Enhancing the security of electronic commerce transactions

Vorapranee Khu-smith

Technical Report
RHUL–MA–2003–7
June 2003

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

Enhancing the security of electronic commerce transactions

by

Vorapranee Khu-smith

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

Department of Mathematics
Royal Holloway, University of London
2003

Declaration

These doctoral studies were conducted under the supervision of Professor Chris Mitchell and Professor Peter Wild.

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

Vorapranee Khu-smith
8 April 2003

Acknowledgements

This thesis would not have been possible without the support of a number of people. First and foremost, I would like to express my deep gratitude to my supervisor Chris Mitchell for his endless support, patience, interest, and guidance throughout my studies at Royal Holloway. Without his valuable comments, expertise, and patience in editing and teaching, it would not have been possible for me to finish this research. Chris, you would not realise how much I have learnt from you and how much your excellent supervision, patience, and care have changed my life. I would also like to thank Peter Wild for many valuable pieces of advice. I would like to especially thank Keith Martin, Scarlet Schwiderski-Grosche, Dr. Stephen Hailes, Professor Vesna Hassler, and the anonymous referees for their helpful comments that have significantly improved the quality of this thesis.

I cannot thank my parents enough for their love and support throughout my whole life. Without them, I would not have this great opportunity to study for the degree. I also would like to thank them for everything I have and everything I am.

I am very grateful to my sister and brother, P'Dear and Duke, for their love and care. P'Dear has been my inspiration since I was young. I am so lucky to have her to talk to and tell about everything. The trust and faith that Duke has in me has always encouraged me through difficult times in my life. I feel gifted that we were born to be sisters and brother. Special thanks to my brother in law, P'Dod for being so supportive and caring that I feel as if he is my real brother.

I would like to thank Oak for his love, tolerance, kindness, and support over the past decade. He has tirelessly taken care of me and helped me with everything he can. Oak, thank you for always being there for me and for giving me your love.

Particular thanks to my friend Niklas Borselius for having been so nice to me since the first day we met. I also would like to thank him for being so patiently helpful since I always run to him for help. Special thanks to my officemate Ioannis Michalopoulos for being such a caring person to me. My thanks also go to my friends and colleagues Scarlet Schwiderski-Grosche, Po Yau, Anand Gajparia, Julie Fu, P'Pitt, Peng, Sattam Al-Riyami, Sungbaek Cho, Simos

Xenitellis, Paulo Pagliusi, Arnold Yau, and Keith Martin for helping to make my stay at Royal Holloway so pleasant and enjoyable.

List of Publications

A number of papers resulting from this work have been presented in refereed conferences.

- V. Khu-smith and C. J. Mitchell, Enhancing the security of cookies, in: K. Kim (ed.), *Information Security and Cryptology — ICISC 2001 — Proceedings of the 4th International Conference, Seoul, Korea, December 2001*, Springer-Verlag (LNCS 2288), Berlin (2002), pp.132-145.
- V. Khu-smith and C. J. Mitchell, Electronic transaction security: An analysis of the effectiveness of SSL and TLS, *2002 International Conference on Security and Management (SAM '02), Las Vegas, Nevada, USA, June 2002*, CSREA Press, pp. 425-429.
- V. Khu-smith and C. J. Mitchell, Using EMV cards to protect e-commerce transactions, in: K. Bauknecht, A. Min Tjoa and G. Quirchmayr (eds.), *Proceedings of EC-Web 2002, 3rd International Conference on Electronic Commerce and Web Technologies, Aix-en-Provence, France, September 2002*, Springer-Verlag (LNCS 2455), Berlin (2002), pp.388-399.
- V. Khu-smith and C. J. Mitchell, Using GSM to enhance e-commerce security, in: WMC '02, *Proceedings of the Second ACM International Workshop on Mobile Commerce, Atlanta, Georgia, USA, September 2002*, ACM Press, New York, 2002, pp.75-81.
- V. Khu-smith and C. J. Mitchell, Enhancing e-commerce security using GSM authentication, *Proceedings of EC-Web 2003, 4th International Conference on Electronic Commerce and Web Technologies, Prague, Czech Republic, September 2003*, Springer-Verlag.

Abstract

This thesis looks at the security of electronic commerce transaction processing. It begins with an introduction to security terminology used in the thesis. Security requirements for card payments via the Internet are then described, as are possible protocols for electronic transaction processing. It appears that currently the Secure Socket Layer (SSL) protocol together with its standardised version Transport Layer Security (TLS) are the most widely used means to secure electronic transactions made over the Internet. Therefore, the analysis and discussions presented in the remainder of the thesis are based on the assumption that this protocol provides a 'baseline' level of security, against which any novel means of security should be measured.

The SSL and TLS protocols are analysed with respect to how well they satisfy the outlined security requirements. As SSL and TLS provide transport layer security, and some of the security requirements are at the application level, it is not surprising that they do not address all the identified security requirements.

As a result, in this thesis, we propose four protocols that can be used to build upon the security features provided by SSL/TLS. The main goal is to design schemes that enhance the security of electronic transaction processing whilst imposing minimal overheads on the involved parties. In each case, a description of the new scheme is given, together with its advantages and limitations. In the first protocol, we propose a way to use an EMV card to improve the security of online transactions. The second protocol involves the use of the GSM subscriber authentication service to provide user authentication over the Internet. Thirdly, we propose the use of GSM data confidentiality service to protect sensitive information as well as to ensure user authentication.

Regardless of the protection scheme employed for the transactions, there exist threats to all PCs used to conduct electronic commerce transactions. These residual threats are examined, and motivate the design of the fourth protocol, proposed specifically to address cookie threats.

Abbreviations

3GPP	Third Generation Partnership Project
AAC	Application Authentication Cryptogram
AC	Application Cryptogram
ACS	Access Control Server
ARPC	Authorisation Response Cryptogram
ARQC	Authorisation Request Cryptogram
AuC	Authentication Centre
B2C	Business to Consumer
CA	Certification Authority
CD	Card Details
CGI	Common Gateway Interface
CVC	Card Verification Code
CVM	Cardholder Verification Method
DAD	Dynamic Application Data
DSA	Digital Signature Algorithm
E-Commerce	Electronic Commerce
EMV	Europay MasterCard Visa
IAC	Internal Authenticate Command
IC card	Integrated Circuit card
ID	Identifier

IP	Internet Protocol
GSM	Global System for Mobile Communications
HMAC	Hash Message Authentication Code
HTML	HyperText Markup Language
HTTPS	Secure HyperText Transfer Protocol
MAC	Message Authentication Code
ME	Mobile Equipment
MN	Mobile phone Number
MS	Mobile Station
PAN	Primary Account Number
PC	Personal Computer
PD	Payment Details
PI	Purchase Information
PIN	Personal Identification Number
PKI	Public Key Infrastructure
POS	Point of Sale
RSA	Rivest Shamir Adleman
SAD	Static Application Data
SET	Secure Electronic Transaction
SIM	Subscriber Identity Module
SPA	Secure Payment Application
SPC	Software Publisher Certificate
SSL	Secure Sockets Layer
SMS	Short Message Service
TC	Transaction Certificate
TLS	Transport Layer Security

UCAF	Universal Cardholder Authentication Field
UMTS	Universal Mobile Telecommunications System
URL	Uniform Resource Locator
U-SIM	User Services Identity Module
WAP	Wireless Application Protocol

Contents

1 Introduction **19**

1.1 Motivation and challenges 20

1.2 Main contributions of this thesis 22

1.3 Structure of the thesis 23

2 Security Definitions **25**

2.1 Introduction 26

2.2 Security services 26

2.2.1 Confidentiality 26

2.2.2 Integrity 26

2.2.3 Authentication 27

2.2.4 Non-repudiation 27

2.2.5 Access control 28

2.3 Security mechanisms 28

2.3.1 Asymmetric cryptography 29

2.3.2 Symmetric cryptography 30

I Overview of Electronic Commerce Transactions **33**

3 Security Issues **34**

3.1 Introduction 36
3.2 Security requirements 37
3.2.1 Issuers and acquirers 37
3.2.2 Merchants 38
3.2.3 Clients 39
3.3 Meeting the security requirements 40
3.3.1 SSL and TLS 40
3.3.2 Secure Electronic Transaction (SET) 41
3.3.3 Visa 3-Domain Secure 42
3.3.4 MasterCard Secure Payment Application 43
3.3.5 Summary 44

4 SSL/TLS Analysis **45**

4.1 Introduction 47
4.2 An overview of SSL and TLS 47
4.2.1 Secure Sockets Layer (SSL) 47
4.2.2 Transport Layer Security (TLS) 49
4.3 Analysis 51
4.3.1 Confidentiality 51
4.3.2 Integrity 53
4.3.3 Authentication 53
4.3.4 Non-repudiation 56
4.3.5 Replay protection 56
4.3.6 Summary 57
4.4 Conclusions 57

II Enhancing the Security of E-Commerce 59

5 EMV Cards in E-Commerce 60

5.1 Introduction 62

5.2 An overview of EMV 64

5.2.1 Card authentication 69

5.2.2 Cardholder verification 70

5.2.3 Application cryptograms 71

5.3 Using EMV cards for e-commerce transactions 72

5.3.1 System architecture 72

5.3.2 Transaction processing procedures 74

5.3.3 Security services 79

5.4 Threat analysis 80

5.4.1 Threats to the cardholder environment 81

5.4.2 Threats at the Merchant Server 82

5.5 Use of trusted card readers 83

5.5.1 System architecture 83

5.5.2 Transaction process 85

5.5.3 Threat analysis 85

5.6 Advantages and disadvantages 87

5.7 Related work 89

5.8 Conclusions 90

6 GSM and E-Commerce 91

6.1 Introduction 93

6.2 An overview of GSM security 94

6.2.1 Subscriber identity authentication 94

6.2.2 Data confidentiality 95
6.3 Using GSM authentication for e-commerce 95
6.3.1 System architecture 96
6.3.2 Transaction processing 98
6.3.3 Threat analysis 99
6.3.4 Advantages and disadvantages 105
6.3.5 Summary 107
6.4 Using GSM data encryption for e-commerce 108
6.4.1 System architecture 108
6.4.2 Transaction processing 111
6.4.3 Threat analysis 114
6.4.4 Advantages and disadvantages 123
6.4.5 Summary 124
6.4.6 A comparison of the two proposed protocols 124
6.5 Related work 125
6.6 Extending the protocols to 3G/UMTS 126
6.6.1 3G/UMTS security 126
6.6.2 Protocol extension 127
6.7 Conclusions 128

