TOWARDS ACCOUNTABLE ANONYMITY IN DIGITAL RIGHTS MANAGEMENT

A thesis submitted to the University of Manchester
for the degree of Master of Philosophy
in the Faculty of Engineering and Physical Sciences

2010

By

Mark Jones

School of Computer Science

Contents

Glossary **8**

Abstract **10**

Declaration **12**

Copyright **13**

Acknowledgements **14**

1 Introduction **15**

1.1 Research Motivation 18

1.2 Research Aim and Objectives 18

1.3 Thesis Structure 19

2 Setting the Scene **20**

2.1 Digital Rights Management: An Overview 21

2.2 Our Vision: Accountable Anonymity in DRM 25

2.3 State-of-the-Art Accountable Anonymity in DRM 27

2.3.1 Smart-Card-Based DRM Systems 29

2.3.2 Identity-Based DRM Systems 36

2.4 Requirements for an Accountable Anonymity Service in DRM . . 39

3 Existing Solutions for Achieving Anonymity 42

3.1 Pseudonymity 44

3.1.1 Digital Pseudonyms and Return Addresses 45

3.1.2 Pseudonyms and Credentials 49

3.1.3 Pseudonymity with Accountability 53

3.2 Digital Signatures 58

3.2.1 Blind Signatures 62

3.2.2 Fair Blind Signatures 63

3.2.3 Group Signatures 67

3.3 Digital Certificates 69

3.3.1 Anonymous Identity Certificates 72

3.3.2 Anonymous Authorisation Certificates 76

3.3.3 k-Times Anonymous Authentication 81

3.4 Identity Escrow 84

3.4.1 Appointed Verifiers 88

3.4.2 Multiple Collating Trusted Third Parties 90

3.5 Chapter Analysis 94

4 Privacy-Preserving Online Payment 98

4.1 Signature-Based Online Payment 102

4.2 Online Payment with a Trusted Third Party 104

4.3 Online Payment without Keys 109

4.4 What is Missing? 114

5 Achieving Accountable Anonymity in DRM 120

5.1 Best Way Forward 121

5.2 A^{2}DRM Design Preliminaries 122

5.2.1 Design Requirements 122

5.2.2 Design Assumptions 124
5.2.3 Building Blocks 125
5.2.4 Notation 137
5.3 The A^{2}DRM Protocol 138
5.3.1 A^{2}DRM Overview 138
5.3.2 A^{2}DRM in Depth 146
5.4 Evaluation 161
5.4.1 A^{2}DRM Evaluation Metrics 161
5.4.2 A^{2}DRM Evaluation 169

6 Conclusions and Directions for Future Work **192**

Bibliography **196**

Word count: 49861

List of Tables

3.1 Evaluation of existing anonymity solutions against our requirements. 95

4.1 Evaluation of existing privacy-preserving online payment solutions against our requirements. 119

5.1 A^2DRM protocol notation. 137

5.2 The respective probabilities $P_0 \rightarrow P_9$ for each consumer $C_0 \rightarrow C_9$ for each example case $1 \rightarrow 4$. 172

List of Figures

2.1 A typical DRM model. 22

2.2 A typical session (S_1) comprising a typical set of transactions (t_1, t_2, t_3 and t_4) used by Alice to acquire some digital content and a corresponding licence. 24

2.3 A typical set of sessions: two different sessions to acquire two different contents and a corresponding licence. Each session consists of four transactions, as detailed in Figure 2.2. 26

3.1 A simple mix system used by Alice to send ciphertext c to Bob. c is the encryption of a message using Bob's public key. 47

3.2 X.509 certificate structure. 71

3.3 The tree-like structure of a consumer's anonymous certificates. . . 74

3.4 The relationship between an authorisation certificate and an identity certificate. 77

3.5 Multiple collating TTPs work together to issue Alice with a secure identity token $E_2(E_1(E_0(PK)))$. 91

4.1 The withdrawal, payment and traceback protocols from Camenisch *et al.* [CMS96]. 108

4.2 A simple hash-tree. 111

5.1 A pseudonym certificate. 139
5.2 The link between Alice's civil identity and her pseudonym. Her identity is stored in the corresponding certificate's *Holder* field. It is encrypted so that it can only be revealed by the issuer, CA_0. . 140
5.3 Pseudonym tree. 142
5.4 Amended DRM model. The section of the DRM model we have amended is enclosed within the dotted area. 143
5.5 A^2DRM protocol messages. 148
5.6 Our anonymity set for our test-environment. 170
5.7 Alice's pseudonyms for our example cases. 171
5.8 The link between Alice's civil identity and her transaction pseudonym. 186

Glossary

A²DRM Anonymous but Accountable Digital Rights Management 122

AA Authorisation Authority 77

AS Anonymity Set 162

CA Certification Authority 71

CRL Certificate Revocation List 61

DRM Digital Rights Management 16

DVD Digital Versatile Disk 20

DVLA Driver and Vehicle Licensing Agency 49

FTP File Transfer Protocol 154

G-10 Governors of the Group of 10 Nations 115

HTTP HyperText Transfer Protocol 154

IDA Information Dispersal Algorithm 91

IETF Internet Engineering Task Force 29

IP Internet Protocol 48

ISO International Organisation for Standardisation 42

MD5 Message Digest Algorithm, version 5 30

MPEG Moving Picture Experts Group 23

PDA Personal Digital Assistant 28

PIN Personal Identification Number 35

PKI Public Key Infrastructure 36

REL Rights Expression Language 23

RFC Request For Comments 29

RSA Rivest, Shamir and Adleman 59

SP Service Provider 91

TCP/IP Transmission Control Protocol and Internet Protocol 154

TTP Trusted Third Party 28

Abstract

Digitalisation, coupled with the distribution channels of the Internet, provides a platform for efficient digital content duplication and distribution. Some individuals have abused this opportunity, duplicating and distributing digital content without the owner's permission. Digital Rights Management, or DRM for short, has emerged as a mechanism to protect the rights of content creators by obstructing illegitimate access to the content; whilst allowing legitimate access. In doing so, DRM systems typically use a consumer's identity in the process of identification and authentication.

In this thesis we present and discuss ideas on how a consumer of a DRM system could acquire digital content without disclosing his identity, provided he acts legitimately. To prevent identity theft, we argue that a consumer's identity should not be used for identification purposes. However, if a consumer is found to be acting illegitimately, an authority should be able to determine his identity to hold him accountable. To highlight such a need, we examine existing DRM solutions, critiquing their ability to provide accountable anonymity. In a similar vein, we critique anonymity primitives that we could base such a DRM system on.

Most digital contents are not free and therefore must be paid for before access is granted. Payment is therefore an important sub-system. We believe that, as part of preserving consumers' privacy, consumers should be able to complete

their payment without disclosing their identity. In line with our accountability requirement however, malicious consumers should have their identity revealed so they can be held accountable. We examine and critique existing payment systems that provide consumers with accountable anonymity.

From our discussions, we propose a new anonymous but accountable service in DRM. We call this, A^2DRM. We piece together anonymity primitives to provide our consumers with accountable anonymity. We believe that by satisfying both content owners and content consumers, such a system could prove to be successful in the world of digital content distribution. To realise such a system, we utilise pseudonymity, identity escrow, identity certificates, authorisation certificates, digital signatures and rights expression languages.

Key terms—digital rights management, anonymity, accountability, pseudonymity, and unlinkability.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

I would like to take this opportunity to thank my supervisor, Ning Zhang. Her endless fountain of knowledge aided me greatly throughout the duration of this research project. Her continuous support and patience proved to be pivotal in my success.

I would also like to thank both my parents, Karon and Brian Jones, for their support that was especially important to me. I am also grateful for the help I received from Victoria Jones. Finally, I would particularly like to say a big thank you to Roísín Mulligan for her continuous encouragement throughout the project's duration.

Chapter 1

Introduction

Anonymity is the property exhibited by a subject if it is not identifiable within a set of similar subjects [PH08]. It is apparent in regular, everyday activities; be it riding a bus, being a member of a peer support group, voting in political elections, or telephoning services such as the Samaritans [sam10]; everyone experiences some sort of anonymity in their everyday lives. These simple examples illustrate how freedom from detection, retribution, and embarrassment enable people to participate in scenarios that they wouldn't usually feel comfortable in. People, as a result, may feel more confident in communicating or seeking advice with their identities hidden — *i.e.* communicating anonymously. However, in a modern society, anonymity could be frowned upon if an individual makes too much of an effort to remain anonymous; conforming to the belief that: one would wish to remain anonymous only if one had something to hide. In the digital world however, it isn't so clear-cut.

With the wide adoption of the Internet, the digital world has seen an upward trend in the diversity of online communications [Rob00]. This has in turn lead to an increased use of online transactional services. It is subsequently becoming

easier for organisations, or commercial online service providers, to store a wide range of information concerning an individual's online activities, *e.g.* Google [Goo08] [JK10]. Based on such information, organisations could build individual-specific dossiers, from which trends could be inferred and monitored. Without the close-knit communities of yesteryear, it doesn't necessarily require too much personal information for an adversary to impersonate someone. Knowing as little as one's date of birth and mother's maiden name, along with one's name and home address, might be enough to authenticate one over the telephone. This widespread use of personal information for authentication purposes has made it particularly valuable. Individuals have therefore become more concerned about their privacy, and how their information is being used for unintended purposes; as a result becoming more stringent of the information they release. Individuals thus tend to feel more secure when participating in anonymous services where their identity privacy can be preserved. Identity privacy preservation, or anonymity, has therefore become an important security property for online transactional services [HP99].

A user of an online transactional service exhibits anonymity if he is able to access the service without disclosing his identity [ISO05]. Thus, for an individual to remain anonymous in an online transactional service, we should ensure that:

1. the individual's identity remains undisclosed;

2. his transactions cannot be linked back to his identity;

3. his transactions cannot be linked together.

Digital Rights Management (DRM) is one particular type of online transactional service. It is a mechanism used by content owners to distribute digital content(s) to consumers. In an ideal DRM environment, under certain circumstances

or conditions (*e.g.* when a consumer acts legitimately), the consumer should be allowed to acquire digital content(s) without revealing his civil identity[1]. Under extraordinary circumstances (*e.g.* when a consumer acts illegitimately by accessing a piece of content more times than he is entitled to do so), the consumer should have his identity traced to be held accountable. Ideally, a DRM system should therefore cater for *conditionally anonymous* consumer content acquisition.

Past experiences have shown that an unconditional anonymity service, or more precisely, anonymity without accountability, can be easily abused. Unsolicited bulk email, or spam, for example, is a typical problem experienced by millions of computer users [CL98]. It sprouts from the difficulty of holding anonymous email senders accountable. It is very difficult to trace such emails back to the sender's identity; making it difficult to hold them accountable. Examples such as this illustrate how anonymous abuse can be particularly problematic. It is therefore imperative that if we provide anonymity for consumers in a DRM system, then this service should encompass accountability; anonymity and accountability should go hand-in-hand. We should therefore piece together *consumer anonymity* and *transaction accountability* to provide the seemingly paradoxical notion of *accountable anonymity*:

1. **consumer anonymity**: a consumer is able to acquire some digital content(s) anonymously, provided that he, for example, satisfies some conditions — defined *a priori*;

2. **transaction accountability**: under exceptional circumstances, *e.g.* if a consumer violates the conditions, an authority is able to trace his civil identity to hold him accountable.

[1]Throughout this thesis we refer to different types of identities. By civil identity, we are referring to the identity the holder has in his everyday life with his bank, government, *etc.*. It is a collection of information such as name, address, job status, bank account number, *etc.*.

Accountable anonymity, or accountable privacy [BDWY06], is particularly important in the domain of DRM. In the remainder of this thesis, we focus on how accountable anonymity could be achieved in the context of DRM.

1.1 Research Motivation

Our research revolves around the field of accountable anonymity in DRM; including the three constituent parts: anonymity, accountability and DRM. The constituent parts lead us to pinpoint low-level primitives or building blocks that we can adopt and adapt to provide a viable solution.

DRM has emerged as a mechanism to protect the rights of content owners by obstructing illegitimate access to the content; whilst allowing legitimate access. In doing so, DRM systems typically use a consumer's identity in the process of identification and authentication. We believe that, if they wish, consumers should be allowed to preserve their privacy. However, if a consumer is found to be acting illegitimately, an authority should be able to trace his identity to hold him accountable.

1.2 Research Aim and Objectives

We aim to provide a means for an honest DRM content consumer to acquire digital content without revealing his identity. Furthermore, we also aim to provide a means for the identity of dishonest users to be traced without neglecting the right of identity privacy of the other content consumers. To do so, we have defined a set of project objectives:

1. identify state-of-the-art consumer/user privacy mechanisms;

2. identify state-of-the-art consumer/user privacy mechanisms for DRM;
3. identify weaknesses or/and security loopholes of these systems;
4. develop a set of requirements for which possible solutions should adhere to;
5. propose possible solutions to these problems.

1.3 Thesis Structure

The remaining part of this thesis is organised into six chapters. Chapter 2 sets the scene by presenting a typical scenario to explain why a content consumer may wish to acquire some content using a DRM service, and by doing so, exposes some security issues. It then introduces a typical DRM model from which we lay down our vision: accountable anonymity in DRM. We then critique existing DRM systems attempting to provide its consumers with accountable anonymity, against our vision. To realise our vision, we specify a set of requirements that should be adhered to in order to provide a viable solution.

Chapter 3 introduces a selection of anonymity-specific related works, critiquing them against the set of requirements specified in Chapter 2, finding their merits and exposing their weaknesses accordingly. A similar critical analysis of related works is undertaken in Chapter 4 for online payment. Based on these analyses we identify knowledge gaps on which we could focus our research efforts on.

Chapter 5 proposes a possible way forward where we provide our anonymous but accountable service in DRM, A^{2}DRM. Here we set out our assumptions and the design challenges — *i.e.* what the current challenges are and why they are challenges, conjecturing possible solutions and evaluating our proposal. We conclude this thesis in Chapter 6 where we outline possible areas for future research.

Chapter 2

Setting the Scene

Imagine a university student, Alice, who considers buying the movie tetralogy, Die Hard[1]. Alice however, being a student, cannot afford the recommended retail price offered by the high-street retailers such as HMV [hmv10] or WH Smiths [smi10]. She therefore considers an alternative medium. She considers purchasing a digital version of the tetralogy from an online digial content provider. Not only is this considerably cheaper for Alice but it is also more convenient. The cost of the Digital Versatile Disk (DVD)/Blu-ray teratology box-set includes the cost of the DVD/Blu-ray writing process, the complimentary booklet, the packaging, and the disk itself. Alice however, doesn't want these; she just wants the movies. The online digital content provider has the ability to provide Alice with just the movies; whereas the high-street retailers do not. One particular online digital content provider selling the Die Hard teratology uses a DRM service to supply its consumers with digital content.

In the remainder of this chapter we provide a DRM overview, explaining its importance and how it relates to Alice's problems. We then propose our vision:

[1]The Die Hard movie tetralogy is a Fox [fox10] production. Details of the movies can be found on the IMDb [imd10] website.

to supply digital content consumers with an anonymous but accountable DRM service. From this, we discuss the current state-of-the-art DRM systems, and critique the existing anonymous but accountable DRM service systems. From their limitations, we develop a set of requirements that, if fulfilled, would provide a more secure anonymous but accountable DRM service. We present these in Section 2.4.

2.1 Digital Rights Management: An Overview

Some digital content owners[2] use DRM as a platform for distributing their content(s) to consumers. In doing so, a DRM system aims to ensure that the owner's rights are protected; preventing consumers from obtaining and distributing content copies illegitimately. Its main goal is to obstruct illegitimate access to the content; whilst allowing legitimate access. DRM is a technical means of achieving this (as opposed to legal or social, for example).

A typical[3] DRM model, as illustrated in Figure 2.1, consists of six distinct entities: a content owner, a content distributor, a licence broker, a payment gateway, a content consumer and a playback device[4]. A content owner, who typically does not have access to distribution-channels for large-scale public distributions, acquires help from a dedicated content distributor.

The owner supplies the distributor with some content(s), which is locked in a *secure container* (1)[5]. The secure container is usually constructed using a

[2]In this thesis we amalgamate the content owner and the content creator into a single entity. Such extensions can be imagined where the content owner and content creator differ but this would be easy to abstract into our model.

[3]It is easy to imagine DRM systems that differ from our model. However, most DRM systems are modelled in a similar way. There may be examples where entities are merged or partitioned for operational segregation, but this is easy to abstract into our model.

[4]The content consumer and playback device usually reside at the same site. We call this, the *consumer site*.

[5]Although we concentrate on content protection through encrypted containers, other DRM

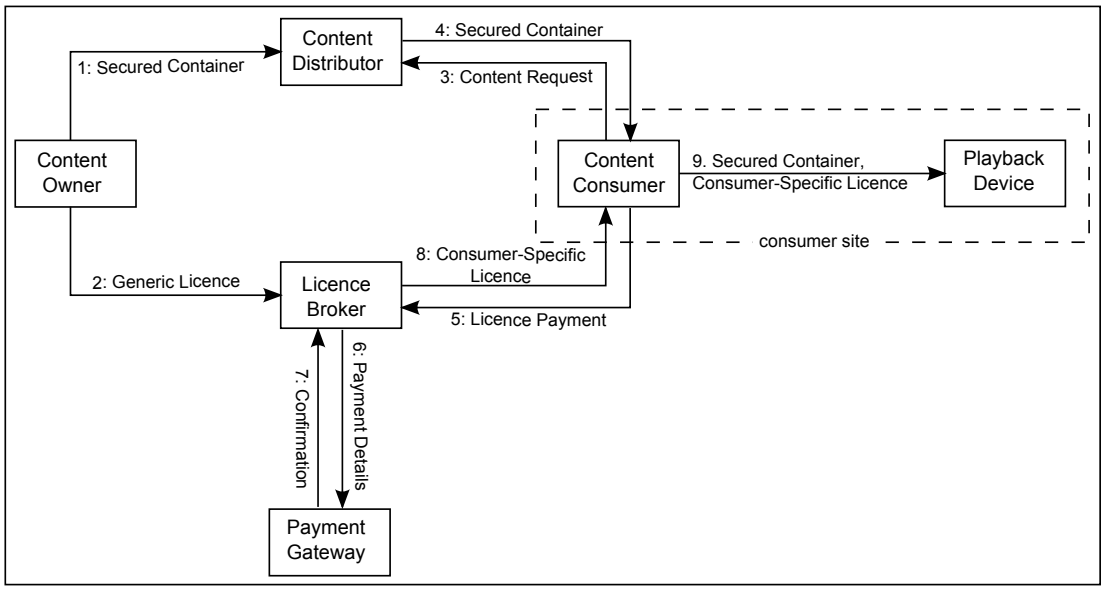

Figure 2.1: A typical DRM model.

cryptographic function and an encryption key. By distributing a secure container, consumers who are not granted the required access rights are prevented from accessing it. Those that are granted the required access rights can acquire a licence from the licence broker. Generic licences are passed to the licence broker from the content owner. It is the responsibility of the licence broker to use the generic licences to distribute consumer-specific licences. The licence contains the cryptographic key needed to unlock the container.

To acquire a licence, the consumer usually has to make a payment (5). This generally requires the help of a dedicated e-commerce party, or an electronic payment network such as Visa [vis10]; we call this, the *payment gateway*. The licence broker forwards the payment details to the payment gateway (6) and awaits a confirmation. Upon receiving confirmation of a successful payment (7), the licence broker issues the consumer with the appropriate licence (8). The consumer can then use the licence with a *compliant* playback device (9) to access

paradigms also exist: content watermarking [SLO94], content fingerprinting (*e.g.* Rivest [Riv92]), and even user fingerprinting, for example.

the content.

It is vitally important that the playback device is *compliant*: complying to DRM rules and regulations. Some rules and regulations might be:

- to ensure that the properly identified user owns the required rights to access the content;
- to enforce the access rights in the consumer's corresponding licence;
- to not act in any malicious way by, for example, transmitting or storing identifying information.

The playback device uses the key contained within this licence to unlock the secure container, granting the consumer the access he has been granted. A consumer's access rights are contained within the licence, generally encoded using a *rights expression language* (*e.g.* MPEG [Rig03]).

Rights expression languages, commonly abbreviated to Rights Expression Languages (RELs), are a widely used method for licence brokers to convey access rights in a machine readable format. By using RELs, the licence is able to express the access rights the holder has over the content. It is the role of the playback device to ensure that the rights are enforced. RELs are a well researched area (*e.g.* Coyle [Coy04]), and many such languages exist (*e.g.* Wang *et al.* [WDW+05], XrML [WLD+02], Moving Picture Experts Group (MPEG) [Rig03], and the Open Digital Rights Management [Ian00], to name a few).

The content owner, content distributor, licence broker, payment gateway, and playback device co-ordinate to supply consumers with digital content whilst enforcing copyright rules. For every piece of content a consumer acquires, he must also acquire a consumer-specific licence (8) and use a compliant playback device to access it (9). We call this process of content acquisition, a *session*.

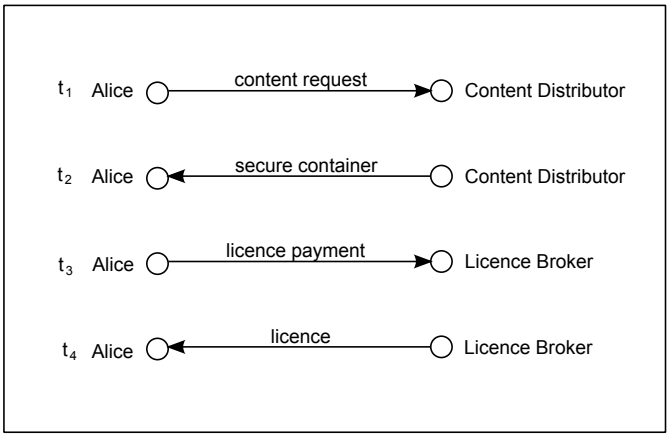

Figure 2.2: A typical session (S_1) comprising a typical set of transactions (t_1, t_2, t_3 and t_4) used by Alice to acquire some digital content and a corresponding licence.

Sessions are typically instigated by a consumer[6] with the sole aim of acquiring access to some digital content. In the context of DRM, each session consists of a series of sub-sessions that we call, *transactions*. As shown in Figure 2.2, a session, in its simplest form, consists of four transactions, namely:

1. a content request (t_1);
2. a content acquisition (t_2);
3. a licence payment (t_3);
4. a licence acquisition (t_4).

We can further explain this with the use of an example.

With reference to Figure 2.2, suppose Alice did wish to acquire the Die Hard tetralogy using a DRM service; so she instigates a session by requesting some content from the content distributor with a transaction (t_1). If Alice meets all the necessary requirements (*e.g.* she's not a included on a blacklist[7]), the content

[6]One could imagine distributor-instigated sessions, perhaps by handing out free disks to passers by in a town centre, but this is not particularly important.

[7]A blacklist is a register or list of persons who, for some reason or another, have been revoked

distributor supplies Alice with the corresponding secure container (t_2). To unlock the container, Alice must acquire a licence, which she requests from the licence broker (t_3). If Alice meets all the requirements (*e.g.* she's paid for the licence), the licence broker supplies Alice with the corresponding licence (t_4). By using the key within the licence to unlock the container, Alice is able to use her playback device to access the content. The series of transactions we have used here (t_1, t_2, t_3 and t_4) make up a session (S_1).

2.2 Our Vision: Accountable Anonymity in DRM

Alice is aware of DRM services (such as Windows Media DRM [Mic10], Apple Fairplay [App10b], and Sony OpenMG [Son10], being the most widely known) from which she could acquire digital content from, but she has some privacy concerns (like those discussed by Cohen [Coh03], for example). She believes that DRM services are unlikely to take unprofitable measures to protect her privacy; perhaps quite the opposite, acquiring plentiful personal information to use or share for commercial advantage [CA99]. So Alice remains reluctant to use a DRM service to acquire the movies.

Current DRM systems tend to neglect consumers' rights, such as Alice's concerns. These systems are generally geared towards ensuring that consumers do not violate the owners' rights, and in the process, fail to consider that consumers' rights are equally important. They typically use a consumer's identity during the process of identification and authentication. We believe that, as part of respecting consumers' rights, Alice, as a DRM consumer, should be allowed to acquire digital contents without having to expose her identity — to prevent her transactions

from the system. Persons appearing on the blacklist are assumed to be malicious and are thus not serviced.

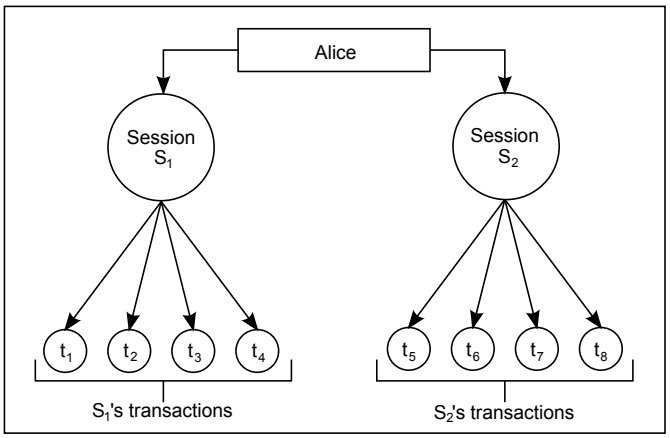

Figure 2.3: A typical set of sessions: two different sessions to acquire two different contents and a corresponding licence. Each session consists of four transactions, as detailed in Figure 2.2.

from being associated to her identity, provided she acts legitimately; otherwise, her anonymity is revoked so she can be held accountable. We can illustrate our concerns with an example.

If we refer back to our example in Section 2.1, Alice participated in one session to acquire some content (S_1). We now further this example (with reference to Figure 2.3) so that Alice uses a similar session (S_2) to acquire some more content and a corresponding licence. An observing adversary could piece together these two sessions and start to infer Alice's shopping habits. The more sessions the adversary observes, the more accurate the inferences are, and the more beneficial they become. We believe that an ideal DRM solution should prevent an adversary from piecing together sessions from a single consumer to preserve his privacy. To achieve this, we should hide the relationship between the consumer and his associated sessions; thus preventing an observing adversary from knowing which session belongs to which consumer. As a result, the consumer's shopping habits cannot be inferred. In line with this approach, we should also consider that, as shown in Figure 2.3, each session consists of a series of transactions. We should therefore also hide the relationship between the session and its associated

transactions.

Now, if a transaction or session is deemed to violate a DRM right or regulation, then it should be possible for an authority to determine the corresponding consumer's identity to hold him accountable. We should therefore ensure that these relationships are conditionally unlinkable; only linkable when the consumer is found to be acting maliciously.

2.3 State-of-the-Art Accountable Anonymity in DRM

From our discussions in Section 2.1 we have seen how consumer privacy is often neglected by DRM current systems. We have a vision (Section 2.2) in which we hope to succeed in providing Alice with the anonymous DRM service that she desires; whilst ensuring that, under exceptional circumstances (*e.g.* if she acts with malicious intent), she can be held accountable. In this section we discuss and critique DRM systems that do not neglect consumer privacy and attempt to provide consumers like Alice with conditional anonymity.

There are many existing systems available in the domain of digital rights management (*e.g.* Windows WMDRM [Mic10], Apple Fairplay [App10b], Sony OpenMG [Son10], and Open DRM [Ian00]); although few are prevalent in today's commercial content distribution services. We believe that this is a consequence of failing to preserve the consumers' privacy. Consumers like Alice are unwilling to use services in which their privacy is not preserved. Most DRM services concentrate on the legality of providing copyright enforcement. Whilst we believe that this is vitally important, we also believe that it is equally important to provide

consumer privacy-preservation[8]. In doing so, we should also ensure that under exceptional circumstances (*e.g.* if a consumer acts maliciously) a consumer can be held accountable for his actions.

Arguably, current DRM systems can be segregated into three categories:

1. device-based systems;

2. identity-based systems;

3. smart-card-based systems.

Device-based systems bind a content/licence to a particular playback device so that it can only be accessed on that particular device. Since Alice does not share an identity with her playback device, by concealing her association to it, her privacy can be preserved. If this link can only be calculated or reconstructed by a dedicated Trusted Third Party (TTP), then she can be held accountable under exceptional circumstances. However, by restricting her to a particular device, such systems can be considered as undesirable for consumers; they do not fit into the ubiquitous environment of our everyday lives. Consumers tend to own many playback devices: Personal Digital Assistant (PDA)[9], music players, personal computers, and DVD/Blu-ray players, *etc.*, for which they want to use to access digital content.

Identity-based systems, on the other hand, can be considered to be rather more flexible. Alice can access content on any compliant playback device, provided that her unique identifier is registered to do so; neatly fitting into the prevalent consumer ubiquitous environment. This however heavily relies on a unique

[8]Many have argued the importance of privacy-preservation in DRM systems; *e.g.* Kenny *et al.* [KK02], Feigenbaum at al. [FFSS02], Cohen *et al.* [Coh03], von Lohmann *et al.* [vL02], Blomqvist *et al.* [BFO05] and Cavoukian [Cav02].

[9]PDAs vary in their functionality but one common use is for accessing digital content. PDAs are often considered to incorporate mobile telephones, palmtops, music players, and tablet computers.

identifier; most commonly, the consumer's civil identity (as seen in Microsoft's WMDRM [Mic10], for example). By using a unique identifier, DRM entities are able to piece together transactions emanating from a single user, which prevents us from preserving Alice's privacy.

Smart-card-based systems attempt to fill the gap between the inflexibility of device-based systems and the openness of identity-based systems. Here, Alice is issued with a smart-card, that she can use on any compliant device, and it can also be used as her unique identifier. By concealing her link to her smart-card, we can preserve her privacy. This link can then be exposed by a TTP under exceptional circumstances, *e.g.* if she acts maliciously.

In the remainder of this section we discuss existing systems in both the smart-card-based and identity-based segregations.

2.3.1 Smart-Card-Based DRM Systems

In 2003 Conrado *et al.* [CKSJ03] proposed a smart-card-based DRM solution to protect consumers' privacy. Here, Alice is supplied with an anonymous smart-card, which contains a unique key pair[10]. By using her smart-card during identification, authorisation and authentication procedures, Alice is able to conceal her identity. Therefore, by issuing a licence that's bound to a consumer's smart-card (by using IETF RFC[11] X.509 attribute certificates [Far02], for example), the licence broker issues a licence without knowing the identity of the consumer it is issued to. To prevent the licence broker from linking content/licences to a particular smart-card (and thus a single consumer), Alice supplies the licence

[10]Throughout this thesis when we refer to a *key pair* we mean a *pair of cryptographic public/secret keys*. We assume that the secret key is kept secret.

[11]A Request For Comments (RFC) is a memorandum published by the Internet Engineering Task Force (IETF) [IET10] describing a particular avenue of research with the request for peers to publish discourse in the form of an RFC.

broker with a *hash* of the smart-card's public key.

A hash function (*e.g.* Message Digest Algorithm, version 5 (MD5) [Riv92]) is a mathematical function that converts a string into a *hash value*. The hashed value bears no resemblance to the original value; it is considered [Knu98] to be computationally difficult to reverse a hash function to find the original value from the hash value.

By requesting a licence that's bound to a hash value instead of a public key, Alice is able to hide her public key from the issuer. We must remember here, that when given a hash value, it is computationally difficult to calculate the input value. So Alice is able to acquire a licence without revealing her public key. Under such a setting however, it is important to note that, since Alice's public key is public, anyone (*e.g.* Eve) is able to use the hash value to acquire a licence. It is easy for Eve to hash Alice's public key and use it to acquire a licence. Eve would however, be unable to use the licence, as she would be unable to perform the two-stage authentication procedure a verifier requires to be performed before it is assured that she is indeed the owner.

For the first stage of authentication, Alice supplies her public key to the verifier (*e.g.* her playback device when she is requesting access to a content). The verifier is able to prove that this is the correct public key by hashing it, and then checking the computed hash value against the stored hashed value in the licence. Only by supplying her public key here, does Alice reveal the link between her licence and her public key. Other untrusted entities that are not presented with this link cannot learn it; the link can thus remain hidden. This first stage authentication procedure however, uses publicly available information. So, to prevent Eve from maliciously acquiring a licence, the second stage authentication procedure must be executed.

For the second stage of authentication, Alice provides knowledge the corresponding secret key. We know that Alice is the only holder of her secret key, so by proving she knows it, she is able to prove that she owns the corresponding public key and thus, the licence. Provided that Alice keeps her secret key secret, Eve is unable to execute the second stage of authentication. Eve is thus unable to use Alice's licence. By executing both stages of the authentication procedure, Alice is able to demonstrate her licence ownership on two levels. Firstly by proving she knows the input to the hash function to generate the hash value stated in the certificate; and secondly, by proving she knows the corresponding secret key.

However, purely using a hash value of the public key may not be sufficient. We know that, given a fixed input, the hash function will always generate a fixed output. This would prevent Alice from using the same public key for multiple sessions if she wanted to preserve her privacy. So, to further secure her privacy and hide the link between all her sessions, Alice can concatenate a random salt value[12] to her public key before she hashes it. Using a different salt value for each licence will result in a different, unlinkable hash value. Now to prove licence ownership, Alice must perform a three-stage proof. In addition to the two stages mentioned above, Alice must also provide the salt value used during the execution of the hash function. By providing all three proofs, Alice can prove licence ownership.

In 2007, Sun *et al.* [SHC07] extended Conrado *et al.*'s system so that the content is encrypted before it is sent to Alice. Under the original system, the role of the playback device was to grant or deny access to the content; non-compliant devices could also provide access since the content was not encrypted. Under the system proposed by Sun *et al.*, compliant devices decrypt the content to grant access; non-compliant devices cannot provide access to the content since they do

[12]A salt value comprises a random set of bits used as a secondary input value to a cryptographic function. Its primary function is to complicate the association between the inputted plaintext and outputted ciphertext.

not hold the required decryption key. Only playback devices used by consumers who own a licence that contains the required decryption key, are able to decrypt the content. Here, the licence broker encrypts the decryption key using a random symmetric key computed and issued by the software embedded in the smart-card; therefore hiding it from the consumer. The symmetric key is hidden from the consumer as the smart-card encrypts it using the playback device's public key. Only the playback device is able to learn the symmetric key; Alice learns neither the symmetric nor decryption key.

Even with the amendments made by Sun *et al.*, the system failed to provide any form of accountability. For accountability to be realised, an authority must be able to identify a malicious individual. Now as consumers only reveal their hashed public key to the licence broker, it is unable to place the corresponding public key on a blacklist of unacceptable keys. Moreover, as all smart-cards are issued anonymously, the holders cannot be reprimanded.

We have learnt from our discussions in Section 2.2, how anonymity without accountability can be misused by malicious consumers. So, in 2004, Conrado *et al.* [CPJ04] introduced a smart-card-based DRM system that provides its consumers with a form of accountable anonymity. Here, unlike that of Conrado *et al.* [CKSJ03], to acquire a licence, Alice must first reveal her smart-card's public key to the licence broker. Now malicious persons can have their keys placed on a blacklist of unacceptable keys and they can then be subsequently refused a licence.

In revealing her public key however, Alice provides the licence broker with a unique identifier to associate the issued licences to. Continual use of the same public key during licence acquisitions provides the foundation for the licence broker to build comprehensive dossiers on each consumer it deals with. A system with a single licence broker would allow a single entity (*i.e* the licence broker) to

monitor all consumers' activities; even though the licence broker does not learn the consumers' identities, it can still monitor their respective activities. The licence broker would be able to link the licences it issued to a public key. As each licence is specific for a single piece of content, the licence broker would be able to use this knowledge of a consumer's public key to link it to the content. The licence broker could then start to infer shopping habits to build dossiers of the consumer's tastes and interests.

The licence itself is encrypted by the licence broker using the smart-card's public key. This prevents Alice or Eve from learning the contents of the licence. The licence contains the decryption key required to unlock the secured container; revealing this could allow Alice to access the content irrespective of her granted access rights — which are also contained within the licence. The playback device co-ordinates with the smart-card to decrypt the content without Alice's intervention. It is assumed that the playback device is compliant so that the decryption key can remain hidden from Alice.

Now although the playback device is assumed to be compliant, Conrado *et al.* provide a mechanism to protect Alice's privacy from it. To prevent the playback device from monitoring Alice's activities, she is issued with a random identifier; we call this a *pseudonym*[13]. The pseudonym has no bearing to Alice's smart-card or indeed her civil identity. It is issued by a dedicated TTP entrusted with maintaining the blacklist of unacceptable public keys. The TTP only issues pseudonyms to consumers who use a public key during the authentication procedure that is not listed in the blacklist. The TTP signs[14] the pseudonym to prove its authenticity and, to prevent Eve from learning Alice's relationship to the pseudonym, the TTP also encrypts the pseudonym using the smart-card's

[13]Pseudonymity is discussed in detail in Section 3.1.
[14]Digital signatures are discussed in Section 3.2.

public key. It is only Alice, whose smart-card contains the required secret key, who can acquire the pseudonym.

Now, by identifying herself to the playback device using her pseudonym, Alice is able to hide her public key. Conrado *et al.* argue that it is important that Alice is able to hide her public key from the playback device. If Alice were to use the same unique identifier for every piece of content she accesses, then the playback device would be able to link her accesses together. From the contents accessed, the playback device could infer Alice's interests, compromising her privacy. By using different unlinkable pseudonyms here instead, she prevents a malicious or compromised playback device from storing or transmitting privacy compromising information. The pseudonym itself is signed to prove that the pseudonym is genuine.

