Hardware-based Identification and Authentication Systems

By

Vijayakrishnan Pasupathinathan

A thesis submitted to Macquarie University

for the degree of

Doctor of Philosophy

Department of Computing

December 2009

© Vijayakrishnan Pasupathinathan, 2009.

Typeset in LaTeX 2_ε.

Statement of Candidate

I certify that the work in this thesis entitled "Hardware-based Identification and Authentication Systems" has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree to any other university or institution other than Macquarie University.

I also certify that the thesis is an original piece of research and it has been written by me. Any help and assistance that I have received in my research work and the preparation of the thesis itself have been appropriately acknowledged.

In addition, I certify that all information sources and literature used are indicated in the thesis.

Vijayakrishnan Pasupathinathan (40390705)
December 2009

Abstract

The digitisation of the traditional brick-and-mortar applications has increased our dependence on computing systems. It has also led to an increased usage of such systems, to store and transact, sensitive personal and financial information. The failure to secure such information adequately has resulted in identity theft, credit fraud and related crimes. The lack of protection offered by software-only solutions has led to the development of new security mechanisms, designed using an intermediate hardware device. Such hardware-based security solutions are playing a vital role in many applications that aim to provide security and privacy; this makes it important to study, and solve security issues that arise during the design and implementation of such security solutions. This PhD thesis presents the results of our analysis and design of cryptographic protocols used to construct hardware-based secure systems.

In this thesis, we first analyse the security of the passive hardware devices. Standard cryptographic primitives though provide strong security; they are demanding with respect to, power consumption, memory size and circuitry design, thus making them unsuitable for the design of security solutions for passive devices. To illustrate, we consider an application of passive devices in the supply chain and product identification, and provide a security analysis of the EPC Gen2 standard. To resolve the weaknesses identified, we propose an EPC Gen2 compliant authentication protocol that provides locational privacy.

We then examine the use of hardware devices in identity documents, where the

device capabilities are comparable to those of the semi-passive devices. Here, we consider two recent proposals for electronic passports that rely on hardware-based security. We provide a security analysis of the ICAO first-generation ePassport standard and the EU's proposal for a second generation of ePassports. Resolving the weaknesses identified, we outline our proposal for an online secure ePassport protocol.

Lastly, we analyse the identification and authentication mechanisms in active devices. We propose a new pseudonym system based on the security of preserving a high value secret that solves the security and privacy issues arising from deploying active hardware modules. We further discuss the integration of our proposal based on active hardware devices in an application level security protocol.

Acknowledgements

I would like to express my sincere gratitude to Prof. Josef Pieprzyk for his guidance, support and encouragement, personal and academic, and for numerous conversations that has profoundly influenced my thinking about the subject of this thesis. Equally important was the support and encouragement provided by A/Prof. Huaxiong Wang. Without their insight, knowledge and invaluable suggestions this thesis would have been incomplete.

I would also like to thank my friends and colleagues at the Department of Computing for their suggestion and constructive criticism. I am especially grateful to Venkat Balakrishnan, for his friendship and conversations that has shaped my research in more ways than I am consciously aware of; Joo Yeon Cho, who was always helpful, friendly and a source of inspiration in my research work. I am also grateful to Aarthi Nagarajan, Sarath Indrakanthi, Udaya Tupakula, and my colleagues at ACAC, Christophe Tartary, Qingsong Ye and Gaurav Gupta for their friendship and stimulating discussions. Sincere thanks also to all my previous supervisors and teachers, in particular to Kapalee Vishwanathan, Colin Boyd and Ed Dawson. Their advise and guidance, during my study at Queensland Univeristy of Technology, has been a motivating factor for my interest in cryptography and security in general.

A special acknowledgement is necessary for the technical and administrative staff at the Department of Computing for their continuous effort to facilitate my research activities, and I am especially grateful to Adam Shah and Dilshan Jayarathana for their efficiency and commitment towards supporting my computing needs.

Lastly, but most importantly, I wish to thank my wife and my parents; my gratitude to them is beyond words.

Contents

Abstract **v**

Acknowledgements **vii**

List of Figures **xv**

List of Publications **xvii**

1 Introduction **1**

1.1 Identification and authentication 2

1.1.1 Identity Problem 2

1.1.2 Hardware-based security 3

1.2 Hardware based authentication 4

1.2.1 Application of Hardware Security Modules 5

1.3 Thesis Scope and Outline 8

1.3.1 Organisation 9

2 Background **13**

2.1 Classification of Hardware Devices 14

2.1.1 Based on Communication Mechanism 14

2.1.2 Based on Power Source 14

2.2 Standards for Hardware Identification 15

2.2.1 Standardisation Bodies ISO and IEC 15

2.2.2 Application Specific ISO standards 17
2.2.3 Federal Information Processing Standards 18
2.2.4 EPCglobal RFID standard 19
2.2.5 International Civil Aviation Organization 19
2.2.6 Comité Européen de Normalisation Standards 20
2.2.7 Trusted Computing Group 21
2.3 Cryptographic Mechanisms 21
2.3.1 Hash functions 21
2.3.2 Symmetric key primitives 23
2.3.3 Public key primitives 24
2.3.4 Public Key Infrastructure 26
2.3.5 Key establishment protocols 27
2.3.6 Identification protocols 27
2.4 Formal Methods in Cryptographic Protocols 29
2.4.1 Requirement specifications 29
2.4.2 Protocol modelling 30
2.4.3 Protocol verification 31
2.5 Attacks on Hardware-based Devices 34
2.5.1 Physical Attacks 34
2.5.2 Communicational Attacks 35
2.6 Security Goals for Hardware-based Devices 36
2.6.1 Essential Security Goals 37
2.6.2 Desirable Security Goals 37

3 Hardware-Based Product Identification 39
3.1 EPCglobal Class 1 Generation 2 and its Security 40
3.1.1 EPC Gen2 Memory Requirements 41
3.1.2 EPC Gen2 Functions 42
3.1.3 EPC Gen2 Operations 44
3.1.4 Security Issues with EPC Gen2 45

3.2 Related Work 47
3.3 Mutual Authentication Protocol for EPC Gen2 compliant Tags 50
3.3.1 Preliminaries 50
3.3.2 Security Goals 51
3.3.3 Protocol Description 52
3.4 Security Analysis 53
3.5 Summary 56

4 Hardware-based Security in Identity Documents 59
4.1 Electronic Identity Documents 60
4.2 ICAO ePassport Specification 61
4.2.1 Operation of ePassport 62
4.2.2 Data Structure 62
4.2.3 ePassport PKI 63
4.2.4 Passive Authentication 65
4.2.5 Active Authentication 66
4.2.6 Basic Access Control 67
4.2.7 Key Derivation 68
4.3 Analysis of ePassport 68
4.3.1 Security Goals 69
4.3.2 Related Work 71
4.3.3 Formal Verification 72
4.3.4 Modelling ePassport protocols 73
4.4 Verification Using Casper/FDR 74
4.5 Summary 81

5 Securing ePassports 83
5.1 Extended Access Control 84
5.1.1 ePassport Operation with the EAC 84
5.1.2 Chip Authentication 86
5.1.3 Terminal Authentication 87

5.1.4 Security issues in EAC-based ePassports 87
5.2 An On-line Secure ePassport Protocol 90
5.2.1 Initial Setup 91
5.2.2 Phase One - IS Authentication 91
5.2.3 Phase Two - ePassport Authentication 93
5.3 Analysis of ePassport scheme 94
5.3.1 Requirement Analysis 94
5.3.2 Security Analysis of OSEP protocol 95
5.4 Summary 100

6 Privacy Enhancements for Active Devices 103
6.1 Privacy Issues with Active Devices 103
6.2 Pseudonymns 106
6.2.1 Scope and Contribution 107
6.2.2 Anonymous Certification System 108
6.3 Pseudonym System Colligated with Master Secret Key 111
6.3.1 Assumptions 111
6.3.2 System Setting 112
6.3.3 Identity Generation 113
6.3.4 Certification 113
6.3.5 Identification 115
6.3.6 Tracing 116
6.4 Security 116
6.4.1 Adversary Goals 116
6.5 TPM Integration 120
6.6 Summary 120

7 HSMs in Application Level Security Protocols 123
7.1 Electronic Procurement 124
7.1.1 Related Work 125
7.1.2 E-tendering vs. Auction Systems 126

7.1.3 A Generic E-Tendering System 127
7.1.4 Security Requirements for E-Tendering 129
7.2 A Secure E-Tendering System 130
7.2.1 System Setting 131
7.2.2 Registration 132
7.2.3 Submission 133
7.2.4 Trace 135
7.3 Security Analysis 136
7.4 Summary 139

8 Conclusion **141**
8.1 Open Problems 143

A Verification using Casper and FDR **147**
A.1 Modelling protocols in Casper 147
A.2 Interpreting the FDR output 152

B Casper Representation of the ICAO First-generation ePassport Protocols **157**

C CK Model **161**
C.1 Attacker Model 161
C.2 Session Key Security 162

D Trusted Platform Module **165**

References **167**

List of Figures

3.1 EPC Gen2 Memory Banks 42
3.2 RFID Communication Channels 47
3.3 EPC Gen2 Compliant Mutual Authentication Protocol 54

4.1 ICAO Public Key Infrastructure 64
4.2 Passive Authentication 65
4.3 Active Authentication 66
4.4 Basic Access Control 67

5.1 EAC Public Key Infrastructure 85
5.2 Chip Authentication 86
5.3 Terminal Authentication 87

6.1 Pseudonym Certification 114
6.2 Identification of Colligated Pseudonyms 115
6.3 Tracing Colligated Pseudonyms 116

7.1 A Generic E-Tendering System 128
7.2 E-tender Registration 132
7.3 E-tender Submission Phase One 133
7.4 E-tender Submission Phase Two 134
7.5 Trace Wining Tenderer 135

List of Publications

Vijayakrishnan Pasupathinathan, Josef Pieprzyk, Huaxiong Wang *Formal Security Analysis of Australian E-passport Implementation.* Sixth Australasian Information Security Conference, Wollongong, Australia, 2008.

Vijayakrishnan Pasupathinathan, Josef Pieprzyk, Huaxiong Wang *Security Analysis of Australian and E.U. E-passport Implementation.* Journal of Research and Practice in Information Technology, Vol. **40**, No. 3, August 2008.

Vijayakrishnan Pasupathinathan, Josef Pieprzyk, Huaxiong Wang *An On-Line Secure E-Passport Protocol.* Information Security Practice and Experience - ISPEC'08, Sydney, Australia. Lecture Notes in Computer Science, Vol *4991* pp. 14-28, 2008.

Vijayakrishnan Pasupathinathan, Josef Pieprzyk, Huaxiong Wang *A Fair E-Tendering Protocol.* International Conference on Security and Cryptography (SECRYPT 2008), Porto, Portugal. pp. 294-299, 2008

Vijayakrishnan Pasupathinathan, Josef Pieprzyk, Huaxiong Wang *Certified Pseudonyms Colligated with Master Secret Key.* International Conference on Security and Cryptography (SECRYPT 2009), Milan, Italy, 2009

Vijayakrishnan Pasupathinathan, Josef Pieprzyk, Huaxiong Wang *Mutual Authentication for EPC Gen2 Devices.* Technical report, preliminary version available from `http://www.vkrishnan.com`, 2009.

1

Introduction

Since the innovation of integrated circuits, advances in semiconductor technologies have continued to grow at an incredible pace. Computing devices have become smaller and faster, capable of performing complex computing and communicational tasks. These advances have enabled manufactures to embed these *micro-chips* into products that range from cars, microwave ovens, books, packaging materials, medical supplies, to other physical objects that traditionally have not relied on electronics. This blend between physical objects and electronic hardware has had a significant impact on an array of applications, such as supply chain inventory control [179], border control [96, 110], road tolls [186], livestock and asset tracking [192], currency tracking [111], pharmaceutical product and patient identification [107].

However, together with these advantages, many security concerns have arisen, primarily relating to identification and authentication. The increased deployment of such

hardware-assisted products in sensitive areas, such as health care, treasury and border control has also heightened the need to develop secure protocols that protect these microchips and the associated information stored within them. The goal of this thesis is to evaluate the cryptographic protocols employing hardware-based technologies to construct secure systems, in particular, identification and authentication of the physical objects with the embedded hardware.

1.1 Identification and authentication

Identification is a necessary first step occurring prior to authentication. Informally, we can define *identification* as, the act of claiming an identity, where an identity is a set of attributes that distinctly determines the entity, and *authentication* as the act of verifying that identity.

1.1.1 Identity Problem

In a traditional society, an entity's identity is based on physical attributes, *habeas corpus*, and attributes based on perceptual observations, *empiricism*. During authentication, a claim was made about the identity by presenting the entity, together with the attributes that could uniquely establish the identity as evidence to a verifier. The attributes presented (either individually or collectively) formed a unique relationship with the person presenting the claim. If a verifier were satisfied with the evidence presented, the verifier would then authenticate the entity.

In contemporary society, identification relies on computing systems that typically communicate remotely through an electronic medium. Such electronic transactions are corpus-free; the only thing a verifier may see is the claim itself. Because the traditional connection between the physical body and the attributes presented is no longer present, the possibility of an identity claim being misrepresented is high.

Since the development of computing systems there has been a great emphasis on developing software solutions, and this is true even with an identification system. There have been numerous proposals that provide various identification and authentication

techniques (to highlight a few, [43, 45, 58, 76, 146, 169]). Security solutions based on software implementations are prone to a variety of attacks. A primary concern with software-based solutions is that they are prone to code tampering, leading to the exposure of the confidential data. The storage of data can be protected by encryption, but no software based secure solution exists that can protect the encryption key itself, because the key is required to be stored in plain text in a computer's memory during the encryption and decryption of the data.

The rapid increase in the use of computing devices to store and transact, using sensitive personal and financial information together with the failure to secure this information has resulted in identity theft and related crimes. Protection by using encryption and passwords is a good first step, but the encryption keys and the software used to access the information must also be protected.

1.1.2 Hardware-based security

The above-mentioned problems have led to the development of new security mechanisms that are designed by using an intermediate hardware device, commonly known as *hardware-based security*. Such hardware devices[1] have the ability to tie a system or an object together with its users to the physical world. Hardware-based security incorporates a wide range of topics that includes random number generators, cryptographic co-processors, smart cards, identity tokens, location awareness, biometric devices and trusted computing systems.

A key underlying feature (and a requirement) in a hardware security module is that it can be physically secured against tampering. Cryptographic tools provide data security and enable secure communications; using tamper-resistant hardware further enhances the security by providing protection for the cryptographic keys used. When combined with carefully developed software, hardware-based security solutions can also offer significantly stronger overall system security, because hardware devices provide a trusted and non-observable execution environment to process confidential data. Hardware-based security also makes security transparent to an end-user and improves

[1] Also referred to as Hardware Security Modules (HSM)

the performance of the security solution. Furthermore, the increased dependence on electronic networks for communicating sensitive information has also heightened the need to develop systems that protects the user's information through a tamper-evident and tamper-resistant hardware.

1.2 Hardware based authentication

Typically, authentication involves one or more factors, which help to determine whether an entity is in fact who he/she claims to be. These factors are:

1. **Knowledge**: Something the user *knows*, e.g. passwords.

2. **Biometrics**: Something the user *is*, e.g. fingerprints, DNA.

3. **Possession**: Something the user *has*, e.g. hardware tokens.

Something the user knows, is the most common method for user authentication in a computer-security system. Most systems employ a simple PIN (Personal Identification Number) or password as a data authenticator. Because password-based authenticators typically tends to be software-based, they are prone to various attacks and vulnerabilities [86, 115, 123], from both human and software.

Biometric data authenticators belong to the second category, *something the user is*. Most common biometric data authenticators are facial recognition, fingerprints and iris scans. The biometric data is captured using a terminal and processed against an existing template for verification. A major advantage of biometric data authenticator is that the data is static, unique and protects against credential transfers. The unique nature of the biometric data is also its primary disadvantage, because compromised biometric data cannot be replaced.

Hardware-based security belongs to the third category, *something the user has*. In this category, the user controls a hardware device that can be used to demonstrate their identity. In subsequent chapters, we highlight many variants of such devices that employ various security mechanisms to provide authentication. An advantage of such

devices is that they can provide stronger security by employing cryptographic tools, because of their superior processing and storage capabilities.

The main concern with hardware-based and knowledge-based authenticators is that they are prone to credential transfers. Because these authenticators are not linked to any specific entity, they can be easily transferred between different entities. Biometric authenticators are not prone to this weakness, because it is not easy for a user to transfer his physical attributes. Each factor on its own is relatively weak and prone to various attacks, but the combination of more than one factor yields a much stronger authentication mechanism. In a multi-factor authentication system, individual factors can compensate for the disadvantages present in each, such as the transferability of possession and the non-replicability of biometric data. Consequently most hardware systems typically employ authenticators from multiple factors.

1.2.1 Application of Hardware Security Modules

Using specialised hardware to provide security features has been available since the 1970's [23, 56, 84]. However, because of the limitations on the storage and processing capabilities, these hardware devices were restricted only to a few specific applications [191, 193, 197]. New refinements in both electronics and cryptography have led to a new generation of hardware-based security applications, in particular towards deploying multi-factor authentication systems. Here, we highlight three classes of applications employing hardware-based security specifically employed towards the identification and authentication of entities.

Inventory and Supply Chain Management

Effective inventory management depends upon identifying, consolidating, integrating, and analysing data collected from multiple sources. Conventional identification and tracking systems required manual intervention that was labour-intensive, time consuming, and error-prone. The introduction of barcodes helped to speedup the process.

However, barcodes are very limited, because the reading of barcodes requires line-of-sight, they cannot be read over non-conducting material such as cardboard, and they can be read only at short distances. The introduction of Radio Frequency Identification (RFID) tags not only solved many issues that barcodes have, but also provided extra functionalities. The microchips within the RFID tags have the ability to store a large amount of information about the products they are embedded in and therefore capable of uniquely identifying individual items even within the same brand of products. Because of the RF communication channels, RFID readers also have the ability to read product information at considerably faster rate than barcode readers.

These advantages have seen RFID technology applied in various commercial sectors, for example, at various stages of a supply chain and for inventory management [108, 127, 172], animal tracking [89], tracking pharmaceutical products [151], tracking passenger baggage in the airline industry [52], inventory and tracking library books [171], and including currency tracking and identification [111, 202].

The main hindrance to the widespread adoption of these low-cost RFID tags has been security and, in particular, privacy. Because of their low-cost, their microchip circuitry is resource-limited, typically capable of storing only a few hundred bits of data. They approximately have between 5000 and 10000 logic gates for overall computational purposes, of which only around 500 to 4000 gates are available for security related computations. Therefore, the traditional cryptographic mechanisms are no longer suitable; for example, a standard implementation of the Advanced Encryption Standard (AES) requires between 20K and 30K gates, far beyond what is available. Also, because of the tag's passive nature, there are even power and communicational restrictions. To solve these problems, there is a need for a new approach using light-weight cryptographic tools that can guarantee a high-level of security guarantees.

Electronic Identity Cards and Passports

The need to monitor and strengthen both internal and border security have prompted many countries to adopt microchip-enabled ID cards and passports. Currently, over 45 countries have adopted biometric-enabled passports and over 15 countries have adopted

an electronic national identity card. Though the main motivating factor was to enhance national security, identify potential criminals, mainly terrorists and protect against illegal immigration, this technology has also enabled in enhancing other applications, such as library cards, health care cards, driver's licenses and governmental benefit programs.

Though there exists separate standards (*cf.* Section 2.2) that differ in their implementation, the underlying communicational and security mechanisms are similar. Both electronic ID cards and electronic passports employ a microchip that can communicate with a terminal using radio frequency (RF), and employ cryptographic modules such as digital signatures, encryption and PKI (Public Key Infrastructure). For example, both electronic ID cards and electronic passports employ ISO/IEC 11770-2 *Key Establishment Mechanism 6*, ISO/IEC 9797-1 *MAC algorithm 3* and a contact-less microchip that conforms with ISO/IEC 14443.

Trusted Computing Platform

The Trusted Platform Module (TPM) [187, 188] is a secure cryptographic chip that provides hardware-based security functionalities, such as, terminal identification, user authentication, data protection and cryptographic key storage. A key advantage of the TPM is that it can be combined with widely-used enterprise hardware such as computer terminals, network policy enforcement points (such as, firewalls, switches and routers), mobile devices, and can natively support PKI. The TPMs complement biometric readers, because they can store securely the biometric templates and associated keys. Because the TPMs are an active device integrated within a device's circuit board, they are capable of performing complex cryptographic functions at a high communicational data rate.

The deployment of the TPM raises some valid privacy concerns. Each TPM has a unique key-pair, called the *endorsement key* that identifies the device and the platform. Authentication based directly on the TPM's key will compromise the privacy of the module, because all transactions performed by the same TPM can be linked. Furthermore, it also compromises the privacy of the user associated with the module.

Such concerns have also led to parliamentary inquiries [95] to ascertain whether the TPM technology can have a negative impact on user privacy and whether it creates customer lock-in.

In spite of these concerns, the TPM's technical advantages have helped in deploying them on a very large scale. Currently, there are about 130 organisations that are either promoters, contributors or adopters of the trusted computing group (TCG) specifications [3].

1.3 Thesis Scope and Outline

Hardware-based security is continuously emerging and being deployed in many new applications. Apart from the applications mentioned in the previous section, hardware-based identification and authentication can also be found in payments systems, voting systems, mobile phones, digital broadcasting, multimedia devices, such as, DVD players, virtualisation systems, transit and travel cards, and many others.

Though the number of applications of such hardware devices has grown, fundamentally, these devices can be classified into three groups (*cf.* Section 2.1) that distinguish their computational and communicational capabilities and thus determine the cryptographic functions that can be employed. Therefore, in this research work, we focus on developing cryptographic tools that can achieve the security goals distinct to each class, rather than each application. Also, it is important to note that not all systems need, or are capable of, offering the same level of security.

For example, RFID devices used in electronic product codes are capable of generating only a 16-bit pseudo random number for security purposes. Because of this low level of security, one can argue that such RFID devices are vulnerable, even to cipher-text only attacks that can exhaust the 16-bit range. Though such attacks are certainly feasible, one needs to consider application-specific aspects (such as, the life time of the RFID devices, cost and purpose).

In this research work, we purse the following two goals:

1. The analysis of the existing security mechanisms in each class of hardware-based

security devices, passive, semi-passive and active devices.

2. The development of stronger security mechanisms that achieve the security goals distinct to each class of hardware-based security.

1.3.1 Organisation

The subject matter of this thesis is organised such that it can be visualised as falling into the following groups

- **Chapter 2: (*Background*)** This chapter surveys the background theory on the subject matter, beginning by presenting a classification scheme based on the communication medium and based on the powering mechanism. This is followed by an overview of the existing international standards that define hardware-based security. We then provide an overview of the cryptographic notions and protocols necessary to understand identification and authentication in hardware-based security protocols, and present security goals that are used to measure the security characteristics of the protocols presented in this thesis.

- **Chapter 3: (*Passive Devices*)** This chapter deals with resolving the security issues in the identification schemes for passive devices, primarily in the wireless (radio frequency) medium. Recently, there has been a tremendous growth in their application towards product identification and supply chain management, where the current de-facto standard is EPC Gen2. In this chapter, we start by providing a brief overview of the EPC Gen2 standard, and then identify its security weaknesses to show that it is vulnerable to various attacks. After analysing the weaknesses identified, we outline our proposal that extends the previous research work to provide mutual authentication. We then present a security analysis of our proposal and show that our protocol satisfies the security goals for passive devices, and also provide new features, such as, location privacy and un-traceability. Most of the material in this chapter was obtained in joint work with Josef Pieprzyk and Huaxiong Wang. An earlier version of the results presented in this chapter can be found at [157].

- **Chapter 4 and 5: (*Semi-Passive)*** In these chapters, we focus on the hardware devices that are either semi-passive in nature or passive devices that are computationally more capable and are comparable to that of the semi-passive devices. We primarily focus on the use of hardware security modules in identity documents and concentrate on two most prominent applications, electronic passports and identity cards. We initially present a brief overview of electronic document identification and the factors that have accelerated their uptake. We then provide a technical overview of ePassport implementations, both for first generation ePassports, (ICAO proposal) [96] and second generation ePassports, (EU Proposal) [94]. We then provide a formal security analysis of ePassport implementations using model checking, and identify the security flaws present in both proposals. To resolve those security issues, we propose an on-line secure authentication mechanism for electronic passports. Based on our formal analysis, the first generation ePassport is shown to be vulnerable, in the joint work with Josef Pieprzyk and Huaxiong Wang [153]. Our proposal for an online secure ePassport protocol resolving the weaknesses in the second generation ePassports was presented in [154]. An extended security analysis for both ICAO and EU proposal was also published in a journal [155]. The materials presented in these chapters are a combination of analysis and the results that have appeared in the corresponding three publications.

- **Chapter 6 and 7: (*Active Devices)*** In this group, we focus on active devices and examine emerging identification and authentication mechanisms that provide a hardware-based security endpoint, such as trusted platform modules. We start by analysing various security and privacy concerns that arise when deploying such hardware security modules. We then propose a new pseudonym system based on the security of preserving a high value secret, this intends to resolve those security and privacy issues. We then provide a security analysis of our proposed construction and discuss the integration of our proposal in a TPM based setting. Next,

we evaluate active hardware security modules applied to an application-level security protocol and, as an example, consider an electronic commerce application, electronic tendering. We provide an overview of the main components in an e-tendering system and define its security requirements. We then describe our proposal for a secure e-tendering system that satisfies the security requirements identified previously. An earlier version of our proposed pseudonym system providing restricted anonymity and supporting colligation between a trusted high value secret key and generated pseudonyms was presented in a joint work with Josef Pieprzyk and Huaxiong Wang [156]. Our proposal for a publicly verifiable fair e-tendering system using hardware devices appeared at [152].

We then conclude with some open problems and future work in our final chapter.

2

Background

The presentation in this chapter is divided into three parts. The first section presents two classification schemes for hardware devices and identifies existing international standards applicable to hardware-based security. In the second section, we provide an overview of the cryptographic notions and protocols necessary to understand identification and authentication techniques used in hardware-based security protocols and also provide an overview of the verification techniques for such protocols. Owing to the vastness of the literature, we present only those notions and techniques that are applicable to hardware-based security protocols. In the third section, we identify key security goals for designing hardware-based security protocols that also serve as a guideline against the protocols that are analysed in subsequent chapters.

2.1 Classification of Hardware Devices

2.1.1 Based on Communication Mechanism

A hardware device communicates either using a wireless medium, or using a wired medium. Based on the way the device communicates, we can classify a device as:

- **Contact-based**: Contact-based hardware devices require physical connectivity with the terminal to allow communication. Typically, such devices have contact pads that supply the necessary energy and communicate through direct electrical contact with the reader. Examples of such devices include contact-based smart-cards and TPMs.

- **Contact-less**: Contact-less hardware devices communicates with a terminal by means of radio frequency (RF). In most cases, a micro-chip and an antenna are integrated into the body of the hardware device, for example, RFID tags and ePassports.

Many hardware devices now are equipped with dual-interface, that is, with both contact and contact-less capabilities. Typically, such dual-interface devices sharing the same data are based on an application's requirement, and can communicate through the interface that is most suited to its capabilities. For example, in dual-interface devices contact-based are used for the transaction of large data, such as PKI credentials, and communicate contact-less for mutual authentication.

2.1.2 Based on Power Source

A hardware device can either obtain power from the signal it receives or it can have its own internal source of power. Based on how power signals are received, we can classify a device as:

- **Passive**: Passive devices do not have an internal source of power. They harvest their power from the RF electromagnetic energy sent by the reader. They are restricted in their read/write range, because they rely on the RF energy from

the reader for both power and communication. Even though they may contain a battery to maintain volatile memory or to power sensors, they rely entirely on the energy of the incoming signal to respond, example: RFID tags.

- **Active**: Active devices have a power source that powers the microchip's circuitry and broadcasts signals to the reader. Active devices can also contain a battery to generate a device's signal or even boost the signal strength to increase its range. Example: SaviTags, TPMs.

- **Semi-Passive**[1]: Semi-passive devices are an extension of passive devices, but have additional capabilities with respect to processing and communication. Typically semi-passive devices use a battery to run the microchip's circuitry and communicate by harvesting power from the reader's signal, example: E-tolls and ePassports.

2.2 Standards for Hardware Identification

Standards are an important aspect of hardware-devices that help achieve inter-operability among various participating organisations and vendors. Standards typically describe the physical and logical characteristics, covering aspects such as communication protocols, anti-collision mechanisms and security functions. In this section we list the most dominant standards in relation to hardware-based identification systems.

2.2.1 Standardisation Bodies ISO and IEC

The International Organisation for Standardisation (ISO) and International Electrotechnical Commission (IEC) are the world's leading standard development organisations. ISO specifies the requirements for products, services, processes and systems, and the IEC prepares international standards for all electrical, electric and related technologies.

[1]Devices that fall into this category have also been commonly labelled as Battery Assisted Passive (BAP)

The following ISO/IEC standards apply to hardware-based devices used in identification systems.

ISO/IEC 7816

ISO/IEC 7816 [102] is an international standard related to electronic identification cards with *contacts*, particularly smart cards. ISO/IEC 7816 is a fourteen part international standard document, where Parts 1-3 deals with contact smart cards and defines various aspects of the card and its interfaces, including the card's physical characteristics, electrical interface and communications protocols.

ISO/IEC 7816 Parts 4-6, 8-9, 11, 13 and 15, are relevant to all types of smart cards (both contact and contact-less) and define the elements, such as, cryptographic services, biometric verification, logical structure (files and data elements) and various Application Programming Interface (API) commands.

ISO/IEC 7816 Part 10 is used by memory cards for applications such as prepaid telephone cards or vending machines, whereas Part 7 defines a secure relational database approach for smart cards based on the SQL interfaces.

EMV: Europay, MasterCard, and Visa (EMV) publishes standards in relation to financial smart cards. The EMV specification is built to the ISO/IEC 7816 standard, but expanded to accommodate debit and credit transactions. The four books of the EMV standard describes the minimum functional, security, procedural requirements for financial smart cards and terminals.

ISO/IEC 7810

ISO/IEC 7810 [101] defines contact-less identification cards. Based on communicational characteristics, such cards can be distinguished into the following three types:

- ISO/IEC 10536 [98] (Close-coupled cards): These cards operate at a very short distance from the reader (0-1 mm). The standard was originally intended to be a direct replacement for the standard covering contact cards, but ISO 10536 is now very rarely used.

- ISO/IEC 14443 [99] (Proximity cards): These cards operate at an approximate distance of 0-10 centimetres from the reader. It describes two types: Type A and Type B, the main difference being the modulation methods, coding schemes and protocol initialisation procedures. Both type of cards use the same high-level protocol.

- ISO/IEC 15693 [100] (Vicinity Cards): This describes cards operating at a frequency of 13.56 MHz with a maximum read distance of 1-1.5 metres and are usually only incorporated in expensive machines.

NFC Forum: Near-Field Communication (NFC) is designed for interactions between devices in close proximity (0-10 cm). The NFC Forum is working on technical specifications that will help drive the interoperability of the technology in all market sectors. The Near Field Communication Interface and Protocol-2 (NFCIP-2) [149] specifies the communication selection mechanism where NFCIP-2 compliant devices can enter in three different communication modes, NFCIP, ISO 14443 and ISO 15693.

2.2.2 Application Specific ISO standards

ISO/IEC 18013 [103] establishes guidelines for the content and formatting of the data stored on an ISO compliant driving licence (IDL) with regard to human-readable features, machine-readable technologies, and access control, authentication and integrity validation. It is currently being deployed in countries such as Japan to combat counterfeiting, to streamline license administration, improve driver convenience and protect driver privacy.

ISO/IEC 7501 [104] describes machine-readable passports (MRPs) and consists of three parts. Part 1 of this standard is for passports, Part 2 for visas and Part 3 for travel documents. It also defines the specifications to be used by countries wishing to issue an electronically-enabled MRP (ePassport) access to an expanded set of details, including biometric data.

ISO/IEC 11889 [106] defines the Trusted Platform Module (TPM) and contains four parts. Part 1 describes trusted platform concepts; Part 2 defines the principles of

TPM operations including base operating modes, cryptographic algorithms and key sizes for the algorithms, basic protocols and their usage; Part 3 defines the structures and constants that enable the interoperability between the TPM implementation; and Part 4 defines the commands for APIs to provide TPM functionality.

ISO/IEC 24727 [105] is independent of the physical interface technology and does not cover internal implementation within the card or the outside world. It applies to cards accessed by one or more of the following methods, contacts, close coupling and radio frequency. ISO/IEC 24747 specifies the system architecture and the principles of operation, capabilities, discovery mechanism and security rationale. ISO/IEC 24727 is a set of programming interfaces for interactions between integrated circuit cards and external applications to include generic services for multi-sector use. The organisation and the operation of ICC conforms to ISO/IEC 7816-4.

There are other standards that have some application-specific requirements, but most of the technical implementation still refers to ISO/IEC 7816 or ISO/IEC 7810. For example, ISO 11784, ISO 11785 and ISO 14223 standardised tags for animal identification; ISO/IEC 6346 Freight containers - Coding and marking; ISO/IEC 14816 road traffic and transport telematics; ISO/IEC 1736X and ISO 10374.2 relate to large shipping containers (17363 Freight containers, 17364 Returnable transport items, 17365 Transport units, 17366 Product packaging, 17367 Product tagging).

2.2.3 Federal Information Processing Standards

Federal Information Processing Standards (FIPS) publications are issued by the National Institute of Science and Technology (NIST). The FIPS standards are designed to protect the US Federal computer and telecommunications systems. The following FIPS standards apply to hardware-based technology:

FIPS 140 [68]: This standard applies to areas related to secure design and the implementation of a cryptographic module, specifically: cryptographic module specification, cryptographic module ports and interfaces, physical security, cryptographic key management, design assurance and the mitigation of attacks.

FIPS 201 [69]: Provides the specifications for a standard Federal smart ID card,

called the Personal Identity Verification (PIV) card that is to be used for both physical and logical access. The PIV card is a smart card with both contact and contact-less interfaces.

2.2.4 EPCglobal RFID standard

EPCglobal is a joint venture between GS1 and the Uniform Code Council (UCC). It was formed to develop standards for the Electronic Product Code (EPC) to support the use of RFID devices. To encourage the adoption of RFID technology and the support inter-operability amongst the various vendors, EPCglobal has defined the standards for tag and reader interfaces, and communication protocols. EPCglobal currently classifies RFID tags into four classes: Class 1 identity tags; Class 2 higher-functionality tags; Class 3 semi-passive tags; Class 4 active tags. The predominant class in product identification and supply chain, is Class 1, where the current standard for tag and reader communication is EPC Class 1 Generation 2 UHF Air Interface Protocol [66] (commonly known as 'EPC Gen2'). This standard has been ratified by both EPCglobal and ISO, and can be considered as the *de facto* standard for low-cost RFID tags.

2.2.5 International Civil Aviation Organization

The International Civil Aviation Organisation (ICAO) is the United Nation body responsible for setting international passport standards. It is responsible for issuing guidance on the standardisation and specification for Machine Readable Travel Documents (MRTD). The ICAO standard DOC 9303 [96] for MRTD describes a contact-less smart card microchip that conforms with ISO-14443 and is embedded within an ePassport booklet. The microchip duplicates the information that appears on a passport's bio-data page and data recorded in the Machine Readable Zone (MRZ). The ePassport standard provides details about establishing a secure communication between an ePassport and an Inspection System (IS), the authentication of an ePassport, details on storage mechanism and biometric identifiers that should be used.

The European Commission (EC) is also involved in defining the standards for ePassports issued by member states. For example, its regulation 2252/2004 [67] defines the standards for the security features and biometrics in passports and travel documents. The regulation describes the technical specification to enable biometric markers to be included on travel documents. The first phase included facial biometric images on all new ePassports and, the second phase included the use of fingerprints as a second biometric marker. It also specified that information stored in the second generation of ePassports must be protected by Extended Access Control (EAC).

2.2.6 Comité Européen de Normalisation Standards

The Comité Européen de Normalisation (CEN) contributes to the objectives of the European Union (EU) and the European Economic Area (EEA) with voluntary technical standards to promote interoperability. The work of most interest to the hardware-based industry is undertaken through the Technical Committee 224 (TC224) that covers personal identification, electronic signature and cards. The CEN has now published a number of standards covering applications such as ID card systems, the European Citizen Card (CEN/TC 224/WG 15), transport card (CEN/TC 224/WG 11), machine-readable cards, IC cards for payphones and ePurse. In addition to the standards already published, a number are still being developed by the CEN, such as prEN 1332-3 Identification card systems - Man-machine interface, prEN 14890-1 Application Interface for smart cards used as secure signature creation devices, and prEN 1332-1 Identification card systems - Human-machine interface.

CEN Technical Specification 15480, Identification card systems - European Citizen Card, physical, electrical and transport protocol characteristics, was published in May 2007, covering ID cards, smart cards, identification methods, travel and administrative documents, data processing, data security, and cryptography for eID cards issued in Europe. It describes the ID card as a chip card conforming to ISO/IEC 7186-1 and -2 standards for the contact interface and ISO/IEC 14443 for the contact-less Interface.

2.2.7 Trusted Computing Group

The Trusted Computing Group (TCG) is an international industry standards group that develops and promotes Trusted Computing, which is based on a hardware root-of-trust. The TCP developed the Trusted Platform Module (TPM) [188], a secure cryptographic chip that provides hardware-based security functionalities such as, terminal identification, user authentication, data protection and cryptographic key storage. A TPM consists of a unique *endorsement key* (EK) pair that is built into the hardware module during manufacture. The public part of the EK is certified by the manufacturer and the secret part is sealed inside the TPM and is never revealed to the outside. A primary function of the TPM is attestation that is, the TPM provides guarantees to a remote service that the platform has not been tampered with and is therefore secure. A TPM can also provide other security services, such as secure boot and sealed storage. ISO/IEC 11889 is an effort to have the trusted computing specifications standardised by ISO.

2.3 Cryptographic Mechanisms

In this section we provide an overview of the cryptographic tools aimed towards understanding the cryptographic specific implementations in hardware security modules. A major source for the notation and definitions used in this thesis is from the handbook of applied cryptography [141] and Goldreich's notes [79, 81].

2.3.1 Hash functions

A hash function takes a message as an input and produces an output referred to as a hash value, where the hash-value serves as a compact representative image of an input string, and can be used as if it were uniquely identifiable with that string.