7 The Remaining Threats 129

7.1 Introduction 131
7.2 Active content 131
7.2.1 Java applets 132
7.2.2 ActiveX controls 133
7.2.3 Security implications 134
7.3 Web browser flaws 136

7.4 Cookies 137
7.4.1 Monitoring user behaviour using cookies 138
7.4.2 Compromising confidentiality of cookie contents 139
7.4.3 Malicious cookies 140
7.5 Conclusions 141

8 Enhancing the Security of Cookies 143

8.1 Introduction 145
8.2 Security requirements 146
8.2.1 Cookie confidentiality 146
8.2.2 Cookie integrity 147
8.2.3 Cookie authentication 147
8.3 Meeting the security requirements 148
8.3.1 Browsers 148
8.3.2 Secure channels 148
8.3.3 Access control for user PCs 149
8.3.4 Cryptographic protection within cookie files 150
8.3.5 Summary 150
8.4 Server-managed cookie encryption 151
8.5 User-managed cookie encryption 153
8.5.1 Using symmetric cryptography 153
8.5.2 Using asymmetric cryptography 155
8.6 User-managed cookie protocols 156
8.6.1 Cookie encryption using symmetric cryptography 156
8.6.2 Cookie encryption using asymmetric cryptography 158
8.6.3 Comparisons 160
8.6.4 Security services 161

8.7 Secure cookies vs. user-managed cookies 161
8.7.1 User authentication 162
8.7.2 Integrity and confidentiality 162
8.7.3 Cookie authentication 163
8.7.4 Other issues 163
8.7.5 Further development 163
8.8 Summary and conclusions 164

9 Conclusions and Directions for Future Research 166

9.1 Summary and conclusions 167
9.2 Directions for future research 170

Bibliography 172

List of Tables

4.1 Differences between SSL and TLS 50

4.2 Security services provided by SSL and TLS 57

8.1 Secure Cookie Components 152

8.2 Comparisons of server-managed and client-managed cookies . . . 164

List of Figures

3.1 Debit/credit card payment system. 36

5.1 EMV card process flowchart. 68

5.2 Remote EMV system architecture 72

5.3 Transaction flow for remote EMV 75

5.4 Possible trusted card readers 84

5.5 Transaction flow for remote EMV with secure card reader 86

6.1 System architecture — GSM-based cardholder authentication. . . 96

6.2 GSM-e-commerce identity authentication process. 99

6.3 Revised protocol. 104

6.4 Another revised protocol. 105

6.5 GSM-e-commerce payment system architecture. 109

6.6 GSM-e-commerce payment protocol. 112

6.7 Revised protocol. 122

8.1 Cookie encryption using symmetric cryptography. 157

8.2 Cookie encryption using asymmetric cryptography. 159

Chapter 1

Introduction

Contents

1.1 Motivation and challenges 20
1.2 Main contributions of this thesis 22
1.3 Structure of the thesis 23

The aim of this chapter is to provide an introduction to the research in this thesis, including an outline of the motivation for the research and the major challenges in the research area. The chapter also describes the main contributions of this thesis, and the overall structure.

1.1 Motivation and challenges

According to the International Electrotechnical Consortium (IEC)[1], 'electronic commerce (e-commerce) can be broadly defined as a model of selling and buying in which buyers are able to participate in all phases of a purchase decision, while stepping through those processes electronically rather than in a physical store. The processes in electronic commerce include enabling a customer to access product information, select items to purchase, pay for the items securely, and have the payment settled financially.'

Today, the Internet has become the main medium for conducting electronic commerce. Many products, tangible and intangible, are browsed through and sold over the Internet. There are a number of possible payment methods, such as electronic cash, electronic cheque, debit/credit card, and electronic wallets. However, debit/credit cards are by far the most common payment method used over the Internet.

As the scale of electronic commerce transactions has grown, it has become very attractive to criminals, and the volume of fraudulent e-commerce transactions is growing rapidly. Therefore, there has been an increase in the amount of attention given to the security of the payment systems used to process online transactions. Probably the main current concern of most Internet users relates to the confidentiality of payment card information, since there is a growing realisation that stolen card details can be used to make fraudulent transactions.

One may argue that card account information is often revealed in everyday use, for example when one makes a card payment in petrol stations, supermarkets, restaurants or any other shop. However, what makes online payments different from those performed at conventional retail outlets is that the latter have the advantage of face-to-face interactions and hand-written signatures,

[1] http://www.iec.org

which act as countermeasures to card fraud.

It is clear that if data confidentiality is not ensured, it would be possible for an adversary to obtain sensitive information such as card details and then use them to make payments at the expense of the legitimate cardholder. However, security requirements for online transactions are not limited to data confidentiality, but also include other security services such as user authentication, non-repudiation and data integrity. It is clear, for example, that online transactions need to be shown to be authentic, i.e. they have not been modified in a way that could enable fraud, and have originated from a legitimate user. Moreover, it is important that once a transaction has been made, both merchant and client cannot deny receiving or making a payment.

For the past few years, a number of solutions have been introduced to improve online transaction processing security. Examples include First Virtual, NetCash, and SET (Secure Electronic Transaction). However, Secure Socket Layer (SSL), and its standardised version Transport Layer Security (TLS), remain by far the most widely used means for providing security services for e-commerce transactions [23, 86], despite the fact that these protocols were designed to provide security for communications links, and not for entire e-commerce transactions.

Although SSL/TLS does eliminate some security risks such as eavesdropping and unauthorised modification, it only protects information while it is being transmitted. Therefore, there remain a number of risks and threats which can lead to card fraud. As a result, there is a need for electronic payment protocols which enhance the security of e-commerce transaction processing over and above the level provided by SSL/TLS. The protocols also need to be sufficiently lightweight, i.e. not posing too great an overhead on the participants, so that there are not serious disincentives to their adoption.

1.2 Main contributions of this thesis

This thesis looks at ways to enhance the security of electronic commerce transaction processing. The main contributions of this thesis are as follows.

- Security requirements for an Internet payment have been identified and the currently available security protocols have been reviewed against these requirements.
- A thorough analysis of the effectiveness of SSL/TLS in meeting the identified security requirements is given.
- Four novel protocols have been proposed to help meet the identified security requirements. The main characteristics of these four protocols are now described.

In the first protocol, a way of using EMV IC cards for secure remote payments, such as those made in e-commerce, is proposed. The scheme only requires users to possess an EMV-compliant debit/credit card and a smart card reader. Threats to, and advantages and disadvantages of, the scheme are also examined. A variation of the protocol to incorporate the use of a trusted card reader is also discussed, and the impact of this modification on the overall level of security is considered.

In the second protocol, user authentication is provided using GSM 'subscriber identity authentication'. A consumer is required to possess a GSM mobile station with a subscriber name corresponding to that on his/her debit/credit card. The cardholder identity is combined with the GSM subscriber identity in such a way that without a mobile station, in particular the SIM, and the corresponding debit/credit card, an unscrupulous user will find it difficult to make a fraudulent payment at the expense of the legitimate cardholder. This is achieved in such a way that no management overhead is imposed on the user.

The third protocol, which also uses a GSM security service, is a payment protocol in which the risk of having debit/credit card details stored at a merchant server is eliminated. User authentication is also provided. This is achieved by using the GSM data confidentiality service to encrypt sensitive information as well as to provide user identity authentication. Since the 3rd generation mobile communications system is becoming a reality, variations of the GSM-based techniques to allow use of a 3GPP U-SIM instead of a GSM SIM are also described and evaluated.

Even when a protocol designed to secure an electronic commerce transaction is used, there are certain remaining threats which are inherent when user PCs are connected to the Internet. One major problem of this type arises from the use of cookies. As a result, a fourth protocol is proposed specifically to deal with threats arising from cookies. Two user-controlled cookie encryption schemes are presented. The approaches are also compared with a server-controlled approach, to illustrate their relative advantages and disadvantages.

1.3 Structure of the thesis

This thesis is divided into two main parts. Part I provides an overview of, and background information regarding, e-commerce security. Part II describes and analyses four protocols which have been proposed with the goal of enhancing the security of e-commerce transactions.

Chapter 2 is a preliminary chapter containing the definitions of security terminology used in the thesis.

Part I then consists of two chapters. In Chapter 3, the security requirements necessary for electronic transaction processing are identified, followed by a review of some of the main currently available security protocols. A review of

the security services each protocol provides is subsequently given. Chapter 4 provides a thorough analysis of the SSL and TLS protocols in terms of how well they meet the security requirements previously outlined.

Part II consists of five chapters. While Chapter 5 proposes a way in which EMV cards can be used to enhance e-commerce security, Chapter 6 proposes two payment protocols in which GSM subscriber identity authentication and GSM data confidentiality are used to support e-commerce security services. Chapter 7 examines general threats to a PC used to conduct electronic commerce transactions. In Chapter 8, a novel approach to cookie encryption is described, which addresses one of the major sources of threats identified in Chapter 7. Finally, Chapter 9 discusses possible future research directions and concludes the thesis.

Chapter 2

Security definitions

Contents

2.1 Introduction 26
2.2 Security services 26
2.2.1 Confidentiality 26
2.2.2 Integrity 26
2.2.3 Authentication 27
2.2.4 Non-repudiation 27
2.2.5 Access control 28
2.3 Security mechanisms 28
2.3.1 Asymmetric cryptography 29
2.3.2 Symmetric cryptography 30

The aim of this preliminary chapter is to provide definitions of the security terminology used in this thesis.

