a domain owner. This authentication is used for binding an authorised domain to a single domain owner. These schemes can utilise existing techniques for protecting the domain credentials, including the means for accessing content. Hence, by relaxing the strong TP security assumptions described above, the proposed schemes can be implemented using today's devices. In this case, the difficulty of breaking one of the DRM schemes would be equivalent to the difficulty of hacking other schemes using current protection techniques (essentially this is a trade-off between security and cost). Also, we have designed the DRM schemes so that if a device is hacked, it might reveal the content stored in that device but it does not cause a global problem. Most importantly, hacked devices can be revoked, and hence cannot receive new content.

There are always trade-offs between security and cost, security and usability, and security and performance. Hence, the above TP security requirements can be relaxed or strengthened depending on the intended application that will use the proposed authorised domain management schemes. For example, protecting sensitive content in an enterprise network typically requires higher security assurance than protecting sensitive content in a personal network, as revealing an enterprise's sensitive content is likely to have a greater impact and affect more people than compromising content belonging to a personal network.

6.3 Using TCG-conformant Devices

In this section we summarise how a system conforming to the TCG specifications can satisfy the requirements described in section 6.2; these issues are discussed in greater detail in chapter 3. In this section we do not discuss problems associated with the TCG specifications in practical life, as these have been discussed in section 3.8.

1. The TP has a hardware component referred to as Trusted Platform Module (TPM), that is physically and cryptographically bound to the TP. It is a self-contained processing module with specialist security capabilities such as random number generation, asymmetric key generation, digital signing, encryption capabilities, hashing capabilities, an HMAC engine, monotonic counters, as well as memory, non-volatile memory, power detection and I/O. Support for platform integrity measurement, recording and reporting is also provided. The TPM is typically implemented as a processing engine that is separate from the TP's main processing environment.

2. An asymmetric encryption Storage Root Key (SRK) pair is securely associated with each TPM. The SRK private key is statistically unique, and is created and stored inside the TPM. The public part of the SRK acts as the root for encrypting sub-tree key objects that are used as data or signing key objects. The private part of the SRK is used for decrypting sub-tree objects.

3. A certification authority, the privacy CA, certifies public identity keys belonging to TPMs. The generated certificate binds an identity of the

TPM together with a description of the platform's security properties to a public key used for the verification of digital signatures. A TPM requests such a certificate from a privacy CA by providing its platform credentials. These credentials are generated by the Trusted Platform Module Entity (TPME), that vouches that a TPM is a genuine TPM, the Platform Entity (PE), that attests to the correct incorporation of a particular TPM into a platform, and the Conformance Entity (CE), that attests that the design of the TPM in that class of platform meets the TCG specifications, and that the way that the platform incorporates that type of TPM also meets the TCG specifications.

4. A TPM can generate two types of keys, known as migratable and non-migratable keys. Migratable keys can be transmitted to other TPs if authorised by both a selected trusted authority and the TPM owner. A non-migratable key is bound to the TP that created it. Data encrypted under non-migratable keys can leave the TP if and only if the software agent authorises the release of the data to other platforms. We assume that software agents that are authorised to read data encrypted using non-migratable keys will not release the data outside the TP boundaries.

5. A TP can perform integrity challenge and response exchanges with other TPs. One TP can verify the trustworthiness of the state of another TP by computing an expected set of trustworthy integrity metrics and comparing them with the current platform software state obtained from the integrity response. This enables an entity to verify that the DRM agent is running correctly on a remote TP.

6. A TP can protect secret keys (e.g. as required to support DRM) by encrypting them using a non-migratable key, as described in point (4). The

TP associates the current platform software state, as stored in the Platform Configuration Registers (PCRs), with the non-migratable key, and then protects them using the SRK. Stored secrets are only released after the current values in the platform's PCRs have been compared with the values associated with the stored key. Reporting, storage, and retrieval are carried out by the TPM. Therefore, if a process relies on the use of secrets, it cannot operate unless it and its software environment are correct. The system assumes that if the operating system and application are as expected, then the integrity and secrecy of data is subsequently guaranteed.

6.4 Role Model

The role model used in the framework involves the following main entities.

- The **domain owner**. It is assumed that a domain will always be associated with a single individual, referred to as the domain owner.
- A scheme-specific **certification authority (CA)**. This CA could be distributed across multiple locations and needs to be trusted by domain devices, content distributors and rights issuers. TPM manufacturers could play the role of the CA. The CA is required to maintain and disseminate a revocation list.
- A **domain device**, which is assumed to be a TP, possesses the properties described in section 6.2. In addition, a domain device is assumed to have the following properties.

    1. Each domain device is assumed to be pre-equipped with a DRM-

scheme-specific signature key pair, the public key of which is certified by the scheme-specific CA. The certificate contains a general description of the tamper-resistant security module in the domain device and its security requirements. The private part of the key is stored within the module, and is used for entity authentication. Domain devices are required to possess the public key of this CA.

2. Each domain device is assumed to possess a DRM agent, which must be trusted to perform the DRM scheme correctly. In addition, each domain device can verify that the DRM agent is running correctly in another device, using TP functionality.

3. Each domain device is assumed to possess an asymmetric encryption key pair certified using the signature key described in point (1). The corresponding private decryption key is bound to a particular environment configuration state. We assume that the DRM software agents that are authorised to read data encrypted using this key will not release the data outside the domain device; even the domain owner should not be able to retrieve these data in clear.

- The **content distributor** distributes items of protected digital content to consumers. Each item of protected content is encrypted using a content-specific key (K_T). This key is stored in the rights object associated with this piece of content; this rights object also contains usage rules that apply to the content. Content distributor devices, which distribute content to consumers, must have the TP features described in section 6.2 and must also satisfy the domain device assumptions described above.

- The **rights issuer** is in charge of issuing the rights object associated with an item of content. The DRM agent in each domain device that renders protected digital content enforces the rules inside the associated

rights object. Rights issuer devices, which distribute rights objects to consumers, must have the TP features described in section 6.2 and must also satisfy the domain device assumptions described above.

- A **trusted authority** is responsible for managing the parameters controlling the size of a domain and the total number of times that a device can join a domain. Trusted authorities must be trusted by content distributors and rights issuers, and are required to possess the public key of the scheme-specific CA. A trusted authority might also be responsible for other functionality, such as domain key backup and recovery. The nature and functionality of a trusted authority is implementation-dependent, and specific examples of trusted authorities are provided in the implementations of the framework described in chapters 7–10. A trusted authority might, for example, be a trusted third party of some kind.

- A **trusted domain controller** is responsible for managing the membership of a domain; in particular it is responsible for authenticating a user prior to allowing a device to join a domain. Trusted domain controllers must be trusted by content distributors and rights issuers, and are required to possess the public key of the scheme-specific CA. The means by which this user authentication is achieved is implementation-dependent; the framework instantiations described in chapters 7–10 employ a variety of approaches.

  The nature of a trusted controller is also implementation-dependent; it could, for example, be a trusted third party or a trusted agent running on a user device. The software that implements the trusted domain controller functionality is referred to as the 'DRM agent'. Also, the trusted controller and trusted authority could, in some circumstances, be implemented by the same entity.

6.5 General Framework

In this section we describe a general framework for the creation and ongoing management of an authorised domain. This framework involves creating a domain owned by a single owner, where all devices joining the domain are bound in some way to that domain owner. Each domain has a unique domain-specific secret key K_D that is securely generated and is given to a device when it joins the domain (after the domain owner has been authenticated). This key, that is used to encrypt content encryption keys, is not available in the clear even to the domain owner, and cannot be copied between devices.

Each domain has two associated limits that are maintained and controlled by the trusted authority, one to control the number of devices that can simultaneously access domain content, and the other to control the total number of devices that can join a domain. The latter limit is designed to stop domain owners abusing the system by allowing multiple devices to join and then leave their domains.

In the remaining part of this section we present the workflow of the general framework, which is divided into five main phases. The first covers the process of domain creation. The second phase involves adding devices to a domain. The third phase covers the removal of a device from a domain. The fourth phase covers the exchange of content between domain devices and content distributors/rights issuers, between devices in the same domain, and between devices in different domains. The fifth and final phase involves domain backup and recovery.

6.5.1 Domain Establishment

This phase covers the creation of a domain, and involves the following steps.

1. The trusted domain controller authenticates the domain owner. This could be achieved in a variety of different ways. The four implementations of this framework specified in chapters 7–10 incorporate a variety of approaches to this step.

2. The trusted domain controller generates a domain-specific key K_D using a random number generator, securely generates two counters, c_T and c_C, initialised to zero, associates the two counters with the key K_D, and binds the result to the created domain. The counter c_T represents the total number of devices that have joined the domain, and c_C represents the number of devices currently present within the domain. The maximum permitted values of these counters are maintained and controlled by a trusted authority, as outlined above.

6.5.2 Adding a Device to a Domain

This phase covers the addition of a device to a domain. It involves the following steps:

1. The DRM agent running in the trusted domain controller authenticates the domain owner in some way, as described in section 6.5.1. If authentication fails, the agent exits with an appropriate error message.

2. If authentication succeeds, the joining device and the trusted domain con-

troller validate each other's trustworthiness by some means.

3. If the trustworthiness check succeeds, the trusted domain controller increments the values of both c_T and c_C. If the new value of c_T or c_C is greater than the maximum permitted value for this domain, then the agent exits with an appropriate error message. The maximum values of c_T and c_C can be increased by an explicit authorisation from the trusted authority. Domain owners could be charged more for higher maximum c_T and/or c_C values.

4. If the new values of c_T and c_C are within the permitted ranges for this domain, the trusted domain controller securely transfers K_D to the joining device. K_D is securely stored in the joining device, using the functionality described in section 6.2; as a result K_D cannot be copied between devices, and is only released from the trusted domain controller to a device after successfully performing steps 1–4 above. Hence content owners will have assurance that K_D will not be released to third parties.

6.5.3 Removing a Device from a Domain

The third phase covers the case where a domain owner wishes to remove a device from the domain. In order for a device to leave a domain, the domain owner follows similar steps to those used for devices joining a domain, with the exception that c_T does not change, and c_C is decremented. This process involves the following steps.

1. The trusted domain controller authenticates the domain owner in some way, as described in section 6.5.1. If authentication fails, the agent exits with an appropriate error message.

2. If authentication succeeds, the leaving device and the trusted domain controller validate each other's trustworthiness by some means. This step is to ensure that the leaving domain device can be trusted to delete its stored copy of K_D.

3. If the trustworthiness check succeeds, the DRM agent decrements the value of c_C.

If a domain device is hacked or fails in some way (e.g. because of a hardware failure), the domain owner must inform the scheme-specific CA, which can then add the domain device public key to the revocation list. The trusted domain controller then ensures that the hacked device public key has been revoked, and decrements the value of c_C. As described in the next section, hacked devices cannot receive new content, and the value of c_T is not decremented when a device is removed from a domain, even if the device has failed or been hacked. This ensures that the domain owner cannot abuse the system by adding devices and then claiming that they have failed or been attacked in some way.

6.5.4 Exchanging Content

In chapter 4 we described the workflow for a typical DRM system, involving digitising content, identifying and labeling content, associating content with usage rules stored inside a rights object, and protecting and downloading content. The rights object contains the rules governing the use and the distribution of associated content. These rules are expressed in a REL, as described in section 4.4. A rights object might contain two types of REL elements: permissions (e.g. display, print, play) and constraints on permissions (e.g. play 10 times, display for 1hr). A rights object that does not contain constraints on permissions we

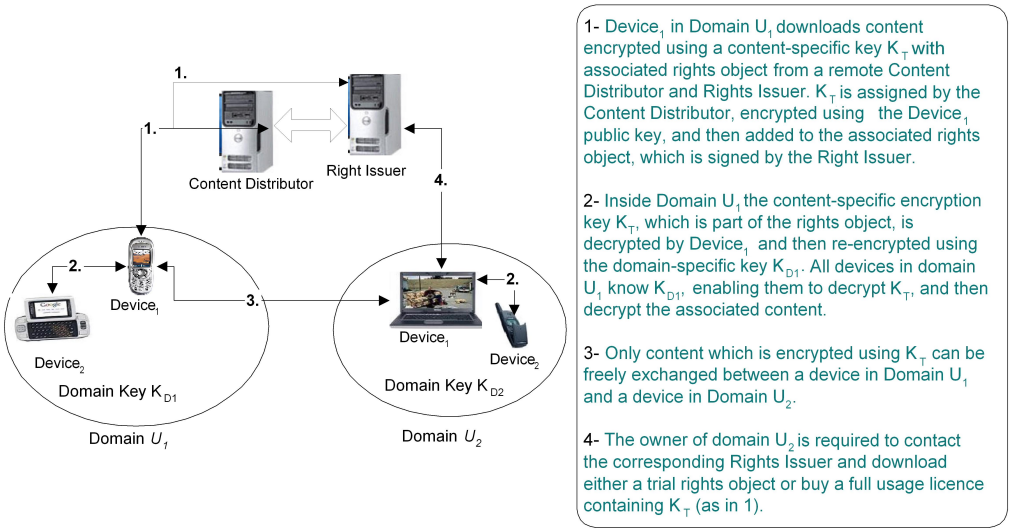


Figure 6.1: Exchanging Content Scenario

refer to as a static rights object, and a rights object that does have constraints on permissions we refer to as a dynamic rights object.

In this section we briefly outline how such a DRM workflow can be integrated into this general framework, in order to enable digital content and an associated rights object to be downloaded from a content distributor/rights issuer to any domain device. This workflow involves the following steps; we assume that a domain device and a content distributor/rights issuer have a trusted copy of each other's public signature verification and encryption keys (e.g. as can be achieved by exchanging certificates). One way in which the workflow could be implemented is described in section 7.3.4.

1. A domain device downloads content and an associated rights object from a remote content distributor (or rights issuer), where the content is encrypted using a content-specific secret key K_T generated by the content distributor. The rights object associated with the content holds a copy

of K_T, encrypted using the public key of the domain device. The rights object is also signed by the rights issuer.

2. Once content has been downloaded to a domain device, the content encryption key K_T is decrypted by the domain device using its private key, and then re-encrypted using the domain-specific key K_D. The rights object is then signed using the domain device signature key.

3. Encrypted content associated with a static rights object can be freely transferred between domain devices. However, this should not be the same case with a dynamic rights object. For example, if a rights object has a constraint 'play nine times', then the domain owner should be able to play the content nine times in total in the domain; i.e. the domain owner could play the content three times on device_1 and six times on device_2. The DRM agent running on a domain device can enforce such a mechanism. The main threat to a dynamic rights object would involve backing it up a rights object and then restoring it at a later time to reuse an expired licence. To our knowledge, there is no solution for this problem, for authorised domains. In this thesis we do not address this problem, which remains a possible topic for future research.

4. All devices in a domain possess a copy of the secret key K_D, and hence the protected content associated with the rights object can be freely transferred between devices in the same domain (after verifying that the destination device public key has not been revoked). The DRM agent on each domain device enforces the rules inside the associated rights object. Figure 6.1 shows a typical usage scenario for buying and exchanging content.

6.5.5 Backup and Recovery

A DRM system must be capable of recovering digital content in the event of system failure. For backup purposes, digital content encrypted using K_D can be stored in an offline medium, for example a USB memory stick or a or CD-ROM[2]. If the domain key K_D is lost and cannot be recovered, it follows that the domain content on the backup cannot be decrypted. Thus, backup provisions for K_D are needed.

Our backup strategy is based on the assumption that the trusted domain controller has a trusted backup and restore agent that can backup and recover K_D. The way in which this is implemented depends on the nature of the trusted domain controller. Chapters 7–10 describe four ways of backing up K_D, as part of the respective implementations of the framework.

6.6 Discussion and Analysis

In this section we consider how the framework controls content sharing, and how domain membership is managed.

6.6.1 Controlling Content Sharing

The main goal of the proposed framework is to prevent content decryption keys stored inside rights objects from being transferred unprotected to devices in different domains. This is achieved, as described in section 6.5.4, by encrypting

[2]In this thesis we do not consider problems associated with dynamic rights objects, which is explained in section 6.5.4

each content-specific key K_T with the domain-specific key K_D, where K_D is only available to domain devices. This domain key is transferred to a device only after the domain owner has been authenticated, the values of both domain counters have been incremented, and the trustworthiness of the device has been verified. Hence the only devices that can access content are those that are part of the domain, and devices can only be added to a domain if the domain owner is personally involved (as ensured by authentication of the user). The right to access content in a domain is thus bound to the domain owner.

As explained in section 6.5.3, K_D is removed from a device when it leaves a domain, which prevents protected rights objects from being used by devices in multiple domains. However, this does not stop legitimate controlled content sharing; protected content can move between devices belonging to different domains. A consumer could, for example, obtain protected content from anywhere; however, he/she can only use protected content by contacting the corresponding rights issuer and downloading a trial rights object, which will enable him/her to temporarily use the protected content. If the consumer is interested, he/she could then buy a full usage licence, as explained above. This concept is known as ***super-distribution***, and has been proposed by OMA [74] as a means of allowing consumers to obtain digitally protected content from anywhere, and to use a restricted licence. This allows consumers to use content for a limited period, with lower quality and/or limited features. When the consumer is happy with the protected content and decides to get a full licence, only then will he/she need to download the rights object, which is much smaller than the encrypted content.

6.6.2 Controlling Domain Membership

The framework allows both the Root Distribution and the Leaf Distribution problems to be addressed. The Root Distribution problem is solved by introducing the following two factors into domain management.

- The first is controlling both domain size and changes in domain membership by associating two limits with each domain, as discussed in section 6.5, in such a way that these limits can be changed where necessary. Changes to these limits are controlled by the trusted authority, and do not affect domain content in any way, such as requiring content or content encryption keys to be re-encrypted whenever a device joins or leaves a domain.

- The second is by ensuring that the domain-specific key K_D is generated automatically, is bound to the domain owner, and is unavailable in the clear, even to the domain owner. This prevents the domain owner from disseminating the key K_D. Consequently, distributing content to a device outside the domain will not enable access to the content unless this device joins the domain and thereby receives K_D. When a device leaves a domain it deletes its stored copy of K_D.

The problem of Leaf Distribution is solved by ensuring the uniqueness and confidentiality of the domain-specific key K_D. In addition, as stated above, K_D is bound to the domain owner in such a way that only the domain owner can authorise the transfer of the key K_D to other devices, and this only occurs when a device joins the domain. Although domain devices possess the key K_D, they are incapable of redistributing it to other devices; this addresses the Leaf

Distribution problem.

6.7 Summary

In this chapter we have proposed a general DRM framework, designed to support the authorised domain concept and which is also designed to address the two main security issues described in section 5.2. That is, the scheme is designed to restrict Root Distribution to devices owned by the domain owner and to prevent Leaf Distribution.

A successful DRM solution is required to satisfy other fundamental DRM requirements; see, for example, section 4.5. In chapters 7–10 we propose four different instantiations of this general framework which are designed to address these fundamental DRM requirements.

Chapter 7

Authorised Domain Management using a Master Control Device

Contents

In this chapter we describe an implementation of the generic authorised domain DRM framework given in chapter 6. This scheme involves managing an authorised domain using a master control device which supports: authentication of a

domain owner; domain establishment; adding devices to and removing devices from a domain; and backing up and recovering domain credentials. The system security requirements, threats, and services are analysed. How the proposed scheme counteracts the main content piracy threats is also discussed. Finally, the pros and cons of two possible approaches to user authentication, i.e. the use of a password/PIN and biometric authentication mechanisms, and possible countermeasures to the identified vulnerabilities in these two approaches are discussed. Most of the material in this chapter has previously been published in [10].

7.1 Introduction

The system described in this chapter is an implementation of the generic authorised domain DRM framework given in chapter 6. The proposed system works by binding all devices in a domain to a single master control device, which is itself bound to a single owner. The unique domain key K_D (introduced in section 6.5) is securely generated and stored inside the master control device. This device is implemented so that K_D is not available in the clear even to the domain owner. The domain key is transferred from the master control device to a device joining a domain after the master control device has authenticated the domain owner; this authentication uses either biometric or password/PIN authentication techniques.

This chapter is organised as follows. Section 7.2 describes the master control device functionality. Section 7.3 describes the proposed solution and the process workflow. Section 7.4 describes how the proposed scheme controls domain membership. Section 7.5 analyses the system security requirements, threats, and

services. Section 7.6 described how these security requirements can be met using TCG functionality. Section 7.7 discusses user authentication methods, and section 7.8 provides conclusions.

7.2 The Master Control Device

In addition to the entities identified in chapter 6, the system model includes a master control device that provides the trusted domain controller functionality. The master control device is responsible for: securely storing a domain owner authentication credential (a password/PIN or a biometric reference template) that is used to authenticate the domain owner whenever an attempt is made to add a device to the domain; securely generating and protecting a secret domain key (K_D), used to encrypt content encryption keys; and authorising devices to join the domain by checking that their processing environment is trusted. Only the master control device is permitted to transfer the domain key K_D to devices joining its domain, and this key cannot be copied between domain devices. The master control device is not required to be a dedicated device; it could, for example, be part of a domain device.

The master control device securely stores the authentication credential (the exact nature of which could vary — see section 7.7) alongside K_D and the two counters c_T and c_C, as defined in section 6.5.1. The master control device is required to have all the TP features described in section 6.2. The master control device is required to possess a software agent that implements the master control device functionality. The master control device is also required to possess a signature key pair, used in managing the domain. The public signature verification key is certified by the scheme-specific CA, as introduced in section

6.4. The resulting certificate is called a **domain certificate**. Such a certificate includes the following information: the fact that it is a domain certificate, a general description of the master control device and its security properties, the maximum number of devices that can be in the domain at any one time (i.e. the maximum permitted value of c_C), and the maximum number of devices that can join the domain (i.e. the maximum permitted value of c_T). The CA that signs the domain certificate provides the trusted authority functionality, as described in section 6.4. This CA could be distributed across multiple locations and needs to be trusted by content distributors and rights issuers. TPM manufacturers could play the role of the CA.

The master control device is assumed to incorporate a trusted backup agent that can backup and recover K_D, the authentication credential, the current values of both c_T and c_C, and the public key of every device that is a member of the domain. This information (i.e. K_D, the user authentication credential, the current values of the two counters, and the public keys of all the domain devices) is collectively referred to as the *domain credentials*.

7.3 Process Workflow

We now describe in detail how the five phases of the system workflow, as defined in section 6.5, are implemented.

7.3.1 Domain Establishment

The domain establishment phase, defined in section 6.5.1, is implemented as follows. The first time that a consumer uses the domain-specific master control

device, or on resetting the master control device, the following initialisation procedure must be performed.

1. The software agent in the master control device M executes, and instructs the domain owner to provide his/her authentication credential. This credential is stored by M.

2. The software agent securely generates an asymmetric encryption key pair $(P_M,\ R_M)$, where R_M is protected using the functionality described in section 6.2. Note that we use $(P_X,\ R_X)$ throughout the remainder of this thesis to denote the encryption/decryption key pair of entity X.

3. The software agent generates a secret key K_D. The software agent securely associates the counters c_T and c_C, initialised to zero, with the stored authentication credential and the key K_D.

7.3.2 Adding a Device to a Domain

This phase, defined in section 6.5.2, is implemented as follows. Adding a new device J to a user domain with master control device M involves the following steps, as summarised in Figure 7.1.

J first sends a Join_Domain request to M, of the form:

(1) $J \rightarrow M$: $P_J||S_J||N_1||\text{Cert}_{I_J}||\text{Sign}_J(P_J||S_J||N_1)$.

where S_J is the execution status of the DRM agent on J (the exact nature of S_J is implementation dependent, see, for example, section 7.6), N_1 is a fresh nonce generated by J, Cert_{I_J} is a certificate signed by the CA for the public key I_J (as

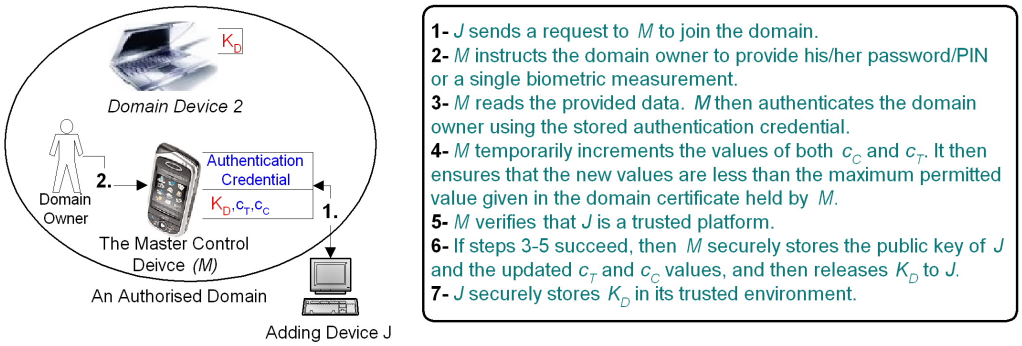


Figure 7.1: The Workflow for Adding a Device into a Domain

described in section 6.4, bullet 3), I_J is the signature verification key of J, and $\text{Sign}_J(Y)$ denotes a signature on data Y created using the private signing key of J (note that we use S_X, Cert_{I_X}, I_X and $\text{Sign}_X(Y)$ throughout the remainder of the thesis to denote the corresponding objects associated with entity X).

M verifies Cert_{I_J}, extracts I_J from Cert_{I_J} and checks that it has not been revoked, e.g. by querying an OCSP service. M then verifies J's signature using I_J. M verifies that the DRM agent is running correctly in J by checking the value of S_J. How this verification occurs is implementation-dependent; see, for example, section 7.6. M then authenticates the domain owner using the stored authentication credential. If authentication succeeds, M checks whether the public key of J is already a member of the domain (a device might need to rejoin a domain, for example, in the event of hardware/software failure, as discussed in section 7.3.5). If the public key of J is not in the domain, M temporarily increments the values of both c_T and c_C (it does not store the incremented values at this stage). If the new value of c_T or c_C is greater than the maximum permitted value given in the domain certificate held by M, then the agent running on M exits with an appropriate error message.