It must be considered however, that a single playback device is typically owned and used by a single consumer. Even if a consumer used different, unlinkable pseudonyms, the playback device could predict that all content accesses emanated from a single consumer. Using a pseudonym during authentication procedures, in place of a public key, must therefore be questioned. Every time Alice requires access to a different piece of content, she must first acquire a new pseudonym from the TTP. The continual communication between the TTP and consumers could result in a system bottleneck and added run-time costs. Moreover, it does not cater for situations in which a consumer acquires a pseudonym *before* her public key is revoked. To explain this point clearly, let's consider an example.

Imagine that Alice acquires a collection of pseudonyms and licences. Now if Alice has her public key revoked (after *e.g.* being found acting maliciously) and placed on the blacklist, she is still able to use her licences with her pseudonyms to access content. To provide full accountability, after being caught with malicious intent, Alice should be prevented from continuing from accessing contents. Since

Alice conceals her smart-card's public key from the playback device, it is unable to check the key's status (*i.e.* its presence of absence on the blacklist). Alice is therefore able to continue to access the content, even after her smart-card's public key has been revoked. To counter this, pseudonyms could contain validity periods, but Alice could still access content until the invalidity period is reached.

If each consumer is issued with a smart-card anonymously, from a set of analogous cards, an identity cannot be linked to a smart-card. Conrado *et al.* [CPJ04] suggest that each card should be protected by a Personal Identification Number (PIN) (set during activation by the consumer) so that only the designated holder can use the smart-card. From a content owner's perspective, this system only provides a form of weak or primitive accountability as a malicious consumer's civil identity cannot be traced. Even if a consumer is deemed to be acting maliciously, there are no procedures in place to reveal the consumer's civil identity. The consumer's smart-card can be revoked by placing its public key on the blacklist — to prevent him from acquiring more licences — but the consumer is not prevented from accessing content immediately; it is only when the pseudonym expires that access is refused. By placing its public key on a blacklist, his smart-card can be reduced to be effectively useless. The money he spent on the card is subsequently wasted, but the consumer himself cannot be reprimanded.

Smart-card-based DRM systems suffer from an inherent inflexibility. Any device the consumer wishes to access the content on must include a smart-card reader. This exceptional hardware requirement would add a considerable cost to consumers who must now only purchase devices supporting such a requirement — or upgrade existing devices. Assuming the smart-card is of similar dimensions to that of a regular smart-card (*e.g.* a credit or debit card), it also eliminates small content accessing devices such as the iPod shuffle [App10a], for example. This

would obviously not be a particularly desirable DRM service from the consumers' perspective.

2.3.2 Identity-Based DRM Systems

Although we have seen smart-card-based systems that provide forms of accountable anonymity, the overriding deficiencies include the intrusive nature of their hardware requirements. Moreover, the smart-card-based systems we have discussed above, fail to propose a procedure in which a malicious consumer can have his identity traced. Conrado *et al.* [CPJ04] describe a method in which a malicious consumer's/smart-card's public key can be revoked; but not a method in which his civil identity can be revealed to hold him accountable for his actions. Moreover, since the consumer is able to acquire multiple public-keys/smart-cards anonymously, the consumer is able to continue to use the system despite revocation. After having his smart-card's public key revoked, a consumer can just acquire a new smart-card.

In 2008, Feng *et al.* [FZ08] proposed a DRM system to protect Alice's privacy by hiding the link between her licence payment and acquisition without the necessity of a smart-card. Here, instead of binding a key pair to Alice's smart-card, it is bound to Alice herself[15], using, for example, a public key infrastructure[16]. So, instead of using a smart-card to authenticate herself with the playback device, she uses her knowledge of her secret key. It is important to note that Alice's playback device is assumed to be compliant so it will not compromise her identity.

[15]Feng *et al.* highlight that their system was designed with the flexibility in which the key pair could be bound to a playback device or a consumer (*i.e.* it could either be device-based or identity-based), depending on the application needs. In our discussion, we shall assume it is bound to the consumer and that it is identity-based. Although it is easy to abstract between the two.

[16]Generally speaking, a Public Key Infrastructure (PKI) is a system that binds public keys to individuals. We discuss one particular method to bind a key to the holder's civil identity in Section 3.3 when we discuss *digital identity certificates*

For Alice to playback a particular piece of content, she must show her corresponding licence to the playback device. The licence expresses the access rights that Alice has been granted for a particular content. For every piece of content Alice wishes to access, she must first acquire a licence from a licence broker. The licence broker will only issue licences after a correct payment has been made. To prevent the licence broker from linking the payment to the licence, Alice makes her payment to a payment gateway; and in return, she receives an anonymous receipt of payment. Feng *et al.* [FZ08] however, did not discuss how this receipt is acquired, or the format it takes; it is just assumed that such a receipt of payment can be acquired. By presenting this receipt to the licence broker, Alice can prove that she has made the correct payment. After verifying that Alice has indeed paid, the licence broker issues Alice with her licence.

For the playback device to accept Alice's licence, it must be both content-specific (so that Alice cannot *e.g.*, use the same licence for every piece of content) and consumer-specific (so that Alice cannot *e.g.*, share her licence with other consumers). To ensure that it is consumer-specific, the licence broker encrypts it with the Alice's public key. To ensure it is content-specific, Alice must supply the licence broker with the content's identifier. This would however, allow the licence broker to link Alice's public key to the content she is accessing — providing the foundations for dossier building. To prevent such occurrences, Alice utilises a *blind signature* scheme.

Blind signatures are a security primitive introduced by Chaum [Cha83] where the message signer and message owner are different entities. They are discussed in detail in Section 3.2. Here however, it is sufficient to know that blind signature schemes provide the ability for a user to acquire a signature on a message without revealing either the message content or the resultant signature to the signer.

So, by using a blind signature scheme, Alice is able to *blind* the content

identifier to prevent the licence broker from learning it; the licence broker only learns Alice's public key, and not the content she wishes to access. However, by blinding the content identifier, the licence broker cannot verify that Alice has paid the correct amount. To overcome this, contents are partitioned into groups dependent on their monetary value. So, by supplying the group identifier from which the content resides with her receipt, Alice can prove to the licence broker that she has indeed paid for the licence. As long as it can be proved that Alice made the correct payment, she need not reveal the content identifier — nor indeed her identity. By using large groups, Alice is able to keep her content access requests secret. From the supplied group identifier, the amount required for the licence can be learnt, and thus the payment can still be verified without knowing for which content the payment was for. The licence broker cannot therefore, associate the consumer's public key to the content she accesses. The licence broker can issue licences bound to the blinded content identifier so that Alice can, after receiving the licence, *unblind* the content identifier so that the licence becomes content-specific for the content she wishes to access.

Unlike our discussion in Section 2.1, the content decryption key is not contained within the licence. From the details contained within the licence, Alice is able to generate it. The content decryption key can be generated from the signed content identifier and the group the content resides; both of which are stored within the licence itself. It is important to note that, if Alice were to supply a different group to the group that the content actually resides during payment (to *e.g.* be charged a cheaper price), then the playback device would be unable to compute the correct decryption key. Alice would thus, be unable to access the content. If Alice supplies the correct information, her playback device is able to compute and then use the required decryption key. The playback device is then able to grant Alice access to the content in accordance to the access rights stored

in the licence.

Feng *et al.* describe an anonymous licence acquisition protocol. From our discussions in Section 2.1, we have seen that licence acquisition is a single transaction within a DRM system; DRM systems require an array of transactions for their complete execution. Feng *et al.* do provide a form of anonymity, but one could not provide a complete anonymous but accountable DRM system using this protocol alone; although it could be utilised as part of a larger system. Feng *et al.* for example, did not discuss how to achieve anonymous payment, or how to trace the identity of a malicious consumer. From our discussions in Section 2.2, we know that providing anonymity without accountability can be dangerous.

Moreover, if this protocol were to be applied to an identity-based DRM service, then once Alice decrypts the acquired licence, she is able to distribute it illegally; either freely to her associates, or at a reduced price to undercut the genuine licence broker. Anyone is able to generate the decryption key from the licence and thus access the content; although Alice couldn't offer different access rights to the ones she has been offered, but this is irrelevant if *non-compliant* playback devices are used. Once the content has been decrypted, Alice is not forced to use a *compliant* playback device. Conversely, if this protocol is applied to a device-based DRM service it would inherit the inflexibilities that device-based system present.

2.4 Requirements for an Accountable Anonymity Service in DRM

To provide Alice with the privacy preservation she desires, we need to provide her with anonymity. In Section 2.3 we discussed two types of DRM systems:

smart-card-based and identity-based; both of which displayed certain degrees of anonymity. We saw with Conrado *et al.* [CKSJ03] a system that supported the strongest anonymity, but little accountability. We know that accountability is important for commercial services; particularly for anonymity services. Conrado *et al.* [CPJ04] support a form of weak accountability as it doesn't allow an authority to trace a malicious consumer's identity. Although Feng *et al.* [FZ08] provided a more flexible solution, without the necessity of smart-cards, the accountable anonymity offered is weak. None of the systems we discussed so far provide satisfactory accountable anonymity and, at the same time, suffer from additional insecurities.

The discussed systems do however protect the linkability between a single consumer's multiple licence acquisitions. If a single DRM entity could link Alice's licences together, then from the corresponding contents, the DRM entity could execute a form of dossier building so that her shopping habits can be learnt. It is important however that, after the detection of a malicious consumer, all her transactions are linked together and her anonymity revoked; none of the DRM systems we discussed provide a mechanism to do so.

By using a system such as that proposed by Conrado *et al.* [CPJ04], we believe that consumers' privacy would be preserved to an extent that would satisfy consumers. However, from the content owners' perspective, the lack of strong accountability mechanisms would make it a poor choice. For an accountable anonymity service in DRM to be realised, both content consumers and content owners must agree on a set of rules; there must be a threshold in which they can agree. Undoubtedly, consumers are likely to want total anonymity so that their privacy cannot be compromised; whilst owners are likely to want total openness so that malicious consumers can be easily held accountable. Neither however, can come to fruition. There must be a middle-ground in which both parties

can be happy. We believe that a threshold can be established in a DRM service where a consumer's privacy is conditionally preserved; although under exceptional circumstances (*e.g.* after the detection of malicious intent) a consumer's identity can be revealed. We believe that such a system can exist; although a set of strong requirements must be agreed upon and enforced. We define these below:

R1 **Consumer identity anonymity**: a consumer should be able to acquire some digital content(s) anonymously, provided he adheres to the corresponding rights and regulations.

R2 **Session accountability**: under exceptional circumstances, if a consumer violates any rights or regulations, an authority should be able to determine his identity from the information of the session performed.

R3 **Consumer session unlinkability**: a consumer's constituent sessions should not be linked to his identity; only when the consumer is deemed to be acting illegitimately, the session unlinkability is revoked.

R4 **Multiple session unlinkability**: a consumer's constituent sessions should not be linkable to each other; only when the consumer is deemed to be acting illegitimately, an authority is able to link them together.

We use this '*R-notation*' throughout the remainder of this thesis so we can neatly reference our requirements when necessary.

Chapter 3

Existing Solutions for Achieving Anonymity

According to the International Organisation for Standardisation (ISO), anonymity ensures that an individual may use a resource or service without disclosing his identity [ISO05]. However, this alone may not be sufficient for providing complete anonymity.

Imagine a voting system consisting of ten voters. Each voter wants to keep his vote secret so he refrains from disclosing his identity by, for example, writing his name on his vote. Each voter writes down his vote on a piece of paper, and places it in a locked box. Once all votes are collected, the box can be unlocked and emptied with the results published. An observing adversary would not be able to learn who voted for what. Even if he watched all the voters place their votes in the box, he could not link a specific vote to a specific individual.

Now imagine that the adversary had a key to the box. After the first voter placed his vote in the box, the adversary could unlock the box and find out how

he voted. By watching the voter put the vote in the box, the adversary knows who voted; and by what is written on the piece of paper, he knows how he voted. He could repeat this act after every vote to learn how each voter voted. By not writing his name on his vote, a voter may think his vote is anonymous. This simple example however, proves the contrary. To effectively nullify this security loophole, the system could employ a TTP.

A TTP could stand next to the box and make sure it remained locked, for example. If he saw anyone open the box before the voting had finished, the TTP could call the relevant authorities to hold the responsible individual accountable. Since every voter trusts the TTP, each voter has the confidence that their vote will remain hidden.

With the addition of a TTP, the best an adversary could do is to *predict the probability* of how an individual voted after learning the outcome of the vote. A published result always leads to the possibility of an adversary calculating the probability an individual voted in a certain way. A unanimous result would inherently reveal how every voter voted.

A more relevant anonymity definition has been devised by Pfitzmann *et al.* [PH08]: anonymity is the state of being unidentifiable within a set of all possible subjects. A voting system for a million voters would typically offer more privacy for its voters than a voting system for ten voters. The more voters there are, the more anonymity the voters typically experience.

There are many other applications of anonymity; each of which utilises a range of anonymity primitives, protocols, and systems. We discuss the most applicable to our line of research in the remainder of this section.

3.1 Pseudonymity

Anonymity is an important concept, as we have discussed above. It provides the key for a privacy-protecting DRM service. One method of achieving a form of anonymity is to issue each consumer with a random identifier, which we call a *pseudonym*. The process of identifying and individual by a pseudonym is called *pseudonymity*.

In this section we discuss the concept of pseudonymity, explaining its importance in an anonymous but accountable environment. We then discuss a range of ideas, protocols, and systems proving pseudonymity, critiquing them against our set of requirements for an accountable anonymity service in DRM (see Section 2.4).

In the real world, pseudonyms are often used as an alternative to an individual's civil name; typically holding no bearing to the holder's civil identity. An individual, say Alice, could use a pseudonym instead of her civil identity during identification procedures to protect her civil identity. Alice could then still identify herself, whilst keeping her identity secret. Continual use of her pseudonym could lead to the accumulation of a reputation.

Reputations can be an important notion for anonymity services as used by ebay [eba10], the online auction service. Here, after each transaction, the participants are encouraged to rate each other with positive, negative or neutral feedback [eba09]. Over time, a fair participant should accumulate many positive feedbacks and a good reputation; whereas an unfair participant should accumulate many negative feedbacks and a bad reputation. Reputable participants should subsequently find it easier to sell and buy items than disreputable ones. Interestingly, all members of the ebay community are only identified by a pseudonym, which typically holds no bearing to their civil identity. This nicely illustrates how unfair

or dishonest actions could hamper a pseudonym's reputation.

Pseudonyms are also used by individuals who participate in controversial work. It is not uncommon for actors and writers, for example, to credit their work to a pseudonym, distancing themselves from their controversial work. It allows them to obtain a reputation for their work (so people keep purchasing their work), without anyone being able to link it to their civil identity; allowing the author to continue his life without any controversy or confrontations.

We now look at ideas, procedures, protocols and systems that support the property of pseudonymity in the digital world.

3.1.1 Digital Pseudonyms and Return Addresses

In 1981, David Chaum [Cha81] proposed a mechanism to replicate pseudonyms in the digital world when he introduced *digital pseudonyms*. Here, a digital pseudonym is a public key used to verify signatures made by the anonymous holder of the corresponding secret key. Similarly to pseudonyms in the real world, Chaum's digital pseudonyms hold no bearing to the holder's civil identity, anyone is able to create a public key and subsequently use it as a pseudonym. The main purpose of Chaum's pseudonym is to allow the anonymous communication between two participants.

To initiate the anonymous communication protocol, Alice passes her newly created pseudonym (*i.e.* her public key) to a TTP. The TTP has two main roles: firstly to accept or decline the pseudonym; and secondly, to publish and maintain a roster of acceptable pseudonyms. The acceptance decision takes into consideration, for example, whether or not the pseudonym already appears on the roster, or whether or not it appears on a blacklist of unacceptable pseudonyms.

Let's consider an example in which Alice might be known to Bob by a pseudonym that appears on the roster of acceptable pseudonyms. Bob can verify this pseudonym against the roster. Its presence or absence, easily allows Bob to determine whether or not Alice's pseudonym has been verified. If Bob is convinced that Alice's pseudonym has been verified, he would then be able to identify Alice through this pseudonym. He would be unable to link her pseudonym to her civil identity but he is assured that all signed[1] messages, that can be verified with Alice's pseudonym, did indeed originate from Alice[2]. Alice is thus able to communicate with Bob, anonymously. In saying that however, some communication protocols (*e.g.* the Internet [ber92]) can provide the recipient of a message (*e.g.* Bob) with enough information to trace the sender's IP address, therefore compromising the anonymity provided by the pseudonym. Chaum [Cha81] suggests using an untraceable mail system to prevent such information from being made available to Bob.

Chaum's untraceable mail system uses a mix, or a proxy, that processes each item of mail before it is delivered in such a way that the original sender's address is hidden. We illustrate a simple mix system in Figure 3.1. If Alice wishes to send a message to Bob using untraceable mail, she must first encrypt the message using Bob's public key, hiding the content of the message from the mix. She then sends the resultant ciphertext (c) to the mix so that it can hide her address from Bob. Alice therefore encrypts this ciphertext and Bob's address, using the mix's public key; sending the resultant encrypted package to the mix. Upon receipt,

[1]We provide a detailed discussion of digital signatures in Section 3.2. It is sufficient to know here, that signatures can be constructed by a holder of a secret key and verified with the corresponding public key. Signatures can be used to provide a proof of a message's integrity and authenticity.

[2]Similarly to public key cryptography, we assume that Alice is the only individual in possession of her secret key. Therefore, she is the only individual able to sign messages with her secret key. The secret key is assumed to be kept secret by the holder; whilst Alice's ownership of the corresponding public key is assumed to be published.

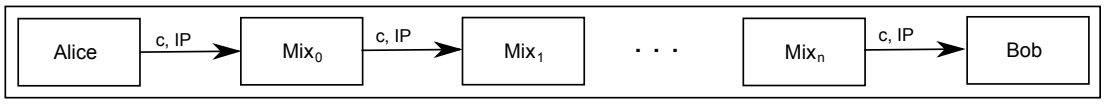

Figure 3.1: A simple mix system used by Alice to send ciphertext c to Bob. c is the encryption of a message using Bob's public key.

the mix decrypts the package to learn Bob's address and the encrypted message. The mix then sends the encrypted message to Bob's newly learnt address. In doing so, the mix masks Alice's address with its own so that the mix appears to be the sender, not Alice. All the information the communication protocol can provide Bob with will, at most, only point to the mix as the original sender; Alice's address cannot be learnt by Bob. Bob is still however, able to reply.

If Alice wants a reply from Bob, she can send him an untraceable return address using Chaum's untraceable mail system. To do so, Alice includes a ciphertext of her address, of which only the mix possess the required key to obtain her address, so Bob can send her an anonymous message through the mix.

When Bob receives a message for which Alice requests a reply, in addition to the message, Bob acquires Alice's encrypted address and the relevant mix's address. Bob can then send a message to Alice by sending the message to the mix along with Alice's encrypted address. Bob knows that the mix is able to learn Alice's address and relay the message on to her. Bob is able to do this, still unaware of Alice's address.

It is clear to see that using untraceable mail in such a way could satisfy the need for content consumers to remain anonymous in a DRM system. Let's say that Alice represents an anonymous consumer, and Bob represents a known DRM entity (*e.g.* a licence broker). Now it is conceivable that Alice could acquire content and licences anonymously, providing identity anonymity (R1). If Alice was found acting maliciously, an authority could co-ordinate with the mix to learn Alice's address to provide accountability (R2). In today's world however,

this may only be considered as weak accountability.

It is relatively easy for an Internet user to change his Internet Protocol (IP) address; so it does not necessarily provide a link to the user's identity. Moreover, Alice could easily use a different, unlinkable pseudonym for every session to provide her sessions with unlinkability (R4) but it would be difficult under conditions of malicious intent to identify (R1) such users from their sessions (R3). It would therefore be difficult to hold such users accountable (R2) for their malicious actions without significant modifications to the system.

Furthermore, the added run-time costs experienced by using public key cryptography must be considered. Public key cryptography is notoriously slow[3] [Dif88] and the constant use of this, particularly for large systems, could have an adverse affect on scalability and run-time efficiency. Adding this to the fact that, to validate a pseudonym, it must be checked against a roster of acceptable pseudonyms, leading to poor scalability. For a system encompassing many consumers and sessions, this could result in a system bottleneck.

On the other hand, with the introduction of mixes and an authority, such a system segregates the entities' duties. Here, one entity (*i.e.* a mix), or set of entities, is responsible for providing anonymity; whilst another, is responsible for revoking it. By using mixes here, one particular type of anonymity is provided: anonymous communication. We shall see from our discussions throughout this thesis that for an accountable anonymity service to be successfully built, many types of anonymity should be provided, including anonymous communication.

[3]Compared to their symmetric cryptosystem counterparts, public key cryptography typically use substantially larger keys and thus lead to more computationally costly implementations.

3.1.2 Pseudonyms and Credentials

Although being known by a single identifier, be it a civil identity or a pseudonym, allows reputation building, this might not be a desired property in all cases. It could provide a platform for consumer dossier building. Moreover, it becomes easy for organisations[4] to exchange information on any one individual. The fact that the individual's civil identity remains unknown, does not prevent organisations from exchanging information on that individual. The very fact that the individual is known by a single identifier, means he is prone to unlawful organisations sharing his information. Individual-specific information could be easily transferred between organisations without the individual's knowledge or consent.

Further still, the organisations may have different operational-scopes. For example, a hypermarket, where the individual holds a loyalty card, and a football club, where the individual holds a season ticket. Both organisations will hold different information, but both sets of information are bound to a common identifier. By colluding, it is therefore easy for these two different organisations to acquire individual-specific information outside their operational scope. An individual could use different pseudonyms with different organisations, but this becomes problematic if the individual requires the co-operation of multiple organisations: the organisations cannot be sure they are communicating about the same individual.

In the real world, an individual who is known by two different organisations can acquire a credential from one organisation to prove some fact, qualification or ability to another. In the UK, for example, the Driver and Vehicle Licensing Agency (DVLA) [dvl10] knows all UK-registered drivers. The DVLA issues a driver with a driver's licence (*i.e.* a credential) once he has passed his driving

[4]Here an organisation is a loose term used to describe an entity of knowledge. In our context one organisation may be the payment gateway, another maybe the licence broker.

test. However, before a driver is able to legally drive on public roads, he must first take out an insurance policy on his vehicle. An insurance company only issues a driver with insurance if they are certain that the driver actually holds a valid driver's licence. To do so, a driver could show his driver's licence to the insurance company, that could then, by contacting the DVLA, verify that he is both legal and able to drive. However, this heavily relies on the fact that both organisations know the same individual by the same identifier (*i.e.* his civil identity).

If the organisations knew the same individual by different, unlinkable identifiers (*e.g.* pseudonyms), then this form of credential would only prove to the insurance company that *someone* is legal and able to drive; not necessarily the individual in question. The two organisations need to know they are communicating about the same individual.

In 1985, Chaum [Cha85] extended his idea of digital pseudonyms [Cha81] to allow two organisations to communicate about a single individual, whilst ensuring neither are able to illicitly learn information outside their scope of operation. To do so, Chaum introduced a system in which a single individual holds many conditionally unlinkable pseudonyms; with each one known by only one organisation. Here, an individual is able to transfer information between two organisations; even though both organisations know him by a different pseudonym. To facilitate this, Chaum mimics real-world credentials when he introduced *digital credentials*.

A digital credential provides a certificate of authorised access — just like a driver's licence acts as a certificate of authorised access to the public roads. It proves that the holder is authorised for some access; *e.g.* this could be a statement proclaiming that the holder has passed an exam, or has acquired a qualification. To explain this clearly, let's use a DRM example.

Suppose Alice has acquired a secure container of the movie tetralogy, Die Hard, using the DRM mechanism discussed in Section 2.1. Now let's say that

before she can acquire a licence, she must pay the payment gateway the required amount *and then* prove to the licence broker that she has indeed paid for it. To protect her privacy, Alice identifies herself using a different pseudonym to the licence broker from the one used to identify herself to the payment gateway. The payment gateway could issue Alice with a receipt of payment but since the licence broker knows her by a different pseudonym, the receipt will be rejected as it won't appear to belong to her. The payment gateway could issue Alice with an anonymous receipt, but this could get stolen or copied. The receipt could contain a unique identifier, to allow only one use, but if the licence broker received many receipts of payment, all with the same identifier, the licence broker wouldn't be able to determine which receipt was the legitimate one. Alice must acquire a receipt of payment that proves she, and she alone, has paid the payment gateway the correct amount for the required licence.

Chaum's digital credential, formally defined by Chaum *et al.* [CE86], provides a method in which Alice can prove to the licence broker that she has paid for a licence, only if she has indeed paid. She is able to achieve this despite identifying herself with different, unlinkable pseudonyms. The digital credential mechanism can be explained with the use of an analogy.

Consider carbon-lined, windowed envelopes: they are regular envelopes that contain a transparent window, allowing one to look through and see a portion of the paper inside the envelope. Some of the paper is concealed by the envelope, whilst some is visible through the window. Any markings inscribed on the envelope itself produce a carbon copy on the paper inside.

Now, consider that Alice holds two pseudonyms, lets say x and y, one is known by the payment gateway (x), and the other by the licence broker (y). Alice writes down both x and y on to a piece of plain paper, and then places it in the envelope. Alice ensures that the only visible pseudonym through the

window is x, the pseudonym known by the payment gateway. After Alice makes the payment, the payment gateway signs the envelope, repeatedly, covering the whole envelope. The signature used is known by both the licence broker and the payment gateway to confirm that the pseudonym showing through the envelope's window has paid for the licence — only the payment gateway can produce such a signature.

Afterwards, Alice removes the piece of paper from the envelope. The paper now has a carbon copy of the payment gateway's signature, repeated over its whole area. She places it in a new windowed envelope. This time however, she ensures that the only pseudonym showing through the window is the one known by the licence broker, y. This portion however, also shows a carbon-copy of the payment gateway's signature. Alice can now show this envelope to the licence broker that knows her by the pseudonym showing, y, confirming that she has indeed paid for the licence — after verifying the shown signature to be the payment gateway's. Since x is obscured by the envelope and cannot be seen by any onlooker, the licence broker cannot determine the pseudonym that the payment gateway knows Alice by, and thus cannot link the two pseudonyms together.

In this analogy, the signature acts as the credential. It proves Alice is authorised to acquire a licence. This way, the licence broker knows that Alice has paid but does not know her payment details. Whereas the payment gateway knows Alice's payment details but doesn't know which licence she has acquired. In other words, both the payment gateway and the licence broker only acquire the information they require to function. Since they both know Alice under different pseudonyms, they cannot collude to acquire additional information about Alice — providing strong unlinkability (R3). The anonymity still holds (R1) but it is difficult to see how accountability (R2) could be achieved without any modifications. Chaum does not present a protocol or procedure to, under exceptional

circumstances when a user acts maliciously, link a pseudonym to the holder's civil identity.

Chaum formally defines his implementation [Cha85]; but we discuss an implementation [LRSW00] more relevant to our field of interest in Section 3.1.3. By using credentials in this manner one particular type of anonymity is provided: anonymous payment. We shall see from our discussions throughout this thesis that for an accountable anonymity service to be successfully built, many types of anonymity should be provided, including anonymous payment.

3.1.3 Pseudonymity with Accountability

We have seen that a system supporting pseudonymity can provide added security properties; but at the same time, can also provide inherent security flaws. We have already discussed how reputations could be utilised by organisations to monitor the activities of a pseudonym holder, but reputations could also be utilised by malicious holders. By supporting pseudonymity and reputations, some individuals may revel in the negative publicity of illicit or unlawful activities. Computer hackers, for example, often identify themselves with a pseudonym. The more publicity their malicious code or actions (often credited to their pseudonym) receives, the greater their 'hacking reputation' becomes.

Now in accordance with our requirements (Section 2.4), there mustn't be an apparent link between a consumer and his issued pseudonym (R1); possibly aiding such a notion. We could issue pseudonyms randomly for each session to achieve identity anonymity and eliminating reputation building, but we must also satisfy our need for accountability (R2). One possible solution could result from providing a link from the pseudonym to the holder's civil identity — as long as it's subtle and only known, learnt, or computed by a TTP. There must be a

procedure in which dishonest users can have their civil identity traced to prevent continued malicious intent; malicious users caught should have their pseudonyms revoked and be reprimanded accordingly. So, if a user identifies himself with a pseudonym, then there should be a procedure in place such that a TTP can trace a dishonest user's identity to hold him accountable if, for example, he is caught acting with malicious intent.

In order to provide anonymity with accountability using pseudonyms, Lysyanskaya *et al.* [LRSW00] built on Chaum's [CE86] digital pseudonym system to introduce a TTP with which all users must first register their civil identity and master key pair — only then, will a user be granted access to the service. As a result, Lysyanskaya *et al.* are able to issue a user with a pseudonym that's intrinsically linked to the user's identity, or more precisely, the user's master secret key. So, by verifying that he has the knowledge of the pseudonym's intrinsic secret key, a user is able to prove ownership of a pseudonym — and that he has not stolen or borrowed it. He is therefore the only user who is able to use the pseudonym during identification, authentication, and credential procedures. This has two important consequences:

1. **Conditional unlinkability**: all pseudonyms owned by a single user have an intrinsic relationship: the user's secret key. Provided that the user acts without any malice, then this intrinsic relationship is hidden and only known by the user himself. However, by acting with malice and attempting to use a single credential twice, a malicious consumer effectively presents the organisation with two equations — one for each attempt. By using simple simultaneous mathematics, the organisation is able to calculate the user's secret key. This can be cross-referenced by the TTP[5] to reveal the

[5]This is also achievable for a system operating with multiple TTPs from which a user selects a single TTP to reveal his identity to. Under such systems, presuming the user has proved to

user's civil identity. The user can then be held accountable for his actions.

2. **Non-repudiation**: upon being caught with malicious intent and attempting to use a single credential multiple times, a user cannot later reject such a claim if he has indeed been acting maliciously — he is the only user with the knowledge of the pseudonym's intrinsic secret key and therefore the only user capable of using the credential.

A credential here, in accordance to Chaum *et al.* [CE86], proclaims some fact. The master credential proclaims that the user has indeed registered his identity with the TTP. Other credentials may proclaim that the holder owns some privilege or entitlement, for example. The master pseudonym is generally used by the holder to prove to service providers that he is indeed a registered user so he can gain access to a service. To do this the user must reveal his master credential to the service provider, verifying it against the user's service pseudonym. This is achieved without revealing the pseudonym that the credential was originally issued for, or indeed, his civil identity.

A service pseudonym is the identifier the holder uses during identification procedures with the service provider that issued it. For every service provider the user wishes to interact with, he must construct a service pseudonym with that service provider. During this construction, the consumer must prove that he knows his secret key without revealing it, or indeed, the corresponding public key.

To prove that a user knows the secret key, Lysyanskaya *et al.* utilise Chaum's [CP92] interactive equality proof of discrete logarithms. A successful execution of this function proves that the same secret key was used during the construction

the organisation (or else it is unlikely the organisation would communicate with the holder) that he is indeed a registered user through the use of a credential, then the organisation knows with which TTP the user has registered his identity with.

of two different values. It is important to note that the user could fabricate the information he passes by *e.g.* supplying information that proves he knows the corresponding secret key of another, random public key. This however is futile. The user would then be unable to prove that he has revealed his identity to the TTP by revealing his master credential — the same secret key must be used to construct each pseudonym for the credential to be used successfully. Without using the master credential as proof of revealing his identity, the service provider would be unable to learn the user's identity if he were to act maliciously. So a user who is unable to prove that he has revealed his identity to a TTP is unlikely to be issued with a pseudonym or granted access by a service provider.

When the user holds a valid service pseudonym, he is able to show the master credential to the service provider to prove that he has revealed his identity to the TTP. The user can prove that the same secret key was used during the construction of the master credential as during the construction of the service pseudonym. This is achieved without revealing the service pseudonym for which the credential was originally issued for. The credential itself provides a non-interactive transcript of the proof used during the master pseudonym construction.

Taking the above into account, Lysyanskaya *et al.* provide an accountable pseudonym system in which DRM consumers like Alice could acquire pseudonyms anonymously; with each pseudonym being known by a different DRM entity. From hereon in, Alice is able to identify herself to a DRM entity with a single, anonymous unlinkable pseudonym — thus satisfying the need for anonymity (R1). To prevent her activities from being monitored, she could obtain and use a different pseudonym for every session she participates in. By using credentials issued by one entity, she is able to prove her relationship with other entities without revealing any linkable information (R3) — *i.e.* the pseudonyms known by the other entities.

However, if Alice were to act maliciously and use a single credential multiple times, she consequentially reveals linking information — *i.e.* information that allows the organisation to learn her identity. It therefore provides strong conditional unlinkability. Linking Alice's other pseudonyms together however, is more difficult (R4). Each credential is intrinsically linked to her secret key to prevent others from using it, but it would require the TTP (that she registered her secret key with) to manually check every pseudonym to see if Alice's secret key was used to construct it. This is rather inefficient, particularly in large systems.

We must also consider the additional overhead added to a communication channel using such a mechanism. By requiring Alice to acquire a new pseudonym for each transaction, we require her to participate in Chaum's interactive proof for each transaction. Every time she wishes to use multiple entities, she must acquire a new credential. This technique is considered to be computationally intensive [AT99] so continual execution could have negative affects on the system's run-time efficiency.

If Alice uses a pseudonym maliciously (*i.e.* uses the anonymity with malicious intent) then the pseudonym can be reported to the TTP that is able to trace her identity and hold her accountable (R2). The very fact that different DRM entities know the same user by different pseudonyms, prevent the DRM entities from colluding to acquire information outside their operational scope.

However, a malicious or compromised TTP could have devastating effects on the system's security. The TTP possesses the required knowledge to reveal the holder's identity of any pseudonym. The TTP therefore has all the knowledge to monitor a user's activities to build detailed, user-specific dossiers. A dishonest TTP could easily introduce malicious users into the system by issuing them with master credentials and then refusing to reveal their identities at a later stage.

A possibly greater concern however, could arise from the likelihood of a user

revealing his master key pair. The system proposed by Lysyanskaya *et al.* heavily relies on the user revealing his master key pair to a TTP. The TTP thus has the knowledge required to masquerade as a user, holding no accountability. Although this is unlikely since the TTP is inherently trusted, it this could become problematic if it became compromised. It could therefore prove to be too much of a risk for some users.

Lysyanskaya *et al.* provide a pseudonym system with good security properties, which could be incorporated into a DRM model. However, the over reliance on the honesty of a single TTP could provide added security concerns. It maybe that multiple TTPs (an idea we discuss in more detail in Section 3.4), or with the addition or amalgamation of other primitives could provide a more secure solution.

3.2 Digital Signatures

Handwritten signatures have become an important concept, used by many people in their everyday lives. They provide a form of authenticity and non-repudiation paramount to our society; whether it's authenticating a cheque of payment, a legal testament, or a certificate of qualification, signatures have become fundamental to our need for authentication. Handwritten signatures typically comprise the signatory's name, written in a unique, non-conformist manner; making it difficult for an adversary to forge, therefore proving its authenticity. Once issued, the signatory cannot later claim he did not indeed issue the signature; he is the only one able to issue it. A signature's authenticity can thus be verified by comparing the signature against a known original. If identical, the signature is proved to be authentic and processed accordingly; different signatures imply fraudulent behaviour, and are disregarded. In a DRM scenario, it is important

that Alice is able to prove the authenticity and integrity of her sessions. Proving the authenticity and integrity of a digital message (or session/transaction) is typically achieved using a *digital signature*.

Digital signatures are an important security primitive. Throughout this section we shall explain their importance and move to discuss how they could be used to provide accountable anonymity. We then critique existing digital signature systems against our set of requirements for an accountable anonymity service in DRM (as in Section 2.4).

Digital signatures, introduced by Deffie and Hellmen [DH76], replicate traditional handwritten signatures to authenticate a message. Unlike their handwritten counterparts, they are unique to the message signed. Consequentially, digital signatures provide the added property of integrity confirmation. If a signed message changes *enroute*, the issued signature will no longer be valid. The authenticity of the message cannot therefore be verified and thus the message is considered to be corrupt.

In 1978, Rivest *et al.* [RSA78] introduced the Rivest, Shamir and Adleman (RSA) algorithm for public key cryptography, providing one digital signature implementation. It has since become prevealanet in online security, now providing the foundations for VeriSign [ver10]: a leading online security organisation. On the whole, digital signatures work in a similar vein to its handwritten, paper-based equivalent.

A digital signature comprises a bitstring difficult to compute without the required knowledge (*i.e.* the required secret key); therefore making it difficult to forge. The secret key is related to the public key in such a way that it allows anyone who knows the public key to authenticate a signature constructed with the corresponding secret key. One particular property of digital signatures is message integrity. For the purpose of explanation, let's consider an example using RSA.