Definition 2.3.1. *A hash function is a mathematical function* $\mathbf{H}$ *with the following two properties:*

- (*Compression*) *Maps a variable length input string* $\{x\}$ *to a fixed length* (n) *output string* $\{y\}$*, called hash-value or digest.*
- (*ease of computation*) *Given an input* x, $\mathbb{H}(x)$ *is easy to compute.*

According to [141], hash functions can either be, unkeyed (or simply hash functions) or keyed (message authentication codes).

Unkeyed Hash Functions

In addition to the above properties, If an unkeyed hash function is used for cryptographic purposes, it is required to have one or more of these three properties:

- (*Collision-resistant*) Where it is computationally infeasible to find the same hash-value for two sets of distinctive inputs. That is, it is computationally infeasible to find a pair (x, x') such that $\mathbb{H}(x) = \mathbb{H}(x')$.
- (*One-Wayness*) For a given x, it is easy to compute $y = \mathbb{H}(x)$, but computationally infeasible to find the input x, from a given y.
- (2^{nd} *Preimage-resistant*) Given an input x, it is computationally infeasible to find another input $x' \neq x$, such that $\mathbb{H}(x) = \mathbb{H}(x')$.

Message Authentication Codes

A message authentication code (MAC) algorithm is a family of functions $\mathbb{H}_k$ parameterised by a secret key $\{k\}$, with the following properties:

- (*Ease of Computation*) Given a value k and an input x, $\mathbb{H}_k(x)$ is easy to compute. The value of $\mathbb{H}_k(x)$ is called the MAC-value of x.
- (*Compression*) For an arbitrary length input x, the MAC function $\mathbb{H}_{k(x)}$ is of fixed length n.
- (*Computation-resistance*): Given a fixed number of input-output pairs $(x_i, \mathbb{H}_k(x_i))$, for $i = 1, \ldots, m$ and any other input $x \notin \{x_1, \ldots, x_m\}$, it is computationally infeasible to compute $\mathbb{H}_k(x)$ without a knowledge of k.

The primary applications of cryptographic hash functions are in digital signatures and data integrity. Because public key cryptography is computationally (in terms of time and space) an intensive process, the messages are normally hashed and the hash-value is signed. The user receiving the message rehashes and checks with the digitally-signed hash-value. Hash function are normally performed only by semi-passive or active devices, whereas MACs can be also be found in passive devices. For example, the TPM makes extensive use of the Secure Hash Algorithm-1 (SHA-l) that provides a unique 20-byte hash.

2.3.2 Symmetric key primitives

Symmetric key primitives include block ciphers and stream ciphers, where the encryption key (e) is equal to the decryption key (d).

Definition 2.3.2. *For encryption and decryption transformations* $\{E_e : e \in \mathcal{K}\}$ *and* $\{D_d : d \in \mathcal{K}\}$ *where* $\mathcal{K}$ *is the key space, then under symmetric key cipher* $e = d$ *or it is computationally 'easy' to obtain* d *from* e*.*

Symmetric block ciphers

In block ciphers the unencrypted or plain-text message ($\mathcal{M}$) is split into blocks of equal length (t) and the encryption algorithm is applied one block at a time to obtain the resulting encrypted or cipher-text message ($\mathcal{C}$).

Definition 2.3.3. *If a message* $\mathcal{M} = m_0, \ldots, m_i, \ldots, m_n$ *is split into* n *parts, where the length of each part* m_i *is* t*, then,*
$E_e(\mathcal{M}) = e(m_0), \ldots, e(m_i), \ldots, e(m_n) = (c_0, \ldots, c_i, \ldots, c_n) = \mathcal{C}$ *and*
$D_d(\mathcal{C}) = d(c_0), \ldots, d(c_i), \ldots, d(c_n) = (m_0, \ldots, m_i, \ldots, m_n) = \mathcal{M}$

Block ciphers are computationally efficient and are used to encrypt large data blocks. The most popular block ciphers are data encryption standard (DES) and advanced encryption standard (AES).

Stream ciphers

In stream ciphers the plain-text message $\mathcal{M}$ is encrypted one unit m_0 (typically bits of data) with a key stream to produced a cipher-text unit c_0. A key stream is a sequence of units $e_0, e_1, e_2, e_3, \ldots \in$ key space $\mathcal{K}$. In essence, stream ciphers are more like block ciphers but with a block length $t = 1$, but unlike a basic block cipher, the transformation of successive digits varies during the encryption. An alternative name is a 'state cipher', because the encryption of each digit is dependent on the current state.

Definition 2.3.4. *Let* $\{e_0, e_1, e_2, e_3, \ldots\}$ *be a key stream from* $\mathcal{K}$*. A stream cipher takes a plain-text string* $\mathcal{M} = \{m_0, m_1, m_2, \ldots\}$ *and produces a cipher-text string* $\mathcal{C} = \{c_0, c_1, c_2, \ldots\}$.

Stream ciphers are designed to be computationally faster than block ciphers and are ideally suited for ubiquitous computing devices. Stream ciphers are found extensively in hardware security modules used in e-tolls, ePassports and EPC identification.

2.3.3 Public key primitives

Public key primitives include public key ciphers and digital signatures. A distinguishing feature of the public key ciphers compared to symmetric ciphers is that the encryption key $\{e\}$ (public key) is not only different from that of the decryption key $\{d\}$ (private key), but it is computationally infeasible to obtain a decryption key from a corresponding encryption key and vice versa. The keys (e, d) is known as a key pair.

Definition 2.3.5. *For an encryption and corresponding decryption transformations* $\{E_e : e \in \mathcal{K}\}$ *and* $\{D_d : d \in \mathcal{K}\}$ *where* $\mathcal{K}$ *is the key space, then under the asymmetric key cipher for each associated encryption/decryption pair* (e, d)*, the key* e *is called the public key and the key* d *is called the secret key and for a given random cipher-text* $c \in \mathcal{C}$*, it is computationally infeasible to find a message* $m \in \mathcal{M}$ *such that* $E_e(m) = c$.

The encryption key (public key) is available in a public domain and the user who wishes to receive a secure message holds the private key (secret key) in a secure environment. Public key cryptography is computationally inefficient in comparison to symmetric key ciphers and, thus, is used to encrypt only small data blocks. A typical use of a public key cipher is in the key establishment protocol, where the sender encrypts a symmetric key cipher using the public key and successive data encrypted using the corresponding symmetric key cipher. The most popular public key cryptosystems are RSA and ElGamal.

Digital signatures

Digital signatures provide authentication, data integrity and non-repudiation services by associating a message with the sender identity.

Definition 2.3.6. *For a message space $\mathcal{M} = \{m_1, m_2, \dots\}$, a user $\mathcal{A}$ key pair (e, d) and a signing transformation $\mathcal{S}_A$, the digital signature $\mathcal{S}_A(m)$ on m is the tranformation of message m using the signing transformation $\mathcal{S}_A$ and the secret key d into a set of signature elements $\mathcal{S} = s_1, s_2, \dots$ of fixed length.*

A digital signature scheme consists of a signature generation and a verification algorithm. The sender applies a signature generation algorithm on the signing message space $\{\mathcal{M}_s\}$ (normally a hash of the message $\mathcal{M}$ that needs to be signed) and his secret key to obtain a signature $\{\mathcal{S}\}$ on the message. The verifier applies a signature verification algorithm on the signed message space $\{\mathcal{M}_s\}$ and public key of the signer typically accessible in the form of a certificate and compares the value with the received signature $\{\mathcal{S}\}$ to prove message authenticity.

The most popular digital signature schemes are RSA, DSA, ElGamal and Schnorr. Digital signatures are essential for developing a cryptographic system because they provide non-repudiation services. But because of their computational requirements, they are found only in semi-passive and active devices, such as smartcards, ePassports and TPMs.

2.3.4 Public Key Infrastructure

Public Key Infrastructure (PKI) helps to bind public keys to entities and enables other entities to verify those bindings. The infrastructure consists of the following components:

- Certification Authority (CA): It is the central component in a PKI, and performs the following functions - issuing certificates, maintaining certificate revocation lists and publishing certificates and revocation lists. The CA issues certificates to PKI users by digitally signing a certificate with its private key and, during verification, a user confirms the authenticity of the certificate by verifying the CA's signature using the CA's public key.

- Registration Authority (RA): It is an entity trusted by the CA, to register or validate the identity of users to a CA, that is, its primary responsibility is to verify whether the certificate contents reflect the information presented by the entity requesting the certificate.

- Repository: A repository is a database of active (valid) digital certificates. The repository provides information, to allows users who receive digitally signed messages to confirm the status of the digital certificates.

- Public Key Certificates: The CA issues a public key certificate for each identity. A digital certificate typically includes the public key, information about the identity of the entity holding the corresponding private key, the validity of the certificate, and the CA's own digital signature.

- Certificate Revocation List (CRL): CAs also issues and processes certificate revocation lists (CRLs), which list revoked certificates. Every PKI user validating a certificate is also required to process the CRL to check if the certificate has been revoked.

- PKI users: PKI users are those who use and rely on PKI components to obtain and verify certificates of other entities with whom they transact.

Using PKI is the dominant method for verifying an entity's pubic key, and thus plays an important role in both semi-passive devices and active hardware-based security devices, for example, ePassports and TPM rely extensively on PKI for the validation of certificates.

2.3.5 Key establishment protocols

A cryptographic protocol is a set of rules that provides a formal definition for the actions required by two or more entities to achieve a specific security goal. A cryptographic protocol is built on various cryptographic primitives, such as, encryption and signature schemes to achieve the required security goal.

The goal of key establishment protocols is to provide a shared secret between two or more parties at the end of a protocol run. Protocols can be further classified as *key transport protocols* where one party creates a secret value and distributes it to others (for example, digital pay TV broadcasting) and *key agreement protocols* where two (or more) parties contribute towards forming the shared secret (for instance, a Diffie-Hellman key agreement).

A key agreement protocols is an authenticated key agreement protocol if it provides key authentication, that is, where the participating entities are assured that the secret is not available to any other third party not involved in the protocol.

2.3.6 Identification protocols

The purpose of identification protocols is for one entity (prover) to prove its identity by interacting with other entity/entities (verifier). Two popular categories of identification protocols are discussed below.

Challenge-response identification protocol (CRIP)

Challenge-response identification protocols involve an entity (prover) $\mathcal{U}$, proving its identity to an verifier $\mathcal{V}$, by demonstrating knowledge of a secret associate with $\mathcal{U}$ without actually revealing the secret to $\mathcal{V}$. Challenge-response identification protocols

can be based on both symmetric key system (Kerberos protocols) or on public key system (X.509).

Zero-knowledge identification protocol (ZKIP)

Goldwasser, Micali, and Rackoff [83] propose the concept of Zero-Knowledge (ZK) proofs. A ZK proof system allows the prover to prove the knowledge of a secret by allowing the verifier to gain confidence on the assertion made by the prover, but without revealing any knowledge about the secret. This technique is highly suited for identification and entity authentication in untrusted environments.

Fiat and Shamir [71] propose an identification scheme based on ZKP. It assumes the verifier is honest and hence is an Honest-Verifier Zero-knowledge proof (HVZKP). The identification protocols is described in Table 2.1.

Prover		**Verifier**
$a \in_R \mathcal{Z}_q$		$c \in_R \mathcal{Z}_q$
	$\xleftarrow{C}$	$C = g^c \bmod p$
$u = g^a \bmod p$	$\xrightarrow{u}$	
	$\xleftarrow{c}$	
$C \stackrel{?}{=} g^c \bmod p$		
$r = a - cx \bmod q$	$\xrightarrow{r}$	
		$u \stackrel{?}{=} y^c g^r \bmod p$

TABLE 2.1: Honest Verifier ZKP

The HVZKP constructs are also useful in creating non-interactive protocols such as signatures protocols and, when used together with secure hash algorithms, the technique is known as the Fiat-Shamir heuristic [71]; its security is provable only under the random oracle model [167].

Because ZK protocols are computational and communicational demanding, they are used only in active devices. The TPMv2 attestation makes extensive use of the ZK protocol to achieve privacy.

2.4 Formal Methods in Cryptographic Protocols

The process of designing a protocol normally consists of three phases, requirement specification, protocol modelling and protocol verification. A formal approach in designing a protocol provides an assurance to the designer that the security goals are met by the protocol. A formal model does not cover all aspects of a cryptographic protocol. Normally, the underlying cryptographic functions are assumed to be secure. A formal approach to designing the protocol does not necessarily mean the protocol is secure against attacks, but only suggest the protocols requirements are satisfied.

Up to now, most of the work in formal methods have been to provide verification of the existing protocols but, more recently, there has been an effort to formalise the construction of protocols. This section presents a overview of the previous work on the use of formal methods for designing protocols. A more thorough survey can be found in [139] [173] and [87].

2.4.1 Requirement specifications

A protocol designer commences the design of cryptographic protocols by specifying a set of requirements the protocol needs to satisfy. The two main approaches have been to develop a set of requirements that can be applied to protocols in general and to develop ways to express the requirements for a number of different types of protocols.

Diffie, van Oorschot, and Wiener present an informal requirement of correctness of an authentication protocol [57], where they say that session keys should remain secret and the protocol runs should match that is, if users $\mathcal{A}$ and $\mathcal{B}$ participate in a protocol run, then the messages sent and received by both $\mathcal{A}$ and $\mathcal{B}$ should be identical. This notion was formalised by Bellare and Rogaway in [20], using a model based on communicating probabilistic Turing machines and by Syverson in his extension of the Abadi-Tuttle logic to include temporal formalisms [184]. Woo and Lam define a semantic model characterising protocol executions, based on two security properties, secrecy, and correspondence [198]. Correspondence referred that certain events can take place only if others have taken place previously. This notion is very similar to the

notion of matching runs, but it is much broader, because the events do not have to be sending and receiving the same message.

The second approach to specify protocol requirements, is presented in the requirements language developed for the NRL Protocol Analyser [185]. The requirements specified in this language have a form similar to the notion of correspondence of Woo and Lam, in that the requirements are given on a sequence of events. The difference being that instead of giving a general requirement for correspondence that applies to all protocols, the user of the language can specify the requirements for the protocols of a particular class. Thus, requirements can vary according to the intended function of the protocol. A similar approach, but with different goals was taken by Yahalom [199], where the goal was to use the formulation of the protocol requirements to achieve a greater performance within the bounds set forth by the requirements. Formalising the requirements has also been applied to signature schemes [164], were the goal was to provide a classification system for various signature schemes and their security requirements.

Formalising the requirements needs to be carried out with the utmost care, because there is a serious danger of overlooking things that may be security-relevant. Most attacks found on the protocols are because of errors that were introduced during the stage of requirement discovery and specification. If appropriately carried out, formalising can be very useful in understanding the problems that the designer of the protocols has to solve.

2.4.2 Protocol modelling

There are two approaches in modelling cryptographic protocols using formal methods. One approach is to develop specific methodologies for the design of the protocols so that they will be more amenable during analysis. Based on this approach, Heintze and Tygar [145] designed a family of tools for reasoning about protocol security and proved a composition theorem. The theorem stated sufficient conditions that needed to be imposed on two secure protocols, so that when combined, they formed a new secure protocol. Later, Gong and Syverson propose a new methodology to facilitate the

design and analysis of secure protocols by restricting the protocol designs to well-defined practices [121]. They introduced the notion of fail-stop protocols that automatically halt in response to any active attack, thus reducing the protocol security analysis to passive attacks only. They also suggested types of protocols that are fail-stop' however, their suggestions might not be practical for some applications.

The second approach is a layered approach [185] in which a relatively abstract model is used at the top layer, and each succeeding layer is proved to be an implementation of the layer above it; until finally, either a detailed specification or the actual protocol code is produced. This approach was also taken by Buttyan *et al.*, where they propose a BAN-like logic, extended with the notion of channels and various access restrictions that uses an abstract model at the top layer [36]. Others, such as, Boyd and Mao proposed a technique to design key exchange protocols that are to be correct [30], in the sense that a specified security requirement will not be violated if the protocol participants act correctly. Meadows developed a model of computation for NRL Protocol Analyser [140], and compared it with Abadi and Tuttle [136] that were developed for BAN Logic.

2.4.3 Protocol verification

The following subsections present the most commonly-followed approaches for the verification of security protocols in hardware security modules:

- Based on Inductive Approach
- Based on State Exploration
- Computational Complexity Proofs.

Based on a Inductive approach

Based on the principle of mathematical induction, the inductive approach to formally model cryptographic protocols was first proposed by Paulson [160]. This approach relies on the concept of trace and list events occurring on a network, while a group of agents are running a protocol. Traces are defined inductively, as is the set of all traces

admissible under a specific protocol. This set, which represents the formal protocol model, is unbounded. Proofs are carried out by induction on a generic trace of the model, establishing the trace properties that represent the goals of the underlying protocol. The interactive theorem prover, Isabelle [159], supports the inductive modelling of the protocol in Higher Order Logic, where nested quantification is permitted over functional symbols, and mechanises the proofs developed by the user.

Though, the inductive approach has been used to successfully model and analyse numerous cryptographic protocols [4], the verification of the hardware-based security protocols using this approach is rather limited, with only a few semi-passive and active protocols being modelled [15, 16, 150]. Some of the reasons include, the limited set of functions available to model certain cryptographic mechanisms, the process of verification tends to be significantly longer and laborious and the lack of any explicit attack even if the proof fails.

Based on State Exploration

Communicating sequential processes (CSP) [92] has had vast applications in the field of formal methods. Ryan [174] and Schneider [177] applied this approach to successfully analyse cryptographic protocols. The users are modelled as processes, and can exchange the messages encompassed by the protocol via specific channels. An idealised specification is defined that accounts for no malicious entity. Then, another specification is obtained by introducing the adversary as a new process that can perform illegal operations. If this specification is equivalent to the idealised one, then the protocol is claimed to suffer no attacks. The two specifications are considered equivalent if all the states that are reachable by the second can be also reached by the first.

Because verification using hand tools can be tedious, Lowe employed a general purpose tool Failure Divergence's Refinement (FDR) checker [134], and later developed Casper [75] to automate the generation of a CSP description of the protocol. However, the processes in FDR are constrained by the intrinsic limitations of model checking: only finite systems of reasonably small size can be tackled and typically account for at most three or four agents, including the adversary. Many attacks have been discovered

by model checking techniques [131] [132] [133]. However, if the system of limited size does not suffer any attacks, it is not obvious that neither does the system of arbitrary size.

Other model checkers that have achieved results in the field are the NRL Protocol Analyzer [140, 185], ASTRAL [54], Murphi [142], SMV [51] and, more recently, AVISPA [5]. Some of these checkers are based on temporal logic rather than process calculus.

Model checking has been very successful in verifying hardware-based security protocols. It has been applied to passive devices [114], semi-passive devices [153, 155, 158] and active devices [120, 137, 144].

Computational Complexity Proofs

The computational view for providing security proofs originates from the works of Blum and Micali [28], Yao [201], and Goldwasser and Micali [82] and is based on the notions of probability and computational power. Based on the reductionist proof presented by Bellare and Rogaway [20], this method has been applied to many identification and authentication systems. Many authors have extended the idea presented in [20] for simple entity authentication protocol to include server-based protocols [22, 22], public-key-based key transport [25], key agreement protocols [26, 31], password-based protocols [17, 19] and more recently, in hardware-based security protocols [12, 34, 77, 109].

In computational complexity, the security of the algorithm or protocol is *reduced* to the security of another problem, that is, the claim being that if there is an efficient algorithm that can break the protocol, then there exits an efficient algorithm to solve the underlying problem, on which the security of the protocol is based. This approach models a powerful adversary who controls all protocol principals and has the ability to initiate protocol-runs between any principals. The security of the scheme is then measured, in terms of adversarial *advantage*, and a scheme is regarded as good if an adversary's maximal advantage is a slow-growing function of the adversary's computational resources.

2.5 Attacks on Hardware-based Devices

For any system implementing hardware-based security modules it is essential that the system protect information residing both within the hardware device as well as in the external components that receive such information (such as, contact-less and contact-based readers/terminals, databases and back-end servers). The following are typical attack scenarios to which hardware-based devices are vulnerable.

2.5.1 Physical Attacks

Tampering: A primary security concern with almost all hardware security modules is physical tampering.

Because these hardware devices store and process confidential and sensitive information, it is important to protect them against any external intrusion capable of either retrieving or modifying the information stored. Typically, security is achieved by restricting information processing only to the embedded software within the devices and explicitly prohibiting any external access. Implementing such a tamper feature is extremely difficult; therefore, hardware-based security devices are vulnerable to a variety of attacks [196]. Such attacks range from simple probing techniques, to machine methods involving material removal, using shaped charge technology , radiation and high voltage imprinting and using imaging technologies such as X-rays and ultrasound.

In recent years new high technology designs [180, 181] have emerged and have been standardised [68]. This has helped designers to determine the hardware's operating requirements and to provide safeguards against various attacks. Most hardware security modules are equipped with one or more of the following tamper security features:

- *Tamper Resistance*: Make it hard to tamper with the hardware-device.
- *Tamper Detection*: Hardware-device has the ability to identify itself when tampering is occurring.

- *Tamper Evidence*: Hardware-device ensures that tampering causes some observable result.
- *Tamper Response*: Hardware-device takes appropriate countermeasures when tampering occurs.

Counterfeiting and Cloning: The integration between the hardware security modules and the sensitive applications such as currency [111, 202], passports [96, 110] or pharmaceutical products [107], has raised new security concerns. Counterfeit devices need to only convince the verifier of their legitimacy, which is accomplished by duplicating only the data needed to convince a verifier. Such data can either be obtained after tampering with the module or by breaking the security of the embedded software.

The best defence against cloning is to use strong encryption mechanisms and enforce access restrictions on the decryption keys. Most hardware-devices support a reasonable level of security and vary based on the type of information protected and the capabilities supported. For example, passive devices in a supply chain have a lower level of security than semi-passive devices such as ePassport or Identity cards.

2.5.2 Communicational Attacks

Eavesdropping: Eavesdropping is one of the most basic attack on a communication protocol and, as a passive form of attack, it requires no interaction with legitimate protocol participants. Eavesdropping attacks are a serious threat, especially to contact-less hardware devices, because sensitive information is transmitted over a wireless channel. Consider passive RFID tags, where the tag responds to the reader by backscattering. During a protocol run the reader transmits tag-specific information to the tag and, because readers transmit at a much higher power than tags, they are subject to eavesdropping at much greater distances.

Nearly all hardware-based security protocols counter eavesdropping by using encryption.

Tracking: Tracking, clandestine or otherwise, breaches the privacy of the user associated with the hardware device. Most contact-less hardware security modules are *promiscuous*, that is, can be read by any compliant reader. Also, in most cases the reader initiates the communication and this can facilitate covert monitoring and tracking by an unauthorised reader. Tracking is also prevalent and a major concern in contact-based hardware systems. For example, in a computer system with TPM, the certified public key pair stored within the TPM chip uniquely identifies the system. Revealing the identity of the machine to every verifier would not only compromise the privacy of the machine but also the privacy of the user(s) using the machine, because each user can be bound to the system. A popular mechanism to protect against tracking is to use pseudonyms [35, 156].

Replay and Relay Attacks: Replay attacks occur when an attacker is able to intercept and retransmit messages in a protocol run. The attacker may either use messages from the current protocol run or from any previous protocol runs. Most common techniques to avoid replay attacks include using nonces, sequence numbers or clock synchronisation [9]. Replay attacks become more problematic when hardware devices are promiscuous, because it make the attack possible even without the knowledge of the user associated with the hardware-device.

Denial of Service: A denial of service is an attack on the availability of the service. Though there is no true solution, it is the responsibility of the protocol designer to incorporate preventive measures against this attack. Some examples of denial of services attacks include using a Faraday Cage to shield RF signals from reaching contact-less devices and the active jamming of RF signals.

2.6 Security Goals for Hardware-based Devices

Hardware devices under each category have some unique requirements that is dictated by the capabilities of the device and application specific attack scenarios. In

this section, we highlight only those security goals that are common to all such implementations, and consider class-specific requirements during our security analysis in subsequent chapters.

2.6.1 Essential Security Goals

Confidentiality: Confidentiality is a fundamental requirement that protects information by restricting access to only authorised users of the system. As communication is carried over insecure networks, the system should provide communication security that secures information exchanged between all participants. Typically, this is achieved through encryption. It is also essential that the system provide strong storage security by encrypting information within the devices and databases, and making it accessible only to legitimate entities.

Integrity: Integrity property provides assurance that message either in storage or during communication was not modified. As with any assurance mechanism, an important aspect is in providing the ability to determine if there was a violation. Usually, this is achieved using hash functions or digital signatures.

Availability: Availability refer to whether the system is available for use when the user intends. Denial of service attacks are usual threats against availability.

2.6.2 Desirable Security Goals

Mutual Authentication: All protocols participants should obtain an unforgeable proof of other participant's identity before engaging in a protocol run with each other. This proof must not be 're-playable' and should be valid only for a single instance of the protocol, that is, it should not be usable as proof in any other protocol run. This is usually achieved using nonce, sequence numbers or time-variant parameters.

Authorisation: Entities have the right to access and change data on the hardware devices; authorisation determines what level of access a certain entity should

have. Because this access may require the knowledge of the cryptographic keys the devices might use, it also makes key management an important issue.

Forward Secrecy: Forward secrecy is concerned with providing long-term security solutions. Security using cryptographic tools relies primarily on the secrecy of the keying material and the forward secrecy ensures that loss of such keys does not compromise any future communication.

3

Hardware-Based Product Identification

Over the past few years, passive devices have had a tremendous growth, both in technological development and in their application. Their small size, cheap manufacturing cost and the integration of contact-less radio frequency technology has further extended its uses into new applications, such as product identification and supply chain management. In this chapter, we consider security issues in identification schemes for such passive devices[1]. Primarily, we focus on the EPCglobal Class 1 Gen2 (EPC Gen2) standard that has become the *de facto* standard for passive RFID tags. We first present a brief security analysis of the current EPC Gen2 standard that has fuelled the need for the new proposals. We then review some alternatives that were proposed to resolve the security issues in EPC Gen2 and propose a new mutual authentication protocol that is compliant with EPC Gen2 standard. Finally, we conclude the chapter by presenting

[1]Commonly referred to as merely *tags* or *labels*

some open research issues.

3.1 EPCglobal Class 1 Generation 2 and its Security

To encourage the adoption of passive RFID technology in addition to supporting interoperability amongst various vendors, EPCglobal [66] together with other standard and regulatory bodies have defined standards for tag and reader interfaces, and communication protocols. EPC Gen2 specifies the operation and functionality of passive RFIDs. The EPCglobal currently classifies RFID tags into four classes:

- *Class 1 (identity tags).* Passive-backscatter tags with the following minimum features:
  - An electronic product code (EPC) identifier
  - A Tag identifier (Tag ID)
  - A function that renders a tag permanently non-responsive
  - Optional decommissioning or re-commissioning of the tag
  - Optional password-protected access control
  - Optional user memory.
- *Class 2 (higher-functionality tags).* Passive tags with the following features in addition to Class-1 tags:
  - An extended Tag ID
  - Extended user memory
  - Authenticated access control.
- *Class 3 (Battery-assisted passive or semi-passive tags).* Passive tags with the following features in addition to Class-2 tags:

  - A power source that may supply power to the tag and/or to its sensors, and/or
  - Sensors with optional data logging.

- *Class 4 (active tags).* Active Tags with the following features:
  - An electronic product code (EPC) identifier
  - An extended tag ID
  - Authenticated access control
  - A power source
  - Communications via an autonomous transmitter
  - Optional user memory
  - Optional sensors with or without data logging.

The predominant class in the product identification and supply chain is the Class 1, where the current standard for tag and reader communication is EPC Class 1 Generation 2 UHF Air Interface Protocol [66].

EPC Gen2 also defines the physical and logical requirements for those RFID systems primarily operating in the 860-960 MHz frequency range. The system is made up of two main components, interrogators, also known as readers, and tags, also known as labels. By modulation of the RF signal (860-960 MHz), a reader transmits information to a tag. Because tags are passive, both information and operating energy are extracted from the signal sent by the reader. A tag can only answer after a message is sent from the reader by backscattering the signal to the reader. Thus, communication between the tag and the reader is half-duplex, which implies that when a reader transmits the tag listens and vice versa.

3.1.1 EPC Gen2 Memory Requirements

The following are the minimal on-chip memory (non-volatile) features for EPC Gen2 (refer Figure 3.1):

- Reserved memory, which contains a 32-bit kill password (KP) to permanently disable the tag and a 32-bit access password (AP).
- EPC memory, which contains parameters for a 16-bit cyclic redundancy code (CRC16), 16-bit protocol control (PC) and a 32-bit electronic product code (EPC) that identifies the object to which the tag is (or will be) attached.
- TID memory, which contains sufficient information to identify to a reader the (custom/optional) features that a tag supports and the tag/vendor specific data.
- User memory allowing for user-specific and user-defined data storage.

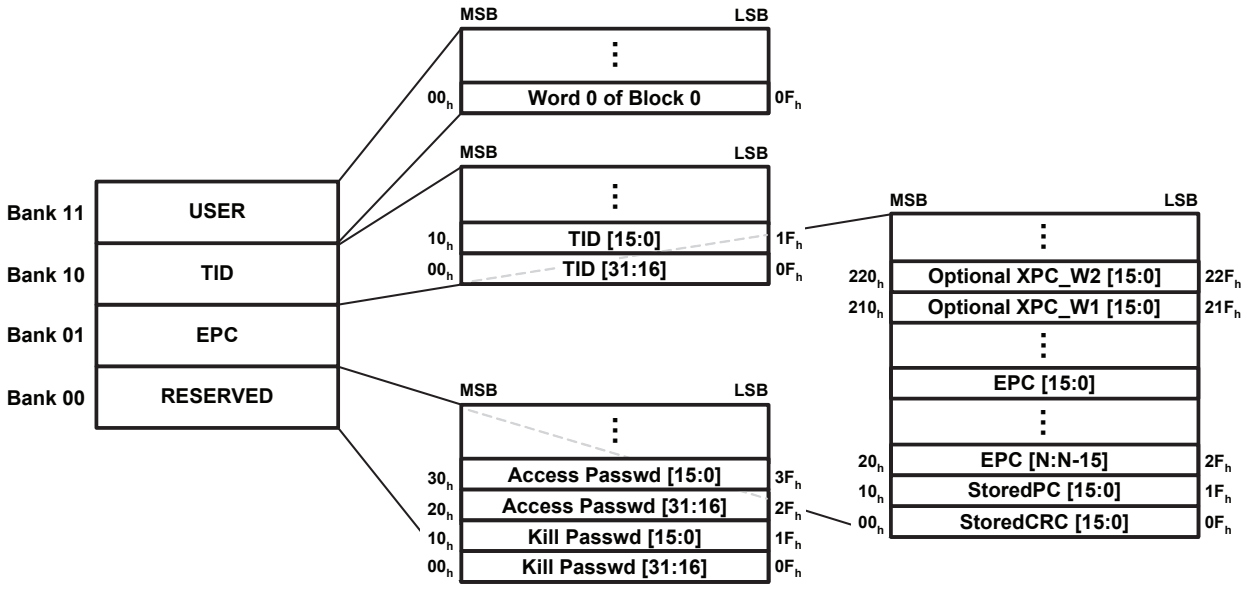

FIGURE 3.1: EPC Gen2 Memory Banks

3.1.2 EPC Gen2 Functions

Two basic mechanisms can be used to provide authentication and confidentiality for the tag-reader communication.

- A 16-bit Pseudo-Random Number Generator (PRNG)
- A 16-bit Cyclic Redundancy Code (CRC).

A pseudo-random number generator is a deterministic function that takes a uniform random bit string as input and outputs a longer bit string that cannot be distinguished from a uniform random string by any computationally-efficient algorithm. According to the EPC Gen2, tags shall have the ability to generate a 16-bit pseudo-random number ($RND16$). Though the EPC Gen2 standard does not specify any particular pseudo-random function, the generator should meet the following randomness criteria:

- *The probability of a single* $RND16$: The probability that any $RND16$ drawn from the RNG has a value $RND16 = j$ for any j , shall be bounded by $0.8/2^{16} < \mathrm{P}(RND16 = j\ ) < 1.25/2^{16}$.

- *The probability of simultaneously identical sequences*: For a tag population of up to 10,000 tags, the probability that any of two or more tags simultaneously generate the same sequence of $RND16$s shall be less than 0.1%, regardless of when the tags are energised.

- *The probability of predicting an* $RND16$: An $RND16$ drawn from a tag's RNG 10ms after the end of T_r , shall not be predictable with a probability greater than 0.025% if the outcomes of prior draws from the RNG, performed under identical conditions, are known.

A cyclic redundancy code is a checksum algorithm that can be used to detect transmission errors (typically one or two bit flips or bursts) in a very efficient way. A CRC can be described by a polynomial over $GF(2)$ (that is, each polynomial coefficient being zero or one) and performing a polynomial division by a generator polynomial $G(x)$, which is commonly called a CRC polynomial. The remainder of the division operation provides an error detection value (also called a syndrome) that is sent to the recipient with a message to provide data integrity checks. Error detection is performed by comparing the received syndrome and the re-computed syndrome of the message. An error is detected if the values are not equal. The EPC Gen2 proposes the use of CRC-CCITT. The hardware requirements of a CRC generator are not very demanding. Specifically, an n-bit CRC consists of an n-bit shift register with some XOR gates.

3.1.3 EPC Gen2 Operations

A reader interacts with a tag using three basic operations, SELECT, INVENTORY and ACCESS.

- SELECT: This involves the reader issuing a Select command for choosing a particular tag from the tag population.
- INVENTORY: This involves the process of identifying tags. A Query command initiates an `inventory` round and decides which tags participate in the round. After receiving the Query command, a tag picks a random value, loads this value into its slot counter and awaits a QueryAdjust or QueryResp command. Assuming that a single tag answers, the query-response algorithm proceeds as follows:
  - $\mathcal{T} \Rightarrow \mathcal{R}$: the tag backscatters a 16-bit random number $RND16$ and enters the `reply` state.
  - $\mathcal{R} \Rightarrow \mathcal{T}$: the reader acknowledges the tag with an ACK command containing the same $RND16$.
  - $\mathcal{T} \Rightarrow \mathcal{R}$: on acknowledgement, the tag transitions into the `acknowledged` state and backscatters its counter value (PC) and its electronic product code (EPC).
  - $\mathcal{R} \Rightarrow \mathcal{T}$: the reader sends a QueryAdjust or QueryRep command, and the identified tag inverts its inventory flag and transitions to the `ready` state.
- ACCESS: The operation is performed when a reader wants to access the tag contents. The access command-set comprises of Req_RN , Read, Write, Kill, Lock, Access, Blockwrite and Blockerase. For every Write, Kill or Access command sent to a tag, a 16-bit word (either data or half of its 32-bit access or 32-bit kill password) is transmitted over the channel. The following sequence of messages are exchanged:
  - $\mathcal{R} \Rightarrow \mathcal{T}$: the reader sends a Req_RN command to the acknowledged tag.
  - $\mathcal{T} \Rightarrow \mathcal{R}$: the tag generates a new $RND16$ and backscatters it to the reader.

– $\mathcal{R} \Rightarrow \mathcal{T}$: the reader computes a 16-bit cipher-text that is composed of the bitwise XOR of the 16-bit word to be transmitted with the new $RND16$.

– $\mathcal{T} \Rightarrow \mathcal{R}$: the tag decrypts the received cipher-text performing a bitwise XOR of it with the original $RND16$.

3.1.4 Security Issues with EPC Gen2

EPC Gen2 combines a 16-bit CRC and a 16-bit PRN to achieve the security goals, such as confidentiality, integrity and authentication. Because the EPCGen2 standard supports only a very basic RNG, any EPC Gen2-compliant protocol is potentially vulnerable. For example, a EPC Gen2-compliant protocol would be vulnerable to cipher-text-only attacks that can exhaustively search to obtain the keying material. Nevertheless, there is a need for implementing security features in such low cost and low performance devices. The security level may not be sufficient for sensitive applications, but considering the device's constraints (such as, life time and application) it is essential to develop a secure EPC Gen2 compliant protocol that would be satisfactory for applications, such as in supply chain management.

Apart from the limited support available for secure cryptographic functions, the EPC Gen2 has protocol-specific security issues. A primary concern is the security of the inventory process, which is extremely weak. Tags transmit *EPC* in plain text that would enable even a passive eavesdropper to obtain information about the tag. This raises some strong privacy concerns, because the protocol does not offer confidentiality or data privacy. A related concern is that the EPC Gen2 does not offer mutual authentication. A tag responds unconditionally to a query. Therefore, a malicious reader can obtain unauthorised information from a tag and moreover, the tag holders might be unaware that they are communicating with a malicious reader. The above-identified security issues show the lack of essential security services that are not guaranteed by the EPC Gen2, even when considering a very weak attack scenario, such as a passive attack.

Another concern is that the *EPC* value is fixed, thus when interrogated, a tag

always transmits the same *EPC* value. This enables a tag to be associated with its holder, allowing its tracking and compromising location privacy. Providing locational privacy is an important requirement, especially for such ubiquitous devices, because they cam be integrated into products that are available from supermarkets to retail stores. When combined with other weaknesses an attacker will be able to not only track an individual, but also obtain information about the product the individual is associated with.

To provide protection against the location privacy related attacks, the EPC Gen2 proposes the use of a kill command. The kill is a command issued when a product is sold to an individual that will essentially, disable all features and render the device inoperable. Though this will help to protect against devices being tracked after point-of-sale, they also diminish their use and induce other inventory and auditing problems. One of the main advantage of passive RFID devices is their ability to store an electronic record about products they are embedded into. If the tag was completely disabled, it would not only be useless for processes such as handling returns and product recalls, but would also make the execution of such processes difficult, because all or most of the essential information about the product would be lost.

The protocol concerning the use of the kill command is also poorly implemented. Because confidential information is sent to both, the kill and the access commands, the reader tries to protect it by means of a *cover-code*. Essentially, the cover-code acts as a password and is obtained by the reader requesting a random number from the tag, and then computing a bit-wise XOR of the cover-code with the 16-bit *RND* obtained from the tag. The final XOR is then sent to the tag where the tag recovers the cover-code by performing a bit-wise XOR with the 16-bit *RND* it previously generated. This can be exploited even by a passive adversary, owing to the asymmetric nature of the RFID communication channel (refer Figure 3.2).