2.1 Introduction

When designing a security protocol, it is important firstly to define its security goals. These goals are generally referred to as security services. It is also important to identify the mechanisms which can provide each security service.

In this chapter, therefore, we first define the security services of relevance to this thesis (Section 2.2). In Section 2.3, security mechanisms which can be used to provide the security services are then examined.

2.2 Security services

There are five main security services which are of importance in this thesis. They are confidentiality, integrity, authentication, non-repudiation, and access control. The following definitions are based on those given in [22, 29, 50, 72, 81].

2.2.1 Confidentiality

Confidentiality means that the assets of a computer system and transmitted information and/or data are protected against access by unauthorised entities. Possible methods of access include printing, displaying, and other forms of disclosure, including simply revealing the existence of the information. According to Ford [22], 'confidentiality services protect against information being disclosed or revealed to entities not authorised to have that information.'

2.2.2 Integrity

Integrity means that the assets of a computer system and transmitted information and/or data can be modified only by authorised parties and only in

authorised ways. Data integrity services therefore are 'safeguards against the threat that the value or existence of data might be changed in a way inconsistent with the recognized security policy' [22]. Modification or changing the value of a data item includes writing, changing, changing the status, deleting, substituting, inserting, reordering, and delaying or replaying of transmitted messages [22, 72].

The Clark-Wilson model [10] defines integrity as those qualities which give data and systems both internal consistency and a good correspondence to real-world expectations for the systems and data. Controls are needed for both internal consistency and external consistency.

2.2.3 Authentication

Authentication can be subdivided into origin authentication and entity authentication.

According to [50], origin authentication provides corroboration to an entity that the source of received message is as claimed. The data origin authentication service provides the corroboration of the source of a data unit. However, the service in itself does not provide protection against duplication or modification of data units.

Entity authentication ensures that an identity presented by a remote party participating in a communication connection or session is genuine [22]. It is 'an ability to verify an entity's claimed identity, by another entity' [28].

2.2.4 Non-repudiation

Non-repudiation is the ability to prove that an action or event has taken place, so that this event or action cannot be repudiated later [36]. In other words, the non-repudiation service provides protection against one party to a communication

exchange later falsely denying that the exchange occurred. Non-repudiation of receipt or transmission provides the sender or the receiver respectively with the means to establish that the message was indeed received or transmitted. According to Ford [22], a non-repudiation service, in itself, does not eliminate repudiation. He states that 'it does not prevent any party from denying another party's claim that something occurred. What it does is ensure the availability of irrefutable evidence to support the speedy resolution of any such disagreement.'

2.2.5 Access control

The goal of an access control service is to protect against unauthorised access to any resource, for example a computing resource, communications resource, or information resource [22]. Unauthorised access includes unauthorised use, disclosure, modification, destruction, and issuing of commands. It requires that access to the protected resources be controlled.

2.3 Security mechanisms

Security mechanisms are means to achieve the security services described above. There is no single mechanism that can provide all the security services. However, there is one main class of techniques that underlies most of the security mechanisms in use, namely cryptographic mechanisms [81].

In this section, we briefly describe each security mechanism of importance to this thesis in terms of what they are, rather than the details of their operation. Such details are outside the scope of the thesis and can be found in many cryptography textbooks, see for example [22, 66, 83].

2.3.1 Asymmetric cryptography

The concept of asymmetric cryptography, or public key cryptography, was first introduced in 1976 by Diffie and Hellman [12]. An asymmetric cryptosystem is a cryptographic scheme in which two distinct keys, known as the public key and the private key, are used. The public key, as its name suggests, can be made public to everyone in the communications system. On the other hand, the private key must be kept secret, and known only to its legitimate owner.

Although confidentiality is not important for the public key since it must be made accessible to anyone, it is important to ensure its integrity. To be precise, it must not be possible to alter a person's public key. The concept of a Public Key Infrastructure (PKI) has therefore been introduced as a means to generate, distribute and manage 'public key certificates' which are used to bind the identifier of a party to that party's public key [22]. A widely adopted standard for the format of digital certificates is X.509 [51].

There are a number of broad classes of asymmetric cryptographic scheme, including encryption schemes, digital signature schemes, and key agreement schemes. However, only the first two will be described here since they are most relevant to this thesis.

2.3.1.1 Asymmetric encryption

Asymmetric encryption schemes use public keys for encryption and private keys for decryption. The best known algorithm for public key encryption is RSA, which was proposed in 1978 by Rivest, Shamir, and Adleman [74]. Specifications for public key encryption, including the use of RSA, can be found in [35], and also in the emerging international standard [47]. Asymmetric encryption can be used to provide data confidentiality.

2.3.1.2 Digital signature

[50] defines a digital signature as 'data appended to, or a cryptographic transformation of, a data unit, that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery'. In other words, as stated in the introduction to [39], 'digital signature mechanisms are asymmetric cryptographic techniques which can be used to provide entity authentication, data origin authentication, data integrity and non-repudiation services'.

A signature scheme consists of two components, namely a signing algorithm and a verification algorithm. The signing algorithm involves the transformation of the message into a signature, using the signing entity's private key. It should be clear that, for a digital signature scheme to work, there is a need for a verification process, so that it is possible to verify whether a signature on a message was genuinely created by the claimed entity. This verification process takes as input the signature, the message, and the signer's public verification key, and outputs an indication as to whether or not the signature on the message is valid.

Many digital signature schemes have been proposed over the last 25 years (see, for example, [66]). For digital signature standards, see for example [20, 39, 40]. Digital signature schemes can be used to provide data origin authentication, data integrity, and non-repudiation.

2.3.2 Symmetric cryptography

Symmetric cryptography is a cryptographic scheme in which either the same key (secret key) or two keys that can be easily computed from each other are used [66]. That is, in a symmetric cryptographic scheme, the communicating parties are required to share a key which must be kept secret.

There are a number of symmetric cryptographic schemes, including encryption schemes, message authentication codes, and cryptographic hash functions.

2.3.2.1 Symmetric encryption

Unlike asymmetric encryption, symmetric encryption uses a single key for both the encryption and decryption transformation [46]. According to Menezes et al. [66], the encryption is said to be symmetric if, for each associated encryption/decryption key pair, it is computationally 'easy' to determine a decryption key knowing only the encryption key and vice versa.

There are two commonly used types of symmetric encryption scheme, namely stream ciphers and block ciphers. A block cipher is an encryption scheme which breaks up the plaintext messages to be transmitted into strings (called blocks) of a fixed length and encrypts one block at a time [66]. On the other hand, a stream cipher is 'an encryption mechanism such that using a running key or a fresh one-time-pad key stream, an encryption encrypts a plaintext in bit-wise or block-wise manner' [49]. For block cipher and stream cipher standards see, for example, [49, 48]. As for asymmetric encryption, symmetric encryption schemes can be used to provide data confidentiality.

2.3.2.2 Message authentication codes

The second type of symmetric cryptographic technique we describe is the Message Authentication Code (MAC). This mechanism can be used to provide data origin authentication and data integrity services. The originator of data inputs the data to be protected into a MAC function, together with a secret key — the resulting output (a short fixed-length bit string) is known as the MAC. This MAC can then be sent or stored with the data being protected. The verifier of the MAC simply uses the same secret key to recompute a MAC value on

the data, and the data is accepted as valid if and only if the recomputed MAC agrees with the value sent or stored with the data.

There are a number of widely used methods for computing MACs; see for example [66]. Many of them are based on either block ciphers or cryptographic hash functions. There are also standards for MAC computations, notably [41, 45].

2.3.2.3 Cryptographic hash function

Hash functions take a message as input and produce an output referred to as a hash-code, hash-result, hash-value, message digest, or just hash. More formally, 'a hash function is a function which maps strings of bits to fixed-length strings of bits' [42, 66]. It must also satisfy the following three properties.

- it must be computationally infeasible to find for a given output, an input which maps to this output, and
- it must be computationally infeasible to find for a given input, a second input which maps to the same output.
- it must be computationally infeasible to find two different inputs which map to the same output.

Hash-functions form a vitally important part of almost all commonly used digital signature schemes. There are a number of types of hash functions, for example those based on block ciphers, those based on modular arithmetic, and dedicated hash functions. For cryptographic hash function standards, see for example [42, 43, 37, 38].

Part I

Overview of Electronic Commerce Transactions

Chapter 3

Security issues for Internet payments

Contents

3.1 Introduction 36
3.2 Security requirements 37
3.2.1 Issuers and acquirers 37
3.2.2 Merchants 38
3.2.3 Clients 39
3.3 Meeting the security requirements 40
3.3.1 SSL and TLS 40
3.3.2 Secure Electronic Transaction (SET) 41
3.3.3 Visa 3-Domain Secure 42
3.3.4 MasterCard Secure Payment Application 43
3.3.5 Summary 44

The aim of this chapter is to outline security requirements (Section 3.2) for electronic payment transactions, in order to provide the design motivation for the protocols proposed in the following chapters.

In Section 3.3, a number of currently available protocols are described, followed by a review of the security services they provide.

3.1 Introduction

Electronic commerce is growing in significance. Many products, tangible and intangible, are sold over the Internet, with payments typically made by debit or credit cards. In parallel with this, there is an increase in concerns associated with the security of the payment systems used to process online transactions. Probably the main concern of most Internet users relates to the confidentiality of payment card information, since disclosure of this information to a hostile third party could enable that party to make fraudulent transactions at the user's expense. However, security for online transactions is not limited to data confidentiality, but also includes other security services such as authentication, identification, non-repudiation and data integrity.

In a typical debit/credit card payment system there are four parties involved, namely a client, a merchant, an acquiring bank and a card issuing bank [33, 70]. A client, i.e. the cardholder, makes a payment using a card issued by the card issuing bank (issuer) for something purchased from a merchant. The acquiring bank (acquirer) is the financial institution with which a merchant has a contractual arrangement for receiving (acquiring) card payments. The underlying payment model is shown in Figure 3.1.