Next, M generates a nonce N_2, and then sends it, with other information, to J,

as follows:

(2) $M \rightarrow J$: $\text{Cert}_{I_M}||A_J||N_1||N_2||S_M||\text{Sign}_M(A_J||N_1||N_2||S_M)$

where A_J is an identifier for J, as contained in Cert_{I_J} (note that we use A_X to denote the identifier of X throughout), and Cert_{I_M} is the domain certificate (see section 7.2). J verifies Cert_{I_M}, extracts I_M from Cert_{I_M} and verifies that it has not been revoked, e.g. by querying an OCSP service, and then checks that this certificate was issued for a master control device, as discussed in section 7.2. J uses I_M to check M's signature, verifies A_J to ensure it is the intended recipient, and verifies message freshness by comparing N_1 with the value sent in (1). J then verifies that the software agent is running correctly in M by checking the value of S_M. As above, how this verification occurs is implementation-dependent.

Next, J generates a nonce N_3, and then sends a reply to M, as follows:

(3) $J \rightarrow M$: $A_M||N_2||N_3||\text{Sign}_J(A_M||N_2||N_3)$

M verifies J's signature, verifies A_M to ensure that it is the intended recipient, and verifies message freshness by comparing N_2 with the value sent in (2). Steps 1–3 conform to the three-pass mutual authentication protocol specified in [46].

Each domain must have a single domain device assigned for use for backup purposes, which is referred to as the backup device. There are two possible cases for the next part of this phase, depending on whether or not a backup device has already been assigned for the domain.

If J is the first device to join the domain, then it becomes the backup device. M first securely stores P_J, and updates the stored values of the counters c_T and

c_C. M then passes a backup request to J, as follows:

(4.a) $M \rightarrow J$: $e_{P_J}(K_D||A||c_T||c_C||P_J)||N_3||\text{Sign}_M(e_{P_J}(K_D||A||c_T||c_C||P_J)||N_3)$

where A is the authentication credential, $e_X(Y)$ denotes the asymmetric encryption of data Y using public key X, and where we assume that the encryption primitive in use provides non-malleability, as described in section 2.3.3.1. J verifies message freshness by comparing N_3 with the value sent in (3), verifies the signature of the received message and then decrypts it. The recovered secret key K_D is securely stored, as described in section 6.5.2. J also securely stores and protects the received domain backup information consisting of the received encrypted message and Cert_{I_M}.

If, on the other hand, the domain already has an assigned backup device B, i.e. J is not the first device to join the domain, then M first generates a random nonce N_4 and sends B the following request:

(4.b.1) $M \rightarrow B$: $N_4||A_B||\text{Sign}_M(N_4||A_B)$

B verifies the signature in the received message, and verifies that it is the intended recipient by checking the value of A_B. If the verifications succeed, B generates a random nonce N_5 and sends an acknowledgment to M, including the nonce N_4 received in (4.b.1).

(4.b.2) $B \rightarrow M$: $N_4||N_5||A_M||\text{Sign}_B(N_4||N_5||A_M)$

Next, M verifies the signature of the received message, and verifies it is the intended recipient by checking the value of A_M. If the verifications succeed,

M sends B the following backup request, including the nonce N_5 received in (4.b.2):

(4.b.3) $M \rightarrow B$: $e_{P_B}(K_D||A||c_T||c_C||D)||N_5||\text{Sign}_M(e_{P_B}(K_D||A||c_T||c_C||D)||N_5)$

where D is a string consisting of the public keys of all the devices that are members of the domain. B verifies the signature in the received message, and verifies message freshness by comparing N_5 with the value sent in (4.b.2). If the verifications succeed, then the received encrypted message replaces the existing domain backup information. B then sends the following to M, including the nonce N_4 received in message (4.b.1):

(4.b.4) $B \rightarrow M$: $\text{Result}||N_4||\text{Sign}_B(\text{Result}||N_4)$

M verifies the signature, and verifies that N_4 equals the value sent in (4.b.1). M then checks the value of the Result, that can be set to either success or fail. If it indicates success, M securely stores P_J and updates the stored values of the counters c_T and c_C. M then passes the domain key to J, as follows:

(4.b.5) $M \rightarrow J$: $e_{P_J}(K_D)||N_3||\text{Sign}_M(e_{P_J}(K_D)||N_3)$

J verifies message freshness by comparing N_3 with the value sent in (3), verifies the signature of the received message and then decrypts it. The recovered secret key K_D is then securely stored, as described in section 6.5.2.

7.3.3 Removing a Device from a Domain

This phase is exactly as described in section 6.5.3.

7.3.4 Exchanging Content

This phase, defined in section 6.5.4, involves transferring content from a content distributor to a domain, and exchanging content between domain devices. In this section we describe both these procedures.

7.3.4.1 Transferring Content from a Content Distributor

Transferring content from a content distributor to a device in a domain involves the following steps. An existing domain device V sends a Get_Content message to the content distributor to request content C, as identified by id. The request includes the payment details, e.g. credit card details, to be used to pay for access to C. The content distributor and V mutually authenticate each other, and the content distributor attests to the state of V, as described in section 7.3.2, steps 1–3. If these verifications succeed, the content distributor securely generates a content-specific secret key K_T, symmetrically encrypts C using the key K_T, and sends a Generate_RO request to the rights issuer, as follows:

(1) content distributor $\rightarrow$ rights issuer: $P_V||K_T||h(E_{K_T}(C))||id$

where we assume that a secure session has been pre-established between the content distributor and rights issuer, h denotes a globally agreed cryptographic hash-function, $E_X(Y)$ denotes the symmetric encryption of data Y using key X, and where we assume that E provides authenticated encryption, as defined in section 2.3.2.3. The rights issuer generates the rights object R, where:

$R =$ (C's consumption rules $||e_{P_V}(K_T)||h(E_{K_T}(C))||id$)

and where $h(E_{K_T}(C))$ is used to bind R to the content C. Subsequently, the rights issuer sends the following message to the content distributor:

(2) rights issuer $\rightarrow$ content distributor: $\text{Cert}_{I_{RI}}||R||\text{Sign}_{RI}(R)$

The content distributor then associates message (2) with $E_{K_T}(C)$ and forwards the result to V, as follows:

(3) content distributor $\rightarrow V$: $\text{Cert}_{I_{RI}}||R||\text{Sign}_{RI}(R)||E_{K_T}(C)$

V verifies that R is bound to C by recomputing $h(E_{K_T}(C))$, verifies $\text{Cert}_{I_{RI}}$, extracts I_{RI}, checks its validity, e.g. by querying an OCSP service, and then checks the rights issuer's signature. Finally, V decrypts K_T using its private decryption key, re-encrypts it using K_D, and retains $R||\text{Sign}_V(R)||E_{K_T}(C)||E_{K_D}(K_T)$.

7.3.4.2 Transferring Content between Domain Devices

Transferring content from a source device V_s to a destination device V_d in the same domain involves the following steps. In the description below we implicitly assume that V_d and V_s have a real-time communications link; if such connectivity is not available, then the same messages can be exchanged using a portable storage medium, e.g. a USB memory stick. In the protocol below, only V_s needs to query an OCSP service; this means that not all domain devices need to be connected to the Internet whilst exchanging content. For example, a domain will typically contain at least one device with Internet access, as used to download content from a content distributor. This Internet-connected device can also be used to access an OCSP service, and to distribute downloaded content to other

devices that do not have a real-time communications link.

V_d first sends the following Get_Content message to V_s:

(1) $V_d \rightarrow V_s$: $\text{Cert}_{I_{V_d}}||P_{V_d}||id||\ \text{Sign}_{V_d}(P_{V_d}||id)$

On receipt of (1), V_s verifies $\text{Cert}_{I_{V_d}}$, extracts I_{V_d}, checks its validity, e.g. by querying an OCSP service, and then checks V_d's signature. If the verifications succeed, V_s sends C, as identified by id, and the associated rights object, to V_d in the following message:

(2) $V_s \rightarrow V_d$: $E_{K_T}(C)||e_{P_{V_d}}(R||E_{K_D}(K_T))$

V_d now decrypts $e_{P_{V_d}}(R||E_{K_D}(K_T))$ using its private key, and verifies that C is bound to R, as described above. If the verification succeeds, V_d retains $E_{K_T}(C)||E_{K_D}(K_T)||R||\text{Sign}_{V_d}(R)$. When accessing the content, K_D is used to decrypt K_T, as needed to decrypt the content.

7.3.5 Backup and Recovery Procedure

This phase, as described in section 6.5.5, is implemented as follows. The backup strategy is based on the assumption that the master control device has a trusted backup agent, and that a selected domain device has a trusted restore agent that can backup and recover K_D, the user's authentication credential, the current values of both c_T and c_C, and the public signature verification keys of all devices in the domain. The master control device acts as a central point in backup and recovery; specifically, the master control device decides whether or not to accept a particular device for backup purposes. Moreover, restoration of the key K_D

can only be implemented via a certified master control device.

In this section we describe the procedure for recovering a copy of K_D to a failed device, the procedure for recovering a master control device, and the procedure for changing a domain backup device.

7.3.5.1 Device Recovery

The procedure for recovering a copy of K_D to a failed device is as follows. The domain owner must first add the failed device back into the domain in order to provide it with a copy of K_D, as described in section 7.3.2. However, as described in 7.3.2, before the master control device increments the values of c_T and c_C, it checks whether the joining device public key is already a member of the domain; if so, it does not increment the values of c_T and c_C.

7.3.5.2 Recovery of a Master Control Device

The procedure for recovering an existing master control device M_1 to a new master control device M_2 is as follows.

1. Suppose M_1 has a hardware failure and cannot be repaired. The backup software running on the domain backup device first checks that the public key certificate for M_1 has been revoked, e.g. by querying an OCSP service. If this certificate has not been revoked, then the recovery process terminates with an appropriate error message.

2. The backup software then checks that the public key certificate for M_2 has not been revoked, e.g. by querying an OCSP service, and checks whether

M_2 is a device capable of acting as a master control device. As described in section 7.2, we assume that M_2's domain certificate contains a field to show that it corresponds to a master control device.

3. If M_2 is trusted and possesses a master control device certificate, then the backup software releases to M_2 the string $(K_D||A||c_T||c_C||D)$, encrypted with M_2's public key.

4. M_2 securely stores the received information. The backup device stores the new domain certificate (i.e. M_2's certificate, as described in section 7.2) in its trusted storage.

7.3.5.3 Changing Backup Device

As discussed above, only a single domain device is used for backup. As described in section 7.3.2, the first device that joins a domain is defined to be the default backup device (B_1). However, the domain owner could assign a different device (B_2) to act as the backup device, although it must be a member of the domain. A domain device is always in one of the following four states:

1. 'active-backup' is the state indicating a current domain backup device; a device in this state holds a valid copy of the domain backup information.

2. 'removed-pending' is the state to which a domain backup device is switched when it is in the process of being replaced with a new domain backup device. Whilst a device has this state, it keeps the existing domain backup information. This copy of the information can be used if there is a hardware failure before the state of the new domain backup device changes to 'active-backup'.

3. 'removed-permanent' is the state of a device that does not store any backup information. Normally, all but one of the devices in a domain will be in this state.

4. 'pending-backup' is the state to which a newly nominated domain backup device is transferred while it is in the process of replacing an existing domain backup device. A device in this state will have a valid copy of the domain credentials; however, it will not respond to a request to update its copy of the backup credentials, i.e. it will ignore messages of type (4.b.1) specified in section 7.3.2.

The process for changing the nominated backup device from B_1 to B_2 is split into three phases. The first involves updating the state of B_1 from 'active-backup' to 'removed-pending'. The second phase involves updating the state of B_2 from 'removed-permanent' to 'pending-backup'. The last phase involves updating the state of B_1 to 'removed-permanent', and updating the state of B_2 to 'active-backup'. These phases are implemented as follows.

1. The master control device M generates a random nonce N_1 and sends B_1 the following request:
   (a) $M \rightarrow B_1$: $N_1||A_{B_1}||\text{Sign}_M(N_1||A_{B_1})$

2. B_1 verifies the signature of the received message, and verifies that it is the intended recipient by checking the value of A_{B_1}. If the verifications succeed, B_1 generates a random nonce N_2 and sends an acknowledgment to M, including the nonce N_1 received in (a).
   (b) $B_1 \rightarrow M$: $N_1||N_2||A_M||\text{Sign}_{B_1}(N_1||N_2||A_M)$

3. M verifies the signature of the received message, and verifies it is the intended recipient by checking the value of A_M. If the verifications succeed,

M sends a Remove_Backup_Pending request to B_1, including the nonce N_2 received in (b), as follows:

(c) $M \rightarrow B_1$: $P_{B_2}||N_2||\text{Sign}_M(P_{B_2}||N_2)$

4. On receipt of message (c), B_1 verifies the signature, and verifies message freshness by comparing N_2 with the value sent in (b). If the verifications succeed, B_1 updates its state to 'removed-pending'; however, it does not delete its stored backup information at this stage. B_1 then sends the following message to M, including the nonce N_1 received in message (a):

   (d) $B_1 \rightarrow M$: $N_1||\text{Sign}_{B_1}(N_1)$

5. When M receives message (d), it verifies the message signature, and verifies that N_1 equals the value sent in (a). If the verifications succeed, M updates its stored domain information so that B_1 is no longer registered as the domain backup device. M then generates a random nonce N_3 and sends B_2 the following request:

   (e) $M \rightarrow B_2$: $N_3||A_{B_2}||\text{Sign}_M(N_3||A_{B_2})$

6. B_2 verifies the signature of the received message, and verifies that it is the intended recipient by checking the value of A_{B_2}. If the verifications succeed, B_2 generates a random nonce N_4 and sends an acknowledgment to M, including the nonce N_3 received in (e).

   (f) $B_2 \rightarrow M$: $N_3||N_4||A_M||\text{Sign}_{B_2}(N_3||N_4||A_M)$

7. When M receives message (f), it verifies the signature, and verifies that N_3 equals the value sent in (e). If the verifications succeed, M sends a Backup_Domain request to B_2, as follows:

   (g) $M \rightarrow B_2$: $e_{P_{B_2}}(K_D||A||c_T||c_C||D)||N_4||\text{Sign}_M(e_{P_{B_2}}(K_D||A||c_T||c_C||D)||N_4)$

8. When B_2 receives message (g), it verifies the signature, and verifies message freshness by comparing N_4 with the value sent in (f). If the verifications succeed, B_2 sends a Backup_Domain_Pending instruction to M, including the nonce N_3 received in message (e), as follows:

   (h) $B_2 \rightarrow M$: $N_3 || \text{Sign}_{B_2}(N_3)$

9. When M receives message (h), it verifies the signature, and verifies that N_3 equals the value sent in (e). If the verifications succeed, M replaces the stored domain backup device public key with P_{B_2}, and then sends a commit message to both B_1 and B_2, as follows.

   (i) $M \rightarrow B_1$: $N_2 || \text{Sign}_M(N_2)$

   (j) $M \rightarrow B_2$: $N_4 || \text{Sign}_M(N_4)$

10. On receipt of message (i), B_1 verifies the signature, and checks message freshness by comparing N_2 with the value sent in (b). If the verifications succeed, B_1 permanently deletes the stored backup information and updates its state to 'removed-permanent'. On recipient of message (j), B_2 verifies the signature, and checks freshness by comparing N_4 with the value sent in (f). If the verifications succeed, B_2 updates its state to 'active-backup'.

The integrity of the above protocol is ensured as follows. Only a single backup device is active at any time. In addition, there is always at least one valid copy of the domain backup information at all times. This protects domain credentials against a hardware failure of M whilst performing any of the above steps. For example, if M failed between steps 2–6, the status of the old backup device can be reset to 'active-backup'.

7.4 Controlling Domain Membership

Controlling domain membership is described in section 6.6.2. The scheme described in this chapter provides additional protection to help prevent illicit content proliferation. The only way in which a domain owner can transfer the content protection key to another user's device is by adding the device to the domain. Whilst possible in principle, such a procedure is unlikely to be attractive to the domain owner, as it means that the other user's device would become part of the domain controlled by the master control device, which would mean that fewer of the domain owner's own devices could be added to the domain. Most importantly, devices which have joined the domain are not able to re-transfer the domain key, as only the master control device can transfer the domain key, and this will only take place after it has authenticated the domain owner. Section 7.7 describes possible options for the means to be used to authenticate the domain owner, some of which could further increase the restrictions on content piracy.

7.5 Security Analysis

We now considers the security threats, services, and mechanisms that apply to the storage, execution, transmission, backup and recovery of the digital content and the domain credentials. The domain credentials, as defined above, consist of the public keys of all the devices that are members of the domain, K_D, the authentication credential, and the counters c_T and c_C.

7.5.1 Security Threats

The main goal of the proposed scheme is to prevent the transfer of the means for accessing content (i.e. the domain-specific key K_D) to unauthorised devices using physical or digital means. This is achieved by ensuring that this key is transferred to a device only after the domain owner has been authenticated, the values of both domain counters have been incremented, and the trustworthiness of the device has been verified. Hence, only devices which are part of a domain can access the domain content. Devices can only be added to a domain if the domain owner is personally involved, as ensured by authenticating the domain owner. This system has the following requirements.

- The master control device must securely generate, process and store the domain-specific credential, so that it is not available in the clear even to domain owner. This process is vulnerable to the following security threats: (1) unauthorised manipulation of the domain credentials during use in the master control device; and (2) unauthorised manipulation of the domain credentials whilst stored in the master control device.

- The master control device must securely send the key K_D to a device J joining the domain. Such a procedure is exposed to the following security threats: (3) unauthorised reading or alteration of K_D whilst in transit; (4) the master control device unwittingly sending K_D to a malicious entity; (5) J unwittingly receiving K_D from a malicious master control device; and (6) replay of communications between the master control device and J.

The above requirements ensure that the domain key (i.e. the means for accessing content) is securely generated, protected, and distributed to domain devices. The next step is to protect a piece of content C and the associated rights object R when it is stored, executed and transferred between domain devices. Security threats to the transfer of C and R between domain devices are:

> (7) unauthorised reading or alteration of C, and unauthorised alteration of R, while in transit; and (8) transfer of C and R to an unauthorised entity.

Security threats to the storage and execution of C and R in domain devices are:

> (9) unauthorised reading or updating of C, and unauthorised updating of R, whilst stored in a domain device; and (10) unauthorised reading or updating of C, and unauthorised updating of R while being accessed on a domain device.

Security threats to the backup and recovery of the domain credentials, including K_D, c_T, c_C, A and D, are:

> (11) unauthorised reading or alteration of domain credentials whilst in transit; (12) unauthorised reading or alteration of domain credentials whilst stored in the backup device; (13) the master control device unwittingly sending domain credentials to a malicious entity; (14) the backup device unwittingly receiving domain credentials from a malicious master control device; and (15) replay of communications between the master control device and the backup device.

In addition to the above threats, there is a special type of threat associated with dynamic rights objects, i.e. backing up a dynamic rights object and then restoring it at a later time to reuse an expired licence. As we discussed in section 6.5.4, there is no solution to this threat in authorised domains. In this thesis we do not address this problem, and it remains a possible topic for future research.

7.5.2 Security Services and Mechanisms

The security services required to counteract threats 1, 2, 4, 9, 10, 12, and 13 (listed above) can be provided using trusted platform functionality, as discussed in section 6.2. In section 7.6 we describe how such trusted platform functionality can be implemented using a platform conforming to the TCG specifications. Threats 3, 5–8, 11, 14 and 15 are addressed using standard cryptographic mechanisms. A direct mapping exists between the threats outlined above and the list of services and mechanisms given below.

1. *Confidentiality and integrity of the domain credentials during execution on a master control device.* Providing this service requires process isolation techniques, which provide protection for a message against being read or altered by an unauthorised entity whilst being executed [34]. This service can be provided using trusted computing technology, discussed in chapter 3.

2. *Confidentiality and integrity of domain credentials whilst stored in a master control device.* Providing this service requires protected storage, typically by storing domain credentials inside a tamper-resistant module, as discussed in section 6.2.

3. *Confidentiality and integrity of K_D whilst in transit.* This service is provided by the use of asymmetric encryption and a digital signature — see section 7.3.2.

4. *Entity authentication of a joining device J to a master control device.* The provision of this service is implementation-dependent, and involves a protocol exchange between J and the master control device; see, for example, section 7.6. It is initiated when the master control device and J mutually authenticate each other — see section 7.3.2. This mutual authentication attests to the DRM agent execution status, i.e. S_J, and whether the platform is trusted, as discussed in section 6.2.

5. *K_D origin authentication.* The joining device J checks the origin of K_D by checking the master control device's signature on the received encrypted value of K_D — see section 7.3.2.

6. *Prevention of replay of communications between a master control device and a device.* This is provided by the inclusion of nonces in protocol messages — see section 7.3.2.

7. *Confidentiality and integrity protection of content C, and and integrity protection of rights object R whilst in transit.* This service is provided by encrypting C and K_T using an authenticated encryption technique, and signing R; see section 7.3.4.

8. *Entity authentication of the destination device V_d whilst transferring content C and rights object R.* The source device V_s validates V_d's public key to make sure it has not been revoked, and then encrypts R and the encrypted K_T using V_d's public key — see section 7.3.4. V_s does not need to verify whether V_d is trusted, because the transferred K_T is protected

using K_D, which is known only by devices in the same domain and is revealed only in a trusted environment, as discussed in section 6.2.

9. *Confidentiality and integrity of content C, and integrity of rights object R in domain devices.* The integrity of R is protected using a digital signature. C is encrypted using the secret key K_T that is itself encrypted using a secret key K_D. As described in section 7.3.4, the symmetric encryption technique in use is assumed to provide authenticated encryption — see section 7.3.4. Also, the encryption key K_D is bound to the device's trusted environment, as discussed in section 6.2.

10. *Confidentiality and integrity of content, and integrity of rights object during execution* is provided exactly as discussed in (1) above.

11. *Confidentiality and integrity of domain credentials whilst in transit* is provided exactly as discussed in (3) above.

12. *Confidentiality and integrity of domain credentials whilst stored in a backup device* is provided exactly as discussed in (2) above.

13. *Entity authentication of a backup device to a master control device* is provided exactly as discussed in (4) above.

14. *Domain credentials origin authentication* is provided exactly as discussed in (5) above.

15. *Prevention of replay of communications between a master control device and a backup device* is provided exactly as discussed in (6) above.

7.6 Implementing the Protocols Using Trusted Computing

We described in section 6.3 how a system conforming to the TCG specifications [92, 93, 94] can satisfy the requirements described in section 6.2. In this section we consider how security requirements 1, 2, 4, 9, 10, 12, and 13, discussed in section 7.5, can be met using functionality in the TCG specifications. In this section we do not discuss practical problems associated with TCG specification, as these have been discussed in section 3.8.

1. *Confidentiality and integrity of domain credentials during execution.* A challenger can verify that a platform is trustworthy by validating the platform integrity metrics. The TP measures the integrity of software executed from platform start-up and stores the result in the platform's PCRs; this provides assurance to the challenger that the expected version of the OS, and of any other measured software, is running on the platform. The private key that is needed to decrypt the user domain credentials will only be released to the DRM software agent if the PCR values are as expected. The system assumes that if the OS and the application are as expected, then the integrity and secrecy of data is subsequently guaranteed.

2. *Confidentiality and integrity of domain credentials whilst stored in domain devices.* The use of asymmetric encryption provides the confidentiality service. When stored in a domain device, domain credentials are encrypted using a non-migratable key that is bound to the domain device TPM, and only released when the domain device integrity metrics are in a state that matches the values stored with the key. Integrity is provided by inserting an authorisation value into the key object (under the assumption

that the method of encryption used has appropriate properties) that must be provided in order to reveal the key. If the encrypted key object has been altered, the decrypted authorisation value will not match the stored authorisation value.

4. *Entity authentication of a device J to a master control device.* This service is achieved using the TCG challenge-response authentication protocol. This is initiated when the master control device generates a nonce N_1 and sends it in an integrity challenge to J. J replies with an integrity response that includes the master control device's identity, N_1, and J's integrity metrics, i.e. S_J, all signed by J's TPM. In addition, it provides the measured software logs from the Trusted Platform Measurement Store (TPMS), as well as certificates for the measured software. The software measurements and the certificates enable the master control device to verify the current state of J [34]. The master control device verifies the signature and checks that the necessary properties hold for the platform associated with the identity.

9. *Confidentiality and integrity of content C, and integrity of rights object R in a domain device V.* The integrity of R is protected using a digital signature. C is encrypted using the secret key K_T that is itself encrypted using a secret key K_D. As described in section 7.3.4, the symmetric encryption technique in use is assumed to provide authenticated encryption. The key K_D is protected using a non-migratable key that is bound to V's TPM and access control information. The protected storage mechanism is used to ensure that the non-migratable key is only accessed when the device's execution environment state matches that associated with this key.

10. *Confidentiality and integrity of C, and integrity of rights object during execution* is provided exactly as discussed in (1) above.

12. *Confidentiality and integrity of domain credentials whilst stored in a backup device* is provided exactly as discussed in (2) above.

13. *Entity authentication of a backup device to a master control device* is provided exactly as discussed in (4) above.

7.7 Methods of User Authentication

In the system described in section 7.3, users are subject to two-factor authentication that involves "Something the user has", i.e. the master control device, that binds devices joining the domain to itself using the domain key K_D, as described in section 7.2, and either "something the user is", i.e. biometric verification, or "something the user knows", i.e. a password or PIN. The authentication credential, which is kept in the protected storage of the master control device and is associated with the domain key, will thus be either a biometric reference template or a password/PIN. The authentication credential binds the master control device and its domain to a single owner. In the remainder of this section we present the pros and cons of the two approaches to user authentications and possible countermeasures to the identified vulnerabilities.

Using biometric authentication ensures that adding a device to a domain requires the physical presence of the domain owner, which imposes more stringent restrictions on content piracy than use of a password/PIN. Using biometric authentication has the following advantages: biometric features are bound to a person, they cannot be shared, and there is no password to lose or forget. How-

ever, the following possible problems (and possible countermeasures specific to the proposed scheme) are associated with biometric technology.