The communication distance between the tags and readers depends primarily on the communication frequency. In most RFID systems the communications range for information sent by the reader (forward channel) is much greater than the information

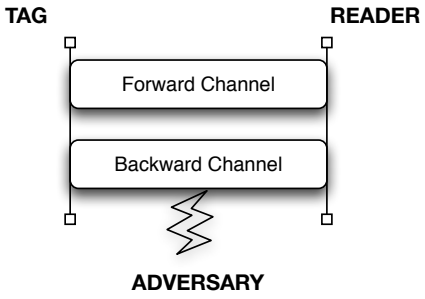

Figure 3.2: RFID Communication Channels

returned from the tag to the reader (backward channel). For example, for an RFID using the 860-960 MHz range, the communication range for the forward channel is up to a hundred metres and for the backward channel is only a few metres. But these limits are based on standards defined by the regulator bodies; however an attacker would be capable of exceeding these ranges by using high-powered readers. The attacker will be able to use a high-powered reader to eavesdrop on the backward channel communication and obtain the random ($RND16$), the tag's response to the reader request to secure the cover-code during the access protocol. The attacker can then easily deduce the cover-code by XORing the message transmitted from the reader and the random acquired.

3.2 Related Work

The research literature for RFID security is extensive and continually growing. In this section, we focus on the RFID security proposals that meet (or were intended to meet) the EPC Gen2 standard and refer the reader to [13] for a more detailed and constantly-updated listing of RFID-based literature.

To resolve privacy related issues in the EPC Gen2, many researchers proposed the use of pseudonyms. A commonly-found solution in the literature consists of repeatedly applying a hash function to a tag's identifier. Since the work of Sarma [176], there have been many proposals based on this idea [91, 162, 200], some based on using hash functions, together with pseudo-random number generators and CRCs [59, 182], and some with public key cryptography [130]. A primary problem with these proposals

is that they do not conform to the EPC Gen2 standard, because the implementation of the hash function in passive RFID chips is still not feasible. Also, pseudonyms guarantees only tag privacy that is, tag identifiers (EPC) are not released into the communication channel. For the requirements of locational privacy, which is needed to prevent an attacker from being able to track a holder's tag, the tag must transmit a fresh pseudonym every time the tag is interrogated.

To solve mutual authentication in the EPC Gen2, Konidala and Kim [116] propose a mutual authentication scheme (TRMA) to protect the tag's access password. However, Lim and Li [128] show that even a passive attacker could recover the password. Lim and Li found that a passive attacker could recover the tag's access password by eavesdropping a single run of the protocol and performing correlation analysis on the captured information. In [117], Konidala *et al.* propose an improved version that uses the tag's access and kill passwords. The authors propose using a *PadGen* function chain of length 2. The outer function is computed over the kill password, while the inner function works over the access password. This newly-proposed *PadGen* function is currently not employed in the EPC Gen2 and also their protocol requires the *PadGen* function to be invoked about four times during every protocol run. A well-known requirement with the RFID is that to keep the delay in communication between a tag and reader minimal. The time-outs for communication messages in EPC Gen2 are very short to achieve a fast read time. A tag performing many operations will lead to a longer delay and, thus, frequently being dropped from the protocol session.

Peris *et al.* propose ultra-light weight mutual-authentication protocols, M^2AP [163] and EMAP [162]. The protocols use pseudonyms to guarantee tag privacy; specifically, an index-pseudonym is used by an authorised reader to retrieve the information associated with a tag. In their protocol a key is divided into several sub-keys that are shared between the legitimate tags and the readers (back-end database). Both readers and tags use these sub-keys to construct the messages exchanged in the mutual authentication phase. In [124, 125], Li *et al.* present a de-synchronisation attack and a full disclosure attack on Peris's proposals. The attack was based on an active adversary obtaining the secret information on the tag using several incomplete protocol runs. In

[49], Chien *et al.* also propose a more efficient full-disclosure attack also using an active adversarial model. However, a serious attack on Peris's schemes was demonstrated by Bárász *et al.* [14] that showed how even a passive attacker could find the tag's identifier and the secrets shared by the reader and the tag after eavesdropping on only on a few consecutive protocol rounds.

In [65], Duc *et al.* propose a tag-to-backend server authentication protocol. During the manufacturing period, the EPC, and the tag's access PIN are assigned by the manufacturer. Then, the manufacturer chooses a random seed and stores it in the tag's memory and the corresponding back-end server's database entry. The security of Duc *et al.* 's scheme greatly depends on key synchronisation between the tags and the back-end server that is achieved by a session-end command sent to both tags and readers. As observed by Chien-Chen in [50], the interception of this command message will cause a synchronisation loss between the tag and the server, thereby causing them not to authenticate each other. This de-synchronisation can be exploited by a counterfeit tag that can replay the data from the previous run to authenticate itself. The protocol also lacks forward secrecy, because an attacker in possession of a compromised tag is able to identify the past communications where the compromised tag was involved. Chien-Chen in [50] also propose an extension to Duc *et al.* 's protocol that resolves its security issues and offers forward secrecy, but they were found to be flawed by Burmester *et al.* [35]

Burmester *et al.* [35] propose a family of trivial RFID authentication protocols (TRAP), where security was proven under the universal composability model [40]. The two protocols of interest are TRAP-2 that provides anonymity and TRAP-3, which provides strong privacy with forward secrecy. The authors also proposed a modification to the EPC Gen2 PRNG, based on the argument that the security of the EPC random generator is weak. Their main concern was the EPC Gen2 probability requirement for the next number prediction. The EPC Gen2 requires that the 16-bit random drawn from a tag's PRNG should not be predictable with a probability better than 0.025%, given the outcomes of all prior draws. For cryptographic security, the next-bit prediction in a random bit generation should be $p = 0.5 + \epsilon$, where ϵ is negligible. The EPC

Gen2 bounds P by 0.025%, therefore, for $(0.5+\epsilon)^{16} < 0.025\%$, we obtain the bound of ϵ as 0.094. The authors suggests that this bound is not sufficiently small, and thus the 0.025% bound should be lowered by at least one order to provide cryptographic security for the tag's 16-bit PRNG. The main drawbacks with the Burmester *et al.* proposal are that only TRAP-2 is EPC Gen2 compliant and that it does not offer either mutual authentication or strong privacy guarantees. Though the author proposed a TRAP-3 that aims to provide mutual authentication, the protocol requires a 32-bit PRNG and 48-bit keys, which are currently beyond the capabilities of low-cost RFID tags and are not supported by the EPC Gen2.

3.3 Mutual Authentication Protocol for EPC Gen2 compliant Tags

A primary goal for our proposed protocol is to provide mutual authentication but still be compliant with the EPC Gen2 RFID requirements. The authentication protocol involves only three moves. It uses a pseudo random number to provide anonymity for the tag, as in [35]. Though we also recommend the modifications to the PRNG proposed in [35], our proposal still uses only a 16-bit PRNG and 16-bit CRC in compliance with the EPC Gen2.

3.3.1 Preliminaries

Assumptions

The protocol works under the following assumptions.

1. The communication channel between the reader and the server (back-end) is secure and authenticated. However, the tag and the reader communicate over an insecure channel and the communication is subject to both passive and active attacks.

2. The server maintains a secure database containing a set of values for each tag

that it manages. We assume adequate security measures are in place to protect both the server and the database.

3. The PRNGs used are cryptographically secure and all entities agree on the functions used in the protocol.

4. Unlike many prior works, we also assume that the reader has limited computational resource. In most real life scenarios, the reader is typically a hand-held device that provides mobility and, thus, is not capable of performing resource-intensive computations (such as database searches).

Restrictions

Computational: the primary restriction in protocol design relates to the number of computations that can be performed on the tag. For the EPC Gen2, we must restrict ourselves to using only ***one*** call to the 16-bit PRF. Though, the EPC Gen2 does not place any restriction on the number of CRCs that can be used, we would like the tag computation to be minimum. In our proposal, the tag employs only ***one*** 16-bit CRC and ***two*** XOR operations, during a protocol run.
Communicational: a tag is in the reader's communication range for only a short duration of time. Thus, there is a communicational restriction on the amount of data exchanged between the tag and the reader. The amount of data transfer should thus be limited to only a few bytes. Our proposal uses three rounds and the total amount of data transferred is only 8-bytes, which is a significant improvement over many previous RFID proposals supporting mutual authentication.

3.3.2 Security Goals

The EPC Gen2 compliant protocol should meet the following security goals:

- Mutual Authentication: a reader should be able to verify and securely identify a tag securely that might be subject to a replay or spoofing attack. Conversely, a tag should be able to confirm that the reader requesting the information is authentic. By assuming that only an authenticated reader can access the back-end

server database (where the communication would normally be achieved using a wired or high-speed medium), server authentication can be excluded from consideration.

- Confidentiality or Data Privacy: an adversary should not be able to obtain any information from the tag-reader communication that would allow the attacker to obtain the tag's identity (EPC) or its passwords.
- Privacy (Location): the communication between a reader and a tag should provide no information that could enable an attacker to trace and/or recognise tags. That is, the messages from a tag should be indistinguishable and non-linkable to any specific tag.
- Forward Privacy: a compromised tag should not reveal any information about previous tag-reader transactions.

3.3.3 Protocol Description

Initialisation

- An initiator (for example, the interrogator of a tag manufacturer) energises the tag $\mathcal{T}$ and assigns a tag identifier (EPC) that will identify the object to which the tag will be attached. The initiator also assigns a value to a pseudonym P. Though the value of P can be the same as the 16-bit EPC, it is beneficial to have them segregated. The value P is constantly updated during every protocol round and thus requires it to be written over the previous value. The segregation helps when storing the EPC in a read-only memory and the pseudonym P in write-access memory. It can be also noted that the initial value of P can be obtained as a function of the EPC, which can provide additional verification on the first run of the protocol.
- The initiator also stores other information such as, the access password (K), the kill password (KP), program counter values, CRC values etc. in the appropriate memory bank as specified in [66].

- The initiator also sends any information pertaining to tag identification to the back-end server $\mathcal{S}$ of the object owner that interconnects the various tag readers $\mathcal{R}$. The server $\mathcal{S}$, stores the information in a database $\mathcal{DB}$, where for each tag an entry is of the form: $\langle EPC, P, K \rangle$.

Authentication Protocol

The proposed authentication protocol is given in Figure 3.3.

- The server $\mathcal{S}$ generates a 16-bit random nonce N and send it to the reader $\mathcal{R}$, which is forwarded to the tag $\mathcal{T}$
- The tag retrieves the 32-bit access password K, its pseudonym P and computes $L_1 = (K_1 \oplus P) \| N$, where $K = K_0 \| K_1$ and K_1 is 16 least significant bits of the access password. The tag then draws two 16-bit numbers M_0 and M_1 from the 32-bit PRF and computes $L_2 = K_0 \oplus M_1$, where K_1 is the 16 most significant bits of the access password. The tag replies the reader with its pseudonym P and M_0
- The reader transmits the tag message (P, M_0) to the server. The server computes $L' = (K_1 \oplus P) \| N$ and draws M'_0 from $\mathrm{PRF}(L'_1)$ for every key $K = K_0 \| K_1$ in $\mathcal{DB}$. If there is a match $M'_0 = M_0$, then the tag is authentic. Else it is rejected. The server then computes $L'_2 = (K_0 \oplus M_1)$ and computes a 16-bit CRC on L'_2. The server sends the reminder α' from the CRC division back to the reader, which in turn transmits it to the tag.
- The tag computes the confirmation value α by computing the CRC on L_2. If there is match on the confirmation value $\alpha' = \alpha$ the reader is authentic. The tag updates its pseudonym with $P = M_1$.

3.4 Security Analysis

In this section, we give a brief security analysis of our proposed scheme. We first provide an overview of the CRC security and then present our claims on how our

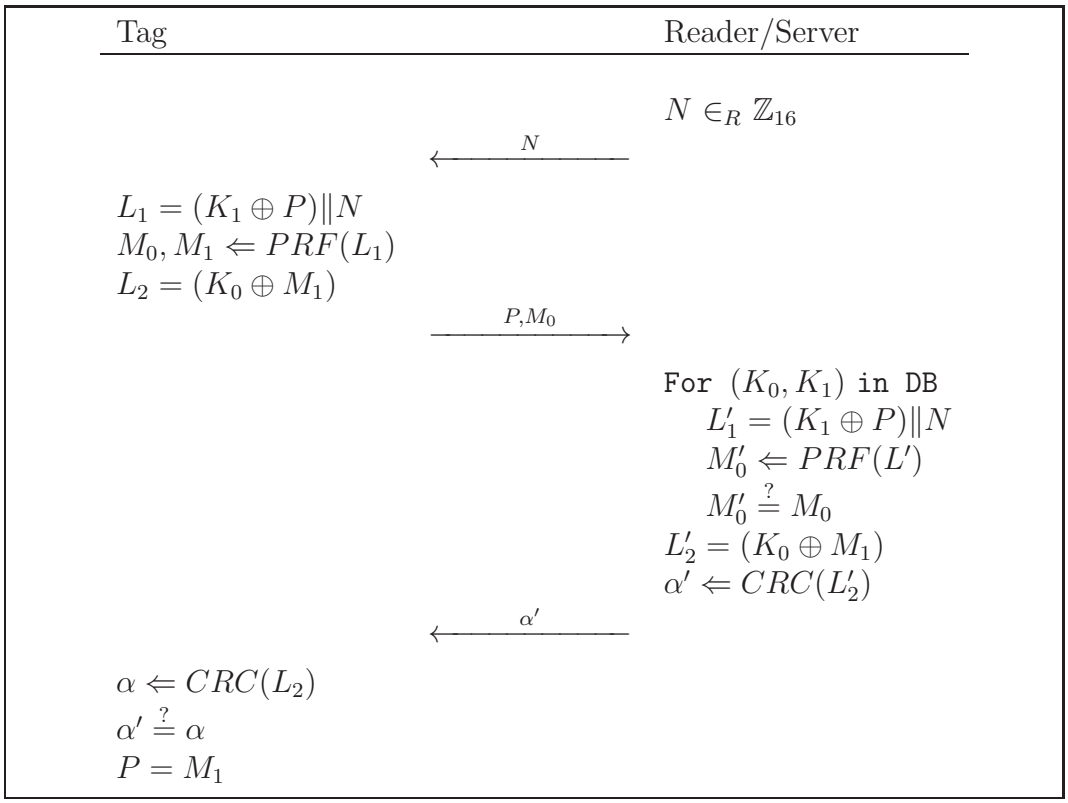

FIGURE 3.3: EPC Gen2 Compliant Mutual Authentication Protocol

proposal satisfies the security requirements that were identified in Section 3.3.2.

CRC SECURITY: normally when a CRC checksum is employed, the message and its corresponding CRC checksum are sent together. But, in our protocol, we do not reveal the message, but rather send only the 16-bit checksum. We are able to employ this treatment, because the message is known to both the sender (reader) and the receiver (tag) and this implementation provides us with some distinct advantages. An adversary trying to convince a tag about the legitimacy of the checksum (α') has either to have knowledge of the input values (K_0 and M_1) or be able to guess the checksum (with a high probability). The input M_1 is pseudo-random, and thus the probability an adversary being able guess the inputs is relatively low. According to the birthday paradox [138], if the key is n-bit long, the chance that there exists a collision on M_1 is roughly $\frac{1}{2^{n/2}}$.

Claim 1 *The proposed protocol provides mutual authentication*: Authentication is

based on the security of the PRF and the key K. To prove mutual authentication we have to show that (a) the adversary cannot identify itself to a reader as a tag and (b) to a tag as a reader.

For case (a), the tag embeds the pseudonym P, the key K_1, together with the random N as inputs to the PRF. A reader can re-compute the output of the PRF (M_0') and verify against the value (M_0) received from a tag. An attacker to impersonate $\mathcal{T}$, needs to compute a valid response pair (P,M_0), that is, the attacker either needs to compute the input values to the PRF or guess the PRF output M_0. To compute PRF input values, the attacker needs to know the output value from the PRF for the last protocol run (M_1 which is the pseudonym P in the current run) and the access password K. Because M_0 is pseudo-random and K is secret, the protocol provides secure tag authentication.

For case (b), to impersonate as a reader the adversary should be able to compute L_2' and the confirmation value α' with a high probability. But as previously identified, the probability an adversary is able guess the values is negligible.

Claim 2 *The protocol guarantees information privacy*: the tag details (EPC, P, K) are stored in the server's database which is assumed to be secure. Thus, only a legitimate server will be able to extract the tag-specific keys and the identifiers based on the information sent by T. Because it is also assumed that the reader and the server communicate via a secure and authenticated channel, the privacy of the tag information is guaranteed.

Claim 3 *The protocol guarantees location privacy*: the tag uses a pseudonym P that is pseudo-random, and is updated after each run (every interrogation of either a legitimate or a malicious reader) of the protocol. The tag responds to a reader's request only by sending P and M_0. An eavesdropper can neither link tag responses to previous responses from the same tag, nor distinguish one tag's response from another's. Therefore, it is difficult to track the location of a tag.

Claim 4 *The protocol is forward secure*: to prove forward secrecy we need to show

that the adversary cannot desynchronize the updating process of the key K of a tag or link a tag whose key was compromised by earlier protocol flows.

For the first condition, even if an adversary either prevents or replaces the confirmation value α' before reaching $\mathcal{T}$, the disruption in the protocol will not result in $\mathcal{T}$ updating its pseudonym P, because $\mathcal{T}$ does not update its pseudonym P until it receives a valid confirmation value α. Even if such attacks are repeated, an adversary will be unable to produce a valid α to be able to convince the tag $\mathcal{T}$. For the second condition, we can observe that, even if the key K is compromised, the values N, M_0, M_1, L_1, L_2 and P cannot be linked, because they are psuedo-random.

Claim 5 *The protocol is secure against replay attacks*: the protocol is a challenge-response authentication scheme that uses pseudo-random numbers to resist replay attacks. The messages N, P, M_0 and α are fresh for every protocol round and cannot be reused in other sessions.

3.5 Summary

The EPC Gen2 standard has led to the wide-spread deployment of passive RFID tags, but it has also raised many security concerns. Designing an EPC Gen2-compliant secure protocol is particularly challenging owing to computational and communicational restrictions. In this chapter, we discussed and identified its weaknesses and showed that it is vulnerable to various attacks. We also showed that many proposals in the literature have failed to develop a secure protocol that is compliant with the EPC Gen2. Recognising this need, we proposed a new mutual authentication protocol and analysed its security. In contrast to many previous protocols, our newly-proposed mutual authentication protocol also considers the reader to be resource-restrictive, which yields us an efficient protocol with little restriction on the type of reader necessary for communication. The security of our proposed protocol can be easily enhanced if we consider adopting the cryptographic hash functions or the MAC instead of the CRC

and the PRNG. However, currently it is infeasible in the EPC Gen2 tags to support these cryptographic functions.

4

Hardware-based Security in Identity Documents

The next two chapters examine the use of hardware security modules in identity documents. The communication mechanism is similar to the RF-based devices used in product identification, where the operating energy is extracted from the signal sent by the reader. Devices implemented in document identification are computationally more capable and are comparable to those of the semi-passive devices. They are capable of performing complex cryptographic functions, such as public-key and symmetric-key encryption, digital signatures and hash functions, and also have the higher storage capabilities to support such functions. The two most dominant applications for such hardware devices are electronic passports (ePassports) and electronic identity cards (eID cards). Though, there are differences in the type of data stored or accessed,

both ePassport and eID cards have a very similar design with regard to implementation and security protocols. Both are based on pre-defined standards, such as the ISO/IEC 14443, ISO/IEC 7816 and ISO/IEC 9796, that determine their data storage, communication protocols, authentication and digital signature mechanisms.

In this chapter, we first present a brief overview of electronic document identification and the factors that have accelerated their uptake. We then provide a technical overview of ePassport implementation and perform a formal security analysis of ePassport implementation using model-checking tools. In this chapter, we focus primarily on the first-generation implementation of ePassports, which is currently the *de facto* implementation for ePassports, whereas the next chapter focuses on the second generation of ePassports that are still being standardised and are expected to replace the first generation ePassports within a few years.

4.1 Electronic Identity Documents

Since the 9/11 attacks in 2001 in the USA, countries have begun evaluating their nation's intelligence and security infrastructure. To strengthen their border security, the United States Federal Government enacted 'The Enhanced Border Security and Visa Entry Reform Act of 2002' that, among other measures also dictated the use of "technology standards and interoperability requirements respecting development and implementation of the integrated entry and exit data system and related tamper-resistant, machine-readable documents containing biometric identifiers" [1]. The legislation called for countries participating in the Visa Waiver Program (VWP) to be compliant with The International Civil Aviation Organisation (ICAO), DOC 9303 standard [96] for electronic and biometric-enabled passports. The VWP enables citizens from about 27 countries to travel to the USA and stay up to 90 days for either tourism or business without obtaining a visa. However, the USA Border Security Act placed a new restriction that required citizens of participating countries to have an ePassport issuing system in place by 26 October 2006, to continue as members of the program.

Currently, over 45 countries have adopted biometric-enabled passports and the wide

deployment of ePassports can be attributed mainly to ICAO efforts. Even before the USA Border Security Act, the ICAO commenced a comprehensive revision of its documents and presented the first version of the ePassport specifications in 2004. The USA VWP considerably accelerated ePassport deployment throughout the world.

The ePassports introduced stronger border security mechanisms and, similarly, the eID cards are being introduced to replace identification cards, such as drivers licences and healthcare cards, to strengthen a country's internal security. The USA introduced the Real ID Act of 2005 [2] that calls for the interoperability of driver license databases among its States, together with machine-readable technology for all driver licenses issued in the USA. Similarly, the UK's National Identity Cards [85], Belgium's eID [11], Germany's Personalausweis [70] and other countries have introduced, or are in the process of deploying, electronic identity cards that can support biometric identification. Similar to ePassports, most eID cards also have an integrated contact-less microchip that stores biometric and identification data and communicates with a reader using a radio frequency.

4.2 ICAO ePassport Specification

The ICAO established five task forces under the New Technology Working Group (NTWG) to develop a standard for Machine Readable Travel Documents (MRTD) [96]. The ICAO standard DOC 9303 [96] for MRTD describes a contact-less smart card microchip that conforms to ISO-14443 [99], embedded within an ePassport booklet. The microchip duplicates the information that is recorded in the Machine Readable Zone (MRZ) of an ePassport and the information that appears on an ePassport bio-data page. The ePassport standard provides details about establishing a secure communication between an ePassport and an Inspection System (IS), the authentication of an ePassport, details on storage mechanisms and biometric identifiers that should be used. The chip also includes an electronic copy of the bearer's photo. The digital photograph of the individual provides a facial biometric that can be used for automated identification processes by employing facial recognition technology. Most implementations of the

ePassports by various countries have a single identifier only, the facial biometric. But the chip has sufficient capacity to include extensions, such as fingerprints and electronic visas if necessary, for future applications.

4.2.1 Operation of ePassport

An ePassport bearer presents the document to a border security officer. The border security officer scans the MRZ information in the ePassport through a MRZ reader and then places the ePassport near a chip reader to fetch data from the chip. The border security officer verifies the content stored in the chip using Passive Authentication (PA) (*cf.* §4.2.4). The ICAO also recommends the use of a Basic Access Control protocol (BAC) (*cf.* §4.2.6), so that the communication between the chip and the reader is via an encrypted communication channel. The channel is then used to verify the integrity of the documents by either Active Authentication (AA) (*cf.* §4.2.5) or passive authentication. Both basic access control and active authentication are optional, whereas, passive authentication is mandatory.

4.2.2 Data Structure

For interoperability, the ICAO ePassport guideline specifies details on how the data should be stored in the microchip. The data elements are grouped together as a Data Group (DG) and collectively stored in a Logical Data Structure (LDS). The ICAO guideline segregates data elements into 19 data groups and the LDS is categorised into three parts:

1. MANDATORY. Data defined by the issuing state or organisation contains the details recorded in the Machine Readable Zone (MRZ) that include passport number, passport bearer's name, nationality, date of birth, date of expiry, encoded facial biometric image and a checksum of the individual data elements that are used to derive the session key.

2. Optional. Data defined by the issuing state or organisation, contains optional biometric data for identification, such as, finger prints, iris scans, displayed identification data such as a digitised signature and any additional personal or document details, such as contact details, proof of citizenship and endorsements.

3. Optional. Data defined by the receiving state or organisation, contain details for automated border clearance, electronic visas and other travel records.

The data groups from one to 16 are defined by the issuing state and are write-protected, whereas the data groups for 17 to 19 will be open for write-access to authorised receiving states or organisations. Write-access is not supported in the first generation, but is available in the second generation of ePassports. The LDS is stored in the microchip using the file system as defined in ISO/IEC 7816-4 [102]. The dedicated file (DF) in the chip file system hierarchy stores the encryption, the MAC keys used in basic access control protocol and the private key of the ePassport bearer that is used in active authentication protocol. The elementary file (EF) in the chip hierarchy will store the security object descriptors (SO_D) and data groups. The SO_D contains the hashes of the LDS data elements digitally signed by the issuing organisation (document signer (DS)) and the corresponding certificate ($\mathtt{CERT}_{CSCA}\langle\mathtt{PK}_{DS}\rangle$). An important security feature is that the data groups are individually hashed and collectively signed by the issuing state and stored in SO_D, thus binding the biometric details with the ePassport bearer details.

The PKI section of the ICAO's ePassport document [96] makes an important distinction between an issuing state and an issuing organisation. The issuing state represents the country of ePassport origin whereas the issuing organisation represents a passport issuing office within a country.

4.2.3 ePassport PKI

Each Country Signing Certification Authority ($CSCA$) is required to forward their self-signed certificate ($\mathtt{CERT}_{CSCA}\langle\mathtt{PK}_{CSCA}\rangle$), document signer certificates ($\mathtt{CERT}_{CSCA}\langle\mathtt{PK}_{DS}\rangle$)

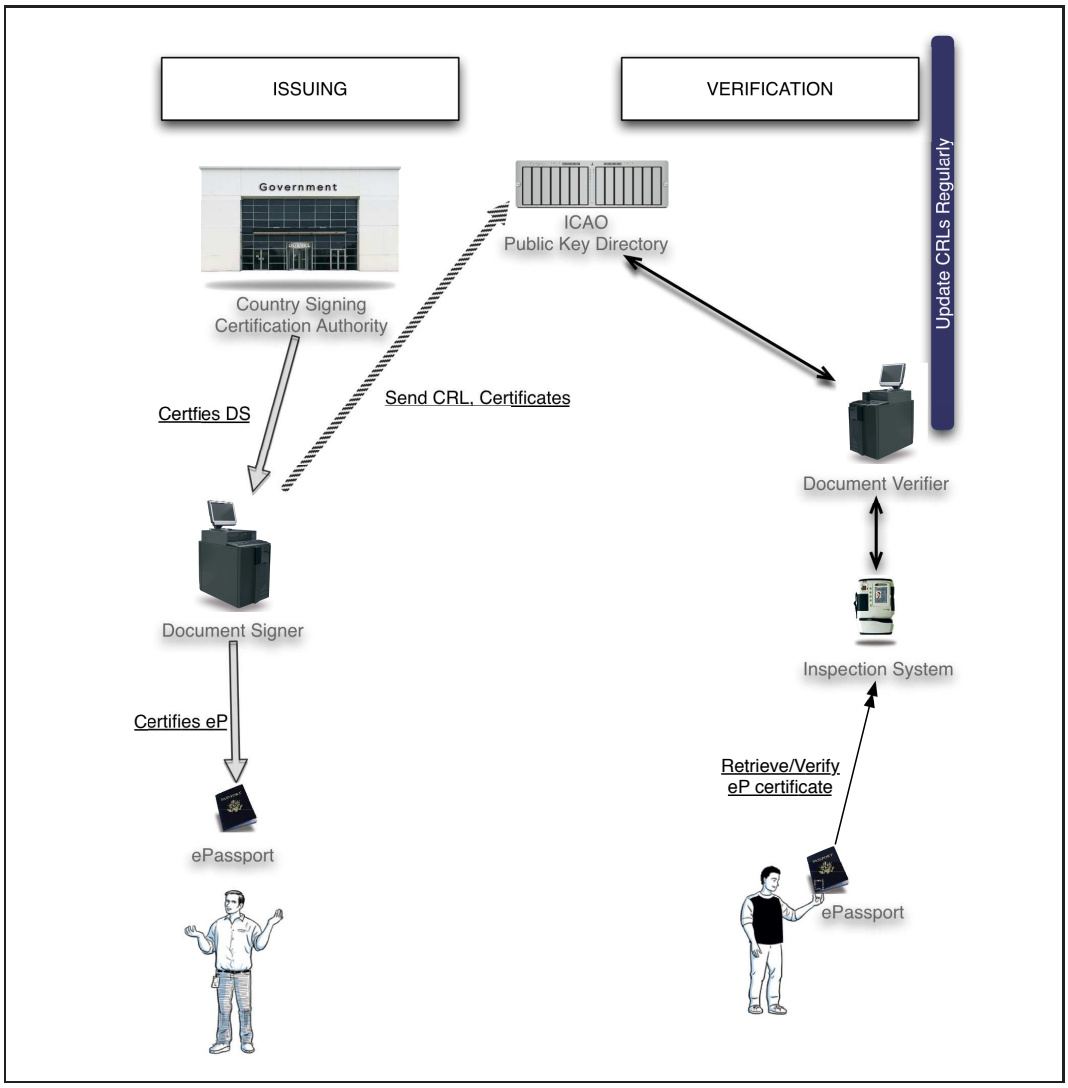

FIGURE 4.1: ICAO Public Key Infrastructure

and certificate revocation lists (CRL) to the ICAO to be published at the ICAO PKI directory (PKD). ICAO also recommends that issuing states replicate the PKD and CRL both locally and bilaterally among participating states every 90 days.

ICAO suggests the $\texttt{CERT}_{CSCA}\langle\texttt{PK}_{DS}\rangle$ be also stored in an ePassport chip, so a border security centre could continue with active authentication in case a PKD was unavailable, but this can compromise security as identified later in §4.4.

4.2.4 Passive Authentication

The mandatory passive authentication mechanism provides only a basic level of security, because the communication is unencrypted. It is used to verify the integrity of the data stored in the ePassport and, the authentication is passive because the ePassport is not involved in any processing during the protocol. The mechanism by itself does not authenticate the ePassport, but only confirms the correctness of the data in the ePassport chip; therefore, it does not detect cloning or skimming attacks.

When an ePassport is presented to an IS, the IS retrieves the certificate of the document signer $\texttt{CERT}_{CSCA}\langle\texttt{PK}_{DS}\rangle$ (either through the ICAO PKD, and if PKD is not available, from the ePassport itself), data elements from the LDS and, their signature SO_D from the ePassport. The IS verifies the signature on the SO_D. It recalculates the hash of the data elements in the LDS and verifies it with the hash in the SO_D. The protocol is deemed successful if both the signature and the hash verification algorithms returns *true*.

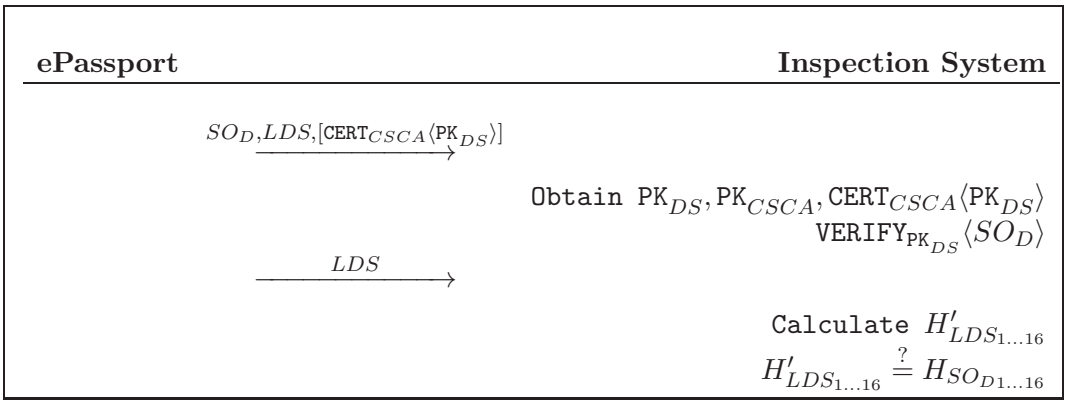

FIGURE 4.2: Passive Authentication

Currently, the USA is the only country that is implementing this level of security, that is, only using the passive authentication protocol for ePassport verification. Owing to the considerable debate and pressure from researchers and privacy advocates, the US Federal Government is considering other optional security measures recommended by the ICAO. This is in line with ePassport implementation from other countries that make use of both basic access control and active authentication to provide better security.

4.2.5 Active Authentication

Active authentication is an optional security feature that relies on public key cryptography to protect against chip modification and chip cloning. The ICAO guideline uses ISO/IEC 7816 *Internal Authenticate mechanism* together with signature computation according to ISO 97986-2 *Digital Signature scheme 1.* Active authentication is a challenge-response mechanism designed to detect if the chip within an ePassport has been substituted or cloned. If active authentication is supported, the ePassport chip stores the public key of the ePassport ($\mathtt{PK}_{eP}$) in DG 15, and its hash representation in the SO_D. The corresponding private key ($\mathtt{SK}_{eP}$) is stored in a secure section of the chip's memory. The protocol is initiated by the IS sending a 8-byte random nonce to the ePassport. On receiving a challenge from the IS, the ePassport digitally signs and returns the result. The IS then verifies the signature using the $\mathtt{PK}_{eP}$ obtained from the SO_D.

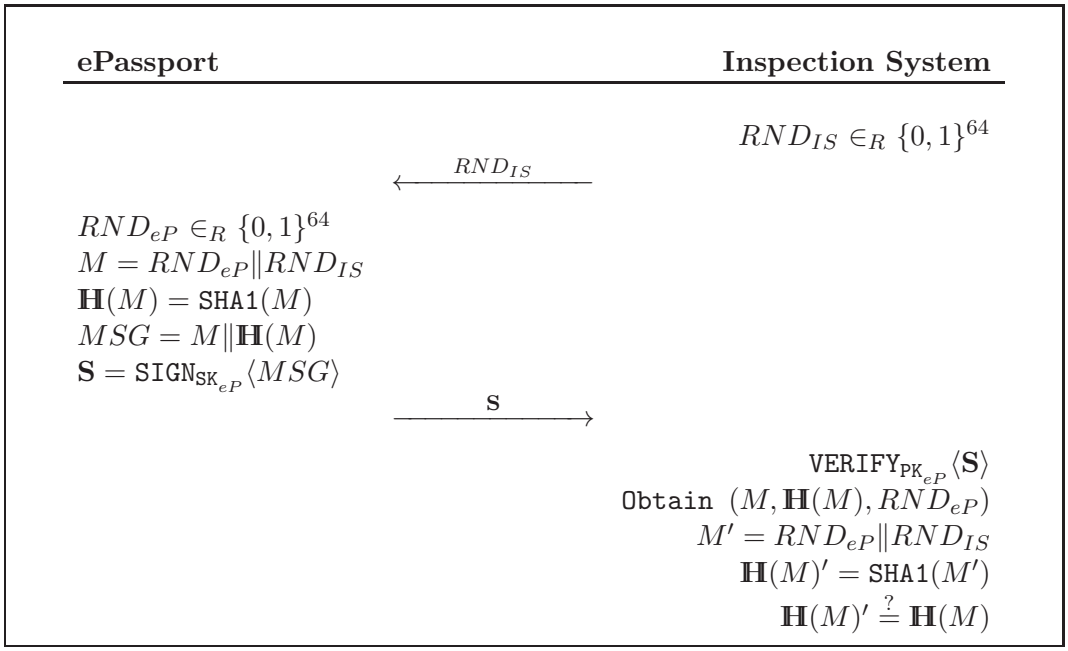

FIGURE 4.3: Active Authentication

4.2.6 Basic Access Control

Basic access control is an optional security mechanism that uses the ISO 11770-2 *Key Establishment Mechanism 6* to form a secure communicational channel between an IS and an ePassport chip. The protocol uses two secret keys (K_{ENC}, K_{MAC}) that are stored in the ePassport chip. The IS derives both these keys using scannable data present in the MRZ, *viz.*, the ePassport number, date of birth of the ePassport bearer, date of ePassport validity, and then check digits for those values.

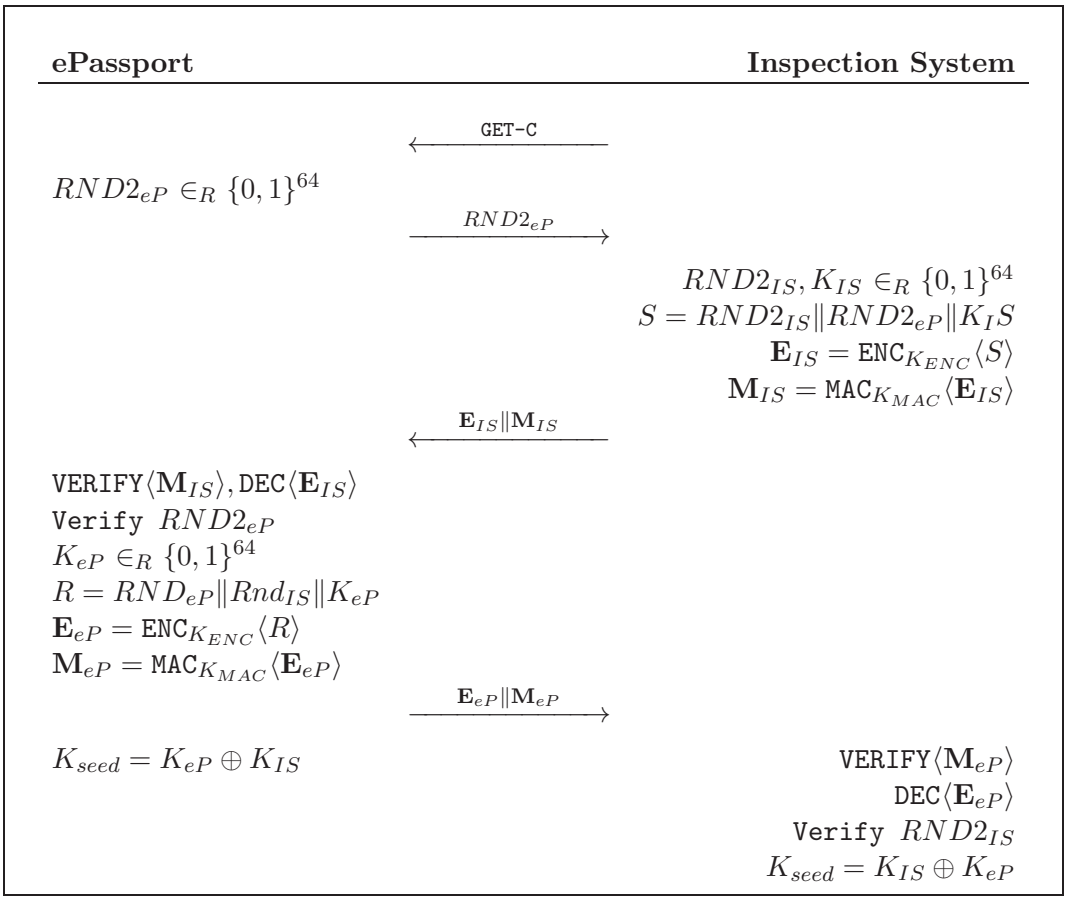

Figure 4.4: Basic Access Control

The three-pass challenge-response protocol is initiated by the IS requesting a challenge from the ePassport chip. On receiving the challenge $(RND2_{eP})$, the IS creates a checksum according to the ISO/IEC 9797-1 *MAC algorithm 3* over the cipher-text that contains the IS's response to the chip's challenge $RND2_{eP}$ and the keying material K_{IS}. The ePassport on obtaining the IS's response creates a checksum that includes

its keying material K_{eP}. Both the IS and ePassport verify the MAC's obtained and decrypt the encrypted message to reveal both keying materials that form the 'key seed' K_{seed}. The K_{seed} is then used to derive a shared session key using the key derivation algorithm described in following section (§4.2.7).