Alice wants to send message m to Bob. Bob however, will only accept messages that have their integrity proved. So, by using her key pair[6] ($< n, e >$ and $< n, d >$ respectively), Alice constructs a digital signature for integrity verification. For efficiency purposes, she first hashes m to get the hash value h. It is important to remember, from our discussions in Section 2.3, that a hash function generates a different hash value for a different input value. We can also utilise the fact that the hash value is typically considerably smaller than the inputted message to improve efficiency. So Alice calculates the signature s thus:

$$s = h^d \bmod n. \tag{3.1}$$

By attaching s to m, Bob is able to verify the integrity of the message; ensuring that it hasn't been modified *enroute*. To do so, Bob calculates:

$$h' = s^e \bmod n. \tag{3.2}$$

Now, to verify the integrity of m, Bob hashes it to get hash value h, and checks that

$$h \equiv h' \tag{3.3}$$

to be assured that the message has indeed not been modified *enroute*. Now, if Alice's public key were to be publicly bound to Alice, then this would also provide Bob with non-repudiation of the message's origin. Bob could then be further assured that, if m were to be a malicious message, then Alice can be identified and held accountable for her actions. One way to achieve this is by using digital certificates under a PKI.

[6]Throughout this thesis when we refer to a key pair we mean a pair of cryptographic public/secret keys. We assume that the secret key is kept secret.

A PKI (*e.g.* VeriSign) is a system that binds public keys to individuals. Generally speaking, it's a collection of hardware, software, policies and procedures needed to create, manage, store, distribute and revoke *digital certificates*. We discuss digital certificates in Section 3.3; here however, it is sufficient to know that digital certificates certify the ownership of a public key. Revoked digital certificates are placed on a blacklist of unacceptable certificates, know as a Certificate Revocation List (CRL). Public keys should be checked against the CRL *before* being used to authenticate a signature. A signature verified using a public key appearing in a certificate listed in the CRL, should be rejected.

Now by using a PKI, Bob is able to authenticate Alice's signature by using her publicised public key to verify the origin of the message. Only Alice, who holds the corresponding secret key, is able to sign the message so that it can be verified with her public key. Since Alice is bound to her public key, Bob is able to verify that the message was indeed sent by Alice. Furthermore, since Alice is the only holder of the secret key, Alice is unable to repudiate against claims that she signed a message that she did indeed sign.

Since Diffie and Hellmen introduced digital signatures, they have become an important cryptographic tool. They provide a mechanism for a sender, Alice, to authenticate her messages by issuing a signature of that message that no one else can compute. A signature using the same secret key on a different message would result with a different signature — although both can be verified using the same public key. Following on from this, digital signatures are not transferable. A signature is a function of a message, unique to that message. By signing one message, Alice can be assured that an adversary is unable to use this signature to prove the integrity or authenticity of another message.

We now proceed to discuss ways in which signatures are used in existing

settings to provide accountable anonymity; we critique them against our requirements for an accountable anonymity service in DRM (Section 2.4).

3.2.1 Blind Signatures

Non-repudiation is an important security property, particularly when proving the origin of a message is important. Our research however, revolves around accountable anonymity in DRM, which might seem to be a conflicting property to non-repudiation. We know from our previous discussion, that non-repudiation can be provided by amalgamating digital signatures with digital certificates. Alice can be prevented from denying that she signed a message that she did indeed sign with her secret key; and therefore publicly associating her to the message. This poses a problem for a service hoping to provide accountable anonymity for DRM consumers.

Furthermore, we also know that signatures can provide message integrity confirmation. Alice may understand that its necessary to provide session integrity checks and session non-repudiation (*e.g.* for malicious consumers to be held accountable) but it compromises her privacy. Accountability (R2) can however, be difficult to sustain without providing non-repudiation. Ideally, Alice should be able to remain anonymous whilst signing her sessions. To cater for such scenarios, Chaum [Cha83] developed and introduced an adaptation of digital signatures, *blind signatures* to provide unlinkability in signatures.

A blind signature scheme is an interactive, two-party digital signature scheme, where the message owner and signer are typically different entities. Using such a scheme, a DRM content consumer such as Alice is able to anonymously request a signature from a signer, say a DRM entity, without revealing either the message content, or the resultant signature. The blind signature bears no link to Alice's

identity, nor her other blind signature(s). To achieve this, Alice first disguises, or blinds, the message to be signed. She then requests a signature from the signer who signs the message using his secret key, and sends the resultant blind signature to Alice. Alice reverses the blinding procedure to reveal an unblinded message for which the signature still holds. To add further security, Alice could use an anonymous communication channel such as Chaum's mix system discussed in Section 3.1. Any other DRM entity can authenticate the blind signature with the signer's public key.

Now Alice, who wants to acquire the Die Hard tetralogy (see example in Chapter 2), could anonymously authenticate her DRM sessions by acquiring a blind signature from a trusted DRM entity. Neither the blind signature nor her pseudonym presents a link to her civil identity, and therefore anonymously (R1) non-repudiates her message. However, since it is not Alice who has signed the session or transaction, then it is not her who can be held accountable if she were to act maliciously. Although accountability is an important security property (R2), because of the strong anonymity provided by blind signatures, it cannot be realised with blind signatures alone.

3.2.2 Fair Blind Signatures

From our previous discussions, we know that providing anonymity without accountability can be potentially dangerous. Anonymity services should encompass some sort of accountability to prevent the anonymity from being exploited by those with malicious intent. An example of how the strong anonymity of blind signatures could be exploited is presented by Von Solms and Naccache [vSN92]. They use a genuine criminal ransom case in which the criminal was identified, to explain how the criminal could have remained anonymous if he had used blind

signatures. Examples such as this show that blind signatures alone should not be used by a DRM service providing *accountable* anonymity.

In 1993 Micali [Mic93] introduced the concept of *fairness* for strong cryptographic tools such as blind signatures. The idea is that the users of such tools remain anonymous, as long as they act legitimately; whilst those who misuse the anonymity for malicious purposes, can have their identity traced. Micali demonstrates how any public key cryptosystem can be transformed into a fair one. Stadler *et al.* [SPC95] adopted this concept, transforming blind signatures to *fair blind signatures*.

Fair blind signature schemes provide the anonymity of blind signatures but fairly, or conditionally. Similarly to blind signatures, a user here is able to obtain a signature on a message without revealing his identity to the signer. Anyone is able to authenticate this blind signature with the corresponding public key. The blindness of the signature still holds; in that the signer does not see the message's content or the resultant signature. However, with the introduction of a TTP, the blinded property can be revoked. In other words, the signer is able to learn the message he signed and indeed the resultant signature.

Before Alice can acquire a fair blind signature, she must acquire two signed pseudonyms from a TTP: one to be used during the singing procedure (P_1); whilst the other is used with the signature (P_2). The TTP stores the relationship between P_1 and P_2 so that it is able to link the signature construction to the signature verification under exceptional circumstances (*e.g.* after the detection of malicious activities). The fair blind signature scheme conditionally hides the message and the signature from the signer. With help from the TTP, this link can however, be revealed. To achieve this, P_1 and P_2 are related such that:

$$P_2 = P_1^n, \tag{3.4}$$

where n is a random, secret value known by the TTP.

To hide the link between signature construction and signature verification, Alice identifies herself with different pseudonyms: one with the signer (P_1); and another with the verifier (P_2). For the signature to hold for P_2, she must supply the signer with P_2 when requesting a fair blind signature on a message. To hide P_2, and thus the link between P_1 and P_2 from the signer, she blinds it before supplying it to the signer. Now, by supplying her blinded P_2, the signer is still able to issue Alice with a signature intrinsically linked to her P_2, without learning it. Alice is then able to unblind the message and show it to the verifier. So, by first unblinding and then identifying herself with P_2, Alice is able to prove the integrity and authenticity of the message. Through the blindness property, the signer is unable to link P_2 the resultant signature to P_1. Anyone can authenticate the signature with the signer's public key.

It is important to note here that Alice could fabricate P_2, and supply a blinded, random pseudonym (*e.g.* P_3) that has no relation to P_1. Under such circumstances, the signer is still able to issue Alice with a signature but a verifier would be unable to authenticate it. For it to successfully verify the signature, it must use Alice's pseudonym. Now if Alice were to identify herself with P_3, the verifier would be unable to authenticate the signature; she must identify herself with P_2 so that Equation 3.4 holds.

Now, if Alice were to use this message maliciously, the TTP is able to trace her P_1 directly from the attached signature. Remembering that the signature is intrinsically linked to her P_2, by simply cross-referencing P_2 the TTP can easily reveal the corresponding P_1. Revealing this to the signer effectively removes the blinding property. The signer can then link the message to the signature; the blinding property prevented the signer from achieving this previously. P_1 can then be placed on a blacklist of unacceptable pseudonyms so that any individual

subsequently identifying themselves with P_1 to a signer, will be rejected. Alice would therefore be unable to continue to use the system with her P_1; preventing her from acquiring further fair blind signatures. However, Alice's civil identity cannot be traced from her P_1. There is no use of a PKI or a digital certificate system (as we discussed previously) to bind it to an identity. She can therefore remain anonymous, and she cannot be held accountable for her actions.

In saying this however, Alice has no incentive to use the same P_2 more than once. To prevent her messages from being linked together, she cannot identify herself with the same P_2; this means that she must acquire a new pseudonym pair (*i.e.* P_1 and P_2). This could arguably make the blacklist of unacceptable pseudonyms obsolete. A simple extension could be applied in which a user must bind his P_1 to his civil identity. This could be achieved with the addition of another TTP for which users' must register their identities with.

Bringing this into the domain of DRM, by first acquiring two pseudonyms from a TTP, Alice could use a fair blind signature scheme to anonymously (R1) acquire blind signatures on her sessions. The TTP can then link her pseudonyms together (R3) to prevent her from acquiring additional fair blind signatures. This is however, only up to a point.

Neither of Alice's pseudonyms provide a link to her civil identity. Her identity cannot be traced from her pseudonym. So, under exceptional circumstances (*e.g.* if she acts maliciously), she cannot be held accountable. It can therefore be considered to provide weak accountability (R2). This could be strengthened with the addition of a registration-TTP for which users have to register their identity against their P_1, but this is not discussed by Stadler *et al.*.

Assuming that the TTP is not involved with the general running of such a DRM system, then the TTP could be called upon during registration and revocation procedures only. By using a different entity for registration and revocation,

the DRM processes can be separated from these procedures. The DRM entities would thus be unable to link a DRM sessions to a consumer's civil identity. However, as we mentioned above, for Alice to be protected by unlinkability, she would have to acquire a different pair of pseudonyms for every session. This would require the co-operation of the TTP for every user in every transaction; this is highly inefficient and could result in a system bottleneck.

3.2.3 Group Signatures

Group signatures, introduced by Chaum and van Heyst [CvH91] and formally defined by Bellare *et al.* [BMW03], provide a mechanism for Alice, as a group member, to anonymously sign a message on behalf of the group. Anyone is able to authenticate Alice's signature by using the group's public key without learning Alice's identity (R1) or her other signatures (R4). Providing that users are not partitioned into groups dependent on identifying information such as age, location or interests, the only divulged information by such a system is the identity of the member's group.

Now we know that both anonymity (R1) and unlinkability (R3) are important security properties. Alice, by using a group signature scheme, could prevent an adversary from learning any of her identifying information. Unlike blind signature schemes, if Alice were to act maliciously under the safety of anonymity and unlinkability, group signature schemes provide a mechanism in which Alice's identity could be traced to hold her accountable.

In a group signature scheme, each group is usually managed by a TTP. Ideally, these TTPs would not be involved in the regular workings of the system (*i.e.* during the signing and verification procedures). The main role of the TTPs is to facilitate member registration and member revocation to and from their respected

group. During the registration procedure, the TTP learns enough identifying information to uniquley identify the user. In return, the TTP issues the user with a proof that enables him to prove his group membership to another party, without revealing his identity. In simple systems (*e.g.* [CvH91]), this proof may be a user specific secret key whose corresponding public key is concatenated to the other members' public keys to form the group public key. Any message signed with a group member's secret key can then be verified with one of the keys in the group's public key. Only the managing TTP, that knows the correspondence between a member's key pair, is able to identify the original sender. Adding more members to or removing new members from the group obviously necessitates a change to the group's public key, making it not particularly scalable — obviously not desirable in environments with an inconstant consumer platform such as DRM.

More advanced systems have been proposed (*e.g.* Chen *et al.* ([CP95], Camenisch [Cam97] and Petersen *et al.* [Pet97]) but suffer a common set of deficiencies, *e.g.* either the public key must be changed as members are added or removed; the length of the signature or the size of the group public key is proportional to the size of the group; or the TTP must open *all* signatures to revoke a misbehaving member. In 1997, Camenisch and Stadler [CS97] did propose a group signature scheme that used a constant sized public key, but at a significant computational cost.

Camenisch and Stadler proposed a group signature system in which, to get a signature verified, the group member has to prove knowledge of double discrete logarithms and discrete logarithm roots. Both techniques are considered to be computationally intensive [AT99]. From a DRM perspective, with many sessions for which signatures must be verified, it could have an adverse affect on system run-time.

As a general rule, most group signature schemes suffer from a common set of weaknesses and limitations:

1. the size of the group public key is dependent upon the size of the group;
2. the length of a group signature is dependent upon the size of the group;
3. the addition of some group member(s) requires members' keys to be reissued;
4. the addition of some group member(s) requires a new group public key to be published;
5. revoking some group member(s) requires members' keys to be reissued;
6. revoking some group member(s) requires a new group public key to be published.

Considering that a DRM consumer-base is volatile in the sense that consumers are likely to enter and leave the system sporadically, these weaknesses and limitations render them inadequate.

There are many flavours of group signature schemes, although all have a common structure: a TTP managing a set of group members. Consequentially, all schemes cater for accountable (R2) anonymity (R1). Simple implementations suffer from their inflexibility of adding and removing members; whereas more complex implementations suffer from their inscalability of key size being dependent upon the group size, affecting efficiency.

3.3 Digital Certificates

In Section 3.2 we discussed how Alice could use digital signatures to prove the integrity of her sessions to a DRM entity. We further explained how, if Alice

was publicly bound to her public key, we could provide non-repudiation for the origin of a message. By utilising signatures in such a way, Alice could provide the DRM entity with confidence that, if she were to act maliciously, the very fact she signed her sessions ensures that she can be held accountable for her actions. The signature confirms that it was indeed her who participated in the session — *i.e.* prevents her from repudiating her participation. The public key used to verify the signature could then be placed on a blacklist of unacceptable keys, and subsequent signatures verified with this public key could be then rejected.

In Section 3.2 we suggested one particular method for binding Alice to her public key: digital certificates. Throughout this section we shall discuss digital certificates in more detail, explaining how Alice could indeed be bound to her public key. We then explain why this is important in a DRM environment, and more importantly, how this could be achieved anonymously but accountably. We look at existing ideas, procedures, protocols and systems attempting to provide accountable anonymity in digital certificates and critique them against our requirements for accountable anonymity in DRM (Section 2.4).

From our discussions in Section 3.2, we know that, on their own, digital signatures only provide the first level of accountability. They allow a DRM entity to link Alice's sessions to a public key. Signatures verified with the same public key can be subsequently linked together. On their own, signatures do not allow the DRM entity to learn Alice's civil identity. If Alice is free to generate her own key pair and use them without any TTP co-ordination, then she would be able to change her key pair whenever she so wishes (*e.g.* after being caught with malicious intent). There is no intrinsic link between Alice's identity and her key pair. Without such a link, she can effectively sign messages anonymously. To provide second level accountability, digital signatures must be certified.

In 1976, Diffie and Hellman [DH76] introduced *digital identity certificates* as

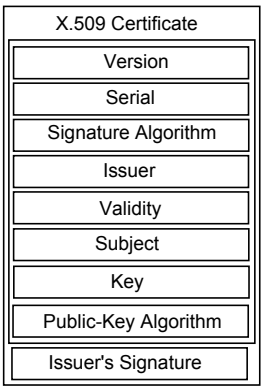

Figure 3.2: X.509 certificate structure.

a method to bind a user's key pair to his civil identity. A digital certificate works in a similar manner to that of its traditional, paper-based equivalent. Here, a certificate is typically a piece of paper that formally and officially affirms some fact. University graduates, for example, will often be presented with a certificate upon graduation, which proves that the holder has received the stated academic degree. The certificate's authenticity is confirmed with a, or a set of, handwritten signature(s) that can be easily verified by contacting the certificate issuer — *i.e.* in this case, the university and the signatories. A digital certificate mimics such a paper-based certificate.

A digital certificate is a digital construct used to formally and officially affirm some fact: usually the relationship between a civil identity (labelled *Subject* in Figure 3.2) and a public key (labelled *Key* in Figure 3.2). The certificate comprises a collection of fields, each one stating some information regarding the holder and the certificate itself. The IETF certificate standard, X.509 [FPS99], for example, is depicted in Figure 3.2.

The certificate issuer, a TTP known as a Certification Authority (CA), interactively confers with Alice to construct her digital certificate. This requires Alice to reveal her identity and her public key so that the CA can bind the two together in the certificate. Alice can then use her private key to construct a signature,

which she can subsequently prove her ownership, by showing her certificate.

The CA signs the certificate so that it can later be authenticated with the CA's public key. Keys that have been revoked are listed on a blacklist of unacceptable keys.

To prove her relationship to her public key, Alice can show her certificate whenever she signs a message. The recipient can prove the validity of the signature by using the Online Certificate Status Protocol [MAM+99], or simply verifying:

1. the supplied signature with the public key of the certificate;
2. the CA's signature of the certificate with the CA's public key;
3. that the certificate is within its validity period (labelled $Validity$ in Figure 3.2);
4. that the public key (labelled Key in Figure 3.2) has not been revoked and is not listed on a blacklist of unacceptable keys.

A PKI is a collection of hardware, software, policies and procedures needed to create, manage, store, distribute and revoke digital certificates.

The remainder of this section details how we can achieve the accountability that certificates provide whilst ensuring that the holder remains conditionally anonymous.

3.3.1 Anonymous Identity Certificates

In 2000, Zhang *et al.* [ZQM00] introduced a mechanism in which users are able to acquire *anonymous identity certificates*. An anonymous identity certificate is similar to that of an X.509 identity certificate (as discussed above). The main difference is that anonymous certificates contain fabricated information that only

the issuing CA is able to link to the holder's civil identity. From Figure 3.2, an anonymous certificate may contain a pseudonym instead of Alice's civil identity (in the *Subject* field) and a different public key (to the one publicly and officially bound to her). Other than that, the structure is similar. Alice is therefore able to use her anonymous certificate in a similar manner to an X.509 identity certificate — including the process of acquiring an anonymous digital certificate. Before she can acquire an anonymous certificate however, she must first acquire an X.509 identity certificate (like that defined above).

The anonymous certificate contains a new public key (as opposed to the one publicly and officially bound to her) and an associated pseudonym (see Section 3.1) in place of Alice's real identity. Only the issuing CA, which stores the link between Alice's civil identity and her pseudonym, is able to establish this link. The holder of an anonymous certificate is able to provide certified non-repudiation, just as an X.509 identity certificate does; although here, he's protected with conditional anonymity. Any signature that can be verified with the public key held in the anonymous certificate, can be credited to the pseudonym held in anonymous certificate.

Now, should a dispute arise, the CA is able to trace Alice's identity from the pseudonym supplied in the anonymous certificate, to hold her accountable for her actions. Since certified signatures can provide non-repudiation, Alice is not able to rebuke signing any messages she did indeed sign. Moreover, Alice is able to prove that she did not sign a message that she did indeed not sign.

To further complicate the association between her pseudonym and her identity, Alice could use an already acquired anonymous certificate during the process of acquiring another anonymous certificate. By using an anonymous certificate here, instead of an X.509 identity certificate, she reduces the likelihood that a compromised CA could lead to her identity being learnt. If all CAs are single,

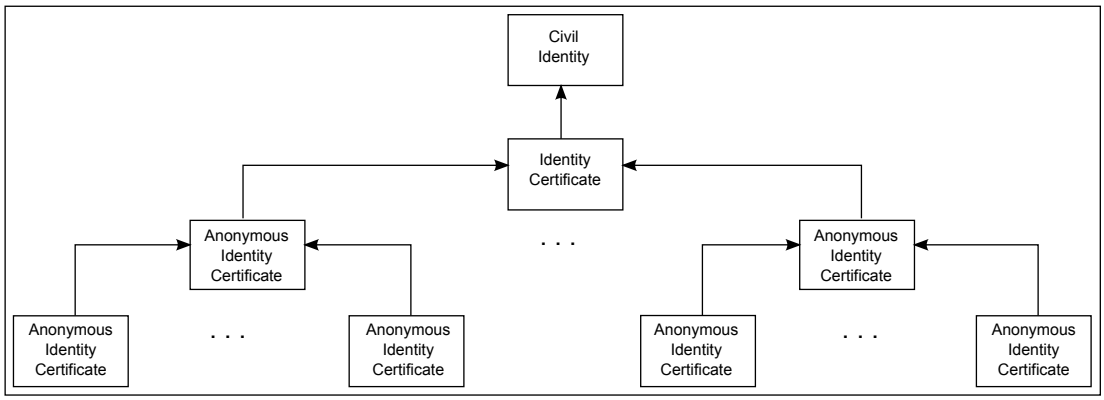

Figure 3.3: The tree-like structure of a consumer's anonymous certificates.

separate entities, then the likelihood of many becoming compromised is less likely, as opposed to a single CA. It follows that, the more times she repeats this, the more likely she is to keep her identity hidden. The more CAs she uses, the more confidence Alice will have that her identity is safe. It only takes one CA to remain honest for her identity to remain safe. By choosing a large number of CAs, she increases her chances of keeping her identity concealed. This could however have an adverse affect on the system's run-time efficiency.

To prevent a user from applying for anonymous certificates indefinitely, each anonymous certificate has an associated 'level'. For every issued anonymous certificate, the CA ensures that the certificate's level is one greater than the supplied certificate. One could imagine a tree-like structure of certificates (see Figure 3.3), where the root would be the X.509 identity certificate. The system could set a predefined limit to the depth of the tree, where all CAs reject requests for anonymous certificates whose level would exceed the declared maximum depth. This would restrict the depth of the tree but not the width.

To control the width of a certificate tree, the CA could issue certificates with expiration times. This would limit the number of certificates in any one level, not strictly by quantity, but quantity tends to be proportional to time (since it

takes time to acquire a certificate). There could however, be a situation in which the leaf certificate has not expired, whilst the root certificate (the X.509 identity certificate) has expired. This would make identity tracing particularly difficult, as the user may have changed his details (*i.e.* his identity) by then, *e.g.* where he lives. So, the expiration time of the newly constructed anonymous certificate should not exceed the expiration time of the one used during its construction.

Using the protocols defined by Zhang *et al.* [ZQM00], a trusted authority is able to trace a malicious user's identity from any correctly formed anonymous digital certificate. The authority then has the required knowledge to reprimand an anonymous malicious user for his misdemeanours.

Revoking an anonymous certificate is however, not trivial like revoking a single, unlinked, X.509 identity certificate. Due to the sequential nature of certificate issuing, to successfully revoke an anonymous certificate, simply revoking the certificate associated to the malicious activities is insufficient; all the user's certificates must be revoked to prevent the offender from continuing to use the system.

In 2005, Zhang *et al.* [ZSM05] extended their work to provide an anonymous certificate revocation procedure. Here, to revoke an anonymous certificate, a recursive procedure is initialised. Once the authorities have learnt the user's civil identity, as a side-effect, the root CA is discovered — the CA that issued the X.509 identity certificate. The root CA, that holds a reference to its children certificates, then notifies the corresponding CAs, and requests them to revoke its certificates. Each CA repeats this process, notifying its children and requesting certificate revocation. Since the leaf certificates have no children, the associated CAs revoke the certificates, and acknowledge the request from the CAs associated with the parent certificates. A certificate only gets revoked when it has no children certificates, or it has received an acknowledgement from all its children. This

revocation procedure is traversed down and up the tree until all certificates have been revoked.

So, Alice, who wishes to acquire the Die Hard tetralogy anonymously, could acquire a pseudonym and a corresponding anonymous certificate from a CA. Alice could further complicate her relationship with her pseudonym by using many CAs. Alice knows that she is able to use her pseudonym and the corresponding anonymous certificate to help her acquire the content anonymously (R1). She also knows that if she misbehaves, her identity could be traced to hold her accountable (R2). By using different pseudonyms for each session, Alice could ensure that her sessions are unlinkable (R3). The introduction of Zhang *et al.*'s [ZSM05] extension, allows all of Alice's sessions to be conditionally unlinkable from each other (R4). The run-time cost is managed with a declared maximum depth and certificate expiration times.

3.3.2 Anonymous Authorisation Certificates

According to Ellison [Ell99], an identity certificate can be considered to be like a passport: it identifies the holder, tends to last for a long time, and should not be trivial to obtain. An *authorisation certificate* is more like an entry visa: it is typically issued by a different authority, and does not tend to last for as long. As acquiring an entry visa typically requires one to present a passport, acquiring an entry visa can be a simpler process. Whilst Zhang *et al.* [ZQM00][ZSM05] focused their attention on anonymous authentication with identity certificates, Benjumea *et al.* [BLMT] focused their attention on anonymous authorisation with authorisation certificates.

Authorisation certificates are generally geared to extending an identity certificate to provide the holder with the rights or privileges needed to perform a

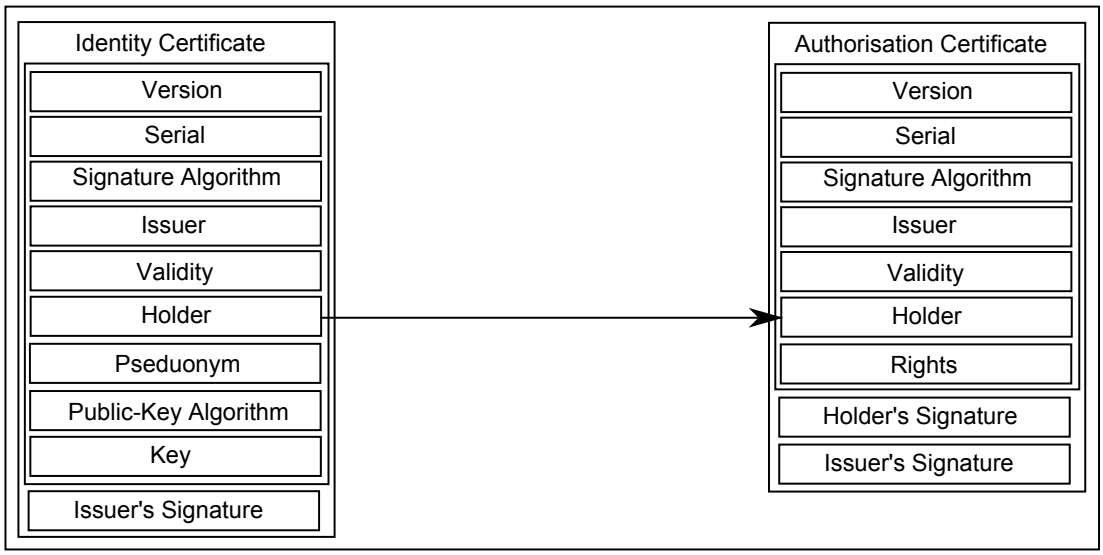

Figure 3.4: The relationship between an authorisation certificate and an identity certificate.

particular task. One particular implementation is illustrated in Figure 3.4. Here, the holder of the authorisation certificate corresponds to the individual's identity certificate so they can be linked. Authorisation certificates are interesting in our line of research as they allow one to transport authorisation information in distributed systems, such as DRM. They would be particularly interesting if they can be used whilst preserving the holder's privacy.

Benjumea *et al.* proposed a new system in which Alice could reveal her identity to one authority, and acquire her certificate from another[7]. Here Alice acquires a two-part pseudonym (P_1 and P_2) from a TTP (in accordance to fair blind signatures, discussed in Section 3.2). She uses P_1 to identify herself to an Authorisation Authority (AA); and P_2 to identify herself to a CA. She must then be granted authorisation from an AA before being issued with an authorisation certificate from a CA. To prove that she has been authorised by an AA she acquires a fair blind signature. Alice can then show this to a CA as proof of her authorisation.

[7]This could bee seen as a form of escrow; an idea we discuss further in Section 3.4

To acquire authorisation, Alice reveals her civil identity and P_1 to an AA that signs, blindly but fairly, her public key under her P_1. Now Alice is able to show the unblinded fair blind signature to prove that the AA has granted her authorisation. The signature however, was made under Alice's P_1. Showing this unmodified signature to a CA would allow it to collude with the AA to monitor Alice's activities. So she transforms the signature so that it is under her P_2 (in accordance to fair blind signatures). She must then construct a pseudonym certificate to certify P_2.

A pseudonym certificate here is similar to an X.509 identity certificate in that it binds a public key to a identity; but in this case, it's a pseudonym, not an identity. In addition to this, a pseudonym certificate also contains the condition in which the holder's identity will be revealed. Thus the main purposes of the certificate's are:

1. to prove that the holder knows the supplied public key's corresponding secret key;

2. to provide condition-acceptance non-repudiation;

3. to prove that a TTP is capable of revealing Alice's civil identity.

If the holder himself signs the certificate, then it proves that the owner has agreed to the stated condition and cannot, at a later date, deny it.

Now to acquire an authorisation certificate, Alice identifies herself with P_2 to a CA, and certifies P_2 with its corresponding pseudonym certificate. If the CA agrees with the condition within the certificate, it constructs an authorisation certificate. The authorisation certificate is similar in structure to a X.509 identity certificate discussed above. The authorisation certificate here however, contains a $Holder$ field. The $Holder$ field contains a hash of Alice's identity certificate. To

prove ownership of the authorisation certificate to a verifier, Alice must execute a two-stage authentication procedure.

Unlike the hash functions we discussed in Section 2.3, the hash function here requires an input hash key. As a result, only the holder(s) of the hash key is able to generate the hash stored in the *Holder* field. So, if this hash key remains hidden, the link between Alice's identity certificate and her authorisation certificate, can remain hidden. Alice is able to prove the certificates integrity and authenticity by supplying the hash key to the verifier.

For first stage authentication, Alice supplies her identity certificate and the hash key. By hashing her identity certificate with the hash key, a verifier is able to compare the resultant hash value against the value stored in the *Holder* field of the authorisation certificate. If they are the same, it implies that the certificates are indeed linked. In a similar way to the identity certificate, the authorisation certificate is signed by the issuer to prove its integrity and authenticity. For second stage authentication, Alice must prove that she holds the knowledge of the secret key corresponding to the public key stored in the identity certificate. Only when both stages of the authentication procedure are executed, will a verifier be assured that Alice is indeed the genuine holder.

By hiding the hash key, Alice is able to restrict unauthorised entities from linking her certificates together. Under a regular hashing scheme, anyone would be able to hash her identity certificate to link it to the authorisation certificate. However, only entities holding the hash key (issued at Alice's discretion) are able to link her certificates together.

Now, if Alice is deemed to be acting maliciously, then the issuing AA, that stores each consumer-specific hash key, can release it to a dedicated revocation authority. This authority can then learn the link between the two certificates, and thus Alice's identity. She can subsequently be reprimanded, and her public

key placed on a blacklist of unacceptable keys. If Alice attempts to use keys on the blacklist, requests for additional authorisation certificates can be rejected.

Whenever Alice requests authorisation, she first identifies herself with her P_2, certifying it with her pseudonym certificate. Then, to authorise herself, she supplies her authorisation certificate. The link between her pseudonym certificate and authorisation certificate is easy to verify. The verifier must just check that the hash value stored in the authorisation certificate can be constructed using the supplied pseudonym certificate and the hash key. A simple handshaking protocol can be executed to prove that the individual is the genuine holder and that he hasn't stolen or borrowed the certificate by proving the knowledge of the corresponding secret key.

Now, if Alice has been, *e.g.*, caught with malicious intent, her identity can be traced. To trace her identity from her P_2, both the AA and the TTP must co-ordinate. The AA knows the link between Alice's civil identity and her P_1; whilst the TTP knows the link between Alice's P_1 and P_2; together they know the link between Alice's P_2 and her identity. Alice can thus be held accountable for her malicious actions. It is important to note that it is not only one entity that is responsible for revoking anonymity. The revocation authority must work with the CA, the pseudonym issuing TTP, and the AA.

Allowing a user to acquire conditionally (R2) anonymous (R1) authorisation certificates is particularly interesting in our line of research. We could allow a consumer in a DRM system to have his authorisations modelled using authorisation certificates. The condition in which a user's identity is revealed could be expressed using a commercially available, a well defined REL such as MPEG [Rig03]. Using RELs would allow a user to transport his authorisations around a DRM environment without revealing his identity. His identity could however be learnt, if the condition was not adhered to, by a trusted revocation authority.

Benjumea *et al.* did not however, propose a method of certificate revocation. If we took a similar stance to that of Zhang *et al.* [ZSM05] (mentioned above) here, then if a user is to be found acting maliciously, all his authorisation certificates should be revoked (R4), not just the responsible one (R3). Benjumea *et al.* proposed a mechanism in which a user's identity can be revealed if certain conditions are met; learning, and subsequently revoking all the authorisation certificates a given user holds, could prove to be more difficult.

3.3.3 k-Times Anonymous Authentication

As anonymous certificates provide a means for Alice to anonymously authenticate herself or her sessions, they provide an attractive measure for anonymous authentication in a DRM environment. Using such a system, Alice could anonymously authenticate her access to a piece of content continually, as long as the certificate has not expired or been revoked. It may be however, that Alice only wishes to access the content one, two, three, or k-times. She may not be interested in accessing it continually; similar to the concept of a cinema theatre where an individual pays to see a movie once. As a result, Alice may be offered a reduced price, providing a more attractive market.

The notion of k-times anonymous authentication is particularly interesting to our field. One could imagine a situation in which a DRM consumer is granted access to a piece of content, provided that his number of accesses do not exceed k — when an additional fee is required before further accesses are granted. We have seen methods in which authorisation can be granted anonymously in group signatures (Section 3.2) and authorisation certificates (Section 3.3.2), but there is no easy way to restrict the *number* of authentications. We could revoke Alice's authorisation certificate after it has been used k-times; this would not only require

a large revocation list to be traversed during authentication procedure but also an authority to monitor the number of times the certificate has been used. In 2004, Teranishi *et al.* [TFS04] proposed an alternative.

Teranishi *et al.* proposed an anonymous authentication scheme in which a user remains anonymous, provided that he adheres to a k-limited authentication request. In other words, any user is limited to the number of times he can be authenticated to a predefined k. Any user who requests authentication more than k-times shall, on his $k+1$ attempt, consequentially reveal his identity. Any user requesting authentication at most k-times, keeps his identity concealed.

To partake in the system, Alice participates in an interactive, two-party procedure with a central-TTP to acquire a secret key, and a *partial* public key. This partial public key is, on its own, useless; it cannot be used to authenticate a signature, nor can it be used during the authentication system described here. The central-TTP only issues Alice with a partial public key here because, if she wants the full key, she must first publish her identity's association to this partial public key. Issuing Alice with the full public key would provide little motivation for her to adhere to this rule. So Alice, who reveals her identity during registration with an identification-TTP for example, adds her identity's association with her partial public key to a publicly verifiable identification list; maintained by the identification-TTP, for example, that authenticates it with a signature. After verifying that Alice has indeed added her identity's association with her partial public key to the identification list, the central-TTP issues Alice with the remaining elements of her public key so she now holds the full public key corresponding to her secret key.

Now, to authenticate herself to a service provider, Alice computes a *tag-signature*. A tag-signature is similar to a regular signature in that it is based on the holder's secret key but here it is also based on a *tag-base*. Each service

provider publishes the maximum number of accesses allowed for its service, k, and publishes k tag-bases from which the tag-signatures are constructed. For every authentication, Alice must select a different tag-base to construct her tag-signature on. A tag-signature computed using an unpublished tag-base will be rejected by the service provider. Now, if a user were to construct two tag-signatures using the same tag-base, the service provider is able to compute Alice's public key (using basic simultaneous equation mathematics). By cross-referencing her public key against the published identification list, Alice's identity can be traced.

Unlike the signature and certification schemes we have previously discussed, Teranishi *et al.* [TFS04] provide a mechanism in which Alice is restricted to only authenticate a message a predefined number of times. Under an anonymous signature scheme, she can issue a signature anonymously to prove a message's integrity and authenticity; but by using the system proposed by Teranishi *et al.*, she is restricted to the *number* of times she can authenticate her message without revealing her identity. By selecting a different tag-base for each authentication, she is able to keep her identity secret. However, by using a single tag-base on multiple occasions, her identity can be revealed. Unlike many systems we have discussed, the ability to trace an identity is not enforced by honesty; it is enforced by the mechanics of the system. By using tag-signatures, a user is able to conditionally conceal his public key (and thus identity) by using a different tag-signature for each authentication.

Unlike group signatures and the anonymous identity/authorisation certificates we have previously discussed, Teranishi *et al.* describe a system in which a compromised TTP or service provider cannot collude to trace Alice's identity, providing strong anonymity (R1). The most damage a compromised TTP could cause is limited to removing entries from the identification list to hide the identity of a user who requested authentication more than k-times; although in doing this,

the TTP would reveal the fact it is indeed compromised.

Further to Alice's authentication being conditionally unlinkable to her identity, her authentications are unlinkable from each other. In actual fact, linking Alice's authentications together could prove to be computationally hard. The only thing linking her authentications is her secret key; which is obviously unobtainable. The introduction of a TTP (generating key pairs) and a service provider (verifying authentications) prevents one entity from acquiring too much private information.