- Biometric authentication requires biometric samples captured from a live user to be matched against a stored biometric reference template. The processing required to perform this matching might slow down the authentication process; however, in the proposed scheme, biometric authentication is required only when creating a domain and when a device joins or leaves the domain. These are likely to be relatively infrequent events, and hence the use of biometrics will not affect the overall system performance.
- Biometric characteristics are not secret, and can be copied and used to create fake artifacts to gain access to the system. Biometric characteristics can be copied from a variety of sources, such as detached real fingers, collecting fingerprints from surfaces, iris pictures, face pictures, masks, videos, voice recorders, etc. In addition, biometric samples could be copied whilst being transferred from a biometric sensor to a processing device [68, 90]. Two measures need to be implemented to reduce the effect of these problems:

  1. Biometric liveness detection, which cannot be achieved using cryptographic mechanisms, can be achieved using one of the following three techniques: the intrinsic properties of a living body, e.g. physical properties, electrical properties, visual properties, etc; involuntary signs of a living body, e.g. blood pressure, perspiration, brain wave signals, etc; and bodily responses to external stimuli, e.g. blinking, smiling, pupil dilation, etc. For more information about these techniques, see, for example, [90].
  2. Protecting captured biometric samples whilst being transferred from

the biometric sensor to the signal processing subsystem (in the proposed scheme the latter is part of the master control device). This can be implemented using cryptographic techniques; see, for example, [21]. In addition, the biometric reference template needs to be protected inside the master control device. These points are addressed in the proposed scheme, as discussed in section 7.5.

- The extracted biometric samples will vary, even for the same user, so an exact match between extracted features and a stored biometric template cannot be expected. This means that a feature-matching algorithm has associated tolerance settings, so that a sample is considered valid if the difference between the sample and the template is within the tolerance bounds. More relaxed tolerance settings will result in a higher False Acceptance Rate (FAR), and a lower False Rejection Rate (FRR), while stricter tolerance settings will result in a lower FAR and a higher FRR. Moreover, some biometric schemes possess a degree of uncertainty. For example, fingerprint biometric accuracy depends on the position of the finger on the reader, changes in external finger conditions, etc. Very high accuracy in biometric identification typically requires relatively expensive biometric readers such as retina or iris biometric measurement systems; more information can be found in [63, 65].

  In our scheme, the use of more relaxed tolerance settings reduces the effect of FRR problems without raising serious problems because of the higher FAR. This is because the master control device is associated with a single owner and a single domain, and the domain key cannot be replicated on multiple master control devices. These factors reduce the risk that a master control device will be exposed to multiple users, and hence reduce the risks associated with a relatively high FAR.

Using a password/PIN as a method of authentication has the following advantages: it is used directly for user authentication, its verification is a simple process, it does not require excessive storage, and it is the most widely used user authentication method. However, the following possible problems (and possible countermeasures specific to the proposed scheme) are associated with the use of passwords/PINs.

- It can be shared with others. This does not affect our scheme because each password/PIN is bound to a single master control device, within which the domain key is stored. Consequently, sharing the password/PIN does not enable devices to be added to the associated domain without also sharing the associated master control device. This is relatively hard to accomplish, and a master control device owner is not likely to wish to hand over his/her master control device.

- It is not bound to a person. In our scheme, the password/PIN is bound to a single master control device, reducing the significance of this problem.

- It can be forgotten. A password needs to be long and complex to protect against dictionary analysis or brute force attacks, which makes it hard to remember [97]. There are approaches that can be used to help solve this problem, such as implementing a password reminder, implementing a graphical password system [97, 98], implementing a challenge-response scheme, etc. In addition, a challenge-response scheme can be used to help protect against the shoulder surfing problem [98].

- It is subject to both offline and online attacks, as described by Pinkas and Sander [78]. These can be counteracted by: preventing access to the password file, implementing delayed responses, and account locking

that locks a user account for a fixed period after a limited number of unsuccessful login attempts. These measures prevent an attacker from checking a large number of passwords in a reasonable time.

7.8 Summary

This chapter contains a description of an implementation of the authorised domain DRM framework given in chapter 6. In this scheme, domain owners are authenticated using two-factor authentication, involving "something the domain owner has", i.e. a master control device that controls and manages consumers domains, and binds devices joining a domain to itself, and "something the domain owner is or knows", i.e. a biometric or password/PIN authentication mechanism that is implemented by the master control device. These measures establish a one-to-many relationship between the master control device and domain devices, and a one-to-one relationship between domain owners and their master control devices, ensuring that a single consumer owns each domain. This helps to prevent illicit content proliferation.

In addition, we have presented the pros and cons of the two approaches to user authentication, as well as possible countermeasures to the issues identified with these approaches.

Chapter 8

Authorised Domain Management Using an Electronic Payment System

Contents

In this chapter we describe an implementation of the generic authorised domain DRM framework given in chapter 6. This scheme involves managing an authorised domain using a trusted authority which supports: authentication of a domain owner using the domain owner payment cards; domain establishment; adding devices to and removing devices from a domain; and backing up and recovering domain credentials. The system security requirements, threats, and services are analysed. Finally, how the proposed scheme counteracts the main content piracy threats is also discussed..

8.1 Introduction

The system described in this chapter is an implementation of the generic authorised domain DRM framework given in chapter 6. The proposed system works by binding the unique domain key K_D (introduced in section 6.5) to one of the domain owner's payment cards. The domain-specific key is generated automatically by a trusted third party referred to as the trusted authority (TA). Before the TA transfers the domain-specific key to a device being added to a domain, it must authenticate the domain owner and ensure that the joining device belongs to the domain owner. This is achieved by validating the owner's payment card; this involves the TA's Acquirer bank, which contacts the payment card issuer bank to ensure that the payment card used when adding a device to the domain belongs to the creator of the domain.

In addition, the TA ensures that a joining device is in close proximity to another device already in the domain. The existing domain device could be portable,

which provides flexibility to enable devices to be added to the domain at multiple locations (wherever the domain owner with the portable device is physically present). If the domain owner is authenticated successfully, the TA releases the domain key to the joining device. The domain key is then securely stored by the joining device, as described in section 6.5.2. These mechanisms bind the domain key to the domain owner.

One advantage of authenticating domain owners using payment cards is that cardholders typically do not trust other users with their payment card details, which helps to ensure that only a domain owner can authorise devices to join the domain. The use of a physical proximity check with another device already a member of the domain helps to prevent the domain key from spreading via the Internet. However, our proposed scheme does not stop legitimate controlled content sharing or the downloading of digital content from a remote location, as outlined in section 8.3.4.

The system is designed in such a way that payment cards and user identities are not revealed to the TA, which helps to protect consumer privacy. The participating banks and the payment network provider, which have agreed that payment cards and the authorisation network can be used to support this scheme, might wish to charge for this service; however, the banks and the payment network provider are only required when creating a domain and when devices join the domain, and hence the cost impact of a modest charge should be manageable. The extra costs in implementing the solution could be covered from the expected reduction in piracy.

This chapter is organised as follows. Sections 8.2 and 8.3 describe the system and the process workflow. Section 8.4 describes how the scheme controls domain

membership. Section 8.5 analyses the system security requirements, threats, and services. Section 8.6 described how these security requirements can be met using TCG functionality. Section 8.7 discusses alternative methods to implement the proposed scheme. Section 8.8 discusses the main differences between our scheme and other schemes using electronic payment systems for user authentication, and section 8.9 provides conclusions.

8.2 System Model

In addition to the entities identified in chapter 6, the system model includes TAs, payment cards and Banks.

8.2.1 Trusted Authority

The scheme relies on the existence of a TA, a trusted third party that provides the trusted domain controller functionality. We assume the TA has a reliable backup and recovery strategy. Each TA must have an associated Acquiring bank, which it must trust for the purpose of verifying payment card details. Also, each TA has a private signing key used for entity authentication, that is securely stored by the TA. The corresponding public key is certified by the scheme-specific CA (introduced in section 6.4), and the certificate contains a general description of the TA and its security properties. In addition, each TA is assumed to possess an encryption key pair certified by the TA itself using its signing key. Both certificates must be known to all domain devices in domains which the TA supports, and the certificate for the public signature verification key must be known to the TA's Acquirer bank.

A TA is in charge of creating, maintaining and controlling consumer domains. It creates and maintains an identifier (U) and the secret key (K_D) for each domain, which it installs on every device joining that domain. These two values are referred to collectively as the domain credentials. In addition, it controls domain membership by tracking devices joining and leaving a domain, and verifying devices joining a domain. For each domain, the TA maintains the two sequentially incremented domain counters introduced in section 6.5, both of which are initially set to one.

8.2.2 Payment Cards and Banks

For the purposes of this chapter, a payment card can be either a credit or a debit card. Every payment card has a unique number (typically of 16 decimal digits). A domain owner can use multiple payment cards as long as they all belong to him/her. This enables the system to cope with changes in card use.

Debit and credit cards are very widely used for e-commerce. The workflow for a traditional web-based electronic card payment is as follows. A customer first provides his/her payment card number to a merchant via an SSL connection. The merchant then sends the transaction details and the card number to an Acquirer bank for authorisation, through an Acquirer payment gateway. The Acquirer payment gateway contacts the card Issuer bank for authorisation, using the inter-bank private network (e.g. VisaNet or BankNet). The Issuer bank verifies the card number, checks the availability of funds, and then sends an authorisation response back to the Acquirer bank. The Acquirer bank forwards the authorisation response to the merchant. If the Issuer authorises the transaction, the merchant can supply the ordered goods/services to the customer,

and present the charges to the Acquirer bank, which sends a settlement request to the Issuer bank. The Issuer bank then puts the money into an inter-bank settlement account and charges the transaction value to the customer account. More information about electronic payment systems can be found, for example, in [39, 73].

In the scheme described in this chapter, the verification of a domain owner's payment card is performed in a similar way to the electronic payment process described above. This should help ease implementation issues, and increase the acceptability of the scheme.

In addition to the keys it uses for secure payments, each participating Acquirer bank is assumed to have a private signing key used for entity authentication, that it securely stores. The corresponding public key is certified by the scheme-specific CA. This certificate contains a general description of the Acquirer bank and its security properties. In addition, each Acquirer bank is assumed to possess a unique encryption key pair that is certified by the Acquirer bank itself, using its signing key. Also, each domain device needs a trusted copy of both certificates, which could, for example, be obtained from the TA website, the URL for which we assume is listed in the TA certificate. Finally, each card issuing bank participating in the scheme is assumed to know the date of birth and full name of its cardholders, and to have the permission of its cardholders to use this information to support this scheme.

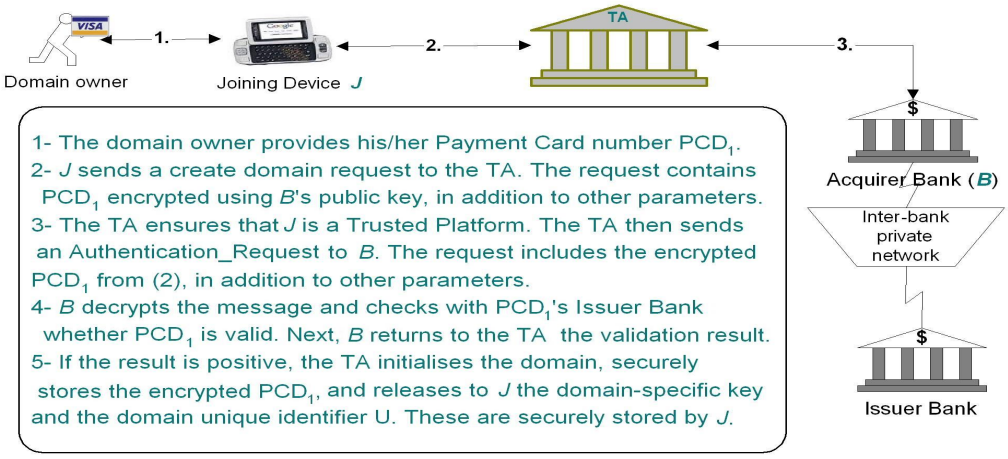


Figure 8.1: Domain Establishment Workflow

8.3 Process Workflow

We now describe in detail how the five phases of the system workflow, as introduced in section 6.5, are implemented. Domain establishment and adding devices into a domain involve interactions between a TA T, an Acquirer bank B, and the DRM software agent running in both a joining device J and an existing domain device V.

8.3.1 Domain Establishment

The domain establishment phase, as described in section 6.5.1, is implemented as follows. This procedure is followed when a consumer wishes to create a domain (and simultaneously add a first device J to this domain). The process workflow is summarised in Figure 8.1. J instructs the domain owner to provide his/her payment card number PCD_1. J then sends a Create_Domain request to T to establish a domain. The request has the form:

(1) $J \rightarrow T$: $P_J||S_J||N_1||A_T||\text{Cert}_{I_J}||\text{Sign}_J(P_J||S_J||N_1||A_T)||M_1$

where $M_1 = e_{P_B}(PCD_1||h(P_J))$ (all other symbols are as defined in section 7.3).

Next, T verifies Cert_{I_J}, extracts I_J from Cert_{I_J}, and checks that I_J has not been revoked, e.g. by querying an OCSP service. T then verifies J's signature using I_J, verifies A_T to ensure that it is the intended recipient, and verifies that the DRM agent is running correctly in J by checking the value of S_J. How this verification occurs is implementation dependent; see, for example, section 8.6. T then generates a nonce N_2, and sends the following message to J:

(2) $T \rightarrow J$: $A_J||N_1||N_2||\text{Sign}_T(A_J||N_1||N_2)$

J verifies T's signature (joining devices have T's certificate, as stated in section 8.2.1), verifies A_J to ensure that it is the intended recipient, and verifies message freshness by comparing N_1 with the value sent in (1). J then sends the following reply to T:

(3) $J \rightarrow T$: $N_2||N_3||\text{Sign}_J(N_2||N_3)$.

T verifies J's signature and checks message freshness by comparing N_2 with the value sent in (2). Steps 1–3 conform to the three-pass mutual authentication protocol described in [46].

Subsequently, T sends the following Authentication_Request to B to request the verification of PCD_1:

(4) $T \rightarrow B$: $M_1||\text{Sign}_T(P_J)$

B decrypts M_1, and then verifies $\text{Sign}_T(P_J)$ using the value of $h(P_J)$ recovered from M_1. This ensures that M_1 was created in the same device that sent the

Create_Domain request, preventing M_1 from becoming a global password. If T's signature is verified successfully, B then sends the following Verification_Request to the card issuer, via the inter-bank private network, to request the verification of PCD_1:

(5) $B \rightarrow PCD_1$_Issuer: PCD_1

The card Issuer verifies that PCD_1 is a currently valid card number, but is not required to verify the name of the card owner. This is because T, for privacy reasons, does not identify the domain by the owner name; T, via its Acquirer bank, ensures that all devices in the domain are owned by the owner of PCD_1 without needing to know the PCD number and owner name, as explained in the next section. The card issuer informs B whether the verification has succeeded by sending the following message:

(6) PCD_1_Issuer $\rightarrow B$: Verification_Result$||PCD_1$.

B verifies PCD_1 equals the value sent in (5). If so, B signs the Verification_Result and M_1, and sends the following message to T:

(7) $B \rightarrow T$: Verification_Result$||M_1||\text{Sign}_B$(Verification_Result$||M_1$).

T checks B's signature, the Verification_Result, and that M_1 equals the value sent in (4); if the checks succeed, T generates the domain-specific secret key K_D and a domain identifier U, and initialises the two counters c_T and c_C to 1, as defined in section 6.5.1. The domain-specific values K_D, U, c_T, c_C, P_J and M_1 are securely stored by T. Subsequently, T sends back to J the domain credentials, as follows:

(8) $T \rightarrow J$: $e_{P_J}(U||K_D)||N_3||\text{Sign}_T(e_{P_J}(U||K_D)||N_3)$.

When J receives this message, it verifies message freshness by comparing N_3 with the value sent in (3). J verifies T's signature and then decrypts $e_{P_J}(U||K_D)$. The key K_D and U are securely stored by J, as described in section 6.5.2. U is used both as a domain identifier and to ensure that joining devices are in close proximity to any domain device before joining a domain, as explained in the next section.

8.3.2 Adding a Device to a Domain

This phase, defined in section 6.5.2, is implemented as follows. In order for a device J to join a domain, it must communicate with T using an existing domain device V as a proxy, as illustrated in Figure 8.2. J instructs the domain owner to provide his/her payment card number. Once the domain owner has provided this number, PCD_2 say, J sends a Join_Domain request to V. The request has the form:

(1) $J \rightarrow V$: $P_J||S_J||N_1||A_T||\text{Cert}_{I_J}||\text{Sign}_J(P_J||S_J||N_1||A_T)||M_2$

where $M_2 = e_{P_B}(PCD_2||h(P_J))$. Next, V checks that J is in close proximity to itself, e.g. by using the Near Field Communication (NFC) protocol, or by measuring the Round-Trip Time (RTT) between V and J, see, for example, [36, 42, 49]. V then creates the string $e_{P_T}(U||h(P_J))$, associates it with J's request, and forwards it to T as follows:

(2) $V \rightarrow T$: $P_J||S_J||N_1||A_T||\text{Cert}_{I_J}||\text{Sign}_J(P_J||S_J||N_1||A_T)||M_2||e_{P_T}(U||\ h(P_J))$

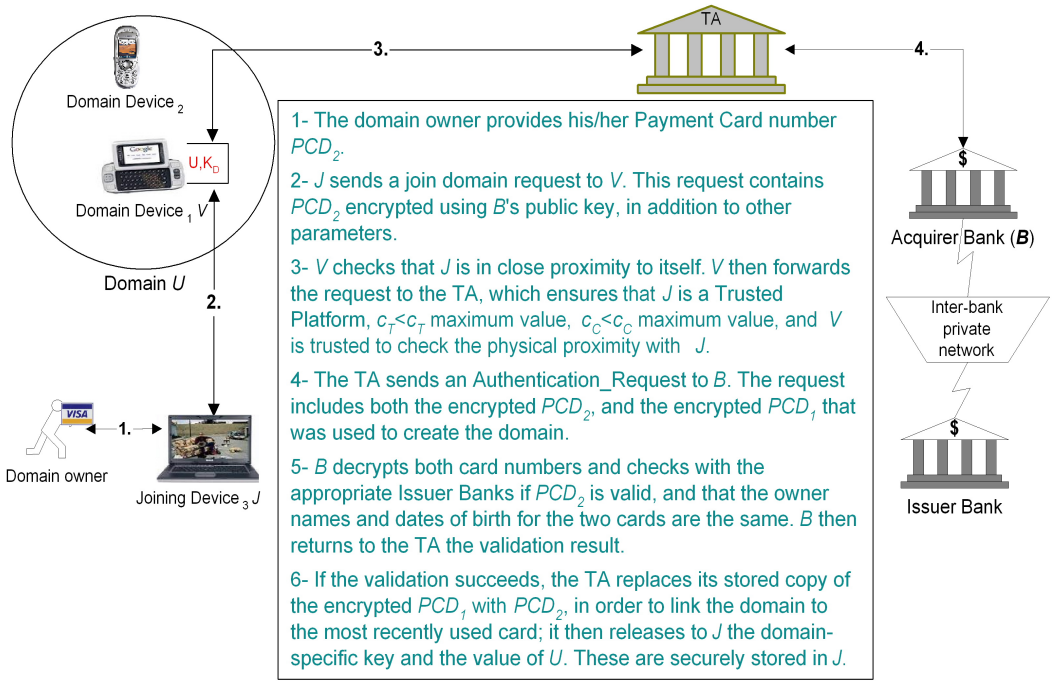


Figure 8.2: Joining Devices Workflow

On receipt of this message, T first decrypts $e_{P_T}(U||h(P_J))$ and verifies that U identifies a valid domain. Because U is securely stored by T, and by devices that are members of U's domain, it is not available even to the domain owner, and it can only be revealed by the trusted DRM software agent. This ensures that the join request is coming via an existing domain device that is trusted to check the physical proximity of J. T then computes $h(P_J)$ from the value of P_J sent in (2) and compares it with the value recovered from $e_{P_T}(U||h(P_J))$. Subsequently, T verifies the state of J, and T and J mutually authenticate each other, as described in steps 1–3 of section 8.3.1. If these processes complete successfully, T checks whether the public key of J is already a member of the domain (a device might need to rejoin a domain, for example, in case of hardware/software failure, as discussed in section 8.3.5). If the public key of J is not in the domain, T temporarily increments the values of c_T and c_C (it does not store the new values at this stage). If both counters do not exceed the maximum value specified by

T for U's domain, T sends an Authentication_Request to B, to ask B to verify that the owner of PCD_2 is the same as the owner of PCD_1, as used to create the domain. The request has the form:

(3) $T \rightarrow B$: $M_1||M_2||\text{Sign}_T(P_J)$

B decrypts M_2, and then verifies $\text{Sign}_T(P_J)$ using the value of $h(P_J)$ recovered from M_2. This ensures that M_2 was created in the same device that sent the Join_Domain request, preventing M_2 from becoming a global password. If T's signature is verified successfully, B then sends, via the inter-bank private network, a Verification_Request including PCD_2 to the card issuer, in the form:

(4) $B \rightarrow PCD_2$_Issuer: PCD_2

The card issuer verifies that PCD_2 is a valid card number, and retrieves the name and the date of birth of the card owner. It then sends to B the following message:

(5) PCD_2_Issuer $\rightarrow B$: Verification_Result$||PCD_2||h(PCD_2$_Owner_details).

where PCD_2_Owner_details contains the name of the owner of the payment card with number PCD_2 and his/her date of birth. If PCD_2 equals the value sent in (4) and the Verification_Result indicates success, B decrypts M_1, and then sends, via the inter-bank private network, a Compare_Owner request message, including PCD_1 and h(PCD_2_Owner_details), to the issuer of the card with number PCD_1 as follows:

(6) $B \rightarrow PCD_1$_Issuer: $PCD_1||h(PCD_2$_Owner_details)

The card issuer retrieves the name and the date of birth of the owner of the card with number PCD_1, and compares their hash output with $h(PCD_2\_\text{Owner\_details})$. It then informs B whether or not they match in the following message:

(7) $PCD_1\_\text{Issuer} \rightarrow B$: $\text{Verification\_Result}||PCD_1$.

B verifies that PCD_1 equals the value sent in (6). If so, B signs the Verification_Result and M_2, using its signing key, and sends the following message to T:

(8) $B \rightarrow T$: $\text{Verification\_Result}||M_2||\text{Sign}_B(\text{Verification\_Result}||M_2)$.

On receipt of this message, T checks B's signature, the Verification_Result, and that M_2 equals the value sent in (3). If the checks succeed, then it replaces its stored copy of M_1 with M_2, in order to link the domain to the most recently used payment card, stores P_J, the updated c_C and c_T values, and releases the domain credentials to J via V as follows:

(9) $T \rightarrow V \rightarrow J$: $e_{P_J}(U||K_D)||\text{Sign}_T(e_{P_J}(U||K_D))$.

When J receives this message, it verifies T's signature and then decrypts $e_{P_J}(U||K_D)$. The key K_D and the identifier U are securely stored by J, as described in section 8.3.1.

8.3.3 Removing a Device from a Domain

This phase is exactly as described in section 6.5.3.

8.3.4 Exchanging Content

This phase is exactly as discussed in section 7.3.4.

8.3.5 Backup and Recovery Procedure

The backup and recovery phase, as described in section 6.5.5, is implemented as follows. There are two options for this phase, where the first is applied if no domain devices can be recovered. We assume that the TA has a backup and recovery strategy, allowing U and K_D to be restored to a domain device. The TA releases the secure domain key K_D and the identifier U to one device in the domain, once the TA is satisfied with the domain owner identity. Subsequently, other devices in the domain must rejoin the domain as described in section 8.3.2.

The second possibility applies when at least one domain device does not require recovery. In this case, the recovery procedure for other domain devices simply involves re-adding them to the domain.

As described in section 8.3.2, before the TA increments the values of the domain counter it checks whether the joining device public key is already a member of the domain; if so, it does not increment the domain counters.

8.4 Controlling Domain Membership

Controlling domain membership is described in section 6.6.2. The scheme described in this chapter provides additional protection to help prevent illicit content proliferation, as it imposes more stringent restrictions on piracy. The do-

main key K_D, which is used to encrypt content encryption keys, is securely stored inside the TA, is not available in the clear, and can only be transferred from the TA to other devices after their physical proximity has been checked. That is, the physical location check, in conjunction with the use of counters (c_C and c_T), addresses the Root Distribution problem.

The only way in which a domain owner could transfer the content protection key to another user's device is by transferring the encrypted domain content, an existing domain device, and the domain owner's payment card details. Whilst possible in principle, such a procedure is unlikely to be attractive to the domain owner. Such a process would also mean that the other user's device would become part of the domain owned by payment card owner, which would mean that fewer of the domain owner's devices could be added to the domain. Most importantly, devices which have joined this domain would not be able to re-transfer the domain key (only the TA can transfer this key to other devices).

8.5 Security Analysis

We now consider the security threats, services, and mechanisms that apply to the storage, execution and transmission of K_D, U, the payment card number, and digital content. The security threats, services, and mechanisms that apply to the transmission of content and rights objects between domain devices, and the security threats, services, and mechanisms that apply to the storage and execution of content and rights objects in domain devices, were discussed in sections 7.5 and 7.6.

Note that in this thesis we do not consider the security threats that applies to

a traditional web-based electronic card payment. Security threats of this type are addressed elsewhere; see, for example, [73]. Following are some examples of such security threats: lack of user authentication resulting in card not present problem, card detail are only protected during transmission making them subject to attack whilst being processed/stored at a merchant server, merchant has clear access to consumer payment card details enabling the merchant to link separate transactions bought using the same card details, which raises a serious privacy concerns.

8.5.1 Security Threats

The main goal of the proposed scheme is to prevent transferring the means for accessing content (i.e. the domain-specific key K_D) to unauthorised devices using physical or digital means. This is achieved by ensuring that this key is transferred to a device only after the domain owner has been authenticated, the values of both domain counters have been incremented, and the trustworthiness of the device has been verified. Hence, only devices which are part of a domain can access the domain content. Devices can only be added to a domain if the domain owner is personally involved, as ensured by authenticating the domain owner. This system has the following requirements.