4.2.7 Key Derivation

The value c is a 32-bit counter that permits for deriving multiple keys from a single seed. Depending on the whether a key is used for encryption or for a MAC, a value is assigned to c:

- $c = 1$ (ie., '0x 00 00 00 01') for encryption
- $c = 2$ (ie., '0x 00 00 00 02') for MAC

The following steps are performed to derive both encryption and MAC keys that are to be used in 3DES.

1. $D = K_{seed} \| c$
2. $H_{1...20} = SHA - 1(D)$
3. $k_a = H_{1...8}$ and $k_b = H_{9...16}$
4. Adjust parity bits for k_a and k_b to form correct DES keys.

4.3 Analysis of ePassport

Because passports are used as a primary form of identification and, because of the nature of their stored contents (biometric and personal details) within an ePassport's chip, it is crucial that the document is tamper-resistant and also maintains the secrecy of the data. DOC 9303 [96] provides a brief description of security goals that are achieved and cannot be achieved by the proposed mandatory and optional security mechanisms. If a country implements only the mandatory security requirement (passive authentication), then authenticity and integrity of both the SO_D and LDS are

Method	Security benefits	Vulnerabilities/Weaknesses
Passive Authentication	• Provides authenticity, integrity for SO_D and LDS	• Failure to detect chip substitution. • Failure to prevent against chip copy, unauthorised access and skimming.
Active Authentication	• Prevents against duplication of SO_D and chip modification	• Implementation complexity as extra resources (Memory, CPU) are needed.
Basic Access Control	• Prevents against skimming and eavesdropping	• Failure to detect chip substitution. • Failure to prevent against chip copy. • Implementation complexity as extra resources (Memory, CPU) are needed.

Table 4.1: DOC9303 security benefits and drawbacks

provided. It does not, however, prevent data copy, chip substitution or skimming, and also does not prevent against an unauthorised access to the ePassport. For a greater security the ICAO recommends the implementation of other security mechanisms, such as, (1) active authentication to prevent copying of the SO_D and chip substitution and (2) basic access control to prevent skimming and eavesdropping on communications between the ePassport and the IS. An overview of the DOC 9303's security benefits and drawbacks is given in Table 4.1.

4.3.1 Security Goals

We analyse ePassport protocols by first identifying their security goals that are both explicit and implicit. We assume that a country implements the highest level of security, that is, the country implements all three security measures, passive authentication, active authentication and basic access control.

1. **Data Confidentiality**. Confidentiality is an important requirement for protecting the secrecy and privacy of ePassport details. Confidentiality also provides protection against forgery and spoofing attacks. To provide data confidentiality, the communication channel between the IS and the ePassport should be secure, typically, via encryption. An unauthorised party should not have access to any

data elements within the LDS or keys stored in the DF of the chip file system that contain the session and private keys.

2. **Data Integrity**. A strong integrity mechanism protects against tampering with the chip's contents. The DF, SO_D and LDS should be secure against any unauthorised modifications, that is, any data tampering should be easily detectable by the border security centre.

3. **Data Origin Authentication**. The data on the chip should be bound to information on the MRZ and to the data that appears in the ePassport bio-data page currently being examined by a border security officer.

4. **Non-repudiation**. The ePassport has an advantage, because the ePassport bearer will be physically present at the border security checkpoint. Nevertheless, it would be important to obtain an undeniable digital data from the ePassport for future processing, for example, in case of an aftermath after a terrorist attack to validate the entry of the ePassport bearer at a particular security checkpoint.

5. **Mutual Authentication**. As described in goal 3, it is important for the IS to authenticate the ePassport, but it is also important for the ePassport to authenticate the IS before divulging any personal information to the IS. This is important to prevent an unauthorised ePassport reader from obtaining biometric and personal details from an ePassport.

6. **Certificate Manipulation**: The IS should have a guarantee that certificates presented by the ePassport are valid and match the data on the ePassport. ICAO has implemented a PKI [189] that would store signature certificates from issuing states and organisations.

7. **Key Freshness and Key Integrity**. The IS and the ePassport must have satisfactory proof that a nonce generated during both active authentication and basic access control protocols is fresh and the integrity of the derived session key is preserved. Both parties should also have undeniable proof that the other party

is in possession of a valid session key. Any previous compromised key should be easily detected and the protocol run should terminate.

8. **Forward Secrecy**. The loss of session key or key used to generate a session key (K_{ENC} and K_{MAC}) should not compromise any future communications.

4.3.2 Related Work

Juels *et al.* [110] present some security and privacy issues that apply to ePassports. The contact-less chip embedded in an ePassport allows the ePassport contents to be read without direct contact with the chip reader and, importantly, even with the ePassport booklet closed. The authors raised concerns on whether the data on the chip could therefore be covertly collected by means of *skimming* or *eavesdropping*, when the encryption key used in basic access control protocol gets compromised owing to the low entropy of the key (the length of key being only 56 bits). Gaurav Kc and Paul Karger [113] suggest that an ePassport may also be vulnerable to a *splicing attack*, *fake finger attack*, and attacks that can be carried out when an passport bearer presents the passport to hotel clerks. Hoepman *et al.* [93] discusses unauthorised access, *traceability* and *skimming* attacks on the basic access control in the context of a Dutch passport. Monnerat *et al.* [143] focus on active authentication and discuss some of the privacy issues, and, as a countermeasure, propose a GQ-based authentication protocol.

There have also been practical attacks owing to weaknesses in the basic access control protocols: these were demonstrated by Witteman [194] on Dutch passports and Grunwald [88] on German passports. Recently, Laurie [122] demonstrated a practical attack on UK ePassports, where he was able to successfully clone an ePassport's chip while still hidden in an envelope. The *cloning* attack does not compromise border security, because to do that, the attacker should be able to modify the details and still maintain the integrity of the data and its corresponding hashes. However, the cloning of an ePassport is a major privacy issue because the attacker would not only be able to obtain the passport bearer's details but also his biometric details stored in the ePassport. The risk of eavesdropping is increased by the surveillance environment in

which the border checks occur, particularly when border control processes become more automated, as in the Australian *SmartGate* system [10], eavesdropping will ultimately assist in the covert collection of ePassport data.

4.3.3 Formal Verification

Previous research works have identified many weaknesses in ePassports. We continue this line of investigation by using a formal analysis of the ICAO ePassport standard, using the model checking method (*cf.* §2.4.3).

The model checking approach has been very successful in finding faults in many protocols [54, 131, 133, 142, 145, 158, 178]. The approach is based on modelling a protocol as a finite state machine by specifying its properties and then using a model checker to verify the systems properties. The advantages of using model checkers is that the verification process is usually automated and if a verification fails on a required property, the model checker lists the sequence of events that led to the property being broken. An apparent limitation of this approach is that the verification of a complex protocol suite can lead to an exponential growth of the state space making the verification impossible. Thus, a formal model does not cover all aspects of a protocol. Normally, the underlying functions are assumed to be true. The verification of the simplified protocol that was formalised does not necessarily mean the full version of the protocol is secure against attacks, but only suggests the well-defined protocol's requirements are satisfied. Nevertheless, it does provide an assurance to users and designers about the relevant security goals that are met by the protocol.

To analyse ePassport protocols, we make use of Casper [75], a tool developed by Gavin Lowe. Casper is a compiler that converts a high level specification of the protocol to a CSP [92] script. The CSP script can then be run on a model checker, such as the FDR2 [134], to verify whether the protocol meets its security requirements. An overview of the verification process using CASPER/CSP/FDR2 is provided in Appendix A

4.3.4 Modelling ePassport protocols

The ICAO ePassport is a complex protocol suite that consists of three protocols, basic access control, passive authentication and active authentication. Such a complex protocol suite is not only difficult to formalise, but also the verification of such systems more often leads to an exponential explosion of the state space. Therefore, we do not find many publicised works on the verification of such systems and, to the best of our knowledge, ePassport protocols have not been formally verified.

We model the flow of the ePassport protocol as follows:

1. When an ePassport is presented at a border security checkpoint, the ePassport and the IS execute the basic access control protocol to establish a session key to secure all future communications.

2. On the successful completion of the basic access control protocol, the IS performs a passive authentication.

3. On the successful completion of the passive authentication protocol, the ePassport and the IS execute the active authentication protocol.

The ePassport authentication mechanism relies heavily on the PKI. We model only one level of certification hierarchy up to the document signer and we assume that the document signer public key certified by its root (country signing authority) is valid and secure. This does not weaken the verification process of the ePassport protocol suite, but only indicates that the model does not consider any weaknesses that might exist in the PKI implementation by the countries and the ICAO. We also assume that cryptographic primitives used in the system such as the hash functions, MAC, and the generation of keys (3-DES) are secure against various forms of attacks that exist in the literature. Our modelling of the ePassport protocols using Casper is presented in Appendix B.

4.4 Verification Using Casper/FDR

In ePassports data, confidentiality is provided by the basic access control protocol, whereas the integrity of the ePassport chip contents, the LDS and the SO_D is verified by the IS using the passive authentication and active authentication protocols. The keys K_{ENC} and K_{MAC} are stored in the DF on the ePassport and the keys are generated by an IS before initiating communication using the data in the MRZ that includes the ePassport number, date of birth, ePassport validity date, and corresponding check digits. The ICAO ePassport guideline states that the entropy of the key is, at most, 56 bits. The Juels *et al.* [110] analysis of USA passports reports that the entropy of the key can further be reduced to ≈ 52 bits because of the USA ePassport numbering scheme, because the first two digits are assigned to 15 ePassport issuing offices. Because of low entropy, the key would be vulnerable to brute force attacks as demonstrated by [122].

The analysis of the ePassport protocol using the Casper and FDR2 verification software proves that the protocol is vulnerable to the Grand-master Chess Attack [55]. Compiling with security specifications creates corresponding refinement assertions.

The secrecy specification results in an assertion
`SECRET_M::SECRET_SPEC [T= SECRET_M::SYSTEM_S` and its verification using the FDR2 results in an erroneous trace after 30 states with 135 transitions and the FDR2 debugger reveals:

```
send.Reader.Chip(Msg1.GETC,<>)
INTRUDER_M::say.GETC
send.Chip.Reader.(MSG2,RNDC2,<>)
INTRUDER_M::say.RNDC2
send.Reader.Chip(Msg3,Sq.<
        Encrypt.(KEYE,<RNDR2,RNDC2,KR>),
        Encrypt(KeyM,<RNDR2,RNDC2,KR>)>)
INTRUDER_M::say.Sq<
        Encrypt.(KEYE,<RNDR2,RNDC2,KR>),
        Encrypt(KEYM,<RNDR2,RNDC2,KR>)>
```

which can be intepreted as:

```
1. Reader -> I_Chip : GETC
1a. I_Chip -> Chip : GETC
2. Chip -> I_Chip : {RNDC2}
2a. I_Chip -> Reader : {RNDC2}
3. Reader -> I_Chip :
        {RNDR2, RNDC2, KR}{KEYE},
        {RNDR2, RNDC2, KR}{KEYM}
3a. I_Chip -> Chip :
        {RNDR2, RNDC2, KR}{KEYE},
        {RNDR2, RNDC2, KR}{KEYM}
4. Chip -> I_Chip :
        {RNDR2, RNDC2, KC}{KEYE},
        {RNDR2, RNDC2, KC}{KEYM}
4. I_Chip -> Reader :
        {RNDR2, RNDC2, KC}{KEYE},
        {RNDR2, RNDC2, KC}{KEYM}
```

and for assertion `AUTH1_M::AuthenticateRESPONDERTo INITIATORAliveness [T=` which corresponds to the security goal that an ePassport believes that it is involved in a conversation with the IS. Its verifications using the FDR2 results in an erroneous trace after 12 states with 35 transitions and the FDR2 debugger reveals:

```
send.Reader.Chip.(Msg1,GETC,<>)
INTRUDER_M::hear.GETC
send.Reader.Chip.(Msg3,Sq.<
        Encrypt.(KEYE,<RNDR2,KM,KR>),
        Encrypt.(KEYM,<RNDR2,KM,KR>)>,<>)
INTRUDER_M::hear.Sq.<
        Encrypt.(KEYE,<RNDR2,KM,KR>),
```

```
        Encrypt.(KEYM,<RNDR2,KM,KR>)>
INTRUDER_M::say.Sq.<
        Encrypt.(KEYE,<RNDR2,KM,KR>),
        Encrypt.(KEYM,<RNDR2,KM,KR>)>
```

which can be interpreted as:

```
1. Reader -> I_Chip : GETC
2. I_Chip -> Reader : KM
3. Reader -> I_Chip :
        {RNDR2, KM, KR}{KEYE},
        {RNDR2, KM, KR}{KEYM}
4. I_Chip -> Reader :
        {RNDR2, KM, KR}{KEYE},
        {RNDR2, KM, KR}{KEYM}
```

The trace from the security assertion can be interpreted that an intruder, during the communication with an IS, is replaying messages from the chip, that is, the IS establishes a session key even though it is not sure if a chip is genuine.

Can this weakness be exploited? Once a secure communication is established between an IS and an ePassport, the IS retrieves data stored within the LDS and performs an integrity verification using the issuing state's certificate. A border security officer, on receiving evidence that the LDS has not been tampered with, would authenticate an ePassport bearer by using the facial biometric image stored in the LDS against the person physically present at the checkpoint. Therefore, even if the messages are only being replayed the data still has to be from an issuing state certified ePassport chip.

This weakness can be exploited because facial biometrics are view-dependent, and are prone to inter-class similarities within large population groups, such as identical twins, similar ethnic groups and are certainly possible in cases of human cloning. Because the probability of uniqueness using facial biometric is low, it is certainly possible that a border security officer might not be able to differentiate between the facial biometric data in the LDS and the person physically present at the checkpoint. Philips *et*

al. [165] also point out that the false rejection rate could be as high as 43%, because the majority of the algorithms used in facial biometrics are subject to illumination issues and also depend on the type of camera used to obtain the initial image. Note that ePassports store high-resolution images of the ePassport bearer to make verification independent of the processing algorithms used by various countries. This introduces another serious security weakness, an attacker can manipulate less significant bits of images to find collisions for the hash functions used.

Even with these drawbacks, basic access control is important and should be implemented because it prevents eavesdropping. The protocol is vulnerable to replay attacks, but an intruder cannot decrypt the values ($\mathbf{E}_{IS}$ or $\mathbf{E}_{eP}$) used to form the session key (K_{seed}).

The active authentication protocol, in addition to providing integrity, also protects the ePassport against chip modification, that is, it binds the LDS with the ePassport bearer's secret key $\mathtt{SK}_{eP}$ and authenticates the ePassport chip. Our verification of an ideal active authentication protocol, that is, assuming that the basic access control protocol was carried out in a secure way, indicates that there is no security weakness in the protocol.

Assertions

```
SECRET_M::SECRET_SPEC [T= SECRET_M::SYSTEM_S
AUTH1_M::AuthenticateRESPONDERToINITIATOR
        Aliveness [T= AUTH1_M::SYSTEM_1
AUTH2_M::AuthenticateINITIATORToRESPONDER
        Aliveness [T= AUTH2_M::SYSTEM_2
AUTH3_M::AuthenticateINITIATORToRESPONDER
        Agreement_rndr1 [T= AUTH3_M::SYSTEM_3
AUTH4_M::AuthenticateRESPONDERToINITIATOR
        Agreement_rndc1 [T= AUTH4_M::SYSTEM_4
```

which corresponds to secrecy, the authentication of an ePassport to an IS, and from an IS to an ePassport does not yield any erroneous traces. However, if we consider

that an intruder was able to successfully run the basic access control protocol with the IS by obtaining K_{ENC} and K_{MAC} through a brute force attack as in [122] and thus successfully computing the session key K_{seed}, then the assertions:

```
SECRET_M::SECRET_SPEC [T= SECRET_M::SYSTEM_S
AUTH2_M::AuthenticateINITIATORToRESPONDER
        Aliveness [T= AUTH2_M::SYSTEM_2
AUTH3_M::AuthenticateINITIATORToRESPONDER
        Agreement_rndr1 [T= AUTH3_M::SYSTEM_3
```

yields erroneous traces that indicate that weakness exists in the protocol.

Assertion `SECRET_M::SECRET_SPEC [T= SECRET_M::SYSTEM_S` yields an error trace after four states and eight transitions and analysis using the FDR2 debugger revels the following first-level trace.

```
send.Reader.Chip.(Msg1,Encrypt.
        (KEYCR,<RNDR1>),<RNDR1>)
leak.RNDR1
```

This attack is obviously true, because the intruder is now in possession of the session key and therefore able to decrypt any communication between the ePassport and the IS. This would compromise the privacy of an ePassport bearer because of their personal details and increase the risk of identity fraud.

Assertion `AUTH3_M::AuthenticateINITIATORTo RESPONDERAgreement_rndr1[T=AUTH3_M::SYSTEM_3` yields an erroneous trace after eight states and 149 transitions. The FDR2 debugger revels the following second-level trace

```
env.Chip.(Env0,Reader,<RNDC1,Reader>)
receive.Reader.Chip.(Msg1,
        Encrypt.(KEYCR,<RNDM1>),<RNDM1>)
signal.Commit3.
        RESPONDER_role.Chip.Reader.RNDM1
```

From the above traces, we can interpret that an attacker is able to successfully authenticate to the IS as a genuine ePassport. This is possible, because the session key is compromised. This attack is theoretically possible, but in practice would not be easy to implement, because the data is protected by a digital signature and it is computationally in-feasible to generate a valid signature for a modified data. Nevertheless, this weakness can be exploited by the attacker in lieu with the weakness in the facial biometric systems, as discussed above. The combination of the weakness in both the basic access control and the active authentication can be exploited by the intruder. An attacker can now make a copy of the ePassport and authenticate successfully, thus defeating the primary security goals of the basic access control and active authentication to prevent against chip substitution and chip copy.

Assertion `AUTH2_M::AuthenticateINITIATORTo RESPONDERAliveness[T=AUTH2_M::SYSTEM_2` yields an error trace after three state and six transitions with the FDR2 debugger revealing the following second-level trace

```
env.Chip.(Env0,Reader,<RNDC1,Reader>)
receive.Reader.Chip.(Msg1,Encrypt.
        (KEYCR,<RNDM1>),<RNDM1>)
signal.Commit2.RESPONDER_role.Chip.Reader
```

The above trace points to an important security goal that is not met: the mutual authentication between an ePassport and an IS. The IS believes that it has successfully authenticated the ePassport but there is no proof that the ePassport has successfully authenticated the IS. The authentication of the IS by the ePassport depends on the fact that only a genuine IS would be able to obtain K_{ENC} and K_{MAC} from the MRZ to perform the basic access control protocol and compute the session key K_{seed} used in the active authentication protocol. We have seen that this is not necessarily true. An attacker in possession of the keys K_{ENC} and K_{MAC} (because of low entropy and brute force attacks as in [122]) will be able to masquerade as an IS and successfully authenticate to an ePassport.

From the above traces it is also clear that the ePassport protocol does not satisfy any

key-related security goals such as freshness and integrity. Key integrity is not satisfied, because an attacker is able to successfully run the basic access control protocol and obtain the session key K_{seed} used to form a secure communication channel. There are no guarantees provided to either the ePassport or the IS regarding key freshness. The nonce generated by either the IS, the ePassport or both may not contain the sufficient randomness that is necessary for a security protocol. An eavesdropper might be able to collect information about several runs of the protocol and perform a cipher-text with a known partial plain-text attack to obtain the session key and/or the MRZ information that is necessary to create K_{ENC} and K_{MAC}. This would also compromise the security goal of forward secrecy. An ePassport has an average validity of about 10 years. Any loss of K_{ENC} or K_{MAC} keys makes the ePassport vulnerable to skimming and snooping attacks.

Though we were unable to make a formal analysis of the security goals non-repudiation and certificate manipulation, an informal analysis of the ePassport protocols reveals they are also prone to infrastructure-based attacks. Public key certificates (for both the document signer and the country signing certificates) are held by the ICAO in a central repository. The ICAO's ePassport guideline states that each border security checkpoint should update their certification hierarchy list individually. This is necessary to perform a valid verification during the active authentication protocol, because the secret key of an ePassport is certified by the issuing country. The drawback is that an attacker may be able to mount a DOS attack on the border security checkpoint certificate server before arriving or, in co-ordination with others, to prevent the certificate server from updating and thus prevent the border security checkpoint from verifying the validity of the signatures in the ePassport. Because the border security checkpoint now relies on the $\texttt{CERT}_{CSCA}\langle DS\rangle$ stored within the ePassport and does not have access to an updated revocation list, the attacker will be able to validate a non-valid signature in the ePassport. The ICAO ePassport guideline acknowledges this issue and states that in such a case a border security checking officer should rely on the conventional methods that were in place before ePassport for verification of the ePassport bearer existed. However, this defeats the entire purpose of introducing ePassports.

4.5 Summary

In this chapter, we presented an overview of the current ePassport implementation and provided a formal security analysis. We used the Casper/CSP/FDR model checker to verify our security goals. Our analysis has shown that the current security measures that are in place are weak and do not meet our security goals.

- The ePassport protocols do not satisfy our goal for data origin authentication, because it can be subject to replay and grand-master chess attacks, and the weakness can be exploited in cases where problems with facial biometrics exists.
- Data confidentiality is also compromised when an attacker is able to obtain the encryption and MAC keys stored in the ePassport chip using information presented in the MRZ.
- We were able to prove that this further affects the security goals for active authentication protocol, mutual authentication, key freshness and key integrity.
- An informal analysis of the ePassport system reveals that it may also be vulnerable to certificate manipulation, because it is dependent on the PKI, which is prone to DOS attacks.

5

Securing ePassports

As identified in the previous chapter, the ICAO ePassport proposal has numerous security weaknesses. To resolve those concerns, the ICAO instructed the New Technology Working Group (NTWG) to develop a set of new protocols for ePassports. The NTWG proposed a new standard called the Extended Access Control (EAC), based on the European Union's (EU) ePassport proposal [94]. A primary goal of the EAC is to provide mutual authentication (in particular, authentication of the IS) and additional security for biometrics. Current implementations of the first-generation ePassports have a single biometric identifier based on the facial biometric, whereas the second-generation will include both fingerprints and iris scan biometric identifiers.

In this chapter, we start by providing an overview of the EAC-enabled ePassports. We then perform a security analysis and identify weaknesses in the newly proposed protocols. Because the EAC proposal fails to provide adequate security and introduces

new security weaknesses and implementation issues of its own, we propose an alternative to the EAC that covers the entire ePassport protocol suite. Our proposed solution fixes the drawbacks in the current EAC proposal and provides the following enhanced security features (1) prevention of biometric information being released to a malicious IS in possession of the MRZ details, (2) enhanced communication security between an ePassport and an IS, (3) protection against passport skimming and, (4) reduced PKI interactions.

5.1 Extended Access Control

To resolve the security and privacy concerns that have been identified in the first-generation ePassports, the EU issued an ePassport specification [94] to restrict access to secondary biometric identifiers such as fingerprints and iris scans. Countries with bilateral agreements will be able to perform EAC-based authentication and obtain access to fingerprint and iris images. The EAC ePassport specification is based on the authentication techniques proposed by D. Klüger from German Federal Office for Information Security (BSI) [118, 119], and now includes two new authentication protocols, Chip Authentication (CA) and Terminal Authentication (TA). The EAC also included modifications to the existing PKI. The Country Signing Certification Authority (CSCA) is now required to certify Document Verifiers (DV) in other countries that in turn certifies Inspection Systems (IS) present at a country's border security checkpoint. Figure 5.1 provides an overview of the modified PKI hierarchy.

5.1.1 ePassport Operation with the EAC

The EAC involves the following four steps:

1. The ePassport bearers presents their document to a border security officer who scans the MRZ on the ePassport through a MRZ reader and then places the ePassport near an IS to fetch data from the ePassport chip. The ePassport and

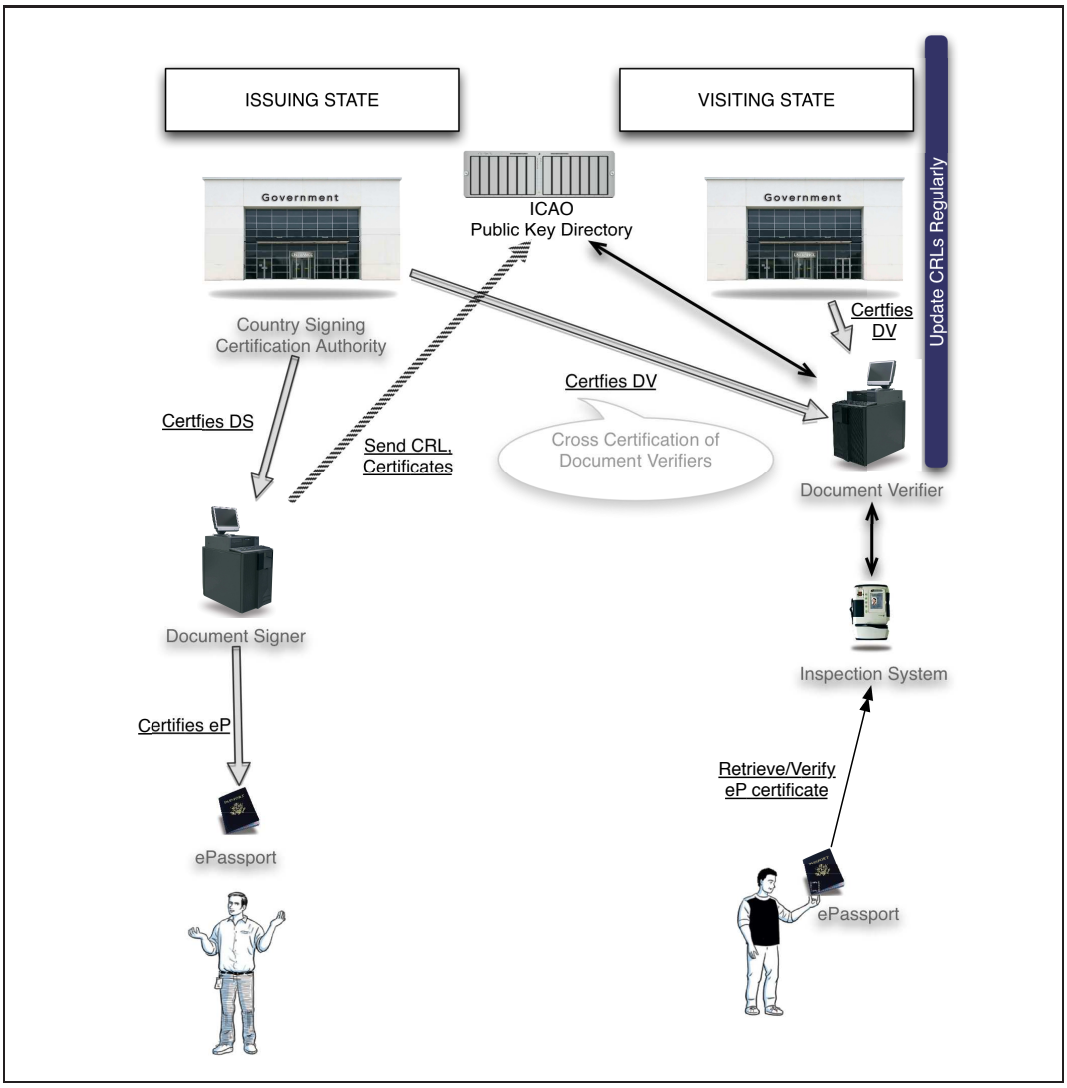

Figure 5.1: EAC Public Key Infrastructure

the IS establish an encrypted communication channel by executing the Basic Access Control (BAC) protocol.

2. The IS and the ePassport then perform a mandatory chip authentication (CA).

3. The chip authentication is followed by passive authentication (as in the first-generation ePassport).

4. Terminal authentication (TA).

Only when all the protocols are completed successfully, does the ePassport release sensitive information such as secondary biometric identifiers. If an IS does not support

the EAC, the ePassport performs the collection of protocols as specified in the first-generation ePassports.

5.1.2 Chip Authentication

Chip Authentication is a mandatory EAC mechanism that replaces the active authentication proposed in the first-generation ePassports. It involves a Diffie-Hellman key agreement (DHKA) [58] and is followed by passive authentication. It is performed after a successful basic access control and provides both a means of authenticating the ePassport and generating a new session key. The ePassport uses a static Diffie-Hellman public key while the IS uses an ephemeral key. The protocol begins with the ePassport sending its public key ($\mathtt{PK}_{eP}$) and its domain parameters ($\mathtt{D}_{eP}$) to the IS. The IS then generates an ephemeral Diffie-Hellman key pair ($\mathtt{SK}'_{IS}$,$\mathtt{PK}'_{IS}$) using the same domain parameters and sends the newly-generated public key to the ePassport. Both the ePassport and the IS derive a new session key K. Chip authentication is immediately followed by passive authentication, which allows an IS to verify if the $\mathtt{PK}_{eP}$ is genuine. The authenticity of the ePassport is established once the ePassport proves that it knows the session key; this happens implicitly when the derived session key is used to communicate successfully with the ePassport.

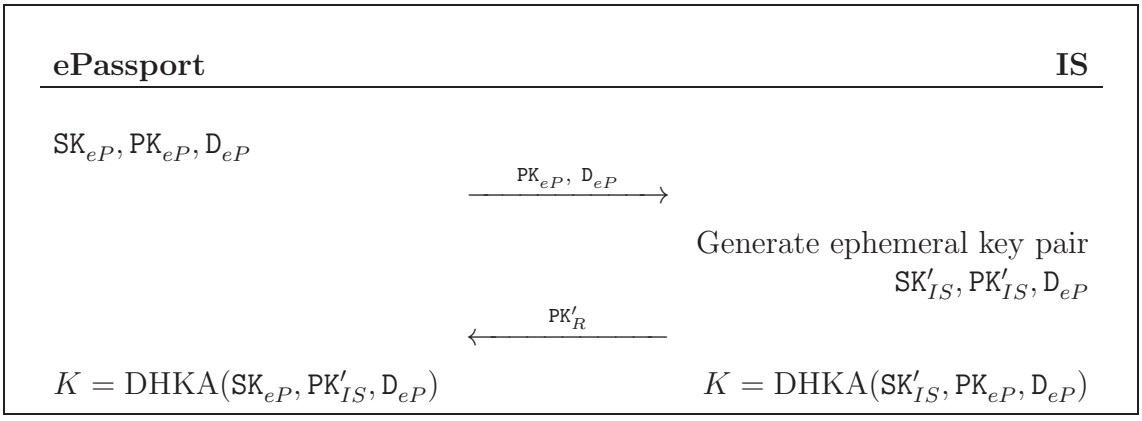

FIGURE 5.2: Chip Authentication

5.1.3 Terminal Authentication

Terminal Authentication is also a mandatory EAC mechanism that involves a two-pass challenge-response protocol and allows the ePassport to authenticate an IS. The TA is only carried out after a successful run of both the chip authentication and the passive authentication, because it provides only an a unilateral authentication of the IS. During terminal authentication, the IS is required to send a certificate chain ($\texttt{CERT}_{IS}\langle\rangle \leftarrow \texttt{CERT}_{DV}\langle\rangle \leftarrow \texttt{CERT}_{CVCA^H}\langle\rangle$), where the certificate $\texttt{CERT}_{CVCA^H}\langle\rangle$ is from the ePassport's home country CA, which is also stored within the ePassport. The chain indicates that the visiting country's IS is certified by a visiting country's Document Verifier (DV) that, in turn, is certified by the ePassport's home country CVCA. After the certificate chain is validated by the ePassport, it sends a challenge to the IS. The IS responds with a digitally-signed message that contains the received challenge, the ephemeral public key used in chip authentication and the ePassport ID (ID_{eP}), where, ID_{eP} is the document ID obtained from the ePassport's MRZ. If the ePassport successfully verifies the signature received, then it has successfully authenticated the IS, and provides access to the extra biometric identifiers.

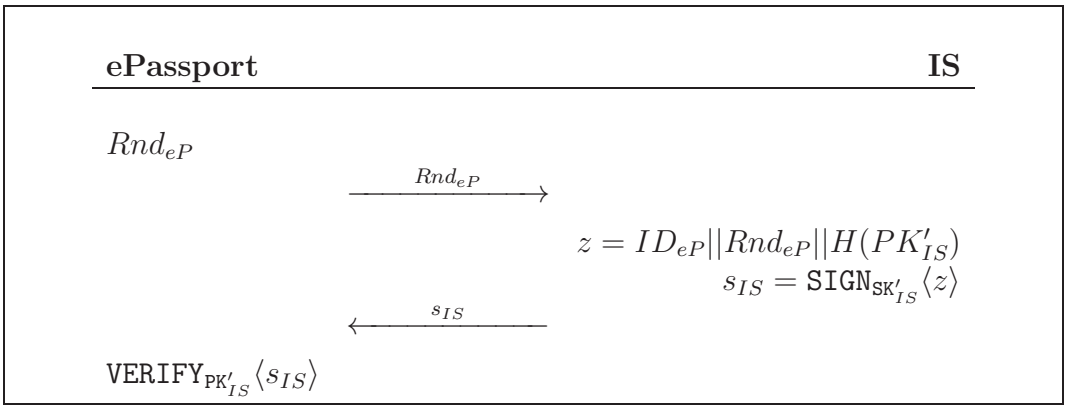

Figure 5.3: Terminal Authentication

5.1.4 Security issues in EAC-based ePassports

The EAC provides a much better security compared to the first-generation ePassports. Nevertheless, the EAC proposal still relies on the basic access control protocol to derive

the initial session key needed to access the ePassport bearer's details, including the facial biometric. Therefore, because of the inherent weaknesses in the basic access control protocol as discussed in the previous chapter keys that have insufficient entropy, the EAC proposal also suffers from the same weaknesses. Though the access to the secondary biometric is restricted, an adversary can still obtain the biographic and facial biometric information.

The EAC proposal makes extensive use of the PKI. Both chip and terminal authentication protocols require the verification of the certificates involving the entire certification hierarchy. The ePassport initially contains the root level certificate ($\texttt{CERT}_{CVCA^H}\langle\rangle$). This is written into the ePassport chip by the country's document issuer at initialisation[1]. Because the ePassport chips are time-less devices, that is, they do not have any internal clock, this makes them vulnerable to attacks using expired certificates.

Klüger [118, 119] acknowledges this vulnerability and proposes that an ePassport should write the $\texttt{CERT}_{CVCA^H}\langle\rangle$ with the latest certificate obtained when it performs a terminal authentication with a visiting country's IS. During the first run of the terminal authentication, the time of expiry of the $\texttt{CERT}_{CVCA^H}\langle\rangle$ (that was initially written) is used as a reference time to validate the visiting country's IS certificate, and after a successful run of the protocol, the ePassport will store the $\texttt{CERT}_{CVCA^H}\langle\rangle$ that is present in the certificate chain received from the IS. Nevertheless, the protocol is still vulnerable to attacks that use expired IS certificates. The validity of the IS certificates is considerably shorter when compared to the CVCA certificates. A compromised IS even if its certificate had expired, would still be able to authenticate itself to an ePassport and obtain access to sensitive ePassport information including fingerprints and iris scans, that were intended to be protected by the EAC. The attack is more effective for infrequently-used ePassports. Such ePassports would be more vulnerable because they have only the initially written $\texttt{CERT}_{CVCA^H}\langle\rangle$, which itself may have expired. Because the ePassport uses the time on the $\texttt{CERT}_{CVCA^H}\langle\rangle$ as a reference point, it would accept any certificate, as long as its validity is prior to the current reference time recorded on the ePassport.

[1]Initialisation here refers to the time an ePassport is issued

The approach of sending the certificate chains can also lead to a Denial-of-Service (DOS) attack on an ePassport. Because an IS terminal is not authenticated during or before the chip authentication, a malicious terminal could flood the ePassport by sending many public keys and certificates. Because of the limited memory that is available in an ePassport chip, the ePassport chip could run out of memory, essentially, stopping the ePassport from functioning in the desired manner.

Compared to the first-generation ePassport standard, the EAC proposal also introduces new security weaknesses. The ePassport now requires write-access to the ePassport chip to update its $\texttt{CERT}_{CVCA^H}\langle\rangle$. This could be used by an illegitimate ePassport bearer to update the ePassport chip with false information. The EAC does not specify any guideline about how write-access would be controlled; it can, potentially, lead to conflicting implementations among various countries. Another drawback of the EAC proposal is the cross-certification among the countries. Every country implementing the EAC would be required to certify the other country's document verifiers. Essentially, that means that each document verifier certifying an IS will need to be certified by the CSVA of every participating country. The EAC recommends that the validity of the document verifier certificates be one third of the CVCA certificate's validity period. This becomes a extremely complex undertaking for each country with respect to certifying other participating country's document verifiers and the maintenance of the revocation lists.