3.4 Identity Escrow

Escrow is often used to describe a process, procedure or situation in which a TTP is called upon to mediate in a payment transaction. In such a situation, the TTP typically holds the money whilst the buyer waits for the product to be delivered. A simple example can be found from a typical confectionery dispensing vending machine

A user normally inserts his coins into the vending machine with the intention of purchasing some confectionery. The coins are placed into an 'escrow area' so that if the user changes his mind, or hasn't got enough money, he can push a button to receive his money back; only when his confectionery has been dispensed, will his money be put into the machine's internal 'money bank'. *Identity escrow* is a mechanism used in a similar vein but in the context of identity.

Throughout this section we focus on identity escrow in the domain of DRM. We first explain its importance to a DRM system, expressing its strengths and weaknesses. We then discuss identity escrow systems appropriate to our field, critiquing them against our requirements for an accountable anonymity service in DRM.

Identity escrow, an idea proposed by Killian and Petrank [KP98], allows a user, say Alice, to authenticate herself without divulging her identity. Here, Alice need not supply her identity (or her pseudonym) to identify herself. Instead, she only supplies a guarantee that she has revealed her identity to a TTP. The TTP is not involved in regular identification schemes; it is only called upon when the anonymity must be revoked. This is known as first tier authentication: Alice supplies as little information as is strictly necessary for the transaction. A football supporter, for example, may just supply his entrance ticket in order to authenticate his presence at a football match. It would be unnecessary for him to supply his name and address whilst showing his ticket. First tier authentication is typically used by a user of a service to prove his authenticity to the service provider without revealing his identity, which we consider to be an important security property (R1).

For second tier authentication, Alice supplies a much more precise statement about her identity, which is used to uniquely identify her. Alice is only required to provide second tier authentication to the TTP, which may, under extraordinary circumstances (*e.g.* after the detection of malicious activity), be used to revoke her anonymity to hold her accountable (R2). If a single football ticket is used multiple times for the same match, the owner's identity should be learnt to hold him accountable for fraudulent use of the ticket.

Continuing from our example in Chapter 2, imagine that Alice, a DRM user, wants to access a DRM service to download the movie tetralogy, Die Hard. She is however reluctant to divulge her identity to the DRM service as she is wary of identity profiling. She does however understand that, to provide a successful service, dishonest users should have their identities revealed to be held accountable. She is therefore willing to divulge her identity to a TTP, as long as it is separate from the DRM processes, and is not controlled by an identity within the

DRM service. So Alice supplies the TTP with her identity and in return, after it verifies her identity, the TTP provides Alice with an *escrow certificate*.

An escrow certificate is similar to a X.503 identity certificate in most respects (certificates are covered in Section 3.3); it certifies some facts. However, unlike digital certificates that certify a public key against an identity, an escrow certificate certifies a public key against a guarantee that a TTP can reveal the holder's identity. By showing the certificate, the holder does not reveal his identity; he only reveals information that would allow an authorised TTP to reveal the holder's identity. Each escrow certificate contains unique information that the TTP, and only the TTP, is able to link to the holder's identity.

So, after receiving her escrow certificate, Alice could use it to anonymously apply for access to a DRM service provider. Alice could use her escrow certificate to prove that she has already divulged her identity to the TTP. It therefore gives the DRM service provider confidence that, in the case she acts illegitimately, her identity can be learnt (with help from the TTP), so she can be held accountable for her actions. Alice does not however, supply her certificate to the service provider. Instead, she uses a *cut-and-choose protocol* to reveal information that proves that she does indeed own a genuine escrow certificate.

A cut-and-choose protocol, first introduced by Rabin [Rab79], is based on the dividing up of a cake. It is based on the premise that the person cutting the cake selects his portion last. He is therefore motivated to cut the cake as evenly as he can, otherwise he'll be left with the smaller, less-desirable portion. The cut-and-choose protocol [CCD88][BJK06] realises this abstraction in setting where a user must prove the correctness of a series of statements. The prover then selects a subset of these statements and asks the user to reveal their construction. If the user is able to successfully demonstrate the construction of the chosen statements, then the prover assumes that all the other statements are also correct. On the

other hand, if the user is unable to successfully demonstrate the construction of the chosen statements, then the prover assumes that all the other statements are also incorrect.

Alice uses the cut-and-choose protocol to prove that she owns a valid escrow certificate, and consequentially, prove that she has divulged her identity to the TTP. To achieve this, Alice constructs a series of statements, which only the certificate's holder can construct. The service provider then selects *one* at random, and asks Alice to prove its construction. The purpose of this is two-fold: to prove that Alice holds to necessary information to construct it; and to prevent the service provider from acquiring her private information — all constructions must be revealed for this to occur. If Alice can successfully demonstrate the construction, the the service provider assumes that the remaining statements are also constructed correctly.

Now the service provider then knows that if Alice were to act illigitimatley, by providing the TTP with the cut-and-choose transcript, Alice's identity can be revealed so that she can be held accountable for her actions. If Alice is subsequently deemed to be the genuine holder of a valid escrow certificate, the DRM service provider grants Alice access — this could be done with a *service certificate* to prove, hereon in, that she has authentic access to the service that could be presented for each DRM session.

If at any time the DRM service detects that Alice is acting maliciously, the DRM service provider can request the TTP to reveal her identity. The DRM service provider could present the TTP with a transcript of the cut-and-choose proof, which gives the TTP enough information to learn her identity. The TTP can then provide an appropriate authority with the appropriate information to hold Alice accountable. The cut-and-choose protocol is however, considered to be inefficient [CMS96]. In a DRM scenario, where the user-base is likely to be

large, inefficiencies such as this could cause a system bottleneck.

3.4.1 Appointed Verifiers

Now imagine a DRM situation in which Alice is able to purchase the same licence from multiple, competing content distributors; all of which offer different promotional incentives. Alice shouldn't be restricted to using a single licence broker or content distributor as the service, without any meaningful competition, would inevitably lack value to a consumer. Alice should be able to choose from a wide range of licence brokers and content distributors. The problem with such an ideology is that illegitimate brokers and distributors could come to the fore; many of whom may charge only a fraction of the recommended retail price to entice potential consumers without rewarding the content owner accordingly. So to minimise the possibility of illegitimate brokers and distributors infiltrating the system, there needs to be a mechanism in place such that Alice is able to choose from a selection of licence brokers and content distributors that are *legitimate*, preventing her from using illegitimate licence brokers and content distributors.

Camenisch and Lysyanskaya extended Kilian and Petrank's escrow system [KP98] such that a user is only able to prove his membership to an appointed verifier — and no other entities. Only entities verified as genuine are able to authenticate an escrow certificate. It therefore follows that Alice is only able to show her certificate to members of a set of verified entities, thus preventing users from using illegitimate licence brokers and content distributors, either intentionally or unintentionally.

In this system, the CA partitions the escrow certificate before issuing it to the user. The first segment is supplied to the user in its regular format; the second segment however, is encrypted with the appointed verifier's public key. Now the

only entity able to learn the full certificate is the appointed verifier, since it is the only entity with knowledge of the corresponding secret key. The appointed verifier can learn the full certificate by decrypting the encrypted segment and concatenating it to the plain segment.

This is particularly interesting since, in a DRM system, Alice could be issued with a certificate that states she has registered her details with an entity within the system. This certificate could be used during the process of licence payment and acquisition. Allowing only appointed verifiers to verify the certificate reduces the possibility that Alice may chose illegitimate brokers to acquire a licence from. Thus preventing Alice from participating with entities outside the system; perhaps entities with a less than scrupulous background or agenda. Otherwise, Alice could be tricked to use an illegal or illegitimate licence broker or content distributor — that perhaps does not credit the content owner with the correct monetary contributions.

Using identity escrow as a means to manage identity, a DRM system could provide strong anonymity (R1); whilst also providing accountability (R2). An escrow system could support unlinkability (R3) if a consumer were to be issued with a new escrow certificate for each session. Constant communication with a TTP during every transaction could however, become burdensome. For every consumer to communicate with the TTP during every transaction, the reliance on a single TTP could cause a system bottleneck; and this would be proportional to the number of consumers; adversely affecting the run-time efficiency.

However, only allowing a set of appointed verifiers to verify an escrow certificate provides an added layer of security. Users will not reveal, either intentionally or unintentionally, any identifying information except to those appointed verifiers. Users cannot be tricked into revealing such information to illegitimate entities as they don't hold the required secret key. Providing a stronger separability such as

this limits the access to those entities that require the knowledge of the certificate to function; those that do not require the knowledge, are not supplied it.

3.4.2 Multiple Collating Trusted Third Parties

Although Camenisch and Lysyanskaya [CL01] introduced appointed verifiers to identity escrow, a single TTP is still entrusted with the consumers' identity. A dishonest or compromised TTP could either impersonate consumers or introduce malicious users into the system. Marshall and Molina-Jiminez [MMJ03] introduced an escrow system that, instead of relying on a single TTP to issue escrow certificates, relies on a set of TTPs. Here, all TTPs must work together to issue an escrow certificate and to subsequently revoke a user's anonymity. Anonymity can be provided for a legitimate user, even if all but one TTP are corrupt.

To take part in the system, Alice must first acquire an *identity token*. Here, an identity token is constructed by a TTP to hide a user's identity; it's a signed encryption of Alice's public key. Only the issuing TTP is able to decrypt the token since it uses its public key to encrypt it (and it is the only holder of the corresponding secret key). The TTP however, knows the relationship between Alice's public key, PK, and her identity token. Since the token provides no direct link to Alice, she could use it as a pseudonym during identification procedures. Any verifier can verify the token's attached signature with the TTP's public key to verify that, under extra ordinary circumstances where Alice acts maliciously, the issuing TTP can decrypt the token and learn her public key. From her public association to this public key, the TTP can trace her identity to hold her accountable, and place her public key on a blacklist of unacceptable pseudonyms.

In saying that however, the TTP is still able to monitor Alice's activities. By issuing her with an identity token, the TTP has prevented other entities from

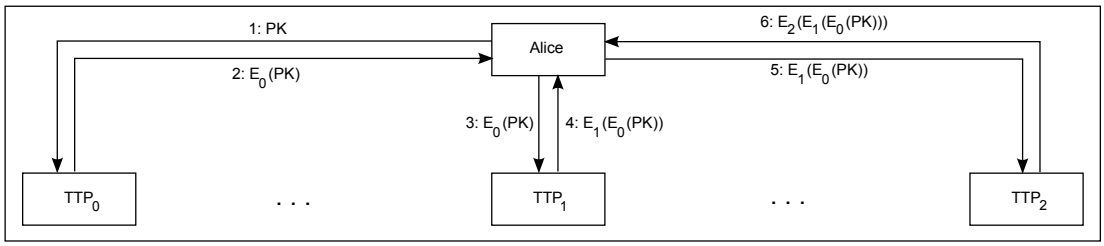

Figure 3.5: Multiple collating TTPs work together to issue Alice with a secure identity token $E_2(E_1(E_0(PK)))$.

monitoring her activities, but not the TTP itself. By storing the token issued to Alice, the TTP has a direct link from the token to her identity. So, with reference to Figure 3.5, Alice complicates her association with her token by acquiring more tokens. She uses her already acquired identity token (from TTP_0) to acquire another one from another TTP (TTP_1). By verifying the token's signature, TTP_1 has no qualms in issuing Alice with another identity token. It knows that under exceptional circumstances, with help from TTP_0 Alice's identity can be traced. So, instead of encrypting Alice's public key, TTP_1 encrypts Alice's previously acquired identity token; supplying her with a new identity token. By acquiring a different token for different sessions, Alice can hide the link between them from the issuing TTP. TTP_1 and TTP_0 could collude to trace her actions, but she could repeat this identity-acquisition process as often as she deems necessary to preserve her privacy. At any time, Alice's identity can be learnt by simply reversing the order of encryption, and decrypting at each stage.

For further security, she acquires a session key from the Service Provider (SP) she is communicating with. To anonymously identify herself to the SP, Alice uses her identity token, which the SP encrypts, signs and partitions[8]. The SP sends each partition to a different TTP and issues Alice with a session key, which from

[8]The partitioning procedure utilises Rabin's Information Dispersal Algorithm (IDA) [Rab89]. IDA is developed to break a file F of length $L = |F|$ into n pieces F_i, $1 <= i <= n$, each of length $|F_i| = L/m$, so that every m pieces suffice for reconstructing F.

hereon in, she uses as her service-identifier.

Now, for Alice to partake in service she must use her session key during identification. Under exceptional circumstances where Alice acts maliciously after identifying herself with her session key, the SP initiates a vote. The vote requires all TTPs involved to decide if Alice's identity should be traced or not. This gives Alice the confidence that her identity should not be unfairly revealed; whilst also giving the SP confidence that misbehaving users should have their identity revealed. If the predefined percentage of voters vote to reveal her identity, then a backwards trace of her token can be undertaken to learn Alice's identity.

The backwards trace is initiated by the last TTP to encrypt and sign Alice's token. This TTP decrypts the token and sends it to the previous TTP to encrypt the token. This process is repeated until the user's identity is revealed. To further explain this, let's use an example.

In Figure 3.5, we illustrate a possible TTP arrangement. Three TTPs co-operate with Alice to provide her with a more secure token. Alice starts by sending her identity to TTP_0 (1). TTP_0 encrypts Alice's identity using its public key, and returns this to Alice (2). Alice can then use her encrypted identity as her token (t_0). To further secure her identity, and to prevent TTP_0 from associating t_0's activities with her identity, she uses t_0 to acquiring another token; so she sends t_0 to TTP_1 (3). TTP_1 encrypts t_0 using its public key, and returns new token t_1 (4). Alice is also able to use this as her token. However, she secures her identity even further by acquire another token from TTP_2 (5). TTP_2 encrypts t_1 using its public key, and returns new token t_2 (6). Alice is now able to use t_2 as her identity token. For Alice's identity to be linked to the token, TTP_0, TTP_1 and TTP_2 would have to collude.

It is important to note here that SP is unable to 'cheat' and revoke a user's anonymity illegitimately as it requires the help from all involved TTPs. A TTP

would not reveal the decrypted token without a proven ballot unless it was compromised. For Alice's identity to be unfairly revealed, all involved TTPs must be compromised.

As we have discussed, identity escrow provides conditional anonymity for its users, which is an important DRM requirement (R1). An identity escrow mechanism could be used to prevent a single entity from being responsible for identity management; whilst supporting DRM entities from performing DRM operations. A separate set of TTPs, not involved in the process of distributing content or licences, could be solely involved in identity management. By adopting such a mechanism, we could facilitate the anonymous acquisition of content and licences conditionally (R1). Under exceptional circumstances, the content distributor or the licence broker could initiate a vote for a consumer's identity to be traced (R4). Then, depending on the result, the consumer's identity could be traced from his certificate (as in Camenisch and Lysyanskaya [CL01]) or identity token (as in Marshall and Molina-Jiminez [MMJ03]) associated to a session to hold him accountable (R2).

Allowing a user to use different session keys could allow a user's sessions to remain unlinkable (R3); although linking them together (R4) under exceptional conditions could be inefficient or difficult to achieve without extending these systems. Using the same session key or certificate during each session could facilitate this, but all sessions would be intrinsically linked. Any onlooker could learn the required knowledge to accumulate all the sessions a certain individual participated in. The onlooker would be unable to credit these to a civil identity but he would be able to piece them together and develop activity dossiers.

3.5 Chapter Analysis

Throughout this chapter we have discussed various ideas, concepts, systems, and protocols concerning accountable anonymity. We did not restrict ourselves to DRM-specific primitives; we explored a range of anonymity primitives that *could* provide the required building blocks for an accountable anonymity service in DRM.

From our discussions throughout this section, and with reference to Table 3.1, it's apparent that a single primitive does not meet all our requirements (defined in Section 2.4). Although both 'anonymous identity certificates' and 'identity escrow with multiple TTPs' cater for our requirements to a certain extent, they do so with their own respective weaknesses — as discussed throughout this section. We therefore do not consider these, nor the other primitives discussed in this section, to be wholly acceptable primitives for our research model. However, by combining a selection of security primitives, we can start to build a viable solution.

It is clear from our discussions that to utilise these primitives, consumers must be known by a unique identifier. For consumers to preserve their privacy, they cannot be identified by their civil identity; but under exceptional circumstances (*e.g.* after the detection of malicious intent) a consumer must be identified. They must thus be identified by an identifier that is not their civil identity; *i.e.* with a pseudonym.

Pseudonymity however, only provides the first level of an accountable anonymity. There must be steps in place to provide unlinkability (R3). By issuing a consumer with a different pseudonym for each session he participates in, the consumer is able to hide his relationship with his sessions. For this to come to fruition however, and for accountability to be supported, there must be an entity or authority that, under exceptional circumstances (*e.g.* after the detection of

Table 3.1: Evaluation of existing anonymity solutions against our requirements.

	R1	R2	R3	R4
Pseudonyms and credentials	✓	✓		
Group signatures	✓	✓		
Fair blind signatures	✓	✓		
Anonymous identity certificates	✓	✓	✓	✓
Anonymous authorisation certificates	✓	✓	✓	
k-times anonymous authentication	✓	✓	✓	
Identity escrow with appointed verifiers	✓	✓	✓	
Identity escrow with multiple TTPs	✓	✓	✓	✓

malicious intent), is able to link them together (R4). This could be subtle so that only a trusted authority is able to link them together, or the pseudonym issuer could remember to whom it issues pseudonyms to. There must be a route from a consumer's civil identity through to his pseudonyms so he can be held accountable under exceptional circumstances (R2).

We have seen from our discussions in Section 3.1 that Chaum [Cha85] developed digital credentials for a pseudonymous system. By using credentials, a consumer, who holds many pseudonyms, is able to transfer information between two DRM entities; even if they each know him by a different pseudonym. By using such a system, a consumer could identify himself with a different pseudonym for each entity he operates with. We know that the consumer is not restricted to only dealing with a single entity. For operations that require multiple DRM entities (such as licence payment and acquisition, which are closely linked) the consumer can use a credential. We could further this position by utilising an identity escrow scheme. A TTP, that has no interest in DRM activities, could manage the link between a consumer's civil identity and his pseudonyms. By removing the TTP from the DRM processes, it would be unable to monitor a consumer's activities.

As a part of providing accountability, it is also important to provide non-repudiation. It is important that if a session is deemed to be malicious, the owner cannot later reject ownership. From our discussions in Section 3.2, we know that combining digital signatures with digital certificates can provide a form of non-repudiation. Only the holder of the secret key can issue signatures that can be verified using the corresponding public key. Since the secret key is kept secret, a verifier can be assured that the message is authentic and originated from the said user. As a consequence, the consumer can therefore not, at a later time, reject that it was indeed him who issued the signature. It is computationally difficult to compute a signature that can be authenticated with a public key whose corresponding secret key is unknown.

It is thus vitally important that, with the use of digital signatures with pseudonyms, that the pseudonyms are certified. We know that certification can be achieved anonymously from our discussions in Section 3.3. Under such a setting, it would therefore be possible for a consumer, who is identified by a pseudonym, to sign sessions to prove their authenticity and at the same time, satisfy non-repudiation. The digital signatures we have discussed can however only be realised under settings in which the secret key remains secret.

Now, similarly to public key cryptography, we assume that Alice is the only individual in possession of her secret key. Therefore, she is the only individual to prove ownership of her secret key[9]. The secret key is assumed to be kept secret by Alice; whilst Alice's ownership of the corresponding public key is assumed to be published. Critchlow *et al.* [CZ04] however, consider this to be insufficient.

To prove ownership of a public key, simply demonstrating the knowledge of one piece of information (*i.e.* the correct secret key) could be insufficient. If an adversary were to acquire this, then he could illegitimately prove ownership

[9]As used as an authentication step during signature authentication.

of items he did not indeed own. It is particularly insecure since Alice's identity could be intrinsically linked to the key pair (using *e.g.* digital certificates).

To counter such malicious activities, Critchlow *et al.* proposed a system for the acquisition of anonymous certificates that minimises the risk of an adversary from acquiring an anonymous certificate with a compromised secret key. Here, Alice must demonstrate the knowledge of two pieces of information, effectively two secret keys before a certificate is issued. We mention this here as it could provide a safer DRM environment. However, an important requirement of public key cryptography is that the secret key is kept private. Exposing one's secret key could cause irreparable damage. We therefore believe that, if public key cryptography is as widely accepted as it is currently, then this is not a particularly pressing concern. We must however endeavor to ensure that the secret key is secured.

From our discussions, we can see that the foundations for a DRM service supporting accountable anonymity can be constructed from these security primitives. By using the strengths of different primitives for different settings, we can start to develop an accountable anonymity service in DRM. For every primitive however, there are also weaknesses. Our discussion has centred on the pseudonymity of each consumer; and to do so, we have based our pseudonyms on public key encryption; from which we can utilise (certified) digital signatures. However, for public key cryptography to be secure, the keys must be large. This has a direct affect on the computational cost. Further to this, it is important to consider the security of the secret key. Revealing one's secret key (*e.g.* as proposed by Lysyanskaya *et al.* [LRSW00]) would severely affect the security of such a system. We summarise our finding in Chapter 3 in Table 3.1.

Chapter 4

Privacy-Preserving Online Payment

From our discussion in section 2.1, we know that a DRM session encompasses a set of transactions. For a consumer (*e.g.* Alice) to acquire some content, she must perform all transactions within her session. One transaction, licence payment, typically requires help from a dedicated payment gateway. This requires Alice to reveal her payment details, which are usually the same for every payment. Thus, given that payment details are unique to the consumer, an observing adversary may potentially form a link between payment transactions for different sessions. Moreover, using Alice's name and address as a component of a her payment details can provide a direct link to her civil identity. We know from our discussions in Section 2.2 that *conditional* unlinkability is vitally important to provide accountable anonymity. It is therefore important that consumers are able to conditionally hide this information to prevent onlooking adversaries from linking a consumer's sessions together. We should prevent any malicious onlookers from

linking a consumer's payment transaction to his identity.

Throughout this chapter we focus on how online payment can be achieved whilst satisfying our requirements for an accountable anonymity service in DRM (see Section 2.4). In doing so, we must ensure that the consumer's payments cannot be linked together unless malicious activities are detected, at which point the consumer's identity should be revealed. Moreover, the link between a consumer's identity and his payment should be conditionally hidden. We should therefore ensure that consumers are protected by accountable anonymity during payment transactions; just as they should be during other DRM transactions.

In the remaining part of this chapter, we first explain the importance of payment anonymity in both the general online domain and in our specific field of DRM. We then discuss online payment ideas, procedures, protocols and systems that support accountable anonymity at varying levels, critiquing them against our set of requirements for an accountable anonymity service in DRM.

Under a typical scenario in which Alice acquires content from a DRM service, she is asked to enter her payment details after selecting the content/licence she wishes to purchase. This usually requires her to reveal her credit/debit card details (provided by *e.g.* Visa [vis10] or MasterCard [Mas10]), her billing address, and her delivery address. We believe that in doing this, Alice compromises her privacy. By supplying her personal information, she supplies the service provider with identifying information. This can be directly linked to her transactions, providing a platform for them to be monitored and her shopping habits inferred. This is highlighted by service providers that store a consumer's shopping history and payment details, like PayPal [pay10c].

Today, the most prominent online payment system is PayPal. Initially developed as a part of ebay [eba10], it has now branched off to be a recognised payment system used throughout the world wide web. Although it appears to be

more convenient, PayPal puts its consumers' privacy in jeopardy. By storing each user's payment details PayPal doesn't require its users to enter them for every transaction but this compromises their privacy. It is imperative that a consumer's payment details are kept hidden. By exposing them, a malicious person could link his payments together, or even use them to illicitly purchase content. It is not just payment details that can be used to link transactions together; email addresses for example, which are often required during identification processes, can be easily linked to a transaction. One particularly concerning policy of PayPal is the liberal nature in which it exposes a user's email adress during a transaction; making it particularly easy for merchants to link it to a transaction. According to PayPal's privacy policy [pay10b]:

> We work with third party merchants to enable them to accept payments from you using PayPal. In doing so, a merchant may share information about you with us such as your email address when you are attempting to pay that merchant. We use this information to confirm to that merchant that you are a PayPal customer and that the merchant should enable PayPal as a form of payment for your purchase.

We believe that payment systems should be stringent in the information they reveal; email addresses in particular can usually be easily linked to the holder. By using his email address in an identification process (possibly the very same email address that PayPal liberally shares) the merchant and payment gateway are able to collude to build individual-specific dossiers. We have discussed (in Section 3.1) how using a common unique identifier with different organisations could compromise a consumer's identity: it provides a common identifier that these two different organisations can use to acquire individual-specific information

outside their operational scope.

Once a PayPal consumer has identified himself with his email address and authenticated himself with a password, he can login. Once logged in, the user (or anyone that knows the login details, either honestly or maliciously) is presented with a detailed shopping history; including, but not limited to: the merchant's details, the product's details, the delivery address, the amount paid, and payment method. PayPal thus holds a detailed user-specific dossier that it *could* use illegitimately for commercial advantage. We believe that to preserve a consumer's privacy, histories such as this should not be readily available.

PayPal however also provides, for its eBay users specifically, with what they call 'full protection': a realisation of consumer- and merchant-accountability. Here, PayPal mediates and facilitates communication between the buyer and the seller in the event of a dispute. By providing accountability, under a dispute where a buyer doesn't receive the product he has bought, for example, PayPal is able to credit the buyer's account and debit the seller's. We know that accountability is an important security property in privacy-preservation systems.

In line with this approach, Amazon [ama10], one of the largest commercial online service providers, recently deployed an online payment system, PayPhrase [pay10a]. PayPhrase similarly stores consumer-specific histories. We believe that PayPhrase and PayPal acquire enough information to monitor users' activities, build dossiers and infer trends and habits. As a result, the users' privacy is compromised. To protect Alice's privacy, we believe that the service provider should be segregated into two distinct entities: one that manages the products (*e.g.* a licence broker); whilst another manages the payment (*i.e.* a payment gateway). By segregating duties like this it is more difficult fore Alice's privacy to be compromised as each entity is supplied with less information. The licence broker is only supplied with the content/licence Alice wants; whilst her payment

gateway is only supplied with her payment details. Under such a segregation it becomes difficult for either the payment gateway or the licence broker to monitor Alice's activities without collusion.

We believe that an ideal payment system would not store, or more to the point not see, any of the content-related transaction details; only the payment details. A simple solution could be sought from simply hiding the product bought from the payment system. So whenever a product is bought, the merchant does not share the product's identity with the payment system, merely the amount of money required. Although this could alleviate some of the security issues, it is not necessarily a complete solution. This may not be as convenient for consumers who will be unable to determine which payment is for which product in their payment statements, for example; however, their privacy is more strongly preserved.

In the remainder of this chapter we discuss and critique anonymous but accountable payment systems.

4.1 Signature-Based Online Payment

The first electronic anonymous payment system was proposed by Chaum [Cha83] in 1983. Whilst introducing blind signatures, Chaum described how they could be used to provide an untraceable payment system. From our discussions in Section 3.2, we know that a blind signatures scheme can provide anonymity. By using a payment system based on blind signatures, consumers' privacy can therefore be protected.

Chaum's untraceable payment mechanism works on the premise that a message signed with a bank's secret key is worth a fixed amount, say £1.00. Alice can then use such a message as a digital coin. The signature cannot be forged so it must have originated from the said issuer — *i.e.* the bank.

To construct a coin, Alice blinds a value she chooses at random. She forwards the coin to her bank that debits her account, returning the signed coin. By unblinding the coin Alice is able to hide her relationship to the coin, from the bank; whilst the signature still holds. From the bank's perspective, the coin is no different from any other coin, and it cannot be linked to Alice. Alice can now use this coin during payment processes.

For a coin to be accepted during a payment process, its signature must be valid. This can be verified by using the bank's public key. Even if the merchant and bank collude at this stage, they are unable to link the coin to Alice. The coin has no link to Alice; nor does it, from the bank's perspective, have any resemblance to the coin issued to her. A valid signature confirms the authenticity of the coin, and its absence from a global list of spent coins proves it hasn't be spent yet. The merchant can present the coin to the bank, which adds the coin to the list of spent coins, and credits the merchant's account accordingly.

By basing online payment system on a blind signature scheme, Alice is able to acquire digital coins, and spend them anonymously (R2). The unconditional nature of the anonymity provided by a blind signature scheme however, inevitably lacks the accountability that we desire for an anonymous but accountable service in DRM. We consider the payment process to be part of the DRM service, and thus, as a result, it should conform to our set of requirements for an accountable anonymity service in DRM (defined in Section 2.4). By basing a payment system on blind signatures, we are unable to provide accountability (R2). Although highly desirable for consumers, we know from our discussions in Section 2.2 that anonymity without accountability can be dangerous — malicious users cannot be reprimanded for their actions. The system is fundamentally flawed for our domain of interest; it provides Alice with *unconditional* anonymity for her payments. In line with our requirements for an accountable anonymity service in DRM, it is

important that we provide Alice with *conditional* anonymity for her payments.

The system itself can also be considered to be inefficient. After a coin has been spent, it must be added to a list of spent coins. Subsequent attempts to spend a coin on such a list are rejected. The inherent problem with a such a list is its maximum size, which is, theoretically, infinity. Coins could include some sort of expiration mechanism but the concept of coins expiring would not be particularly desirable for its users. Moreover, it is computationally easy for a malicious user to steal, or even copy, digital coins. Since the coins are just a series of binary digits, a precise copy will result with a legitimate coin. Provided the malicious user spends the coin before the genuine holder, it will be accepted; the genuine holder will be rebuffed when attempting to spend it later.

4.2 Online Payment with a Trusted Third Party

Unconditional anonymity, as mentioned previously, can be easily misused by those with malicious intent. Basing a payment system on blind signatures opens up the possibility of unconditional anonymity. Chaum *et al.* [CFN90] proposed an extension so that doubly spent coins can be detected without verifying a list of spent coins. It does not however, provide methods or procedures to trace malicious identities.

For a conditionally anonymous payment system to be viable, there must be a process or procedure in which an authorised entity can trace a malicious consumer's identity — in other words, a process or procedure to link a withdrawal to a deposit in the event of any irregularities. Assuming that each withdrawal is performed by the identified holder of the account, linking the withdrawal to the deposit implicitly reveals the identity of the holder (*i.e.* the withdrawer).

A simple way to provide accountability would be for the bank to only issue

coins that are inherently linked to the consumer. This way, any coin can be directly linked to the owner. However, under such a setting, the consumer's identity would be severely compromised. Not only would the bank be able to monitor the consumers' actions but so would any other onlooker. The bank must therefore disguise the consumer's ownership so that only an authority is able to learn a consumer's actions.

By disguising or encrypting the identifying information, the bank is able to conceal a consumer's activities from other entities but not from the bank itself. From our discussions previously, we know that it's ideal for entities that are involved with the regular processes from not participating in the process of anonymity revocation — as they could display a form of bias. Moreover, as we have already mentioned, a payment transactions is part of a larger session consisting of many transactions. We must therefore ensure that a session's constituent transactions must remain unlinkable to each other. If we were to utilise this particular approach in the domain of DRM, we must ensure that the account activation process, where the consumer reveals his identity, is hidden from the deposit process. By collating payment information with information from another transaction, an onlooking adversary may be able to start linking activities together. Payment information typically contains personal information; whilst other transaction information may include content information. Revealing both types of transactions to a single DRM entity could thus become problematic — it could provide the foundations for dossier building. To prevent such occurrences, we could introduce a TTP to deal with different aspects of the service.

Jakobsson *et al.* [JY96] suggested that a TTP could be involved in every payment transaction — effectively acting as an intermediary. Here, Jakobsson *et al.* utilise blind signatures so that the TTP is responsible for blinding and unblinding the coin. For Alice to acquire a blind signature in this system, she

must supply the TTP with the unblinded coin she wishes to be signed. The TTP blinds the coin and then acquires a blind signature from the bank. The TTP can then forward the unblinded coin with the attached signature to the consumer. The TTP is thus able to link the signature to the unblined coin. The blinding property prevents the bank from linking the signature to the coin. Involving the TTP during every transaction, is however, inefficient.

Camenisch [CPS96] proposed an anonymous payment system where the TTP is only involved during account activation; so it need not be involved during every transaction. Since the process of opening an account is less frequent than the process of withdrawing a coin, the TTP is used less often and thus, it is more efficient. From the information the TTP gains during account activation, it is able to link a merchant's deposit to a consumer's withdrawal. In other words, whilst the blinding property prevents the bank from tracing the coins deposit, it does not prevent the TTP; the TTP is able to link a coin's withdrawal to when it is deposited by the merchant. By colluding with the bank, the TTP is able to learn the identity of the consumer; whereas the TTP is able to collude with the merchant to learn what the consumer bought. By segregating duties like this, the consumer's privacy is less likely to be compromised. Although this is more efficient than the previously mentioned systems, its efficiency can still be improved. Ideally, the TTP would only be involved during anonymity revocation; *only* when a consumer's anonymity must be revoked should the TTP be used. Consumers who act without any malice, should never interact with the TTP.

Brickell *et al.* [BGK95] and Stadler *et al.* [SPC95] independently proposed a system in which the TTP is *only* used during anonymity revocation. This frees up the system from the bottleneck of using it during every transaction, providing a more efficient system. Here, each coin contains encrypted information that provides a direct link to the consumer it was issued to. The information is

encrypted using the TTP's public key so that it is only the TTP that is able to learn the link between a merchant's deposit and a consumer's withdrawal. Both these solutions however, rely on the cut-and-choose protocol which, from our discussions in Section 3.4, we know to be inefficient. In a DRM scenario, where the consumer-base is likely to be large, inefficiencies such as this, which requires cumbersome communications and calculations, is likely to cause a system bottleneck. In 1996, Camenisch *et al.* [CMS96] proposed a more efficient solution based on fair blind signatures.

We know that, from our discussions in Section 3.2, fair blind signatures provide the anonymity of blind signatures but fairly, or conditionally. Camenisch *et al.* provide a compromise between the legitimate need for consumer identity protection, and effective prevention and protection against its misuse, to provide a payment system which is *fair*. By using a fair payment system, consumers can be provided with accountable anonymity.

Under the system proposed by Camenisch *et al.*, consumers hold regular bank accounts from which they can debit and credit as they wish in the regular manner. Here however, each account is also bound to a key pair. In line with fair blind signatures, the system incorporates a TTP. The TTP is the sole entity responsible for consumer anonymity revocation. It is not necessary to involve the TTP in every transaction; only during anonymity revocation.

Each consumer participates in the system in one of two stages: withdrawal or payment. The withdrawal stage is performed by the identified holder of the account; no-one else is able to withdraw from the account. As a result, the consumer's identity is intrinsically linked to his withdrawal. The consumer uses fair blind signatures to disguise the link between his withdrawal and the merchant's deposit to hide his association to the deposit. The TTP is however, able to unblind this relationship to expose a consumer's withdrawal to the merchant's

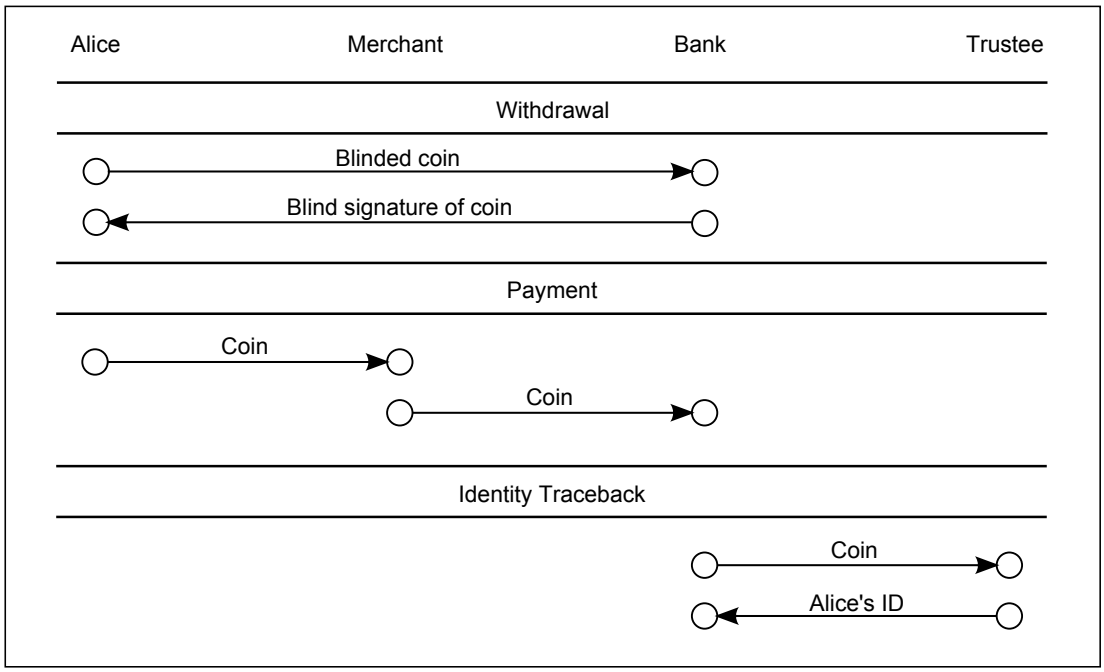

Figure 4.1: The withdrawal, payment and traceback protocols from Camenisch *et al.* [CMS96].

deposit; otherwise, the blinding property hides this relationship.