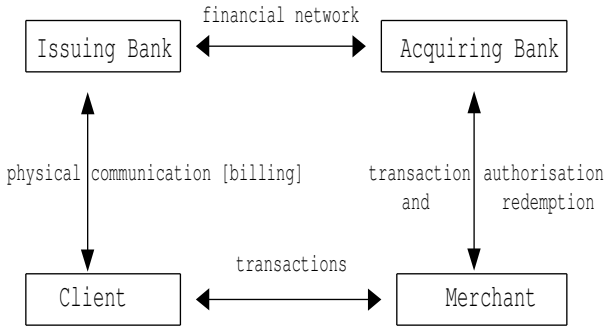


Figure 3.1: Debit/credit card payment system.

In this chapter, we consider the security requirements for each of the four parties involved in an electronic commerce transaction. Some currently proposed electronic commerce protocols are then described and briefly analysed in terms of how well they satisfy these security requirements. Note that the analysis is based on a typical Internet payment transaction, i.e. where a user makes a payment by entering card details through a web interface, the merchant server processes the transaction using a back-end authorisation system, and this enables the transaction to be sent to the acquirer and subsequently to the financial network.

Further discussion on security issues for e-commerce, and details of many of the schemes described here, can be found in recent books on e-commerce, e.g. [2, 26, 27, 33, 70, 82].

3.2 Security requirements

As shown in Figure 3.1, a typical card payment system involves four parties, namely a card issuer, an acquirer, a merchant and a client. The security requirements for each party vary and hence they will be examined individually. However, the requirements for acquirers and issuers are combined since they are both financial institutions, they are both contractually obliged to abide by the rules of the relevant payment system, and it can reasonably be assumed that they have a similar risk model.

3.2.1 Issuers and acquirers

1. **Non-repudiation:** Issuers and acquirers need to ensure that neither clients nor merchants can deny their participation in a transaction (where the transaction may involve a refund from merchant to client). In order to achieve non-repudiation, identity authentication may also be needed.

2. **Authentication:** Client authentication is required for the issuers and acquirers so that they can prove that it is the client who authorised the payment and that he/she is a legitimate cardholder. Otherwise, a client can deny making a transaction and the issuer may end up being liable for refunding the amount to the client. On the other hand, if an electronic transaction is found to be fraudulent, merchants are liable for 'card not present' chargebacks. Therefore, it is important for the acquirer to ensure merchant non-repudiation to prevent them challenging their liability.

3. **Integrity:** It is also important to ensure that once details of a transaction have been confirmed, no one can maliciously modify them. Merchants must not be able to alter the amount that a client has agreed to pay. To be more specific, it should not be possible for a merchant to change the amount after it has been authorised by the card issuer. Similarly, a client must not be able to change the amount that has been authorised.

4. **Replay protection:** A malicious merchant should not be able to use a once authorised transaction to obtain a repeat payment. Additionally, merchants should not be able to use an old transaction to request a new payment authorisation no matter how many similar transactions the client has made with them. Issuers and acquirers need a mechanism to detect if a transaction has been replayed so that they do not authorise an illegitimate transaction.

3.2.2 Merchants

1. **Non-repudiation:** A merchant needs evidence that a customer has agreed to pay the amount associated with a transaction. A merchant also needs to verify that the client is the legitimate cardholder; otherwise, the merchant can be liable for chargebacks. This occurs when a client tells his/her issuer that a particular transaction was not made. The card issuer then

immediately submits a chargeback to the acquirer to recover the amount from the account of the merchant in question. Within a predefined period of time, the merchant can dispute the chargeback by providing evidence of, for example, purchase or delivery. Therefore, it is important for merchants to have non-repudiable evidence of the transaction, i.e. to have client non-repudiation. Furthermore, an issuer should not be able to deny having authorised a payment.

2. **Authentication:** As stated before, merchants need client authentication to make sure that the client is the legitimate cardholder. Moreover, they need to be sure that they are communicating with the genuine acquirer. Otherwise, an adversary may masquerade as an acquirer and authorise an illegitimate transaction.

3. **Integrity:** No one should be able to change the details of a transaction once they have been agreed upon. A merchant will not wish to be credited with payment for less than the amount agreed. In addition, an acquirer or issuer should not be able to modify a transaction that has been authorised.

4. **Replay protection:** A malicious client should not be able to present an old proof of purchase to claim for repeat delivery of goods. Likewise, it should not be possible for an acquirer to claim that a merchant has obtained a payment using an old transaction.

3.2.3 Clients

1. **Confidentiality and privacy:** Transaction confidentiality, especially card information confidentiality, may be the security service of most concern to users. It is important that cardholder account details are kept secret from any party except the issuer, since they are the main basis on which Internet payments are made. Moreover, some users may require confidentiality protection for the nature of their transactions.

2. **Integrity:** As for the other parties, transaction integrity is important to the client. No one should be able to maliciously modify the transaction details once they have been confirmed. Clients will not want an adversary to change a delivery address, the price, or the description of the merchandise after they have agreed a payment.

3. **Authentication:** A client needs to be sure that he/she is dealing with a trustworthy merchant. When shopping on the Internet, it is relatively easy to be lured into visiting a site which appears to sell something but is actually simply collecting card details. Even though a client may have made a purchase from a site before, it is not always obvious whether the page that is being fetched is authentic.

4. **Replay protection:** Clients need a mechanism to ensure that a malicious merchant or an adversary will not be able to reuse previously authorised payments to make a repeat charge.

5. **Non-repudiation:** Clients also require non-repudiation, for example a proof of payment so that no one involved in the transaction can repudiate that a payment has occurred.

3.3 Meeting the security requirements

In this section, we examine various currently available security protocols. A short description of how each protocol works is given, followed by a review of the security services provided.

3.3.1 SSL and TLS

The Secure Sockets Layer (SSL) [25, 33, 85] protocol was launched in 1994 by Netscape, with the primary goal of providing secure communications between

web browsers and web servers. Security services provided include server authentication, data confidentiality, (optional) client authentication and data integrity.

In 1995, the Internet Engineering Task Force (IETF) introduced a similar protocol named Transport Layer Security (TLS) version 1.0 [11]. SSL and TLS are by far the most widely used protocols providing security for transactions made over the Internet. Because of their importance, the whole of Chapter 4 is devoted to a detailed analysis of both SSL and TLS.

3.3.2 Secure Electronic Transaction (SET)

The Secure Electronic Transaction (SET) [76, 77, 78, 79] protocol was developed by Visa, MasterCard and various computer companies to facilitate secure electronic commerce transactions. The protocol provides confidentiality of payment card details, data integrity, authentication of both merchant and cardholder, and the ability to validate or authorise transactions. SET extensively employs public key cryptographic techniques to provide such security services. As a result, one of the most important prerequisites for the protocol is that all the parties involved must have their own distinct key pairs with corresponding public key certificates. Moreover, cardholders have to install special software before they can start using SET.

SET is a complex protocol involving more than ten steps for each transaction, and it is not the aim of this chapter to provide full details of the protocol's operation. In brief, the protocol requires every participating party to cryptographically sign transmitted messages. Sensitive information is also encrypted using a secret session key. One of the most innovative features of SET may be the use of a 'dual signature' which allows merchants to verify the integrity of the order information yet not see the card details.

SET appears to be a well designed protocol that aims to provide a high

level of security for Internet transactions, and that satisfies all the security requirements outlined previously. Unfortunately, SET has not been adopted to any significant extent — indeed, it is not clear whether it will ever become widely used. One of the most important obstacles to SET implementation is that the protocol is so complex that it is difficult and costly for the parties involved to implement it. Moreover, the use of public key cryptographic techniques is costly in terms of computational overhead, performance, and the Public Key Infrastructure needed to support it. As a result, the benefits of security that SET gives may not be sufficient to bring about its adoption.

3.3.3 Visa 3-Domain Secure

The 3-D Secure protocol has recently been developed by Visa [90, 91]. The protocol aims to provide cardholder authentication for merchants using two types of servers: Access Control Servers (ACSs) operated by card issuers and the Visa Directory Server. The cardholder must enroll with his/her issuer ACS before using the service.

When a transaction is to be made, the merchant server queries the Visa Directory Server, using a Merchant Plug-In, to determine whether the cardholder has registered for the authentication service. If so, the Visa Directory Server returns the web address of the appropriate ACS to the merchant. The merchant server then redirects the cardholder browser to that ACS which will in turn prompt the cardholder to authenticate him/herself using a password and/or a Visa smart card. If the authentication process is successful, the ACS redirects the cardholder browser back to the the merchant server, which can then proceed with the traditional authorisation process. The Visa 3-D Secure also employs another server called the Authentication History server to log all the authentication attempts of each cardholder.

It is clear that the protocol does not meet all the security requirements.

Indeed, it does not attempt to do so, since its aim is only to provide cardholder authentication and the associated non-repudiation to reduce or eliminate the risk of card-not-present chargebacks, i.e. where the merchant is required to take liability for a disputed transaction if the card was not physically present at the merchant premises at the time of the transaction. The protocol is also kept simple to facilitate implementation.

3.3.4 MasterCard Secure Payment Application

The MasterCard Secure Payment Application (SPA)[1] is a security solution for securing payments between Merchants and Issuers for card-not-present transactions via the Internet i.e. where the card does not present at the merchant site when the transaction is made. As for Visa 3-D Secure, the scheme is designed to provide cardholder authentication and hence reduce chargebacks. The scheme is proprietary to MasterCard and its specification is available only to the technology vendor community.

The security of SPA is based on MasterCard's Universal Cardholder Authentication Field (UCAF) infrastructure, which is used to communicate authentication information between the cardholder, issuer, merchant and acquirer. As with other protocols, the cardholder must register with his/her issuer before using the service.

A typical SPA transaction begins when the merchant requests payment card details. The checkout page that the merchant sends will include hidden fields which provide the information necessary to generate UCAF data. The issuer SPA server then verifies the cardholder's identity using an authentication method of the issuer's choice. Subsequently, a transaction-specific authentication token, referred to as 'SPA-UCAF', will be generated by the issuer and returned to the cardholder. The cardholder then presents the SPA-UCAF to

[1] details are available at http://www.mastercardintl.com/spa/demo/main.html

the merchant which will use it, in addition to other information, to submit an authorisation request to its acquirer. Once the issuer received the authorisation request, it validates the SPA-UCAF and proceeds with the 'standard' authorisation process.