- The domain owner payment card number PCD_1 should only be accessible to authorised banks, and it should not leak to unauthorised parties, such as the scheme specific TA or content distributors/rights issuers. Therefore, transferring PCD_1 from a joining device J to an Acquirer bank B is vulnerable to the following security threats: (1) unauthorised reading of PCD_1 while being processed on J; and (2) unauthorised reading and

alteration of PCD_1 while in transit.

- The TA must securely sends K_D/U to a device J whilst joining the domain. Such a procedure is exposed to the following security threats: (3) J unwittingly receiving K_D and/or U from a malicious TA T, or an existing domain device unwittingly sending U to a malicious T; (4) unauthorised reading and alteration of K_D/U while in transit; (5) unauthorised manipulation of K_D/U while stored in J; and (6) T unwittingly sending K_D and/or U to a malicious entity.

In addition to the above threats, there is a special type of threat associated with dynamic rights objects, i.e. backing up a dynamic rights object and then restoring it at a later time to reuse an expired licence. As we discussed in section 6.5.4, there is no solution to this threat in authorised domains. In this thesis we do not address this problem, and it remains a possible topic for future research.

8.5.2 Security Services and Mechanisms

The security services required to counteract threats 1, 5 and 6 (listed above) can be provided using trusted platform functionality, as discussed in section 6.2. In section 8.6 we illustrate how such trusted platform functionality can be implemented using a platform conforming to the TCG specifications. Threats 2–4, are addressed using standard cryptographic mechanisms. A direct mapping exists between the threats outlined above and the services and mechanisms outlined below:

1. *Confidentiality and integrity of PCD_1 during use.* Providing this service

requires process isolation techniques, which provide protection for a message against being read or altered by an unauthorised entity whilst being executed [34]. This service can be provided using trusted computing technology, discussed in chapter 3.

2. *Confidentiality and Integrity of PCD_1 while in transit.* This service is provided by encrypting PCD_1 using an asymmetric encryption technique that provides non-malleability — see section 8.3.4.

3. *K_D origin authentication, and U destination verification.* The joining device J checks the origin of K_D by checking the TA's signature on the received encrypted value of K_D — see sections 8.3.1 and 8.3.2. In addition, encrypting U using the TA public key ensures that only the TA can extract U — see section 8.3.2.

4. *Confidentiality and integrity of K_D/U while in transit.* This service is provided by the use of asymmetric encryption and a digital signature — see sections 8.3.1 and 8.3.2.

5. *Confidentiality and integrity of K_D/U whilst stored in a domain device V.* Providing this service requires protected storage, as discussed in section 6.2.

6. *Entity authentication of a joining device J to a TA T.* The provision of this service is implementation-dependent, and involves a protocol exchange between J and T; see, for example, section 8.6. It is initiated when T and J mutually authenticate each other — see sections 8.3.1 and 8.3.2. This mutual authentication attests to the DRM agent execution status, i.e. S_J, and whether the platform is trusted, as discussed in section 6.2.

8.6 Implementing the Protocols Using Trusted Computing

We described in section 6.3 how a system conforming to the TCG specifications [92, 93, 94] can satisfy the requirements described in section 6.2. In this section we consider how security requirements 1, 5 and 6, discussed in section 8.5, can be met using TCG functionality. In this section we do not discuss practical problems associated with TCG specification, as these have been discussed in section 3.8.

1. *Confidentiality and integrity of a PCD number PCD_1 during execution* is provided exactly as discussed in point 1, section 7.6.

5. *Confidentiality and integrity of K_D/U whilst stored in a domain device* is provided exactly as discussed in point 2, section 7.6.

6. *Entity authentication of a joining device J to the TA T* is provided exactly as discussed in point 4, section 7.6.

8.7 Alternative Implementations

Current 'cardholder not present' card payment systems lack user authentication. This has been the main motivation behind the development of two mechanisms supporting payment card owner authentication, namely Verified by Visa (VbV) [95], and the MasterCard Secure Payment Application (SPA) [67]. The method used to authenticate users is not constrained by these schemes, and could be password-based or involve a user token of some kind. These schemes address fraud problems resulting from misuse of payment card numbers by parties other

than the cardholder. It would be desirable to integrate one of these two mechanisms into the scheme described above, as they help to bind a payment card number to its owner. Such an integration could be achieved with either of the two payment schemes. If either SPA or VbV is to be used, extra steps are needed to authenticate the payment card owner to the payment card Issuer bank, as described in [67, 95].

8.8 Related Work

There are other schemes that use payment cards for user authentication in a somewhat different context to that of the scheme discussed above; see, for example, [53, 76]. These two systems differ from the scheme proposed in this thesis in the following ways.

- The schemes described in [53, 76] require the use of payment cards conforming to the EMV[1] industry standard, and also require consumers to possess a card reader that is able to interact with an EMV-compatible card. The payment card authenticates the cardholder using a PIN, and the payment card then vouches for the identity of the cardholder to a third party. The scheme provided in this chapter does not require an EMV compatible payment card, and it also does not require a card reader. The owner provides the payment card details directly to the joining device, which encrypts the card details using the Acquirer back public key.
- The scheme presented here protects consumer privacy, as only an Acquirer bank can access payment card details, unlike the schemes in [53, 76] in which third parties have access to payment card details.

[1] http://www.emvco.com

- The schemes described in [53, 76] involve authenticating users for every transaction. The scheme described in this chapter only involves authenticating a user when a device is added to a domain, and does not require user authentication for downloading and exchanging content between devices.

8.9 Summary

This chapter contains a description of an implementation of the authorised domain DRM framework given in chapter 6. Domain owners are authenticated using their payment cards. This authentication process is implemented using a message flow analogous to that employed in existing electronic payment systems. The scheme ensures that the name and date of birth of a domain creator are the same for all devices joining a domain.

The proposed scheme helps to protect consumers' privacy; domain owner authentication is only required for managing domain membership, and is not required for creating, downloading or exchanging content. In addition, unlike in the payment system with which the cards are routinely used, the card details are not revealed to third parties.

Chapter 9

Authorised Domain Management Using a Mobile Phone

Contents

In this chapter we describe an implementation of the generic authorised domain DRM framework given in chapter 6. This scheme involves managing an

authorised domain using a mobile network operator. This scheme supports: authentication of a domain owner using a domain-specific mobile phone and the general authentication architecture provided by the mobile phone network operator; domain establishment; adding devices to and removing devices from a domain; and backing up and recovering domain credentials. The system security requirements, threats, and services are analysed. Finally, how the proposed scheme counteracts the main content piracy threats is also discussed. Most of the material in this chapter has previously been published in [11].

9.1 Introduction

The system described in this chapter is an implementation of the generic authorised domain management framework given in chapter 6. The authorised domain is managed by a mobile network operator using a domain-specific mobile phone owned by the domain owner. The domain-specific mobile phone generates the unique domain key K_D (introduced in section 6.5). Before the mobile phone transfers the domain-specific key to devices joining the domain, it authenticates the domain owner using a shared secret, e.g. a PIN, and the mobile phone and the mobile network operator are then mutually authenticated using the 3rd Generation Partnership Project (3GPP[1]) Authentication and Key Agreement protocol.

This mutual authentication procedure establishes secret session keys between the mobile phone and the mobile network operator. These session keys are later fetched by a Network Application Function, which maintains and manages consumer domains. The Network Application Function is provided by the mobile

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

network operator as part of its General Authentication Architecture mechanism. Next, the mobile phone ensures a joining device is in close proximity to itself, to prevent devices joining a domain via the Internet.

If the above procedures succeed, the mobile phone releases the domain key to the joining device, which securely protects it as described in chapter 6.5. Only the domain-specific mobile phone can release the domain key to other devices, and this will only take place after authenticating the domain owner. This binds the domain key to the domain owner.

One major advantage of using mobile phones as domain controllers is that mobile phones are personalised, portable (enabling a domain owner to add devices wherever he/she is physically present) and ubiquitous. In addition, the existing mobile network infrastructure enables the authentication of subscribers, and is capable of providing a Network Application Function service as part of the 3GPP General Authentication Architecture; we employ this infrastructure as a means of managing consumer domains. The participating network operators might want to charge for their service; however, network operator support is only required when creating a domain, and when devices join the domain, and hence the cost impact of a modest charge should be manageable. The extra costs in implementing the solution could be covered from the expected reduction in piracy.

This chapter is organised as follows. Section 9.2 describes the 3GPP General Authentication Architecture. Sections 9.3 and 9.4 describe the proposed solution and the process workflow. Section 9.5 discusses controlling domain membership. Section 9.6 analyses the system security requirements, threats, and services. Section 9.7 described how these security requirements can be met using TCG

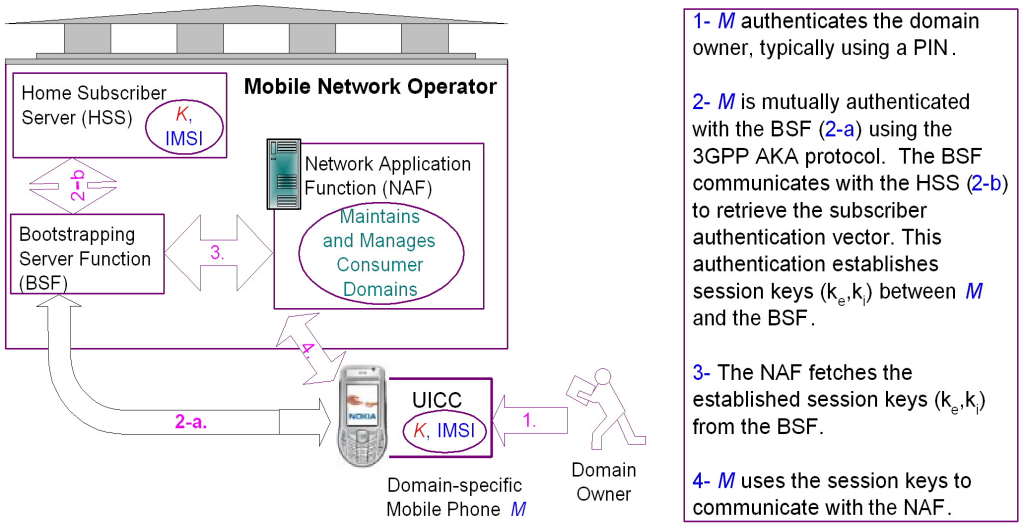


Figure 9.1: The 3GPP General Authentication Architecture

functionality. Section 9.8 discusses the main differences between our scheme and other schemes which use GSM and UMTS user authentication, and section 9.9 provides conclusions.

9.2 General Authentication Architecture

The scheme we propose relies on the mobile network operator supporting the 3GPP General Authentication Architecture (GAA) [6, 7]. Specifically we use the General Bootstrapping Architecture mechanism that builds upon the secret key K shared by the UMTS IC Card (UICC) in the mobile phone and the mobile network operator's Home Subscriber Server (HSS). The General Bootstrapping Architecture incorporates two network elements, known as the Bootstrapping Server Function (BSF) and the Network Application Function (NAF). Figure 9.1 summarises the GAA workflow.

The Bootstrapping Server Function has an interface with the Home Subscriber Server and with the Network Application Function, and is part of the mobile network operator. The Bootstrapping Server Function acts as an intermediary between the mobile phone, the Home Subscriber Server and the Network Application Function. The mobile phone and the Bootstrapping Server Function are mutually authenticated using the 3GPP Authentication and Key Agreement (AKA) protocol [6, 72]. This mutual authentication procedure establishes a secure session that is later fetched by the Network Application Function from the Bootstrapping Server Function, and is then used to secure communications between the Network Application Function and the mobile phone. Communications between the Network Application Function and the Bootstrapping Server Function, and between the Bootstrapping Server Function and the Home Subscriber Server, are beyond the scope of this chapter; details can be found in [6].

The 3GPP Authentication and Key Agreement protocol works as follows. The mobile phone sends an authentication request to the Bootstrapping Server Function, which then contacts the Home Subscriber Server and retrieves the subscriber security settings and an authentication vector RAND||AUTN||XRES||k_e||k_i. This vector is calculated using the mobile-specific secret key K, a random challenge RAND, and a set of functions shared by Home Subscriber Server and the mobile phone. XRES is the "expected response", used later to authenticate the mobile phone, k_e is the session Cipher Key, k_i is the session Integrity Key, and AUTN is equal to SQN$\oplus$AK||AMF||MAC, where SQN is a sequence number, AK is the Anonymity key, AMF is the Authentication management field, and MAC is a Message Authentication Code for the string SQN||RAND||AMF, computed using the key K.

Next, the Bootstrapping Server Function forwards AUTN and the challenge (RAND) to the mobile phone, which uses its UICC to calculate k_i, k_e, RES, and AK, and recover AMF and MAC from AUTN. The UICC then recovers SQN from the calculated AK and the received SQN⊕AK, and uses this recovered value to verify the freshness of the received message. The UICC calculates the MAC, and then compares it with the MAC recovered from AUTN. If they are equal, then the mobile phone has successfully checked the validity of the network. Subsequently, the mobile phone sends an Authentication and Key Agreement Digest calculated using RES, to the Bootstrapping Server Function. The Bootstrapping Server Function recomputes the Digest, using XRES, and deems the mobile phone authenticated if this recomputed value equals the received value.

The Bootstrapping Server Function and the mobile phone use the session keys k_e and k_i to establish a secure channel that protects messages exchanged between them, and between the mobile phone and the Network Application Function. The secure channel is implemented using a stream cipher and a MAC function.

9.3 System Model

In addition to the entities identified in chapter 6, the system model includes Mobile Phones and Mobile Network Operators. The Mobile Phone, which provides the trusted domain controller functionality, must have the TP features described in section 6.2 and must satisfy the domain device assumptions described in section 6.4. The Mobile Network Operator provides the trusted authority functionality.

Our DRM scheme requires that the domain-specific mobile phone which acts as the trusted domain controller is owned by the domain owner, is registered by the mobile network operator and is equipped with a special application. The domain-specific mobile phone should be compatible with the 3GPP specifications [6], and is responsible for: authenticating the domain owner using a shared secret, e.g. a PIN; mutually authenticating itself with the mobile network operator; creating and maintaining the domain unique secret key K_D introduced in section 6.5; and authorising devices to join its domain by ensuring that they in physical proximity to itself, and that their processing environment is trusted. The key K_D is not available in the clear, and only the mobile phone acting as the trusted domain controller is able to copy this key to a device joining the domain.

The mobile phone acting as the trusted domain controller maintains the two counters c_T and c_C (as introduced in section 6.5.1), both of which are initially set to one. Both counters have domain-specific limits, denoted by L_T and L_C (for c_T and c_C, respectively).

For the purposes of our scheme we assume the Network Application Function provides the following functions: initialising and maintaining domain-specific limits, i.e. L_T and L_C, and backup and recovery of the domain-specific secrets, K_D, c_C and c_T. The phone obtains L_T and L_C from the Network Application Function. Both limits can be increased by the mobile network operator. Consumers could be charged more for higher maximum values.

9.4 Process Workflow

We now describe in detail how the five phases of the system workflow, as introduced in section 6.5, are implemented. Before performing any of these processes, an initialisation procedure must be performed between the mobile network operator and the mobile phone. We now describe each of these processes.

9.4.1 Initialisation Procedure

The initialisation of a mobile phone by a mobile network operator involves the authentication of the phone and the establishment of a secret session key between the phone and the Network Application Function. Before creating a domain, or adding or removing devices from a domain, the domain-specific mobile phone M authenticates the domain owner by instructing the domain owner to provide a secret key, e.g. a PIN, shared by the domain owner and M. Once the domain owner has been authenticated, M and the Bootstrapping Server Function perform mutual authentication. As described in section 9.2, this process establishes session keys k_e and k_i, later fetched by the Network Application Function F from the Bootstrapping Server Function, and used to secure communications between F and M.

9.4.2 Domain Establishment

The domain establishment phase, as described in section 6.5.1, is implemented as follows. This phase applies when a consumer wishes to create a domain (and simultaneously add the domain-specific mobile phone M to the domain). We assume the initialisation procedure described in section 9.4.1 has already been

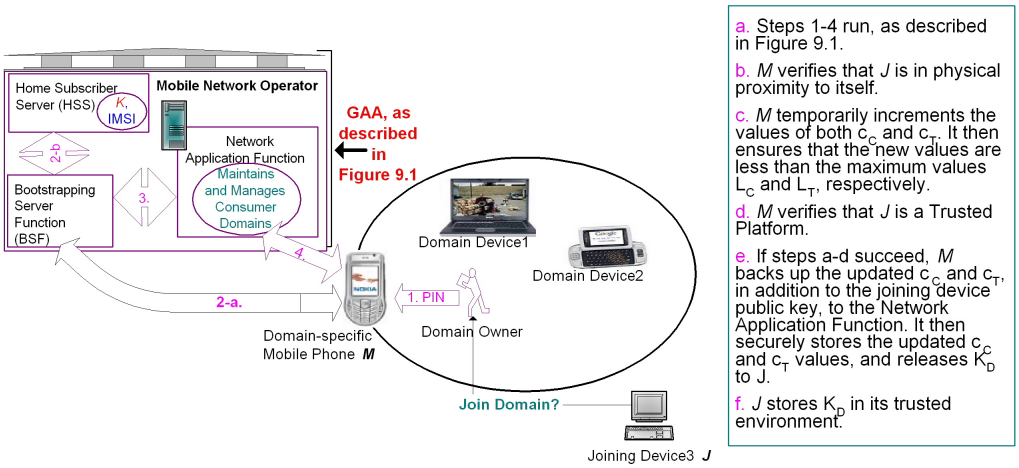


Figure 9.2: DRM System Workflow

performed. M securely generates the domain-specific secret key K_D using a random number generator, and initialises counters c_C and c_T to one. M then sends a Create_Domain request to the Network Application Function F, via the pre-established secure session, to establish a domain. The request has the form:

(1) $M \rightarrow F$: $g_{k_e}(K_D||IMSI)||m_{k_i}(g_{k_e}(K_D||IMSI))$

where $g_{k_e}(Y)$ denotes the symmetric encryption of data Y using key k_e, $m_{k_i}(Y)$ denotes a MAC computed on data Y using key k_i, and IMSI is the International Mobile Subscriber Identity that is used to uniquely identify the domain. When a user changes his/her IMSI, e.g. if the user changes his/her network operator, the new network operator obtains the user domain information from the old network operator (how this occurs is likely to be network operator specific).

Next, F verifies the integrity of the received message and decrypts it, and then initialises the domain, i.e. sets the domain limits L_C and L_T, securely associates them with K_D and IMSI, and stores them in its protected storage. F then transfer these limits to M in the following message:

(2) $F \rightarrow M$: $g_{k_e}(L_C||L_T)||m_{k_i}(g_{k_e}(L_C||L_T))$.

M verifies the integrity of the received message and decrypts it, and then securely associates L_C and L_T with K_D, c_C, and c_T, and stores them in its protected storage.

9.4.3 Adding a Device to a Domain

This phase, defined in section 6.5.2, is implemented as follows. We assume that the procedures described in sections 9.4.1 and 9.4.2 have already been performed. In order for a device J to join a domain, it must communicate with the domain-specific mobile phone M, as shown in Figure 9.2. J first sends a Join_Domain request to M. The request has the form:

(1) $J \rightarrow M$: $P_J||S_J||N_1||\text{Cert}_{I_J}||\text{Sign}_J(P_J||S_J||N_1)$

Next, M checks that J is in physical proximity to itself, e.g. by using the Near Field Communication (NFC) protocol or by measuring the Round-Trip Time (RTT) between M and J, see, for example, [36, 42, 49]. M verifies Cert_{I_J}, extracts I_J, and checks that it has not been revoked, e.g. by querying an OCSP server. M then verifies J's signature using I_J, and verifies that the DRM agent is running correctly in J by checking the value of S_J. How this verification occurs is implementation-dependent, see, for example, section 6.3. M then generates a nonce N_2, and sends the following message to J:

(2) $M \rightarrow J$: $\text{A}_J||N_1||N_2||S_M||Cert_{I_M}||\text{Sign}_M(\text{A}_J||N_1||N_2||S_M)$

J verifies $Cert_{I_M}$, extracts I_M from $Cert_{I_M}$ and verifies that it has not been revoked, e.g. by querying an OCSP service. J then verifies M's signature by using I_M, verifies A_J to ensure that it is the intended recipient, and verifies message freshness by comparing N_1 with the value sent in (1). J then verifies that the DRM agent is running correctly in M by checking the value of S_M. As above, how this verification occurs is implementation-dependent. J then sends the following reply to M:

(3) $J \rightarrow M$: $N_2||N_3||\mathrm{A}_M||\mathrm{Sign}_J(N_2||N_2||A_M)$

M verifies J's signature, verifies A_M to ensure that it is the intended recipient, and checks message freshness by comparing N_2 with the value sent in (2). Steps 1–3 conform to the three-pass mutual authentication protocol described in [46]. M temporarily increments the values of both c_T and c_C. If they are greater than the maximum permitted value (L_T and L_C, respectively) held by M, then the agent running on M exits with an appropriate error message. M now sends F the following backup request for the values of c_T, c_C and P_J:

(4) $M \rightarrow F$: $g_{k_e}(c_T||c_C||P_J||\mathrm{IMSI})||m_{k_i}(g_{k_e}(c_T||c_C||P_J||\mathrm{IMSI}))$

F verifies the integrity of the received message and decrypts it, and then securely associates the new values with the domain identified by IMSI. Subsequently, F sends an acknowledgment back to M as follows:

(5) $F \rightarrow M$: $g_{k_e}(\mathrm{Result}||c_T||c_C)||m_{k_i}(g_{k_e}(\mathrm{Result}||c_T||c_C))$

M verifies the integrity of the received message and decrypts it. M checks message freshness by comparing c_C and c_T with the values sent in (4). M then

check the value of Result, that can be set to either success or fail. If it indicates success, M stores the c_T and c_C values in its trusted storage, and sends the key K_D to J in the following message:

(6) $M \rightarrow J$: $e_{P_J}(K_D)||N_3||\text{Sign}_M(e_{P_J}(K_D)||N_3)$

When J receives this message, it verifies the signature, and then checks message freshness by comparing N_3 with the value sent in (3). J then decrypts $e_{P_J}(K_D)$. The key K_D is then securely stored by J, as described in section 9.4.2.

9.4.4 Removing a Device from a Domain

This phase is exactly as described in section 6.5.3.

9.4.5 Exchanging Content

This part is exactly as discussed in section 7.3.4.

9.4.6 Backup and Recovery Procedure

The backup and recovery phase, as described in section 6.5.5, is implemented as follows. The domain-specific mobile phone M is required to communicate with the Network Application Function regularly, as described in section 9.4.3, and backup K_D, c_T and c_C.

If the mobile phone M cannot be recovered, for example, because it has been lost or stolen, the domain owner must inform the mobile network operator. The

mobile network operator will blacklist both M and the UICC, and then issue a new UICC. The replacement mobile phone will need to contact the Network Application Function and restore the domain settings stored by the Network Application Function. The Network Application Function then releases K_D, c_C, c_T, L_C and L_T, to the new domain-specific mobile phone. Subsequently, other devices in the domain that cannot recover K_D must re-join the domain, as described in section 9.4.3 (before M increments the values of the domain counters it checks with the Network Application Function whether the joining device public key is already a member of the domain; if so it does not increment the counters).

9.5 Controlling Domain Membership

The domain-specific mobile phone M controls domain membership in the same way as described in section 6.6.2. In addition, it imposes stringent restrictions on piracy using digital media such as the Internet. This is because, as described in section 9.3, the content protection key K_D is securely stored inside M, is not available in the clear, and can only be transferred from M to other devices after their physical proximity has been checked. That is, the physical proximity check, in conjunction with the use of counters, addresses the Root Distribution problem.

Our solution stops illicit content proliferation; the only way in which a domain owner could transfer the content protection key to another user's device is by transferring the encrypted domain content, the domain-specific mobile phone M, and the domain owner's authentication credential for M, e.g. a PIN. Whilst possible in principle, such a procedure is unlikely to be attractive to the domain

owner. Such a process would also mean that the other user's device would become part of the domain controlled by M, which would mean that fewer of the domain owner's devices could be added to the domain. Most importantly, devices which have joined this domain using M would not be able to re-transfer the domain key (as described above, only M can transfer this key to other devices).

9.6 Security Analysis

We now consider the security threats, services, and mechanisms that apply to the storage, execution and transmission of K_D, c_T, c_C, L_T, L_C, and digital content. In addition, we address the security threats, services, and mechanisms that apply to communications between the mobile device and the Network Application Function; however, we do not address threats to the 3G security system itself. Security threats of this type are addressed elsewhere; see, for example, [2, 3, 4]. Following are few examples of threats in a 3G security system: man-in-the-middle-attack impersonating a genuine GSM user to the BSF, attacks on the terminal and UICC/USIM (e.g. use of a stolen terminal and UICC, manipulation of the identity of the terminal, manipulation of data on the UICC-terminal interface), and attacks on the radio interface (e.g. eavesdropping user traffic, passive/active traffic analysis, masquerading as another user).

The security threats, services, and mechanisms that apply to the transmission of content and rights objects between domain devices, and the security threats, services, and mechanisms that apply to the storage and execution of content and rights objects in domain devices, are covered in section 7.5 and 7.6.

9.6.1 Security Threats

The main goal of the proposed scheme is to prevent transferring the means for accessing content (i.e. the domain-specific key K_D) to unauthorised devices using physical or digital means. This is achieved by ensuring that this key is transferred to a device only after the domain owner has been authenticated, the values of both domain counters have been incremented, and the trustworthiness of the device has been verified. Hence, only devices which are part of a domain can access the domain content. Devices can only be added to a domain if the domain owner is personally involved, as ensured by authenticating the domain owner. This system has the following requirements.

- The domain-specific mobile phone M must securely processes and stores the domain-specific values K_D, c_T, c_C, L_T and L_C, so that it is not available in the clear even to domain owner. This process is vulnerable to the following security threats: (1) unauthorised manipulation of the domain-specific values during use in M; and (2) unauthorised manipulation of the domain-specific values whilst stored in M.