Payment is achieved when a consumer issues a merchant with his coin in return for a particular product. The merchant is able to then deposit this coin into its account, which is credited accordingly. Even if the merchant uses the same bank as the consumer, the bank is unable to link the deposit to the withdrawal. In other words, the bank cannot see that the coin withdrawn by the consumer has been deposited by the merchant — this link is hidden.

During withdrawal, a properly identified consumer must construct a coin and prove to the bank that it has been correctly formed. To construct the coin, the consumer constructs a random session key based on the TTP's public key. The bank stores this session key alongside the consumer's account details. This session key then becomes part of the coin. The consumer then blinds the coin to not only prevent the bank from linking it to the coin it issued but to also hide the session key.

Upon deposit, under exceptional circumstances (*e.g.* after the detection of malicious activities), the bank can ask the TTP to extract the session key from the coin. The TTP can do this using its private key. The bank can then cross-reference the session key with its records to learn who the coin belonged to. It is important to note here that neither the TTP nor the bank are able to trace a consumer's identity on their own, they must collude.

Here, Camenisch *et al.* provide a system in which a legitimate user is able to make a payment anonymously (R1). An illegitimate transaction could lead to the corresponding user's identity being revealed. All other users in the system would still have their privacy safely secured. After a consumer has been found acting maliciously, it is important that all other coins held by the consumer are revealed and prevented from being spent. There is however, no process defined by Camenisch *et al.* to achieve this.

Moreover, the run-time costs may exceed an acceptable amount, particularly for large systems. All coins must be placed in a database of valid coins until they have been spent. So for every payment transaction that occurs, the bank must check the coin against the database to ensure that it has not already been spent. Not only would this require a large storage medium but the process of checking a coin's existence in the database could be rather cumbersome. Although this could be combated by including an expiration date inside the coin itself (as suggested by Chaum [CCD88]), it is unlikely to be widely accepted by consumers with many of whom unlikely to accept the fact that coins can expire.

4.3 Online Payment without Keys

We have discussed how a single entity (*i.e.* the TTP) or a group of entities (*e.g.* the TTP and the bank) could be given the ability to trace a consumer's

identity from a merchant's deposit. In doing so however, these systems provide no technicality to prevent illegitimate identity traces. The only measure in place to prevent such occurrences is the *honesty* of the TTP (and the bank) — not particularly desirable from the consumers' perspective. Although systems such as these provide consumer anonymity, it can be considered as weak anonymity. A stronger, more viable solution would prevent illegitimate identity traces whilst supporting legitimate ones.

In 1999, Sander and Ta-Shma [STS99] proposed a signature-free solution in which a malicious consumer attempting to spend a coin twice can be identified; whilst consumers acting without any malice cannot be traced. Here, the bank operates in a fully public domain, holding no secret key. The security of the system relies on the ability of the bank to maintain the integrity of a public database of valid coins; as opposed to the security and secrecy of the bank's secret key as we have seen previously. All transactions are published in a database and are publicly verifiable.

During registration, each consumer is issued with a unique identifier made up of two constituent variables. Whenever Alice makes a withdrawal, she must first construct a coin based on these two constituent variables. She disguises them however, so her ownership of the coin is not readily apparent. To do so, she uses a hash function[1].

So, after identifying herself with her unique identifier, Alice supplies the bank with her hashed coin. To prove that she constructed it correctly (*i.e.* that she used her unique identifier in the process), she uses a zero-knowledge proof[2] that she holds the knowledge of the coin's plain value. After a successful verification,

[1]We have touched on the hash function in Section 2.3. A hash function is a mathematical function that converts one string into another unlinkable value, called the hash value.

[2]A zero-knowledge proof is an interactive, two-party protocol for one party to prove to another that a particular statement is true, without revealing any additional information — only the veracity of the statement.

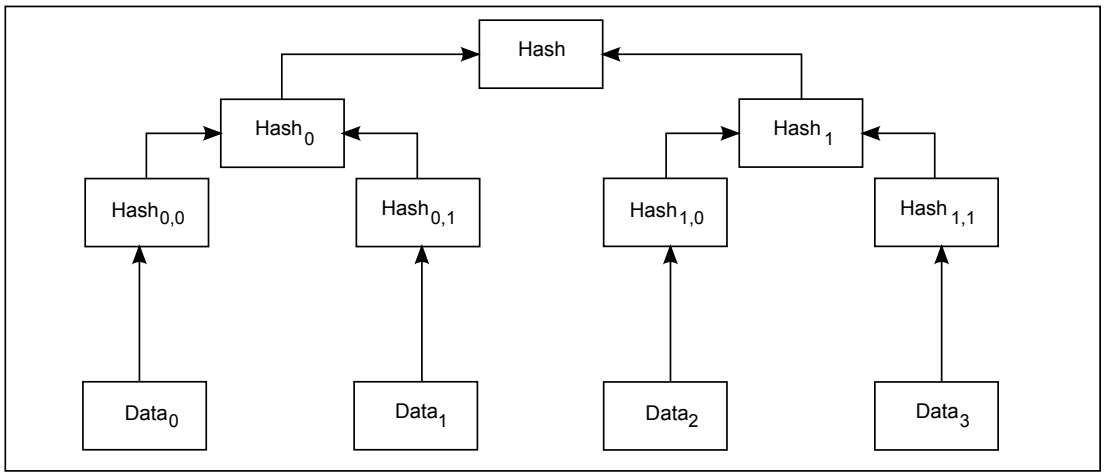

Figure 4.2: A simple hash-tree.

the bank inserts the hashed coin into a vacant leaf of a publicly verifiable *hash-tree* maintained by the bank.

Hash-trees were first proposed by Merkle [Mer87] in 1987. A hash-tree (see Figure 4.2) is a data structure that represents information in a tree structure. Instead of inserting the actual plaintext information into the tree, the information's hash value is inserted. Simply speaking, a hash-tree is a tree of hash values in which the leaves are hash values of information. Hash values are much smaller than typical pieces of information so it is more efficient to store and search through them.

All coins are inserted into leaves of the tree, from which a hash-tree is formed over using a hash function. The root of the hash-tree is authenticated, made public, and holds its integrity. Since storing coins in a hash-tree is more efficient than the plain-list equivalent, it provides a much more efficient method for the merchant (and the bank) to search the hash-tree and determine if the coin is present or not. Now, assuming that the hash-tree is signed by a recognised authority, the coin's presence in the hash-tree proves its validity; its absence proves its invalidity. So, after Alice has identified herself and constructed a valid

coin and sends its hash to the bank, the bank inserts the hash into the tree. The bank then debits Alice's account accordingly. Alice does not reveal the plaintext version of the coin to the bank; she keeps this secret. Since it is Alice who constructed the coin and thus the hash, it is only her who is able to link the hash to the coin. To spend the coin, Alice therefore just needs to provide a zero-knowledge proof that she has the knowledge of its plaintext value. The bank then moves the coin from the database of valid coins, to a database of invalid coins, and credits the merchant's bank accordingly. Alice must therefore make sure that all her coins are kept secret; otherwise they could be stolen or copied and used illicitly.

Before the coin is accepted however, the merchant ensures that the coin exists in the database of valid coins. If it's listed in the database of invalid coins, then Alice is trying to spend the same coin twice — which is a malicious offence. In attempting to spend the same coin twice, the consumer effectively presents the bank with two different linear equations. As both coins contain the consumer's constituent identifiers, simple simultaneous equation mathematics can be used to solve the equations to, in turn, obtain them. From this, Alice's unique public identifier can be computed, and she can be held accountable for her actions (R2). She can then have her privileges revoked and be reprimand accordingly after her anonymity (R1) is revoked.

Although Sander and Shma prevent illegitimate identity traces, they do not present a procedure in which, after detecting Alice's malicious activities, all her coins can be revoked (R4). After being revoked it is trivial (R3) to prevent Alice from withdrawing more coins (since her identity has been revealed), but it is not so trivial to link Alice's coins together to add them to the database of invalid coins. We know that, from our previous discussions (in Section 2.2), it's important to prevent Alice from continuing to use the system after having her

malicious activities exposed.

Further to this, even though Sander and Shma prevent illegitimate identity traces, it is still possible for the bank and the merchant to collude and monitor Alice's activities. The bank knows the link between Alice's identifier and the hashed coin; whilst the merchant knows the link between the hashed coin and the product she has bought. Together, they can construct the link from Alice's identifier through to her product. By following this procedure multiple times, together they can start to infer Alice's shopping habits. For a secure implementation, the bank and the merchant are expected to remain *honest*, but there are no procedures in place to enforce this.

However, one particular weakness of other systems, as suggested by Sander and Shma, is their dependence on the security of users' secret keys. In key pair based systems, it is vitally important that the secret key is kept secret; the security of the system is dependent on it. Here however, such a concern is removed. There is no need for a secret key; and thus no need for any private or secret information. We must however, consider the unique identifier issued to each user. Whilst it is publicly associated to its holder, its constituent components are not. A user's unique identifier is constructed using these components, and so are his coins. It is therefore vitally important that these components are kept hidden. With the motivation of eliminating the need for keys, Sander and Shma introduced different secret values. Although this is not particularly concerning, the users must ensure the secret information remains hidden or coins could be constructed and spent illicitly on their behalf.

Moreover, implementing the database of valid keys as a hash-tree provides added efficiency. However, having to maintain a list of valid coins could prove costly. For every new coin that is constructed, a new database must be published;

the same can be said for spent coins that must be removed from the tree. Ensuring that one is looking at the most up-to-date tree could prove to be difficult; particularly in large systems. Under a large consumer base, the tree could be considerably large. Even though it is more efficient to search through a tree than a list approach, it could prove to be cumbersome. Whilst the bandwidth required to allow many simultaneous accesses could be expensive.

It must also be considered that there is a database of invalid coins. This database would continue to grow in size indefinitely. After a coin has been spent, it must be added to the list of invalid coins. Restricting coins to a validity date could alleviate the problem but, as we know, the very concept of expiring money is unlikely to tempt consumers into using a such a system.

4.4 What is Missing?

Throughout this chapter we have focussed on payment systems that provide customers with privacy preservation. Our domain of interest is accountable anonymity in DRM, and online payment is an integral part of it. We must ensure that, as part of preserving consumers' privacy, all aspects of the DRM environment do indeed protect consumers' privacy — including payment. We have discussed various payment systems; none of which were however, restricted to DRM. For an anonymous but accountable DRM solution to come to fruition, it is important that it is as flexible and consumer-desirable as possible; without neglecting the importance of the content owner's needs.

The main goal of a consumer-desirable system is to provide consumer anonymity (R1). The payment systems we have discussed all achieve this. At the same time however, we must provide accountability (R2). We know that anonymity without accountability can be misused by malicious consumers to profit unfairly.

This is highlighted with what has become known as the *perfect blackmailing attack*.

In 1992, von Solms *et al.* [vSN92] used a genuine criminal ransom case in which the criminal was caught, to explain how if he used an online payment service based on blind signatures, he could have escaped with the money. Here the criminal could have exploited the strong anonymity of blind signature schemes to keep his identity hidden, despite retaining the money. Examples such as this eliminate payment systems based solely on blind signatures (*e.g.* that proposed by Chaum [Cha83]) from an anonymous but accountable DRM service. Blind signatures provide unconditional anonymity, and with that, perfect blackmailing crimes can prevail. Those using fair blind signatures *and* a TTP provide anonymity revocation so that consumers can be held accountable. They can therefore protect against the perfect blackmailing attack. We could therefore utilise the already existing system proposed by Sander and Ta-Shma during our payment in a DRM system. It provides its users with accountable anonymity and thus fits with our requirements for an accountable anonymity service in DRM (Section 2.4).

The perfect blackmailing attack however, is only one such attack for electronic payment systems. Other problems can also persist in an anonymous payment system. In 1996, Jokobsson an Yung [JY96] presented a problem known as the *bank robber attack*. Here, an attacker learns the bank's secret (usually a secret key) and starts counterfeiting money. This is particularly devastating if such activities are difficult to learn. The Basel Committee on Banking Supervision described such an attack as particularly concerning [bis04].

The bank Governors of the Group of 10 Nations (G-10) created the Basel Committee on Banking Supervision [bis10], an institution that formulates broad standards and guidelines for best practices in banking supervision (*e.g.* Basel II [bis04]). In 1998, the Basel Committee on Banking Supervision [bis98] declared

that:

> of direct concern to supervisory authorities is the risk of criminals counterfeiting electronic money, which is heightened if banks fail to incorporate adequate measures to detect and deter counterfeiting. A bank faces operational risk from counterfeiting, as it may be liable for the amount of the falsified electronic money balance. In addition, there may be costs associated with repairing a compromised system.

It is therefore important that the threat of counterfeiting is minimised. We have seen how electronic coins can be constructed using secret information. If the bank could construct coins using its secret key, then a malicious or compromised bank would be able to construct counterfeit coins.

We have seen a solution proposed by Sander and Ta-Shma in which the bank does not hold secret information. Under a DRM solution utilising this system, a compromised bank can have a significantly reduced adverse affect. Conversely, the security of the system proposed using signature-based solutions is heavily linked to the security of the bank's secret key. Exposing it could have devastating repercussions.

However, under the solution proposed by Sander and Ta-Shma, we must consider that the unique identifier, which each user is identified by, is composed of two *secret* values. Exposing these secret values could lead a malicious individual to impersonate him and illicitly generate coins in his name. Moreover, it must be also be considered that under this system, a revoked consumer is still able to use the system and spend coins that were issued previously. Under the signature based system proposed by Camenisch [CPS96], this is not possible. From the session keys stored by the bank, the TTP is able to compute the corresponding coins and place them on a blacklist of unacceptable coins. Subsequent attempts

to spend the coin are thus rejected.

So, under an accountable anonymity DRM service, it is important that, if the payment gateway utilises electronic coins, it does not present malicious or compromised TTPs (or banks) with the ability to create counterfeit coins; whilst protecting against the perfect blackmailing attack and the bank robbery attack.

One solution that we haven't discussed as of yet, is the possibility of using an already existing payment system (like that of Visa [vis10] or MasterCard [Mas10]). If we were to follow this route, we must ensure that the payment details are not revealed to the licence broker. Similarly, we must also ensure that the content/licence details are not revealed to the payment gateway. By completely separating payment from content/licence acquisition in this manner, the consumers' privacy could still be preserved. One particular requirement for an accountable anonymity service in DRM is conditional unlinkabilty: we must ensure that each transaction is conditionally unlinkable to other transactions.

Under such a setting, we would be unable to hide a consumer's identity from his payment — since the consumer's identity is innate within the payment details. However, by restricting the information the payment gateway receives, regarding the content and licence details, we would be able to provide anonymous payment in that regard. The consumer's identity may not be hidden but the licence for which the payment is for, can be hidden — making it unlinkable.

Conditional unlinkability is an important requirement (R3 and R4). Linking a malicious consumer's transactions together to prevent him from continuing to use the service continually is imperative for the security of the system. Allowing him to continue to use the system, despite being found with malicious intent, allows him to continue to act maliciously without any further reprimands. This can be seen in what is often referred to as the *money laundering attack*.

First presented by Molander [MMW98] *et al.*, the money laundering attack

allows a malicious individual to escape from authorities even after committing money laundering offences. Under the smokescreen of anonymity, malicious individuals are able to hide the identity, source, or destination of illegally gained money. This can have dramatic legal ramifications.

To protect an accountable anonymity service in DRM, we must protect against money laundering attacks, perfect blackmailing, counterfeiting and the bank robbery attack. We must ensure that banks, merchants and consumers alike must all agree on the protection of the payment system and the DRM system as a whole.

We must also consider how difficult it is to provide an adequate implementation of a payment system; not just technically, but practically too. Payment systems must adhere to the strictest of recommendations, guidelines, rules and regulations for them to be widely accepted and used commercially. Displacing regular currency systems with electronic money systems is non-trivial, as expressed by the Basel Committee on Banking Supervision in 1997 [bis97]:

> if electronic money does grow to displace currency to a substantial degree, loss of confidence in a scheme could conceivably have broader consequences for the financial system and the economy.

Displacing regular currency systems is considered a particularly difficult task; not necessarily in a technical manner, but because of the sensitive environment it sits in. Despite providing varying forms of anonymity, the systems we have discussed possess too many weaknesses to become prevalent in today's society.

It must therefore be considered that, as anonymous electronic payment systems haven't significantly increased their prevalence since 1997, the Basel Committee on Banking Supervision's argument still stands. Despite the possible advanced security that privacy preserving online payment systems provide, they also bring considerable weaknesses. With the ever-increasing importance of money

and monetary values, it is unlikely that content owners and content consumers alike will wish to use a system with such weaknesses; particularly when such systems are not widely accepted. Consumers are more likely to persist with solutions they are happy with; even if it is at the expense of their own privacy. By providing any form of anonymity with payment, banks and merchants will automatically take a sceptical stance. Using an already tried and tested prevalent payment system like Visa or MasterCard could therefore be a more rewarding research direction. By hiding the payment details from the licence broker and the content/licence details from the payment gateway, a form of unlinkability can be held and accountable anonymity can still be supported. We believe this is a more promising line of research; it is the direction of research we take in Chapter 5.

We believe that for an anonymous payment system to become prevalent, our set of requirements (Section 2.4) should be adhered to. To the best of our knowledge we don't believe that there is a system available to achieve this. We summarise in Table 4.1.

Table 4.1: Evaluation of existing privacy-preserving online payment solutions against our requirements.

	R1	R2	R3	R4
Blind signature based	✓			
Fair blind signature based	✓	✓		
TTP based	✓	✓	✓	
Online Payment without keys	✓	✓	✓	

Chapter 5

Achieving Accountable Anonymity in DRM

This chapter proposes a new anonymous but accountable service in DRM. We call this, *Accountable and Anonymous DRM*, or *A²DRM* for short. We believe that, by satisfying our requirements for an accountable anonymity service in DRM (Section 2.4), A²DRM provides a more privacy perserving service than the state-of-the-art solutions we discussed in Section 2.3. We realise A²DRM by utilising pseudonymity, identity escrow, identity certificates, authorisation certificates, digital signatures, and RELs.

In Section 5.1 we project the best way forward for an accountable anonymity service in DRM. We then set out preliminary design considerations in Section 5.2. Section 5.3 proposes A²DRM; which we evaluate in Section 5.4.

5.1 Best Way Forward

From our state-of-the-art DRM review (see Section 2.3), we have seen that DRM systems can be segregated into three types:

1. device-based;
2. identity-based;
3. smart-card-based.

We believe that device-based DRM systems are too inflexible for consumers' needs. They prevent a single consumer from accessing a single piece of content on multiple devices using a single licence. Nowadays, most consumers tend to own a variety of content accessing devices. Preventing consumers from content access ubiquity is too restrictive; a consumer is likely to want to access his content on any *compliant* device he wishes. We therefore believe that pursuing research of such a system with privacy preservation is rather futile. Likewise, smart-card-based DRM systems provide little flexibility. Consumers would have to replace or upgrade their existing playback devices with special smart-card-reading playback devices. Not only would this be at an additional cost, it removes the possibility of using legacy playback devices. We therefore believe that the only viable route of research is with identity-based DRM systems.

Identity-based DRM systems have the advantage of working with any device. Instead of being restricted to the device (in device-based) or the type of device (in smart-card-based), a consumer is able to use any *compliant* device. We highlight that by identity, we do not necessarily mean the consumer's civil identity. It may be that a consumer could use a pseudonym as his identifier. By using different, unlinkable pseudonyms for each DRM transaction, we can provide consumers with anonymity.

The consumers' acceptance of a DRM system is also heavily bound to the acceptance of the online payment process. Online payment is well used throughout the commercial domain. It is not inherently anonymous however; by revealing his payment details, a consumer reveals aspects of his identity. Online payment, for example, typically requires one to reveal the card holder's name. Although anonymous payment systems do exist (*e.g.* Camenisch *et al.* [CPS96] and Sander *et al.* [STS99]), they are not widely used. Based on the discussions in Section 2.3 their relative weaknesses make them unlikely candidates for Anonymous but Accountable Digital Rights Management (A^2DRM). As a compromise, we amalgamate anonymous principles with the widely used online payment systems. We discuss this in more detail in Section 5.3.

5.2 A^2DRM Design Preliminaries

5.2.1 Design Requirements

A^2DRM revolves around our set of requirements for an accountable anonymity service in DRM (Section 2.4). The design requirements of A^2DRM are therefore:

R1 To provide **consumer identity anonymity** to prevent a consumer's activities from being monitored.

R2 To provide **session accountability** to reveal a malicious consumer's identity from a previously anonymous state so that he can be held accountable.

R3 To provide **conditional consumer session unlinkability** to prevent a consumer's sessions from being linked to his civil identity illegitimately; whilst supporting legitimate traces.

R4 To provide **conditional multiple session unlinkability** to prevent a consumer's sessions from being linked together illegitimately; whilst supporting legitimate traces.

We believe that only once these requirements are satisfied will a DRM system become widely accepted.

We did not however, restrict our design to only satisfy these four requirements; we had other design considerations too. We aim to achieve, what we call, *separation of concerns*. By separating the concerns of each entity, we attempt to restrict the privacy-compromising information revealed to each entity without restricting its ability to function. For example, a consumer might only reveal his payment details to the payment gateway, rather than asking the licence broker to forward them to the payment gateway, as we discussed in Section 2.1. This way, content information is only revealed to the licence broker; whilst payment details are only revealed to the payment gateway.

Furthermore, we consider the run-time efficiency and transaction authenticity. For example, it's important to ensure that consumers do not experience a poor service and that it does not develop any bottlenecks.

For maintaining a high level of security, it's also important to ensure that a transaction is properly authenticated before it is processed; otherwise our system could experience rogue transactions, maliciously exploiting the blanket of anonymity. Composite secrets might therefore be more appropriate — *i.e.* multiple secrets that must be used in unison before it is accepted. To further this, we require that all entities within the solution are trustworthy. In saying that however, the whole solution should not become compromised if one entity is compromised. Safeguards should be built in to protect such situations.

Lastly, we must also consider the user acceptability of the solution. For a solution to become prominent, it must be accepted by consumers and providers alike. We therefore require that any content acquired using a DRM solution can be accessed on multiple compliant devices without the need for additional hardware.

5.2.2 Design Assumptions

For the design of A^2DRM we assume the following:

- The environment in which A^2DRM is placed consists of multiple CAs for which consumers are able to acquire certificates from. There exists a method in which a CA is able to verify that the items that make up a consumer's civil identity are correct and belong to the consumer.

- All entities within A^2DRM are trustworthy; although their trustworthiness could be compromised after an attack. There is no collusion between, or simultaneous misbehaviour by, any two or more of the A^2DRM entities.

- There exists an entity that operates with the utmost trust and acts in an authoritative manner. We call this entity, *The Authority*. The Authority is capable of monitoring consumers' activities and detecting malicious activities. It is the responsibility of The Authority to initiate an identity traceback after detecting malicious activities.

- There exists an entity that is capable of crediting and debiting bank accounts. We call this entity, the payment gateway. There exists a method in which the payment gateway is able to verify that the items that make up a consumer's payment details are correct and belong to the consumer. We assume that the payment gateway is capable of constructing a set of

access rights dependent on the payment made using a well defined REL (*e.g.* MPEG [WDW+05]). The payment gateway is capable of constructing a signature that reflects the amount paid. This could be realised, for example, as a set of key pairs with each one publicly bound to a particular price. Selecting the appropriate key pair to construct a signature could therefore result with a signature bound to a particular price.

- Cryptographic primitives (*e.g.* the RSA algorithm for public key cryptography) are secure. We assume that consumers are capable of constructing RSA key pairs such that decrypting a ciphertext, or generating a message's signature, without the required key is infeasible in polynomial time.

- Secure anonymous communication channels (*e.g.* Chaum's mix system) are available.

- Each piece of digital content contains a unique identifier. We assume that secure containers can only be accessed by *compliant* playback devices.

- There is a published CRL of revoked certificates called, CRL. There is a published blacklist of unacceptable pseudonyms.

- Chaum's proof of equality of discrete logarithms [CP93] is secure.

5.2.3 Building Blocks

A^{2}DRM utilises a series of building blocks; we discuss these in this section. We provide a comprehensive list of notation in Section 5.2.4. For the following building block definitions it is sufficient to know that $A \rightarrow B$: x can be read as 'A sends x to B'; and that $A \leftrightarrow B$: $f(x)$ can be read as 'together, A and B compute $f(x)$'. A comprehensive notation list is detailed in Section 5.2.4.

Hashing a message

To hash a message, we provide Function H. H is a single-party function for user U to hash a message. This could be a utilisation of the hash function proposed by Knuth [Knu98]. The function produces a digest of a larger piece of data. The function has message m as its input and returns hash value h.

Generating an RSA key pair

To generate a RSA key pair, we provide Function *key_gen*. *key_gen* is a single-party function for user U to generate a new key pair. It is a utilisation of the RSA key generation scheme [RSA78]. The function has two inputs: distinct prime numbers p and q. Using these inputs, U executes the following:

1. Compute $n = pq$.
2. Compute $\Phi = (p-1)(q-1)$.
3. Choose $1 < e < \Phi$ such that e is coprime with Φ.
4. Compute $d = e^{-1} \bmod \Phi$.
5. The public key is (n, e) and the secret key is (n, d).
6. The values d, p, q, Φ are kept secret.

On completing the above, the function returns $\{PK = (n, e), SK = (n, d)\}$.

Encrypting a messaging

To encrypt a message, we provide Function E. E is a single-party function for user U to encrypt a message. It is a utilisation of the RSA key encryption scheme [RSA78]. The function has the following inputs:

1. m: the message U wants to encrypt.

2. $PK = (n, e)$: the public key used to encrypt m.

Using these inputs, U computes ciphertext $c = m^e \bmod n$. On completion, the function returns c.

Decrypting a ciphertext

To decrypt a ciphertext, we provide Function D. D is a single-party function for user U to decrypt a message. It is a utilisation of the RSA key decryption scheme [RSA78]. The function has the following inputs:

1. c: the ciphertext U wants to decrypt.

2. $SK = (n, d)$: the secret key used to decrypt c.

Using these inputs, U computes plaintext $m = c^d \bmod n$. On completion, the function returns m.

Generating a signature

To generate an RSA signature, we provide Function *sig_gen*. *sig_gen* is a single-party function for user U to generate a signature of a message. It is a utilisation of the RSA signature generation scheme [RSA78]. The function has the following inputs:

- m: the message U wants to sign.

- $SK = (n, d)$: U's secret key.

Using these inputs, U executes the following:

1. Compute $h = H(m)$.

2. Compute $s = h^d \bmod n$.

On completing the above, the function returns s.

Authenticating a signature

To authenticate an RSA signature, we provide Function *sig_ver*. *sig_ver* is a single-party function for verifier V to verify a signature of a message. It is a utilisation of the RSA signature verification scheme [RSA78]. The function has the following inputs:

- m: the message that's been signed.
- s: the signature to be verified.
- $PK = (n, e)$: the public key used to verify the signature.

Using these inputs, V computes $h' = s^e \bmod n$. V must then verify that $h' \equiv H(m)$. If this is false, then the signature is invalid and the function returns `false`.

On the other hand, if $h' \equiv H(m)$ is true, then the signature is authentic. One can therefore assume that the signature was constructed using the secret key corresponding to $PK = (n, e)$, and that the message has integrity. The function returns `true`. Before a signature can be fully authenticated, $PK = (n, e)$ must be certified.

Authenticating a public key

To provide public key authentication, we provide Protocol *key_auth*. *key_auth* is a two-party protocol between prover P and verifier V to prove that P knows the secret key corresponding to the inputted public key. The function has the following inputs:

1. $SK = (n, d)$: the secret key, known by P only.

2. $PK = (n, e)$: the public key, known by P and V.

Using these inputs, P and V execute the following:

1. V chooses random value r.

2. $V \rightarrow P$: (r).

3. P executes: $s = sig\_gen(r, SK)$.

4. $P \rightarrow V$: (s).

On completing this protocol, V must execute $sig\_ver(s, PK)$. If this is `false`, then P has not been able to prove to V that he knows $SK = (n, d)$. The function returns `false`.

On the other hand P has has been able to prove to V that he knows $SK = (n, d)$. The function returns `true`.

Partitioning a certificate

To partition a certificate, we provide Function *cert_part*. *cert_part* is a single-party function for user U to partition a certificate. The function has one input: certificate C. C has the following fields: Version, Serial, Signature Algorithm, Issuer, Validity, Holder, Pseudonym, Key Algorithm, and Key. Using C, U executes the following:

1. U constructs: $A = (C : Version||C : Serial||C : Signature\_Algorithm||C : Key\_Algorithm)$.

2. U constructs: $B = (C : Issuer||C : Validity||C : Holder||C : Pseudonym||C : Key)$.

On completing the above, the function returns (A, B).

Reconstructing a certificate

To reconstruct a previously partitioned certificate, we provide Function *cert_reco*. *cert_reco* is a single-party function for user U to reconstruct a certificate. The function has two inputs: certificate parts A and B. Using these inputs, the function returns $(A||B)$.

Proving the equality of two discrete logarithms

To prove the equality of two discrete logarithms, we provide Protocol *equ_log*. *equ_log* is a two-party protocol between prover P and verifier V to prove the equality of two discrete logarithms:

$$\log_g h \equiv \log_{\tilde{g}} \tilde{h}, \tag{5.1}$$

where $g, h, \tilde{g}, \tilde{h}$ are know by P and V. It is a utilisation of the equality of discrete logarithms proposed by Chaum *et al.* [CP93]. The protocol also has P's secret key x as P's private input, such that $h = g^x$ and $\tilde{h} = \tilde{g}^x$. With these inputs, the following protocol is executed between P and V:

1. P chooses: r.
2. P constructs (A, B) such that:

   $A = g^r$.

   $B = \tilde{g}^r$.
3. $P \rightarrow V$: (A, B).
4. V chooses: c.
5. $V \rightarrow P$: (c).

6. P computes: $y = r + cx$.

7. $P \rightarrow V$: (y).

On completing this protocol, V must perform the following verifications:

1. $g^y \equiv Ah^c$

2. $\tilde{g}^y \equiv B\tilde{h}^c$

Now, if $g^y \equiv Ah^c$ is `true`, then y must have been computed using the same x that was used to compute h and A on the two challenges r and c. Similarly, if $\tilde{g}^y \equiv B\tilde{h}^c$ is `true`, then y must have been computed using the same x that was used to compute $\tilde{h}$ and A on the two challenges r and c. By combining these two truths, it can be deduced that the same value for x must have been used to compute both: $h = g^x$ and $\tilde{h} = \tilde{g}^x$, which implies that $\log_h g \equiv \log_{\tilde{h}} \tilde{g}$. This case would result in a successful verification. P has proved the equality of the discrete logarithms. V exits the protocol and returns `true`.

On the other hand, if the verifications are unsuccessful, P has not proved the equality of the discrete logarithms. V exits the protocol and returns `false`.

Non-interactive transcript of equality of discrete logarithms

To provide a transcript of the proof of the equality of two discrete logarithms, we provide *equ_log_trans*. Protocol *equ_log_trans* is a two-party protocol between prover P and verifier V. By using the protocol, P and V are able to generate a blinded non-interactive proof of the equality of two discrete logarithms:

$$\log_{\tilde{g}} \tilde{h} \equiv \log_g h \tag{5.2}$$

where $g, h, \tilde{g}, \tilde{h}$ are know by P and V. It is a utilisation of the blinded non-interactive, equality of discrete logarithms proof used by Lysyanskaya *et al.* [LRSW00]. The protocol also has P's secret key x as P's private input such that $h = g^x$ and $\tilde{h} = \tilde{g}^x$. With these inputs, the following protocol is executed between P and V:

1. P chooses: r.
2. P constructs (A, B) such that:

   $A = g^r$.

   $B = \tilde{g}^r$.
3. $P \rightarrow V$: (A, B).
4. V chooses: α, β.
5. V computes: $A' = Ag^{\alpha}h^{\beta}$.
6. V computes: $B' = (B\tilde{g}^{\alpha}\tilde{h}^{\beta})^{\gamma}$.
7. V computes: $c = H(A', B') + \beta$.
8. $V \rightarrow P$: (c).
9. P computes: $y = r + cx$.
10. $P \rightarrow V$ (y).

We note that *equ_log* is made non-interactive by using a sufficiently strong hash function, H, to select the verifier's challenge base on the prover's input of (A, B). On completing this protocol, V must perform the following verifications:

1. $g^y \equiv Ah^c$

2. $\tilde{g}^y \equiv B\tilde{h}^c$

Now, if $g^y \equiv Ah^c$ is `true`, then y must have been computed using the same x that was used to compute h and A on the two challenges r and c. Similarly, if $\tilde{g}^y \equiv B\tilde{h}^c$ is `true`, then y must have been computed using the same x that was used to compute $\tilde{h}$ and A on the two challenges r and c. By combining these two truths, then the same x must have been used to compute both $h = g^x$ and $\tilde{h} = \tilde{g}^x$. This implies that $\log_h g \equiv \log_{\tilde{h}} \tilde{g}$. This case would result in a successful verification and thus P has proved the equality of the discrete logarithms. Therefore $g^{y+\alpha} = Ah^c$ and $\tilde{g}^{y+\alpha} = B\tilde{h}^c$. So V exits the protocol and returns the transcript $\{(A', B'), H(A', B') + \beta, y + \alpha\}$.

On the other hand, if the verifications are unsuccessful then P has not proved the equality of the discrete logarithms. V exits the protocol and returns `false`.

We note that, for each execution, the verifier uses a randomly chosen blinding coefficient, γ. The transcript produced by *equ_log_trans* is therefore equally likely to have resulted from any $\tilde{g}$ and any choice of r and c. Consequentially, the prover will not know, anymore accurately than randomly guessing, which execution-instance the transcript refers to.

Constructing a pseudonym

To construct a pseudonym, we provide Protocol *nym_con*. *nym_con* is a two-party protocol between user U and organisation O. By using the protocol, O and U are able to construct U's pseudonym. It is a utilisation of the pseudonym construction procedure proposed by Lysyanskaya *et al.* [LRSW00]. The protocol has two inputs: g and x which make up U's secret key (x) and corresponding public key (g^x). With these inputs, the following protocol is executed between U and O:

1. U chooses a random value γ.
2. U constructs $(\tilde{a}, \tilde{b})$ such that:

   $\tilde{a} = g^{\gamma}$.

   $\tilde{b} = \tilde{a}^{x}$.
3. $U \rightarrow O$: $(\tilde{a}, \tilde{b})$.
4. O chooses a random value r.
5. O computes $a = \tilde{a}^{r}$
6. $O \rightarrow U$: (a).
7. U computes: $b = a^{x}$.
8. $U \rightarrow O$: (a, b).

On completing the above, O verifies that U used the same value of x to compute both $\tilde{b} = \tilde{a}^{x}$ and $b = a^{x}$. Consequentially, U proves to O that he indeed knows x. This is achieved by U proving to O that:

$$\log_{a} b \equiv \log_{\tilde{a}} \tilde{b}. \tag{5.3}$$

This can be achieved by executing $equ\_log(a, b, \tilde{a}, \tilde{b})$. If this returns `false`, then the pseudonym construction is invalid and O does not issue U with a pseudonym. If `true` is returned, then the pseudonym construction is valid and O issues U with pseudonym $P = (a, b)$.

Constructing a credential

To construct a credential, we provide Protocol *cred_con*. *cred_con* is a two-party protocol between user U and organisation O. By using the protocol, O and U are

able to construct U's credential. It is a utilisation of the credential construction procedure proposed by Lysyanskaya *et al.* [LRSW00]. The protocol has the following inputs:

- O's secret credential key: (S_1, S_2).
- O's public credential key: (g, h_1, h_2), where $h_1 = g^{S_1}, h_2 = g^{S_2}$.
- U's pseudonym with O: $P = (a, b)$, where $b = a^x$.

With these inputs, the following protocol is executed between O and U:

1. $U \rightarrow O$: $P = (a, b)$.
2. O constructs (A, B) such that:

   $A = b^{S_2}$.

   $B = (ab^{S_2})^{S_1}$.
3. $O \rightarrow U$: (A, B).

Once this has been executed, U chooses a secret blinding value γ and, with O, constructs two non-interactive blinded proofs to prove that (A, B) was constructed by using (a, b):

- T_1: proves $\log_b A \equiv \log_g h_2$. It's constructed by executing: $equ\_log\_trans(b, A, g, h_2, \gamma)$.
- T_2: proves $\log_{aA} B \equiv \log_g h_1$. It's constructed by executing: $equ\_log\_trans(aA, B, g, h_1, \gamma)$.

U then blinds a, b, A, B with γ, and constructs credential $C = (a^\gamma, b^\gamma, A^\gamma, B^\gamma, T_1, T_2)$.

Verifying a credential

To verify a credential, we provide Protocol *cred_ver*. *cred_ver* is a two-party protocol between user U and organisation O'. By using the protocol, O' is able to verify U's credential was issued by organisation O. It is a utilisation of the credential construction procedure proposed by Lysyanskaya *et al.* [LRSW00]. The protocol has the following inputs:

- O's public credential key: (g, h_1, h_2), where $h_1 = g^{S_1}, h_2 = g^{S_2}$.
- U's secret key: x, known only by U.
- U's pseudonym with O': $P = (\tilde{a}, \tilde{b})$, where $\tilde{b} = \tilde{a}^x$.
- U's credential from O: $C = (a^\gamma, b^\gamma, A^\gamma, B^\gamma, T_1, T_2)$.