The EAC also does not resolve the Grandmaster Chess Attack [55] to which the first-generation passports were vulnerable. The basic access control protocol is used only to form a session key for an encrypted communication channel between the ePassport and an IS, but it does not provide authentication. Therefore, the ePassport establishes a session key even though it is not sure if an IS is genuine. This also raises concerns regarding the privacy of the ePassport bearer. The ePassport chip sends its identification details (public key) during chip authentication, even before it has authenticated the IS. Therefore, this would make it very easy for an attacker to track an ePassport bearer, because an attacker is not required to authenticate to an ePassport before obtaining the details from the ePassport. The EAC also does not provide any guarantees

regarding the freshness or origin of the messages.

5.2 An On-line Secure ePassport Protocol

To resolve the security issues identified in both the first- and second-generation of ePassports, in this section, we present an on-line secure ePassport protocol (OSEP protocol). The proposed protocol leverages the infrastructure available for the standard non-electronic passports to provide mutual authentication between an ePassport and an IS. Currently, most security organisations are involved in passive monitoring of the border security checkpoints. When a passport bearer is validated at a border security checkpoint, the bearer's details are collected and entered into a database. The security organisation compares this database against the database of known offenders (for instance, terrorists and wanted criminals). The OSEP protocol changes this to an active monitoring system. The border security check-point or the DV can now cross-check against the database of known offenders themselves, thus simplifying the process of the identification of criminals.

The new proposal provides the following security features:

- An ePassport discloses its information stored on the ePassport chip only after a successful authentication of the IS. This prevents revealing the ePassports identity to a third party that is not authorised or cannot be authenticated. This prevents the covert collection of ePassport data from 'skimming' or 'eavesdropping' attacks that were very effective against both the first- and the second-generation ePassports.

- The OSEP protocol provides proof-of-freshness and the authenticity for messages between the participating entities.

- The OSEP protocol uses the existing ICAO PKI implementation (as in first-generation ePassports) and eliminates the need for cross-certification among the participating countries, as required by the EAC (second-generation ePassports).

- The OSEP protocol eliminates the need for certificate chain verification by an ePassport. Only the top level certificate ($\texttt{CERT}_{CVCA}\langle\rangle$) is required to be stored in an ePassport, thus reducing the memory requirements and preventing a malicious reader from performing a DOS attack on an ePassport.

- The OSEP protocol also requires an IS to provide proof-of-correctness for public key parameters to an ePassport. This allows an ePassport to verify that an IS is using the correct domain parameters and to prevent related attacks [8, 195].

5.2.1 Initial Setup

All entities involved in the protocol share the public quantities p, q, g where:

- p is the modulus, a prime number of the order 1024 bits or more.

- q is a prime number in the range of 159-160 bits.

- g is a generator of order q, where $\forall i < q, g^i \neq 1 \mod p$.

- Each entity has its own public key and private key pair ($\texttt{PK}_i$,$\texttt{SK}_i$) where $\texttt{PK}_i = g^{(\texttt{SK}_i)} \mod p$

- Entity i's public key ($\texttt{PK}_i$) is certified by its root certification authority (j), and is represented as $\texttt{CERT}_j\langle \texttt{PK}_i, i\rangle$.

- The public parameters p, q, g used by an ePassport are also certified by its root certification authority.

5.2.2 Phase One - IS Authentication

Step 1 ($\mathcal{IS}$) When an ePassport is presented to an IS, the IS reads the MRZ information on the ePassport using an MRZ reader and issues the command `GET CHALLENGE` to the ePassport chip.

Step 2 ($\lceil\mathcal{P}$) The ePassport chip then generates a random $eP \in_R 1 \leq eP \leq q-1$ and computes $K_{eP} = g^{eP} \mod p$, playing its part in the key agreement process to

establish a session key. The ePassport replies to the `GET CHALLENGE` command by sending K_{eP} and its domain parameters p, q, g.

$$eP \longrightarrow \mathcal{IS} : K_{eP}, p, q, g$$

Step 3 ($\mathcal{IS}$) On receiving the response from the ePassport, the IS generates a random $is \in_R 1 \leq is \leq q-1$ and computes its part of the session key as $K_{is} = g^{is} \mod p$. The IS digitally signs the message containing MRZ value of the ePassport and K_{eP}.

$$S_{\mathcal{IS}} = \mathtt{SIGN}_{\mathtt{SK}_{\mathcal{IS}}} \langle MRZ \| K_{eP} \rangle$$

It then contacts the nearest DV of the ePassports issuing country and obtains its public key. The IS encrypts and sends its signature $S_{\mathcal{IS}}$ along with the ePassport's MRZ information and K_{eP} using the DV's public key $\mathtt{PK}_{\mathcal{DV}}$.

$$\mathcal{IS} \longrightarrow \mathcal{DV} : \mathtt{ENC}_{\mathtt{PK}_{\mathcal{DV}}} \langle S_{\mathcal{IS}}, MRZ, K_{eP} \rangle, \mathtt{CERT}_{\mathcal{CVCA}} \langle \mathtt{PK}_{\mathcal{IS}}, \mathcal{IS} \rangle$$

Step 4 ($\mathcal{DV}$) The DV decrypts the message received from the IS and verifies the $\mathtt{CERT}_{\mathcal{CVCA}} \langle \mathtt{PK}_{\mathcal{IS}}, \mathcal{IS} \rangle$ and the signature $S_{\mathcal{IS}}$. If the verification holds, the DV knows that the IS is genuine, and creates a digitally-signed message $S_{\mathcal{DV}}$ to prove the IS's authenticity to the ePassport.

$$S_{\mathcal{DV}} = \mathtt{SIGN}_{\mathtt{SK}_{\mathcal{DV}}} \langle MRZ \| K_{eP} \| \mathtt{PK}_{\mathcal{IS}} \rangle, \mathtt{CERT}_{CVCA} \langle \mathtt{PK}_{\mathcal{DV}}, \mathcal{DV} \rangle$$

The DV encrypts and sends the signature $S_{\mathcal{DV}}$ using the public key $\mathtt{PK}_{\mathcal{IS}}$ of IS.

$$\mathcal{DV} \longrightarrow \mathcal{IS} : \mathtt{ENC}_{\mathtt{PK}_{\mathcal{IS}}} \langle S_{\mathcal{DV}}, [\mathtt{PK}_{eP}] \rangle$$

The DV may choose to send the public key of the ePassport if required. This has an obvious advantage, because the IS system now trusts the DV to be genuine.

It can obtain a copy of ePassport's PK to verify during ePassport authentication.

Step 5 ($\mathcal{IS}$) After decrypting the message received, the IS computes the session key $K_{ePis} = (K_{eP})^{is}$ and encrypts the signature received from the DV, the ePassport MRZ information and K_{eP} using K_{ePis}. It also digitally signs its part of the session key K_{is}.

$$\mathcal{IS} \longrightarrow eP : K_{is}, \mathtt{SIGN}_{\mathtt{SK}_{\mathcal{IS}}}\langle K_{is}, p, q, g\rangle, \mathtt{ENC}_{K_{ePis}}\langle S_{\mathcal{DV}}, MRZ, K_{eP}\rangle$$

Step 6 $\mathcal{C}$ On receiving the message from the IS, the ePassport computes the session key $K_{ePis} = (K_{is})^{eP}$. It decrypts the message received using the session key and verifies the signature $S_{\mathcal{DV}}$ and $\mathtt{VERIFY}_{PK_{\mathcal{IS}}}\langle \mathtt{SIGN}_{\mathtt{SK}_{\mathcal{IS}}}\langle K_{is}, p, q, g\rangle\rangle$. On successful verification, the ePassport is convinced that the IS system is genuine and can proceed further in releasing its details. All further communications between an ePassport and IS are encrypted using the session key K_{ePis}

5.2.3 Phase Two - ePassport Authentication

Step 1 $\mathcal{C}$ The IS issues an `INTERNAL AUTHENTICATE` command to the ePassport. The ePassport on receiving the command, the ePassport creates a signature $S_{eP} = \mathtt{SIGN}_{\mathtt{SK}_{eP}}\langle MRZ \| K_{ePis}\rangle$ and sends its domain parameter certificate to the IS. The entire message is encrypted using the session key K_{ePis}.

$$eP \longrightarrow \mathcal{IS} : \mathtt{ENC}_{K_{ePis}}\langle S_{eP}, \mathtt{CERT}_{DV}\langle \mathtt{PK}_{eP}\rangle, \mathtt{CERT}_{DV}\langle p, q, g\rangle\rangle$$

Step 2 ($\mathcal{IS}$) The IS decrypts the message and verifies $\mathtt{CERT}_{DV}\langle p, q, g\rangle, \mathtt{CERT}_{DV}\langle \mathtt{PK}_{eP}\rangle$ and S_{eP}. If all three verifications hold then the IS is convinced that the ePassport is genuine and authentic.

During the IS authentication phase, an IS sends the ePassport's MRZ information to the nearest ePassport's DV, which could be an ePassport country's embassy. Embassies are DV's because they are allowed to issue ePassports to their citizens and because most

embassies are located within an IS's home country, any network connection issues will be minimal.

Sending the MRZ information is also advantageous, because the embassy now has a list of all its citizens that have passed through a visiting country's border security checkpoint. We do not see any privacy implications, because, in most cases, countries require their citizens to register at embassies when they are visiting a foreign country.

5.3 Analysis of ePassport scheme

This section identifies the important security goals required in an ePassport protocol providing key agreement with mutual authentication, and performs a security analysis of our proposed OSEP protocol.

5.3.1 Requirement Analysis

The two most important requirements for providing border security are the identification of the passport bearer and the authentication of the passport data. The digital nature of the data stored in an ePassport makes them easy to be either copied or altered. Therefore, an ePassport protocol will need to ensure security requirements that will affect the electronic data storage and transmission. Though, [96, 119] provides a brief overview security goals for ePassports, their description are limited and do not consider the goals that are essential for analysing the cryptographic protocols. Our security goals for an ePassport system are:

Goal 1 *Identification* After the successful completion of an ePassport protocol, both the ePassport and the IS must obtain guarantees (unforgeable proof) of other entity's identity.

Goal 2 *Authenticity* After a successful completion of an ePassport protocol, both the ePassport and the IS must have guarantees on the authenticity of the messages received during the conversation with each other, and should also have an undeniable proof-of-origin of the messages.

Goal 3 *Data confidentiality* Data confidentiality during an ePassport protocol run is guaranteed by the security of the session key agreed between the ePassport and the IS. Therefore, if an ePassport completes a single protocol run with the understanding that it has negotiated a session key K with an IS, the same ePassport is guaranteed that no other third-party has learnt key the K and if the IS completes the protocol run, then it associates the key K with the ePassport. Data confidentiality of the information stored in an ePassport chip is not considered, because it is protocol-independent, but is necessary for the ePassport protocol to detect if information was tampered with; this is provided by our integrity goal.

Goal 4 *Integrity* The integrity of the data in an ePassport chip is guaranteed by signatures. Therefore, during a run of an ePassport protocol, if an IS successfully verifies and validates the signatures on the messages from an ePassport, then the IS obtains a guarantee that the information held in an ePassport chip has not been modified by any third party or the ePassport bearer after its initialisation by the document issuer.

Goal 5 *Privacy* In every run of an ePassport protocol, the ePassport bearers are assured that their ePassport's digital identities are revealed only to the authenticated IS involved in the current protocol run.

Goal 6 *Session key security* Both entities, an ePassport and an IS, have proof that each run of an ePassport protocol is unique and comprises long term keys, and does not compromise the session keys derived in previous protocol runs.

5.3.2 Security Analysis of OSEP protocol

In this section, we present a brief security analysis of the OSEP protocol. We first list our assumptions and then our claims on the OSEP protocol's security that corresponds to our security goals described in §5.3.1.

Assumptions

- In an OSEP protocol, both an ePassport and an IS instantiate a non-concurrent protocol run (session) between them, whereas the session connections between an IS and a DV may run concurrently.
- An IS is always the initiator of a protocol run and an ePassport is always the responder.
- The underlying security for Diffie-Hellman (DH) key exchange; the Decisional Diffie-Hellman (DDH) assumption holds.
- Cryptographic primitives such as symmetric and public key encryption, digital signatures, message authentication codes and hash functions are secure under the standard security notions in the cryptographic literature.

Lemma 1. *If the encryption scheme used in the protocol is secure against a CCA2 attack then at the end of the OSEP protocol, both the eP and the $\mathcal{IS}$ will complete matching sessions and get the same session key.*

Proof (Sketch) *:* Because the signature algorithm is secure against existential forgery by adaptive chosen-message attack (by assumption), the MRZ information together with the randomness of the K_{eP} and K_{is} guarantees the freshness of the message and binds the message with the two communicating parties. Therefore an attacker cannot forge or modify a message. For attackers to forge or modify a message that is acceptable by the $\mathcal{IS}$ or the eP, they would need to forge the signature on the $\mathtt{SIGN}_{\mathtt{SK}_{IS}}\langle K_i, p, q, g\rangle$ in phase 1, step 5 or forge the signature on the S_{eP} in phase 2, step 1. This contradicts our presupposition.

Furthermore, the digitally signature generated by eP contains the freshly generated session key K_{ePis}. This prevents the replay of messages from a previous run by an adversary who does not have the knowledge to generate signatures on both the K_{eP} and K_{ePis}. □

Theorem 5.3.1. *The protocol provided in Section 5.2 is SK-secure if the encryption scheme adopted is secure against a CCA2 attack.*

Proof. To prove our protocols are SK-secure [42] (refer appendix C), we have to prove that the eP and the $\mathcal{IS}$ get the same session key after they complete the matching sessions and that an adversary cannot distinguish the session key K_{ePis} from a random value with a non-negligible advantage. The former directly follows Lemma 1 and the following Lemma provides proof for the later.

Lemma 2. *Assuming the DDH and the signature scheme is secure, then an attacker cannot distinguish the session key K_{ePis} from a random value with a non-negligible advantage.*

Proof (Sketch) *:* The proof is by contradiction. Assume that an attacker can distinguish the session key K_{ePis} from a random value with a non-negligible advantage η. In the C-K model, the key exchange attacker is not permitted to corrupt the test session or its matching session, so an attacker cannot directly get the session key K_{ePis} from an attack on the OSEP protocol. Therefore, the attacker has two possible methods to distinguish K_{ePis} from a random value.

- The attacker learns the session key K_{ePis}
- The attacker successfully establishes a session (other than a test or its matching session) that has the same session key as the test session.

The first method means that, given g, g^{eP}, g^{is}, g^{α}, the attacker is able to distinguish $\alpha = ePis$ from a random value. This contradicts our DDH assumption. For the second method, there are two strategies an attacker can take.

1. After the eP and the $\mathcal{IS}$ complete the matching sessions, the attacker establishes a new session with either the eP or the $\mathcal{IS}$. However, this session key will be not the same as K_{ePis} because the values eP and is are chosen randomly by the eP or the $\mathcal{IS}$.

2. The attacker intervenes during the run of the protocol and makes the eP and the $\mathcal{IS}$ receive the same session key but not complete matching sessions. This is not feasible either because from Lemma 1, we know that an attacker cannot succeed in this intervention.

□

Thus from Lemma 1 and Lemma 2, we know that the $\mathcal{C}$ and the $\mathcal{IS}$ will get the same session key after the completion of the matching sessions and the attacker cannot distinguish the session key from a random value with a non-negligible advantage. In accordance with the definition of the SK-security, the OSEP is SK-secure. □

Theorem 5.3.2. *The OSEP provides undeniable proof of identification of both the eP and the $\mathcal{IS}$*

Proof (Sketch) *:* The message sent to the eP by the $\mathcal{IS}$ in step 5 of the IS authentication phase includes the values, $S_{\mathcal{DV}}$, MRZ and K_{eP}. The signed message $S_{\mathcal{DV}}$ contains public key of the $\mathcal{IS}$ verified by the $\mathcal{DV}$, so it is sufficient for the eP to verify $S_{\mathcal{DV}}$ to successfully identify the $\mathcal{IS}$ as genuine.

An adversary wishing to falsely identify as an $\mathcal{IS}$ will need to forge $S_{\mathcal{DV}}$. $S_{\mathcal{DV}}$ can be only generated with a valid DV's secret key ($\mathtt{SK}_{\mathcal{DV}}$). Thus, an adversary cannot forge $S_{\mathcal{DV}}$, because he does not have the knowledge of the $\mathtt{SK}_{\mathcal{DV}}$.

An adversary, not in possession of either the K_{ePis} or the $\mathtt{SK}_{eP}$ will not be able to identify as a genuine eP, as in EP authentication phase the eP is required to digitally sign its MRZ and the freshly generated session key K_{ePis}. Therefore, the OSEP provides non-refutable proof of identification for both the $\mathcal{IS}$ and the eP. □

Theorem 5.3.3. *The OSEP provides perfect forward secrecy under a relaxed privacy requirement and partial forward secrecy under strict privacy requirement.*

Proof (Sketch) *:* We define a strict privacy requirement as, *the ePassport protocol guarantees that no information about the ePassport bearer will be available to any unauthorised entities.* A relaxed privacy requirement is defined as, *the ePassport protocol guarantees that the digital identity or biometric information of an ePassport bearer will not be available to any unauthorised entities.*

The OSEP provides partial forward secrecy under the strict privacy requirement, because the loss of the long-term secret key of both the $\mathcal{IS}$ and the $\mathcal{DV}$ will reveal the MRZ information of an ePassport. But, the compromise of a long-term key does

not compromise the previous-established session keys. Also, any loss of a session key in the previous protocol runs does not compromise the future runs of an ePassport protocol. Thus under the relaxed privacy requirement, the OSEP provides perfect forward secrecy. □

In addition, in the OSEP, ePassport bearers are guaranteed that the protection of their digital identity from an unauthenticated $\mathcal{IS}$ and *unknown adversaries* because the digital identity of an ePassport bearer $\mathtt{PK}_{eP}$ is revealed only in the step one of the EP authentication phase. The EP authentication phase follows a successful IS authentication phase, therefore the eP is assured on the $\mathcal{IS}$'s identity. The digital identity is also protected from any adversary's eavesdropping on the communication, because it is encrypted using the fresh secure-session key established during the IS authentication phase.

The OSEP also provides a tamper-detectable integrity check for the data in an ePassport's chip. The integrity of the ePassport data provided in the OSEP is similar to that provided by both the first- and the second-generation ePassports. The data stored in an ePassport's chip is hashed and digitally-signed by the ePassport's document signer at the time of initialisation. Therefore, as a consequence of assumption four that hash functions and digital signatures are secure the OSEP protocol provides a tamper-detectable integrity. An adversary wishing to authenticate the modified data will need to forge the digital signature of the document signer on the hash values. This is not feasible because the adversary does not have the knowledge of the document signer's private key $\mathtt{SK}_{\mathcal{DS}}$.

To summarise, the proposed OSEP is a simple and efficient protocol; its main advantage is that it not only protects the ePassport chip's data from an eavesdrop during the communication, but also restricts the access to an unauthenticated IS. The protocol requires very little modification to the existing PKI implemented by the first-generation ePassport standard. An obvious disadvantage of the OSEP is the on-line nature of its authentication mechanism. An IS is required to contact the ePassport country's DV and authenticate itself before it can continue communications with the ePassport. This process might incur some delay, but we expect this delay to be minimal

because the communication between an IS and a DV is generally carried out over a high-speed network.

5.4 Summary

The security techniques implemented in both the first- and second-generation ePassports do not adequately protect an ePassport bearer. The first-generation ePassport standard is vulnerable to brute force attacks, because session keys generated have a very low entropy. The second-generation ePassport proposal requires extensive modifications to the existing infrastructure, and still relies on the first-generation standard to provide a secure connection to protect the primary biometric identifiers. Both the standard have ignored the need to protect the ePassport details while setting up a communication; this makes the ePassport bearer vulnerable to identity theft and covert surveillance.

The risk of identity theft or illegal entries into a country are further increased when ePassports can be used along with a *SmartGate* system [10], that are currently being implemented in Australia. ePassports also introduce facial biometric recognition for the verification of an ePassport bearer, which is less intrusive when compared to other biometric systems. However, facial biometrics are not very secure because of the relatively low uniqueness factor and are prone to inter-class similarities. Unattended border control check-ins increase the risk of fraudulent facial biometric verifications being undetected and eavesdropping on communications between the ePassport and the IS.

In this chapter, we have proposed a new on-line ePassport protocol that resolves many of the weaknesses in both the first- and second-generation ePassport protocols. The proposal also offers significant security advantages. These security measures will make it extremely hard for a malicious user to authenticate as a genuine ePassport bearer or as an IS. The proposed protocol also protects the details of an ePassport bearer from an unauthorised IS, thus reducing the threat of identity theft. The proposed protocol also uses the existing PKI infrastructure used for the first-generation

ePassport standard and eliminates the need for sending a certificate chain as proposed in the second-generation ePassport standard. This makes an ePassport under the OSEP protocol less vulnerable to DOS-based attacks.

Electronic passports are an important step in the right direction. They enable countries to digitise their security at the border control and provide faster and safer processing of the ePassport bearers. The OSEP strengthens this process by providing an enhanced ePassport security measure.

6

Privacy Enhancements for Active Devices

6.1 Privacy Issues with Active Devices

While passive and semi-passive devices harvest their operating energy from a reader, active devices have a directly-connected power source that enables them to communicate with a reader and power their circuitry. Having a directly-connected power source makes active devices extremely suitable for long-range and high-speed communication, in addition to being capable of performing computationally-intensive tasks. For example, the EPC Global Class 4, UHF-based (Ultra High Frequency) active devices operating at 433MHz can communicate over 100 meters, and thus are typically used in container identification in warehouses, medical equipment identification and the identification of defence equipment (in war zones) that are prone to various environmental

interference. Recently, active devices in the super high frequency or microwaves operating at 2.45 GHz and 5.8 GHz have been developed [6], which integrate into the existing IEEE wireless LAN standards (such as 802.11 a/b/g/n) that also operates at similar frequencies.

Passive devices require a reader to initiate the communication, thus restricting the nature of the information that is obtained. For example, the tracking of an ePassport can be performed only in selected locations where a compliant reader exists. With active devices, the communication is typically initiated by the device itself; thus, the tracking of the entity attached to a device is no longer restricted to only a selected few locations. Because active devices are being integrated into location based solutions (such as, using satellites and assisted-GPS [129, 183]) with real-time monitoring capabilities, the privacy problems become obvious.

Dedicated hardware-based co-processors performing cryptographic and mathematically-intensive operations have been in existence for a long time. Recently, many companies have begun embedding active hardware-based security modules into various computing devices, such as mobile telephones, GPS devices, personal computers and network equipments, to provide a hardware-based endpoint security. One such implementation that is widely popular is the Trusted Platform Module (TPM). Promoted by the Trusted Computing Group [3], the Trusted Platform Module (TPM) is a tamper-resistant hardware chip that forms the basis of trusted computing. The TPM is a secure chip that provides a hardware-based approach to manage user authentication, network access and data protection. The hardware chip is bound to the motherboard of the computing device, and stores certificates, keys and performs the necessary cryptographic operations. A TPM also enhances multi-factor authentication and complements biometric readers by securely storing the keys associated with the biometric within the module. Because TPMs can be integrated with other traditional enterprise tools, such as an active directory and a Novell eDirectory, they provide a seamless method of integrating certificate-based, biometric and hardware-based authentication techniques. A TPM also provides other security services, such as a secure boot and sealed storage: an overview of a TPM and its security features is provided in Appendix D

The deployment of the TPM raises some valid privacy concerns. Privacy protection in a TPM currently involves two mechanisms, Privacy CA-based attestation (TPM v1.1) [187] and Direct Anonymous Attestation (TPM v1.2)[33, 188]. In a Privacy CA-based attestation, the authentication is based on the direct use of the TPM's EK. This method will compromise the anonymity of the module, because all transactions performed by the same TPM can be linked. Furthermore, it will compromise the anonymity of the user associated with the module, because the users/TPMs activity can be tracked. Based on the security requirement of a non-revealable master public key in a TPM, Brickell *et al.* in [33] propose a method for direct anonymous attestation (DAA) that provides anonymity for a user, based on the Camenisch-Lysyanskaya credential system [38]. The current solution for the DAA is a computing intensive construction. To complete all cryptographic calculations in real time, the computation has to be distributed between the TPM and its host, typically, the device to which it is attached. This introduces an obvious security weakness in the system. Now the cryptographic computations are not performed only by the trusted hardware security module and are therefore prone to software-based vulnerabilities in the host system that the TPM is intended to protect against. Also, the scheme [33] does not provide a secret key linkability for the pseudonyms that are generated. Consequently, the TPM must maintain a database of those identities and the associated secret keys. This database can get quite large if the TPM serves a large group of users. Typically, the database would have to be stored outside the chip, thus defeating the purpose for which the TPMs were introduced.

Another issue that is often voiced with respect to TPMs is users privacy when using Digital Rights Management (DRM)-based software. A DRM system, when combined with a TPM introduces a new security layer where the encryption keys and certificates can be bound to a specific platform and copies can be limited to only that specified platform. This has raised concerns from privacy groups about the purchase of such a TPM-based DRM enabled software. For example, a primary concern is that the purchase of music or movie content should remain anonymous, protecting the user's privacy. Because the DRM-based software will now employ information about users

and their platforms, there are fears that the owners of the software products can employ a tracking system that records information about the users and their platform, thus linking them to specific content.

In this chapter, we try to resolve these privacy issues by proposing a pseudonym system for active devices. The system provides restricted anonymity and supports colligation between a trusted *high value* secret key and newly-generated pseudonyms. We first provide a brief overview of the pseudonym systems and their related work. We then provide the details on the construction of an anonymous certification system and cryptographic techniques that underpins our construction. We then present our pseudonym system and discus the security of our proposed construction. Finally, we discuss the integration of our proposed pseudonym system in a TPM-based setting.

6.2 Pseudonymns

The use of pseudonyms have been proposed as a mechanism to hide a user's identity by providing anonymity, while still being suitable to authenticate the holder of the pseudonym in a communication system [45]. David Chaum argues that using pseudonyms provides a way to allow a user to work anonymously with multiple organisations by allowing the users to obtain a credential from one organisation using their pseudonym and obtain services using that credential from another organisation without revealing their true identity [45]. To this end, Chaum and Evertse developed a pseudonym system and propose a RSA-based implementation while relying on a trusted centre that must sign all credentials [46]. Chen extends the scheme from [45] and presents its discrete-logarithm version that relies on a trusted centre [48]. An advantage of these schemes is that they allow the users to generate pseudonyms, giving the users a greater degree of control over their identity. However, these schemes have a common weakness. Although the identity of a user is hidden, the credentials (such as the certificates of their public key) or pseudonyms can be easily shared (unauthorised transfer) with other users.

Based on the security of preserving a high-value (*master*) secret key, Canetti *et*

al. [41] and Lysayanskaya *et al.* [135] independently propose non-transferable pseudonym systems. Though credentials obtained on pseudonyms can be used anonymously, the authors of [41] assume that the certification authority (CA) grants credentials only when each user has revealed their true identity to them. This makes their scheme prone to collusion between a CA and a verifier, because the real identity associated with the user pseudonym can be deduced. The scheme from [135] protects against an unauthorised transfer of the user credentials by forcing a user to reveal the master secret key should they choose to share their credentials. But the scheme shares the same weakness as in [41] – during the registration phase, users are required to disclose their true identity (master public key) to a CA.

6.2.1 Scope and Contribution

This chapter presents a pseudonym system that is based on the public key cryptosystem. The main idea is to use a single, trusted master secret key with many matching public keys (pseudonyms). The proposed system gives users the ability to generate multiple pseudonyms (that are independent of the master public key) from a trusted master secret key. An important property of the system is that it provides the users with the ability to generate signatures using the master secret key, which are verifiable using certificates that were issued against pseudonyms.

To consider an example, a TPM contains a certified public-secret key pair. The public key is certified by its manufacturer and recorded on the TPM chip at the time of manufacturing. The certified public key of the chip can be used to authenticate the machine with the TPM. The TPM is used to further certify public keys of users associated with the machine. A verifier can authenticate a user based on the certificate chain consisting of the user certificate, the TPM certificate and the manufacturer's certificate. However, revealing the identity of the machine to every verifier would not only compromise the anonymity of the machine, but also the anonymity of the user(s) of the machine. It is possible to identify a user using their pseudonyms, but the verifier trusts only the TPM chip's certified public key and not the operating system of the machine or any newly-generated pseudonyms. Therefore, we require a system that gives

a user the ability to generate and control the use of the multiple identities, based on a trusted master identity (TPM's certified public key). Here, the pseudonyms should not only be independent of the master identity (anonymity), but also there is a relation between all pseudonyms generated[1] and the trusted master secret key stored in the chip (we call this relation *colligation*).

Anonymity and colligation are in some sense contradictory. Anonymity requires that, it is impossible (at least computationally) for an entity with knowledge of a pseudonym, to link that pseudonym with either the master identity or any other generated pseudonym. Whereas, colligation requires that the prover be guaranteed that there is an underlying link that exists between all pseudonyms (that appear to be unrelated to each other) was generated from the trusted master secret key. Previously published proposals like, [38, 41, 45, 48, 53, 135] that achieved anonymity have considered a user's identity that consists of public-secret key pair as a single unified structure. Under such assumption it is unfeasible to obtain both anonymity and colligation. We aim to segregate the structure and provide anonymity to a user but still maintain colligation between pseudonyms generated using the user's master secret key. The implication of this structure is that, a user's master secret key becomes highly valuable, as all his pseudonyms are linked directly to the secret key.

6.2.2 Anonymous Certification System

User anonymity and colligation between the master secret key and the user-generated identities is of paramount importance. To provide anonymity to the user-generated identities (pseudonyms), our proposal will make use of an anonymous certification scheme, such as a scheme with blind signatures. An anonymous certification system is necessary to provide anonymity to a user and to prevent collusion between a certifier and a verifier. To this end, we will employ the modified blind signature scheme

[1]To a certifier it is essential that the system provides a guarantee that all pseudonyms from a particular TPM can be traced back to a single secret key; but a verifier needs proof of this binding between the master secret key and only the pseudonym that are currently presented with. We do not make this distinction here.

(refer Section 6.3.4) proposed by Pointcheval [166]. Note that any anonymous certification scheme that supports non-transferability and the revocation of anonymity can be employed with some necessary modifications. To provide colligation between the generated pseudonyms and the master secret key, we can use any one-way function. In our construction we use a squaring modulo a composite integer. In this section, we first we describe the model of an anonymous certification scheme that will provide certificates for user-generated identities (pseudonyms). In the remainder of this section, we summarise the main cryptographic building blocks used in our constructions.

The anonymous certification system (ACS) represents the certification process of a public key by a certifier who does not know the public key. This is essentially a Chaum blind signature [44] on the public key of the user, that is, it provides anonymity to the receiver[2].

A typical ACS consists of four entities and three protocols. The entities are, a user $\mathcal{U}$, a verifier $\mathcal{V}$, a certifier $\mathcal{C}$ and a trustee (tracer) $\mathcal{T}$. The protocol suites include: a *certification protocol*, where $\mathcal{U}$ interacts with $\mathcal{C}$ to obtain a certified pseudonym, that is, the pseudonym is blindly signed; an *identification protocol*, where $\mathcal{V}$ interacts with $\mathcal{U}$ to authenticate $\mathcal{U}$'s credential and provide services; a *trace protocol*, where $\mathcal{T}$ participates and is invoked to trace the real identity associated with $\mathcal{U}$'s pseudonym.

System setting

The user, $\mathcal{U}$, chooses a modulus N_i, such that a $N_i = p_1^{(i)} p_2^{(i)}$, is a product of two distinct large primes each congruent to 3 (mod 4), ($p_1^{(i)}, p_2^{(i)}$ are Blum integers [27]), an element $g \in \mathbb{Z}_{N_i}$ whose order is $\phi(N_i) = (p_1^{(i)} - 1)(p_2^{(i)} - 1)$ and where i is the number of pseudonyms. We also require the modulus for pseudonyms to be different; otherwise anonymity can be compromised trivially by merely maintaining a list of modulus. The user chooses a master secret key $SK_{\mathcal{U}_0} \in \mathbb{Z}_{N_0}$ and publishes the master public key $PK_{\mathcal{U}_0} = g^{SK_{\mathcal{U}_0}} \mod N_0$ (which represents the user's true and public identity). The certifier $\mathcal{C}$ publishes its public key $PK_{\mathcal{C}} = g^{SK_{\mathcal{C}}} \mod N_c$ while keeping the corresponding secret key private. The certifier also publishes the public key of the Trustee $\mathcal{T}$, (for tracing

[2]Whereas, group signature schemes as employed by [33] provide anonymity to the source.

and revocation) which would be of the form $PK_{\mathcal{T}} = g_1^{SK_{\mathcal{T}}} \mod N_T$, where $g_1 \in \mathbb{Z}_{N_T}$. Every user registers with a certification authority to obtain a certificate of the form $\texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_0}\rangle$.

Protocol Certify

The certification involves two steps, the certification of the master public key and the certification of the pseudonyms. In an TPM-based setting the master public key is certified by the manufacturer, and the following describes the certification of the pseudonyms.

The user, $\mathcal{U}$, generates pseudonyms of the form $(PK_{\mathcal{U}_1}, \ldots, PK_{\mathcal{U}_l})$ using the identity generation process described in Section 6.3.3. The users then identify themselves (using the master public key) to the certifier and engages in a *certify* protocol to obtain a certificate for a pseudonym $PK_{\mathcal{U}_i}$. The value of $PK_{\mathcal{U}_i}$ is never revealed to the certifier. We shall express this phase as

$$(PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle) \leftarrow Certify(\mathcal{U}, \mathcal{C}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_0}\rangle)$$

that is, ' $\mathcal{U}$ engages in the certify protocol with $\mathcal{C}$ using $\texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_0}\rangle$ to obtain a certificate on $PK_{\mathcal{U}_i}$, $\texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle$'.

Protocol Identify

A user $\mathcal{U}$ who wishes to avail services offered by a verifier $\mathcal{V}$, engages in an identification protocol to convince that they possess the necessary credentials. We shall express this phase as

$$\langle \texttt{PROOF}_{\mathcal{U}_i}\rangle \leftarrow Identify(\mathcal{U}, \mathcal{V}, PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle, PK_{\mathcal{T}})$$

that is, ' $\mathcal{U}$ engages in an identification protocol with a verifier $\mathcal{V}$ using the psuedonymn $PK_{\mathcal{U}_i}$ and $\texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle$ and which contains the encryption of the identity under the public key $PK_{\mathcal{T}}$'.

Protocol Trace

A verifier who needs to trace the identity of the user contacts the trustee $\mathcal{T}$ by providing the transcript from an identification protocol $\langle \texttt{PROOF}_{\mathcal{U}_i} \rangle$. We shall express this phase as

$$(PK_{\mathcal{U}_0}) \leftarrow Trace(\mathcal{V}, \mathcal{T}, PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i} \rangle, \langle \texttt{PROOF}_{\mathcal{U}_i} \rangle)$$

that is, ' $\mathcal{V}$ engages in the tracing protocol with $\mathcal{T}$ using the values $PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i} \rangle$ and proof of identity use $\langle \texttt{PROOF}_{\mathcal{U}_i} \rangle$ to obtain the master identity $PK_{\mathcal{U}_0}$'.

6.3 Pseudonym System Colligated with Master Secret Key

We first outline our security assumptions and cryptographic tools used, and then present our scheme that consists of four phases, identity generation, certification, identification and trace.

6.3.1 Assumptions

Our system relies on the following assumptions:

- **Assumption 1** (*Factoring*) For any probabilistic poly-time algorithm $\mathcal{G}$ that on input $1^{|N|}$ produces factors of$N$, where N is a composite of two prime number, p_1 and q_1, such that for any probabilistic polynomial time algorithm $\mathcal{A}$, the probability that $\mathcal{A}$ can factor N is negligible, that is, the probability of its success is negligible in the length of $\frac{1}{poly(|N|)}$.

- **Assumption 2** (*Square Root*) for any probabilistic polynomial-time algorithm $\mathcal{A}$ that on input N and a, where N is a composite of two prime numbers, p_1 and q_1 and $a \in \mathrm{QR}_N$ is a quadratic residue, the probability that $\mathcal{A}$ can output b, such that $b^2 \equiv a \mod N$ is negligible, that is, the probability of success is smaller than $\frac{1}{poly(|N|)}$.

- **Assumption 3** (*Square Decisional Diffie-Hellmann*) The square decisional Diffie-Hellman (SDDH) problem is to distinguish between distributions of the form (g, g^a, g^{a^2}) from (g, g^a, g^r), where r is random and a uniformly-chosen integer from $\{1, \ldots, N-1\}$. We assume that there exists no probabilistic polynomial-time algorithm $\mathcal{G}$ that can solve a random instance of the SDDH problem with probability $\frac{1}{2} + \frac{1}{poly(|N|)}$.

We also use the Chaum and Pederson construction [47] as a sub-protocol for an interactive proof of knowledge for the discrete log problem (DL-EQ). Their protocol [47] was designed for the case when a group of the exponents has a prime order, whereas, in our protocol, the group of the exponents have a composite order. However, as suggested by [39], the proof of knowledge of a discrete logarithm from different groups (DL-EQ) holds even when working over a cyclic sub-group of $\mathbb{Z}_N^*$. We combine the DL-EQ with the El-Gamal encryption over a composite modulus [74] to encrypt the master identity of the user under the public key of the trustee, verifiable by the certification authority.

6.3.2 System Setting

The system involves four entities. A user $\mathcal{U}$ who holds a long-term certified public key $PK_{\mathcal{U}_0}$ (we shall call it the master public key), and wishes to hide his identity from a verifier $\mathcal{V}$. The public keys are certified by a certification authority $\mathcal{C}$ and a trustee $\mathcal{T}$ responsible for tracing the pseudonym used by the user.

The $\mathcal{U}$ master public-secret key-pair is generated as in Section 6.2.2. $\mathcal{U}$ then obtains a certificate on the master public key $PK_{\mathcal{U}_0}$ from a certification authority $\mathcal{C}$, which represents the $\mathcal{U}$'s true identity.

The public key of the certification authority is $PK_{\mathcal{C}} = g^{SK_{\mathcal{C}}}$ and the trustee is $PK_{\mathcal{T}} = g_1^{SK_{\mathcal{T}}}$, where $SK_{\mathcal{C}}$ and $SK_{\mathcal{T}}$ are the corresponding secret keys for the certification authority and the trustee respectively.