- M must securely sends the key K_D to a device J whilst joining the domain. Such a procedure is exposed to the following security threats: (3) unauthorised reading or updating of K_D whilst in transit; (4) M unwittingly sending K_D to a malicious entity; (5) J unwittingly receiving K_D from a malicious M; and (6) replay of communications between M and J.

- Security threats related to the exchange of messages between a mobile phone M and the Network Application Function F are: (7) unknowingly,

M communicating with a malicious entity, or F communicating with a malicious entity; (8) replay of communication between M and F; and (9) unauthorised reading or updating of exchanged messages.

In addition to the above threats, there is a special type of threat associated with dynamic rights objects, i.e. backing up a dynamic rights object and then restoring it at a later time to reuse an expired licence. As we discussed in section 6.5.4, there is no solution to this threat in authorised domains. In this thesis we do not address this problem, and it remains a possible topic for future research.

9.6.2 Security Services and Mechanisms

The security services required to counteract threats 1, 2, and 4 (listed above) can be provided using trusted platform functionality, as discussed in section 6.2. In section 9.7 we illustrate how such trusted platform functionality can be implemented using a platform conforming to the TCG specifications. Threats 3 and 5–9 are addressed using standard cryptographic mechanisms. A direct mapping exists between the threats outlined above and the services and potential mechanisms outlined below:

1. *Confidentiality and integrity of the domain-specific values during execution on M.* Providing this service requires process isolation techniques, which provide protection for a message against being read or altered by an unauthorised entity whilst being executed [34]. This service can be provided using trusted computing technology, discussed in chapter 3.

2. *Confidentiality and integrity of the domain-specific values whilst stored in M.* Providing this service requires protected storage, typically by stor-

ing domain credentials inside a tamper-resistant module, as discussed in section 6.2.

3. *Confidentiality and integrity of K_D whilst in transit.* This service is provided by the use of symmetric encryption and a MAC, see section 9.4.2, or asymmetric encryption and a digital signature, see section 9.4.3.

4. *Entity authentication of a joining device J to the domain-specific mobile phone M.* The provision of this service is implementation-dependent, and involves a protocol exchange between J and M; see, for example, section 6.3. It is initiated when M and J mutually authenticate each other — see section 9.4.3. This mutual authentication attests to the DRM agent execution status, i.e. S_J, and whether the platform is trusted, as discussed in section 6.2.

5. *K_D origin authentication.* The joining device J checks the origin of K_D by checking M's signature on the received encrypted value of K_D — see section 9.4.3.

6. *Prevention of replay of communications between M and a device.* This is provided by the inclusion of nonces in messages — see section 9.4.3.

7–9. These threats are counteracted by the use of the secure session established between M and F, as described in section 9.2; mutual authentication between M and F counteracts threat 11; threat 12 is counteracted by the inclusion of nonces — see section 9.4.3; and threat 13 is counteracted by the use of symmetric encryption and a MAC — see section 9.4.2 and 9.4.3.

9.7 Implementing the Protocols Using Trusted Computing

We described in section 6.3 how a system conforming to the TCG specifications [92, 93, 94] can satisfy the requirements described in section 6.2. In this section we consider how security requirements 1, 2, and 4, discussed in section 9.6, can be met using TCG functionality. In this section we do not discuss practical problems associated with TCG specification, as these have been discussed in section 3.8.

1. *Confidentiality and integrity of K_D, c_T, c_C, L_C and L_T during execution* is provided exactly as discussed in point 1, section 7.6.

2. *Confidentiality and integrity of K_D, c_T, c_C, L_C and L_T whilst stored in domain devices* is provided exactly as discussed in point 2, section 7.6.

4. *Entity authentication of a device J to the domain-specific mobile phone M* is provided exactly as discussed in point 4, section 7.6.

9.8 Related Work

Schemes have previously been proposed that exploit the GSM and UMTS user authentication schemes in other applications — see, for example, [54, 55, 75]. However, not only are the proposed application domains rather different to that applying here, but the protocols themselves have significant differences.

- The schemes described in [54, 55, 75] require a mobile phone to directly communicate with a user device, which then communicates with a veri-

fier; i.e. the user device acts as an intermediary between the mobile phone and a verifier. The verifier validates the identity of the user by sending messages to the mobile phone, and then validates the responses to these messages provided by the mobile phone by communicating with the mobile phone network operator. However, our scheme does not require the user to directly prove his/her identity to the verifier, as the user communicates directly with the mobile network operator using GAA. This protects consumer privacy, as users are authenticated directly by the mobile network operator itself.

- The schemes given in [54, 55] require user verification to be performed for every system transaction. However, the system described here does not require verifiers to authenticate a user every time a transaction is processed, as a user is authenticated only when a device is added to the domain.

9.9 Summary

This chapter focuses on the problem of preventing illegal copying of digital assets without jeopardising the right of legitimate licence holders to transfer content between their own devices, where these devices make up a user domain. The scheme proposed here involves the use of a domain-specific mobile phone and the mobile phone network operator to authenticate the domain owner before devices can join a domain. This binds devices in a domain to a single owner, that, in turn, enables the binding of domain licences to the domain owner. In addition, the way in which we control domain membership, and the use of the domain-specific mobile phone that enables a domain owner to add devices wherever

he/she is physically present, ensures that devices joining the domain are in physical proximity to the mobile phone, preventing illicit content proliferation.

Chapter 10

Authorised Domain Management Using Location Based Services

Contents

In this chapter we describe an implementation of the generic authorised domain DRM framework given in chapter 6. This scheme involves managing an autho-

rised domain using a trusted authority. This scheme supports: authentication of a domain owner using Location Based Services, ensuring that devices, when joining a consumer domain, are located in physical proximity to the domain registered addresses; domain establishment; adding devices to and removing devices from a domain; and backing up and recovering domain credentials. The system security requirements, threats, and services are analysed. Finally, how the proposed scheme counteracts the main content piracy threats is also discussed. Most of the material in this chapter has previously been published in [9].

10.1 Introduction

The system described in this chapter is an implementation of the generic authorised domain management framework given in chapter 6. The authorised domain is managed by a special-purpose third party known as a Trusted Authority (TA), which generates the unique domain key K_D (introduced in section 6.5) and binds it to the domain owner's registered geographical addresses. Before the TA transfers the domain-specific key to devices joining a domain, it must ensure that the joining device belongs to the domain owner. This is achieved by ensuring that the joining device is located close to one of the domain owner registered addresses. This is enforced by requiring the joining device to communicate with the TA either directly or via another device already in the domain as a proxy.

In the first case the joining device must be able to determine its own location, e.g. by incorporating a Global Positioning Satellite (GPS) receiver, or be capable of having its location determined by a third party, e.g. as is the case for mobile phones. In the second case the joining device does not need to be able to

determine its location, and a device already in the domain (which is capable of determining its location or of having its location determined) acts as a proxy. The proxy device must ensure that the joining device is in physical proximity to itself. The TA authenticates the domain owner using a password, and verifies that the joining device (or the proxy device) is close to the location of one of the domain registered addresses.

If the above procedure succeeds, the TA releases the domain key to the joining device, which is then protected by the device, as described in chapter 6.5. Only the TA can release the domain key to other devices, and this will only occur after it has authenticated the domain owner. These mechanisms bind the domain key to the domain owner.

The physical proximity check prevents devices joining a domain via the Internet. However, the proposed scheme does not stop legitimate controlled content sharing or downloading of digital content from a remote location; location-based service is only required for managing domain membership, and is not required for creating, downloading or exchanging content (these issues are discussed further in section 10.4.4).

This chapter is organised as follows. Section 10.2 briefly describes the IETF Geopriv protocol. Sections 10.3 and 10.4 describe the proposed solution and its process workflow. Section 10.5 describes how to control domain membership. Section 10.6 analyses the system security requirements, threats, and services. Section 10.7 described how these security requirements can be met using TCG functionality. Section 10.8 discusses an alternative implementation using 3GPP[1] Location Based Services. Section 10.9 discusses the main differences between

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

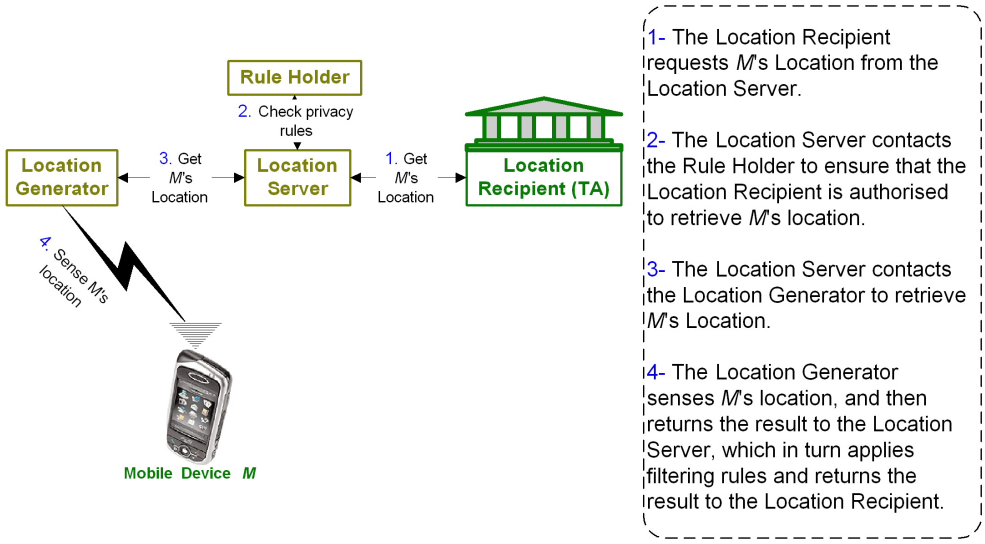


Figure 10.1: The Geopriv Protocol Workflow

our scheme and other schemes that use Location Based Services for user authentication, and section 10.10 provides conclusions.

10.2 The Geopriv Protocol

The scheme we describe in this chapter is based on the IETF Geographic location/privacy protocol (Geopriv) [23] (in section 10.8 we describe an alternative implementation of the scheme using the 3GPP Location Based Service (LBS)). The Geopriv model involves the following main entities: Location Generator, Location Server, Location Recipient (which corresponds to the TA in the DRM scheme), Rule Maker (which acts as the domain owner in the DRM scheme), and Rule Holder. The Geopriv protocol requires the Location Server and the Location Recipient, the Location Server and the Rule Holder, the Location Server and the Location Generator, and the Mobile Device and the Location Generator, to mutually authenticate each other.

The Mobile Device is owned by the Rule Maker, and should be able to calculate its own position or be capable of being located, e.g. by being equipped with a radio receiver able to receive Global Navigation Satellite System (GNSS) signals. The Rule Maker defines privacy protection rules and filters, which are stored by the Rule Holder. These rules and filters are used to determine whether a Location Recipient can be sent the location of the Mobile Device, and how the Location Server should filter any particular Location for the Location Recipient. Filtering is the process of reducing the precision or resolution of location data. The Location Server acts as an intermediary in communication between the Location Recipient, the Rule Holder and the Location Generator.

When the Location Server receives a location measurement request from the Location Recipient, it applies the privacy protection rules and filters defined by the Rule Maker for the Location Recipient. If the Location Recipient is authorised to retrieve the location of the Mobile Device, the Location Server forwards the request to the Location Generator.

Next, the Location Generator obtains and/or measures the Mobile Device location, and creates a location object describing the Mobile Device location (for possible means of locating a Mobile Device see, for example, [5, 52, 60]). The Location Generator then transfers the location object back to the Location Server, which specifies whether the Location Recipient can receive the location object and how it should be filtered for this Location Recipient.

10.3 System Model

In addition to the entities identified in chapter 6, the system model includes TAs. Each TA is required to have a private signing key used for entity authentication. The corresponding public key is certified by the scheme-specific CA (as described in section 6.4), and the certificate contains a general description of the TA and its security properties. In addition, each TA is assumed to possess an encryption key pair certified by the TA itself using its signing key. These certificates must be available to all domain devices in domains which the TA supports.

10.4 System Workflow

We now describe in detail how the five phases of the system workflow, as introduced in section 6.5, are implemented. We assume that the Location Server and the TA, the Location Server and the Rule Holder, the Location Server and the Location Generator, and the Mobile Device and the Location Generator, communicate via channels offering integrity and confidentiality protection. These secure channels and their properties are implementation-dependent.

10.4.1 Domain Establishment

This phase, defined in section 6.5.1, is implemented as follows. The domain owner must register with a TA, e.g. by connecting to the TA website over an SSL session. During registration the domain owner completes an application form, containing the primary domain street address and (possibly) alternative addresses. The alternative addresses provide flexibility in cases where a domain

has multiple locations, e.g. if the domain is owned by an organisation that has multiple branches, or the domain owner lives in more than one place.

The TA then validates the domain owner request and applies its registration policies, e.g. the maximum number of alternative addresses per domain, the maximum domain size, etc. In some cases the TA might request additional credentials from the domain owner, e.g. a public key certificate for an organisation or a proof of residence for a domain owner. The domain owner could be charged for a larger maximum domain size, or for a larger maximum number of alternative locations. The system design allows the domain owner to subsequently update these values, subject to the policy of the TA.

Next, the TA generates a domain-specific secret key K_D, a domain unique identifier U, and the two counters c_T and c_C, as defined in section 6.5.1. Each counter has a domain-specific limit, set and maintained by the TA. The domain-specific values K_D, U, c_T, c_C, and the domain addresses, are securely stored by the TA. The TA also sends U to the domain owner; U is later used as a password to authenticate the domain owner when adding new devices. This ensures that devices joining the domain are authorised by the domain owner, as described below.

10.4.2 Adding a Device to a Domain

The joining domain phase, as described in section 6.5.2, is implemented as follows. In order for a device J to join a domain, J must communicate with the TA T. As described earlier, this communication takes place either directly, if J has the necessary properties, or using an appropriate domain device M as

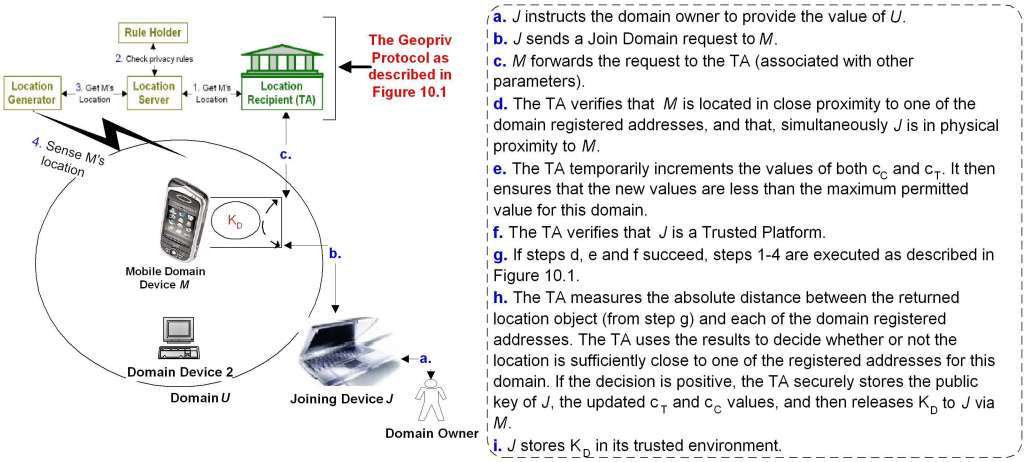


Figure 10.2: DRM System Workflow Using a Proxy

a proxy.

10.4.2.1 Adding a Device to a Domain Without a Proxy

This procedure is followed when a consumer wishes to add a device J to a domain by communicating directly with the TA T (Figure 10.2 summarises the main steps for adding a device to a domain using a proxy (the same steps apply for adding a device without a proxy; however, in steps (b, c) J sends a join domain request directly to the TA)). J is assumed to have a unique identifier ID_J that is assigned by its service provider; for example, for a 3GPP LBS, ID_J would typically be the mobile phone IMSI (International Mobile Subscriber Identity). J sends a Join_Domain request to T. The request has the form:

$$(1)\ \ J \rightarrow T\text{: } S_J||P_J||ID_J||A_T||N_1||\text{Cert}_{I_J}||\text{Sign}_J(S_J||P_J||ID_J||A_T||N_1)$$

Next, T verifies Cert_{I_J}, extracts I_J, and checks that it has not been revoked, e.g. by querying an OCSP service. T then verifies J's signature using I_J, verifies

A_T to ensure that it is the intended recipient (a joining device must have a copy of T's certificate, as stated earlier), and verifies that the DRM agent is running correctly on J by checking the value of S_J. How this verification occurs is implementation-dependent, see, for example, section 10.7. T then generates a nonce N_2, and sends the following message to J:

(2) $T \rightarrow J$: $A_J||N_1||N_2||\text{Sign}_T(A_J||N_1||N_2)$

J verifies T's signature, verifies A_J to ensure that it is the intended recipient, and verifies message freshness by comparing N_1 with the value sent in (1). J instructs the domain owner to provide the value of U, and then sends the following reply to T:

(3) $J \rightarrow T$: $e_{P_T}(U||N_2)||N_3||\text{Sign}_J(e_{P_T}(U||N_2)||N_3)$.

T verifies the signature, decrypts $e_{P_T}(U||N_2)$, and checks message freshness by comparing N_2 with the value sent in (2). T then verifies that U identifies a valid domain. Steps 1–3 conform to the three-pass mutual authentication protocol described in [46].

If the authentication protocol succeeds, T checks whether the public key of J is already a member of the domain (a device might need to rejoin a domain, for example, in the event of hardware/software failure, as discussed in section 10.4.5). If the public key of J is not in the domain, T temporarily increments the value of c_T and c_C (it permanently stores the result at a later stage, i.e. just before step 9). If the counters do not exceed the maximum value specified by T for U's domain, T sends the following Get_Location request to the Location Server, via the pre-established secure channel:

(4) $T \rightarrow$ Location Server: $ID_J||A_T$

The Location Server checks with the Rule Holder if T is authorised to retrieve J's location, as follows:

(5) Location Server $\rightarrow$ Rule Holder: $ID_J||A_T$

The Rule Holder checks the rules associated with ID_J, and sends the result to the Location Server, as follows:

(6) Rule Holder $\rightarrow$ Location Server: Result$||ID_J$

If the value of Result is positive, and ID_J equals the value sent in (5), then the Location Server forwards the request to the Location Generator as follows:

(7) Location Server $\rightarrow$ Location Generator: ID_J

The Location Generator contacts J, as identified by ID_J, and retrieves its location L in some way; see, for example, [5, 52, 60, 99]. The Location Generator sends L to the Location Server, which sends it back to T as follows:

(8) Location Generator $\rightarrow$ Location Server $\rightarrow T$: $L||ID_J$

Next, T checks that L represents a valid location object and that ID_J equals the value sent in (4). If the validation succeeds, T measures the absolute distance between L and each of the domain registered address. T uses the results to decide whether or not location L is sufficiently close to one of the registered addresses for this domain. If the decision is positive, T securely stores P_J and the updated values of c_C and c_T, and releases K_D to J, as follows:

(1) $J \rightarrow M \rightarrow T$: $N_1||S_J||P_J||A_T||\text{Cert}_{I_J}||$
$\text{Sign}_J(N_1||S_J||P_J||A_T)$
(2) $T \rightarrow M \rightarrow J$: $N_1||N_2||A_J||\text{Sign}_T(N_1||N_2||A_J)$
(3.a) $J \rightarrow M$: $e_{P_T}(U||N_2)||N_3||\text{Sign}_J(e_{P_T}(U||N_2)||N_3)$
(3.b) $M \rightarrow T$: $e_{P_T}(U||N_2)||N_3||\text{Sign}_J(e_{P_T}(U||N_2)||N_3)||e_{P_T}(K_D||ID_M||h(P_J))$
(4) $T \rightarrow$ Location Server: $ID_M||A_T$
(5) Location Server $\rightarrow$ Rule Holder: $ID_M||A_T$
(6) Rule Holder $\rightarrow$ Location Server: Result$||ID_M$
(7) Location Server $\rightarrow$ Location Generator: ID_M
(8) Location Generator $\rightarrow$ Location Server $\rightarrow T$:
Location Object$||ID_M$
(9) $T \rightarrow M \rightarrow J$: $e_{P_J}(K_D)||N_3||\text{Sign}_T(e_{P_J}(K_D)||N_3)$

Figure 10.3: Adding a Device to a Domain Using a Proxy

(9) $T \rightarrow J$: $e_{P_J}(K_D)||N_3||\text{Sign}_T(e_{P_J}(K_D)||N_3)$.

When J receives this message, it verifies T's signature and checks message freshness by comparing N_3 with the value sent in (3). J then decrypts $e_{P_J}(K_D)$. Finally, the key K_D is securely stored by J, as described in section 6.5.2.

10.4.2.2 Adding a Device to a Domain Using a Proxy

This procedure is followed when a consumer wishes to add a device J to a domain using an existing domain device M as a proxy (Figure 10.2 summarises the main steps involved in adding a device to a domain). As above, M must have a unique identifier ID_M that is assigned by its service provider. J and T mutually authenticate each other via M, as described in steps 1–3 of section 10.4.2.1 (see also Figure 10.3); however, ID_J is not included at step (1) as J does not need to be capable of determining its position or having its location determined. In addition, in step (3), before M forwards the request to T, M checks that J is in close proximity, e.g. by using the Near Field Communication (NFC) protocol or measuring the Round-Trip Time (RTT) between M and J, see, for example, [36, 42, 49]. M creates the string $e_{P_T}(K_D||ID_M||h(P_J))$,

associates it with J's request, and then forwards it to T, as follows:

(3.a) $J \rightarrow M$: $e_{P_T}(U||N_2)||N_3||\text{Sign}_J(e_{P_T}(U||N_2)||N_3)$

(3.b) $M \rightarrow T$: $e_{P_T}(U||N_2)||N_3||\text{Sign}_J(e_{P_T}(U||N_2)||N_3)||e_{P_T}(K_D||ID_M||h(P_J))$

Next, T decrypts $e_{P_T}(U||N_2)$ and $e_{P_T}(K_D||ID_M||h(P_J))$. T verifies the signature, that U identifies a valid domain, and that K_D is the valid domain key. Because K_D is securely stored inside T and in devices that are already members of U's domain, it is not available in the clear, and can only be revealed by the trusted DRM software agent; this ensures that the join request comes via an existing domain device that is trusted to check the physical proximity of J. T then computes $h(P_J)$ from the value of P_J sent in (1), and compares it with the value recovered from $e_{P_T}(K_D||ID_M||h(P_J))$, ensuring that this string is unique per joining device. If this process succeeds, T follows the procedure described in section 10.4.2.1, steps 4-9 (where T communicates with J via M).

10.4.3 Removing a Device from a Domain

This phase is exactly as described in section 6.5.3.

10.4.4 Exchanging Content

This phase is exactly as discussed in section 7.3.4.

10.4.5 Backup and Recovery Procedure

The backup and recovery phase, as described in section 6.5.5, is implemented as follows. Domain credential backup can be performed by the TA, which recovers the domain key by re-joining devices that require recovery as described in section 10.4.2 (before the TA increments the values of the domain counter it checks whether the joining device public key is already a member of the domain; if so it does not increment the counters).

10.5 Controlling Domain Membership

The TA controls domain membership in the same way as described in section 6.6.2. Thus it imposes stringent restrictions on piracy using digital media such as the Internet, because, as described in section 10.4.1, the content protection key K_D is securely stored inside the TA, is not available in the clear, and can only be transferred from the TA to other devices after their physical location has been checked. That is, the physical location check, in conjunction with the use of counters, addresses the Root Distribution problem.

The scheme addresses illicit content proliferation; the only way in which a domain owner could transfer the content protection key to another user's device is by ensuring the new device is geographically close enough to the domain owner registered addresses. Whilst possible in principle, such a procedure is unlikely to be attractive to the domain owner. Such a process would also mean that the other user's device would become part of the domain, which would mean that fewer of the domain owner's devices could be added to the domain. Most importantly, devices which have joined this domain are not able to re-transfer

the domain key (as described above, only the TA can transfer this key to other devices).

10.6 Security Analysis

In location based services a mobile device location and a predefined privacy rules are the two important factors that control releasing a mobile device location to third parties. In addition, such privacy rules control the precision of released location measurements. Note that in this thesis we do not consider the security threats arising to communications between Geopriv components, i.e. the TA, Location Server, Rule Holder and Location Generator, as they are beyond the scope of this thesis. Security threats of this type are addressed elsewhere; see, for example, [24]. In addition, we do not address threats arising to communications between the Location Generator and the Mobile Device, as these are implementation-dependent; see, for example, [4]. Following are examples of such threats: interruption of location sensing service, altering the measured location and/or privacy rules, stealing location-aware device, masquerading as an intended recipient of data, eavesdropping user traffic, unauthorised access to data stored by system entities, and denial of service attacks.

We now consider the security threats, services, and mechanisms that apply to the storage, execution and transmission of K_D, U, and digital content. The security threats, services, and mechanisms that apply to the transmission of content and rights objects between domain devices, and the security threats, services, and mechanisms that apply to the storage and execution of content and rights objects in domain devices are covered in section 7.5 and 7.6.

10.6.1 Security Threats

The main goal of the proposed scheme is to prevent transferring the means for accessing content (i.e. the domain-specific key K_D) to unauthorised devices using physical or digital means. This is achieved by ensuring that this key is transferred to a device only after the domain owner has been authenticated, the values of both domain counters have been incremented, and the trustworthiness of the device has been verified. Hence, only devices which are part of a domain can access the domain content. Devices can only be added to a domain if the domain owner is personally involved, as ensured by authenticating the domain owner. This system has the following requirements.

- The TA T must securely sends the key K_D and U to a device J whilst joining the domain. Such a procedure is exposed to the following security threats: (1) unauthorised reading and alteration of K_D/U while in transit; (2) T unwittingly sending K_D to a malicious entity; (3) J unwittingly receiving K_D from a malicious TA T, or J unwittingly sending U to a malicious T; and (4) replay of communication between T and J.

- The domain specific-key K_D should be securely protected inside an existing domain device V. This results in the following vulnerabilities: (5) unauthorised manipulation of the domain-specific key during use in V; and (6) unauthorised manipulation of the domain-specific key whilst stored in V.