To verify credential C, O' must perform the following verifications:

1. $\log_{\tilde{a}} \tilde{b} \equiv \log_{a^\gamma} b^\gamma$: to verify that the same value for x was used to construct pseudonym $(\tilde{a}, \tilde{b})$ and blinded pseudonym (a^γ, b^γ) — we apply *equ_log* with U's private key to achieve this.
2. $\log_{b^\gamma} A^\gamma \equiv \log_g h_2$: to verify that A^γ was constructed using b^γ and O's secret credential key — we apply *equ_log* with O's private credential key to achieve this.
3. $\log_{a^\gamma A^\gamma} B^\gamma \equiv \log_g h_1$: to verify that B^γ was constructed using a^γ and O's secret credential key — we apply *equ_log* with O's private credential key to achieve this.

Note: by revealing a credential twice, the verification procedure is executed twice. This means that Function *equ_log* must be executed twice on the same

credential. By asking P two different challenges (c_1, c_2) on the same credential, V is able to extract P's secret value x by solving:

$$\begin{aligned} y_1 &= r + c_1 x \\ y_2 &= r + c_2 x \end{aligned} \tag{5.4}$$

5.2.4 Notation

Table 5.1 denotes the notation we use throughout the remainder of this thesis.

Table 5.1: A^{2}DRM protocol notation.

Actors & Groups	
CC	content consumer.
CA_i	certificate authority i.
CD_i	content distributor i.
PG_i	payment gateway i.
LB_i	licence broker i.
PD_i	playback device i.
DRM	group of DRM entities.
AUX	group of auxiliary, non-DRM entities.

Keys, Identifiers and Certification

PK_e	public key for entity e, constructed using key_gen.
SK_e	private, secret key for entity e, constructed using key_gen.
id_i	the civil identity of i.
P_i	pseudonym i, constructed using nym_cons.
C_i	certificate i.
C_i^n	part n of certificate C, constructed using $cert\_part$.
$C : attr = x$	attribute $attr$ in certificate C is assigned the value x.

Functions & Operators

$E_k(m)$	encryption of message m using key k, acquired by $E(m, k)$.
$D_k(m)$	decryption of message m using key k, acquired by $D(m, k)$.
$S_k(m)$	signature of message m using key k, acquired by $sig\_gen(m, k)$.
$A \rightarrow B$: x	A sends x to B.
$A \leftrightarrow B$: $f(x)$	together, A and B compute $f(x)$.
$\|\|$	read as 'concatenated to'.

5.3 The A^{2}DRM Protocol

In this section we present A^{2}DRM: an accountable anonymity service in DRM. It provides DRM consumers (*e.g.* Alice) with accountable anonymity using pseudonymity, identity escrow, identity certificates, authorisation certificates, digital signatures and RELs. We give an overview in Section 5.3.1 and a detailed description in Section 5.3.2.

5.3.1 A^{2}DRM Overview

A^{2}DRM is built on a similar DRM model to that discussed in Section 2.1. The set of A^{2}DRM entities are: consumers (*e.g.* Alice), Certification Authority (CA), and DRM entities (*i.e.* the content distributor, the licence broker, the payment gateway, and the playback device). Here however, when communicating with another entity, Alice identifies herself with a pseudonym issued by a CA. By using a different, unlinkable pseudonym for every transaction she partakes, her

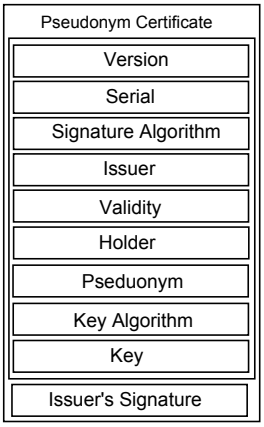

Figure 5.1: A pseudonym certificate.

transactions are unlinkable. We do however provide a conditional link between a pseudonym and the holder's identity so we can provide *accountable anonymity*.

We now give a brief overview of how Alice may use our system to acquire and access a piece of digital content. We first assume that her identity should be conditionally hidden; and then explain how her anonymity can be revoked to hold her accountable for a malicious action.

A^{2}DRM Setup

In A^{2}DRM, a unique pseudonym that Alice uses as an identifier is bound to a public key by an identity certificate; encryption, decryption and signatures can thus be credited to a pseudonym. We call these certificates, *pseudonym certificates*.

A pseudonym certificate, as illustrated in Figure 5.1, is structurally similar to that of an X.509 identity certificate (as discussed in Section 3.3). It consists of a series of fields detailing the pseudonym, the public key, and certificate itself. To prove its authenticity, it's signed by the issuer. Entities that issue pseudonym/certificate pairs are called, CAs. By allowing Alice to use the pseudonym during

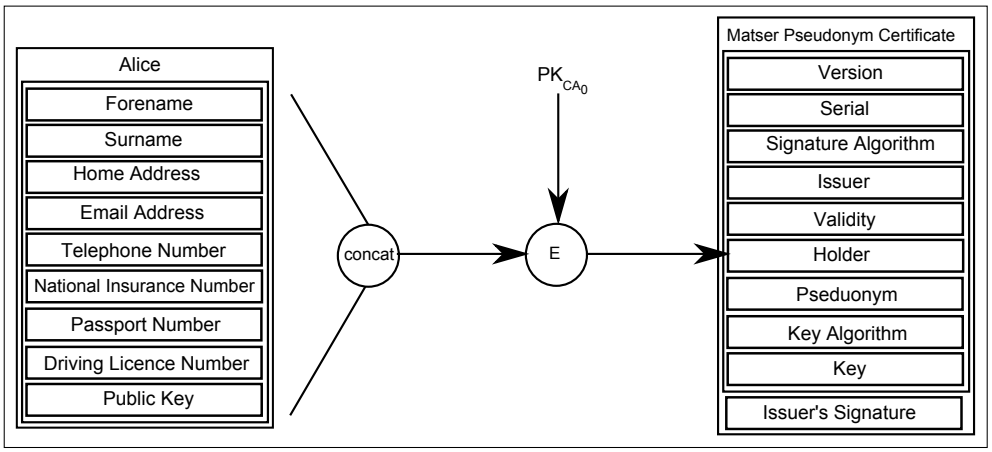

Figure 5.2: The link between Alice's civil identity and her pseudonym. Her identity is stored in the corresponding certificate's *Holder* field. It is encrypted so that it can only be revealed by the issuer, CA_0.

identification processes, and its corresponding public key in authentication processes, she can remain anonymous. To ensure that this anonymity cannot be exploited for malicious activities, each pseudonym certificate holds a conditional link to its holder's civil identity.

Each pseudonym certificate has a *Holder* field that provides the conditional link to its holder. Before a CA issues a certificate to Alice, she must reveal her civil identity. When the CA (*e.g.* CA_0) constructs the certificate, it inserts an encryption of Alice's identity into *Holder*, as depicted in Figure 5.2. This way, it is only the CA that is able to trace the holder's civil identity from a pseudonym certificate. If it has been revealed that Alice has been acting maliciously, then The Authority can co-ordinate with the CA to trace Alice's civil identity. She can then be held accountable for her malicious activities.

We do not however, restrict Alice to dealing with a single, specific CA; Alice can choose any CA to acquire her pseudonym/certificate pair from. Without any meaningful competition, the system would inevitably lack value to Alice. However, the problem with such an ideology is that illegitimate entities could

come to the fore. To prevent Alice from revealing identifying information to such entities, CAs only provide certificates that can be read by trusted entities. This way, Alice cannot certify her pseudonym to untrusted entities.

CAs that have proven themselves to be honest are members of a group that know a group key pair, similar to group signature schemes discussed in Section 3.2. By encrypting the certificate with a group public key, untrusted entities, that are not members of the group, are unable to read it. So that the consumers are not issued with random, unintelligible data structures, CAs first partition the certificate into two segments: one that contains non-identifying information; and another that contains identifying information. By encrypting the segment containing the identifying information with the group public key, we can be assured that only trusted entities can learn the whole certificate. Therefore only trusted entities are able to certify pseudonyms.

Although this provides added security, the CA, that knows Alice's civil identity *and* her pseudonym, is able to monitor her activities. To complicate the relationship between her civil identity and her pseudonym, we allow Alice to use an already acquired pseudonym to acquire another pseudonym. By selecting a different CA to acquire her new pseudonym from, Alice reduces the likelihood of her identity from being maliciously revealed; both CAs would have to be compromised in order to achieve this. We call the first pseudonym that Alice acquires as a result of revealing her civil identity, a *master pseudonym*. We call a pseudonym acquired as a result of revealing a master pseudonym, a *session pseudonym*; and pseudonyms acquired from session pseudonyms, *transaction pseudonyms*. Pseudonym acquisition requests from consumers identifying themselves with transaction pseudonyms are rejected. Transaction pseudonyms are used as identifiers during transactions; master and session pseudonyms are not.

However, to acquire a session pseudonym/certificate pair, Alice does not reveal

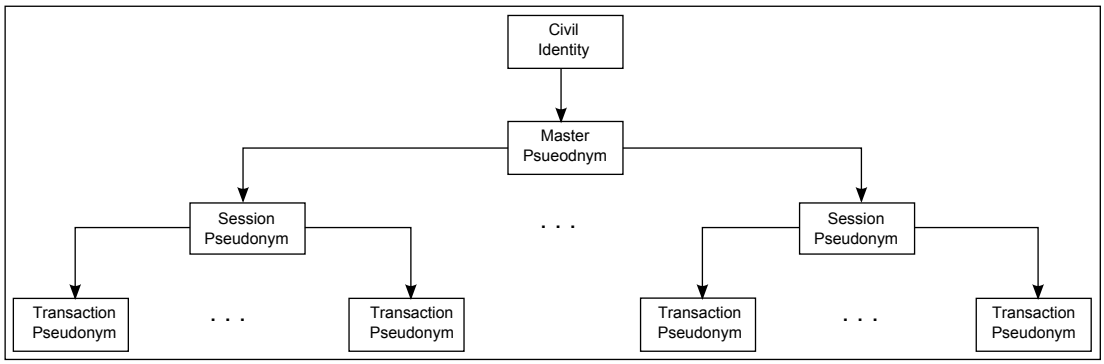

Figure 5.3: Pseudonym tree.

her civil identity to the CA. Instead, she reveals her master pseudonym/certificate pair. From $Holder$, the CA can extract an encryption of Alice's identity. Instead of encrypting Alice's civil identity, the CA encrypts the value in $Holder$. This encryption is then inserted into the session certificate. Now The Authority must co-operate with two CAs to reveal Alice's civil identity: the CA that issued the session certificate; and the CA that issued the master certificate. A similar process is followed by the CA that issues the transaction certificate so that The Authority must co-ordinate with three CAs to reveal Alice's identity.

As a result of this behaviour, the relationship between Alice's multiple pseudonyms can be represented as a tree, as illustrated in Figure 5.3; we call such a tree, a *pseudonym tree*. The leaves of the tree are the transaction pseudonyms; whilst the root is the civil identity. To reveal the link between a transaction pseudonym and the corresponding civil identity it would require The Authority to trace an entire branch with the co-operation of three CAs. For this to be achieved without the consent of The Authority, would require all three CAs to be compromised. The likelihood of this occurring is minimal.

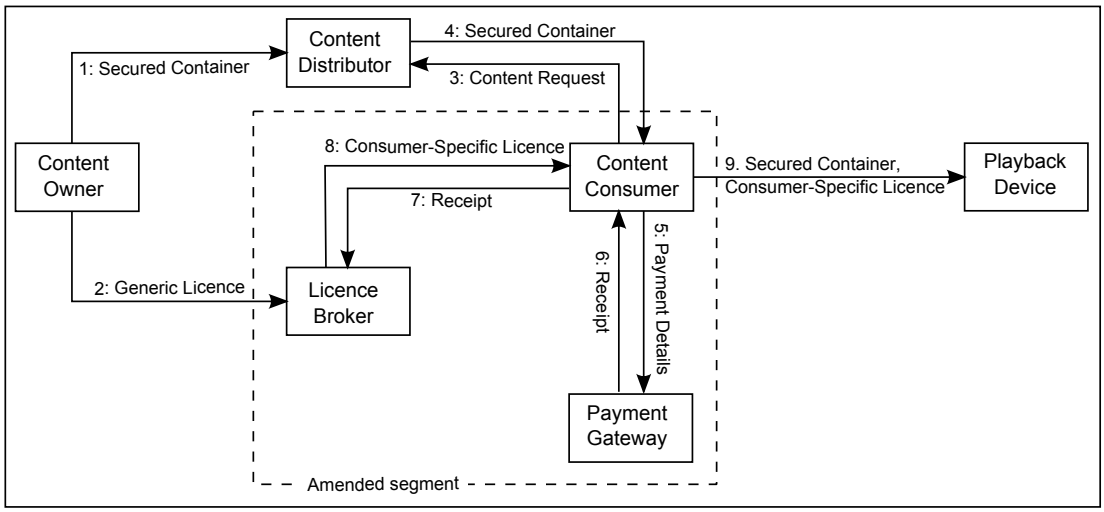

Figure 5.4: Amended DRM model. The section of the DRM model we have amended is enclosed within the dotted area.

A^{2}DRM Transactions

To preserve the consumers' privacy, we have amended the DRM model discussed in Section 2.1 to neatly fit our requirements; we illustrate this in Figure 5.4. Here, the consumer performs licence payment and licence acquisition separately. For payment, the consumer only deals with the payment gateway; whilst for licence acquisition, the consumer only deals with the licence broker. In this way, the consumer's payment details (*e.g.* credit/debit card details *etc.*) can remain hidden from the licence broker; whilst the consumer's licence and access rights can remain hidden from the payment gateway.

To preserve her privacy, Alice identifies herself with a different, unlinkable, transaction pseudonym for each entity she communicates with. She therefore uses a different transaction pseudonym to identify herself with the licence broker (P_1) from the one she uses to identify herself with the payment gateway (P_2). Both P_1 and P_2 have an associated transaction certificate to certify their respective pseudonym.

Now, before the licence broker will issue Alice with a licence, she must prove

that she has indeed paid for it. So, after a successful payment, Alice obtains a receipt of payment from the payment gateway. Alice can then present this receipt to the licence broker as a proof of payment. A receipt of payment takes the form of a credential — as proposed by Lysyanskaya *et al.* [LRSW00].

The receipt of payment is essentially a blinded signature of a blinded pseudonym. To issue such a receipt, the payment gateway constructs a signature of P_1, and issues it to Alice. The signature is constructed using the payment gateways secret key and P_1. Alice then constructs the credential by blinding both the pseudonym and the payment gateway's signature. The blinding property prevents the signature from being linked to its construction. In other words, the licence broker is unable to link the blinded signature to the unblinded signature, even when colluding with the payment gateway.

When supplied with such a receipt, the licence broker must ensure that:

1. the blinded pseudonym was constructed using the same secret key as P_2;

2. the blinded signature was constructed using the payment gateway's secret key, and the blinded pseudonym.

This therefore means that for a consumer to use a receipt of payment, she must ensure that she constructs both P_1 and P_2 with the same key pair.

Now, if both these statements are true, then the licence broker is assured that Alice has indeed paid for the licence. Even if the licence broker colludes with the payment gateway, the blinding property prevents them from linking Alice's payment details to her licence details. The licence broker then issues Alice with the corresponding licence. We assume that the payment gateway holds a set of key pairs; with each pair reflecting a different amount paid by Alice. The licence broker then knows which access rights Alice has over the content. Alice can then

use her licence to access the content within the secure container she acquired from a content distributor.

A^{2}DRM Identity Traceback

Under exceptional circumstances (*e.g.* after the detection of malicious intent), it is important that The Authority is able to trace a malicious consumer's identity, so that he can be held accountable. The ability to revoke anonymity lies with the ability to reveal a consumer's pseudonym tree. Revealing a branch of the tree would allow The Authority to reveal the link between a transaction pseudonym and Alice's civil identity.

When The Authority detects a malicious transaction, it co-ordinates with the corresponding DRM entity to reveal the associated pseudonym certificate. The Authority then co-ordinates with the certificate's issuer to decrypt $Holder$. This reveals another encryption of the consumer's identity — generated by the CA that issued the session pseudonym pair. The Authority co-ordinates with this CA to decrypt it. A similar process is performed with the CA that issued to master pseudonym to reveal the consumer's civil identity. Only after all three CAs have co-operated and decrypted their respective ciphertexts, will Alice's civil identity be revealed. The Authority can then hold her accountable for her actions.

As a consequence of tracing Alice's civil identity, her master pseudonym is revealed. This can be placed on a blacklist of unacceptable pseudonyms to prevent her from continuing to use the system. By broadcasting updates of the blacklist, all CAs can verify that they have not issued pseudonyms to consumers identifying themselves with a blacklisted pseudonym. Pseudonyms that have been issued to consumers who have since been blacklisted, are also placed on the blacklist. To prevent the blacklist from growing indefinitely, pseudonyms are removed once

they reach their expiration date.

5.3.2 A^{2}DRM in Depth

This section focusses on the design of the A^{2}DRM protocol; we provide a detailed explanation of its inner workings. To aid our explanation, we refer the reader back to our example scenario in Chapter 2. Here, consumer Alice wished to acquire the Die Hard tetralogy from a DRM service without compromising her privacy. The A^{2}DRM protocol provides a way for Alice to achieve this. In Figure 5.5 we depict the transactions Alice must partake to acquire and access the Die Hard movies using the A^{2}DRM protocol.

In order to operate, the A^{2}DRM protocol utilises a series of entities. All entities involved in the DRM processes (*i.e.* the content distributor CD, licence broker LB, payment gateway PG, and playback device PD) are members of a group, DRM. The entities used to provide, maintain and revoke consumer anonymity (*i.e.* CAs) are members of a different group, AUX. More precisely:

- $\{CD, PG, LB, PD\} \in DRM$;
- $\{CA_0, CA_1, CA_2\} \in AUX$.

To prevent the DRM entities from colluding with the AUX entities to learn information outside their operational scope, the following condition should be enforced:

$$|\{CD, PG, LB, PD, CA_0, CA_1, \ldots, CA_n\}| = n + 5. \tag{5.5}$$

In other words, all entities within A^{2}DRM should be different. Both DRM and AUX have their keys pairs ($\{PK_{DRM}, SK_{DRM}\}$, and $\{PK_{AUX}, SK_{AUX}\}$ respectively) in accordance to group signature schemes (discussed in Section 3.2) using, Chaum's simple group signature scheme [CvH91], for example.

The A^{2}DRM protocol is realised with a set of procedures. Depending on the particular instance of the protocol, the procedures may be executed a different number of times in a different order. One particular example instance is depicted in Figure 5.5.

In Figure 5.5, Alice uses Procedure 1 to acquire her master pseudonym/credential pair. She then uses Procedure 2 to acquire a session pseudonym/credential pair and two transaction pseudonym/credential pairs; $2A$, $2B$ and $2C$ represent three different instances of Procedure 2. Alice then acquires the secure container using Procedure 3. To pay for the licence, she uses Procedure 4 and then acquires the licence using Procedure 5. She then accesses the content within the container using Procedure 6. Under exceptional circumstances (*e.g.* if Alice acts maliciously) her identity can be traced using Procedure 7 so that she can be held accountable. Procedure 7 is only used during identity traceback; it is not used during a typical cycle of A^{2}DRM[1]. We have therefore refrained from depicting Procedure 7 on Figure 5.5. We now provide a detailed explanation of these procedures.

Procedure 1: Acquiring a master pseudonym/certificate pair

Procedure 1 is used by correctly identified consumer Alice to acquire a master pseudonym/certificate pair from certificate authority CA_0. It, and indeed the A^{2}DRM protocol, is initiated when Alice makes a master pseudonym/certificate pair request (*e.g.* $M1$ in Figure 5.5) to CA_0 consisting of the following:

- PK_0: the public key that Alice selects to be bound to her master pseudonym. To generate PK_0, and indeed corresponding secret key SK_0, she executes $key\_gen(p, q)$ with two randomly chosen distinct prime numbers p and q.

[1]To successfully terminate, Procedure 7 requires the co-operation of all entities involved. This requires multiple communications. Adding it to Figure 5.5 would complicate it.

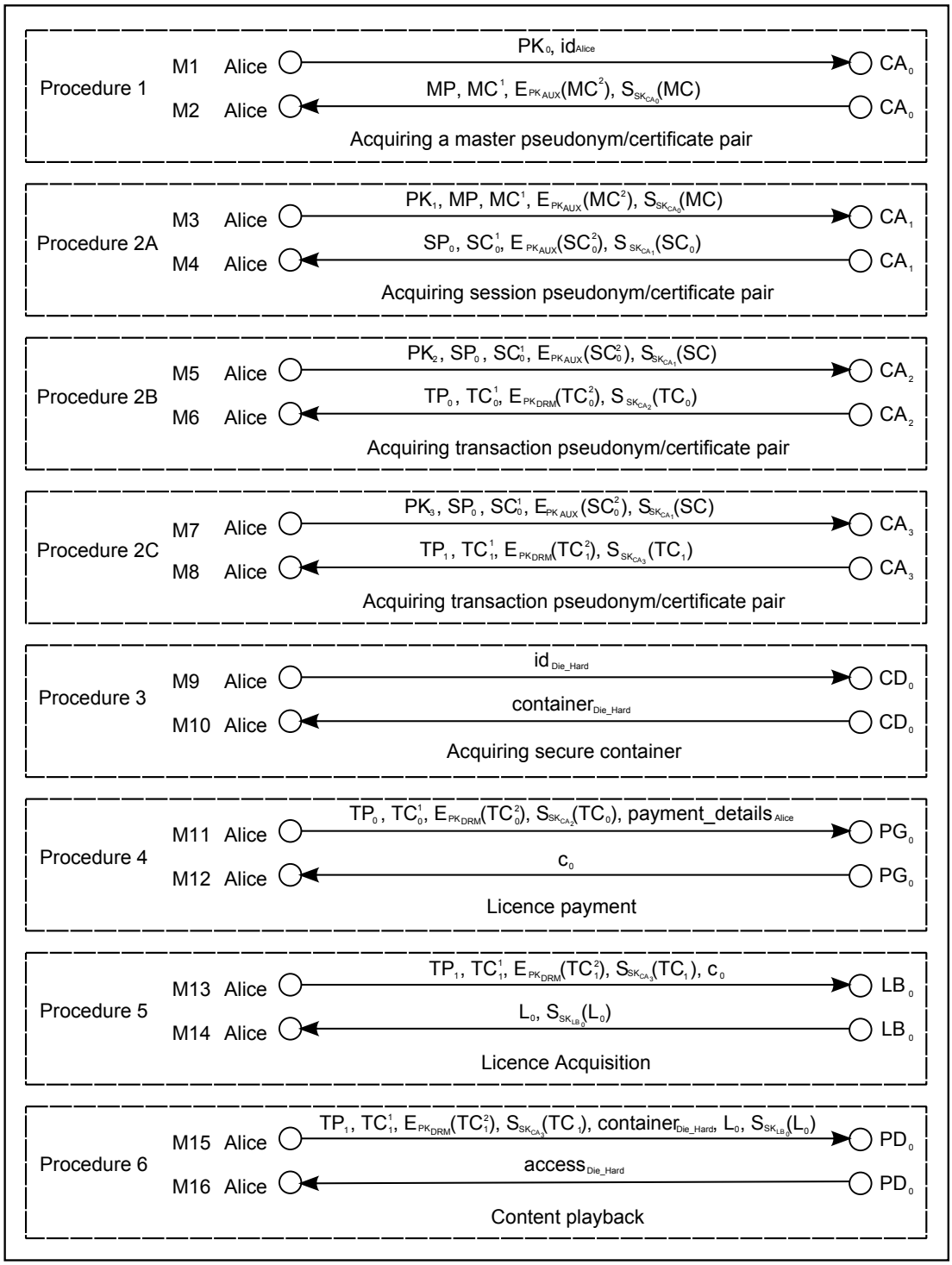

Figure 5.5: A^{2}DRM protocol messages.

- id_{Alice}: a concatenation of the items that make up Alice's civil identity.

Upon receiving the request, CA_0 must carry out the following verifications:

- execute $key\_auth(PK_0)$ to verify that Alice knows the secret key corresponding to PK_0.
- we assume that there is a method in which CA_0 can verify that the items that make up her identity do indeed belong to Alice.

If these verifications are unsuccessful, CA_0 notifies The Authority and exits the procedure. If the verifications are successful, CA_0 is assured[2] that Alice has revealed *her* civil identity. Alice then randomly selects g and x, and with the help of CA_0, executes $nym\_con(x, g^x)$ to generate master pseudonym MP. CA_0, known by public key PK_{CA_0} and corresponding secret key SK_{CA_0}, then constructs MP's corresponding master certificate MC such that:

- $MC : Pseudonym = MP$;
- $MC : Holder = E_{PK_{CA_0}}(id_{Alice})$;
- $MC : Key = PK_0$;
- $MC : Issuer = CA_0$;
- $MC : Level = 0$;
- $MC : Validity = d$, the date that CA_0 selects for MC to expire.

For possible revocation purposes, CA_0 encrypts the concatenation of items that make up Alice's civil identity with its public key, PK_{CA_0}, and inserts it into $MC : Holder$. Under exceptional circumstances, The Authority is thus able

[2]In this setting, assurance is attained when the entity in question is convinced it can proceed to the next step of the protocol.

to ask CA_0 to decrypt $MC : Holder$ to reveal Alice's civil identity. CA_0 then partitions MC, by executing $cert\_part(MC)$, before issuing Alice with:

- MP: Alice's master pseudonym.
- MC^1: the first partition of MP's corresponding certificate; this is in its plaintext form.
- $E_{PK_{AUX}}(MC^2)$: the second partition of MP's corresponding certificate, encrypted by executing $E(MC^2, PK_{AUX})$.
- $S_{SK_{CA_0}}(MC)$: CA_0's signature on MC. Its purpose is two-fold: to prove that MC was indeed issued by CA_0; and to prove the integrity of MC. CA_0 generates the signature by executing $sig\_gen(MC, SK_{CA_0})$.

Procedure 2: Acquiring session/transaction pseudonym/certificate pair

A session pseudonym/certificate pair is similar to a transaction pseudonym/certificate pair. We therefore provide a single procedure for a consumer to acquire a *generic* pseudonym/certificate pair. A generic pseudonym/certificate pair can take the form of either a session pseudonym/certificate pair or a transaction pseudonym/certificate pair. There are only subtle differences between the two. The *generic* procedure we discuss varies slightly, depending on which type of pseudonym/certificate pair is requested; these will be highlighted when they occur.

Procedure 2 is used by consumer Alice to acquire a generic pseudonym/certificate pair from certificate authority CA_j. Let's assume that Alice has already executed a pseudonym acquisition procedure with CA_i to acquire a generic pseudonym GP_i and generic certificate $(GC_i^1,\ E_{PK_{AUX}}(GC_i^2))$ — this may be a

master pseudonym/certificate or a session pseudonym/certificate pair. The certificate is bound to public key PK_i with corresponding secret key SK_i. Procedure 2 is initiated when Alice makes a generic pseudonym/certificate pair request to CA_j consisting of the following:

- PK_j: the public key that Alice selects to be bound to her pseudonym. To generate PK_j, and indeed corresponding secret key SK_j, she executes $key\_gen(p, q)$ with two randomly chosen distinct prime numbers p and q.
- GP_i: the previously acquired pseudonym Alice identifies herself with. This may be master pseudonym MP if she is acquiring a session pseudonym; or session pseudonym SP_0 if she is acquiring a transaction pseudonym.
- GC_i^1: plaintext part of GP_i's corresponding certificate. $GC_i : Level$ states whether it's a session certificate or transaction certificate.
- $E_{PK_{AUX}}(GC_i^2)$: encrypted part of GP_i's corresponding certificate. CA_j is able to decrypt this by executing $D(E_{PK_{AUX}}(GC_i^2), SK_{AUX})$. CA_j then reconstructs GC_i by executing $cert\_reco(GC_i^1, GC_i^2)$.
- $S_{SK_{CA_i}}(GC_i)$: CA_i's signature on GC_i. Its purpose is two-fold: to prove that CG_i was indeed issued by CA_i; and to prove the integrity of CG_i.

Upon receiving the request, CA_j must perform the following verifications:

- execute $key\_auth(PK_j)$ to verify that Alice knows the secret key corresponding to PK_j.
- execute $key\_auth(GC_i : Key)$ to verify that Alice knows the secret key corresponding to $GC_i : Key$.
- $sig\_ver(GC_i, S_{SK_{CA_i}}(GC_i), PK_{CA_i})$: to verify that the certificate's signature is correct.

- $GC_i : Issuer \in AUX$: to verify that the certificate's issuer is a trusted one.
- $GC_i : Validity >= NOW()$: to verify that the certificate is in its validity date.
- $GC_i : Pseudonym \equiv GP_i$: to verify that the certificate belongs to the pseudonym Alice identified herself with.
- $0 <= GC_i : Level < 2$: to verify that the pseudonym Alice identified herself with is a session or a transaction pseudonym.
- $GC_i \notin CRL$: to verify that the certificate has not been revoked.

If these verifications are unsuccessful, CA_j notifies The Authority and exits the procedure. If the verifications are successful, CA_j is assured that Alice has supplied a valid pseudonym such that $GC_i : Issuer$ can reveal her identity. Alice then randomly selects g and x, and with the help of CA_j, executes $nym\_con(x, g^x)$ to generate generic pseudonym GP_j. CA_j, known by PK_{CA_j} and corresponding generic secret key SK_{CA_j}, then constructs GP_j's corresponding certificate GC_j such that:

- $GC_j : Pseudonym = GP_j$;
- $GC_j : Holder = E_{PK_{CA_j}}(GC_i : Holder || GC_i : Issuer)$;
- $GC_j : Key = PK_j$;
- $GC_j : Issuer = CA_j$;
- $GC_j : Level = GC_i : Level + 1$;
- $GC_j : Validity = d \leq GC_i : Validity$, where d is the date that CA_j selects for GC_j to expire.

For possible revocation purposes, CA_j encrypts the concatenation of $GC_i : Holder$ and $GC_i : Issuer$ with its public key, PK_{CA_j}, and inserts it into $GC_j : Holder$. Under exceptional circumstances, The Authority is able to ask CA_j to decrypt $GC_j : Holder$ to reveal an encryption of Alice's civil identity. CA_j then partitions GC_j, by executing $cert\_part(GC_j)$, before issuing Alice with:

- GP_j: Alice's generic pseudonym.
- GC_j^1: the first partition of GP_j's corresponding certificate; this is in its plaintext form.
- **if** $GC_j : Level \equiv 1$ **then**:
  $E_{PK_{AUX}}(GC_j^2)$: the second partition of GP_j's corresponding certificate, encrypted by executing $E(GC_j^2, PK_{AUX})$.
- **else if** $GC_j : Level \equiv 2$ **then**:
  $E_{PK_{DRM}}(GC_j^2)$: the second partition of GP_j's corresponding certificate, encrypted by executing $E(GC_j^2, PK_{DRM})$.
- $S_{SK_{CA_j}}(GC_j)$: CA_j's signature on GC_j. Its purpose is two-fold: to prove that GC_j was indeed issued by CA_j; and to prove the integrity of GC_j. CA_j generates the signature by executing $sig\_gen(CG_j, SK_{CA_j})$.

Procedure 3: Acquiring secure container

Procedure 3 is used by content consumer Alice to acquire a secured container of a particular piece of content from content distributor CD_0. It is initiated when Alice makes a request to CD_0 (*e.g.* $M9$ in Figure 5.5) consisting of $id_{content}$: the published identifier of the content that Alice wishes to access. Following on from our example scenario in Chapter 2, this may be id_{Die_Hard}. CD_0 then supplies Alice with secure container $container_{Die\_Hard}$.

Only consumers issued with a licence are able to access the content sealed in the secure container. Acquiring the secure container can therefore be achieved anonymously. One could imagine a simple implementation emanating from a simple File Transfer Protocol (FTP)[3] or HyperText Transfer Protocol (HTTP)[4] site from which consumers could download the secure containers from. The inherent identifying information that consumers transmit using protocols such as the Internet could be reduced by using an anonymous communication channel (*e.g.* Chaum's [Cha81] mix system). Malicious consumers, hoping to exploit the anonymity provided here, will be unable to access the content without the decryption key contained within a valid licence.

Procedure 4: Licence payment

Procedure 4 is used by consumer Alice to make a payment to payment gateway PG_0. In return, PG_0 issues Alice with a receipt of payment; this takes the form of a credential. The procedure is initiated when Alice makes a payment request to PG_0 (*e.g.* $M11$ in Figure 5.5) consisting of the following:

- TP_0: transaction pseudonym that Alice identifies herself with.
- TC_0^1: plaintext part of TP_0's corresponding certificate.
- $E_{PK_{DRM}}(TC_0^2)$: encrypted part of TP_0's corresponding certificate. PG_0 is able to decrypt this by executing $D(E_{PK_{DRM}}(TC_0^2), SK_{DRM})$. PG_0 then reconstructs TC_0 by executing $cert\_reco(TC_0^1, TC_0^2)$.
- $S_{SK_{TC_0:Issuer}}(TC_0)$: $TC_0 : Issuer$'s signature on TC_0. Its purpose is twofold: to prove that TC_0 was indeed issued by $TC_0 : Issuer$; and to prove

[3]FTP is a standard Transmission Control Protocol and Internet Protocol (TCP/IP) communication protocol for transferring files between networked machines.

[4]HTTP is a standard TCP/IP communication protocol for distributed hypermedia information systems.

the integrity of TC_0.

- $payment\_details_{Alice}$: a concatenation of the items that make up Alice's payment details. These may be credit/debit card details, for example.

Upon receiving the request, PG_0 must perform the following verifications:

- execute $key\_auth(TC_0 : Key)$ to verify that Alice knows the secret key corresponding to $TC_0 : Key$.
- $TC_0 : Issuer \in AUX$: to verify $TC_0 : Issuer$ is trusted.
- $TC_0 : Validity >= NOW()$: to verify that TC_0 is in its validity date.
- $TC_0 : Pseudonym \equiv TP_0$: to verify that TC_0 belongs to TP_0.
- $TC_0 : Level \equiv 2$: to verify that TP_0 is a transaction pseudonym.
- $TC_0 \notin CRL$: to verify that TC_0 has not been revoked.
- $sig\_ver(TC_0, S_{SK_{TC_0:Issuer}}(TC_0), PK_{TC_0:Issuer})$: to verify that TC_0's signature is correct.
- we assume there is a method in which PG_0 can verify that the items that make up Alice's payment details as being correct, and belonging to Alice.

If these verifications are unsuccessful, PG_0 notifies The Authority and exits the procedure. If the verifications are successful, PG_0 is assured that Alice has supplied a valid pseudonym and must have thus revealed her identity to a trusted member of AUX. Alice then makes a payment.

To make a payment, Alice reveals her payment details to PG_0. This may include her credit/debit card details and her billing address. After a successful payment, PG_0 constructs a credential as a receipt of payment by executing $C_0 =$

$cred\_con(SK_{PG_0}, PK_{PG_0}, TP_0)$ with Alice. The resultant credential, C_0, is stored by Alice.

Procedure 5: Licence Acquisition

Procedure 5 is used by consumer Alice to acquire a licence from licence broker LB_0. It is initiated when Alice makes a licence request to LB_0 (*e.g.* $M13$ in Figure 5.5) consisting of the following:

- TP_1: transaction pseudonym that Alice identifies herself with.
- TC_1^1: plaintext part of TP_1's corresponding certificate.
- $E_{PK_{DRM}}(TC_1^2)$: encrypted part of TP_1's corresponding certificate. LB_0 is able to decrypt this by executing $D(E_{PK_{DRM}}(TC_1^2), SK_{DRM})$. LB_0 then reconstructs TC_1 by executing $cert\_reco(TC_1^1, TC_1^2)$.
- $S_{SK_{TC_1:Issuer}}(TC_1)$: $TC_1 : Issuer$'s signature on TC_1. Its purpose is two-fold: to prove that TC_1 was indeed issued by $TC_1 : Issuer$; and to prove the integrity of TC_1.
- C_0: the credential Alice received as a result of executing Procedure 4. It is her receipt of payment issued by payment gateway PG_0.

Upon receiving the request, LB_0 must perform the following verifications:

- execute $key\_auth(TC_1 : Key)$ to verify that Alice knows the secret key corresponding to $TC_1 : Key$.
- $TC_1 : Issuer \in AUX$: to verify that $TC_1 : Issuer$ is trusted.
- $TC_1 : Validity >= NOW()$: to verify that TC_1 is in its validity date.
- $TC_1 : Pseudonym \equiv TP_1$: to verify that TC_1 belongs to TP_1.

- $TC_1 : Level \equiv 2$: to verify that TP_1 is a transaction pseudonym.
- $TC_1 \notin CRL$: to verify that TC_1 has not been revoked.
- $sig\_ver(TC_1, S_{SK_{TC_1:Issuer}}(TC_1), PK_{TC_1:Issuer})$: to verify that TC_1's signature is correct.
- executes $cred\_ver(PK_{PG_0}, SK_{TP_1}, TP_1, C_0)$ with Alice to verify that the credential corresponds to TP_1 and is correctly constructed. Note: we call the secret key corresponding to public key $TC_1 : Key$, SK_{TP_1}.