MasterCard SPA provides two security services, namely cardholder authentication and non-repudiation. As for Visa 3-D Secure, it is too early to tell how great an impact this scheme will have.

3.3.5 Summary

By far, the most widely used protocol to provide security for Internet transactions is SSL/TLS, despite the fact that it was not designed as an e-commerce security protocol. SSL/TLS is used by most Internet merchants as a de facto standard means to provide their customers with assurance that the transaction they are making is being protected.

In contrast, SET, which has been specifically designed to provide security for electronic commerce transactions, has not been widely adopted. In the last couple of years, Visa and MasterCard have each developed their own initiatives to provide simpler and more easily implemented security protocols. However, the future for secure Internet transactions remains unclear.

A lesson to learn from this may be that security comes with cost. It is very important to design a protocol that both works (i.e. is not too complex so that it will be adopted) and provides an acceptable level of security. Therefore, it is the aim of the thesis to design protocols which meet as many security requirements as possible and also remain usable.

Chapter 4

Analysis of the effectiveness of SSL/TLS

Contents

4.1 Introduction 47
4.2 An overview of SSL and TLS 47
4.2.1 Secure Sockets Layer (SSL) 47
4.2.2 Transport Layer Security (TLS) 49
4.3 Analysis 51
4.3.1 Confidentiality 51
4.3.2 Integrity 53
4.3.3 Authentication 53
4.3.4 Non-repudiation 56
4.3.5 Replay protection 56
4.3.6 Summary 57
4.4 Conclusions 57

The aim of this chapter is to analyse the effectiveness of SSL/TLS in terms of how well it satisfies the security requirements outlined in Chapter 3. The chapter also aims to identify security services which are not provided by SSL/TLS. This is needed in order to facilitate the design of protocols proposed later in this thesis.

In this chapter, an overview of how SSL and TLS work is first provided in Section 4.2, followed by the analysis (Section 4.3) and conclusions. It is important to note that much of the material in this chapter has previously been described in [56].

4.1 Introduction

SSL and TLS are currently the most widely used protocols for providing security for the client/merchant Internet link. Therefore, in this chapter we investigate how effectively SSL and TLS serve this purpose, also bearing in mind the security requirements for information handling at the client and merchant sites. It is important to note that the analysis in this chapter is based on a business to consumer (B2C) transaction using a debit/credit card. Further detailed discussion of the security properties of SSL and TLS can be found in recent books, e.g. [33] and [73].

An overview of how SSL and TLS work, including a review of their major differences is first provided. We subsequently examine the effectiveness of the protocols by considering how well they satisfy the security requirements outlined in the previous chapter. The final section summarises and concludes the chapter.

4.2 An overview of SSL and TLS

In order to examine the effectiveness of SSL and TLS in securing electronic transactions, it is first necessary to consider how they work. Therefore, in this section we briefly describe how SSL and TLS operate. More detailed specifications can be found in [11, 25, 73, 85].

4.2.1 Secure Sockets Layer (SSL)

The Secure Sockets Layer (SSL) protocol was launched in 1994 by Netscape, with the primary goal of providing secure communications between web browsers and web servers. Security services provided include server authentication, data confidentiality, (optional) client authentication and data integrity. The following

description of SSL operation is based on SSL 3.0, the current version at the time of writing.

SSL is divided into two layers, namely the SSL handshake protocol and the record layer. The handshake protocol, which is the upper layer, is responsible for initialising and synchronising cryptographic state between the communicating parties. The record layer provides confidentiality and authentication, including protection against replay attacks.

In the most typical case, five main steps are required to establish an SSL connection.

1. The client's browser first sends a ClientHello message to the web server. This message consists of a list of the cipher suites the browser supports, the version of SSL it uses, the data compression methods it can employ, and a challenge string (a random number and a session ID).

2. The server sends back a ServerHello message consisting of the SSL version number, a challenge string, and the selected cipher suite and compression method. Then the server sends a ServerKeyExchange message containing the server's public key information. The server can optionally request the client's certificate for user authentication by sending a CertificateRequest message. Finally the server sends a ServerHelloDone message to indicate that it has finished with its initial negotiation messages.

3. The client sends its certificate (if requested by the server) in a Certificate message. This is followed by a ClientKeyExchange message which contains key information, i.e. the 'premaster secret' that will be used as a seed to generate the master secret and keys subsequently used for encryption. The key information is encrypted with the server's public key. If client identification is required, a CertificateVerify message must be sent to prove that the client has the private key corresponding to the public key in the

certificate. The CertificateVerify message essentially contains a signed hash of the key information and all previous SSL handshake messages exchanged so far.

4. The client sends a ChangeCipherSpec message to indicate the starting point of a protected channel, followed by a ClientFinish message which contains a hash of the handshake messages exchanged by the systems and the key information. The ClientFinish message is encrypted and authenticated using the algorithms in the negotiated cipher suite. Note that ChangeCipherSpec messages are not considered as handshake messages and thus are not included in the hash.

5. The server sends back a ChangeCipherSpec message and a ServerFinish message which are similar to the messages with corresponding names sent by the client.

The next section briefly explains how TLS operates, and the differences between SSL and TLS.

4.2.2 Transport Layer Security (TLS)

In 1995, the IETF introduced a similar protocol called Transport Layer Security (TLS) version 1.0 [11, 73, 85]. Operationally, SSL and TLS work in a very similar way. However, there are some significant differences, as follows.

- The protocol version appearing in SSL messages is 3.0 while for TLS it is 3.1.
- TLS offers 11 more alert message types than SSL.
- For message authentication, SSL combines key information and application data in an SSL-unique fashion. By contrast, TLS employs a widely

used and standardised method for computing a Message Authentication Code (MAC), i.e. the HMAC technique [45, 60], to provide message authentication.

- TLS employs a simpler CertificateVerify message. The signed information includes only the handshake messages exchanged so far. However, in SSL, the information consists of a two-round hash of the handshake messages, the master secret and the padding.
- TLS employs a pseudorandom function (prf) to generate key materials using a master secret, a label in which the name of the key is specified, and a seed as initial inputs. SSL, by contrast, uses a complex and rather ad hoc procedure to generate key materials.
- The Finish message of SSL is created in an ad hoc way whereas it is generated by a pseudorandom function in TLS.
- The cipher suites offer in SSL includes Fortezza, while in TLS it does not.

The differences are summarised in Table 4.1 [85].

Table 4.1: Differences between SSL and TLS

Attributes	**SSL v3.0**	**TLS v1.0**
Protocol version in messages	3.0	3.1
Alert message types	12	23
Message authentication	ad hoc	standard
Key material generation	ad hoc	prf
CertificateVerify message	complex	simple
Finished message	ad hoc	prf
Baseline cipher suites	Fortezza	no Fortezza

Although these differences between the two protocols exist, in the remainder of the thesis both protocols will be referred to as SSL/TLS unless explicitly stated otherwise.

4.3 Analysis

This section analyses the effectiveness of SSL/TLS as a method for securing electronic payments. This is achieved by examining how well it satisfies the security requirements described in Section 3.2. Since SSL/TLS was designed to protect communications between web clients and web servers, the analysis will only address interactions between clients and merchants (see Figure 3.1). Clearly, SSL/TLS cannot, by itself, address any security issues relating to interactions between other pairs of parties. In any event, interactions between issuers and cardholders (mainly relating to card issue and billing) are outside the scope of this thesis. Similarly, we can assume that interactions between issuers and acquirers are addressed in the context of securing the financial network, and hence are again outside the scope of this thesis.

As far as the acquirer/merchant interactions are concerned, security services can be provided by separate security mechanisms operating to protect communications between the merchant server and the acquirer host. Such mechanisms would typically be managed by the acquirer.

4.3.1 Confidentiality

SSL/TLS protects transaction confidentiality by using symmetric encryption. The encryption algorithm to be used in any particular connection depends on the cipher suite negotiated in the handshake protocol. Although SSL/TLS protects the confidentiality of transferred data against interception attacks, there remain some risks which need to be examined.

Since SSL/TLS was designed to provide confidentiality between web clients and web servers, transaction information is protected only while it is being transmitted. Therefore, information such as clients' account details and addresses are available to the merchant. The users thus have to rely on the security of the

merchant's web server. If someone succeeds in penetrating the merchant server, potentially large numbers of user account details could be compromised.

Another issue is that the US federal regulations have severely restricted the export of strong encryption technology. Until recently, this meant that popularly available SSL/TLS implementations only used relatively short key lengths unless both the communicating parties were within the US or Canada [6].

In October 2000 the US export restrictions were relaxed to allow SSL/TLS to use longer key lengths when the parties are in the EU or one of eight other countries, namely Australia, New Zealand, Czech Republic, Hungary, Japan, Norway, Poland and Switzerland [6]. However, risks clearly remain for clients and merchants in countries outside the scope of this new exemption. Moreover, this new exemption still only permits 56-bit secret keys, for which exhaustive key searches can be performed [24]. However, it is probably hard to imagine a circumstance where it would be worth the effort of breaking such a key given that it will only reveal the details of a single transaction.

SSL/TLS also protects the confidentiality of information regarding the nature and value of the transaction whilst this information is transmitted across the Internet. Of course, SSL/TLS cannot offer any protection for the confidentiality of this data whilst it is stored at the merchant — in any case such protection is probably meaningless since the merchant will clearly need to know this information. However when using SSL/TLS for security the merchant will also know the account details of the purchaser, and hence can use these to link transactions and build profiles of user purchasing behaviour. If required, consumer anonymity could possibly be achieved by using alternative payment mechanisms — see, for example, [84]. However, if the merchant needs the shipping address to deliver the purchased goods, then achieving purchaser anonymity will be rather difficult!