In addition to the above threats, there is a special type of threat associated with dynamic rights objects, i.e. backing up a dynamic rights object and then

restoring it at a later time to reuse an expired licence. As we discussed in section 6.5.4, there is no solution to this threat in authorised domains. In this thesis we do not address this problem, and it remains a possible topic for future research.

10.6.2 Security Services and Mechanisms

The security services required to counteract threats 2, 5 and 6 (listed above) can be provided using trusted platform functionality, as discussed in section 6.2. In section 10.7 we describe how such trusted platform functionality can be implemented using a platform conforming to the TCG specifications. Threats 1, 3 and 4, are addressed using standard cryptographic mechanisms. A direct mapping exists between the threats outlined above and the services and potential mechanisms outlined below:

1. *Confidentiality and integrity of K_D/U while in transit.* This service is provided by the use of asymmetric encryption and a digital signature — see section 10.4.2.

2. *Entity authentication of a joining device J to a TA T.* The provision of this service is implementation-dependent, and involves a protocol exchange between J and T; see, for example, [91, 92, 93, 94]). It is initiated when T and J mutually authenticate each other — see section 10.4.2. This mutual authentication attests to the DRM agent execution status, i.e. S_J, and whether the platform is trusted, as discussed in section 6.2.

3. *K_D origin authentication, and U destination verification.* The joining device J checks the origin of K_D by checking the TA's signature on the received encrypted value of K_D — see section 10.4.2. In addition, encrypt-

ing U using the TA's public key ensures that only the TA can extract U — see section 10.4.2.

4. *Prevention of replay of communications between T and J.* This is provided by the inclusion of nonces in protocol messages — see section 10.4.2.

5. *Confidentiality and integrity of the domain-specific values K_D/U during execution in a domain device.* Providing this service requires process isolation techniques, which provide protection for a message against being read or altered by an unauthorised entity whilst being executed [34]. This service can be provided using trusted computing technology, discussed in chapter 3.

6. *Confidentiality and integrity of the domain-specific values whilst stored in a domain device V.* Providing this service requires protected storage, typically by storing domain credentials inside a tamper-resistant module, as discussed in section 6.2.

10.7 Implementing the Protocols Using Trusted Computing

We described in section 6.3 how a system conforming to the TCG specifications [92, 93, 94] can satisfy the requirements described in section 6.2. In this section we consider how security requirements 1, 2 and 4, discussed in section 10.6, can be met using TCG functionality. In this section we do not discuss practical problems associated with TCG specification, as these have been discussed in section 3.8.

2. *Entity authentication of a device J to the TA T* is provided exactly as

discussed in point 4, section 7.6.

5. *Confidentiality and integrity of K_D and U during execution* is provided exactly as discussed in point 1, section 7.6.

6. *Confidentiality and integrity of K_D and U whilst stored in domain devices* is provided exactly as discussed in point 2, section 7.6.

10.8 Alternative Implementation

In this section we describe how our scheme can be implemented using the 3GPP LBS. The benefits of using 3GPP LBS include the following: 3G mobile networks have global coverage; the 3G mobile network infrastructure is secure and satisfies all the Geopriv requirements described in [24]; 3G mobile handsets are ubiquitous; and the existing mobile network infrastructure enables the authentication of subscribers. In addition, 3G mobile networks can be used to authenticate the domain owner to a Network Application Function as part of the 3GPP General Authentication Architecture (GAA) mechanism [6, 7]. The Network Application Function needs to provide the services provided by the TA, as described in this thesis. The Network Application Function could be hosted either as part of the mobile network operator, or by a third party trusted by the mobile network operator.

Assuming the Network Application Function service is provided by an external authority trusted by the mobile network operator, the Network Application Function sends a message to the 3GPP GMLC (Global Mobile Location Center), i.e. the Location Server in the proposed scheme, requesting the current location of the Mobile Device. The GMLC verifies the Network Application Function is

registered for this service and then sends a request to the PPR (Privacy Profile Register), i.e. the Rule Holder in the proposed scheme, to check if the Network Application Function is authorised to query the Mobile Device location. The PPR performs the privacy check based on the target Mobile Device's privacy profile, which contains privacy rules; for example, the Mobile Device might only allow its location information to be given to the Network Application Function when it is located in certain areas. The result of the privacy check is sent to GMLC. If the Network Application Function is authorised to query the Mobile Device location, the GMLC sends a "Provide Subscriber Location" message to the location sensing components in the 3GPP infrastructure, i.e. the Location Generator, which calculates the Mobile Device position and then returns the result to the GMLC. The GMLC forwards the result back to the Network Application Function.

Alternatively, if the Network Application Function service is provided by the mobile network operator, the above procedure can again be used with the exception that the Network Application Function can send the "Provide Subscriber Location" request directly to the location-sensing components in the 3GPP infrastructure, i.e. it can bypass the communication with the GMLC. For detailed information regarding 3GPP LBS, see, for example, [5].

10.9 Related Work

A variety of schemes have been proposed that involve authenticating a user based on the user's geographical location or physical proximity to an object; see, for example, [15, 25, 71]. However, these schemes apply to a somewhat different context to that of the scheme discussed above.

The system described in [15] requires each user to possess a physical token, where a user is authenticated to a device if the token is in physical proximity to the device. In addition, it requires each user to possess a smart card, and each device to posses a smart card reader. The smart card is used as an alternative method for user authentication, if location information cannot be established. The scheme described in [71] uses location based services to ensure that data can only be accessed in a predefined location. This system require each device to have an attached dongle, which is capable of calculating its geographical location by communicating with a base station. The dongle regularly monitors its location by ensuring that it is located close to a base station. If so, it authorises the device attached to it to decrypt data.

These systems differ from the scheme proposed in this thesis in the following ways.

- The scheme described in [15] does not measure the device's geographical location, but it requires the device to check that it is in physical proximity to a physical token owned by the user. However, the scheme described in this chapter requires a device to be at a predefined location, and does not require users to possess tokens.
- The scheme described in [71] requires each device to be capable of calculating its position, which increases the overall cost. However, the scheme proposed in this chapter requires only one device per domain to be capable of determining its location.
- The scheme described in [71] solves problems relating to the theft of 'protected' data outside a certain location, e.g. stealing notebooks or hard drives. It allows data to be decrypted only at a predefined location; once

the data is decrypted the scheme does not prevent data from being transferred elsewhere using digital or physical means. However, the scheme in this chapter solves problems associated with uncontrolled transferring of protected data to others.

- The scheme described in [71] involves authenticating users (by measuring their geographical location) for every transaction. In addition, if the dongle cannot calculate its location, e.g. due to weak reception of GPS signals, or hardware problems in the dongle itself or the base station, authorised individuals would not be capable of accessing their data. The scheme described in [15] involves authenticating users (measuring physical proximity) every time a user logs in to a device. In addition, if a device cannot be certain whether it is in physical proximity to the token possessed by the device owner, the device can authenticate the user using a different authentication technique. This undermines location based authentication.

  The scheme described in this chapter only involves authenticating a user (by measuring the user's geographical location) when a device is added to a domain, and does not require user authentication for downloading and exchanging content between devices.

10.10 Summary

This chapter contains a description of an implementation of the authorised domain DRM framework given in chapter 6. This scheme uses location-based services to ensure that devices joining a consumer domain are located in physical proximity to the registered addresses for this domain. This restricts domain membership to devices in predefined geographical locations, helping to ensure

that a single consumer owns and manages each domain. In addition, the scheme helps to protect consumer privacy; the location-based service is only required for managing domain membership, and not for creating, downloading or exchanging content. The scheme uses the Geopriv protocol, which enables domain owners to control the precision or resolution of location data, and when, in which location, and by whom their location can be retrieved.

Chapter 11

Assessment and Analysis

Contents

In this chapter we analyse the general DRM framework presented in chapter 6, and the four different schemes for authorised domain management presented in

chapters 7–10. The framework and the schemes are evaluated using the ideal list of DRM requirements given in chapter 4. We then discuss deployment issues for the proposed schemes, and compare the pros and cons of the four schemes. The chapter concludes with a comparison of the proposed schemes with other schemes.

11.1 Introduction

This chapter contains an evaluation of the generic framework and the four proposed implementations using the ideal list of DRM requirements given in chapter 4. We then discuss possible deployment issues for these schemes, and compare the pros and cons of the four implementations of the general framework. Finally, we compare the proposed schemes with other authorised domain management schemes, which are discussed in chapter 5.

11.2 General Framework

We analyse the general framework described in chapter 6 using the list of requirements given in section 4.5.

1. *Minimum cost.* The framework requires participating domain devices to have Trusted Platform properties. A TCG[1] compliant platform has these properties, as discussed in section 6.3. TCG compliant platforms are not expensive, and are currently available from a range of PC manufacturers, including Dell, Fujitsu, HP, Intel and Toshiba [35]. Many challenges still

[1] www.trustedcomputinggroup.org

remain to be solved in the TCG specifications in order for them to be practical for today's devices, applications and operating systems.

2. *Ease of use.* The framework has five main stages. The first, second, third and fifth phases involve initialising the consumer domain, adding devices to a domain, removing devices from a domain, and backing up and recovering domain credentials. The details of these phases are implementation-dependent. The fourth stage involves exchanging content between devices. The framework does not specify how this is achieved.

3. *Performance.* The performance of the domain management functions of a scheme compliant with the framework (i.e. covering domain establishment, and adding devices to and removing devices from a domain) is implementation dependent. The framework requires content to be encrypted using a symmetric technique, and also for the secret content encryption key to be encrypted using a symmetric algorithm. Symmetric encryption techniques are simple and fast to compute by comparison with asymmetric techniques. Also, the domain-specific key does not change during the life of the domain, which avoids the need to track each content-specific rights object in order to re-encrypt it. Many DRM schemes fail to address this requirement, as discussed in chapter 5.

4. *Content mobility.* The domain-specific key, which is used to protect domain content, is shared amongst all domain devices. Hence all domain content can run in all domain devices, as discussed in section 6.6.1.

5. *Lack of dependence on network infrastructure.* This is implementation dependent.

6. *Lack of dependence on a secure clock.* This is implementation dependent.

7. *Cryptographic robustness.* Neither the framework nor the four instantiations of the framework require the use of specific cryptographic techniques (or any techniques with unusual properties). 'Standard' methods can thus be employed, i.e. there is no bar to using the most robust currently available technology.

8. *Ease of recovery.* The backup and recovery method for network keys and content is implementation dependent. The framework requires each device to have its own private and public key pair, protected using a tamper-resistant module. If one system is hacked, and as a result its private key is revealed, then this does not affect other domains because domain keys are independently selected. However, such an attack would reveal the domain-specific key, as used to encrypt content encryption keys, which might enable abuse of the content in that domain.

9. *Robust content protection.* Piracy is restricted by controlling domain membership, as described in section 6.6.2.

10. *Flexible rights structure.* The framework involves content providers controlling digital asset consumption using rights objects, i.e. files associated with content.

11. *Simple key management.* This is implementation dependent.

12. *Flexible Revocation Mechanism.* This is implementation dependent.

13. *Access and Usage of Content.* This point is not fully addressed; in particular, fair use is not addressed explicitly. However, the framework is designed to allow users to access content on all their devices in a simple and transparent way; as a result it would appear likely to cover most possible interpretations of the notion of 'fair use'.

14. *Consumer Privacy.* In the proposed scheme, the creation and ongoing management of an authorised domain is controlled by a domain controller, which is trusted by rights issuers and content distributors. The nature of the domain controller is implementation-dependent. Also, collaboration between domain controllers and content distributors might enable them to discover the public keys of devices in a domain. However, such collaboration would not necessarily allow content distributors to learn other information about the domain, such as the precise content present in a domain. This issue is implementation-dependent, as it depends on the nature of the trusted domain controller.

15. *Interoperability.* The framework does not require use of a particular device type. Content can run on any device in a domain, as long as it satisfies the trusted platform properties described in section 6.2. This functionality is available in many new PCs and some wireless devices; however, it is not yet provided in other types of device, e.g. TVs, radios, and CD Players. TCG technology is emerging into the market; it will probably take some time until it is available in all device types. Many challenges still remain to be solved in the TCG specifications.

16. *Security and Hardware Issues.* The framework does not limit the use of other protection software.

11.3 Authorised Domain Management Using a Master Control Device

We analyse the scheme described in chapter 7 using the list of requirements given in section 4.5.

1. *Minimum cost.* This is exactly as described in section 11.2, point (1).

2. *Ease of use.* The system has five main stages. The first involves initialising the master control device by generating a domain unique key that is then bound to the domain owner's authentication credential and the two domain limits. This process should be simple for a user to initiate, and is carried out once in the domain's lifetime. The second and third involve adding a device to a domain and removing a device from a domain. These processes are straightforward and are carried out once when a device joins or leaves a domain. The fourth stage covers the exchange of content, which is exactly as described in section 11.2, point (2). The last stage provides for the backup and recovery of domain credentials. Domain credentials are backed up transparently to the end user, and the recovery procedure is a straightforward process.

3. *Performance.* Properly assessing the performance of the domain management process requires prototyping, which is planned future work. However, use of the domain management procedure (which covers initialising a domain, and adding and removing devices from a domain) is likely to occur relatively infrequently, and so the performance impact on the system is minimal. Performance issues associated with protecting and exchanging content are discussed in section 11.2, point (3).

4. *Content mobility.* This is exactly as described in section 11.2, point (4).

5. *Lack of dependence on network infrastructure.* As outlined in section 7.3.4, content can be transferred between domain devices either directly, if the devices have a direct connection, or using a portable storage device, for example, a USB memory stick.

6. *Lack of dependence on a secure clock.* The scheme does not use timestamps;

nonces are used for establishing message freshness.

7. *Cryptographic robustness.* This is exactly as described in section 11.2, point (7).

8. *Ease of recovery.* A backup and recovery procedure for network keys and content is proposed in section 7.3.5. As discussed in section 11.2 point (8), attacking a device might compromise the domain-specific key; this could enable misuse of content belonging to that domain, but will have no effect on the operation of other domains.

9. *Robust content protection.* This is exactly as described in section 7.4, and 11.2 point (9).

10. *Flexible rights structure.* This is exactly as described in section 11.2, point (10).

11. *Simple key management.* The key management scheme operates transparently to end users.

12. *Flexible Revocation Mechanism.* Revoked keys are stored in a revocation list that can be queried, for example, via an OCSP service. There is no need for all devices to be connected at the same time in order to revoke keys.

13. *Access and Usage of Content.* This is exactly as described in section 11.2, point (13).

14. *Consumer Privacy.* The scheme protects consumer privacy; content distributors/rights issuers do not need to monitor user content consumption; instead, a domain-specific master control device, certified by a Certification Authority trusted by content distributors and rights issuers, is used

to control domain membership. In addition, to prevent content distributors and rights issuers from identifying individuals and monitoring domain owners content usage, a privacy Certification Authority (privacy-CA) or Direct Anonymous Attestation technique could be implemented. Moreover, the master control device is managed by the domain owner, which prevents content distributors/rights issuers from collaborating with it to learn what content is present in a domain, or to identify the devices in a domain.

15. *Interoperability.* This is exactly as described in section 11.2, point (15).

16. *Security and Hardware Issues.* This is exactly as described in section 11.2, point (16).

11.4 Authorised Domain Management Using an Electronic Payment System

We now analyse the Authorised Domain Management Using an Electronic Payment System scheme described in chapter 8, using the list of requirements given in section 4.5.

1. *Minimum cost.* This scheme requires the implementation of a TA to control and manage consumer domains. This increases the overall cost of implementing the system; however, the TA could be established by an existing trusted third party, e.g. a bank, which should reduce the overall cost. In addition, the participating banks and the payment network provider, which have agreed that payment cards and the authorisation network can be used to support this scheme, might wish to charge for

this service. However, the participating banks and the payment network provider are only involved in domain creation and in adding devices to a domain, and hence the cost impact of a modest charge should be manageable. Also, as discussed in section 11.2 point (1), the scheme requires participating domain devices to have Trusted Platform properties. The (modest) additional costs of implementing such a solution in every device could be offset by the content owners, e.g. through lower licence costs, since the content owners will benefit from reduced piracy.

2. *Ease of use.* The system has five main stages. The first involves contacting a TA, which initialises the domain by generating a domain unique key and a domain identifier; these are then bound to the domain owner's payment card details and the two domain limits. This process should be simple for a user to initiate, and is carried out once in the domain's lifetime. The second involves adding a device to a domain. This involves contacting the TA, which verifies that the domain owner payment card details match those associated with the domain, and ensures that the joining device is in physical proximity to a device already in this domain. This process should be simple for a user to initiate, and is carried out once when a device joins a domain. The third involves removing a device from a domain, which is similar to the process of adding a device to a domain, without requiring the measurement of physical proximity. The fourth stage covers the exchange of content, which is exactly as described in section 11.2, point (2). The last stage provides for the backup and recovery of domain credentials. This is carried out by the TA, and is transparent to the end user.

3. *Performance.* This is exactly as described in section 11.3, point (3).

4. *Content mobility.* This is exactly as described in section 11.2, point (4).

5. *Lack of dependence on network infrastructure.* This is exactly as described in section 11.3, point (5).

6. *Lack of dependence on a secure clock.* This is exactly as described in section 11.3, point (6).

7. *Cryptographic robustness.* This is exactly as described in section 11.2, point (7).

8. *Ease of recovery.* A backup and recovery procedure for network keys and content is proposed in section 8.3.5. As discussed in section 11.2 point (8), attacking a device might compromise the domain-specific key; this could enable misuse of content belonging to that domain, but will have no effect on the operation of other domains. In this particular scheme, the domain unique identifier could also be compromised; however, this would not enable unauthorised devices to be added to a domain unless domain owner payment card details are also available to the attacker.

9. *Robust content protection.* This is exactly as described in section 8.4, and 11.2 point(9).

10. *Flexible rights structure.* This is exactly as described in section 11.2, point (10).

11. *Simple key management.* This is exactly as described in section 11.3, point (11).

12. *Flexible Revocation Mechanism.* This is exactly as described in section 11.3, point (12).

13. *Access and Usage of Content.* This is exactly as described in section 11.2, point (13).

14. *Consumer Privacy.* This scheme protects consumer privacy; payment card details are not available to the TA, thereby protecting consumer privacy. In addition, to prevent the TA, content distributor, and rights issuer from identifying individuals and monitoring owner content usage patterns, a privacy-CA or Direct Anonymous Attestation technique could be implemented. A TA and the participating banks might collaborate with a content distributor/rights issuer to reveal the domain owner payment card details, which are used to manage the domain. This would allow the content distributor/rights issuer to identify domains and monitor content usage patterns. However, such a situation, which seems unlikely, can be avoided by using two payment cards, one for managing the domain and the other for buying content.

15. *Interoperability.* This is exactly as described in section 11.2, point (15).

16. *Security and Hardware Issues.* This is exactly as described in section 11.2, point (16).

11.5 Authorised Domain Management Using a Mobile Phone

We now analyse the Authorised Domain Management Using a Mobile Phone scheme described in chapter 9, using the list of requirements given in section 4.5.

1. *Minimum cost.* This scheme requires a mobile phone network operator to provide the network application function service, part of the GAA. A participating network operator might wish to charge for providing such a

service; however, the service is only required when creating a domain and when devices join a domain, and hence the cost impact of a modest charge should be manageable. Also, as discussed in section 11.2 point (1), the scheme requires participating domain devices to have Trusted Platform properties. The extra implementation costs could be covered from the expected reduction in piracy.

2. *Ease of use.* The system has five main stages. The first involves initialising a domain-specific mobile phone; this phone communicates with a network application function service, which initialises the domain and sets the domain limits. This process should be simple for a user to initiate, and is carried out once in the lifetime of a domain. The second and third involve adding a device to a domain and removing a device from a domain, during which the domain owner is authenticated to the domain-specific mobile phone, and the mobile phone and the network application function service are mutually authenticated. This process is straightforward, and is carried out once when a device joins a domain or leaves a domain. The fourth stage covers the exchange of content, which is exactly as described in section 11.2, point (2). The last stage provides for the backup and recovery of domain credentials. This is carried out by the network application function, and is transparent to the end user.

3. *Performance.* This is exactly as described in section 11.3, point (3).

4. *Content mobility.* This is exactly as described in section 11.2, point (4).

5. *Lack of dependence on network infrastructure.* This is exactly as described in section 11.3, point (5).

6. *Lack of dependence on a secure clock.* This is exactly as described in section 11.3, point (6).

7. *Cryptographic robustness.* This is exactly as described in section 11.2, point (7).

8. *Ease of recovery.* A backup and recovery procedure for network keys and content is proposed in section 9.4.6. As discussed in section 11.3 point (8), attacking a device might compromise the domain-specific key; this could enable misuse of content belonging to that domain, but will have no effect on the operation of other domains.

   If a domain device is hacked it will not compromise content in other domains, as described in section 11.3, point (8).

9. *Robust content protection.* This is exactly as described in section 9.5, and 11.2 point(9).

10. *Flexible rights structure.* This is exactly as described in section 11.2, point (10).

11. *Simple key management.* This is exactly as described in section 11.3, point (11).

12. *Flexible Revocation Mechanism.* This is exactly as described in section 11.3, point (12).

13. *Access and Usage of Content.* This is exactly as described in section 11.2, point (13).

14. *Consumer Privacy.* This scheme protects consumer privacy; content distributors and rights issuers do not need to monitor domain content consumption or profile users, as each domain is managed by a trusted domain-specific mobile phone. To prevent content distributors and rights issuers from identifying domain owners and monitoring their content usage patterns, a privacy-CA or Direct Anonymous Attestation technique could be

implemented. Moreover, if a content distributor/rights issuer collaborates with the mobile phone provider, they will not be able to identify domain content except if the consumer uses his mobile phone to buy content. In this case content providers, by collaborating with the mobile phone provider, could identify a domain using the domain-specific mobile phone IMSI. This would enable content distributors to identify content downloaded using the domain-specific mobile phone. However, such a case can be avoided by using a different domain device (which could be another mobile phone) for downloading content.

15. *Interoperability.* This is exactly as described in section 11.2, point (15).

16. *Security and Hardware Issues.* This is exactly as described in section 11.2, point (16).

11.6 Authorised Domain Management Using Location Based Services

We now analyse the Authorised Domain Management Using Location Based Services scheme described in chapter 10, using the list of requirements given in section 4.5.

1. *Minimum cost.* This scheme requires the establishment of a TA to manage consumer domains. This authority could be a mobile phone network operator, which should reduce the overall costs. Also, as discussed in section 11.2 point (1), the scheme requires participating domain devices to have Trusted Platform properties. The (modest) additional costs of implementing such a solution in every device could be offset by the content owners,

e.g. through lower licence costs, since the content owners will benefit from reduced piracy.

2. *Ease of use.* The system has five main stages. The first involves contacting a TA, which initialises a domain by generating a domain unique key and domain identifier; These are bound to the domain owner's geographical addresses and the two domain limits. This process should be simple for a user to initiate, and is carried out once in the domain's lifetime. The second involves adding a device to a domain. This involves contacting the TA, which verifies that the joining device is located in physical proximity to the domain registered addresses. This process is straightforward and is carried out just once. The third involves removing a device from a domain, in which the domain owner is authenticated to the TA. This process is straightforward and is carried out once when a device leaves a domain. The fourth stage covers the exchange of content, which is exactly as described in section 11.2, point (2). The last stage provides for the backup and recovery of domain credentials. This is carried out by the TA, and is transparent to the end user.

3. *Performance.* This is exactly as described in section 11.3, point (3).

4. *Content mobility.* This is exactly as described in section 11.2, point (4).

5. *Lack of dependence on network infrastructure.* This is exactly as described in section 11.3, point (5).

6. *Lack of dependence on a secure clock.* This is exactly as described in section 11.3, point (6).

7. *Cryptographic robustness.* This is exactly as described in section 11.2, point (7).

8. *Ease of recovery.* A backup and recovery procedure for network keys and content is proposed in section 10.4.5. As discussed in section 11.2 point (8), attacking a device might compromise the domain-specific key; this could enable misuse of content belonging to that domain, but will have no effect on the operation of other domains.

9. *Robust content protection.* This is exactly as described in section 10.5, and 11.2 point(9).

10. *Flexible rights structure.* This is exactly as described in section 11.2, point (10).

11. *Simple key management.* This is exactly as described in section 11.3, point (11).

12. *Flexible Revocation Mechanism.* This is exactly as described in section 11.3, point (12).

13. *Access and Usage of Content.* This is exactly as described in section 11.2, point (13).

14. *Consumer Privacy.* Each domain owner can define privacy and filtering rules that specify which entity can query its location, and to what level of detail. A TA will learn the location and identities of the domain owner, which might raise a privacy concern; however, a TA service is only required to change domain membership, and is not capable of monitoring content consumption. This protects consumer privacy. To prevent the content distributor and rights issuer from identifying domain owners and monitoring their content usage patterns, a privacy-CA or Direct Anonymous Attestation technique could be implemented. If a content distributor/rights issuer collaborates with the TA, then they will not be able to identify domain

content except if the consumer buys content using a mobile device from a domain registered address. This scheme uses the Geopriv protocol, which enables a domain owner to define privacy rules identifying the entities that can query a device location, and also to specify the precision of the retrieved location. In addition, this case can be avoided by using a specific mobile device to act as a proxy with the TA for domain management. In this case the TA can only query the location of this particular device, and hence collaboration between the TA and the content distributor will not enable them to learn what domain content is downloaded via other domain devices.

15. *Interoperability.* This is exactly as described in section 11.2, point (15).

16. *Security and Hardware Issues.* This is exactly as described in section 11.2, point (16).

11.7 Deployment Issues for the Four Schemes

In this section we consider the main challenges that raise when deploying the four proposed schemes.