If these verifications are unsuccessful, LB_0 notifies The Authority and exits the procedure. If the verifications are successful, LB_0 is assured that Alice has supplied a valid pseudonym and must have thus revealed her identity to a trusted member of AUX. LB_0 then constructs licence L_0 such that:

- $L_0 : Holder = TP_1$;
- $L_0 : Key = E_{DRM}(k)$, decryption key k used to unlock the secure container;
- $L_0 : Issuer = LB_0$;
- $L_0 : Validity = d \leq TC_1 : Validity$, where d is the date that CA_j selects for GC_j to expire;
- $L_0 : Rights = r$, the access rights granted as a result of showing C_0.

LB_0 then issues Alice with licence L_0 and signature $S_{SK_{LB_0}}(L_0)$. The signature's purpose is two-fold: to show that L_0 was issued by LB_0; and to prove the integrity of L_0. We assume the payment gateway is capable of constructing a signature, and thus a licence, that reflects the amount paid.

Procedure 6: Content playback

Procedure 6 is used by consumer Alice to access the content in secure container $container_{id}$ using playback device PD_0. It is initiated when Alice makes an access request to PD_0 (*e.g.* $M15$ in Figure 5.5) consisting of the following:

- TP_1: transaction pseudonym that Alice identifies herself with.
- TC_1^1: plaintext part of TP_1's corresponding certificate.
- $E_{PK_{DRM}}(TC_1^2)$: encrypted part of TP_1's corresponding certificate. PD_0 is able to decrypt this by executing $D(E_{PK_{DRM}}(TC_1^2), SK_{DRM})$. PD_0 then reconstructs TC_1 by executing $cert\_reco(TC_1^1, TC_1^2)$.
- $S_{SK_{TC_1:Issuer}}(TC_1)$: $TC_1 : Issuer$'s signature on TC_1. Its purpose is two-fold: to prove that TC_1 was indeed issued by $TC_1 : Issuer$; and to prove the integrity of TC_1.
- L_0: the licence Alice acquired as a result of executing Procedure 5.
- $S_{SK_{LB_0}}(L_0)$: LB_0's signature on L_0. Its purpose is two-fold: to prove that L_0 was indeed issued by LB_0; and to prove the integrity of L_0.
- *container_id*: the secure container that Alice acquired as a result of executing Procedure 3.

Upon receiving the request, PD_0 must perform the following verifications:

- execute $key\_auth(TC_1 : Key)$ to verify that Alice knows the secret key corresponding to $TC_1 : Key$.
- $TC_1 : Issuer \in AUX$: to verify that $TC_1 : Issuer$ is trusted.
- $TC_1 : Validity >= NOW()$: to verify that TC_1 is in its validity date.

- $TC_1 : Pseudonym \equiv TP_1$: to verify that TC_1 belongs to TP_1.
- $TC_1 : Level \equiv 2$: to verify that TP_1 is a transaction pseudonym.
- $TC_1 \notin CRL$: to verify that TC_1 has not been revoked.
- $sig\_ver(TC_1, S_{SK_{TC_1:Issuer}}(TC_1), PK_{TC_1:Issuer})$: to verify that TC_1's signature is correct.
- $L_0 : Holder \equiv TP_1$: to verify that L_0 belongs to TP_1.
- $L_0 : Issuer \in DRM$: to verify that $L_0 : Issuer$ is trusted.
- $L_0 : Validity >= NOW()$: to verify that L_0 is in its validity date.
- $sig\_ver(L_0, S_{SK_{L_0:Issuer}}(L_0), PK_{L_0:Issuer})$: to verify that L_0's signature is correct.

If these verifications are unsuccessful, PD_0 notifies The Authority and exits the procedure. If the verifications are successful, PD_0 is assured that Alice has supplied a valid pseudonym and must have thus revealed her identity to a trusted member of AUX. PD_0 decrypts $L_0 : Key$ using SK_{DRM} to acquire the content decryption key k by executing $D(L_0 : Key, SK_{DRM})$. PD_0 decrypts then $container\_id$ by executing $D(container\_id, k)$ and grants Alice access in accordance to $L_0 : Rights$.

Procedure 7: Identity traceback

Procedure 7 is used by The Authority to trace a consumer's civil identity from a transaction pseudonym. Under exceptional circumstances (*e.g.* after the detection of malice), The Authority may need to trace a consumer's civil identity to hold him accountable for his malicious activities.

Let's imagine that The Authority has detected a malicious transaction. After referring to the DRM entities, The Authority has learnt that the responsible consumer is Alice after executing the transactions in Figure 5.5. By identifying herself with a pseudonym she performed them anonymously. The Authority can however, revoke this anonymity to hold her accountable.

The Authority and the DRM entities can co-ordinate to reveal that Alice identified herself with transaction pseudonym TP_0 with corresponding certificate TC_0. Now we know that:

$$TC_0 : holder = E_{PK_{CA_2}}(E_{PK_{CA_1}}(E_{PK_{CA_0}}(id_{Alice})||CA_0)||CA_1). \tag{5.6}$$

By decrypting $TC_0 : Holder$, The Authority is able to reveal Alice's identity. This requires the co-operation of CA_2, CA_1 and CA_0, in that order. Each CA uses its secret key in turn so that a complete decryption, id_{Alice}, can be computed. Now The Authority is able to hold Alice accountable for her actions. On its own however, this is not sufficient.

From our discussions throughout this thesis, we know that it is vitally important that, after Alice has been identified as the malicious consumer, that she is prevented from continuing to use the service. We must therefore insert all her pseudonym certificates into CRL. Therefore, to achieve full consumer revocation, her entire pseudonym tree must be traversed with all pseudonym certificates revoked. Once Alice has been identified, her master pseudonym is consequentially revealed. This can be subsequently revoked, and its corresponding certificate inserted into CRL. By broadcasting the CRL's updates, all CAs issuing a pseudonym to a consumer who identified himself with a revoked pseudonym can subsequently place its corresponding pseudonym certificate in CRL. Repeating this process would result in all Alice's pseudonyms being revoked.

5.4 Evaluation

In this chapter we propose a preliminary, accountable anonymity service in DRM, A^{2}DRM. We believe that our solution provides the first step to providing DRM consumers with accountable anonymity. In this section we analyse our solution against our set of requirements for an accountable anonymity service in DRM (Section 2.4) and existing state-of-the-art DRM solutions (Section 2.3).

5.4.1 A^{2}DRM Evaluation Metrics

We have discussed thus far how important anonymity is in a DRM environment; particularly to provide consumer privacy. We believe that only when anonymity is incorporated into DRM will DRM begin to be more widely accepted by the consumers that use it. We have discussed a wide variety of anonymity strategies, theories and systems which provide varying ranges of anonymity and accountability. Some provide strong anonymity (*e.g.* blind signatures that provide no form of traceability or user accountability); whilst others are weaker, providing accountable or traceable anonymity (*e.g.* fair blind signatures that allow a TTP to trace the signer's identity). These two security properties, strong anonymity and weak anonymity, are qualitative. They can be deemed to be subjective; they could be a belief, or a judgement. One system's anonymity described as very strong, may not be as strong as another *only* described as strong. A more useful and meaningful anonymity depiction would have quantitative foundations, providing a metric to quantify the strength of anonymity. This is particularly important since one needs to be able to measure the success of any solution emanating from this research to provide accountable anonymity for DRM consumers.

Anonymity however, could be seen as a broad security property; encapsulating

many meanings. In our field, we can make the distinction between connection-anonymity and data-anonymity: connection-anonymity is concerned with hiding the source and destination of a message; whilst data-anonymity is concerned with hiding identifying information in the data. In the remainder of this section we shall discuss how anonymity can be quantitatively measured.

The Anonymity Set

According to Pfitzmann *et al.* [PH08], anonymity is the state of being unidentifiable within a set of subjects, the Anonymity Set (AS). AS is the set of all possible subjects that might cause an action (which we might call a session in a DRM system). That is to say, each session must exhibit an appropriately sized set of consumers from which it emanated from, or terminated at; and this set of consumers is the AS. Assuming that all subjects have an equal probability of being associated to a session, we can calculate the anonymity, a, of a system to be:

$$a = |AS| \tag{5.7}$$

In other words, as $|AS| \to 1$, the anonymity of the system becomes weaker. It follows that as $|AS| \to \infty$, the anonymity of the system becomes stronger.

Anonymity, or identity privacy, is very much context-dependent. Its strength and effectiveness lie heavily on the anonymity set. The larger the anonymity set is, the stronger the anonymity is; however, only within reason. We must ensure that the probability of any one consumer partaking in a session is evenly distributed amongst all consumers; the probability that any one consumer participated in any one session is the same for every consumer. Assuming possible attackers have no *a priori* knowledge, and they have not executed any profiling, then the more evenly distributed the probability of a member participating in a session is,

the stronger the anonymity experienced is.

Choice, Uncertainty and Entropy

In information theory, entropy is a measure of the uncertainty of a random variable [Sha01]. Given a set of possible events whose probabilities of occurrence are known, Shannon [Sha01] introduced a metric to measure how uncertain one can be of the outcome. For a *fair* flip of a coin, and excluding extremely unlikely events, the metric gives an entropy of one, since it can either land heads-up or tails-up; there cannot be any more uncertainty. It therefore follows that an *unfair* coin flip would have a lower entropy. If the probability of the coin landing heads-up is more likely, then we can be *less uncertain* about the outcome; it is more likely to land heads-up, so one can be *more certain* of the outcome. The anonymity set metric is not ideal. It can only consider events where all events are equally likely to occur; entropy however, can be used in situations where the probabilities are not equal.

Entropy, as a concept, is particularly interesting to our field of research. It allows one to consider events with *different* occurrence possibilities. Most systems, even those that provide strong anonymity, can be subjected to malicious observation. Given enough resources, an attacker (say Eve) could perform a user profiling mechanism to gain probabilistic information about a system's users. Although Eve is likely to be unable to identify a user for having received a particular piece of digital content, she may be able to attach probability attributes to each user. Some users may be more likely (*e.g.* those who are online) to have received a particular piece of content, whilst others (who are offline), may be less likely.

Let us imagine that, after carefully monitoring a system, Eve has been able to attach certain attributes to certain users. She is particularly interested in a

particular session between two parties. Eve is unaware of their corresponding identities, but she has been able to eliminate some users as recipients of a particular piece of digital content and in turn, placed a greater probability on others. Eve may subsequently be able to assign different probabilities to different users from which a session emanated from; by means of traffic analysis, timing attacks, message length attacks or more sophisticated attacks.

Furthermore, we must not ignore the possibility that observing communication between a user and a particular content distributor could allow Eve to build dossiers on a particular user. If the content distributors were partitioned by genre, for example, then Eve could then infer the genre the user is interested in, violating the user's privacy.

In 2003, Diaz *et al.* [DaJCP03] and Stertojv *et al.* [SD03] independently adopted Shannon's entropy metric and applied it to the concept of anonymity. Here, it is assumed that Eve is able to observe and monitor the sessions between Alice and the DRM entities. The sessions could be encrypted to provide a form of data-anonymity; and the users may be using pseudonyms to provide a form of connection-anonymity.

For our discussion, let's assume that the sessions are encrypted with a strong encryption algorithm; making decryption without the required key infeasible. The user's identity may be hidden with a pseudonym but the connection still exists. After observing and monitoring the system, Eve may be able to assign different probabilities to different users from which a session emanated from; by means of traffic analysis, timing attacks, message length attacks or more sophisticated attacks.

Diaz *et al.* [DaJCP03] and Stertojv *et al.* [SD03] adopted Shannon's entropy metric to provide an anonymity-entropy metric to calculate the entropy of the system after an attack has taken place. In order to apply Shannon's entropy idea,

let's make X be the discrete random variable we wish to learn our uncertainty on, which could be a DRM consumer's association to a session. Its probability mass function is:

$$p_i = Pr(X = i). \tag{5.8}$$

In our field of interest, i corresponds to a consumer in the set of all consumers, N. So Eve, whose been observing the system, assigns a probability, p_i, to each member of the anonymity set. The entropy of X can be calculated thus:

$$K(X) = -\sum_{i=1}^{|N|} p_i \log_2 (p_i). \tag{5.9}$$

Stertojv *et al.* [SD03] propose using the value $K(X)$ to define the uncertainty regarding which consumer of the anonymity set sent the session or transaction. As $K(X) \to 0$, Eve becomes more knowledgeable; and it follows that as $K(X) \to \infty$, Eve becomes less knowledgeable. Obviously, if the anonymity set is of size one, then the uncertainty would be zero; Eve would be certain which consumer is associated to the session or transaction.

Diaz *et al.* [DaJCP03] take a different slant; they define a *degree of anonymity*. Here, $K(X)$ is divided by the maximum entropy, K_M — *i.e.* when the probabilities are evenly distributed. The information the attacker has learnt can thus be expressed as $K_M - K(X)$, which is normalised by dividing it by K_M. The degree of anonymity, d, is

$$d = 1 - \frac{K_M - K(X)}{K_M} = \frac{K(X)}{K_M}, \tag{5.10}$$

which gives a normalised entropy, called the *degree of anonymity*. It follows immediately that $0 \leq d \leq 1$ and:

- $d = 0$ when a user appears as being the originator of a message with a probability $p_i = 1$;
- $d = 1$ when all users appear as being the originator with probability $p_i = \frac{1}{N}$; *i.e.* an equiprobable distribution.

The degree of anonymity defined here quantifies the amount of information the system is exposing. If a particular user, or a particular group of users are known to have a higher session-association probability than others, then the system is providing a low degree of anonymity.

One can see correlations between the degree of anonymity defined here, to the anonymity defined by Piftzmann *et al.* [PH08]. The maximum degree of anonymity is achieved when all subjects in a system (*i.e.* anonymity set) are equally likely to have participated in the session. If this were to be the case, then the larger the number of subjects, the greater their individual anonymity. By amalgamating Piftzmann's anonymity and an adaptation of Shannon's concept of entropy, one is able to consider cases which do not provide equiprobable distribution: a system in which an attacker is able to reduce the probability for some users and increase it for others.

Both Stertojv *et al.* and Diaz *et al.* provide a general measurement model to quantify the anonymity of a system. However, using precise probabilistic values in such a metric may prove to be difficult. An attacker with unlimited time and computational power may find it much easier to acquire better probabilistic values (*i.e.* values that tend to the extremes, zero or one). It is also difficult to judge and subsequently quantify the *a priori* knowledge an attacker may have. The degree of anonymity only takes into consideration an attacker's *a priori* and *a posteri* knowledge.

Scaled Anonymity Set

With the knowledge of Diaz *et al.*'s and Stertojv *et al.*'s metrics, Andersson and Lundin [AL08] proposed a set of criteria that they feel anonymity metrics should satisfy. According to their criteria, entropy-based metrics have more strengths than anonymity set-based metrics; although they noted that they both hold some weaknesses.

By using the criteria set by Andersson and Lundin, one may deduce that a metric based on the anonymity set is basic; it doesn't provide an extensive insight to the anonymity a system provides. Andersson and Lundin argue that a well defined anonymity metric should base its analysis on probabilities of occurrence; those based on the anonymity set metrics, do not. They simply reflect the total number of subjects in the anonymity set. It is impractical to think that all members of the anonymity set are equally probable of participating in a session or transaction.

On the other hand, entropy-based metrics provide a more extensive anonymity value. They consider the varying probabilities that DRM consumers are likely to hold. Diaz *et al.* in particular, provide a *normalised* metric — a key criterion, claim Andersson and Lundin. In saying this however, Diaz *et al.* don't provide a metric that reflects the size of the anonymity set. Andersson and Lundin believe that a system with a larger anonymity set should experience a stronger degree of anonymity than a system with a smaller anonymity set, which is not necessarily the case with Diaz *et al.*'s metric. By not normalising their anonymity-metric, Steranjov *et al.* provide a metric that does provide such a correlation. As the size of the anonymity set increases, so does the strength of the anonymity. According to Andersson and Lundin, an anonymity metric should be normalised *and* correspond to the size of the anonymity set; neither Diaz *et al.* nor Stertojv

et al. provide both.

So, in 2008, Andersson and Lundin proposed a normalised anonymity-metric that corresponds to the size of the anonymity set. Here, one is able to calculate [AL08]:

> the number of possible outcomes, given the expected amount of binary questions the attacker needs to answer to identify the sender

by using:

$$a = 2^{K(X)}. \tag{5.11}$$

Here, a grows with an increasing uniformity of $K(X)$, a key criterion defined by Andersson and Lundin; the metric considers the occurrence probabilities of each event. Moreover, a varies between 1 and the size of the anonymity set; and thus a larger anonymity set will generate a higher number of possible outcomes. If, for example, $K(X) = 2$, then $a = 4$. Semantically, this can be explained by the number of possible binary combinations when $K(X) = 2$ — *i.e.* $\{0, 0\}$, $\{0, 1\}$, $\{1, 0\}$, and $\{1, 1\}$. The maximum and minimum values that a may take, correlate with the size of the anonymity set: a is at its greatest when $K(X)$ is at its greatest, which is when $|AS|$ is at its greatest. It follows that a is at its lowest when $|AS|$ is 1, a singleton anonymity set, *i.e.* no anonymity. It is for this reason it is called the *scaled anonymity set size*. An anonymity set of size three, is twice as 'anonymous' as an anonymity set of size two.

Entropy-based metrics can only measure a system's connection-anonymity. It is only focussed on sender-anonymity; although recipient-anonymity can be treated analogously. Data-anonymity is not considered at all. Entropy-based metrics provide a general overview of a system's anonymity. However, a high degree of anonymity (as defined by Diaz *et al.*), a high uncertainty value (as defined by Stertojv et al), or a high number of possible outcomes (as defined

by Andersson and Lundin), does not necessarily mean that a system is secure; although low values do indicate a system is not particularly secure. It may be that with the addition of dummy traffic, a system could achieve a greater anonymity.

We are however researching in the field of accountable anonymity where a user's anonymity can be revoked in exceptional circumstances. The majority of the systems we have discussed heavily rely on the trust of a TTP, or a set of TTPs, to revoke a user's anonymity. Using such metrics under such circumstances is particularly difficult to quantify and may not be taken into consideration when calculating the degree of anonymity. For example, it may be difficult to quantify the likelihood of a DRM entity or TTP from becoming compromised.

5.4.2 A^{2}DRM Evaluation

We now evaluate A^{2}DRM against its ability to provide anonymity, accountability, and unlinkability. We use the metrics discussed in Section 5.4.1; whilst analysing A^{2}DRM against our set of design requirements (Section 5.2.1) and state-of-the-art systems (Section 2.3).

Anonymity

Providing anonymity for consumers in a DRM environment is what this thesis is primarily centred on. From our discussions throughout this thesis, we know that providing consumers with anonymity without accountability can be potentially dangerous. With this in mind, it would be unsafe to provide anonymity without any revocation. In our solution, under the mask of pseudonymity, Alice protects her civil identity by using a pseudonym during identification procedures. By hiding the link between her civil identity and her pseudonym, identification can be achieved whilst preserving her privacy. This is, however, insufficient.

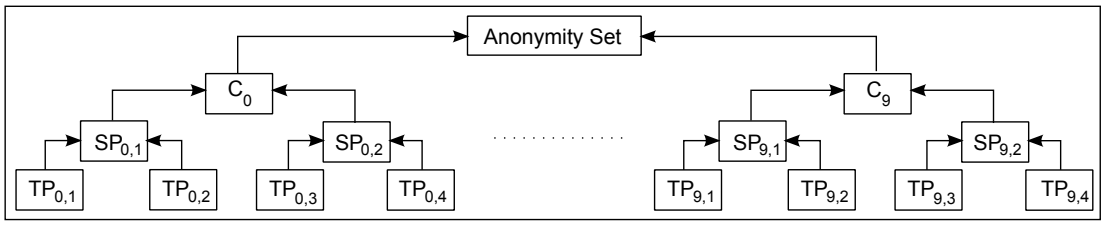

Figure 5.6: Our anonymity set for our test-environment.

From our discussions in Section 2.2 we know that hiding a consumer's identity is only one aspect of a full anonymity service. By identifying herself with a unique pseudonym, she reveals a single identifier that can be linked to her transaction. Continual use of the same pseudonym for multiple transactions could allow an attacker to link her transactions together. So to prevent this, and to preserve her privacy, Alice uses a different, unlinkable pseudonym for every transaction she participates in; we call these, transaction pseudonyms. To acquire a transaction pseudonym, she must have already acquired a master pseudonym and a session pseudonym.

For Alice's privacy to be preserved, the link from her civil identity to her transaction pseudonym remains conditionally hidden. To analyse the anonymity we provide in our solution, we first look back at the anonymity metrics we discussed in Section 5.4.1. Let's first build our test-environment.

Let's assume that for the following discussions, our system is active with ten consumers: $\{C_0, C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9\}$; for a matter of simplicity, we often use Alice as a synonym for one of these consumers. Let's also assume that each consumer has acquired a master pseudonym and two session pseudonyms. Each consumer has subsequently used their session pseudonyms to acquire two constituent transaction pseudonyms. We illustrate this in Figure 5.6, and depict Alice's pseudonym tree in Figure 5.7.

Let's consider some example cases under our test-environment. Imagine that

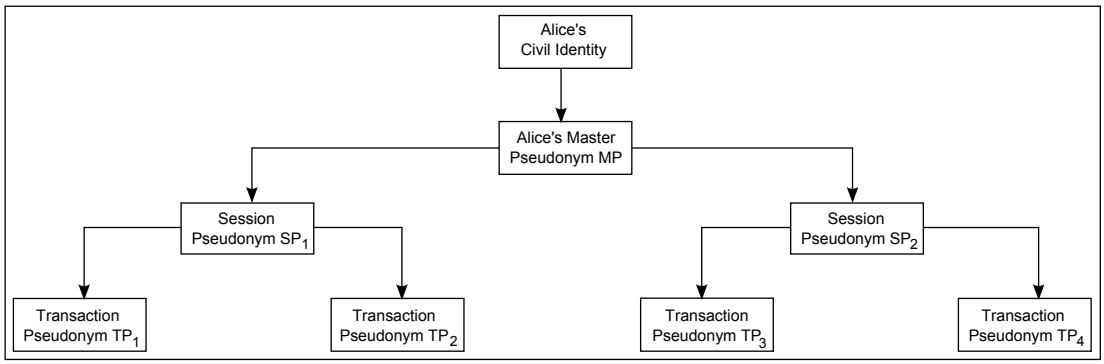

Figure 5.7: Alice's pseudonyms for our example cases.

Eve attempts to attack the DRM system to learn as much private information as she can. We imagine five different cases here to illustrate the security of our system:

Case 1: Eve has not carried out any sophisticated attacks. She only knows the size of the anonymity set. She wants to be able to link a specific transaction pseudonym to a specific consumer.

Case 2: Eve carried out sophisticated attacks to eliminate some consumers as possible participants of a transaction. She has learnt that some consumers were offline (P_1, $P_3 \rightarrow P_6$, P_8, P_9) whilst the transaction was active.

Case 3: Through their respective Internet activities, Eve has been able to attribute each consumer with a probability of participating in a transaction.

Case 4: Eve has attacked a CA and learnt that only three consumers have accessed it to acquire a transaction pseudonym. By monitoring all certificates issued by the CA, Eve knows which pseudonyms it issued. Eve attempts to link these to their respective owners.

Case 5: Eve attacks Alice in an attempt to derive a complete dossier of her activities. By monitoring Alice's communication channels, Eve has been able to extract Alice's identifier.

Table 5.2: The respective probabilities $P_0 \rightarrow P_9$ for each consumer $C_0 \rightarrow C_9$ for each example case $1 \rightarrow 4$.

Probability/Case	1	2	3	4
P_0	$\frac{1}{10}$	$\frac{1}{3}$	0.20	0
P_1	$\frac{1}{10}$	$\frac{1}{3}$	0.50	0
P_2	$\frac{1}{10}$	0	0	$\frac{1}{3}$
P_3	$\frac{1}{10}$	0	0	0
P_4	$\frac{1}{10}$	0	0	$\frac{1}{3}$
P_5	$\frac{1}{10}$	0	0	0
P_6	$\frac{1}{10}$	$\frac{1}{3}$	0.30	0
P_7	$\frac{1}{10}$	0	0	0
P_8	$\frac{1}{10}$	0	0	$\frac{1}{3}$
P_9	$\frac{1}{10}$	0	0	0

The respective probability mass functions for the first four example cases are depicted in Table 5.2, where P_i denotes the probability of consumer C_i participating in the said transaction.

Let's consider Case 1. Here, Eve has not acquired any privacy-compromising information. She does however, know that the anonymity set consists of ten consumers. She has not been able to attribute any probabilistic information to any consumer as a participant of a transaction. She is able to calculate the probability of any one consumer participating in any one specific transaction thus:

$$\begin{aligned} a =& P_i \\ =& \frac{1}{10}. \end{aligned} \tag{5.12}$$

Now this is just a direct reflection of the number of consumers in our test-environment. The only way to improve the anonymity of our system, using this metric, would be to increase the number of consumers. Although this is on par with the DRM systems we discussed in Section 2.3, we know (Section 5.4.1) that the anonymity set metric is a rather weak metric. More sophisticated attacks may allow Eve to acquire more valuable identifying information. Attacking a CA for

example, may allow Eve to attribute certain probabilities to certain individuals as having participated in transaction. So let's now consider Case 2.

In Case 2, Eve carries out more sophisticated attacks to eliminate some consumers as probable participants in a transaction. By monitoring their respective Internet activities, Eve has established that, whilst the transaction was active, some consumers were offline and can then be disregared as possible participants. We can thus use the *entropy metric* proposed by Stertojv *et al.* [SD03] (discussed in Section 5.4.1) to calculate the anonymity of the system:

$$\begin{aligned} K(X) =& -\sum_{i=1}^{|N|} p_i \log_2 (p_i) \\ =& -\frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) \\ =& -\log_2\left(\frac{1}{3}\right) \\ =& 1.58\ (3\ s.f.) \end{aligned} \tag{5.13}$$

From our discussion in Section 5.4.1 we know that this does not necessarily provide a fair reflection of the system's anonymity (Andersson *et al.* [AL08]): it does not represent the size of the anonymity set. A fairer reflection could be calculated using the *scaled anonymity set* metric [AL08]:

$$\begin{aligned} a =& 2^{K(X)} \\ =& 2^{1.58} \\ =& 2.99\ (3\ s.f.) \end{aligned} \tag{5.14}$$

Using even more sophisticated attacks in Case 3, Eve has acquired an uneven probability distribution[5]. By monitoring a DRM entity's connections, whilst a

[5]We have chosen probabilities for this case (relatively arbitrarily) to demonstrate the affect

transaction is active, Eve has been able to attribute different probabilities to different members of the anonymity set. We can find the system's entropy:

$$\begin{aligned} K(X) =& -0.20\log_2(0.20) - 0.50\log_2(0.50) - 0.30\log_2(0.30) \\ =& 1.49\ (3\ s.f.) \end{aligned} \tag{5.15}$$

From which we can calculate anonymity a using the scaled anonymity set:

$$\begin{aligned} a =& 2^{1.49} \\ =& 2.81\ (3\ s.f.) \end{aligned} \tag{5.16}$$

From our discussions in Section 5.4.1 we know that a higher value of a denotes a higher anonymity. So, the best case we have discussed is Case 1, where the probability is uniform across all consumers. As one would imagine, Case 3 provides a greater threat than Case 2 — as 2.81 is less than 2.99. The more precise Eve can be with her probability assignment and the more uneven the distribution is, then the more dangerous her attack can be. In Case 3 Eve has been able to attach a greater probability to P_1 as the participant of the transaction, and as a consequence, attached a lower probability to both P_0 and P_6. From Eve's perspective, Case 3 is therefore a more successful attack.

At this stage we must highlight that, under our test-environment, these first three cases would result with the same anonymity quantification for the DRM systems we discussed in Section 2.3. The cases are not specific to ours; so let's consider one that is.

In Case 4 Eve has attacked a CA (*e.g.* with message timing attacks) and learnt that only three consumers have accessed it to acquire a transaction pseudonym.

of an uneven probability distribution.

By monitoring all certificates issued by the CA, Eve can find which pseudonyms it has issued. From our calculations in Case 2 we already know that:

$$K(X) = 1.58, \tag{5.17}$$

and

$$a = 2.99. \tag{5.18}$$

However, we also know that these three transaction pseudonyms will only be used once. Eve could carry out more sophisticated attacks to improve her probabilities (of $\{P_2 = \frac{1}{3}, P_4 = \frac{1}{3}, P_8 = \frac{1}{3}\}$) but seeing as a transaction pseudonym is only used during one transaction, this may be hard. It therefore may not be worthwhile for Eve to use her resources to continue to trace the pseudonym's owner.

Let's compare this to the system proposed by Conrado *et al.* [CKSJ03]. Here, consumers are issued with different, unlinkable pseudonyms, similarly to our solution. Pseudonyms here however, are issued by a CA after Alice has revealed the identity of her smart-card. The CA is thus able to link all Alice's pseudonym's together. Now, although the playback device is unaware of Alice's association to the pseudonym, it could collude with the CA to monitor her playbacks. It would then be able to infer trends and build a dossier.

Now in our proposed solution, even if the playback device and the CA collude, this is hard to achieve. They could reveal Alice's session pseudonym, but this is different for every session she participates in; so her sessions cannot be linked together as easily. As we have already discussed, this is not particularly problematic; her identity cannot be revealed, nor can her other session or transactions be monitored. Therefore, despite providing a form of accountability (unlike that

of Conrado *et al.*), we are still able to provide a stronger level of anonymity. We must highlight here, that it is likely that the playback device would be distributed by the system. If the playback device was controlled by the system, then this scenario cannot be ignored.

Let's move on to Case 5. Here, Eve chooses to attack a specific consumer, Alice. She's attempting to derive a complete dossier of her activities. She is particularly interested in revealing Alice's interests. By monitoring Alice's communication channels, she has been able to extract Alice's unique identifier. In our system, her identifier correlates to her transaction pseudonym; in the system proposed by Feng *et al.* [FZ08], it correlates to her public key.

Let's assume that the system comprises a set of content distributors. This can be analogised to a shopping high street where each content distributor may distribute different contents at different prices. One might imagine different distributors promoting different promotional incentives in an attempt to lure the consumers' interest in purchasing contents. So let's say that one retailer, CD_0, distributes a set of content; we call this set D.

Now in the system proposed by Feng *et al.*, contents are also publicly segregated into sets dependant on their monetary values; let's call this set M. To acquire a licence from a licence broker, Alice must reveal the monetary set in which her content resides. We define these two sets thusly:

$$\begin{aligned} D &= \{S_0, S_2, S_3, S_4, S_7, S_9\} \\ M &= \{S_0, S_1, S_4, S_7, S_8, S_9\} \end{aligned} \tag{5.19}$$

where S_i denotes the secure container of content i.

Imagine that Alice now uses the system proposed by Feng *et al.* to download a secure container from CD_0, and then attempts to acquire a licence for it from LB_0.

By monitoring Alice's communication channels, Eve has learnt that Alice has downloaded a secure container from CD_0 — although she is unable to determine what the container is securing. She has also learnt the monetary group in which the content resides. By cross-referencing the set of contents D against M, Eve is able to make probabilistic inferences of the content Alice is requesting a licence for. In other words, Eve could calculate:

$$M \cap D, \tag{5.20}$$

which will reveal the set of contents that CA_0 distributes costing the amount paid by Alice. This is likely to result in a set smaller than either M or D. Eve can therefore eliminate some contents from the set of possible contents Alice accesses. So much in fact, that if we were to now apply the anonymity set metric here, it gives:

$$a = 4, \tag{5.21}$$

since there are only four contents that exist in both sets: $\{S_0, S_4, S_7, S_9\}$. Continual observations and more sophisticated attacks (*e.g* of the associated genres) could lead Eve to acquire more accurate inferences of Alice's interests, from which subsequent dossiers can be constructed.

In our system on the other hand, we only partition our contents once — as distributors distribute different contents. Therefore this attack cannot occur. Applying the anonymity set metric here gives:

$$a = 6, \tag{5.22}$$

i.e $|D|$, since Eve cannot eliminate any contents from the set of possible contents

Alice accesses using this attack.

Furthermore, our identifiers are only used once. The transaction pseudonym revealed by Alice is only used once; for one licence acquisition. In the system proposed by Feng *et al.*, where Alice uses the same unique identifier continually, Eve can acquire subsequent inferences for every transaction. Considering that in our solution transaction pseudonyms owned by a single consumer provide no bearing to each other, linking them together would be rather difficult without further attacks.

Bearing all our discussions in mind, it is important to emphasise that all our calculations consider connection anonymity. A consumer's connection may give away identifying information; we have already seen that the Internet can reveal some identifying information. Although it is imperative to provide strong connection anonymity, it could be futile if the same fastidious attitude is not applied to data anonymity. We must not disclose identifying information within the transactions themselves. It is imperative that, for example, a consumer's civil identity is encrypted before it's inserted into a master pseudonym's certificate.

We must also consider that a consumer's privacy is preserved by his pseudonym tree. Revealing a consumer's pseudonym tree would severely compromise his privacy. It is important to remember that the pseudonym tree is effectively maintained by a set of CAs. The CAs could collude to remove the anonymity it provides. We assume that the CAs do indeed act honestly but this possible threat must be considered.

On its own, a CA has the ability to reveal a particular level of a consumer's pseudonym tree, but not an entire branch from the civil identity to the transaction pseudonym. This is important; it is the primary purpose of the pseudonym tree: to make an illegitimate identity trace infeasible; or at best, difficult. Our

calculations above do not take these considerations into account. They still provide a reflection of our solution's anonymity but indeed only that, a reflection, not a true indication of its anonymity.

To demonstrate the relative threats of our example cases, we utilised the anonymity metrics we discussed in Section 5.4.1. The worst case scenarios occur when a more uneven probability distribution can be established by the attacker. In discussing each case, we compared it to the already existing DRM systems to demonstrate the strength of our anonymity.

Accountability

In addition to anonymity, accountability is an important security property, it is fundamental to our research. We know that providing anonymity without accountability can be dangerous. Providing accountability is therefore pivotal. We have seen examples of how anonymity without accountability can be exploited. Despite these examples, we have seen DRM systems that provide anonymity without accountability (in Section 2.3). Our solution provides both: anonymity *with* accountability.

Unlike anonymity however, it's difficult to quantitatively measure accountability: one's either accountable; or ones's not. There's not a range of accountability like there is for anonymity. To provide accountability, we maintain a pseudonym tree for each consumer. To hold Alice accountable, we can use our identity trace-back procedure to trace her civil identity from her transaction pseudonym. This effectively traverses her pseudonym tree. We know that all unsigned transactions will be rejected. This means that for Alice to authenticate her transaction with a signature, she must reveal her certificate. By certifying the signature with the corresponding certificate, one consequentially reveals the transaction pseudonym.

In our proposal, to acquire a transaction pseudonym, Alice must have acquired a master pseudonym. To acquire a master pseudonym, Alice must first reveal her civil identity to a TTP, which we call a CA. In doing so, Alice must prove that this identity does indeed apply to her. This could be achieved in a similar way to opening a bank account. Usually, to open an account, a customer must prove his identity by, for example, supplying official documents such as a utility bill or a passport. Further to this, we also ask Alice to supply the public key officially and publicly bound to her. In the digital domain, this is often used as her unique identifier. With both her physical identity (*i.e* her name, address, *etc.*) and her digital identity (*i.e.* her public key certificate) known, Alice can be held accountable for any malicious actions she may take.

Now, if the correctly identified consumer can demonstrate the knowledge of the corresponding secret key, then the CA issues the consumer with a master pseudonym and its corresponding signed certificate. Alice can then use her master pseudonym to first acquire a session pseudonym, and then a transaction pseudonym. To trace Alice's civil identity from her transaction, a vote must first be established.

Before Alice can have her identity revealed, the members of DRM must vote to decide if her identity should be revealed. The members of DRM are not responsible for executing the identity trace, they merely request it. It is the role of the CAs, members of AUX, to perform the trace. This therefore *separates the concerns* of the entities within the DRM processes, and the entities involved with providing, maintaining and revoking consumers' anonymity. We consider this to be a strength. Separating the concerns is particularly important when dealing with revoking anonymity to provide accountability. Using this approach, we minimise the affect of a bias judgement a single entity may have. Unfair requests for Alice's civil identity to be revealed should be filtered out by fair

votes.

Providing a strong separation of concerns such as this also prevents entities from gaining information outside their operational scope. By allowing DRM entities to manage DRM processes *and* consumers' anonymity, consumers' privacy could be easily jeopardised by dishonest DRM entities. Some DRM entities could have a different agenda to those auxiliary entities. They could trace a consumer's identity and then monitor their activities. Under our setting, where we separate the DRM processes from anonymity revocation, DRM entities are unable to achieve this without the help of the corresponding CAs.

If DRM were to vote to revoke a consumer's anonymity then the members of AUX could co-ordinate to revoke it. We could even add the restriction that AUX must vote *and agree* with DRM before an identity trace is executed — to prevent DRM from making biassed judgements. Revoking the anonymity requires the pseudonym tree to be traversed; resulting with the holder's civil identity from being revealed. Together, the set of CAs issuing the pseudonyms can revoke the consumer's anonymity. This could be called full anonymity revocation: the consumer's identity has been revealed from a previously anonymous state. We know however, that this alone is insufficient; all a consumer's pseudonyms must be revoked too. This brings us on to another of our requirements for an accountable anonymity service in DRM: conditional unlinkability. We evaluate this requirement later in this section.