4.3.2 Integrity

As for confidentiality, SSL/TLS provides integrity protection for transferred data only. Consequently, if an adversary succeeds in compromising either the merchant server or the client PC, it would be possible for them to modify the information stored. As a result, such information will not be helpful if there is a dispute. Moreover, for the same reasons, SSL/TLS offers no protection against modification of transaction information by corrupt merchants or clients.

4.3.3 Authentication

We next consider how SSL/TLS supports the required authentication services — we subdivide this discussion into considerations of merchant authentication and client authentication.

4.3.3.1 Merchant authentication

The SSL/TLS protocol uses the server certificate as the basis of server authentication. The client verifies the server by verifying its ability to decrypt information encrypted using the server's public key. Nevertheless, there remain some risks of server masquerade. One possibility is by means of a 'man in the middle attack'. Such an attack can be launched relatively easily by using a sniffing application such as dsniff[1] to intercept the communications between two entities at the stage of SSL/TLS initialisation.

Briefly, the man in the middle attack operates as follows. After successfully inserting themselves in the middle of the communication, the attacker simply fetches the page requested by the client from the genuine server. Upon receipt of the requested page, the malicious server returns the spoofed page to the client.

[1] `http://www.monkey.org/~dugsong/dsniff/`

The spoofed page is the page containing rewritten URLs of the links on the page. This enables the attacker to maintain a compromised link between the client and whichever server is visited, since if the client clicks on any links on the page, the request will go through the attacker and the process repeats [19, 27, 95].

An alternative means of launching this attack would be to use 'web spoofing' instead of a sniffing application. However, in this latter case, the user must first be lured into visiting the attacker's page [27], and from then on every subsequent site can be modified by the attacker. This can be implemented relatively simply merely by requesting the genuine web page. Upon receipt of the page, the malicious server prefixes the URL of the genuine site with its URL and returns the spoofed page to the user. In other words, the attacker acts as a proxy for the user to visit other web sites, thereby obtaining the ability to read and modify information.

If an SSL/TLS connection is in use, the attacker simply establishes two secure connections, one with the client and the other with the server. Thereby, he/she can read and modify the information sent between the two parties as well as convince them that they are communicating via a secure channel. However, since SSL/TLS requires server authentication, the attack should be prevented by the client examining the certificate or the URL of the page, since the certificate will show the URL of the attacker instead of the genuine server. However, the attacker can control the appearance of the URL to the client by using scripting techniques — moreover, users will often neglect to check such details, since web browsers tend to be designed to make things as easy as possible and minimise the work for the user. Hence, although the server authentication in SSL/TLS prevents such attacks in theory, the practical situation is rather different.

The second place that the attack can be revealed is the status line. When the user points a mouse at a rewritten link, the status line shows the URL to which the link leads. At the moment of loading a requested page, the prefixed

URL also shows in the status line. Nevertheless, scripts such as JavaScript and ActiveX can be used to change what should appear in both the location line and status line [19].

The final means for the user to detect the attack is the source of the page. A user can see the source of a page by choosing `View` ⟶ `Source` in Internet Explorer and `View` ⟶ `Page Source` in Netscape Navigator. Fortunately the attacker cannot hide the rewritten links in the source code of the page. However, the source code of the page is written in HTML and, for the vast majority of users, will be completely incomprehensible. Indeed, very few web surfers are even aware that they can look at the source of every page they view.

Remedies suggested in [19] are to disable JavaScript, make sure that location line is always visible, and take note of the URLs displayed on the location line. However, these solutions depend heavily on individual users strictly following all the suggestions. In other words, it will not be very useful to disable scripts but yet never look at the location line or status line. Moreover, by disabling scripts, some sites may not function properly and users can lose the interactive features inherent in web browsing.

4.3.3.2 Client authentication

While server authentication is mandatory, client authentication is an optional part of SSL/TLS. If client authentication is to be provided, a public key pair and certificate for the client are required. However, most clients do not have key pairs and public key certificates. Even if they do, in most cases the key pair is stored in their PC. This gives rise to a further threat, since anyone who has access to the user's PC and knows the corresponding PIN/password to decrypt the private/secret keys may gain the ability to make transactions on behalf of the user. This is especially the case where the merchant uses the client identity to access records containing user personal information including mailing address

and account details.

Finally, even if the user does possess a client certificate, it may not be of any use to the merchant unless it has been issued by a CA trusted by all participants. This is because a general purpose client certificate will typically only contain a user name and an email address. There will thus be no secure linking between the certificate and the cardholder details being used by the merchant (i.e. the account number), since it is typically possible for a user to obtain a general purpose client certificate in any user's name.

4.3.4 Non-repudiation

Although SSL/TLS uses signatures for session establishment, all protection of communicated data is achieved using symmetric cryptographic techniques. Hence SSL/TLS provides no non-repudiation services; that is, neither client nor merchant has any cryptographic evidence that a transaction has taken place.

4.3.5 Replay protection

SSL/TLS provides protection against third party replay attacks by including random numbers in the handshake protocol. However, since SSL/TLS simply provides a secure means of communication between clients and servers, and provides no long-term 'evidence' regarding transactions (as discussed in Section 4.3.4), SSL/TLS does not provide any protection against manipulation (including replay) or repudiation of transaction information by merchants or clients.

4.3.6 Summary

The security services that SSL/TLS provides are summarised in Table 4.2, where they are mapped against the security requirements identified in Section 3.2.

Table 4.2: Security services provided by SSL and TLS

Security Requirements	**SSL and TLS**
Issuers and acquirers security requirements	Not Applicable
Merchant/Client non-repudiation	Not Provided
Merchant authentication	Provided
Client authentication	Optional
Data integrity	En route protection
Replay protection	Third party protection
Data confidentiality	En route protection

4.4 Conclusions

Although each party involved in an electronic transaction has a different risk model, they share some fundamental security requirements. These are confidentiality, authentication, non-repudiation, integrity and replay protection.

Because of the very nature of the protocol, SSL/TLS provides confidentiality and integrity only while the information is being transmitted. Once the information has reached its destination, SSL/TLS offers no protection, and any security measures depend on the choices of the communicating parties. As a result, there are risks of information being compromised if either side of the communication has been penetrated.

SSL/TLS only mandates server authentication. Therefore, even if SSL/TLS is used to protect electronic transactions, the merchant is not protected against a criminal impersonating a genuine user (using stolen account information). Moreover, SSL/TLS does not provide either clients or merchants with protection against repudiation. This makes transaction information stored by either party

of limited value in the event of a dispute.

SSL/TLS provides only partial protection against replay attacks. It prevents a third party using an intercepted SSL/TLS message. However, it does not prevent corrupt merchants or clients re-using a transaction.

From the analysis, two issues are worth noting. Firstly, since SSL and TLS were not designed specifically to secure payments over the Internet, not surprisingly they do not satisfy all the security requirements for electronic transactions. It is important that e-commerce clients and merchants do not have a false sense of security when using them. Secondly, it would be interesting to see how electronic transaction security could be enhanced by combining use of SSL/TLS with certain additional simple security features, as an alternative to accepting the significant cost of adopting a more complex solution such as SET, or use of a solution with relatively limited security functionality, such as Visa's 3-D Secure. Indeed, this latter issue is the primary aim of this thesis.

Part II

Enhancing the Security of E-Commerce

Chapter 5

Using EMV cards for e-commerce transactions

Contents

5.1 Introduction 62
5.2 An overview of EMV 64
5.2.1 Card authentication 69
5.2.2 Cardholder verification 70
5.2.3 Application cryptograms 71
5.3 Using EMV cards for e-commerce transactions . 72
5.3.1 System architecture 72
5.3.2 Transaction processing procedures 74
5.3.3 Security services 79
5.4 Threat analysis 80
5.4.1 Threats to the cardholder environment 81
5.4.2 Threats at the Merchant Server 82
5.5 Use of trusted card readers 83
5.5.1 System architecture 83
5.5.2 Transaction process 85
5.5.3 Threat analysis 85
5.6 Advantages and disadvantages 87
5.7 Related work 89
5.8 Conclusions 90

The aim of this chapter is to propose a way to use EMV compliant cards to enhance the security of Internet payments. In Section 5.2, the way in which an EMV card transaction is processed is described. This is followed in Section 5.3 by a specification of the new protocol. An analysis of threats that may arise to the proposed protocol is then given in Section 5.4. A way in which a trusted card reader can be used in conjunction with the proposed protocol is examined in Section 5.5, followed, in Section 5.6, by a review of the advantages and disadvantages of the proposed protocols.

Much of the material in this chapter has previously been described in [58].

5.1 Introduction

In this chapter, we are concerned with e-commerce transactions in which a consumer makes a payment using a debit/credit card. As was discussed in Chapter 4, in such transactions the communications link between the consumer PC and the merchant server is typically protected against eavesdropping using SSL/TLS. Although SSL/TLS has become a de facto standard means to secure an electronic transaction made over the Internet, as discussed in Section 4.3 it does not satisfy all the security requirements. To be precise, it only provides security for the communications link between the consumer PC and the merchant server. As a result, there are a number of security risks in such use of SSL/TLS, as pointed out in [82]. One of these is the lack of client authentication and the associated lack of client non-repudiation. Even though SSL/TLS offers client-side authentication, it is optional and rarely used. Consequently, it is not easy to verify if the client is the legitimate cardholder and there is no way to determine if the client actually has the card. A malicious user, who may have obtained a card number by some means, can then use it to make payments over the Internet at the expense of the legitimate cardholder.

Meanwhile, for transactions taking place at the Point of Sale (POS), a variety of frauds are possible against debit/credit card transactions. In recent years this has led the major card brands to develop an industry standard means of employing IC cards to replace the existing magnetic stripe cards, with a view to both reducing fraud and reducing the costs associated with online transaction authorisation at the POS. This collaboration between Europay, MasterCard and Visa resulted in the EMV card/terminal specifications, the latest version of which are known as EMV 2000 [13, 14, 15, 16]. These specifications are now managed by a jointly owned organisation known as EMVCo. The EMV specifications standardise interactions between a debit/credit IC card and a terminal. Nevertheless, the EMV specifications were not designed to provide

security for electronic transactions made over the Internet.