All the proposed schemes require devices to be TPs with certain properties in order to protect the means for accessing domain content. As discussed in section 6, the TCG specifications satisfy all required properties. We also discussed the main practical challenges to the implementation of TCG compliant platforms. These challenges are the subject of ongoing research, and seems likely that it take some time to address all these challenges. Hence we have discussed the possibilities for relaxing the TP security assumptions, and use existing techniques

for protecting the means for accessing content. There are always trade-offs between security and cost, and security and usability. Therefore, strengthening or relaxing the security requirements needs to be carefully considered based on the context of the intended application and the environment in which the proposed authorised domain schemes will be integrated. For example, if the proposed authorised domain management schemes are used in an enterprise environment, then the TP security requirements should be seriously considered, as the sensitivity of content in enterprises is likely to be a major concern. Also, enterprises are typically willing to spend more money on security assurance than owners of personal networks. In addition, opposed to the devices used in a personal network, enterprise devices are typically PCs and servers that already have a TPM chip integrated in their motherboard.

All the proposed schemes require a trusted domain controller. The nature of the domain controller and how it can be realised in practical life is DRM scheme-specific. In the Authorised Domain Management using a Master Control Device schemem, the domain controller is a master control device that does not need to be a dedicated device; it could be a laptop that has a specific software application to implement the master control device functionality. Thus, this scheme is the easiest to implement and imposes the minimum cost. Moreover, the master control device is managed by the domain owner, which prevents content distributors/rights issuers from collaborating with the master control device to learn content usage information. In all other schemes the domain controller is a trusted authority of a specific nature. We now discuss deployment issues for the other three schemes.

In the Authorised Domain Management Using an Electronic Payment System scheme, the domain controller is a TA, which could be established by a bank to

reduce the overall cost. Banks might be reluctant to use their infrastructure for DRM applications in personal network, as their existing infrastructure is mainly for facilitating electronic payment to collect revenue. One might think that this point is enough to make banks reluctant in considering the proposed scheme. However, in the proposed scheme the verification of a domain owner's payment card is performed in a similar way to the electronic payment process. This should help ease implementation issues, and increase the acceptability of the scheme. Also, and most importantly, the payment infrastructure is mainly used whenever a device is added/removed from a domain, and not for buying content. Hence, the performance impact is minimal. In addition, the participating banks and the payment network provider, which have agreed that payment cards and the authorisation network can be used to support this scheme, might charge the consumers for this service. However, the participating banks and the payment network provider are only involved in domain creation and in adding devices to a domain, and hence the cost impact of a modest charge should be manageable. The (modest) additional costs of implementing this scheme could be offset by the content owners, e.g. through lower licence costs, since the content owners will benefit from reduced piracy. Moreover, a TA and the participating banks might collaborate with a content distributor/rights issuer to reveal the domain owner payment card details, which are used to manage the domain. This would allow the content distributor/rights issuer to identify domains and monitor content usage patterns. However, such a situation, which seems unlikely, can be avoided by using two payment cards, one for managing the domain and the other for buying content.

In the Authorised Domain Management using a Mobile phone scheme, the domain controller is a combination of the domain-specific mobile phone and the

network application function, which is provided by the mobile network operator as part of the GAA. Mobile network operators have an established infrastructure for authenticating their subscribers. This should help ease implementation. However, mobile network operators might be worried for using their infrastructure to support such a service, as it might have an effect on the network performance and, in addition, on the maintenance cost; however, their service is only required when creating a domain and when devices join a domain, and hence the performance impact is minimal. Also, mobile network operators will be interested if they will gain profit, for example, today's mobile networks are not only used to make phone calls, it is also used for other purposes such as buying content. Participating network operators' charge for providing such a service is expected to be minimal, as the service is only required when creating a domain and when devices join a domain, and hence the cost impact of a modest charge should be manageable. In this scheme, if a content distributor/rights issuer collaborates with the mobile phone provider, they will not be able to identify domain content except if the consumer uses his mobile phone to buy content. In this case content providers, by collaborating with the mobile phone provider, could identify a domain using the domain-specific mobile phone IMSI. This would enable content distributors to identify content downloaded using the domain-specific mobile phone. However, such a case can be avoided by using a different domain device (which could be another mobile phone) for downloading content.

In the authorised domain management using location-based services scheme, the domain controller is a TA, which could be a mobile network operator. The benefits of using a mobile network LBS include the following: Mobile networks have global coverage; the 3G mobile network infrastructure is secure; Mobile

handsets are ubiquitous; and the existing mobile network infrastructure enables the authentication of subscribers. The deployment analyses for the authorised domain management using a mobile phone equally applies for the deployment of this scheme. If a content distributor/rights issuer collaborates with the TA, then they will not be able to identify domain content except if the consumer buys content using a mobile device from a domain registered address. This scheme uses the Geopriv protocol, which enables a domain owner to define privacy rules identifying the entities that can query a device location, and also to specify the precision of the retrieved location. In addition, this case can be avoided by using a specific mobile device to act as a proxy with the TA for domain management. In this case the TA can only query the location of this particular device, and hence collaboration between the TA and the content distributor will not enable them to learn what domain content is downloaded via other domain devices.

11.8 Comparing the Four Schemes

The main differences between the four schemes are as follows.

- *Identifying Domain ownership.* In the Authorised Domain Management using a Master Control Device scheme, domain owners are authenticated using two-factor authentication, which involves "something the domain owner has", i.e. a master control device, and "something the domain owner is or knows", i.e. a biometric or password/PIN authentication mechanism that is implemented by the master control device. In the Authorised Domain Management using an Electronic Payment System scheme, domain owners are authenticated using their payment cards, building on the existing electronic payment system by ensuring that the name and the date

of birth of a domain creator are the same for all devices joining a domain. The Authorised Domain Management using a Mobile Phone scheme involves the use of a domain-specific mobile phone and the mobile phone network operator to authenticate the domain owner before devices can join a domain. The Authorised Domain Management using Location Based Services scheme ensures that devices being added to consumer domains are located in physical proximity to the domain registered addresses. This restricts domain membership to devices in predefined geographical locations, helping to ensure that a single consumer owns and manages each domain.

- *Implementation Cost.* The Authorised Domain Management using a Master Control Device scheme would appear likely to be the least expensive scheme to implement, as domain management is provided by a master control device, which does not need to be a dedicated device, i.e. it could be part of another domain device. The Authorised Domain Management using an Electronic Payment System scheme is likely to be the most expensive scheme to implement, as it requires a (small) modification to the existing electronic payment infrastructure (to encrypt card details so that they can only be decrypted by authorised banks). In addition, it requires the establishment of a TA to manage consumer domains; however, this service could be provided by an established bank, which should reduce the overall costs. The participating banks and the payment network provider, which have agreed that payment cards and the authorisation network can be used to support this scheme, might want to charge for the provision of this service; however, their services are only required when creating a domain, and when devices join the domain, and hence the cost impact of a modest charge should be manageable. The Authorised Domain Man-

agement using a Mobile Phone and the Authorised Domain Management using Location Based Services scheme do not require major changes to existing mobile network operator infrastructures. The participating network operators might wish to charge for their service; however, such a service is only required when creating a domain, and when devices join the domain, and hence the cost impact of a modest charge should be manageable. The extra costs in implementing the schemes could be recovered from the expected reduction in piracy.

- *Trusted Domain Controller.* The trusted domain controller in the Authorised Domain Management using a Master Control Device scheme is a domain-specific master control device. In the Authorised Domain Management using an Electronic Payment System scheme the trusted domain controller is a TA. In the Authorised Domain Management using a Mobile Phone scheme the trusted domain controller is a combination of the domain-specific mobile phone and the network application function, which is provided by the mobile network operator as part of the GAA. In the Authorised Domain Management using Location Based Services scheme the trusted domain controller is a TA that provides Location Based Services, which could be a mobile network operator.

- *Domains covering multiple locations.* This might be a requirement for organisations that have offices in multiple locations, or for mobile consumers. The Authorised Domain Management using a Master Control Device scheme requires the master control device to be a portable device to enable ease of movement between different locations. The Authorised Domain Management using an Electronic Payment System scheme can be adapted to allow domain expansion in multiple locations; however, it would require a domain device that is already a member of the domain to

be portable. This is required because adding devices into a domain should be performed using a device already in the domain acting as a proxy. The Authorised Domain Management using a Mobile Phone scheme can support domain expansion at multiple locations, because of the portability of mobile phones. The Authorised Domain Management using Location Based Services scheme is the best solution for supporting a domain spanning multiple widely separated locations. Also the latter scheme imposes more stringent restrictions on physical piracy than the other schemes, as it ensures all joining devices are located within the physical proximity of a domain registered address.

- *Convenience for personal networks.* All four schemes are designed for domains owned by a single owner; however, the Authorised Domain Management using a Master Control Device scheme and the Authorised Domain Management using a Mobile Phone scheme could be more practical for consumer domains. This is because the former imposes minimal additional costs, while the latter is not expensive to implement, and the use of a mobile phone gives consumers greater flexibility and convenience, since mobile phones are personalised, portable and ubiquitous.

- *Suitability for Enterprise DRM.* In addition to the above comments on domains covering multiple locations, the Authorised Domain Management using a Mobile phone scheme and the Authorised Domain Management using an Electronic Payment System scheme are not suitable for use by a large organisation. This is because both solutions bind domain ownership to personal devices (payment cards or mobile phones), and it would not be convenient for an organisation to use a personal mobile device or payment card to add devices to its domain. The Authorised Domain Management using Location Based Services scheme would appear to be the most ap-

propriate for corporate use; this is because organisations typically have multiple sites, and a corporate domain is, typically, managed by a group of system and security administrators, who might need to manage domain membership at each domain location. In addition, since corporate locations do not change frequently, this reduces TA management costs. The authorised domain management using a Master Control Device scheme requires slight modifications to its design in order to work effectively in a corporate setting. For example, the master control device needs to be physically bound to organisation premises, so that it cannot be taken elsewhere.

11.9 Comparing the Proposed Schemes with Other Schemes

In this section we compare the four proposed schemes with the schemes discussed in chapter 5.

11.9.1 OMA DRM

In this section we compare the four proposed implementations with the OMA scheme analysed in section 5.1.1. The main differences are as follows.

- In the OMA scheme adding a device into a domain requires the joining device to send a request to the rights issuer managing the domain. The request contains the domain identifier. This domain identifier can be shared with others. Hence leaf distribution arises. Our proposed schemes address this problem by strongly binding devices joining a domain to the

domain owner.

- In the OMA scheme each device needs to securely store the domain keys, domain identifiers and domain expiry time provided by the rights issuer for each domain that it joins. Extra secure storage is needed to store these keys, which potentially increases the cost and complexity of a device. Also, devices that have restricted storage can join a limited number of domains. The proposed schemes requires each device to securely stores a dynamic domain-specific key.

- In the OMA scheme each rights issuer is required to define domains, manage domain keys, and control which and how many devices are included in a domain. This in turn increases the overall complexity and maintenance costs imposed on rights issuers. Our proposed schemes the domain controller is a single entity, which manages consumers domains.

- Based on the OMA scheme domain devices must join all the relevant rights issuers to use all content in a domain. This is not user friendly, and makes administration complex from the user's perspective. Our schemes require devices to join a domain only once. In addition, rights issuers is not involved in domain management in our schemes.

- In the OMA scheme if a device in a domain is hacked, new domain keys are generated by all rights issuers from whom content has been obtained. These keys must be transferred to all registered domain devices (except, of course, for revoked devices). This means that the domain owner must update the keys on all devices for all rights issuers, which is not user friendly. In our schemes the domain-specific key does not change even if the device is hacked, as we use different policy to handle this issue, which is described early on in this chapter.

- In the OMA scheme all rights issuers store sensitive information about consumer domains, such as device public keys and digital content usage. This enables rights issuers to track the usage patterns of domain owners, which potentially raises major privacy concerns. Our schemes does not involve rights issuer in creating consumer domains. Our schemes is based on the assumptions that rights issuers and content distributors trust the domain manager to operate as expected. Thus, they no longer need to monitor content consumption.

11.9.2 eXtensible Content Protection (xCP)

In this section we compare the four proposed implementations with the xCP scheme analysed in section 5.1.2. The main differences are as follows.

- In the xCP scheme there is no binding between the content protection key and a domain owner. This means that any device can obtain the means for accessing content, as long as this device has not been revoked and the maximum number of devices in the domain has not exceeded a pre-specified limit. Hence leaf distribution arises. Our proposed schemes address this problem by strongly binding devices joining a domain to the domain owner.

- The xCP scheme is based on the broadcast encryption protocol, which requires a licensing agency to produce the MKB and assign device keys. The nature of the licensing agency and how it can be realised in real life is not discussed in xCP scheme. In addition, devices must be capable of processing the MKB to produce the domain key. The MKB is a large data structure, and hence moving it between devices and using it to generate

the media key imposes a significant overhead, especially on devices that have limited capabilities. The proposed schemes require a trusted domain controller. The nature of the domain controller and how it can be realised in practical life is discussed in each scheme. In addition, the proposed schemes only require devices to be TP with certain properties. As described in this chapter TCG compliant platforms can satisfy the required properties.

- The xCP scheme each device in a domain requires tamper-resistant storage to store the domain key and the set of keys assigned by licensed media manufacturers. The proposed schemes only require devices to store a single a symmetric key.

- The xCP scheme does not allow increasing the number of devices that can be in a domain, which makes the system less flexible for consumers. In our porposed schemes the number of devices that can be in a domain is controlled by a trusted authority. The authority decides the default number of devices that can be in a domain. This, for example, would depends on the country (different countries have different (average) number of members in a family). In addition, this number could be customised on individual bases with extra charges.

- In the xCP scheme every time the domain membership changes or a new MKB is released, the domain key must be changed. Consequently, the content encryption keys will need to be re-encrypted with the new domain key, which means that all content and associated encryption keys must be tracked. This could result in a significant overhead if the number of items of content is large. The proposed schemes do not change the domain specific key; leaving devices must communicate with the domain

controller that ensures a device removes the domain key before it updates the domain-specific counters. This stops the domain owners abusing the system by adding devices and then removing them to use domain content in unauthorised way.

- In the xCP scheme all devices must be online to obtain a new domain key if a device joins or leaves the domain, or if a device is hacked. It is potentially difficult for some devices to be online (e.g., a car CD Player). The proposed schemes does not require devices to be continuously connected.

11.9.3 Apple Fairplay

In this section we compare the four proposed implementations with the Apple Fairplay scheme analysed in section 5.1.4. The main differences are as follows.

- The Apple Fairplay scheme fails to bind devices in a domain to the domain owner. Adding a device to a domain simply requires the domain owner to be authenticated using an Apple-ID and password that can be shared with others. Hence leaf distribution raises. Our proposed schemes address this problem by strongly binding devices joining a domain to the domain owner.

- The domain key in Apple Fairplay scheme is protected using software techniques only. The software protection has been hacked many times. In addition, content can be backed up to a standard CD in unencrypted form, or transported elsewhere via email or FTP, enabling content proliferation. The proposed schemes protects a domain key using hardware measures. Also, we proposed secure ways for backing up and recovering of domain credentials.

- In the Apple Fairplay the domain size is restricted to five computers. Our proposed schemes the domain size is dynamic and can be changed based on consumers need, as discussed in section 11.9.2.

- In the Apple Fairplay key management is based on sharing a list of secret keys between all devices in a domain. Every time a consumer buys new content, a user key is generated to protect the added content. This key is also transferred to the Apple server so that it can be used by other devices in the same domain. This requires all devices to be online to retrieve the new key. In addition, if the number of pieces of content is large, this might cause a key management problem. Our proposed schemes require a domain specific-key to protect domain content. This key is transfered to a device only once when joining the domain, and it does not change when membership of a domain changes. Hence our schemes do not require devices to be continually available. Our proposed scheme use different measures for protecting domain content when a device joins or leaves a domain, as discussed earlier this chapter.

- The Apple server and iTMS could jointly track consumer content usage, and this raises a potentially major privacy issue. Our proposed schemes the content distributors and rights issuers do not get involved in domain creation and they do not require tracking content consumptions. However, three of the proposed schemes require a trusted authority to control domain membership (i.e. the ones that authenticate domain owners a payment card, a mobile phone and location based services) if the domain controller collaborate with a content distributor, and if the domain owner uses a device in the domain that is used to control domain membership or a payment card that is used to create or add devices into a domain. Then the content distributor would know all content bought from the same

distributor for the domain.

- Only iPod devices and platforms running Mac OS X or Microsoft Windows can join a domain in the Apple Fairplay scheme, limiting interoperability. As stated by Rowell [75], "Apple Fairplay is allowing Apple to lock iPod owners into its proprietary store". Our proposed scheme does not require a specific device type of a special operating system.

11.9.4 SmartRight

In this section we compare the four proposed implementations with the SmartRight scheme analysed in section 5.1.3. The main differences are as follows.

- In the SmartRight scheme there is no binding between the domain key and the domain owner. This means that any device can obtain the means for accessing domain content, as long as the device has not been revoked and the value of the PPN counter is not zero. This raises the leaf distribution problem. Our proposed schemes address this problem by strongly binding devices joining a domain to the domain owner.

- In order for a device to join a PPN, it must be SmartRight device compatible (in particular it must possess a smart card reader) and needs to be equipped with a Terminal module smart card and/or a Converter module smart card. These requirements increase the total system cost especially the costs of using and maintaining smart cards. Moreover, it may be inconvenient to add smart card readers to small devices. The proposed schemes only require devices to have Trusted Platform properties that can be met using TCG compliant devices, as discussed in this chapter.

- In the SmartRight scheme if the system is hacked, the current smart cards must be replaced, which is a potentially expensive and time-consuming process. In the proposed scheme hacking a device does not affect other domains, and the hacked device cannot receive content by eavesdropping on network as content protection keys is transferred encrypted using the destination device public encryption key.

- In the SmartRight scheme the network key is stored on the Terminal card. If all the Terminal cards are lost or fail, then all existing content will be unusable, and will need to be downloaded again from content distributor sites. The proposed schemes propose a secure way to backup and recover domain keys.

- In the SmartRight scheme the network key must be transferred by the Terminal card that most recently joined the network. If the end user forgets which Terminal card most recently joined the domain, then he/she must try all Terminal cards. This could pose a serious usability issue in networks with a large number of Terminal cards. Also, If this Terminal card is lost or stolen, then no devices can be added to the network. All proposed schemes do not have these restriction.

- In the SmartRight scheme a user must have converter smart card(s) in order to add new content to the PPN. In addition, he/she needs to have multiple Terminal smart card(s) to render digital content on a SmartRight device. It may not be convenient for a user to manage a smart card for each device he/she owns. All proposed schemes do not have such restriction.

- In the SmartRight scheme limiting the number of Terminal cards that can be added to a domain using the PPN counter makes the system inflexible for consumers. In addition, once the value of the PPN counter reaches

zero, it cannot be increased, and the only means of adding devices to the PPN is by reinitialising the domain. This would mean that all currently held content, encrypted with the old network key, would need to be re-downloaded, and a new network key generated. Our proposed schemes the domain size is dynamic and can be changed based on consumers need, as discussed in section 11.9.2.

- In the SmartRight scheme all the existing LECM keys are changed every time a Terminal card joins or leaves the network. Therefore, the greater the number of items of content in a domain, the longer it will take to encrypt all the new LECM keys with the network key. The proposed schemes do not change the domain specific key; leaving devices must communicate with the domain controller that ensures a device removes the domain key before it updates the domain-specific counters. This stops the domain owners abusing the system by adding devices and then removing them for using domain content in an unauthorised way.

- In the SmartRight scheme a backup and recovery policy for network keys and content has not been proposed. All proposed schemes propose different methods for backing up and recovering domain credentials.

11.9.5 DRM in a 3G Mobile Phone and Beyond

In this section we compare the four proposed implementations with the DRM in a 3G Mobile Phone and Beyond scheme analysed in section 5.1.5. The main differences are as follows.

- In the 3G Mobile Phone scheme adding a device into a domain requires

the joining device to send a request to the domain authority. The request contains a password. This password can be shared with others. Hence leaf distribution arises. Our proposed schemes address this problem by strongly binding devices joining a domain to the domain owner.

- In the 3G Mobile Phone scheme a domain is protected against abuse in two ways (a) The frequency with which a device is added to or removed from domains is monitored. This requires a potentially complex and sophisticated infrastructure, that processes and records potentially huge numbers of events. Moreover, before authorising a device to join a domain, the domain authority must parse the history log files to ensure the device is not abusing the system. This is likely to adversely affect system performance. (b) Domain creation is bound to owner private information or the ability to spend money. One major problem with this approach is user privacy, as the domain authority must hold confidential user information. In addition, point (b) does not prevent abuse of the system, as owner, for example, can use payment cards with lower limit or cancel his/her payment card after creating the domain.

- In the 3G Mobile Phone scheme the use of a domain authority requires the establishment and maintenance of a complex infrastructure that increases the overall cost and complexity of the system. The scheme does not discuss how such a domain authority can be realised. Three proposed schemes require a trusted authority to manage consumer domains. We described who could act and play the role of the trusted authority, i.e. banks can play the role of a trusted authority for the authorised domain management using an electronic payment system scheme, and a mobile network provider can play the role of a trusted authority for the authorised domain management using a mobile phone and the authorised domain management using

location based services.

- In the 3G Mobile Phone scheme every time a device is revoked or removed from a domain, the domain-specific key pair has to be changed, which requires the user to connect all devices to the Internet in order to install the new domain key. Moreover, all domain content licence keys must be re-encrypted with the new domain key. Users are likely to find such a procedure inconvenient. All domain content licence keys are stored within domain devices, and must therefore be tracked. This results in a significant overhead if a large quantity of content is present. Our proposed schemes do not require the domain specific-key to be changed when membership of a domain changes. Hence our schemes do not require devices to be continually available. Our proposed scheme use different measures for protecting domain content when a device joins or leaves a domain, or if a device is hacked, as discussed earlier this chapter.

11.9.6 DRM Security Architecture for Home Networks

In this section we compare the four proposed implementations with the DRM Security Architecture for Home Networks scheme analysed in section 5.1.6. The main differences are as follows.

- In the DRM Security Architecture for Home Networks scheme adding a device into a domain requires the joining device to send a request to the domain manager. The request contains the domain manager unique identifier. This identifier can be shared with others. Hence leaf distribution arises. Our proposed schemes address this problem by strongly binding devices joining a domain to the domain owner.

- In the DRM Security Architecture for Home Networks scheme the global revocation list, GDRL, is associated with every downloaded item of content, which has a serious impact on the overall system performance. The size of the GDRL increases as more devices are revoked. This causes the time to download content to increase. In addition, the local revocation list, LRL, is very difficult to maintain, because it is associated with every item of content inside the authorised domain rather than being stored in a central location. All proposed schemes use a standard service for revocation, also revoked keys are stored in a centralised location and can be queried using an OCSP service.

- In the DRM Security Architecture for Home Networks scheme if the system is hacked, the authorised domain must be re-initialised and all domain content re-encrypted. In addition, a backup and recovery policy has not been proposed for domain keys and content. As discussed before, in our schemes the domain key is fixed and a secure backup and recovery mechanism is proposed for each DRM scheme.

- The DRM Security Architecture for Home Networks scheme is not flexible in that the authorised domain size is fixed. The maximum number of devices that can join an AD depends on hardware factors. These factors include the storage available for storing secure domain information, i.e. the credential set and the device master key on the least capable device that is expected to join the AD. This is potentially inconvenient for users, especially corporate users. Our proposed schemes the domain size is dynamic and can be changed based on consumers need, as discussed in section 11.9.2.

- The DRM Security Architecture for Home Networks scheme requires de-

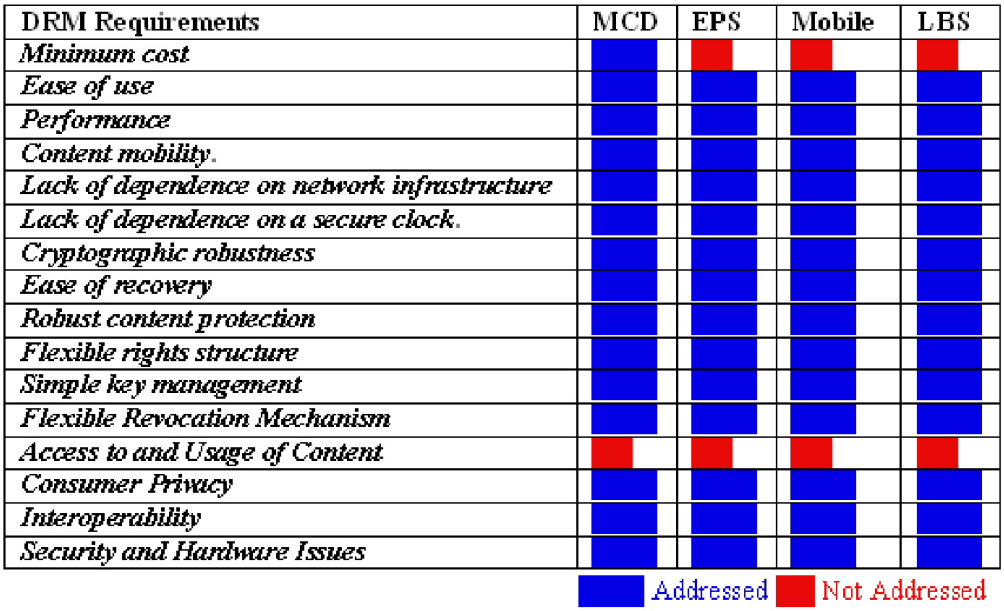
vices to possess a processor, memory, and protected storage to store secret domain information. Also, it requires a dedicated domain manager device. These increases the total costs. In addition, before devices can join a domain they must know the domain manager's GDI. This means joining devices must possess an I/O component to insert the manager GDI value. All proposd schemes require participating devices to have a Trusted Platform properties that can be met using TCG compliant devices, as discussed in this chapter. Also, all proposed schemes require devices to store a single domain-specific key.

11.9.7 Schemes Summary

Figure 11.1 summarises the analysis of the four schemes discussed in this chapter with respect to the ideal DRM requirements list defined in chapter 4. By comparing figure 11.1 against figure 5.2 in chapter 5, we conclude that all the schemes proposed in this chapter satisfy many more requirements than the previously proposed schemes.