6.3.3 Identity Generation

$\mathcal{U}$ generates new identities using the following key generation process, which takes the inputs, N_j, g, a counter value i (indicating the total number of new identities being generated), identity level l (number of identities generated previously) and the master secret key $SK_{\mathcal{U}_0}$.

`I-Generation`(g,i,l,$SK_{\mathcal{U}_0}$)
`For` $j = l,\ldots,i$ `do` $PK_{\mathcal{U}_j} = g^{SK_{\mathcal{U}_0}^{2^j}} \mod N_j$ `EndFor`
`Return`($PK_{\mathcal{U}_l},\ldots,PK_{\mathcal{U}_j}$)

During the first run, the value of the identity level l would be 1 and the counter value i is the number of new identities $\mathcal{U}$ requires. Further calls to the key generation, the identity level would be the counter value that was used during the previous run ($l' = i$). An implicit requirement is that, $\mathcal{U}$ should keep track of the values i and l as long as the master public key remains valid.

We could (and do) treat the identities generated as public keys, that are of the form $(PK_{\mathcal{U}_l},\ldots,PK_{\mathcal{U}_i}) = (g^{SK_{\mathcal{U}_0}^{2^l}},\ldots,g^{SK_{\mathcal{U}_0}^{2^i}})$

6.3.4 Certification

The newly-generated public keys ($PK_{\mathcal{U}_1}$, ..., $PK_{\mathcal{U}_l}$) are required to be certified by $\mathcal{C}$ before they can be used. It is possible to use a normal certification procedure currently employed in public key crypto-systems, where the public key $PK_{\mathcal{U}_i}$ is signed by $\mathcal{U}$ using the master secret key $SK_{\mathcal{U}_0}$ and sent to $\mathcal{C}$ for certification. $\mathcal{C}$ verifies the signature using the master public key $PK_{\mathcal{U}_0}$. On a successful verification, $\mathcal{C}$ digitally-signs using his private key $SK_{\mathcal{C}}$ and sends the certificate to $\mathcal{U}$. This method is quite straightforward, but certain applications (for example, applications based on TPM) require the new identities to be protected even from the certifier. So, we propose a modification to the certification scheme based on a blind signature scheme using a composite modulus by Pointcheval [166]. The blind signature scheme now includes the master public key of

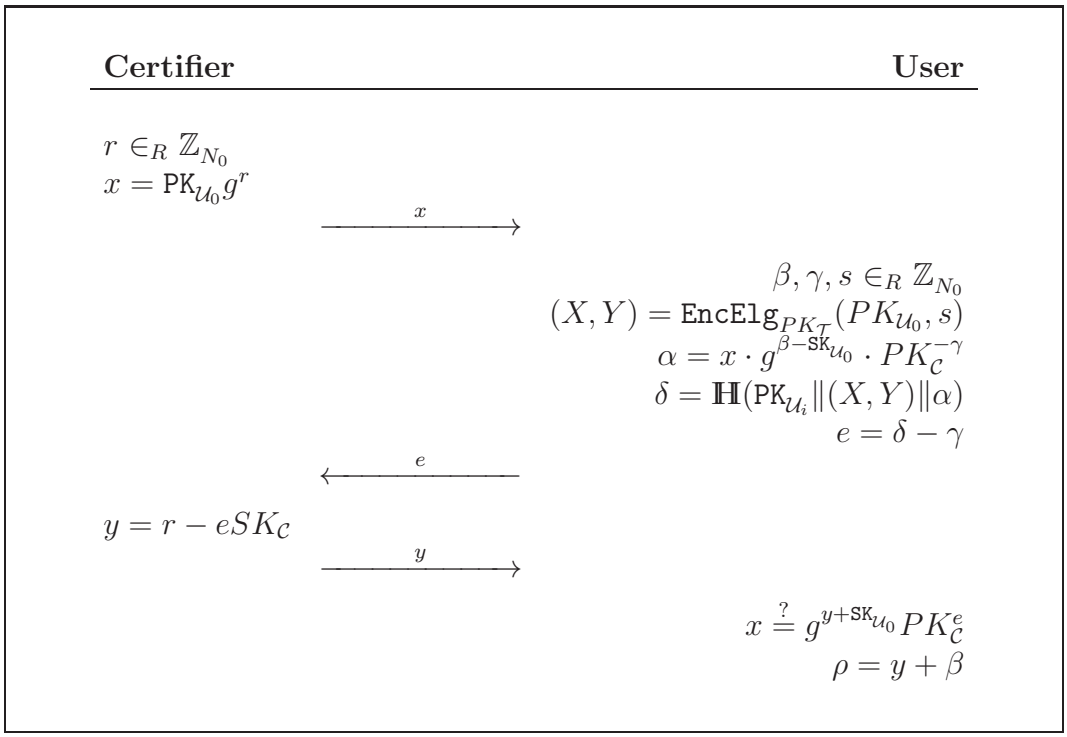

FIGURE 6.1: Modified Blind Certification Protocol of [166]
The signature on $PK_{\mathcal{U}_i}$ is (α, δ, ρ) and a receiver can verify using the relation $\alpha \stackrel{?}{=} g^{\rho} \mathtt{PK}_{\mathcal{C}}^{\delta}$

the user that is used by the certifier to form the commitment and is later verified by the user.

The certification process is represented by:

$$(PK_{\mathcal{U}_i}, \mathtt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle)$$

$$\leftarrow Certify(\mathcal{U}, \mathcal{C}, \mathtt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_0}, (X, Y)\rangle)$$

where, $\mathtt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle$ is the valid blind signature $(PK_{\mathcal{U}_i}, \alpha, \delta, \rho)$ by $\mathcal{C}$ on $PK_{\mathcal{U}_i}$ and (X, Y), accomplished by the three-pass protocol depicted in Figure 6.3.4. The security proof of the modified protocol trivially follows the proof presented in Pointcheval's paper [166].

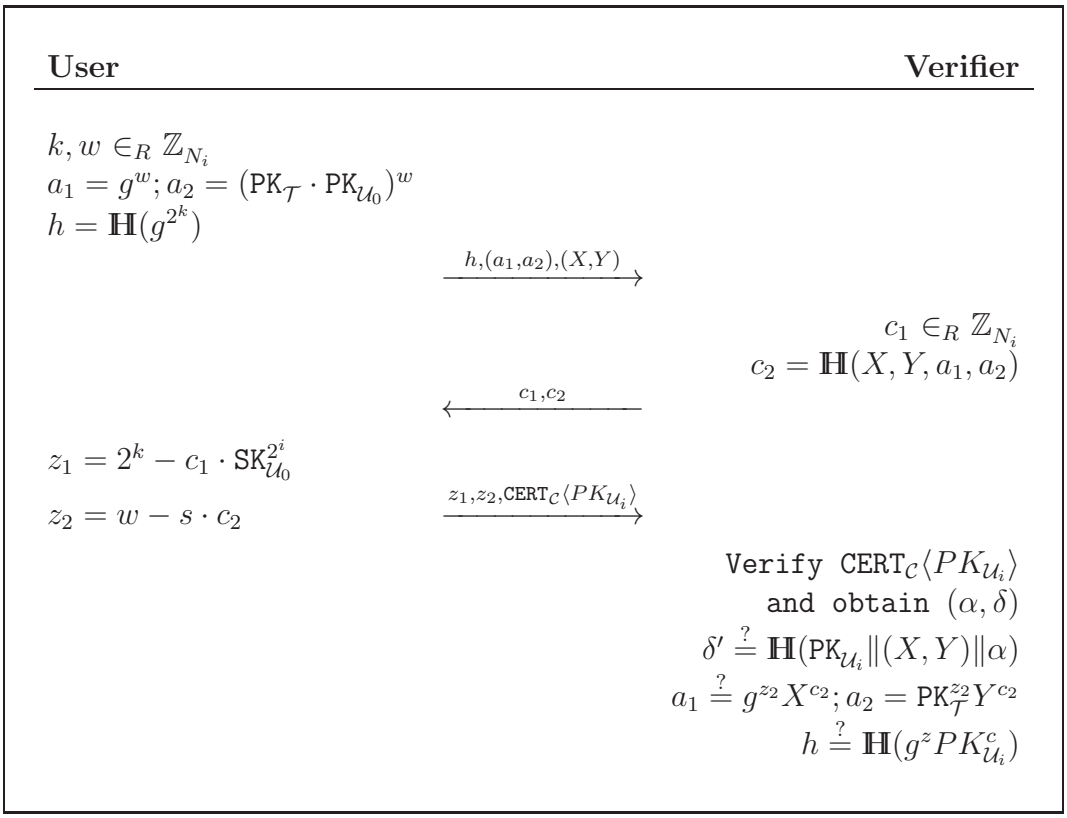

FIGURE 6.2: Identification of Colligated Pseudonyms

6.3.5 Identification

The Identification protocol (Figure 6.2) is based on the Pointcheval optimised identification scheme [166] of Girault's identification scheme [78], but it now also includes the DL-EQ $\log_g X = \log_{\mathtt{PK}_{\mathcal{T}}} Y$. In this protocol, a user $\mathcal{U}$ uses his certified pseudonym to identify himself/herself with a verifier $\mathcal{V}$ and at the end of the protocol the verifier obtains an undeniable proof of $\mathcal{U}$ participation in the protocol. The identification process is represented by

$$\langle \mathtt{PROOF}_{\mathcal{U}_i} \rangle \leftarrow Identify(\mathcal{U}, \mathcal{V}, PK_{\mathcal{U}_i}, \mathtt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle, PK_{\mathcal{T}})$$

6.3.6 Tracing

The trace protocol (Figure 6.3) is invoked by a verifier $\mathcal{V}$ after $\mathcal{U}$ has misused a pseudonym and runs between the verifier $\mathcal{V}$ and the trustee $\mathcal{T}$. To trigger the protocol, $\mathcal{V}$ has to provide proof of protocol participation by $\mathcal{U}$. We shall express this phase as

$$(PK_{\mathcal{U}_0}) \leftarrow Trace(\mathcal{V}, \mathcal{T}, PK_{\mathcal{U}_i}, \mathtt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle, \langle \mathtt{PROOF}_{\mathcal{U}_i}\rangle)$$

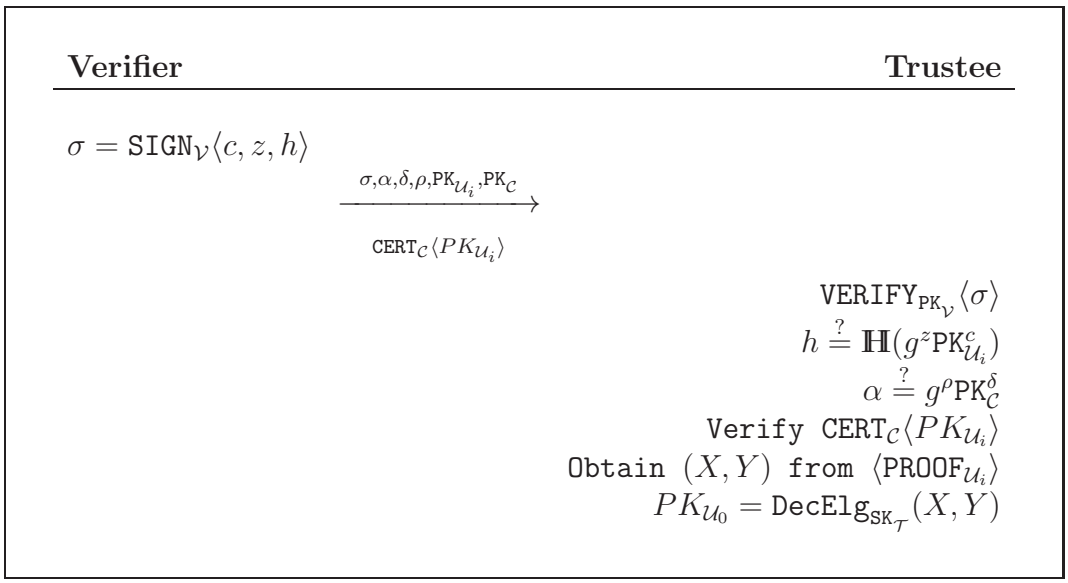

FIGURE 6.3: Tracing Protocol

6.4 Security

6.4.1 Adversary Goals

We assume an active adversary $\mathcal{A}$, who is capable of eavesdropping and injecting messages in the communication medium. We also assume that an adversary may also be a legitimate (but dishonest) participant in a protocol, that is, either the certifier or the verifier or both may be dishonest.

As in [53, 135], we want our pseudonym system to be secure against the following

attacks, that is, an adversary's goal is to mount any of following attacks:

- *Pseudonym forgery* An adversary tries to forge a pseudonym for some user, possibly in association with other participants, including the certifier. That is the attack can be either:

  1. An adversary in possession of a valid proof tuple $(PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle)$ issued to another user or for a tuple of the form $(PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{A}}\rangle)$ is successfully able to execute an identification protocol with a verifier identifying as $\mathcal{U}_i$.
  2. An adversary successfully identifying himself/herself by executing an identification protocol with a tuple of the form $(PK_{\mathcal{A}}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle)$.

- *Identity compromise* An adversary in association with other participants tries to obtain information regarding the user's master public-secret key-pair, that is, an adversary with the knowledge of all user-generated public keys ($PK_{\mathcal{U}_1}$, ..., $PK_{\mathcal{U}_l}$), it should be computationally infeasible for an adversary to obtain the master public key $PK_{\mathcal{U}_0}$.

- *Pseudonym linking and colligation* An adversary tries to obtain information that links a pair of pseudonyms to the same user or to a user's master public key. The goal is that even with the knowledge of all user generated public keys ($PK_{\mathcal{U}_1}$, ..., $PK_{\mathcal{U}_l}$), it should be computationally infeasible for an adversary to prove that any of the PK's in the set ($PK_{\mathcal{U}_1}$, ..., $PK_{\mathcal{U}_l}$), are related.

We now present our claims on the security of our proposal.

Claim 6.4.1. *If the Square Decisional Diffie-Hellman (SDDH) problem is hard, then public keys generated from the master public key are indistinguishable.*

The public keys generated are of the form $g^{SK^{2^i}}$ where, $i \in 0, \ldots, l$. For an adversary $\mathcal{A}$ to distinguish between a newly generated public key from a master or another newly-generated public key, $\mathcal{A}$ should solve the square Diffie-Hellman decision problem, that is, efficiently distinguish between two distributions of the form (g, g^{SK}, g^{SK^2}) and (g, g^{SK}, g^c), which is assumed to be hard.

Claim 6.4.2. *It is computationally infeasible to obtain the master public key of a user by an adversary even with the knowledge of all newly-generated public key.*

Proof (Sketch) For an adversary to obtain the master public key ($\mathtt{PK}_{\mathcal{U}_0}$) from a pseudonym ($\mathtt{PK}_{\mathcal{U}_i}$) presented, the adversary needs to solve, first, the discrete log problem to obtain $\mathtt{SK}_{\mathcal{U}_i}$ and then the square root problem to obtain the value i. This violates our security assumptions. It is also a well-known fact that assuming the factoring of Blum Integers is intractable, the function $f_N = \mathtt{SK}_{\mathcal{U}_0}^{2^i} \mod N_i$ is a trapdoor (one-way) permutation [80].

Claim 6.4.3. *It is computationally infeasible to obtain the master public key of a user by a verifier or a certifier even if the certifier and verifier collude.*

Proof (Sketch) Both $\mathcal{C}$ and $\mathcal{V}$ have knowledge of the public parameters. In addition, $\mathcal{C}$ has the knowledge of the user's master public key $\mathtt{PK}_{\mathcal{U}_0}$, whereas, a verifier has the knowledge of the cipher-text obtain from the El-Gamal encryption (X, Y), the pseudonym of the user $\mathtt{PK}_{\mathcal{U}_i}$, the signature value on both the pseudonym and the cipher-text $(\alpha, \rho, \delta, \mathtt{PK}_{\mathcal{C}})$.

For a dishonest certifier $\widehat{\mathcal{C}}$ and a dishonest verifier $\widehat{\mathcal{V}}$ to obtain the master public key $\mathtt{PK}_{\mathcal{U}_0}$ of a user, either independently or in collusion, any one of the following cases needs to be satisfied.

Case 1 The blind signature protocol during the certification process leaks information about the identity of the user.

Case 2 A verifier is able to deduce the master identity from the pseudonym presented during the identification protocol.

Case 3 The certifier and a verifier with their combined knowledge are able to identify the colligation that exists between a pseudonym and the master secret key.

The security of Case 1 trivially follows the proof of security of the blind signature protocol by Pointcheval in [166]. For Case 2, a verifier can obtain the master public key if the proof of DL-EQ $\log_g X = \log_{\mathtt{PK}_{\mathcal{T}}} Y$ leaks any information regarding the master

public key. A way of proving the security of the scheme is via the oracle replay technique formalised by Pointcheval and Stern [168]. In particular, the Schnorr signature with composite modulus has been proved secure in the random oracle model [21] by Poupard and Stern [169]. They show that if an adversary is able to forge a signature under an adaptively-chosen message attack, then the adversary is able to compute the discrete logarithms in G.

The security of Case 3 is based on the inability of $\widehat{\mathcal{V}}$ and $\widehat{\mathcal{C}}$ to obtain any information that links the user when they interact in the identification and the certification protocols. There are only two possibilities that can identify that a user participated in both the protocols. (a) The pseudonym leaks value about the true identity and, (b) the El-Gamal cipher-text (X, Y) that is used in both the certification and identification protocols can be linked to $\mathtt{PK}_{\mathcal{U}_0}$. If $\widehat{\mathcal{C}}$ and $\widehat{\mathcal{V}}$ in collusion are able to identify that the same (X,Y) that was presented in the identification protocol was used in the certification protocol, then they can positively establish a connection between the pseudonym presented during the identification protocol with the master identity used in the certification protocol.

From Theorem 6.4.2, we can conclude that it is computationally infeasible for $\widehat{\mathcal{V}}$ or $\widehat{\mathcal{C}}$ to obtain the master identity from a given pseudonym. As for possibility 2, the hash value δ computed with the cipher-text (X, Y), the pseudonym $\mathtt{PK}_{\mathcal{U}_i}$ and the value α as inputs, is blindly signed and never revealed to the certifier.

Claim 6.4.4. *If the El-Gamal encryption is secure, then only the corresponding trustee can obtain information about the user from the encrypted cipher-text.*

The proof of this theorem directly follows the proof in [74]. The authors showed that the security of the composite El-Gamal reduces to computing the quadratic residue over a composite modulus that is a product of two primes. And because the master public key is encrypted using the pubic key of the trustee, only the trustee can successfully decrypt the cipher-text.

Remarks. The protocol also provides guarantees of the honest participation of a user. The cipher-text containing the master public key is signed (blindly) by the certifier. A

verifier computes the hash value of the cipher-text (X, Y), the pseudonym and α again to verify against the signed hash value in the blind certificate, thus confirming that the user has performed an El-Gamal encryption over the same values that were used during the certification process.

6.5 TPM Integration

In this section, we present a brief summary of how the protocols can be applied in a TPM-based setting. Because of the tamper-resistant protection offered to the master secret key, the TPM is an ideal candidate; however the protocols can also be applied to other structures, such as, directory based services (for example, active directory, LDAP).

In a TPM-based setting, the endorsement key (EK) within a TPM will be of the form $(\mathtt{PK}_{\mathcal{U}_0},\mathtt{SK}_{\mathcal{U}_0})$. The EK is certified by the manufacturer and embedded into the TPM. A user, wishing to obtain services from an application software on a machine generates a pseudonym of the form $(\mathtt{PK}_{\mathcal{U}_i},\mathtt{SK}_{\mathcal{U}_i})$ as described in Section 6.3.3. The application software and the TPM then perform an identification protocol as in Section 6.3.5. At the end of the identification protocol, the application software is provided with a guarantee of the identity of the user and the associated TPM, but the system still protects the identity of both the TPM and the user associated with it.

6.6 Summary

In this chapter, we presented a pseudonym system by using the property of preserving a high-value secret key. Such systems are ideally suited for active-hardware security modules because they can protect the high-value secret keys and provide a high level of assurance for its security to the end users. The proposed system not only provides restricted anonymity, but also supports colligation between a trusted *high-value* secret key and the generated pseudonyms, therefore protecting the privacy of the user associated with the hardware module. Compared to other pseudonym schemes, our scheme

has an efficient identification protocol where the computation can be carried out on a device that is constrained in processing power. As opposed to other TPM-based schemes [33, 188] that require the computation to be distributed between the TPM and the host computer, the computations in our scheme can be performed on the module itself. Our scheme is also ideally suited for storage-constrained devices, because there are no new secret keys to be generated for each pseudonym, only counter values of the pseudonym. Thus there is no appreciable increase in the storage requirement even when the number of pseudonyms required is high. Finally, in terms of anonymity, in our proposal, not only the applications on a single computer can be associated with a different pseudonym, but also every web-based application used by a user can be associated with a pseudonym.

7

Integrating Hardware Security Modules with Application-Level Security Protocols

In this chapter, we evaluate active hardware security modules applied to an application level security protocol and, towards this end, we consider an electronic commerce application, electronic tendering. We start first by providing an overview of the main components in an e-tendering system and define its security requirements. We then describe our proposal for a secure e-tendering system that satisfies our security requirements previously identified.

Contrary to the research work presented in the previous chapter, where we proposed a pseudonym system based on composite modules, the research work presented here uses a prime modulus to achieve anonymity through pseudonyms. Although, it is possible with some little modifications to extend and apply our previous work to provide

a secure electronic tendering system, the aim here is to propose a solution using protocol constructions based on the prime modulus. This, in turn demonstrates the versatility of the hardware-based security solutions, because they can be independent of the security protocols that employ them. In this research work, we primarily resolve the issue of providing anonymity for an e-tender submission and achieving fairness in an e-tendering system. Our proposed protocol can also be directly applied using software-based solutions even without HSMs, but as identified in previous chapters, the use of the HSMs provides an additional layer of security, because they help to tie a system together with its users to the physical world.

7.1 Electronic Procurement

Procurement is the acquisition of works, supplies or services by public bodies, and tendering is considered one of the fairest means of awarding contracts to obtain such services. Electronic procurement has received considerable attention from governments [97, 126, 148, 170], because of the obvious cost savings that are obtained by the automating tendering and payment processes with electronic tools. Although this interest from governments has led to the development of various commercial and non-commercial e-tendering systems around the world, only parts of the e-tendering process have been successfully deployed. John Barnard [90] refers to this discrepancy in the use of the e-tendering scheme and observes that although more than 75% of tenders are electronically advertised, less than 40% provide the electronic documentation required by the tender process and less than 20% make electronic tender submissions.

In part, this may be explained by the concerns regarding security and the availability of the resources to help with the e-tender submission and review. Many e-tendering security concerns are similar to other electronic commerce systems and they normally relate to inadequate guarantees for confidentiality, authentication and non-repudiation. However, the prime security issue that has been the main obstacle in the wide adoption of e-tendering is the lack of fairness in the e-tendering process. A secure e-tendering

solution should support both fairness and transparency to guarantee tenderers and enable them to view the progress of their submission processing. It is also important that when disputes arise an e-tendering system should be able to provide a full history of the events leading up to contract award that can be publicly verified without compromising confidentiality or privacy.

7.1.1 Related Work

Most studies related to e-procurement have mainly been in the field of electronic contracting. Angelov [7] reviews many frameworks for the B2B e-contracting and Boulmakoul and Sall [29] present an integrated contract management system, pointing out some of the security properties that must be provided to protect legally-binding documents, such as, e-tenders submissions and contracts. These studies do not consider the security issues that are unique to e-tendering. The evident gap in the literature prompted Du *et al.* [61, 62] and Betts *et al.* [24] to define some of the security requirements for an e-tendering system and later to propose a submission protocol [60]. Betts *et al.* [24] also pointed out various security and legal issues that should be resolved when designing an e-tendering system. Though, these studies examine some important security requirements that new and existing e-tendering systems should satisfy, they do not redress the issues concerning the fairness and transparency of the e-tendering process. An essential ingredient in providing fairness is the anonymity of an e-tender submission, because anonymity guarantees that all submitted tenders would be treated in the same unbiased way. Also, because of the legal status of awarded tenders, it is essential for an e-tendering system to provide good auditing and public verification of the tender award process that also meets the evidentiary requirements in courts of law.

The current e-tendering systems attempt to mirror the traditional tendering system. The main parties in an e-tendering system are the principal and the tenderers. The principal advertises tender requests and accepts submissions from tenderers. On receiving the submissions, the principal performs tender evaluations and selects the winning tender. Many of the current e-tendering systems have been implemented on the assumption that tendering systems are similar to auction systems. In the next

section, we first highlight the main differences between an auction and the tendering systems, we then provide an overview of a generic e-tendering system and in the remainder of the section we summarise the main security goals that should be satisfied when designing an e-tendering system.

7.1.2 E-tendering vs. Auction Systems

Though tendering systems do share some of the properties of auction systems, there are some security considerations that are different. In this section we highlight those differences and identify the weaknesses when an auction system is deployed as an e-tendering system. We also evaluate selected seal-bid auction protocols proposed in the literature, because they also enforce the privacy of the competitor bids.

There are a variety of auction systems such as, *English, Vickery, Sealed-Bid, Dutch, Sealed-Double* and each system has distinctive goals and employ decision strategies depending on its own rules. In a traditional auction system, the auctioneer sells the product to a bidder who has placed the highest bid value. Except for sealed-bid auction systems, the biding value generally is not confidential; on the contrary, it is made public to receive the highest possible bid. This is fundamentally different to an e-tendering scheme where the tender value should remain secret from all other tenderers. In a traditional auction system, before the auction closing time the auctioneer opens the bid values, whereas in a tendering system it is important that the principal does not know any tender values before the tender submission deadline. If this security consideration is not taken into account, the tendering system is vulnerable to collusion between the principal and its favourite tenderer.

A seal-bid auction system also shares some security properties that are applicable, even for an e-tendering system. Particularly, in both an e-tendering system and a seal-bid auction system, there is a need to prevent other system participants accessing a tender (bid) submission. Franklin *et al.* [73] present a sealed-bid auction system based on the threshold secret sharing of the bidding price using verifiable signatures to provide non-repudiation. However, their proposal does not protect the privacy of the losers and the losing bids. To preserve fairness in an e-tendering system it is essential

that the privacy of even the losing tenderers is also preserved. Naor *et al.* [147] and Juels *et al.* [112] propose an auction system with a proxy-oblivious transfer. However, the scheme is not publicly verifiable; therefore, such an auction system when applied to e-tendering, compromises the guarantees provided to the tenderers. It is essential that transcripts generated in an e-tendering system are publicly verifiable without compromising the privacy of the tender submissions or the tenderers, because this provides confidence to all parties involved that the e-tendering process is being carried out in a fair and secure manner. Cachin [37] proposes an auction system using homomorphic encryption with a hiding assumption and an oblivious third party. However, this scheme cannot reveal the winning price but only identifies the winner, and thus their system is vulnerable to bidder repudiation. Sakurai and Miyazaki [175] propose a simple and elegant seal-bid auction using an undeniable signature scheme, where the bidder controls the confidentiality of the bid. Unfortunately, their scheme requires every bidder to be on-line during the auction process. This clearly not feasible in an e-tendering system, because the tendering process can range over a few days to many months. Furthermore, the communicational complexity of their system increases with the number of participants, which is not a desirable property for e-tendering.

7.1.3 A Generic E-Tendering System

A generic e-tendering system consists of three phases and normally consists of the six steps identified in Figure 7.1.

Phase One (**Registration and Tender Invitation**) This phase consists of the principal distributing tender documents and is typically carried out via a secure web-based system. It includes the registration of the tenderer and electronic notifications sent to tenderers about updates and queries. The first two steps (Step 1 and 2) in Figure 7.1 refer to a tenderer's registration. In the case of government-based e-tendering the tenderers are generally pre-selected and invited to participate. Therefore, the registration process also acts as the confirmation of a tenderer's intent to participate in the tendering process.

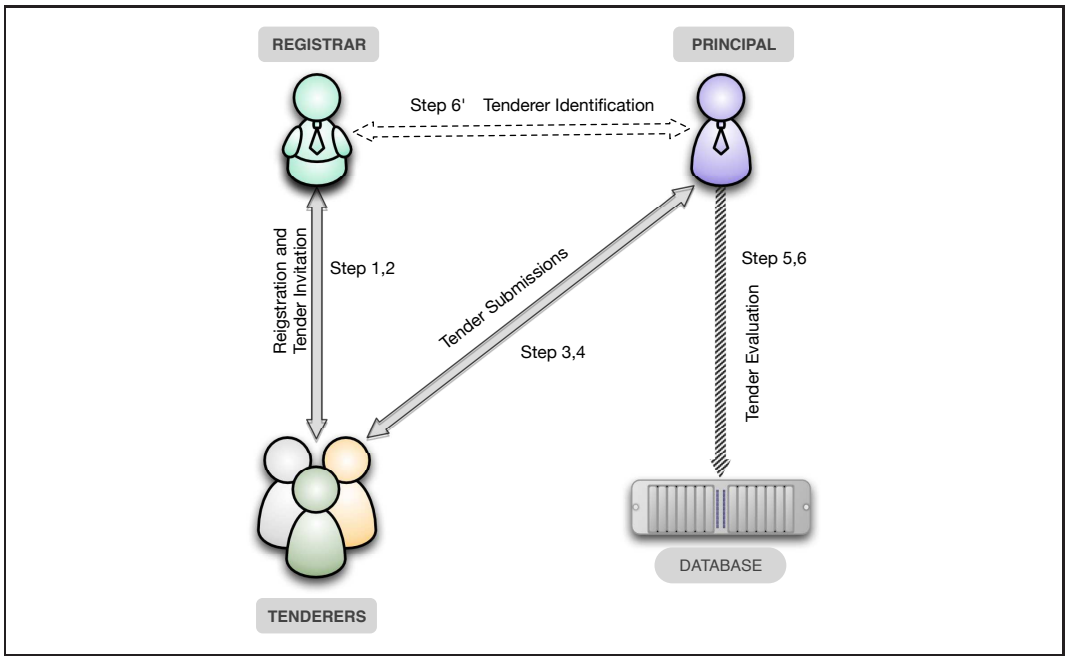

Figure 7.1: A Generic E-Tendering System

Phase Two (**Tender Submissions**) This phase involves all registered tenderers submitting their tenders for evaluation (Steps 3 and 4). The submissions are received by the principal and are stored in a secure database (Step 5). All tenders received should be sealed and opened only after the tender submission deadline has passed. If this security measure is not implemented, then the principal will be able to collude with his favourite tenderer, because the favourite tenderer will be able to resubmit the tender after knowing the competitor's tenders. This obviously guarantees that the favourite tenderer always wins.

Phase Three (**Tender Evaluation**) When the tender submission deadline has elapsed, the principal retrieves and reveals received tenders (Step 6) and based on the project's evaluation criteria, awards the project to the best tender.

In Figure 7.1, Step 6′ refers to the trace operation in our proposal. In our anonymous e-tendering system, all tenderers use anonymous credentials (psuedonymns) obtained from the registrar when submitting their tenders. When a principal has made the final decision on a winning tender, the principal requests the registrar to identify the winner.

7.1.4 Security Requirements for E-Tendering

Du *et al.* [63, 64] provide a list of security requirements for an e-tendering system and, as identified previously, their system does not consider the requirement of fairness. In this section, we list the security goals that an e-tendering system should achieve. Before that, we first provide a definition of fairness for an e-tendering system.

Definition 7.1.1. *An e-tendering system is fair, if and only if:*

1. *It is impossible for a principal to obtain any information about a submitted tender before the tender submission deadline or obtain the true identity of a tenderer without the participation of either the tenderer or the registrar.*

2. *It is impossible for a corrupt participant to obtain (or issue) a valid tender or to prohibit a honest participant from obtaining a valid contract.*

Similar to other electronic commerce systems such as, e-payments and e-auctions, an e-tendering is required to provide generic security requirements such as, confidentiality, integrity, authentication and non-repudiation. Below, we identify some essential security goals for an e-tendering system, both from the perspective of the principal and the tenderers.

Generic Security Requirements

1. *Confidentiality and Integrity of Tender Documents* Confidentiality and integrity are the primary security requirements in providing a fair e-tendering process. Because tendering is carried out over insecure networks, the e-tendering system should provide communication security that protects the information that is sent between all participants. This is generally achieved by using a strong encryption. It is also essential that an e-tendering system provide strong storage security, because submissions are stored in a database.

Principal Requirements

2. *Authentication of Participants* When the principal invites a tender submission,

the principal must be in possession of an unforgeable *proof* that only legitimate and authorised tenderers are participating in the tendering process. This proof must not be 're-playable' and should be valid only for a single instance of the protocol, that is, it should not be usable as proof in any other tendering protocol.

3. *Proof of Tender Submission Binding and Non-repudiation* Our e-tendering system makes use of anonymous credentials. Therefore, on receiving a tender submission, the principal should be in possession of an undeniable *proof* that indicates that there is a valid and strong connection between the submitted tender and the real identity of the tenderer. Because of the legal nature of the tendering process, the proof should also be admissible in a court of law for a tenderer's participation.

Tenderer Requirements

4. *Proof of Tender Submission Acceptance* A tenderer receives an undeniable *proof* of the submission acceptance from the principal. The tenderer is also provided with a guarantee that the submitted tender will be processed during evaluation by the principal.

5. *Guarantees on Fair Processing*

   - *Proof of Tender Submission Hiding* A tenderer receives a *proof* of security for the submitted tender. The proof should convince a tenderer that no participant (such as, other tenderers and the principal) can obtain or reveal any information about the tenderer's submission.
   - *Anonymity* It should be infeasible for any participants in the protocol, including the principal, to obtain the true identity of a honest tenderer without the collaboration of the registrar.

7.2 A Secure E-Tendering System

Our e-tendering system consists of three phases:

1. Tenderer Registration

2. Tender Submission

3. Wining Tenderer Trace

The aim of the system is that when a principal announces the winning tender, every participant (including the principal) is convinced that the tendering process was carried out in a fair and transparent manner. In this section we first describe our protocol. For this purpose, we combine the techniques of *offline e-cash* [72] by Frankel *et al.* and add *signed commitment* to the tenders, using ideas from Pederson [161].

7.2.1 System Setting

The system consists of a principal $\mathcal{P}$, tenderers $\mathcal{T}_i$ (where the index i runs from $1, \ldots, n$) and a trusted third-party called the registrar $\mathcal{R}$. A suitable prime order subgroup, G of $\mathbb{Z}_p^*$, of order q is chosen, such that $p = 2q + 1$ is a large prime and where the discrete logarithm problem is intractable. Suitable generators g, g_1 and g_2 are chosen such that $\log_g g_1$ is not known to any entity. A cryptographically strong hash function $\mathbb{H} : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$ is chosen and the tuple $(p, q, g, g_1, \mathbb{H})$ is published.

Each entity generates its public-secret key pair and the keys are stored within the hardware security module during initialisation, either by the manufacturer or by the entity itself. We use the following notation to represent an entity's public and secret keys. The public key of the principal $\mathcal{P}$ is, $\mathtt{PK}_{\mathcal{P}} = g^{\mathtt{SK}_{\mathcal{P}}}$, where, $\mathtt{SK}_{\mathcal{P}}$ is the corresponding private key. The public keys of the registrar $\mathcal{R}$ is, $\mathtt{PK}_{\mathcal{R}} = g^{\mathtt{SK}_{\mathcal{R}}}$, $\mathtt{PK}'_{\mathcal{R}} = g_1^{\mathtt{SK}_{\mathcal{R}}}$, and $\mathtt{PK}''_{\mathcal{R}} = g_2^{\mathtt{SK}_{\mathcal{R}}}$ and the public key of a tenderer $\mathcal{T}_i$ is, $\mathtt{PK}_{\mathcal{T}_i} = g^{\mathtt{SK}_{\mathcal{T}_i}}$.

Public keys for all entities stored in the hardware modules are certified by their manufacturers. Also, before every tendering process, the tenderer computes $z\prime = (\mathtt{PK}'_{\mathcal{R}})^{\mathtt{SK}_{\mathcal{T}_i}} \cdot \mathtt{PK}''_{\mathcal{R}}$

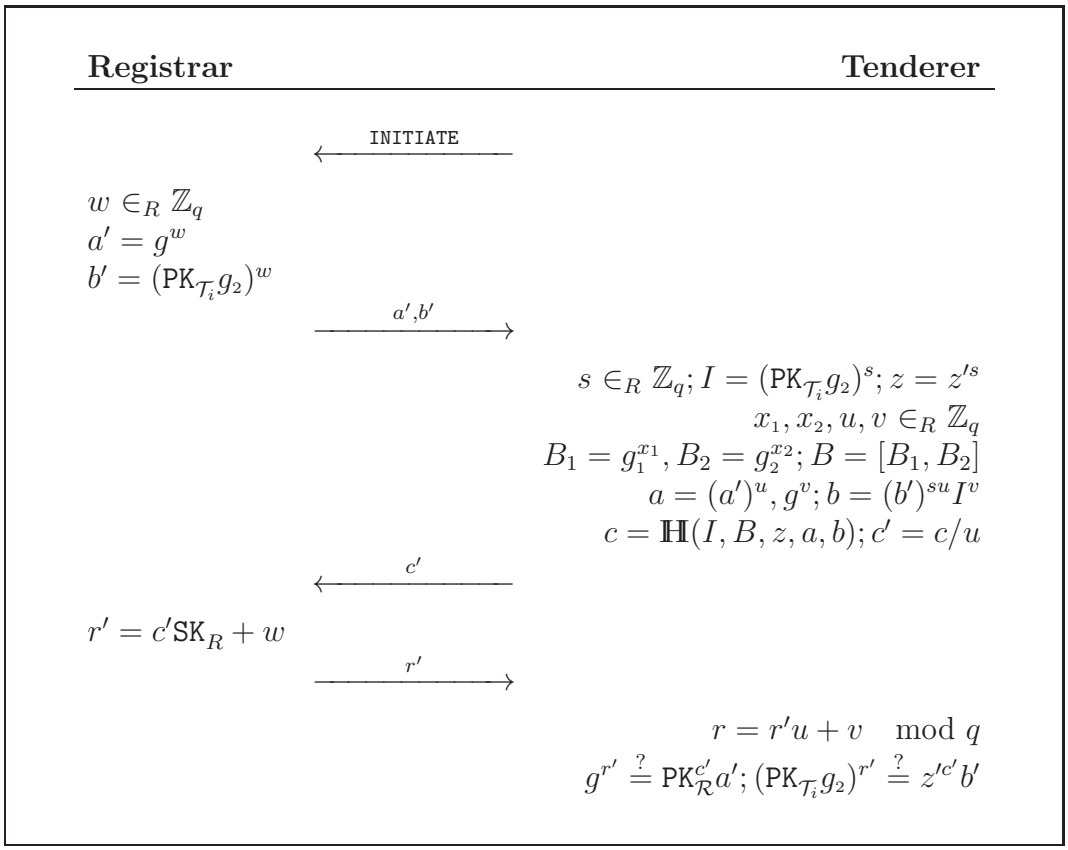

FIGURE 7.2: Registration Protocol – The signature on value $(I, B) = (z, a, b, r)$ satisfies the relations $g^r = \mathtt{PK}_{\mathcal{R}}^{\mathbf{H}(I,B,z,a,b)} a$ and $I^r = z^{\mathbf{H}(I,B,z,a,b)} b$.