In an annex to Book 3 of EMV 2000 [15], the use of an EMV card and the Secure Electronic Transaction (SET) protocol [76, 77, 78, 79] to conduct an e-commerce transaction is defined. However, SET (which provides security for an entire e-commerce transaction) has not been adopted to any significant extent. Although the integration of EMV and SET removes some of the issues with SET, notably it simplifies user registration, there still remain large obstacles to its adoption (see Section 3.3.2).

As a result, in this chapter an alternative way in which the growing use of EMV IC cards can be utilised to enhance the security of e-commerce transactions is proposed. The goal is to design a scheme whereby EMV cards can be used to enhance e-commerce transaction security, and hence reduce fraud, whilst imposing minimal overheads on the involved parties.

In this chapter, an overview of the EMV payment system is first provided in Section 5.2, followed by a description of the proposed technique to use EMV cards for enhancing the security of Internet electronic transaction processing. Threats to the proposed scheme are then examined in Section 5.4.

It is possible that 'trusted' card readers, with simple user interfaces such as a PIN pad and a small display, could become more common in the future. Section 5.5, therefore, examines how the proposed scheme might operate if such card readers were used. The security implications of the use of a card reader in the protocol are also discussed. Finally, Section 5.6 contains an analysis of advantages and disadvantages of the protocols.

5.2 An overview of EMV

The debit/credit card payment system, in which there are four major parties, namely a client, a merchant, an acquiring bank and a card issuing bank, is the model underlying the EMV 2000 system [13, 14, 15, 16, 70]. The payment model is shown in Figure 3.1.

The EMV transaction process involves the following steps. Note that the order of the steps is not completely fixed; for example, cardholder verification can precede data authentication.

1. When the IC card is inserted, the terminal reads application data from the card and performs Terminal Risk Management. Terminal Risk Management provides positive issuer authorisation for high-value transactions and ensures that transactions initiated from IC cards go online periodically to prevent threats that might be undetectable in an offline environment. The exact frequency with which transactions are forced online is not specified in EMV 2000, and is thus acquirer and brand specific.

2. The Data Authentication process enables the terminal to verify the authenticity of the card. There are two options for Data Authentication, namely Static and Dynamic Data Authentication, where Dynamic Data Authentication, unlike Static Data Authentication, requires the card to possess its own signature key pair and to compute signatures. Not all EMV cards are capable of performing Dynamic Data Authentication.

3. After successful Data Authentication, the Processing Restrictions are performed to determine the compatibility of the terminal and IC card applications.

4. Cardholder authenticity is verified by PIN entry. The PIN verification process can be either online to the issuer or offline to the card.

5. After successful Cardholder Authentication, the terminal performs Terminal Action Analysis, which results in a decision as to whether the transaction should be approved offline, declined offline, or an online authorisation performed.

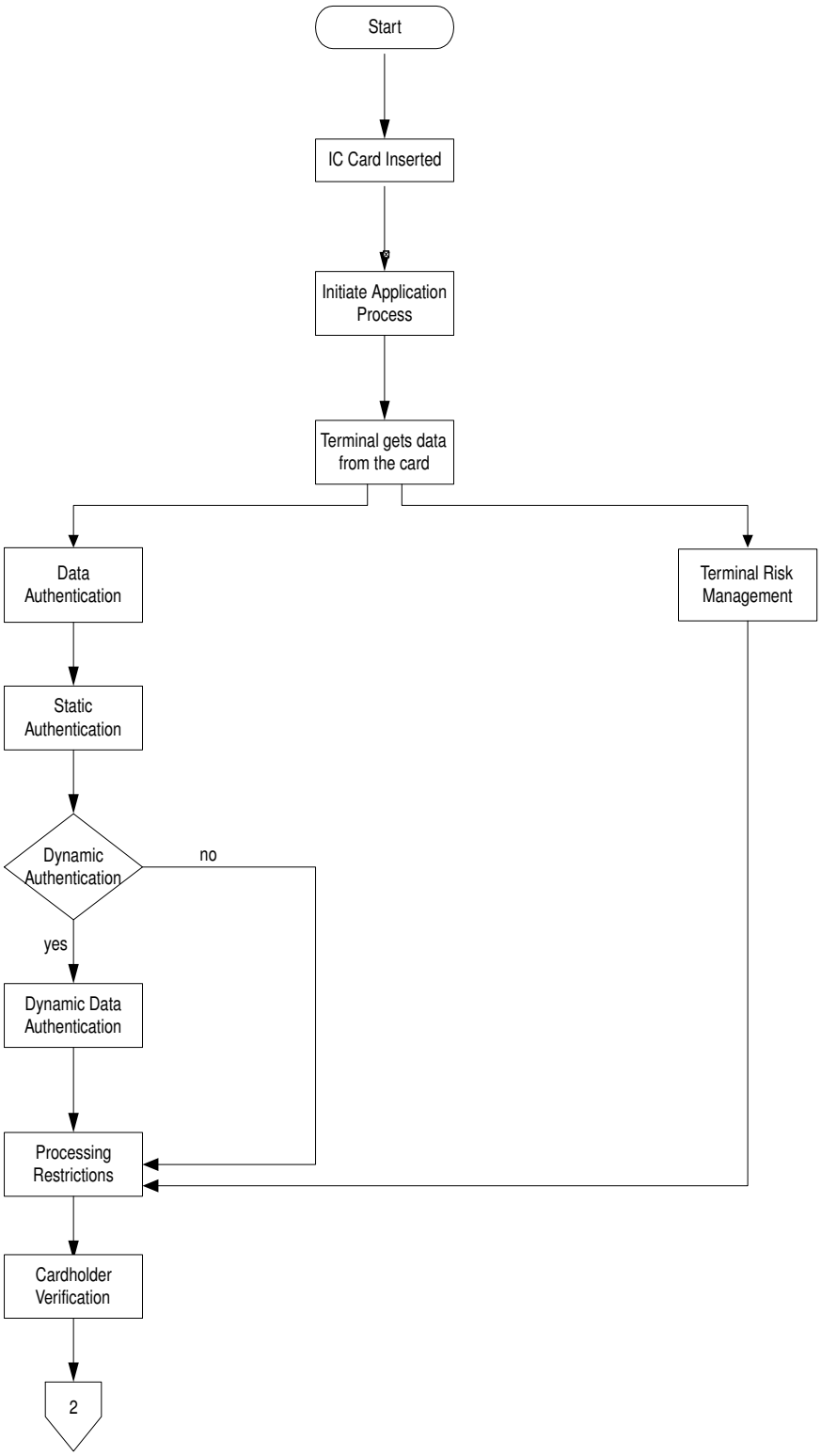
6. The IC card then performs Card Risk Management to protect the card issuer against fraud or excessive credit risk. Details of card risk management algorithms are specific to the issuer and are not specified by EMV.

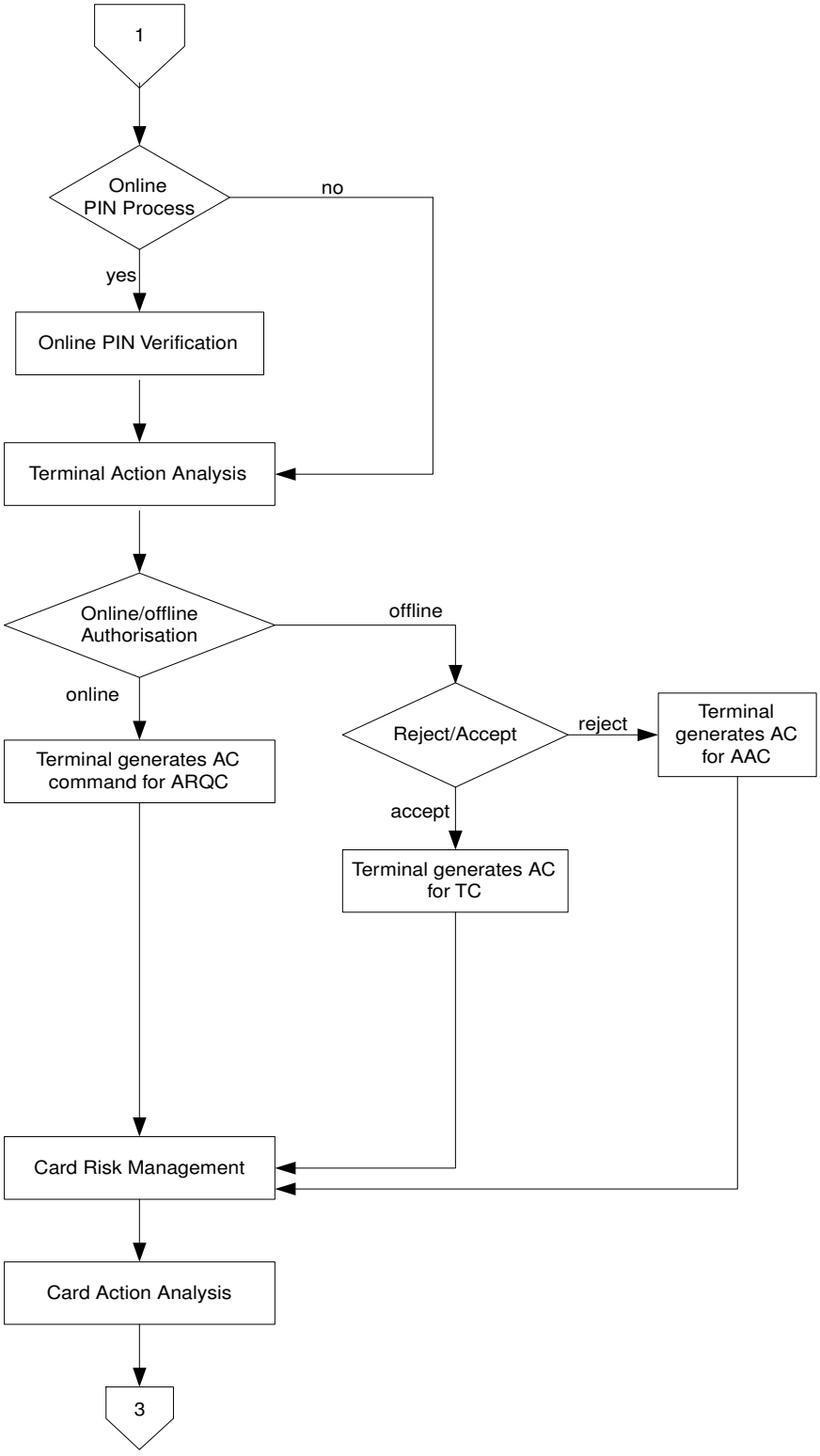
7. The IC card performs Card Action Analysis to further decide whether the transaction will be processed offline, or will need online authorisation (if either Terminal Action Analysis or Card Action Analysis decide that online authorisation is required then this must take place). If the decision is offline processing, the transaction is processed immediately. If the transaction is to be processed online, Online Processing will be performed to ensure that the issuer can review and authorise or reject transactions that are outside acceptable limits of risk defined by the issuer, the payment system, or the acquirer. Issuers can also perform Script Processing, enabling command scripts to be sent to the card by the terminal to perform functions that may not necessarily be relevant to the current transaction but are important for the continued functioning of the card application.