The process of anonymity revocation is heavily dependent on a consumer's pseudonym tree. A consumer's identity can be revealed by tracing a branch from a transaction pseudonym to the holder's civil identity. From our discussions in Section 5.3 we know that the $Holder$ field within a pseudonym certificate contains

a link to the holder's civil identity. Let's consider an example $Holder$ field:

$$TC : Holder = E_{PK_{CA_2}}(E_{PK_{CA_1}}(E_{PK_{CA_0}}(id)||CA_0)||CA_1). \tag{5.23}$$

Here, CA_2 issued the transaction pseudonym; CA_1 the session pseudonym; and CA_0 the master pseudonym. So, The Authority could ask CA_2, CA_1 and then CA_0, to decrypt $TC : Holder$ to reveal id.

By asking a CA to concatenate additional information to $Holder$ before encrypting it, $Holder$, and thus the certificate, grows from an already large size. This may be of importance for smaller systems.

Alternatively, instead of inserting an encryption of the identity into $Holder$, the CA could insert a hash of the civil identity. By inserting a hash, the CA still hides the civil identity from any onlookers. Under revocation procedures, the CA must then be responsible for revealing the consumer's civil identity. This can easily be verified by hashing the revealed identity and comparing it against the hash value stated in the signed certificate. We could further secure this by using random salt values and/or keyed hash functions. We know that a hash value is typically much smaller than the original plaintext value, so it is more efficient. However, for revocation procedures, the CA would have to store the tuple:

$$\{P_i, P_j, C_i : Holder\}, \tag{5.24}$$

where P_i is the pseudonym the consumer uses to identify himself with; P_j is the pseudonym the CA issues; and $C_i : Holder$ is the $Holder$ stated in P_i's corresponding certificate. The CA can then store the relationship the consumer has with the certificate without having to insert the large value of the civil identity

into the certificate. This would however, have the adverse affect of large CA-storage requirements.

By using certificates in this manner, we are able to provide consumers with *accountable* anonymity. Accountability is an important security property, it allows malicious consumers to be held accountable for their activities and to be reprimanded accordingly. Let's consider some example cases:

Case 6: Eve attacks the licence broker. She makes a payment to the licence broker for a certain amount and then attempts to use the acquired receipt of payment to acquire a more expensive licence.

Case 7: Eve attacks Alice and acquires her payment details. She then proceeds to purchase licences using Alice's payment details.

Let's consider Case 6 Eve attempts to use a receipt to prove payment of a more expensive licence. To preserve her privacy, the licence she wants to acquire is hidden from the payment gateway. The price she paid however, is not.

After a successful licence payment, the payment gateway issued Eve with a receipt of payment. The receipt only reveals the amount that she paid. Eve can therefore only use it to acquire licences of that value. So, any request of acquiring a licence exceeding the price she indeed paid, can be rejected. After detecting this malicious activity, the licence broker can, from Eve's transaction pseudonym, trace her identity by tracing her pseudonym tree. In the system proposed by Feng et al., they do not provide details of their proof of payment and thus, it must be assumed, cannot prevent such an attack.

In Case 7 Eve attacks the payment gateway and learns Alice's payment details. She then uses these details to purchase licences. After Alice reports that her details have been used maliciously, The Authority is able to co-ordinate with the payment gateway to trace her identity. By checking its records, the payment

gateway is able to link Alice's payment details to the transaction pseudonyms that were used during payments. Before tracing these however, the payment gateway must ensure that the pseudonyms belonging to Alice are not traced; only those belonging to, and used maliciously by, Eve.

To ensure that Alice's pseudonym tree is not inappropriately traced, Alice performs a two-party communication protocol with the payment gateway, to reveal the transaction pseudonyms she used during payment. In its present state, this cannot be achieved anonymously; it provides the payment gateway with some of Alice's transaction pseudonyms with which it could link Alice's payments together.

However, Alice's civil identity is typically intrinsically linked to her payment details. For her payment to be processed, she must reveal this to the payment gateway. The payment gateway is therefore not presented with any additional information. This does however mean that, even under legitimate use, the payment gateway can link all payments together. This is not ideal. To prevent this, Alice is encouraged to use many, different payment gateways. We could incorporate the electronic coins (as proposed by Chaum [Cha83]) into A^{2}DRM, but we would then inherit different, perhaps greater, weaknesses (as discussed in Chapter 4).

Conditional Unlinkability

We know from our discussions in Sections 2.2 and 2.4 that unlinkability is an important security property; provided that it is conditional. In exceptional circumstances where Alice acts with malice, for example, it's important to revoke her unlinkability. From our evaluation of accountability, we know that Alice's identity can be traced using her pseudonym tree. This alone however, cannot be considered as full accountability.

To provide full accountability, we must reveal and subsequently revoke all pseudonyms belonging to Alice. After being found acting with malicious intent and having her identity revealed, all her pseudonyms should be revealed. To reveal all pseudonyms, the consumer's pseudonym tree must be traversed in its entirety, from top to bottom. At each level, all pseudonyms should be placed on a blacklist of unacceptable pseudonyms. This can be achieved using our procedure for tracing a consumer's civil identity. None of the DRM systems we discussed in Section 2.3 can achieve this, which we consider to be a major weakness.

In our system, once a civil identity has been revealed and the master pseudonym revoked, the TTP responsible for maintaining the revocation list broadcasts its update. Any CA that issued a pseudonym to a consumer identifying himself with any pseudonym on the list, subsequently places this pseudonym on the list too. This process is iterated until all transaction pseudonym have been revoked. We know that a transaction pseudonym cannot be used to acquire more pseudonyms since its corresponding certificate has a *Level* of 2; pseudonym acquisition requests with transaction pseudonyms are rejected. We provide full accountability through the pseudonym tree.

Let's consider some example cases in which Eve attempts to remove the unlinkability we provide so she can link Alice's pseudonyms together:

Case 8: Eve attacks Alice in an attempt to learn all her transaction pseudonyms.

Case 9: Eve attacks Alice in an attempt to link her sessions together so she can infer her trends and construct a dossier specific to her.

Let's consider Case 8, say one particular branch of Alice's pseudonym tree: the link between her civil identity and one of her transaction pseudonyms (as illustrated in Figure 5.8). By tracing this branch, Eve could link Alice's civil identity to one of her transaction pseudonyms. This branch, just like her whole

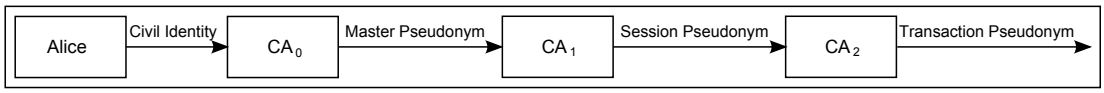

Figure 5.8: The link between Alice's civil identity and her transaction pseudonym.

pseudonym tree, is maintained by a set of CAs that issue Alice with a pseudonym certificate. Now, her civil identity is encrypted and inserted into each certificate. Only the holders of the corresponding secret keys have the ability to link Alice's pseudonyms together. In this case, there are three links: the session-transaction link; the master-session link; and the identity-master link. For Eve to successfully execute her attack, she must reveal each of these links.

Since each link is maintained by a different CA, Eve must make three separate attacks (one for each link) on three separate CAs. Without succeeding in each attack, Eve would be unable to reveal the whole branch. This would require her to attack each CA maintaining the link. Assuming these links are securely stored by each CA, it may take a very sophisticated attack to succeed. Attacking three CAs successfully is much harder than attacking one, as in the system proposed by Feng *et al.*.

In the system of Feng *et al.*, all consumers acquire all their pseudonyms from a single TTP. If this TTP becomes compromised, then the attacker may be able to link all pseudonyms to their corresponding holders. This is in contrast to the mechanics of our system. In our system, if one TTP (*i.e.* CA) is compromised, at most, it only affect one level of a consumer's pseudonym tree and the consumer's privacy is not fully compromised.

Now, the aim of Eve's attack here is to link all Alice's pseudonyms together. To achieve this, she must repeat the above process for each transaction pseudonym Alice holds. This becomes easier the more times she succeeds. The first time she performs the attack, she reveals Alice's master pseudonym; she need not learn

this link again, this pseudonym remains the same. Moreover, after revealing one session pseudonym she need not reveal this again to find other transaction pseudonyms in that session. Despite this consequence, our system is still more secure than that of Feng *et al.* where each attack only concerns one TTP.

Now let's consider Case 9. Here, Eve is attempting to link Alice's sessions together. From this she hopes to construct a dossier of Alice's activities so she can infer her shopping habits. In our discussion for Case 8 we saw how Eve could trace Alice's transaction pseudonyms via the pseudonym tree. To do so, she had to trace Alice's session pseudonyms. Applying a similar attack to Case 8 here would therefore allow her to acquire all Alice's session pseduonyms. In our system however, sessions do not provide any information regarding the contents and licences she accesses; only the use of transaction pseudonyms do. Therefore, in our system, the aim of Eve in Case 8 and Case 9 may differ, but the format of the attacks are the same. It is still difficult for Alice to achieve this and would require sophisticated attacks.

Subsidiary Issues

In order to provide a viable accountable anonymity service in DRM, we should endeavour to satisfy the needs of both content consumers and content owners. Whilst some of these needs are supported by satisfying our requirements, we believe that there are additional properties that should also be satisfied. From our discussions in Section 2.2, we believe that the majority of DRM systems attempt to satisfy the content owners' needs; whilst neglecting the content consumers' by compromising their privacy. We believe that only when *both* the content consumers' and content owners' are satisfied will a DRM system ever come to

fruition. The DRM systems we discussed in Section 2.3 have attempted to preserve consumers' privacy, in an attempt to satisfy both consumers and owners, but we feel that these systems do not wholly satisfy their concerns.

Now, both content consumers and content owners want to use a quick and efficient system. In providing a DRM system that satisfies our requirements we have introduced a number of TTPs (or CAs). The more TTPs that are involved, the more likely the system remains secure. In our setting, it only takes one TTP to remain honest for a consumer's privacy to be preserved. However, the process of constructing the pseudonym may cause an unwanted bureaucratic and cumbersome measure. To alleviate the problem, we could eliminate the session pseudonym level to provide a two-tier anonymity system. This would reduce the number of TTPs the consumer would have to communicate with before content can be acquired. However, this would be at the expense of their privacy: collusion is easier with fewer TTPs. An appropriate level must be established, specific for the system it is applied to.

Moreover, in reality, it may be that many TTPs would be influenced by a small subset of entities — providing a weaker system. In principal this idea is more secure; its realism in reality however, could reduce its effectiveness. Nevertheless, it is still more secure than using a single TTP as we have seen with Feng *et al.* [FZ08], for example.

We also believe that DRM should provide the ubiquity and flexibility that consumers require. We have criticised DRM systems (in Section 2.3) for their inflexibility and inability to support an ubiquitous environment. DRM systems are typically used for distributing media content (*i.e.* audio and visual content). It is easy to relate to content distribution systems to music and movies. Here however, we do not restrict ourselves to any particular type of content. Moreover, we allow a consumer to use his licence on any machine. As long as he is able

to identify himself with an authentic, certified pseudonym, then he is able to access the content on any playback device. The only restriction here is that the playback device must be *compliant*, using our software, which we described (in Section 5.3) as possibly being available over an FTP or HTTP connection. Unlike the smart-card based solutions we discussed earlier, we provide a more flexible and ubiquitous environment.

Furthermore, we also consider the particular importance of *separating the concerns*. Let's consider an example case:

Case 10: The CAs collude in an attempt to link every transaction to its holder.

In the solution proposed by Feng *et al*, two TTPs must collude: the licence broker and the playback device. In our solution, this would require the collusion of four TTPs: the CAs issuing the transaction, session and master pseudonyms; and a DRM entity (*e.g.* the licence broker). We therefore believe that we provide a stronger *separation of concerns*.

By our separating the concern in such a way, we prevent a single CA from monitoring a consumer's activities. We therefore do not allow a consumer to use master pseudonyms during DRM activities. By utilising an idea inspired by Zhang et al. [ZQM00], we are able to issue subsequent pseudonyms to consumers who identify themselves with a pseudonym.

In the system proposed by Zhang et al. [ZQM00], a consumer is able to acquire a pseudonym after identifying himself with a previously acquired pseudonym (as we discussed in Section 3.3). In doing this, we are able to construct our pseudonym tree, complicating the relationship between a consumer's civil identity and the pseudonym he ultimately uses during the DRM processes. To further improve this, we have utilised a form of identity escrow as we adapted ideas suggested by Camenisch et al. [CL01] to allow only appointed validating agents

to validate a certificate's authenticity (and discussed in Section 3.4). By utilising such an idea, DRM consumers are prevented from using untrusted CAs or DRM entities; these could have a dubious agenda, or may distribute illegitimate copies of contents or licences.

Camenisch et al. introduced a mechanism in which a certificate can only be verified by an appointed verifier. That is, only those entities issued with the right to verify a certificate possess the ability to do so. Here the TTP partitions and encrypts the certificate before issuing it. Now it is only the appointed verifier that is able to decrypt the certificate to learn the full certificate, and subsequently, verify it.

Although we utilise this approach, we do not limit ourselves to a single validating agent; we have amalgamated group signatures with Camenisch et al.'s mechanism. Now if each CA encrypts the certificate with the group's public key, then only members of the group (*i.e.* DRM *or* AUX) will be able to decrypt and read the certificate. This would allow a consumer to choose any CA from the set of all trusted CAs, not limiting him to a single one — providing a more competitive environment, and thus, a more attractive environment for consumers.

Now since the set of validating agents is likely to remain stable, we are able to utilise a simple group signature scheme. A governing group manager could issue each validating agent with knowledge of a single group key pair. Since each CA is intrinsically trusted, we need not concern ourselves with the possibility that a CA is likely to use such knowledge maliciously at this stage of our research. A key pair of such a nature would allow the CA to encrypt the certificate such that only a member of the group could indeed validate it. If we did however doubt the validating agent's agenda, then we could utilise one of the more advanced schemes we mentioned in Section 3.2 such as Chaum et al. [CvH91] or Camenisch et al. [CS97] where group members can be revoked.

From our discussion here, we have highlighted the need to think beyond our set of requirements. Although, in our field of research they are particularly important — and this thesis centres on them — there are additional properties that must be considered to achieve an accountable anonymity service in DRM.

Chapter 6

Conclusions and Directions for Future Work

Throughout this thesis we have investigated accountable anonymity for digital rights management systems. The concept of accountable anonymity is non-trivial, and in itself provides challenges. Applying this to the domain of digital rights management provides additional challenges and considerations. With the incorporation of online payment, accountable anonymity becomes a more critical necessity.

As it stands, the use of DRM systems are few and far between. We believe that this is a direct correlation to the fact that most DRM systems tend to concentrate on the legal aspects of content distribution (*e.g.* copyright [cop10] protection for content owners) and in doing so, neglect the importance of the consumer's privacy. From our discussions in Section 2.2 we know that providing full privacy with total anonymity is impractical; accountable anonymity offers an attractive practical alternative, for all parties. In Section 2.3 we have seen

some DRM systems attempting to provide accountable anonymity but these are insufficient.

In an attempt to satisfy our need to conditionally protect consumers' privacy, we have adopted and adapted existing security primitives to develop a DRM system to satisfy our requirements for an accountable anonymity service in DRM (Section 2.4). To highlight our contributions, we evaluated (Section 5.4) against the existing DRM systems attempting to preserve consumers' privacy; discussing the various strengths and weaknesses. We constructed ten exemplar cases in which an attacker could acquire information to emphasise our achievements with respect to existing DRM systems.

In using our system, we believe that, as proven in Section 5.4, we are able to provide a DRM system providing stronger accountable anonymity than the state-of-the-art solutions discussed in Section 2.3. Now if we refer back to Alice who wants to acquire the Die Hard tetralogy (Chapter 1), by using a DRM service, she is concerned that she would become prone to privacy compromising operations; perhaps resulting with the construction of a dossier of her activities from which trends and interests can be inferred.

To limit such inferences, Alice is interested in anonymity DRM services. Due to the insecurities of existing systems (which we discuss in Sections 2.2 and 2.4), their use is scarce; Alice is unable to find a DRM service that the owner of the Die Hard tetralogy uses to distribute it over. By proving accountable anonymity, we provide a service that is likely to be more attractive for both Alice and the content owner. Alice is assured that her privacy will be preserved if she adheres to the DRM rules and regulations. The content owner is assured that if Alice violates these rule and regulations then her civil identity can be revealed and she can be held accountable for her actions.

In saying that however, it is also important to highlight that we provide a

preliminary solution to an accountable anonymity service in DRM. In its current format, our solution would be unable to support a fully functional accountable anonymity DRM service. For example, in its current format, our solution only supports content and licence acquisition. In the real world, consumers are able to purchase products and then, dependent on its condition, return or exchange them.

Now, under our setting, we do not provide a protocol or procedure in which licences can be returned or even resold. We have only concerned ourselves with conditionally anonymous licence distribution. For our proposed system to be realised in the real world, we should provide consumers with the flexibility they would expect when dealing with a physical alternative. This a possible direction for our future work: allowing consumers to return their licences anonymously but accountably.

In a similar train of thought, we could provide content owners with a greater flexibility of their pricing regime. In our proposed solution, licences are distributed dependent on the receipt of payment (*i.e.* the credential) that the consumer presents to the licence broker. The greater the price paid, the more rights over the content the consumer has. However, this is rather restrictive.

In high-street retailers, such as HMV [hmv10] or WH Smiths [smi10], different DVD/Blu-ray box sets are offered at different prices; depending on the number disks, for example. In our system, we assume that all contents are offered at the same price. The price the consumer pays determines the access rights provided in his licence. To allow owners to vary the price of their contents, as well as licences, would require modifications to our licence payment and acquisition procedure; further adapting the credential proposed by Lysyanskaya *et al.* [LRSW00] to allow us to anonymously pass additional information between the payment gateway and the licence broker.

A particularly interesting approach by Feng *et al.* [FZ08] was to generate the decryption key from the licence. This removes the need to securely distribute it. Provided that it can only be constructed by the properly identified owner of the licence (with *e.g.* a pseudonym), then this would provide an added level security. Bringing this back to our solution, where consumers identify themselves with pseudonyms, the conditional anonymity can still be supported if these licences were bound to a consumer's transaction pseudonym. Tracing the consumer's pseudonym tree could then reveal a malicious consumer's civil identity. This is a particular avenue of interest that future work could be focussed on.

Throughout this thesis we have discussed the current trend in online transactional services (Chapter 1). Since the wide adoption of the Internet, we have seen an upward trend in the use of online transactional services. One such transactional service, DRM, has rather stagnated. However, with the shift from the physical-world to the digital-world continuing to grow, it is likely that DRM, in one form or another, will become an important online service. By building a system incorporating accountable anonymity, no single party is favoured; the rights of all parties are supported.

Bibliography

[AL08] Christer Andersson and Reine Lundin. On the Fundamentals of Anonymity Metrics. In *The Future of Identity in the Information Society*, volume 262/2008, pages 325–241. Springer Boston, 2008.

[ama10] Amazon.co.uk: Low Prices in Electronics, Books, Sports Equipment and more, January 2010. `http://www.amazon.co.uk`.

[App10a] Apple. Apple - iPod shuffle - It's small. It talks. And it's in color., January 2010. `http://www.apple.com/ipodshuffle/`.

[App10b] Apple. Apple - Thoughts on Music, January 2010. `http://www.apple.com/hotnews/thoughtsonmusic/`.

[AT99] Giuseppe Ateniese and Gene Tsudik. Some Open Issues and New Directions in Group Signatures. In *FC '99: Proceedings of the Third International Conference on Financial Cryptography*, pages 196–211, London, UK, 1999. Springer-Verlag.

[BDWY06] Mike Burmester, Yvo Desmedt, Rebecca N. Wright, and Alec Yasinsac1. Accountable privacy. In *Security Protocols*, volume Volume 3957/2006, pages 83–95. Springer Berlin / Heidelberg, September 2006.

[ber92] The world-wide web. In *Computer Networks and ISDN Systems*, volume 25, pages 454–459, 1992.

[BFO05] U. Blomqvist, M. Fritzell, and M. Olofsson. DRM - Intrusion or Solution. In *eChallenges*, October 2005. `http://xml.nada.kth.se/media/Research/MusicLessons/Reports/DRM_Intrusion_or_Solution.pdf`.

[BGK95] Ernie Brickell, Peter Gemmell, and David Kravitz. Trustee-based tracing extensions to anonymous cash and the making of anonymous change. In *SODA '95: Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 457–466, Philadelphia, PA, USA, 1995. Society for Industrial and Applied Mathematics.

[bis97] Group of Ten - Electronic Money - Consumer protection, law enforcement, supervisory and cross border issues. In *Basle Committe on Banking supervision*. Publication of the Bank for International Settlements, September 1997.

[bis98] Risk Management for Electronic Banking and Electronic Money Activities. In *Basle Committe on Banking supervision*. Publication of the Bank for International Settlements, March 1998. `http://www.bis.org/publ/bcbs35.pdf`.

[bis04] International Convergence of Capital Measurement and Capital Standards. In *Basle Committe on Banking supervision*. Publication of the Bank for International Settlements, June 2004. `http://www.bis.org/publ/bcbs107.pdf?noframes=1`.

[bis10] Bank for International Settlements, July 2010. `http://www.bis.org`.

[BJK06] Steven J. Brams, Michael A Jones, and Cristian Klamler. Better Ways to Cut a Cake. In *Notices of the AMS*, volume Volume 53, pages 1314–1321. AMS, 2006.

[BLMT] Vicente Benjumea, Javier Lopez, Jose A. Montenegro, and Jose M. Troya. A First Approach to Provide Anonymity in Attribute Certificates. In *Public Key Cryptography PKC 2004*, volume 2947/2004 of *Lecture Notes in Computer Science*, pages 402–415. Springer Berlin / Heidelberg.

[BMW03] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *Proceedings of Eurocrypt 2003, volume 2656 of LNCS*, pages 614–629. Springer-Verlag, 2003.

[CA99] Mary J. Culnan and Pamela K. Armstrong. Information Privacy Concerns, Procedural Fairness, and Impersonal Trust: An Empirical Investigation. 10(1):104–115, 1999. `http://www.jstor.org/stable/2640390`.

[Cam97] J. Camenisch. Efficient and Generalized Group Signatures. In *Advances in Cryptology — EUROCRYPT*, 1997.

[Cav02] Ann Cavoukian. Privacy and Digital Rights Management (DRM): An Oxymoron? 2002. `http://www.ipc.on.ca/docs/drm.pdf`.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgard. Multiparty unconditionally secure protocols. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 11–19, New York, NY, USA, 1988. ACM.

[CE86] David Chaum and Jan-Hendrik Evertse. A Secure and Privacy-Protecting Protocol for Transmitting Personal Information Between Organizations. In *Advances in Cryptology - CRYPTO '86*, pages 118–167, 1986.

[CFN90] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *CRYPTO '88: Proceedings on Advances in cryptology*, pages 319–327, New York, NY, USA, 1990. Springer-Verlag New York, Inc.

[Cha81] David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. In *Communications of ACM*, volume 24, pages 84–90, New York, NY, USA, 1981. ACM.

[Cha83] David Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology - Crypto '82*, pages 199–203. Springer-Verlag, 1983.

[Cha85] David Chaum. Security without identification: transaction systems to make big brother obsolete. In *Communications of ACM*, volume 28, pages 1030–1044, New York, NY, USA, 1985. ACM.

[CKSJ03] Claudine Conrado, Frank Kamperman, Geert Jan Schrijen, and Willem Jonker. Privacy in an Identity-based DRM system. In *DEXA '03: Proceedings of the 14th International Workshop on*

Database and Expert Systems Applications, pages 389+, Washington, DC, USA, 2003. IEEE Computer Society.

[CL98] Lorrie Faith Cranor and Brian A. LaMacchia. Spam! In *Communications of ACM*, volume 41, pages 74–83, New York, NY, USA, 1998. ACM.

[CL01] Jan Camenisch and Anna Lysyanskaya. An Identity Escrow Scheme with Appointed Verifiers. In *Advances in Cryptology - Crypto 2001, LNCS 2139*, pages 388–407. Springer, 2001.

[CMS96] Jan Camenisch, Ueli M. Maurer, and Markus Stadler. Digital Payment Systems with Passive Anonymity-Revoking Trustees. In *ESORICS '96: Proceedings of the 4th European Symposium on Research in Computer Security*, pages 33–43, London, UK, 1996. Springer-Verlag.

[Coh03] Julie E. Cohen. DRM and privacy. In *Communications of ACM*, volume 46, pages 46–49, New York, NY, USA, 2003. ACM.

[cop10] Intellectual Property Office - Copyright, August 2010. `http://www.ipo.gov.uk/copy.htm`.

[Coy04] Karen Coyle. Rights Expression Languages. 2004. `http://www.loc.gov/standards/relreport.pdf`.

[CP92] David Chaum and Torben Pryds Pedersen. Wallet databases with observers. In *Advances in Cryptology - CRYPTO '92*, pages 89–105. Springer-Verlag, 1992.

[CP93] David Chaum and Torben P. Pedersen. Wallet Databases with Observers. In *CRYPTO '92: Proceedings of the 12th Annual*

International Cryptology Conference on Advances in Cryptology, pages 89–105, London, UK, 1993. Springer-Verlag.

[CP95] Lidong Chen and T. Pedersen. On the efficiency of group signatures providing information-theoretic anonymity. In *Advances in Cryptology — EUROCRYPT*, 1995.

[CPJ04] Claudine Conrado, Milan Petkovicacute, and Willem Jonker. Privacy-Preserving Digital Rights Management. In *Secure Data Management*, volume Volume 3178/2004, pages 83–99. Springer Berlin / Heidelberg, December 2004.

[CPS96] Jan Camenisch, Jean-Marc Piveteau, and Markus Stadler. An efficient fair payment system. In *CCS '96: Proceedings of the 3rd ACM conference on Computer and communications security*, pages 88–94, New York, NY, USA, 1996. ACM.

[CS97] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups. In Burton Kaliski, editor, *Advances in Cryptology - CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 410–424. Springer Berlin / Heidelberg, 1997. 10.1007/BFb0052252.

[CvH91] David Chaum and Eugène van Heyst. Group Signatures. In *Advances in Cryptology EUROCRYPT '91*, pages 257–265. 1991.

[CZ04] D. Critchlow and N. Zhang. Security enhanced accountable anonymous PKI certificates for mobile e-commerce. In *Computer Networks*, volume 45, July 2004.

[DaJCP03] Claudia Daz, Stefaan Seys amd Joris Claessens, and Bart Preneel. Towards Measuring Anonymity. In *Privacy Enhancing Technologies*, volume Volume 2482/2003, pages 184–188. Springer Berlin / Heidelberg, Janurary 2003.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. In *Information Theory, IEEE Transactions*, volume Volume: 22, pages 644–654. IEEE, November 1976.

[Dif88] W. Diffie. The first ten years of public-key cryptography. In *Innovations in Internetworking*, pages 510–527, Norwood, MA, USA, 1988. Artech House, Inc.

[dvl10] Driver and Vehicle Licensing Agency, January 2010. `http://www.dft.gov.uk/dvla/`.

[eba09] What is Feedback?, 2009. `http://pages.ebay.co.uk/help/feedback/about-feedback.html`.

[eba10] eBay — The UK's Online Marketplace, January 2010. `http://www.ebay.co.uk`.

[Ell99] C. Ellison. RFC 2692, SPKI requirements. In *IETF Network Working Group*, September 1999. `http://www.faqs.org/ftp/rfc/pdf/rfc2692.txt.pdf`.

[Far02] S. Farrell. RFC 3281, An Internet Attribute Certificate Profile for Authorization. In *IETF Network Working Group*, April 2002. `http://www.ietf.org/rfc/rfc3281.txt`.

[FFSS02] Joan Feigenbaum, Michael J. Freedman, Tomas Sander, and Adam Shostack. Privacy Engineering for Digital Rights Management

Systems. In *Security and Privacy in Digital Rights Management*, volume Volume 2320/2002, pages 153–163. Springer Berlin / Heidelberg, January 2002.

[fox10] Home — 20th Century Fox Studios, July 2010. `http://www.foxstudios.com/`.

[FPS99] W. Ford, W. Polk, and D. Solo. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. Technical report, Internet Engineering Task Force, Janurary 1999. `http://www.ietf.org/rfc/rfc2459.txt`.

[FZ08] Min Feng and Bin Zhu. A DRM System Protecting Consumer Privacy. In *Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE*, pages 1075–1079, 10-12 2008.

[Goo08] Google. Google Checkout Privacy Policy, February 2008. `https://checkout.google.com/files/privacy.html`.

[hmv10] Buy Music CDs, DVDs, Games, Consoles, Blu Ray, MP3s & More - hmv.com - Free Delivery, August 2010. `http://hmv.com/hmvweb/home.do`.

[HP99] Thomas P. Novak Hoffman, Donna L. and Marcos A. Peralta. Information privacy in the marketspace: implications for the commercial uses of anonymity on the web. 15:129–139, 1999.

[Ian00] Renato Iannella. Open Digital Rights Management. 2000. `http://www.w3.org/2000/12/drm-ws/pp/iprsystems-iannella.pdf`.

[IET10] IETF. Internet Engineering Task Force, January 2010. `http://www.ietf.org/rfc.html`.

[imd10] The Internet Movie Database (IMDb), July 2010. `http://www.imdb.com/`.

[ISO05] Information technology — Security techniques — Evaluation criteria for IT security - Part 2: Security functional requirements. Technical Report ISO/IEC 15408-2:2005(E), International Standard, October 2005. `http://www.commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R2.pdf`.

[JK10] The Guardian Jemima Kiss. Google admits collecting Wi-Fi data through street view cars. `http://www.guardian.co.uk/technology/2010/may/15/google-admits-storing-private-data`, May 2010.

[JY96] M. Jakobsson and M. Yung. Revocable and versatile electronic money. In *3rd MCM Conference on Computer and Communications Security*, pages 76–87, 1996.

[KK02] Steve Kenny and Larry Korba. Applying digital rights management systems to privacy rights management. In *Computers & Security*, volume 21, pages 648–664, 2002.

[Knu98] Donald E. Knuth. *The art of computer programming, volume 3: (2nd ed.) sorting and searching*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1998.

[KP98] Joe Kilian and Erez Petrank. Identity Escrow. In *CRYPTO '98: Proceedings of the 18th Annual International Cryptology Conference*

on Advances in Cryptology, pages 169–185, London, UK, 1998. Springer-Verlag.

[LRSW00] Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf. Pseudonym Systems. In *SAC '99: Proceedings of the 6th Annual International Workshop on Selected Areas in Cryptography*, pages 184–199, London, UK, 2000. Springer-Verlag.

[MAM+99] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. In *IETF Network Working Group*, June 1999. `http://tools.ietf.org/html/rfc2560`.

[Mas10] Mastercard in the United Kingdom — MasterCard, January 2010. `http://www.mastercard.com/uk/`.

[Mer87] Ralph C. Merkle. A Digital Signature Based on a Conventional Encryption Function. In *Advances in Cryptology CRYPTO 87*, pages 369–378, 1987.

[Mic93] Silvio Micali. Fair Cryptosystems. Technical report, Cambridge, MA, USA, 1993.

[Mic10] Microsoft. Digital Rights Management (DRM), January 2010. `http://www.microsoft.com/windows/windowsmedia/forpros/drm/default.mspx`.

[MMJ03] Lindsay Marshall and Carlos Molina-Jiminez. Anonymity with Identity Escrow. In *Proceedings of the Workshop on Formal Aspects in Security and Trust (FAST)*, pages 124–129, Pisa, Italy, September 2003.

[MMW98] Roger C. Molander, B. David Mussington, and Peter A. Wilson. Cyberpayments and money laundering. `http://www.rand.org/pubs/monograph_reports/2005/MR965.pdf`, 1998.

[pay10a] Amazon.com: Express Checkout with Amazon PayPhrase, January 2010. `http://www.amazon.com/gp/payphrase/claim/whats-this.html`.

[pay10b] PayPal Privacy Policy - PayPal, January 2010. `https://cms.paypal.com/uk/cgi-bin/marketingweb?cmd=_render-content&content_ID=ua/Privacy_full&locale.x=en_GB`.

[pay10c] Welcome - PayPal, January 2010. `http://www.paypal.co.uk/uk`.

[Pet97] H. Peterson. How to convert any digital digital signature scheme into a group signature scheme. In *Security Protocols Workshop*, 1997.

[PH08] Andreas Pfitzmann and Marit Hansen. Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management - A Consolidated Proposal for Terminology. Februrary 2008. `http://dud.inf.tu-dresden.de/Anon_Terminology.shtml`.

[Rab79] M. O. Rabin. Digitalized Signatures and Public-Key Functions as Intractable as Factorization. Cambridge, MA, USA, 1979. Massachusetts Institute of Technology.

[Rab89] Michael O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. In *J. ACM*, volume 36, pages 335–348, New York, NY, USA, 1989. ACM.

[Rig03] Rightscom. The MPEG-21 Rights Expression Language. `http://www.xrml.org/reference/MPEG21_REL_whitepaper_Rightscom.pdf`, 2003.

[Riv92] R. Rivest. The MD5 Message-Digest Algorithm, 1992. `http://tools.ietf.org/html/rfc1321`.

[Rob00] Lawrence G. Roberts. Beyond Moore's Law: Internet Growth Trends. In *Computer, IEEE Computer Society*, volume 33, pages 117–119, January 2000.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. In *Communations of ACM*, volume 21, pages 120–126, New York, NY, USA, 1978. ACM.

[sam10] Samaritans Home Page, January 2010. `http://www.samaritans.org/`.

[SD03] Andrei Serjantov and George Danezis. Towards an Information Theoretic Metric for Anonymity. In *Privacy Enhancing Technologies*, volume Volume 2482/2003, pages 259–263. Springer Berlin / Heidelberg, January 2003.

[Sha01] C. E. Shannon. A mathematical theory of communication. In *SIGMOBILE Mobile Computer Communication Rev.*, volume 5, pages 3–55, New York, NY, USA, 2001. ACM.

[SHC07] Hung-Min Sun, Chi-Fu Hung, and Chien-Ming Chen. An Improved Digital Rights Management System Based on Smart Cards. In *Inaugural IEEE International Conference on Digital Ecosystems and Technologies*, 2007.

[SLO94] R. G. Van Schyndel, A. Z. Ltrkel, and C. F. Osbome. A digital watermark. pages 86–90, 1994.

[smi10] WHSmith.co.uk: Books, Stationery, Magazines, Gifts Games & Toys & much more, August 2010. `http://www.whsmith.co.uk/`.

[Son10] Sony. Sony Global - OpenMG, January 2010. `http://openmginfo.com/`.

[SPC95] Markus Stadler, Jean-Marc Piveteau, and Jan Camenisch. Fair blind signatures. In *EUROCRYPT'95: Proceedings of the 14th annual international conference on Theory and application of cryptographic techniques*, pages 209–219, Berlin, Heidelberg, 1995. Springer-Verlag.

[STS99] Tomas Sander and Amnon Ta-Shma. Auditable, Anonymous Electronic Cash Extended Abstract. In *CRYPTO '99: Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, pages 555–572, London, UK, 1999. Springer-Verlag.

[TFS04] Isamu Teranishi, Jun Furukawa, and Kazue Sako. k-times anonymous authentication (extended abstract). In *Asiacrypt, volume 3329 of LNCS*, pages 308–322, 2004.

[ver10] Verisign - Internet infrastructure services for the digital world, security (SSL Certificates), Domain Name Services, DDOS Mitigation and Identity Protection, January 2010. `http://www.verisign.com/`.

[vis10] Visa UK Consumer Website, January 2010. `http://www.visa.co.uk`.

[vL02] Fred von Lohmann. Fair Use and Digital Rights Management: Preliminary Thoughts on the (Irreconcilable?) Tension between Them. `http://w2.eff.org/IP/DRM/cfp_fair_use_and_drm.pdf`, 2002.

[vSN92] Sebastiaan von Solms and David Naccache. On blind signatures and perfect crimes. In *Computer Security*, volume 11, pages 581–583, Oxford, UK, UK, 1992. Elsevier Advanced Technology Publications.

[WDW+05] Xin Wang, Thomas DeMartini, Barney Wragg, M. Paramasivam, and Chris Barlas. The MPEG-21 Rights Expression Language and Rights Data Dictionary. In *Multimedia, IEEE Transactions*, volume Volume: 7, pages 408–417, June 2005.

[WLD+02] Xin Wang, Guillermo Lao, Thomas DeMartini, Hari Reddy, Mai Nguyen, and Edgar Valenzuela. XrML – eXtensible rights markup language. In *XMLSEC '02: Proceedings of the 2002 ACM workshop on XML security*, pages 71–79, New York, NY, USA, 2002. ACM.

[ZQM00] N. Zhang, Q. Shi Q, and M. Merabti. Anonymous public-key certificates for anonymous and fair documentexchange. In

Communications, IEE Proceedings, volume 147, pages 345–350, Deptartment of Computer Science, Manchester University, December 2000.

[ZSM05] N. Zhang, Q. Shi, and M. Merabti. Revocation of privacy-enhanced public-key certificates. In *Journal of Systems and Software*, volume 75, pages 205–214, 2005. Software Engineering Education and Training.