7.2.2 Registration

During the registration phase, a tenderer $\mathcal{T}_i$ identifies himself/herself and presents the tuple $(\mathtt{PK}_{\mathcal{T}_i}, \mathtt{CERT}\langle \mathtt{PK}_{\mathcal{T}_i} \rangle)$ to the registrar and obtains a *restrictive blind signature* [32] for a *pseudonym*. The restrictive blind signature restricts the pseudonym to be of the form $I = (\mathtt{PK}_{\mathcal{T}_i} g_1)^s$. The value of I is formed by the tenderer and never revealed to the registrar. We can express this phase as, *'a tenderer $\mathcal{T}_i$ engages in the registration protocol with the registrar $\mathcal{R}$ using a random value s (known only to $\mathcal{T}_i$) to obtain a restrictive blind signature on the pseudonym I, signed using the registrar secret key $\mathtt{SK}_R$, but where the value of I is known only to $\mathcal{T}_i$'*. Figure 7.2 describes the steps involved in the registration protocol.

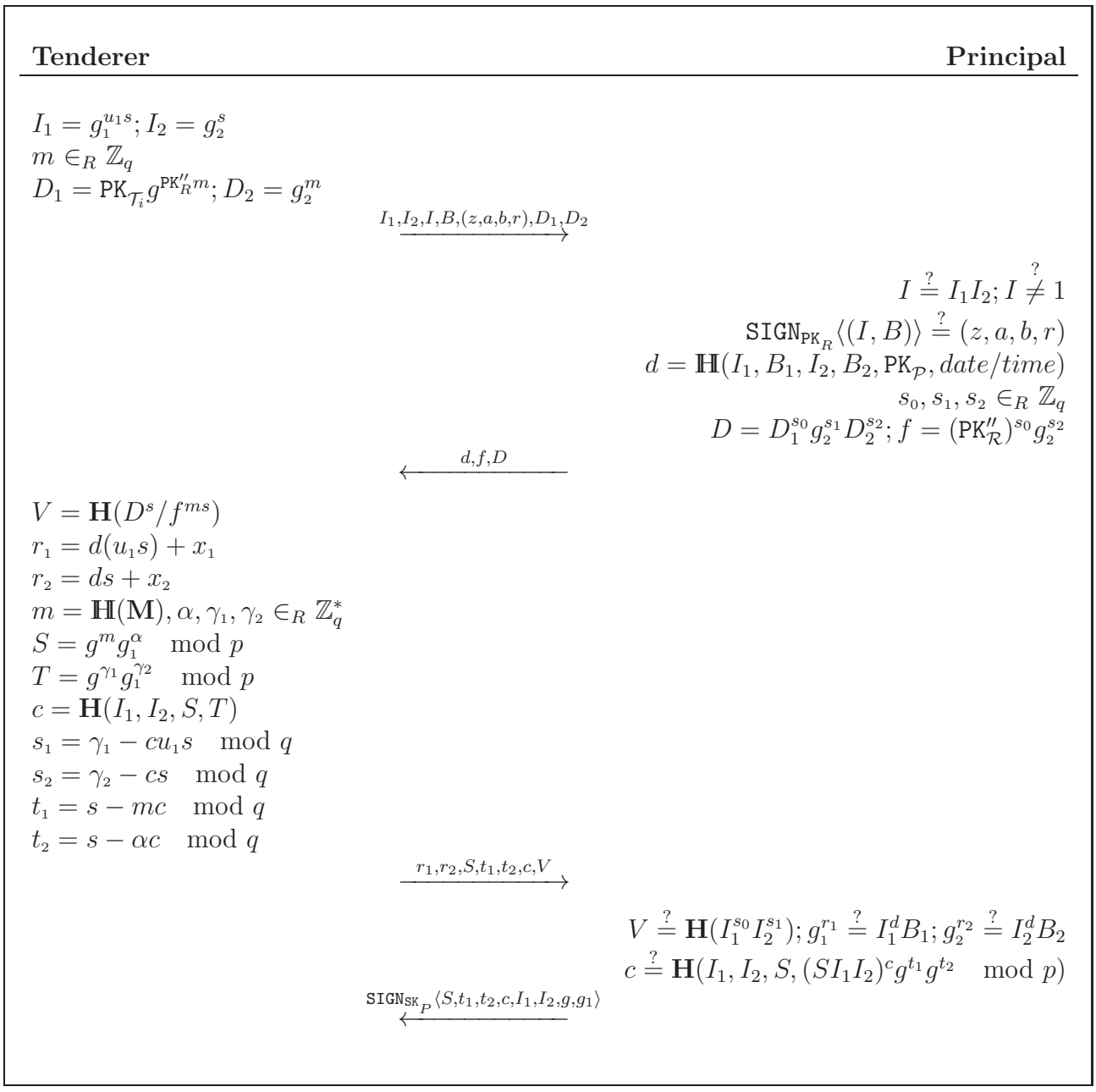

Figure 7.3: Submission Protocol - Phase One

7.2.3 Submission

The submission consists of two sub-phases. Phase One involves the tenderers making a commitment to participate in the tendering process. After the submission deadline has elapsed, Phase Two begins, during which the tenderers reveals their commitment, thus revealing their tender details.

Phase One: During this phase the tenderers identify themselves to the principal and commit on the tender details. $\mathcal{T}_i$ engages in the protocol to convince the principal of the correctness of the pseudonym I. If this phase is successful, then the protocol transcript will contain $I_1 = g_1^{u_1 s}$ and $I_2 = g_2^s$, such that $I = I_1 I_2$. $\mathcal{T}_i$ also creates a hash value m of its tender documents ($\mathbf{M}$), and commits this value (m). We shall express this phase as, '*a tenderer $\mathcal{T}_i$ engages in phase one of the submission protocol with the principal $\mathcal{P}$, using $\mathcal{R} - certified\ (I)$, secret values $(s, \mathtt{SK}_{\mathcal{T}_i})$, and the tender details $\mathbf{M}$, to generate proof transcripts that contain the encryption of $\mathcal{T}_i$'s identity under the public key of the registrar $\mathtt{PK}''_R$, and a signed commitment on $\mathbf{M}$ using the secret keys of the tenderer*'. Figure 7.3 identifies the steps involved in phase one of submission protocol.

Phase Two: This phase begins after the submission deadline has passed. The principal contacts the tenderers and requests them to provide their tenders corresponding to their commitment in Phase One. We shall express this phase as, '*the tenderer $\mathcal{T}_i$ engages in Phase Two of the submission protocol with the principal $\mathcal{P}$, by revealing the tender details and α, to obtain a proof of tender submission acceptance, signed using the secret key of the principal ($\mathtt{SK}_{\mathcal{P}}$)*'. Figure 7.4 identifies the steps involved in Phase Two of submission protocol.

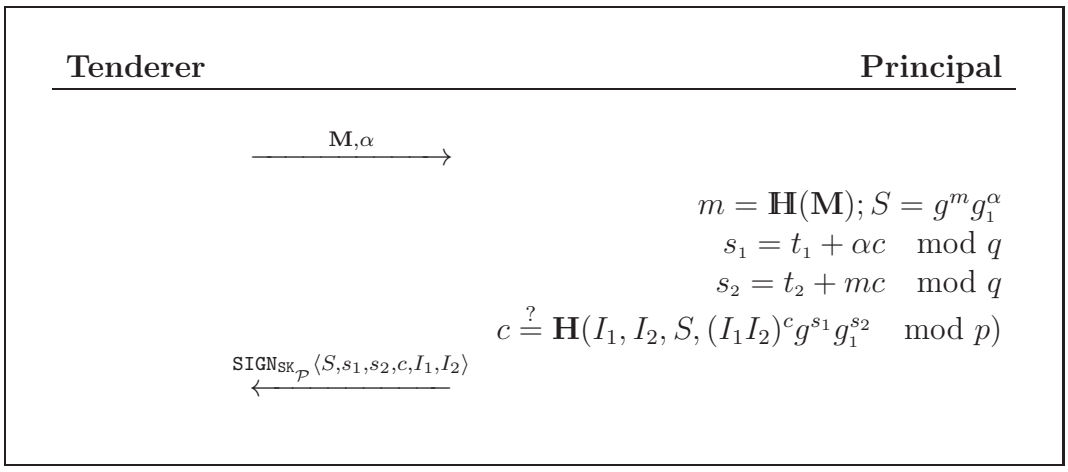

Figure 7.4: Submission Protocol – Phase Two

With those tenders that have been legally received the principal then begins the tender evaluation procedure and announces the winning tender. Note that the anonymity

of the winning tenderer *need not* be revoked, but generally, in a government procurement, the identity is made public. To do so, the principal contacts the registrar and performs the trace protocol.

7.2.4 Trace

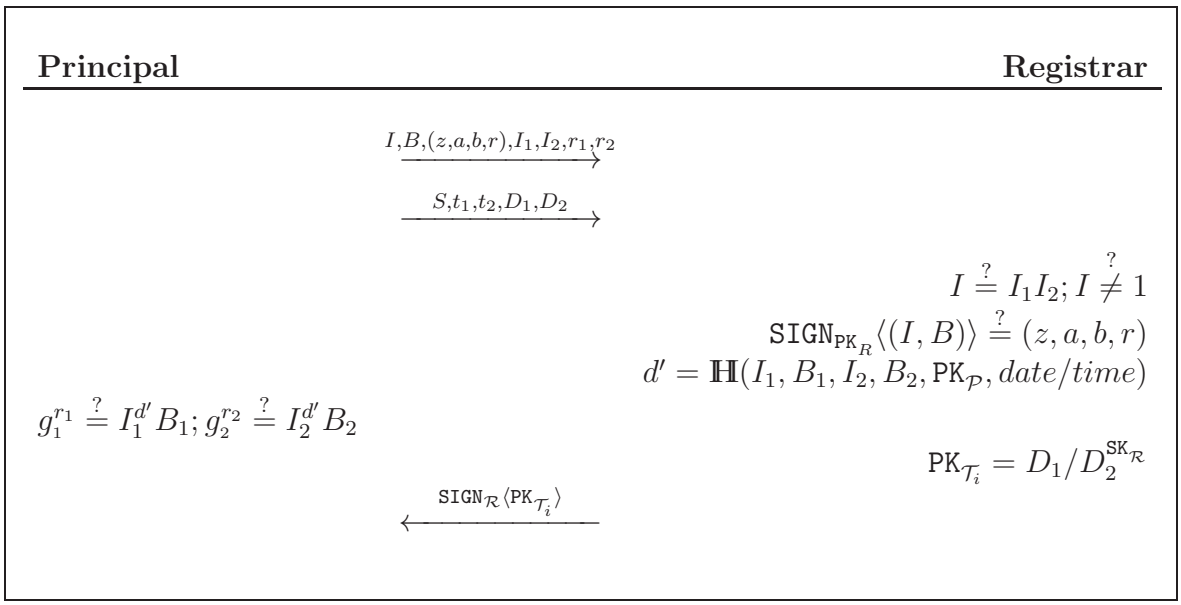

Figure 7.5: Trace Protocol

The trace protocol is invoked when the principal has announced the winning tender and wishes to trace the real identity of the winning tenderer ($\mathtt{PK}_{\mathcal{T}_i}$) that corresponds with the pseudonym I. The trace protocol may also be invoked in case of disputes (such as, no communication from the winning tenderer after the announcement of the results). We shall express this phase as, '*a principal* $\mathcal{P}$ *or any judicial authority engages in a trace protocol with* $\mathcal{R}$ *to obtain the identity* $\mathtt{PK}_{\mathcal{T}_i}$ *using* $R - certified(I)$*, and the proofs obtained during the submission protocol*'. Figure 7.5 identifies the steps involved in the trace protocol.

Remarks Two cases of disputes can occur in the e-tendering system, (a) the winning tenderer does not respond to the principal's announcement, (b) the winning tender is denied the contract. In the former case, the principal contacts the registrar and runs the trace protocol to obtain the true identity of the winning tenderer and in the second

case, the correct winning tenderer needs to contact $\mathcal{R}$ or a judicial authority by producing the signed proof obtained at the end of Phase Two of the submission protocol to identify himself/herself using the pseudonym I_1, and prove that the winning tender belongs to him/her.

7.3 Security Analysis

Theorem 7.3.1. (Fairness) *The e-tendering system described in Section 7.2 is fair.*

To prove our proposed system is fair, we have to prove mainly two things (*cf.* Definition 7.1.1):

(A) The principal is unable to obtain any information regarding any tender before the tender opening time (tender hiding) or the tenderer's details until the principal has made a decision on the tender (anonymity).

(B) A corrupt participant does not gain any advantage.

To prove this, we make use of the following theorems.

Theorem 7.3.2. (Hiding) *Given the tuple (S, T, t_1, t_2) it is infeasible to determine the value of m. Thereby, the e-tendering system hides the value of m.*

Proof. (Sketch) The commitment scheme belongs to a class of three-pass, honest verifier zero knowledge protocols. The protocol transcripts can be simulated by calculating $T = (SI_1I_2)^c g^{t_1} g^{t_2}$ after choosing S, c, t_1, t_2. Because the protocol is zero-knowledge the value of m is hidden from the principal (verifier) until the tender submission time has elapsed. □

Theorem 7.3.3. (Binding) *If the value of the tuples (S, T, c, t_1, t_2) cannot be altered, then the e-tendering system possesses the properties required for binding to the value of m.*

This theorem follows trivially from the theorem presented by Pedersen [161] (Theorem 3.1), which proves that the commitment scheme reveals no information about the

value of m and such a commitment scheme can later be opened by revealing the value of m and α.

Theorem 7.3.4. *If the discrete logarithm problem is hard, a corrupt tenderer who does not know the private keys of a honest tenderer can convince the principal with a probability of* $1/2^{|q|}$*, where* $|q|$ *is bit size of* q*.*

Proof. (Sketch) The proof follows from [190], a corrupt tenderer can cheat the principal by guessing the challenge correctly in advance and can form the correct commitment transcript (From Theorem 7.3.2). If $|q| = log_2 q$, then the number of legal challenges will be of the form $2^{|q|}$. When the principal chooses the challenges at random, the probability that a corrupt tenderer will correctly guess the challenge is $1/2^{|q|}$. □

Theorem 7.3.5. *If the El-Gamal encryption is secure and the discrete logarithm problem is intractable, then the e-tendering system preserves tenderer anonymity.*

Proof. The proof is by contradiction. Assume that the principal can trace the user, that is, given the view of the submission protocol, with the non-negligible probability η, it can compute the true identity of the tenderer. Also assume that the principal has access to a polynomial time algorithm $\mathcal{A}$ which, on input (g,y), produces an output x such that, $x = log_g\ y$. The principal (who is the attacker for the protocol) to obtain $\mathtt{PK}_{\mathcal{T}_i}$, has two options. (1) The principal obtains the secret key $\mathtt{SK}_R$ of the registrar, using the algorithm $\mathcal{A}$ with inputs $(g_2, \mathtt{PK}''_R)$ and therefore can calculate the value of D_1 and D_2, thus obtaining the public key $\mathtt{PK}_{\mathcal{T}_i}$ as in Step 2 of the tracing protocol. (2) The principal uses the algorithm $\mathcal{A}$ to solve for the values $(g, I_1), (g, I_2)$ and obtains the value of u_1 and therefore can calculates $\mathtt{PK}_{\mathcal{T}_i}$. Both of these options depend on the existence of a polynomial time algorithm $\mathcal{A}$ that can solve the discrete logarithm problem that from our assumption, is hard. Therefore, there exists no such algorithm $\mathcal{A}$ to which a principal has access to that can solve the discrete logarithm problem, and thus our e-tendering system preserves tenderer anonymity. □

Based on the above theorems, we now examine the security goals for an e-tendering system.

1. *Confidentiality and Integrity of Tender Documents* The confidentiality of the tender documents is provided by the hiding property until the tender submission closing time. Because, with an overwhelming probability, only the tenderer can open the commitment values correctly, the scheme provides a tenderer-controlled confidentiality. Our current proposal does not cover database security, because it is outside the scope of this research work. However, standard security techniques should be employed to protect the contents of databases used.

2. *Authentication of Participants* Each tenderer is required to register with the registrar and obtain a credential to participate in the e-tendering process. Thus every tenderer is authenticated and the tenderer's credentials are bound to the tender being submitted.

3. *Proof of Tender Submission Binding and Non-repudiation* Non-repudiation is provided by the hiding property and tender binding is provided by the *non-transferability property* of the *e-cash scheme* that is dependent on the tenderer. To transfer the credentials of a corresponding tender to another entity, the tenderers would need to reveal their secret keys and the value of s. Thus a sealed tender is bound to the real identity of the tenderer.

4. *Proof of Tender Submission Acceptance* Proof transcripts generated by all protocols are publicly verifiable. At the end of Phase One of the submission protocol, the commitment produced by every tenderer is digitally-signed by the principal before being sent back to a tenderer. A tenderer or any dispute resolution authority can easily verify the signed commitment response with the original tenderer-signed commitment tuple to identify whether a principal acted dishonestly.

5. *Guarantee on Fair Processing* This directly follows Theorem 7.3.1. The proposed e-tendering system provides a strong submission hiding and anonymity for all tenderers.

7.4 Summary

Electronic procurement has seen a tremendous growth in recent years and thus there is a need for a secure and fair system when awarding contracts. E-tendering has the potential to deliver such a system in a convenient and transparent manner with the integration of hardware security modules in such systems providing confidence for the participants and creating a high degree of trust in the process.

The goal of any electronic system attempting to achieve what is traditionally carried out in the bricks-and-mortar world should be not only to replicate the requirements of the traditional system, but also to improve the system and to provide better services. We have proposed an e-tendering system that achieves such a goal. We identified that the e-tendering systems previously proposed did not adequately provide for the need for fairness and privacy. In resolving those weaknesses, our proposal provides a publicly-verifiable fair e-tendering system that not only meets all the security requirement of the traditional tendering system, but also offers new services, such as anonymity, tendering hiding and binding.

8

Conclusion

Hardware-based security is continuously emerging and being deployed in many new applications. While hardware devices become more integrated into our everyday lives, and range from identity documents, cars, books, packaging materials and medical supplies, to developing strong security mechanisms that protect information the stored and the information collected from such devices become vital. This thesis resolves the two main goals that were stated at the beginning, (1) the analysis of the existing security mechanisms in hardware-based security devices and, (2) developing security mechanisms to achieve the security goals distinct to each class of the hardware-based security devices, the passive, semi-passive and active devices. In this chapter, we summarise the contributions made in each class of the hardware-based security devices and discuss some open problems.

In Chapter 3, we dicussed the issue of secure identification and authentication for

passive devices and, towards this end, we considered the EPC Gen2 [66]-compliant RFIDs employed towards product identification and supply chain management. We discussed and identified the weaknesses in the EPC Gen2 and showed how it is vulnerable to various attacks. Many proposals in the literature, while attempting to develop a secure identification scheme, have failed to develop a secure protocol that is complaint with the EPC Gen2. To close this gap, we proposed a new mutual authentication protocol and analysed its security.

In Chapters 4 and 5, we resolved the security issues in the hardware-based security devices employed in identity documents and considered the application of such devices for securing ePassports. Our formal analysis on the first-generation ePassport not only confirmed the security issues that were previously identified, but also revealed new security issues, such as, an ePassport being subject to replay and grand-master chess attacks that can be exploited in cases where problems with facial biometrics exist. We also proved that this weakness further affects the security goals for the active authentication protocols, mutual authentication, key freshness and key integrity, and makes ePassports vulnerable to certificate manipulation.

We then presented our security analysis of the second-generation of ePassports that were hoping to fix the weaknesses in the first-generation. Our analysis showed that the EAC proposal fails to provide adequate security and introduces new security weaknesses and implementation issues of its own. The second-generation ePassport proposal requires extensive modifications to the existing PKI and also relies on the vulnerable first-generation security mechanisms to protect the biometric identifiers. We later showed that this implementation makes an ePassport bearer vulnerable to identity theft and covert surveillance. To resolve the security issues associated with both first- and second-generation of ePassports, we proposed a new on-line ePassport protocol. Our proposed protocol offers protection for the ePassport details from both malicious and unauthorised readers, thereby protecting against unintentional disclosures and reducing the threat of identity theft. The proposed protocol also eliminates the security related weaknesses introduced in the second-generation ePassports by removing the need for certificate chain verification and uses the well established PKI

infrastructure that is in place for the first-generation ePassport standard.

In Chapters 6 and 7, we examined the security in active devices and, in particular, analysed the privacy issues that arise when employing such devices. Tamper-resistant hardware devices protect the data stored within them and provide a high level of assurance for its security to end-users. Using this security feature and based on the property of preserving a high-value secret key, we presented a pseudonym system for active hardware devices. Our proposal solved the privacy issues that hampered the development of active hardware-based security solutions by providing restricted anonymity and supporting colligation between a trusted high-value secret key and the generated pseudonyms. In addition to proposing a pseudonym based on composite modulus, we also proposed a system based on prime modules that was employed in developing a fair electronic tendering application.

8.1 Open Problems

Although, our results in Chapter 3 resolves the security issues present in the EPC Gen2 and yield us an efficient protocol for a resource restrictive passive tag, it should be noted that the proposed protocol requires a high performance back-end server. The back-end server stores tag-specific information in a database for each tag being initialised. The back-end server also interconnects various tag readers and, during a protocol run, it is responsible for obtaining tag-specific information from its database. To provide tag security and privacy, our proposed system employs a non-static identifier (pseudonym); thus, there would be no identifiable *primary key* within the database that would match the information sent by the tag/reader during a protocol run. Consequently, the back-end server is required to search its entire database to find the tag information; this would be computationally demanding. Though search functions can be optimised, it still requires a high level of computation to be performed for every tag that needs to be identified. In this respect, an interesting open problem is to develop a more computationally efficient protocol (from a server perspective), but still maintain the efficiency obtained by our proposal with respect to tags and readers. In a broader view,

another hard and open problem would be to improve the RNG generation in the EPC Gen2-complaint devices. The EPC Gen2 supports only a very basic RNG (16-bit) to ensure security goals such as confidentiality, integrity and authentication. This makes the EPC Gen2-compliant protocol potentially vulnerable even to brute force attacks that can exhaustively search for the keying material.

Even though one of the main advantages of our proposed online ePassport solution in Chapter 5 is its online nature, it is also its weakness. The proposed online protocol would be ideally suited for ePassport identification in most developed and developing countries where there are sufficient infrastructure resources in place to support the communication requirements. The protocol may not be ideally suited for underdeveloped entry points where the communication infrastructure would be lacking The identification of the ePassports would have to rely on the traditional non-electronic means that we are intending to replace. Also, the proposal would not scale well when implemented with other e-document identification system that do not have a fixed infrastructure, such as, a roadside driver licence identification. An interesting and open problem would be to design a protocol that does not rely on online communication. Such solutions invariably tend to have the security issues present in previous designs, such as certificate manipulation and problems relating to how the certificates would be verified.

The proposed pseudonym system for active hardware devices proposed in Chapter 6 offers a new method for extending hardware-based authentication. Unfortunately, it currently offers only one level of colligation between the root secret key and the pseudonyms generated. An interesting open problem would be develop a solution that would be able to bind all pseudonyms that would form a multi-level chain, colligated under the same root secret key. Such a system would provide greater application and security benefits when used in active devices, such as the TPM. For example, currently, the root key can be used to generate and certify pseudonyms (*directly*), for either the users or the applications on the machine. If the system supported chaining, it would provide the users bound to the active device with the ability to generate and certify pseudonyms for every application they use (both web and system applications)

themselves, but would still convince a verifier about the underlying existing link to the trusted root key stored in the module.

In terms of using hardware devices for identification, an interesting and open problem is related to identity transfer. Though, this may not be necessary for most passive or semi-passive devices, it is essential that active devices support identity transfer. Compared to passive devices, active devices are typically expensive with a longer life span and therefore could be subject to multiple owners during their life cycle. Also, it would be important for previous owners to migrate their information cleanly and safely when they dispose of the old and, may be, acquire a new device. In this respect, an interesting open question would be, how to develop a security system that relies on using hardware devices for identification, when their owners need to transfer credentials and any trust associations the owners would have acquired based on the underlying hardware device.

A

Verification using Casper and FDR

Appendix A describes the verification process for security protocols using Casper and the Failure Divergence's Refinement (FDR). Casper, developed by Lowe [75], is a compiler that converts a high-level notation of the protocol to a Communicating Sequential Processes (CSP) [92] script. The CSP script can then be run on a model checker such as the FDR [134], to verify if the protocol achieves the specific security goals.

A.1 Modelling protocols in Casper

Each agent (users, trusted third parties, CA's) and intruders who can interact in a protocol are modelled as a CSP process. The resulting system is tested against specifications representing desired security goals. The FDR searches the state space to investigate whether any insecure state (sequence of messages) can occur. If the FDR

finds a specification that cannot be met, then it returns a trace of the system that does not satisfy the specification. This trace corresponds to an attack upon the protocol.

The modelling of the CSP description of the protocol is time-consuming and error-prone. The aim of Casper is to simplify this process by allowing the user to specify the protocol at an abstract level. This script, when compiled in Casper, outputs a CSP script that is then run through the model checker software FDR2[1].

The Casper script is divided into two distinct parts, a definition of the way the protocol operates and a definition of the actual system to be checked. Each part further consists of four sections specifying the variables, processes, protocol description, specification, actual variable, functions, system, and the intruder.

The first part can be considered as a function that returns a model of a system running the protocols and contains free-variables, processes, protocol descriptions and a specifications section. The Casper script used to describe the various sections below is a simplified representation of the *i*kP [18] payment protocol.

The *free-variable section* defines the types of variables and functions used in the protocol that also includes key inverses for public keys and session keys, nonces and hash functions. Example:

```
#Free variables
B , S , A : Agent
CCA : Server
pkCCA, pkB, pkS, pkA : PublicKey
saltb, nonces, rb : Nonce
ban : BuyerAuthNumber
expiry : BanExpiry
pin : Pin
ids : SellerId
tids : TransactionId
aprice : AuthPrice
desc : TransDesc
```

[1]FDR2 software is a refinement checker developed by Formal Systems (http://www.fsel.com/)

```
response : ServerResponse
PK : Agent -> PublicKey
SK : Agent -> SecretKey
PKS : Server -> ServerPublicKey
SKS : Server -> ServerSecretKey
InverseKeys = (PK, SK), (PKS, SKS)
h : HashFunction
```

The *processes section* defines information about the agents taking part in the protocol, their roles in a protocol and their initial knowledge. Example:

```
#Processes
INITIATOR(B, A, rb, ban, saltb, expiry, pin, aprice, desc)
        knows PK, SK(B), h, PKS(CCA)
RESPONDER(S, A, nonces, ids, tids, desc, aprice)
        knows PK, SK(S), h, PKS(CCA)
AUTHSERVER(A, B, S, response) knows PK, SK(A), h, PKS(CCA)
SERVER(CCA, B, S, A ) knows PK, PKS(CCA), SKS(CCA), h
```

The *Protocol description section* defines the protocol and the sequence of messages that constitute the protocol. The protocol message can also include the assignment and test expression and encrypted message. Example:

```
#Protocol description
0.   -> B : S
-- CCA issues certificate to all participants
1. CCA -> B : {B, PK(B)}{SKS(CCA)} % certB
1a. CCA -> S : {S, PK(S)}{SKS(CCA)} % certS
1b. CCA -> A : {A, PK(A)}{SKS(CCA)} % certA
-- Protocol
2. B -> S : saltb, h(rb, ban) % hban
3. S -> B : {ids, tids, nonces }{SK(S)},
        certS % {S, PK(S) % pkS }{SKS(CCA)}
```

```
3a. S -> B : {h( aprice, ids, tids, nonces, hban% h(rb, ban),
       h(saltb, desc) ) }{SK(S)}
4. B -> S : certB % {B, PK(B) % pkB}{SKS(CCA)}
4a. B -> S : { { aprice, ban, rb, expiry,
        h( aprice, ids, tids, nonces, h(rb, ban), h(saltb, desc)
        )}{PK(A)} % encslip }{SK(B)} % signenc
5. S -> A : {ids, tids, nonces}{SK(S)}, certS % {S , PK(S) % pkS }
        {SKS(CCA)}
5a. S -> A : {h(saltb, desc), saltb, desc}{SK(S)}
5b. S -> A : signenc % {encslip % { aprice, ban, rb, expiry, h(aprice,
        ids,tids,nonces,h(rb,ban),h(saltb,desc))}{PK(A)}}{SK(B)}
6. A -> S : {response}{SK(A)} % singres
7. S -> B : singres % {response}{SK(A)}
```

The *specification section* defines the requirement or security goals of the protocol. We highlight three of the ten available specifications. A full list of specifications can be found in [75].

- **Secret**(A, s, [B_1 ,...,B_n]) specifies that in any completed run, A can expect the value of the variable s to be a secret; B_1,..., B_n are the variables representing the roles with whom the secret is shared.
- **Agreement**(A, B, [v_1,...,v_n]) specifies that A is correctly authenticated to B, and the agents agree upon the values v_1,..., v_n.
- The specification **Aliveness**(A, B) means that B thinks it has successfully completed a run of the protocol with A.

Example:

```
#Specification
Secret(B, ban, [A])
Aliveness(A,S)
```

```
Aliveness(S,A)
Agreement(B,S,[aprice])
```

The second part can be thought of as defining a particular image of that function by instantiating the parameters of the protocol and contains actual variables, functions, system and an intruder section.

The *actual variable section* contains information defining the actual data types used in the protocol model. Data types defined here appear in the free-variable section. Example:

```
#Actual variables

Buyer, Seller, Mallory, AuthServer : Agent
CertAuth : Server
SALTB, NONCES, RB, NONCEM : Nonce
BAN : BuyerAuthNumber
EXPIRY : BanExpiry
PIN : Pin
IDS : SellerId
TIDS : TransactionId
APRICE : AuthPrice
DESC : TransDesc
RESPONSE : ServerResponse
```

The *function section* declared here provides more information about the function declared in the free-variable section. Example:

```
#Inline functions
symbolic PK, SK, PKS, SKS
```

The *system section* defines the agents in the system that are to be verified. The section also allows for multiple instances of the same agents to be run concurrently. The argument types declared in the system section should match the type of parameters defined in the process section. Example:

```
#System
INITIATOR(Buyer,AuthServer,RB,BAN,SALTB,EXPIRY,PIN,APRICE,DESC)
RESPONDER(Seller,AuthServer,NONCES,IDS,TIDS,DESC,APRICE)
AUTHSERVER(AuthServer, Buyer, Seller, RESPONSE)
SERVER( CertAuth , Buyer , Seller, AuthServer )
```

The intruder section specifies the intruder's identity and the initial knowledge of the intruder. Example:

```
#Intruder Information
Intruder = Mallory
IntruderKnowledge = { Buyer, Seller, AuthServer, Mallory, NONCEM,
       IDS, PKS(CertAuth), PK, SK(Mallory)}
```

A.2 Interpreting the FDR output

The FDR is a model-checking tool for state machines, with foundations in the theory of concurrency based around Hoare's Communicating Sequential Processes (CSP) [92]. The verification technique is based on the method of establishing whether a property holds by testing for the refinement of a transition system and the ability to check the determinism of a state machine that is primarily used for checking security properties. The FDR is designed to mechanise the process of carrying out refinement checks.

Casper generates refinement assertions to check for all specifications. It generates one assertion for all secret specifications and one assertion for each agreement and an aliveness specification. A CSP script file includes statements making assertions about the refinement properties. These statements will typically have the following form:

```
assert Abstract [X= Concrete
```

Example: Secret specification:

```
Secret(B, ban, [A])
```

Assertion generated:

```
SECRET_M::SECRET_SPEC[T=SECRET_M::SYSTEM_S
```

Aliveness specification:

```
Aliveness(A,S)
Aliveness(S,A)
```

Assertion generated:

```
AUTH1_M::AuthenticateSERVERToRESPONDERAliveness[T=AUTH1_M::SYSTEM_1
AUTH2_M::AuthenticateRESPONDERToSERVERAliveness[T=AUTH2_M::SYSTEM_2
```

Agreement specification:

```
Agreement(A, S, [aprice])
Agreement(B, S, [aprice])
```

Assertion generated:

```
AUTH3_M::AuthenticateSERVERToRESPONDERAgreement_aprice
        [T=AUTH3_M::SYSTEM_3
AUTH4_M::AuthenticateINITIATORToRESPONDERAgreement_aprice
        [T=AUTH4_M::SYSTEM_4
```

The selected assertion is submitted for testing by choosing the *Run* option from the *Assert* menu in FDR. The FDR then attempts to prove the conjecture by compiling, normalising and checking the refinement. When a test finishes the symbol associated with the assertion is updated to reflect the result. The symbols projected by the FDR are:

- Tick ($\surd$): Indicates that the check completed successfully, that is, the stated refinement holds.
- Cross (X): Indicates that the check completed, but the refinement does not hold. The FDR debugger is then used to explore the reasons for the failure.

- Exclamation mark (!): Indicates that the check failed to complete for some reason: either a syntax or a type error was detected in the scripts. Some resource was exhausted while trying to run the check or the check was interrupted.

- Zig-zag (*Z*): Indicates that the FDR was unable to complete a check because of a weakness in the currently-coded algorithms.

If we find a refinement is not satisfied, then there might be a weakness in the protocol. To examine the weakness, the FDR debugger is invoked. This will open a new window allowing the behaviour of the processes involved to be examined. The information presented by the debugger is represented as two parts, a hierarchical view of the structure of the process and a series of windows showing the contribution of a selected part of the process to the overall behaviour. The process structure is represented as a tree. The root node represents the process as a whole and when the leaf nodes are expanded, branches are added according to the number of sub-components in that node. For example, a node labelled with a parallel composition symbol ([|..|]) will expand to have two children representing the sub-processes which are combined in parallel. Each child is associated with its own contribution to the overall erroneous behaviour being examined.

When a node in the process structure view is selected, information about the currently selected node is displayed in the behaviour window. The information displayed depends on the nature of the counterexample being examined and the contribution made to it by the selected component. The following types of information may be displayed for each type of counterexample behaviour:

Successful refinement: no information displayed
No direct contribution: a non-erroneous trace
Refusal/acceptance failure: a non-erroneous trace, plus the illegal refusal/acceptance
Divergence: the trace leading to divergence
Divergence (internally): the trace leading to divergence, plus a trace of repeated events

The weakness in the protocol is examined by observing the trace leading to divergence.

B

Casper Representation of the ICAO First-generation ePassport Protocols

The Casper script provided below presents a combined representation of all three protocols and does not represent the modifications that are need when verifying the security properties for the individual protocols.

```
#Free variables
C,R,DS : Agent
getc : InitializeConv
lds : DataGroups
sod : SecurityObject
rndr2,rndc2,kr,kc,rndr1,rndc1 : Nonce
h : HashFunction
```

```
PK : Agent -> PublicKey
SK : Agent -> SecretKey
keyM,keyE,keyCR : SessionKey
InverseKeys = (PK,SK), (keyM,keyM), (keyE,keyE),
        (keyCR,keyCR)

#Processes
INITIATOR(R,C,getc,rndr1,rndr2,kr,keyM,keyE,keyCR)
        knows PK,SK(R)
RESPONDER(C,R,rndc1,rndc2,kc,keyM,keyE,keyCR)
        knows PK,SK(C)

#Protocol description

0.  -> C   : R
0a. DS -> C : {C,PK(C)}{SK(DS)} % CERTC
0b. DS -> R : {C,PK(C)}{SK(DS)}
1. R -> C  : getc
2. C -> R  : rndc2
3. R -> C  : {rndr2,rndc2,kr}{keyE},
        {rndr2,rndc2,kr}{keyM}
4. C -> R  : {rndr2,rndc2,kc}{keyE},
        {rndr2,rndc2,kc}{keyM}
---
5. C - > R : {LDS,SOD}{KeyCR},
        {C,PK(C)}{SK(DS)} % CERTC
---
6. R -> C  : {rndr1}{keyCR}
7. C -> R  : { {h(rndc1,rndr1), rndr1,rndc1}
        {SK(C)} }{keyCR}
```

```
#Specification
StrongSecret(C,kr,[R])
StrongSecret(C,kc,[R])
StrongSecret(R,kr,[C])
StrongSecret(R,kc,[C])
Aliveness(C,R)
Aliveness(R,C)
Agreement(C,R,[kr,kc])
StrongSecret(C,rndr1,[R])

#Actual variables
Chip,Reader,DSigner,Mallory : Agent
GETC : InitializeConv
LDS : DataGroups
SOD : SecurityObject
RNDR2,RNDC2,RNDM2,KR,KC,KM,RNDR1,RNDC1 : Nonce
KEYM,KEYE,KEYCR, KEYMM,KEYEM : SessionKey
InverseKeys = (KEYM,KEYM), (KEYE,KEYE),
        (KEYMM,KEYMM), (KEYEM,KEYEM), (KEYCR,KEYCR)

#Functions

symbolic PK,SK

#System
INITIATOR(Reader,Chip,GETC,RNDR1,RNDR2,KR,
        KEYM,KEYE,KEYCR)
RESPONDER(Chip,Reader,RNDC1,RNDC2,KC,
```

```
        KEYM,KEYE,KEYCR)
CERTAUTH(DS,C,R) knows PK,SK(DS)

#Intruder Information

Intruder = Mallory
IntruderKnowledge = {Chip,Reader,RNDM2,KM,PK,
        SK(Mallory),KEYMM,KEYEM}
```

C

CK Model

Appendix C provides a brief description of the CK model [42]. A key-exchange protocol is run in a network of inter-connected parties where each party can be activated to run an instance of the protocol called a session. A key-exchange session is a quadruple $(\mathcal{A}, \mathcal{B}, X, Y)$ where $\mathcal{A}$ is the identity of the holder of the session, $\mathcal{B}$ the peer, X the outgoing messages in the session, and Y the incoming messages. The session $(\mathcal{B}, \mathcal{A}, Y, X)$ (if it exists) is said to be matching to the session $(\mathcal{A}, \mathcal{B}, X, Y)$. Matching sessions play a fundamental role in the definition of security.