11.10 Summary

This chapter contains an analysis of the general DRM framework presented in chapter 6, and the four schemes implementing the general framework. From this analysis we have concluded that all these schemes address the fundamental threats underlying content piracy, in addition to satisfying most of the defined DRM requirements. In addition, we have observed that the four schemes have differences in the following respects: the means used to identify domain ownership, the implementation costs, the nature of the trusted domain controller,



MCD summarises the main findings for the scheme discussed in section 11.3, EPS summarises the main findings for the scheme discussed in section 11.4, Mobile summarises the main findings for the scheme discussed in section 11.5, and LBS summarises the main findings for the scheme discussed in section 11.6.

Figure 11.1: Summary of the Analysis of the Four Schemes Proposed in this Thesis.

support for domains in multiple locations, convenience for personal networks, and suitability for enterprise DRM.

Chapter 12

Conclusions and Directions for Further Research

Contents

This chapter summarises the primary contributions of this thesis and gives some directions for further research.

12.1 Main Contributions

This thesis focuses on a fundamental consumer, content owner and copyright law requirement, namely how to allow content to be freely used by all devices owned by an individual, and simultaneously stop content from being illegally transferred to other devices. Current DRM schemes do not satisfy many user, content owner, and/or copyright law acceptance criteria for digital content protection and consumption; a system will fail if consumers or content providers

do not accept it. In addition, recent research has demonstrated that billions of dollars are lost as a result of piracy every year [17, 19, 45]. These issues underline the importance of finding an acceptable DRM solution in the near future.

We started by defining an ideal list of DRM requirements from the points of view of users, content providers and copyright law. We then developed an approach for assessing DRM systems based on the defined DRM requirements; we used this approach to analyse six of the most widely discussed DRM schemes incorporating the authorised domain concept. We found that the analysed schemes do not satisfy many user, content owner, and/or copyright law acceptance criteria. Studying the shortcomings of these schemes helped us to isolate the main issues underlying content piracy, i.e. Root Distribution and Leaf Distribution. In addition, we observed that software only mechanisms do not fully protect domain credentials, since such mechanisms have been hacked many times. Most of the analysed schemes require the addition of a hardware module to protect domain credentials and/or to enforce usage rules.

These observations led to a fundamental requirement underlying the model we propose, that is we require domain devices to be trusted platforms with certain properties. We showed that devices meeting the TCG specifications satisfy most of the properties we require of a device. TCG compliant devices are not expensive, and are now emerging into the marketplace; however, it will take some time until TCG functionality is integrated into all types of device, e.g. car CD players, TVs, etc. Also, many challenges still remain to be solved in TCG specifications and are subject of ongoing research.

We then gave a general framework for DRM systems, that involves creating

a domain owned by a single owner, where all devices joining a domain are bound to the domain owner. Each domain has a domain-specific secret key that is securely generated and protected by a trusted domain controller. This key is not available even to the domain owner, is shared by all devices in the domain, cannot be copied between devices, and is used to encrypt individual content encryption keys. The domain key is transferred from the trusted domain controller to a device joining a domain after the controller has authenticated the domain owner. In addition, each domain has two associated limits maintained and controlled by a trusted authority, one to control the number of devices that can simultaneously access domain content, and the other to control the total number of devices that can join a domain. Subsequently, we illustrated how the proposed framework allows content to be freely exchanged between devices in the same domain, and also allows controlled content sharing between devices in multiple domains. In addition, we illustrated how schemes conforming to the framework prevent illicit content proliferation.

Next, we described four implementations of the general framework. Each of these schemes uses a different means for identifying the ownership of domain devices, i.e. for binding a domain to a single owner to ensure that only a single consumer owns and manages each domain. In addition, each of these schemes uses a different type of trusted domain controller, the functionality of which relates closely to the method used to identify domain ownership. The four schemes are as follows.

- In the Authorised Domain Management using a Master Control Device scheme, the trusted domain controller is a master control device, and the domain owner is authenticated using two-factor authentication, which in-

volves "something the domain owner has", i.e. a master control device, and "something the domain owner is or knows", i.e. a biometric or password/PIN authentication mechanism that is implemented by the master control device.

- In the Authorised Domain Management using an Electronic Payment System scheme, the trusted domain controller is a third party TA, and the domain owner is authenticated using a payment card, building on the existing electronic payment system by ensuring that the name and the date of birth associated with the payment card are the same for all devices joining a domain.
- The Authorised Domain Management using a Mobile Phone scheme involves the use of a domain-specific mobile phone and a mobile phone network operator acting as the trusted domain controller. This network operator must authenticate the domain owner before a device is permitted to join a domain.
- The Authorised Domain Management using Location Based Services scheme uses a third party TA as the trusted domain controller. This TA ensures that devices joining a consumer domain are located in physical proximity to one of the domain registered addresses. This restricts domain membership to devices in predefined geographical locations, helping to ensure that a single consumer owns and manages each domain.

The general approaches of these implementations could also be useful in various other applications requiring strong authentication, because they all strongly bind a domain to its owner.

We then assessed the general framework and the four schemes using the defined

list of requirements. From this analysis we concluded that all these schemes address the fundamental threats underlying content piracy, in addition to satisfying most of the defined DRM requirements, including: Ease of use, Content mobility, Performance, Ease of recovery, Robust content protection, Privacy and Interoperability. Finally, we compared the schemes and noted differences in the following respects: the means used to identify domain ownership, the implementation costs, the nature of the trusted domain controller, support for domains in multiple locations, convenience for personal networks, and suitability for enterprise DRM.

12.2 Future Research

We conclude this thesis by noting some areas for further research. One important issue that is not addressed in the four discussed schemes is meeting copyright law requirements. The proposed schemes satisfy many copyright law requirements; however, some requirements have not been fully analysed, e.g. fair use. One important issue that is not addressed in the four discussed schemes is meeting copyright law requirements. The next step is to make any necessary modifications to the schemes so that they satisfy all relevant legal requirements, including copyright law requirements.

More work is needed on in Digital Rights Management for corporate use of content. Large organisations are likely to have requirements which differ from those applying in the personal network environment. As a result, the schemes proposed in this thesis may not be appropriate for corporate use; for example, organisations are likely to have larger domains covering multiple locations, more than one person might manage a corporate domain, etc. It would therefore be

helpful to consider how the framework itself, and the four implementations, might be modified to address the particular needs of large organisations.

In this thesis we do not address the problem of dynamic licensing (which is sometimes also referred to as stateful licensing). The next step is to make any necessary modifications to the schemes so that they address the problem of stateful licensing within an authorised domain.

The proposed protocols have only been subjected to informal, ad hoc, analysis, and security vulnerabilities may remain. Therefore, a more formal security analysis of the security protocols would be highly desirable.

The four schemes have not been implemented. Prototyping would enable the practicality of the schemes to be tested, and would also enable an assessment of their performance characteristics.

Finally, a detailed analysis of the specifications due to be released in the near future by the Mobile Phone Working Group (of the TCG) would be helpful, in particular to see if these specifications provide all the functionality required by trusted platforms, as defined in this thesis.

Bibliography

[1] Property attestation–scalable and privacy–friendly security assessment of peer computers. Technical report, RZ 3548, IBM Research, May 2004.

[2] 3rd Generation Partnership Project. *3GPP TS 21.133 — 3G Security; Security Threats and Requirements. Specification version 4.1.0 Release 4*, December 2001.

[3] 3rd Generation Partnership Project. *3GPP TS 33.120 — 3G Security; Security Principles and Objectives. Specification version 4.0.0 Release 4*, March 2001.

[4] 3rd Generation Partnership Project. *3GPP TS 33.102 — 3G Security; Security architecture. Specification version 7.0.0 Release 7*, December 2005.

[5] 3rd Generation Partnership Project. *3GPP TS 23.271 — Functional stage 2 description of Location Services (LCS). Specification version 7.5.0 Release 7*, June 2006.

[6] 3rd Generation Partnership Project. *3GPP TS 33.220 — Generic Authentication Architecture (GAA) — System Description. Specification version 7.0.0 Release 7*, March 2006.

[7] 3rd Generation Partnership Project. *3GPP TS 33.919 — Generic Authentication Architecture (GAA) — Generic Bootstrapping Architecture. Specification version 7.4.0 Release 7*, June 2006.

[8] Imad Abbadi. Digital asset protection in personal private networks. In *8th International Symposium on Systems and Information Security (SSI 2006), Sao Jose dos Campos, Sao Paulo, Brazil*, November 2006.

[9] Imad Abbadi. Authorised domain management using location based services. In Adrian David Cheak, Peter H J Chong, Winston Seah, and Shum Ping, editors, *Mobility '07: proceedings of the 4th International Conference on Mobile Technology, Applications & Systems*, pages 288–295. ACM Press, NY, September 2007.

[10] Imad Abbadi. Digital rights management using a master control device. In I. Cervesato, editor, *ASIAN '07: Proceedings of the 12th Annual Asian Computing Science Conference Focusing on Computer and Network Security*, volume 4846 of *Lecture Notes in Computer Science*, pages 126–141. Springer-Verlag, Berlin, December 2007.

[11] Imad Abbadi and Chris Mitchell. Digital rights management using a mobile phone. In *ICEC '07: Proceedings of the ninth international conference on Electronic commerce*, pages 185–194. ACM Press, NY, August 2007.

[12] Ross Anderson. Trusted computing frequently asked questions, 2003. http://www.cl.cam.ac.uk/∼rja14/tcpa-faq.html.

[13] Apple Inc. Apple Fairplay, 2006. http://www.apple.com/lu/support/itunes/authorization.html.

[14] Bill Arbaugh. Improving the TCPA specification. *IEEE Computer*, 35(8):77–79, August 2002.

[15] Jakob Bardram, Rasmus Kjr, and Michael Pedersen. Context-aware user authentication – supporting proximity-based login in pervasive computing. In Anind K. Dey, Albrecht Schmidt, and Joseph F. McCarthy, editors, *UbiComp 2003*, volume 2864 of *Lecture Notes in Computer Science*, pages 107–123. Springer-Verlag, Berlin, 2003.

[16] Tobias Bauckhage. Digital rights management: Economic aspects. In E. Becker, W. Buhse, D. Günnewig, and N. Rump, editors, *Digital Rights Management: Technological, Economic, Legal and Political Aspects*, volume 2770 of *Lecture Notes in Computer Science*, pages 234–249. Springer-Verlag, Berlin, 2003.

[17] BBC News. Piracy blamed for CD sales slump, 2002. http://news.bbc.co.uk/hi/english/entertainment/new_media/newsid_1841000/1841768.stm.

[18] Ernie Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In Vijay Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *Proceedings of 11th ACM Conference on Computer and Communications Security*, pages 132–145. ACM Press, Washington DC, 2004.

[19] BSA and IDC Global Software. 2005 piracy study, 2005. http://www.bsa.org.

[20] Norris Carden. iTunes and iPod in the enterprise. *The Journal of the International Systems Security Association*, pages 22–25, May 2007.

[21] L. Chen, S. Pearson, and A. Vamvakas. On enhancing biometric authentication with data protection. In *Proceedings of the Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies*, volume 1, pages 249–252. IEEE, 2000.

[22] Liqun Chen, Rainer Landfermann, Hans Löhr, Markus Rohe, Ahmad-Reza Sadeghi, and Christian Stüble. A protocol for property-based attestation. In *STC '06: Proceedings of the first ACM workshop on Scalable trusted computing*, pages 7–16, New York, NY, USA, 2006. ACM.

[23] J. Cuellar, J. Morris, D. Mulligan, J. Peterson, and J. Polk. Geopriv requirements. RFC 3693, Internet Engineering Task Force, February 2004.

[24] M. Danley, D. Mulligan, J. Morris, and J. Peterson. Threat analysis of the geopriv protocol. RFC 3694, Internet Engineering Task Force, February 2004.

[25] Dorothy E. Denning and Peter F. MacDoran. Location-based authentication: grounding cyberspace for better security. *Computer Fraud & Security, Elsevier Science*, 1996(2):12–16, February 1996.

[26] Alex W. Dent and Chris J. Mitchell. *User's Guide to Cryptography and Standards*. Artech House, Norwood, MA, USA, 2005.

[27] D. Eastlake and P. Jones. US Secure Hash Algorithm 1 (SHA1). RFC 3174, Internet Engineering Task Force, September 2001.

[28] Taher ElGamal. A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.

[29] Federal Information Processing Standards Publication. Digital signature standard (DSS) (FIPS PUB 186-2), 1994. http://www.itl.nist.gov/fipspubs/fip186.htm.

[30] Federal Information Processing Standards Publication. Data Encryption Standard (DES) (FIPS PUB 46-3), 1999. http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf.

[31] Federal Information Processing Standards Publication. Advanced Encryption Standard (AES) (FIPS PUB 197), 2001. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[32] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer-Verlag, Berlin, 1993.

[33] Eimear Gallery. An overview of trusted computing technology. In Chris J. Mitchell, editor, *Trusted Computing*, chapter 3, pages 29–113. IEE, 2005.

[34] Eimear Gallery and Allan Tomlinson. Secure delivery of conditional access applications to mobile receivers. In Chris J. Mitchell, editor, *Trusted Computing*, chapter 7, pages 195–237. IEE, 2005.

[35] Trusted Computing Group. Trusted platform module FAQ.

[36] André Günther and Christian Hoene. Measuring round trip times to determine the distance between WLAN nodes. In Raouf Boutaba, Kevin C. Almeroth, Ramn Puigjaner, Sherman X. Shen, and James P. Black, editors, *Proceedings of 4th International IFIP-TC6 Networking Conference, Waterloo, Canada*, volume 3462 of *Lecture Notes in Computer Science*, pages 768–779. Springer-Verlag, Berlin, May 2005.

[37] S. Haber, B. Horne, J. Pato, T. Sander, and R. E. Tarjan. If piracy is the problem, is DRM the answer? In E. Becker, W. Buhse, D. Günnewig, and N. Rump, editors, *Digital Rights Management: Technological, Economic, Legal and Political Aspects*, volume 2770 of *Lecture Notes in Computer Science*, pages 224–233. Springer-Verlag, Berlin, 2003.

[38] Vivek Haldar, Deepak Chandra, and Michael Franz. Semantic remote attestation: a virtual machine directed approach to trusted computing. In *VM'04: Proceedings of the 3rd conference on Virtual Machine Research And Technology Symposium*, pages 3–3, Berkeley, CA, USA, 2004. USENIX Association.

[39] V. Hassler. *Security Fundamentals for E-commerce*. Artech House, Norwood, MA, USA, 2001.

[40] Natali Helberger, Nicole Dufft, Stef van Gompel, Kristof Kerenyi, Bettina Krings, Rik Lambers, Carsten Orwat, and Ulrich Riehm. Digital rights management and consumer acceptability. Technical report, DG Information Society, December 2004. http://www.indicare.org/soareport.

[41] R. Housley, W. Polk, W. Ford, and D. Solo. Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. RFC 3280, Internet Engineering Task Force, April 2002.

[42] Bradley Huffaker, Marina Fomenkov, Daniel J. Plummer, David Moore, and K. Claffy. Distance metrics in the Internet. In *IEEE International Telecommunications Symposium*, 2002. http://www.caida.org/publications/papers/2002/ Distance/distance.pdf.

[43] Seong Oun Hwang, Ki Song Yoon, Kyung Pyo Jun, and Kwang Hyung Lee. Modeling and implementation of digital rights. *Journal of Systems and Software*, 73(3):533–549, April 2003.

[44] IBM Research Division Almaden Research Center. xCP cluster protocol, 2003. http://www-03.ibm.com/solutions/digitalmedia/doc/content /bin/xCPWhitepaper_final_1.pdf.

[45] International Federation of the Phonographic Industry (IFPI). Music piracy report, 2005. http://www.ifpi.org/site-content/library/piracy2005.pdf.

[46] International Organization for Standardization. *ISO/IEC 9798-3, Information technology — Security techniques — Entity authentication — Part 3: Mechanisms using digital signature techniques*, 2nd edition, 1998.

[47] International Organization for Standardization. *ISO/IEC 10118-1, Information technology — Security techniques — Hash-functions — Part 1: General*, 2nd edition, 2000.

[48] International Organization for Standardization. *ISO/IEC 9797-2, Information technology — Security techniques — Message Authentication Codes (MACs) — Part 2: Mechanisms using a dedicated hash-function*, 1st edition, 2002.

[49] International Organization for Standardization. *ISO/IEC 21481: Information technology — Telecommunications and information exchange between systems — Near Field Communication Interface and Protocol -2 (NFCIP-2)*, 2005.

[50] International Organization for Standardization. *ISO/IEC 18033-2, Information technology — Security techniques — Encryption algorithms — Part 2: Asymmetric ciphers*, 2006.

[51] International Organization for Standardization. *ISO/IEC FCD 19772, Information technology — Security techniques — Authenticated encryption mechanisms*, 2007.

[52] Ghassan Kbar and Wathiq Mansoor. Mobile station location based on hybrid of signal strength and time of arrival. In *Proceedings of the Inter-*

national Conference on Mobile Business, pages 585–591. IEEE Computer Society, 2005.

[53] V. Khu-smith and C. J. Mitchell. Using EMV cards to protect e-commerce transactions. In K. Bauknecht, A. Min Tjoa, and G. Quirchmayr, editors, *EC-Web 2002, 3rd International Conference on Electronic Commerce and Web Technologies*, volume 2455 of *Lecture Notes in Computer Science*, pages 388–399. Springer-Verlag, Berlin, September 2002.

[54] V. Khu-smith and C. J. Mitchell. Using GSM to enhance e-commerce security. In *WMC '02, Proceedings of the Second ACM International Workshop on Mobile Commerce*, pages 75–81. ACM Press, September 2002.

[55] V. Khu-smith and C. J. Mitchell. Enhancing e-commerce security using GSM authentication. In *E-Commerce and Web Technologies – 4th International Conference*, volume 2738 of *Lecture Notes in Computer Science*, pages 72–83. Springer-Verlag, Berlin, September 2003.

[56] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: keyed-hashing for message authentication. RFC 2104, Internet Engineering Task Force, February 1997.

[57] Dirk Kuhlmann and Robert A. Gehring. Trusted platforms, DRM, and beyond. In E. Becker, editor, *Digital Rights Management*, volume 2770 of *Lecture Notes in Computer Science*, pages 178–205. Springer-Verlag, Berlin, 2003.

[58] Ulrich Kühn, Klaus Kursawe, Stefan Lucks, Ahmad-Reza Sadeghi, and Christian Stüble. Secure data management in trusted computing. In *Cryptographic Hardware and Embedded Systems – CHES 2005*, volume 3659 of *Lecture Notes in Computer Science*, pages 324–338. Springer-Verlag, Berlin, 2005.

[59] Ulrich Kühn, Marcel Selhorst, and Christian Stüble. Realizing property-based attestation and sealing with commonly available hard- and software. In *STC '07: Proceedings of the 2007 ACM workshop on Scalable trusted computing*, pages 50–57, New York, NY, USA, 2007. ACM.

[60] Axel Küpper. *Location-Based Services: Fundamentals and Operation*. John Wiley & Sons Ltd, 2005.

[61] K. Kursawe, D. Schellekens, and B. Preneel. Analyzing trusted platform communication, 2005. In: ECRYPT-CRASH.

[62] Qiong Liu, Reihaneh Safavi-Naini, and Nicholas Paul Sheppard. Digital rights management for content distribution. In C. Johnson, P. Montague, and C. Steketee, editors, *Proceedings of the Australasian Information Security Workshop*, volume 21, pages 49 – 58. ACM Press, NY, 2003.

[63] Simon Liu and Mark Silverman. A practical guide to biometric security technology. *IT Professional*, 3(1):27–32, 2001.

[64] J. Lotspiech, S. Nusser, and F. Pestoni. Broadcast encryption's bright future. *Computer*, 35(8):75–63, August 2002.

[65] D. Maltoni, D. Maio, A. K. Jain, and S. Prabahakar. *Handbook of Fingerprint Recognition*. Springer-Verlag, Berlin, 2003.

[66] John Marchesini, Sean W. Smith, Omen Wild, Josh Stabiner, and Alex Barsamian. Open-source applications of tcpa hardware. In *ACSAC '04: Proceedings of the 20th Annual Computer Security Applications Conference*, pages 294–303, Washington, DC, USA, 2004. IEEE Computer Society.

[67] MasterCard International. *Secure Payment Application (SPA)*, 2004. http://www.mastercardintl.com.

[68] T. Matsumoto, H. Matsumoto, K. Yamada, and S. Hoshino. Impact of artificial 'gummy' fingers on fingerprint systems. In *Proceedings of SPIE*, volume 4677, pages 275–289, 2002.

[69] T. S. Messerges and E. A. Dabbish. Digital rights management in a 3G mobile phone and beyond. In Joan Feigenbaum, Tomas Sander, and Moti Yung, editors, *Proceedings of the 3rd ACM workshop on Digital Rights Management*, pages 27–38. ACM Press, NY, 2003.

[70] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol — OCSP. RFC 2560, Internet Engineering Task Force, June 1999.

[71] Ryan Naraine. Wozniak's wheels of zeus tackles enterprise data encryption, 2004. http://www.eweek.com/article2/0,1759,1734857,00.asp.

[72] A. Niemi and J. Arkko. Hypertext transfer protocol (HTTP) digest authentication using authentication and key agreement (AKA). RFC 3310, Internet Engineering Task Force, September 2002.

[73] D. O'Mahony, M. Peirce, and H. Tewari. *Electronic Payment Systems for E-Commerce*. Artech House, Norwood, MA, USA, 2001.

[74] Open Mobile Alliance. *DRM Specification — Version 2.0*, 2006.

[75] A. Pashalidis and C. J. Mitchell. Using GSM/UMTS for single sign-on. In K. Bauknecht, A. Min Tjoa, and G. Quirchmayr, editors, *Proceedings of SympoTIC '03, Joint IST Workshop on Mobile Future and Symposium on Trends in Communications*, pages 138–145. IEEE Press, October 2003.

[76] A. Pashalidis and C. J. Mitchell. Using EMV cards for single sign-on. In S. K. Katsikas, S. Gritzalis, and J. Lopez, editors, *Public Key Infrastruc-*

ture: First European PKI Workshop, volume 3093 of *Lecture Notes in Computer Science*, pages 205–217. Springer-Verlag, Berlin, June 2004.

[77] Fabien A. P. Petitcolas. Digital watermarking. In E. Becker, editor, *Digital Rights Management*, volume 2770 of *Lecture Notes in Computer Science*, pages 81–92. Springer-Verlag, Berlin, 2003.

[78] Benny Pinkas and Tomas Sander. Securing passwords against dictionary attacks. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 161–170. ACM Press, New York, NY, USA, 2002.

[79] Fred C. Piper and Sean Murphy. *Cryptography: A Very Short Introduction*. Oxford University press, New York, 2002.

[80] B. C. Popescu, F. L. A. J. Kamperman, B. Crispo, and A. S. Tanenbaum. A DRM security architecture for home networks. In Joan Feigenbaum, Tomas Sander, and Moti Yung, editors, *Proceedings of the 4th ACM workshop on Digital Rights Management*, pages 1–10. ACM Press, NY, 2004.

[81] R. Rivest. The MD5 Message-Digest Algorithm. RFC 1321, Internet Engineering Task Force, April 1992.

[82] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. In *Communications of the ACM*, volume 21, pages 120–126. ACM Press, NY, 1978.

[83] Laurie Freeman Rowell. The ballad of DVD JON. *netWorker*, 10(4):28–34, December 2006.

[84] A. Sadeghi. Trusted computing — special aspects and challenges. In V. Geffert et al., editor, *SOFSEM*, volume 4910 of *Lecture Notes in Computer Science*, pages 98–117. Springer-Verlag, Berlin, 2008.

[85] Ahmad-Reza Sadeghi and Christian Stüble. Property-based attestation for computing platforms: caring about properties, not mechanisms. In *NSPW '04: Proceedings of the 2004 workshop on New security paradigms*, pages 67–77, New York, NY, USA, 2004. ACM.

[86] Tomas Sander. Golden times for digital rights management? In P. Syverson, editor, *Financial Cryptography*, volume 2339 of *Lecture Notes in Computer Science*, pages 64–74. Springer-Verlag, Berlin, 2002.

[87] Mark Stefik. Letting loose the light: Igniting commerce in electronic publication. In Mark Stefik, editor, *Internet Dreams — Archetypes, Myths, and Metaphors*, pages 219–254. ACM Press, 1997.

[88] S. R. Subramanya and Byung K. Yi. Digital rights management. *IEEE Potentials*, 25(2):31–34, April 2006.

[89] Thomson. SmartRight technical white paper, 2003. http://www.smartright.org/images/SMR/content/SmartRight_tech_whitepaper_jan28.pdf.

[90] Bori Toth. Biometric liveness detection. *The International Journal For Information Assurance Professionals*, 10(8):291–298, 2005.

[91] Trusted Computing Group. *Infrastructure Working Group Architecture, Part II, Integrity Management. Specification version 1.0 Revision 1.0*, 2006.

[92] Trusted Computing Group. *TPM Main, Part 1, Design Principles. Specification version 1.2 Revision 94*, 2006.

[93] Trusted Computing Group. *TPM Main, Part 2, TPM Structures. Specification version 1.2 Revision 94*, 2006.

[94] Trusted Computing Group. *TPM Main, Part 3, Design Principles. Specification version 1.2 Revision 94*, 2006.

[95] Visa International. *3-D Secure Protocol Specification: Core functions Version 1.0.2*, 2004. http://www.international.visa.com/.

[96] Aaron Weiss. Will the open, unrestricted PC soon become a thing of the past? *Journal of Trusted Computing*, 10(3):18–25, September 2006.

[97] Susan Wiedenbeck, Jean-Camille Birget, Alex Brodskiy, Jim Waters, and Nasir Memon. Authentication using graphical passwords: Effects of tolerance and image choice. In *Proceedings of the 2005 symposium on Usable privacy and security*, pages 1–12. ACM Press, New York, NY, USA, 2005.

[98] Susan Wiedenbeck, Jim Waters, Leonardo Sobrado, and Jean-Camille Birget. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In *Proceedings of the working conference on Advanced visual interfaces*, pages 177–184. ACM Press, New York, NY, USA, 2006.

[99] Yilin Zhao. Standardization of mobile phone positioning for 3G systems. *IEEE Communications Magazine*, 40(7):108–116, July 2002.