C.1 Attacker Model

The attacker is modeled to capture the realistic attack capabilities in open networks, including the control of the communication links and the access to some of the secret

information used or generated in the protocol. The attacker, denoted $\mathcal{M}$, is an active 'man-in-the-middle' adversary with full control of the communication links between the parties. $\mathcal{M}$ can intercept and modify messages sent over these links, it can delay or prevent their delivery, inject its own messages, interweave messages from different sessions, etc. (Formally, it is $\mathcal{M}$ to whom the parties hand their outgoing messages for delivery.) M also schedules all session activations and session-message delivery. In addition, to model the potential disclosure of secret information, the attacker is allowed access to secret information via session exposure attacks of three types, state-reveal queries, session-key queries, and party corruption.

State-reveal query A state-reveal query is directed at a single session while still incomplete (before outputting the session key) and its result is that the attacker learns the session state for that particular session (which may include, for example, the secret exponent of an ephemeral DH value but not the long-term private key used across all sessions at the party).

Session-key query A session-key query can be performed against an individual session after completion and the result is that the attacker learns the corresponding session key.

Party corruption Party corruption means that the attacker learns all the information in the memory of that party (including the long-term private key of the party in addition to the session states and session keys stored at the party). In addition, from the moment a party is corrupted, all its actions may be controlled by the attacker. Indeed, note that the knowledge of the private key allows the attacker to impersonate the party at will.

C.2 Session Key Security

In addition to the regular actions of the attacker $\mathcal{M}$ against a key-exchange protocol, it can perform a test session query. That is, at any time during its run, $\mathcal{M}$ is able to choose a test session among the sessions that are completed, unexpired and unexposed at the time. Let k be the value of the corresponding session key. We toss a coin b,

$b \stackrel{R}{\leftarrow} \{0,1\}$. If $b = 0$ we we provide $\mathcal{M}$ with the value k. Otherwise we provide $\mathcal{M}$ with a value r randomly chosen from the probability distribution of the keys generated by this protocol. The attacker $\mathcal{M}$ is not allowed state-reveal queries, session-key queries or party corruptions on the test session or its matching session. At the end of its run, $\mathcal{M}$ outputs a bit b' (its guess for b). An attacker that is allowed test session queries is referred to as a key-exchange adversary.

Definition C.2.1. ***Session-key Security*** *A key-exchange protocol is called Session-key secure (or SK-secure) if the following properties hold for any key-exchange adversary.*

- *The protocol satisfies the property that if two uncorrupted parties complete matching sessions then they both output the same key*
- *the probability that $\mathcal{M}$ guesses correctly the bit b (outputs $b' = b$) is no more than* $1/2$ *plus a negligible fraction ϵ in the security parameter. ϵ is called 'advantage'.*

D

Trusted Platform Module

The Trusted Computing Group (TCG) is an organisation with representatives from leading computer-industry companies formed to improve computer security and authentication. In particular, the TCGs goals were to develop, (a) a means for authenticating a system in the network and, (b) enabling the secure storage of information. Towards this end, the TCG developed the trusted platform module (TPM) that would handle all software and hardware-related security features, including the storage and protection of the keys, the auditing of the surrounding hardware and software components and the creation of the certificates of trust that can then be viewed either remotely by the administrators or employed as part of any associated trusted process. A proper implementation of the TPM must include a strong physical protection (tamper-evidence and tamper-resistance) so that its contents are protected against a physical attack.

As implemented today, the TPM:

- checks the system integrity and the status of the hardware and software environment
- authenticates the system to the network
- enables secure storage, provides hardware-based encryption and key storage
- allows a user to take ownership with strong controls to protect privacy.

Of particular interest to us is the TPM-based authentication that uses two keys to authenticate a platform.

- *Endorsement Key* The TPM consists of a unique *endorsement key* (EK) pair that is built into the hardware module during manufacturing. The public part of the EK is certified by the manufacturer and the secret part is sealed inside the TPM and is never revealed to the outside. The EK is unique to a particular TPM and therefore the particular platform. There are two ways to generate the EK, either by using a specified TPM command or to 'squirt' an externally-generated EK into the TPM. The trust associated with the TPM is based on the fact that EK is unique and protected within the TPM at all times. An endorsement certificate is issued for the public key of the EK; the purpose of the certificate is to provide attestation that the particular TPM is genuine, that is, the EK is secure.
- *Attestation Identity Key* A primary function of the TPM is attestation, that is, the TPM provides guarantees to a remote service that the platform is not tampered with and is therefore secure. To attest, the TPM generates a second key pair called an *Attestation Identity Key* (AIK). The TPM sends the AIK, which is signed by its EK, to a trusted third party (called privacy CA) that verifies its validity, and issues a certificate for the AIK. The host/TPM is now able to authenticate itself with respect to the certificate. The AIKs are always bound to the platform and, thus, can be used to provide attestation for the platform's identification and configuration.

References

[1] *Enhanced border security and visa reform act of 2002*, Pub. L. no. 107-173, 116 Stat. 543, 2002.

[2] *An act making emergency supplemental appropriations for defense, the global war on terror, and tsunami relief, for the fiscal year ending september 30, 2005, and for other purposes*, Pub. L. no. 109– 13, 119 Stat. 231, 2005.

[3] *Trusted computing group*, https://www.trustedcomputinggroup.org/, 2008.

[4] *The archive of formal proof*, http://afp.sourceforge.net/, July 2009.

[5] *Automated validation of internet security protocols and applications*, http://avispa-project.org/, 2009.

[6] *Ekahau tags*, http://www.ekahau.com/, 2009.

[7] S Angelov and P Grefen, *B2B eContract handling - a survey of projects, papers and standards*, Tech. report, University of Twente, The Netherlands, 2001.

[8] ANSI, *Public key cryptography for the financial services industry, key agreement and key transport using elliptic curve cryptography*, Tech. report, American National Standards Institute (ANSI), 2001.

[9] Tuomas Aura, *Strategies against replay attacks*, 10th Computer Security Foundations Workshop (1997), 59–68.

[10] Australian Customs Services, *SmartGate*, http://www.customs.gov.au, 2006.

[11] Federal authority on the electronic identity card, *Belgium identity cards*, http://eid.belgium.be/, 2005.

[12] Gildas Avoine, *Adversarial model for radio frequency identification*, IACR E-print **2005/049** (2005).

[13] ———, *RFID lounge*, March 2008.

[14] M. Baraz, B. Boros, P. Ligeti, K. Loja, and D. Nagy, *Passive attack against the M2AP mutual authentication protocols for RFID tags*, First International EUROSIP Workshop on RFID Technology (2007).

[15] David Basin and Stefan Friedrich, *Modeling a hardware synthesis methodology in Isabelle*, Theorem Proving in Higher Order Logics, vol. LNCS 1125, 1996, pp. 33–50.

[16] Giampalo Bella, *Verifying a smartcard protocol*, Springer Verlag, 2007.

[17] Mihir Bellare, Ran Canetti, and Hugo Krawczyk, *A modular approach to the design and analysis of authentication and key exchange protocols*, ACM Symposium on Theory of Computing'98 (1998).

[18] Mihir Bellare, Juan Garay, Ralf Hauser, Amir Herzberg, Hugo Krawczyk, Michael Steiner, Gene Tsudik, and Micheal Waidner, *iKP – a family of secure electronic payment protocols*, 1995, pp. 89–106.

[19] Mihir Bellare, David Pointcheval, and Phillip Rogway, *Authenticated key exchange secure against dictionary attacks*, Advances in Cryptology - EUROCRYPT'2000, vol. LNCS 1807, 2000, pp. 139–155.

[20] Mihir Bellare and Phillip Rogaway, *Entity authentication and key distribution*, Advances in Cryptology -CRYPTO'93 (1993).

[21] ———, *Random oracles are practical: a paradigm for designing efficient protocols*, ACM Conference on Computer and Communications Security'93 (1993), 62–73.

[22] ———, *Provably secure session key distribution three party case*, ACM Symposium on Theory of Computing'95 (1995).

[23] Robert M. Best, *Microprocessor for executing enciphered programs*, US Patent number 4168396, October 1977.

[24] M Betts, P Black, S Christensen, Ed Dawson, and R Du, *Towards secure and legal e-tendering*, Journal of Information Technology in Construction (2006).

[25] Simon Blake-Wilson and Alfred J. Menezes, *Entity authentication and authenticated key transport protocols employing asymmetric techniques*, 5th International Workship on Security Protocols, vol. LNCS 1361, 1998, pp. 137–158.

[26] ———, *Authenticated diffie-hellman key agreement protocols*, 5th International Workshop on Selected Areas in Cryptography, vol. LNCS 1556, 1999, pp. 339–361.

[27] L Blum, M Blum, and M Shub, *A simple unpredictable pseudo random number generator*, SIAM J. Computing **15** (1986), no. 2, 364–383.

[28] Manuel Blum and Silvio Micali, *How to generate cryptographicaaly strong sequence of pseudo random bits*, Proceedings of the 23rd Annual Symposium on Foundations of Computers Science (FOCS 82), 1982, pp. 112–117.

[29] A Boulmakoul and M Sall, *Integrated contract management*, 9th Workshop of HP OpenView University Association Online Conference, 2002.

[30] Colin Boyd and W Mao, *Designing secure key exchange protocols*, The European Symposium on Research in Computer Security, 1994.

[31] Victor Boyko, Phillip MacKenzie, and Sarvar Patel, *Provably secure password-authenticated key exchange using diffie-hellman*, Advances in Cryptology – EUROCRYPT'2000, vol. LNCS 1807, 2000, pp. 156–171.

[32] Stefan A. Brands, *Untraceable off-line cash in walltes with observers*, Advances in Cryptology – Crypto'93, vol. LNCS 773, Springer–Verlag, 1993, pp. 302–318.

[33] Ernie Brickell, Jan Camenisch, and Liqun Chen, *Direct anonymous attestation*, 11th ACM Conference on Computer and Communications Security, ACM Press, 2004.

[34] Mike Burmester, Breno de Medeiros, and Rossana Motta, *Provably secure grouping-proofs for RFID tags*, Smart Card Research and Advanced Applications'08 **LNCS 5189** (2008), 176–190.

[35] Mike Burmester and Breno De Medeiros, *The security of epc gen2 compliant RFID protocols*, Applied Cryptography and Network Security, ACNS 2008 **LNCS 5037** (2008), 490–506.

[36] L Buttyan, C Gbgauidi, S Staamann, and U Wilhelm, *A note on an authentication technique based on distributed security mangament for the global mobility network*, Tech. Report SSC/98/18, Swiss Federal Institute of Technology, April 1998.

[37] Chritian Cachin, *Efficient private bidding and autions with an oblivious third party*, ACM Conference on Computer and Communications Security'99 (1999), 120–127.

[38] Jan Camenisch and Anna Lysyanskaya, *Dynamic accumulators and application to efficient revocation of anonymous credentials*, Advances in Cryptology -CRYPTO'02 **LNCS 2442** (2002), 101–120.

[39] Jan Camenisch and Markus Michels, *Separability and efficiency for generic group signature schemes*, Advances in Cryptology -CRYPTO'99 **LNCS 1666** (1999), 413—430.

[40] Ran Canetti, *Obtaining universally compoable security: Towards the bare bones of trust*, Advances in Cryptology - ASIACRYPT'07 **LNCS 4833** (2007), 88–112.

[41] Ran Canetti, Moses Samson Charikar, Sridhar Rajagopalan, Shanmugasundaram Ravikumar, Amit Sahai, and Andrew S Tomkins, *Non-transferable anonymous credentials*, Patent No: 7222362, 2000.

[42] Ran Canetti and Hugo Krawczyk, *Analysis of key exchange protocols and their use for building secure channels*, Advances in Cryptology – EUROCRYPT'2001, vol. LNCS 2045, Springer-Verlag, 2001, pp. 453–474.

[43] David Chaum, *Untraceable electronic mail, return addresses, and digital pseudonyms*, Communications of the ACM **24** (1981), no. 2.

[44] ———, *Blind signatures for untraceable payments*, Advances in Cryptology - CRYPTO'82 (1982), 199–203.

[45] ———, *Security without identification: transaction systems to make big brother obsolete*, Communications of the ACM **28** (1985), no. 10, 1030 – 1044.

[46] David Chaum and Jan-Hendrik Evertse, *A secure and privacy–protecting protocol for tranmitting personal information between organisation*, Advances in Cryptology – CRYPTO'86, Springer–Verlag, 1986, pp. 118–167.

[47] David Chaum and T Pedersen, *Transferred cash grows in size*, Advances in Cryptology -EUROCRYPT'92 **LNCS 658** (1992), 390–407.

[48] Lidong Chen, *Access with pseudonyms*, Cryptography: Policy and Algorithms (Ed Dawson and Jovan Golic, eds.), no. 1029, Springer–Verlag, 1995, pp. 232–243.

[49] H.-Y Chien and C.-W Huang, *Security of ultra-lightweight RFID authentication protocol and its improvements*, SIGOPS Operating Systems Review **41** (2007), no. 4, 83–86.

[50] Hung-Yu Chien and Che-Hao Chen, *Mutual authentication protocol for rfid conforming to epc class 1 generation 2 standards*, Computer Standards and Interfaces **29** (2007), 254–259.

[51] E. M. Clarke, S. Jha, and W. Marrero, *Using state space exploration and a natural deduction style message derivation engine to verify security protocols*, IFIP Working Confernece on Programming Concepts and Methods (PROCOMET'98) (D. Gries and W. P. De Roever, eds.), Chapmann and Hall, 1998.

[52] Jonathan Collins, *Delta plans u.s. – wide RFID system*, RFID Journal, 2004.

[53] I Damgard, *Payment systems and credential mechanisms with provable security against abuse by individuals*, Advances in Cryptology - CRYPTO'88 **LNCS 403** (1988), 328–335.

[54] Z. Dang and R. A. Kemmerer, *Using the ASTRAL model checker for cryptographic protocols analysis*, Workshop on Design and Formal Verification of Security Protocols (H. Orman and C. Meadows, eds.), 1997.

[55] Yvo Desmedt, Claude Goutier, and Samy Bengio, *Special uses and abuses of the fiat-shamir passport protocol*, Advances in Cryptology – CRYPTO'87, vol. LNCS 293, Springer Berlin / Heidelberg, 1987, pp. 21–39.

[56] J Dethloff, *Special report: intellectual property rights and smart card patents: the past, the present, the future*, Smart Card News (1996), 36–38.

[57] W Diffie, Paul C. van Oorschot, and M. J. Wiener, *Authentication and authenticated key exchanges*, Designs, Codes, and Cryptography **2** (1992), 107–125.

[58] Whitfield Diffie and Martin E Hellman, *New directions in cryptography*, IEEE Transactions on Information Theory **22** (1976), no. 6, 644–654.

[59] T Dimitriou, *A lightweight rfid protocol to protect against traceability and cloning attacks*, IEEE Internatinal Conference on Security and Privacy in Communication Networks'05 (2005).

[60] R Du, Colin Boyd, and E Foo, *A secure e-tender submission protocol*, International Conference on Trust, Privacy & Security in Digital Business'06 **LNCS 4083** (2006), 213–222.

[61] R Du, E Foo, Colin Boyd, and B Fitzgerald, *Defining security services for electronic tendering*, Australiasian Information Security Workshop'04 (2004).

[62] R Du, E Foo, Juan Gonzalez Nieto, and Colin Boyd, *Designing secure e-tendering systems*, International Conference on Trust, Privacy & Security in Digital Business'05 **3592** (2005), 70–79.

[63] R Du, Ernest Foo, , Juan González Nieto, and Colin Boyd, *Designing secure E-Tendering systems*, TrustBus'2005, Lecture Notes in Computer Science, 2005, pp. 70–79.

[64] Rong Du, Ernest Foo, Colin Boyd, and Brian Fitzgerald, *Defining security services for electronic tendering*, The Australasian Information Security Workshop (ASIW2004), Conferences in Research and Practice in Information Technology, vol. 32, Australian Computer Society, 2004, pp. 43–52.

[65] Dang Nguyen Duc, Jaemin Park, Hyunrok Lee, and Kwangjo Kim, *Enhancing security of epcglobal gen-2 rfid tag against traceability and cloning*, Symposium on Cryptography and Information Security, SCIS 2006 (2006).

[66] EPCglobal Inc, *EPC RFID protocols class-1 generation-2 UHF RFID procol for communciation at 860–960 Mhz version 1.2.0*, (2008).

[67] European Union, *Council regulation (ec) no 2252/2004*, Official Journal of the European Union, 2005.

[68] Federal Information Processing Standards, *Security requirements for cryptographic modules*, 2001.

[69] ———, *Personal identity verification for federal employees and contractors*, 2006.

[70] Federal Ministry of the Interior Germany, *The electronic identity card – a flagship initiative for e-business and e-government*, 2009.

[71] A. Fiat and A. Shamir, *How to prove yourself: Practical solutions to identification and signature problems*, Advances in Cryptology, CRYPTO'86 (New York), vol. LNCS 263, Springer-Verlag, 1986, pp. 186–194.

[72] Yair Frankel, Yiannis Tsiounis, and Moti Yung, *Indirect discourse proofs: Achieving efficient fair off-line e-cash*, Advances in Cryptology - ASIACRYPT'96 (Berlin) (Kwangjo Kim, ed.), vol. LNCS 1163, Springer-Verlag, 1996.

[73] M Franklin and M Reiter, *The design and implementation of a secure auction service*, IEEE Transactions on Software Engineering **22** (1996), no. 5, 302–312.

[74] Matthew Franklin and Stuart Haber, *Joint encryption and message-efficient secure computation*, Advances in Cryptology -CRYPTO'93 **LNCS 773** (1993), 266 – 277.

[75] Gavin Lowe, *Casper – a compiler for the analysis of security protocols, user manual and tutorial, ver1.3*, 1999.

[76] Craig Gentry and Alice Silverberg, *Hierarchical id-based cryptography*, Advances in Cryptology – ASIACRYPT'02, vol. LNCS 2501, Springer-Verlag, 2002, pp. 149–155.

[77] Henri Gilbert, Matthew J B Robshaw, and Yannick Seurin, *HB increasing the security and efficiency of HB^+*, Advances in Cryptology - EUROCRYPT'08 **LNCS 4965** (2008), 361–378.

[78] M Girault, *Self-certified public keys*, Advances in Cryptology – EUROCRYPT'91, vol. LNCS 547, Springer–Verlag, 1991, pp. 490–497.

[79] O Goldreich, *Introduction to complexity theory*, 1989, Lecture notes.

[80] O. Goldreich, *Modern cryptoraphy, probabilistic proofs and pseudo-randomness*, Springer, 1999.

[81] O Goldreich, *The foundations of cryptography*, vol. 1, Cambridge University Press, 2001.

[82] Oded Goldreich and Silvio Micali, *Probabilistic encryption*, Journal of Computer and System Sciences, vol. 28, 1984, pp. 270–299.

[83] S. Goldwasser, S. Micali, and C. Rackoff, *The knowledge complexity of interactive proof systems*, SIAM J. Computing **18** (1985), no. 186-208.

[84] John A. Gordon, *Apparatus for electronic encypherment of digital data*, US Patent number 4165444, November 1977.

[85] U.K. Government, *Identity card act 2006 – c.15*, http://www.opsi.gov.uk/ACTS/acts2006/ukpga_20060015_en_1, 2006.

[86] V. Griffith and Mark Jokobsson, *Messin' with texas: Deriving mothers maiden names using public records*, Applied Cryptography and Network Security (ACNS) (J. Ioannidis, A. D Kermystis, and Moti Yung, eds.), vol. LNCS 3531, Springer-Verlag, 2005, pp. 91–103.

[87] S. Gritzalis, N. Nikitatos, and P. Georgiadis, *Formal methods for analysis and design of cryptographic protocols: A state-of-the-art review*, IFIP Working conference on Communication and Multimedia Security, vol. 3, 1997, pp. 119–132.

[88] L Grunwald, *New attacks against RFID–systems*, GmbH Germany.

[89] David E. Gumpert and William Pentland, *USDA bets the farm on animal ID program*, The Nation, December 2007.

[90] Beverley Head, *Love me E–tender*, http://www.theage.com.au, August 2003.

[91] Dirk Henrici and Paul Muller, *Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers*, IEEE International Conference on Pervasive Computing and Communications'03 (2003), 149–153.

[92] C. A. R Hoare, *Communicating sequential processes*, Prentice Hall International, 1985.

[93] Jaap-Henk Hoepman, Engelbert Hubbers, Bart Jacobs, Martijn Oostdijk, and Ronny Wichers Schreur, *Crossing borders: Security and privacy issues of the european e-passport*, Proceedings of Advances in Information and Computer Security 2006 **LNCS** (2006).

[94] Home Affairs Justice, *EU standard specifications for security features and biometrics in passports and travel documents*, Tech. report, European Union, June 2006.

[95] House Standing Committee on Legal and Constitutional Affairs, Australian Government, *Inquiry into technological protection measures (TPM) exceptions*, http://www.aph.gov.au, March 2006.

[96] ICAO, *Machine readable travel documents*, Tech. report, ICAO, 2006.

[97] Insititute for Information Industry, *Report for the planning of electric system of government procurement*, Public Contruction Commission (Taiwan R. O. C.), 1998.

[98] ISO/IEC, *ISO/IEC 10536 identification cards – contactless integrated circuit(s) cards – close-coupled cards*, 2000.

[99] ISO/IEC, *ISO/IEC 14443 identification cards – contactless integrated circuit(s) cards – proximity cards*, 2000.

[100] ISO/IEC, *ISO/IEC 15693 identification cards – contactless integrated circuit(s) cards – vicinity cards*, 2000.

[101] ISO/IEC, *ISO/IEC 7810 identification cards – physical characteristics*, 2003.

[102] _____, *ISO/IEC 7816 international standard related to electronic identification cards with contacts*, 2003.

[103] ISO/IEC, *ISO/IEC 18013 information technology – personal identification – iso-compliant driving licence*, 2005.

[104] ———, *ISO/IEC 7501 identification cards – machine readable travel documents*, 2005.

[105] ———, *ISO/IEC 24727 identification cards – integrated circuit card programming interfaces*, 2007.

[106] ———, *ISO/IEC 11889 information technology – trusted platform module*, 2009.

[107] Sam Itani, *Successful deployment of rtls in health care*, RFID in Health Care, 2009.

[108] Dave Johnson, *How RFID delivers shipping accuracy and cost savings*, RFID Journal, April 2009.

[109] Ari Juels, *"yoking-proofs" for RFID tags*, IEEE International Conference on Pervasive Computing and Communications'04 (2004).

[110] Ari Juels, D Molnar, and David Wagner, *Security and privacy issues in e-passports*, IEEE Internatinal Conference on Security and Privacy in Communication Networks'05 (2005).

[111] Ari Juels and R Pappu, *Squealing euros: Privacy protection in RFID-enabled banknotes*, Financial Cryptography'03 (2003).

[112] Ari Juels and Michael Szdlo, *A two-server, sealed-bid auction protocol*, Financial Cryptography'03 (2003), 72.

[113] Gaurav S. Kc and Paul A. Karger, *Preventing attacks on machine readable travel documents (MRTDs)*, Cryptology ePrint Archive, Report 2005/404, 2005.

[114] Hyun-Seok Kim, Il-Gon Kim, Keun-Hee Han, and Jin-Young Choi, *Security and privacy analysis of rfid systems using model checking*, High Performance Computing and Communications **LNCS 4208** (2006), 495–504.

[115] D. V. Klien, *A survey of and improvements to password security*, UNIX Security II: USENIX Workshop Proceedings, 1990, pp. 5–14.

[116] D. M. Konidala and K. Kim, *RFID tag-reader mutual authentication scheme utilizing tags access password*, Tech. report, AUTO-ID Labs White Paper WP-HARDWARE-033, 2007.

[117] Divyan M Konidala, Zeen Kim, and Kwangjo Kim, *A simple and cost-effective RFID tag-reader mutual authentication scheme*, RFID Sec 2007 (2007).

[118] Dennis Kügler, *Adavance security mechanisms for machine readable travel documents*, Tech. report, Federal Office for Information Security (BSI), Germany, 2005.

[119] ———, *Security concept of the EU–passport*, Security in Pervasive Computing (2005), 85.

[120] Robert P Kurshan, Vladimir Levin, Marius Minea, Doron Peled, and Hüsnü Yenigün, *Combining software and hardware verification techniques*, Formal Methods in System Design **21** (2002), no. 3, 251—280.

[121] L Gong and P Syverson, *Fail-stop protocols: a new approach to designing secure protocols*, The 5th International Working Conference on Dependable Computing for Critical Applications, 1995, pp. 44–55.

[122] Adam Laurie, *Rfidiot*, http://rfidiot.org/, 2007.

[123] J. Leyden, *Office workes give away passwords for cheap pen*, The Register, 2003.

[124] T. LI and R Deng, *Vulnerability analysis of EMAP – an efficient RFID mutual authentication protocol*, AReS 2007 (2007).

[125] T Li and G Wang, *Security analysis of two ultra-lightweight rfid authentication protocols*, IFIP TC-11 International Information Security Conference'07 (2007).

[126] T. S Liao, M. T. Wang, and H. P Tserng, *A framework of electronic tendering for government procuremnet: a lesson learned in taiwan*, Automation in Construction (2002), no. 11, 731–742.

[127] Mark Lieberman, *The next phase of AIT-enabled distribution at the DLA*, RFID Journal, April 2009.

[128] T L Lim and Tieyan Li, *Addressing the weakness in a lightweight rfid tag-reader mutual authentication scheme*, IEEE Globecom 2007 (2007), 59–63.

[129] Bartec USA LLC, *Direct TPMS*, http://www.tirepressuremonitoringsystem.com, 2009.

[130] Radu loan Paise and Serge Vaudenay, *Mutual authentication in RFID: security and privacy*, ACM symposium on Information, computer and communications security, ASIACCS 2008 (2008), 292–299.

[131] G. Lowe, *Breaking and fixing the needham-schroeder public-key protocol using csp and fdr*, Tools and Alogrithms for the Construction and Analysis of Systems (T. Margaria and B. Steffen, eds.), vol. LNCS 1055, Springer-Verlag, 1996, pp. 147–166.

[132] ———, *Some new attacks upon security protocols*, 9th IEEE Computer Security Foundations Workshop, IEEE Computer Society Press, 1996.

[133] G. Lowe and B. Roascoe, *Using CSP to detect errors in the TMN protocols*, IEEE Transactions on Software Engineering, vol. 3, 1997.

[134] Formal Systems (Europe) Ltd, *Failuers-divergence refinement, FDR2 user manual*, 2003, Available from http://www.fsel.com.

[135] Anna Lysyanskaya, Ronald L Rivest, Amit Sahai, and Stefan Wolf, *Pseudonym systems (extended abstract)*, Selected Areas in Cryptography'99 **LNCS 1758** (1999), 184–199.

[136] M Abadi and M Tuttle, *A semantics for a logic and authentication*, The 10th ACM Symposium on Priciples of Distributed Computing, ACM Press, 1991, pp. 201–216.

[137] Antonio Mana, Antonio MuNoz, and Daniel Serrano, *Hardware protection of agents in ubiquitous and ambient intelligence environments*, 7th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2009), Advances in Soft Computing, vol. 55, 2009, pp. 470–479.

[138] E. H. McKinney, *Generalized birthday problem*, American Mathematical Monthly **73** (1966), 385—387.

[139] C. Meadows, *Formal verification of cryptographic protocols: a survey*, Advances in Cryptology – ASIACRYPT'94, Springer–Verlag, 1994, pp. 135–150.

[140] ———, *A model of computation for NRL protocol analyzer*, The 7th Computer Society Foundations Workshop, IEEE Computer Society Press, 1994.

[141] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone, *Handbook of applied cryptography*, CRC Press, 2001.

[142] J. C. Mitchell, M. Mitchell, and U. Stern, *Automated analysis of cryptographic protocols using murphi*, 16th IEEE Symposium on Security and Privacy, IEEE Computer Society Press, 1997.

[143] Jean Monnerat, Serge Vaudenay, and Martin Vuagnoux, *About machine-readable travel documents privacy enhancements using (weakly) non-transferable data authentication*, RFID Privacy Workshop (2007).

[144] Antonio Munoz and Antonio Mana, *A hardware based infrastructure for agent protection*, 3rd Symposium of Ubiquitous Computing and Ambient Intelligence 2008, Advances in Soft Computing, vol. 51, Springer Verlag, 2008, pp. 39–47.

[145] N. Heintze and J. D. Tygar, *A model for secure protocols and their compositions*, 1994 IEEE Computer Society Symposium on Research in Security and Privacy, IEEE Computer Society Press, 1994, pp. 2–13.

[146] Moni Naor and Benny Pinkas, *Visual authentication and identification*, Advances in Cryptology CRYPTO'97, vol. LNCS 1294, Springer-Verlag, 1997, pp. 322–336.

[147] Moni Naor, Benny Pinkas, and Reuben Sumner, *Privacy preserving auctions and mechanism design*, ACM Conference on Electronic Commerce (1999), 129–139.

[148] New South Wales Government Australia, *NSW government electroic procurement implementation strategy*, http://www.cpsc.nsw.gov.au, 2008.

[149] NFC Forum, *Near field communication interface and protocol – 2*, 2009.

[150] Tobias Nipkow, *Verifying a hotel key card system*, Theoretical Aspects of Computing (CTAC'2006), 2006.

[151] Mary Catherine O'Connor, *Purdue moving OxyContin RFID pilot to full production*, RFID Journal, Feb 2007.

[152] Vijayakrishnan Pasupathinathan, Josef Pieprzyk, and Huaxiong Wang, *A fair e-tendering protocol*, International Conference on Security and Cryptography (SECRYPT 2008) (2008), 294–299.

[153] ———, *Formal security analysis of australian e-passport implementation*, Sixth Australasian Information Security Conference AISC 2008 **CRPIT 81** (2008), 75 — 82.

[154] ———, *An on-line secure e-passport protocol*, Information Security Practice and Experience (ISPEC)'08 **LNCS 4991** (2008), 14–28.

[155] ———, *Security analysis of australian and e.u. e-passport implementation*, Journal of Research and Practice in Information Technology **40** (2008), no. 3, 187–205.

[156] ———, *Certified pseudonyms colligated with master secret key*, International Conference on Security and Cryptography (SECRYPT 2009) (2009).

[157] ———, *Mutual authentication for EPC Gen2 complaint devices*, Tech Report, 2009.

[158] Vijayakrishnan Pasupathinathan, Josef Pieprzyk, Huaxiong Wang, and Joo Yeon Cho, *Formal analysis of card-based payment systems in mobile devices*, Fourth Australasian Information Security Workshop (Network Security) (AISW 2006), Hobart, Australia. **CRPIT 54** (2006), 213–220.

[159] L. C. Paulson, *Isabella: A generic theorem prover*, **LNCS 828** (1994).

[160] ———, *The inductive approach to verifying cryptographic protocols*, Computer Security **6** (1998), 85–128.

[161] T Pedersen, *Non-interactive and information theoretic secure verifiable secret sharing*, Advances in Cryptology - CRYPTO'91 (J Feigenbaum, ed.), Lecture Notes in Computer Science, Springer-Verlag, 1991.

[162] P Peris-Lopez, J Hernandez-Castro, JM Estevez-Tapiador, and Artura Ribagorda, *EMAP: an efficient mutual authentication protocol for low-cost RFID tags*, On the Move to Meaningful Internet Systems Workshops'06 **LNCS 4277** (2006), 352–361.

[163] ———, *M2AP: a minimalist mutual-authentication protocol for low-cost RFID tags*, International Conference on Ubiquitous Intelligence'06 **LNCS 4159** (2006), 912–923.

[164] B. Pfitzmann, *Sorting out signature schemes*, The First ACM Conference on Computer and Communicaiton Security, ACM SIGSAC, November 1993, pp. 74–85.

[165] P. J. Phillips, A. Martin, C. L. Wilson, and M. Przybocki, *An introduction evaluating biometric systems*, IEEE Computer **33** (2000), no. 2, 56–63.

[166] David Pointcheval, *The composite discrete logarithm and secure authentication*, International Workshop on Practice and Thoery in Public Key Cryptography - PKC'2000 (Melbourne, Australia) (H Imai and Y Zheng, eds.), vol. LNCS 1751, Springer-Verlag, Jan. 2000, pp. 113–128.

[167] David Pointcheval and Jacques Stern, *Security proofs for signature schemes*, Advances in Cryptology - EUROCRYPT'96 (Ueli Muarer, ed.), vol. LNCS 1070, Springer-Verlag, 1996, pp. 387–398.

[168] ———, *Security proofs for signature schemes*, Advances in Cryptology - EUROCRYPT'96 **LNCS 1070** (1996), 387–398.

[169] Guillaume Poupard and Jacques Stern, *Security analysis of a practical "on the fly" authentication and signature generation*, Advances in Cryptology - EUROCRYPT'98 **LNCS 1403** (1998), 422–436.

[170] Public Works and Government Services Canada, *MERX: Canada's electronic tendering service*, http://www.merx.com/, 2008.

[171] RFID Gazette, *More bookstores using RFID*, October 2006.

[172] ———, *Wal-Mart doubling RFID-enabled stores*, http://www.rfidgazette.org/walmart/, 2007.

[173] A. Rubin and P. Honeyman, *Formal methods for analysis of authentication protocols*, Tech. Report 93-7, CITI TR, October 1993.

[174] P. Y. A Ryan, *Modelling and analysis of security protocols, research proposal*, Tech. report, Defence Research Agency, 1994.

[175] K Sakurai and S Miyazaki, *A bulletin-bard based digital auction scheme with bidding down stratergy*, CrypTEC'99 (1999), 180–187.

[176] S Sarma, S Weis, and D Engels, *Rfid systems and security and privacy implications*, Cryptographic Hardware and Embedded Systems'02 (2002).

[177] S. Schneider, *Verifying authentication protocols with CSP*, 10th IEEE Computer Security Foundations Workshop, IEEE Computer Society Press, 1997, pp. 2–17.

[178] Steve Schneider, *Verifying authentication protocols with CSP*, 10th IEEE Computer Security Foundations Workshop, IEEE Computer Society Press, 1997, pp. 2–17.

[179] Javed Sikande, *Rfid enabled retail supply chain*, MSDN, 2005.

[180] S. W. Smith, V. Austel, R. perez, and S. Weingart, *Validating a high performance, programmable secure coprocessor or, the world's first fips 140-1 level 4*, 22nd National Information Systems Security Conference, 1999.

[181] S. W. Smith and S. Weingart, *Building a high performance, programmable secure coprocessor.*, Computer Security (Special Issue on Computing Network Security) **31** (1999), 831–860.

[182] Boyeon Song and Chris J Mitchell, *Rfid authentication protocol for low-cost tags*, ACM conference on Wireless network security, WiSec 2008 (2008), 140—147.

[183] StealthTrac, *STEALTH TRAC active GPS unit*, http://www.stealthtrac.com, 2009.

[184] P. Syverson, *Formal semantics for logics of cryptographic protocols*, The Computer Security Foundations Workshop III, IEEE Computer Society Press, June 1990, pp. 32–41.

[185] P. Syverson and C. Meadows, *A logical language for specifying cryptographic protocols requirements*, 1993 IEEE Computer Society Symposium on Research in Security and Privacy (Los Alamitos, California), IEEE Computer Society Press, 1993, pp. 165–177.

[186] Tanscore, *Transcore unveils rfid-based ezgo anywhere tag for national interoperable electronic toll collection*, 76th Annual International Bridge Tunnel and Turnpike Association Meeting Exposition, 2008.

[187] TCG, *Trusted computing group main specification v1.1*, 2001.

[188] ———, *Trusted computing group main specification v1.2*, 2007.

[189] PKI Task Force Tom A. F. Kinneging for ICAO-NTWG, *PKI for machine readable travel documents offering ICC read-only access*, Tech. report, October 2004, Version 1.1.

[190] K. Viswanathan, Colin Boyd, and Ed Dawson, *A three phased scheme for seal bid auction system design*, ACISP'2000 (Ed Dawson, A Clark, and Colin Boyd, eds.), vol. LNCS 1841, Springer-Verlag, 2000, pp. 412–426.

[191] Stephen B Weirstein, *Security mechanisms in electronic cards*, Advances in Cryptology -CRYPTO'81 (1981), 109.

[192] Rhea Wessel, *Tetas textiles tracks deliveries*, RFID Journal, November 2009.

[193] S White and L Comerford, *Abyss: an architecture for software protection*, Software Engineering, IEEE Transactions on **16** (1990), no. 6, 619 – 629.

[194] M Whitteman, *Attacks on digital passports*, Riscure, 2005.

[195] Andreas Wiemers, *Kommentare zu application interface for smart cards used as secure signature creation device, part 1 – basic requirements*, Tech. Report Version 0.14, Bonn, Germany, March 2003.

[196] Steve H Wiengart, *Physical security devices for computer subsystems: A survey of attacks and defenses*, Cryptographic Hardware and Embedded Systems — CHES 2000 **LNCS 1965** (2000), 45–68.

[197] Steve H Wiengart, Steve R White, William C Arnold, and Glen P Double, *An evaluation system for the physical security of computing systems*, Proceedings of the Sixth Annual Computer Security Applications Conference, 1990. (1990), 232–243.

[198] T.Y.C Woo and S. Lam, *A semantic model for authentication protocols*, The 1993 Symposium on Research in Security and Privacy, IEEE Computer Society Press, May 1993, pp. 178–194.

[199] R. Yahalom, *Optimality of asynchronous two-party secure data-exchange protocols*, Journal of Computer Security **2** (1994), no. 2-3, 191–209.

[200] Jeongmo Yang, J Park, H Lee, K Ren, and K Kim, *Mutual authentication protocol for low-cost rfid*, Ecrypt Workshop on RFID and Lightweight Crypto (2005).

[201] Andrew C. Yao, *Theory and applications of trapdoor functions*, Proceedings of the 23rd Annual Symposium on Foundations of Computers Science (FOCS 82), 1982, pp. 80–91.

[202] J. Yoshida, *Euro bank notes to embed RFID chips by 2005*, http://www.eetimes.com, 2001.