8. The final process is Completion which ends the processing of a transaction.

The EMV process can be presented in flow chart form, as shown in Figure 5.1. In essence, the EMV scheme supports both cardholder authentication by PIN entry and IC card authentication through Static or Dynamic Data Authentication. Therefore, an unscrupulous user will find it hard to make a fraudulent transaction without possessing the actual card and the corresponding PIN.

We next focus on the security-related interactions between IC card and terminal. This is of particular interest here since, in the scheme proposed below,


Start
IC Card Inserted
Initiate Application Process
Terminal gets data from the card
Data Authentication
Terminal Risk Management
Static Authentication
Dynamic Authentication
no
yes
Dynamic Data Authentication
Processing Restrictions
Cardholder Verification
2


1
Online
PIN Process
no
yes
Online PIN Verification
Terminal Action Analysis
Online/offline
Authorisation
offline
online
Reject/Accept
reject
Terminal
generates AC
for AAC
Terminal generates AC
command for ARQC
accept
Terminal generates AC
for TC
Card Risk Management
Card Action Analysis
3

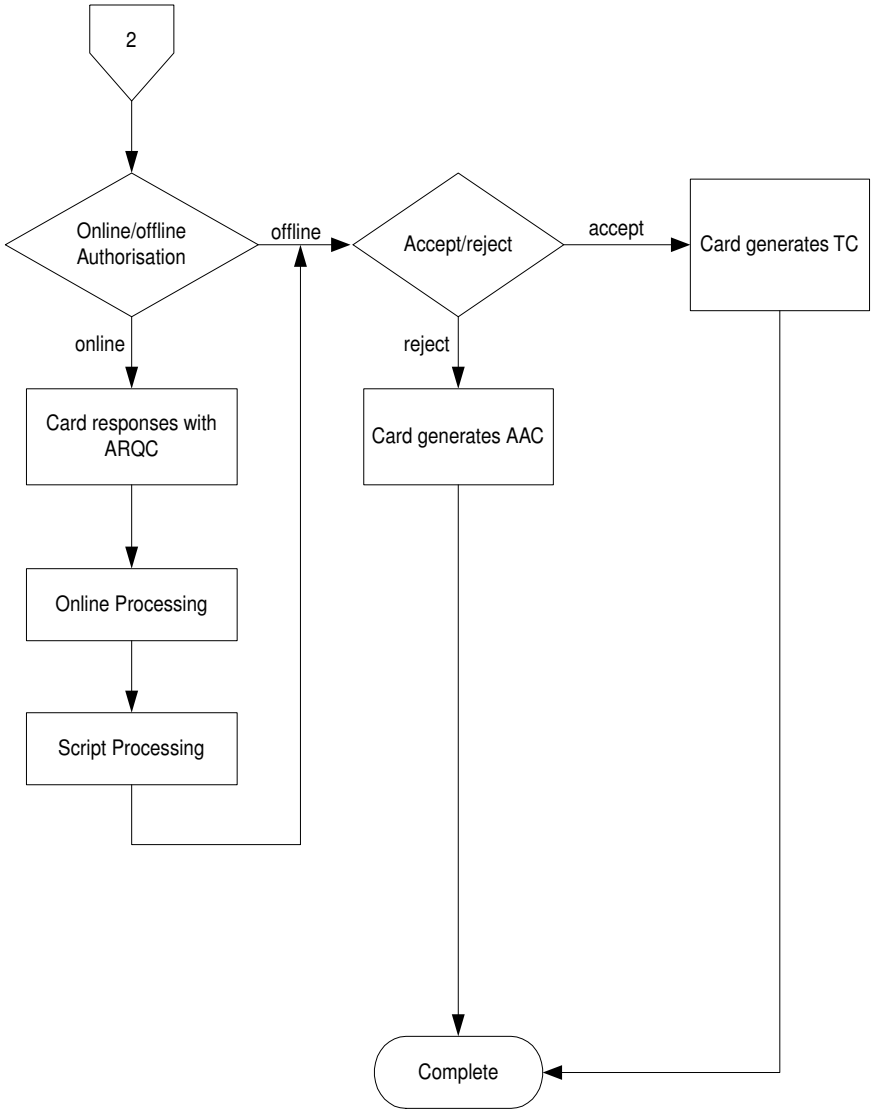


Figure 5.1: EMV card process flowchart.

the user PC plays the role of the merchant terminal and interacts with a merchant server across the Internet. The merchant server communicates with the acquirer, e.g. using the same interface as is currently used for merchant terminal-acquirer communications. An acquirer and an issuer typically communicate via a brand-specific network, the details of which are outside the scope of this thesis.

5.2.1 Card authentication

The EMV specifications allow card authentication to be either offline or online. The decision regarding whether to perform online authentication depends on the capability of both the IC card and the terminal, and can be made at the time of the transaction.

The card authentication process is performed using two kinds of data authentication: static data authentication and dynamic data authentication. A signature scheme with message recovery, compliant with ISO/IEC 9796-2 [44], is used to support both data authentication schemes. Static data authentication is mandatory whereas dynamic data authentication is optional.

5.2.1.1 Static data authentication

Static data authentication involves the terminal verifying the integrity of static data signed by the card issuer and stored in the IC card. The card does not need its own signature key pair to support this type of authentication.

Static data authentication is supported by a two-level key management hierarchy. The top-level certification authority (CA) is operated by the card scheme, i.e. Visa or MasterCard. This CA certifies the issuer public keys. The static data is signed using an issuer private key and stored in the card, along with the CA-signed certificate for the appropriate issuer public key. A terminal with a

trusted copy of the CA public key can then verify the issuer public key certificate and hence can verify the signature on the static data, thereby verifying the IC card.

5.2.1.2 Dynamic data authentication

Like static data authentication, dynamic data authentication is based on digital signatures, although in dynamic data authentication the card has its own key pair. The terminal sends an Internal Authenticate Command (IAC), including an unpredictable number, to the card. The card then digitally signs the IAC data and returns the signature to the terminal. The terminal verifies the signature to authenticate the dynamic data and hence the card.

Dynamic data authentication is supported by a three-level key management hierarchy. The first and second level CAs are operated by the card scheme and issuer respectively, and the card public key is certified by its issuer. In order to verify the signed data, the terminal needs to contain the top-level CA public key to verify the issuer public key certificate. The issuer public key is then used to verify the IC card public key certificate. The terminal can then verify the card signature.

5.2.2 Cardholder verification

The cardholder is verified using a PIN. The EMV specifications require every EMV card to possess a method to limit the number of unsuccessful PIN tries.

PIN verification may occur offline to the IC card, or online to the card issuer (or a third party acting for the card issuer). For offline verification the PIN may be asymmetrically encrypted between the PIN pad and the IC card. Either a key pair assigned especially for PIN encryption or the key pair used in dynamic

data authentication can be used for encryption. In either case, the card public key is first retrieved by the PIN pad or a secure terminal component. The IC card also sends a random number to be concatenated with the entered PIN. The result is encrypted and sent back to the card. The card then decrypts the ciphertext, checks the random number and verifies whether the recovered PIN matches the one stored in the card.

5.2.3 Application cryptograms

Transaction message integrity and origin authentication are guaranteed by the use of Application Cryptograms (ACs), generated by the IC card and issuer using shared-secret-based Message Authentication Codes (MACs). There are four types of ACs, namely Transaction Certificates (TCs), Application Authentication Cryptograms (AACs), Authorisation Request Cryptograms (ARQCs) and Authorisation Response Cryptograms (ARPCs). If the transaction is approved offline, the card generates a TC which is stored by the merchant and passed to the acquirer as part of the clearing process. If the transaction is declined offline, then an AAC is generated. If the transaction needs to be approved online, the card generates an ARQC which will be sent to the issuer. The issuer then responses with an ARPC indicating whether the transaction should be approved or declined. As in the offline case, if the transaction is approved by the issuer, the card computes a TC; otherwise, an AAC is computed.

As mentioned above, ACs are cryptographically protected using a MAC. The issuer and card share a long term secret key MK_{AC} known as the card AC master key. This master key is used to generate an AC session key (SK_{AC}) which is used to compute the AC MACs. The session key SK_{AC} is computed as a function of MK_{AC} and diversification data R; the value of R must be different for each session key generation to prevent replay attacks. Note also that, to avoid the issuer having to store the master key MK_{AC} for every card,

each such master key is derived from an issuer master key MK_I. This key derivation takes as input the Primary Account Number (PAN) and the PAN sequence number.

5.3 Using EMV cards for e-commerce transactions

We now describe one way in which an EMV-compliant IC card can be used to conduct remote transactions. The system architecture is described, as are the transaction processing procedures and how security services are provided.

5.3.1 System architecture

The e-commerce payment system we describe employs five main components: an EMV card, an IC card reader, the Cardholder System, the Merchant Server, and the Acquirer. The system is illustrated in Figure 5.2.

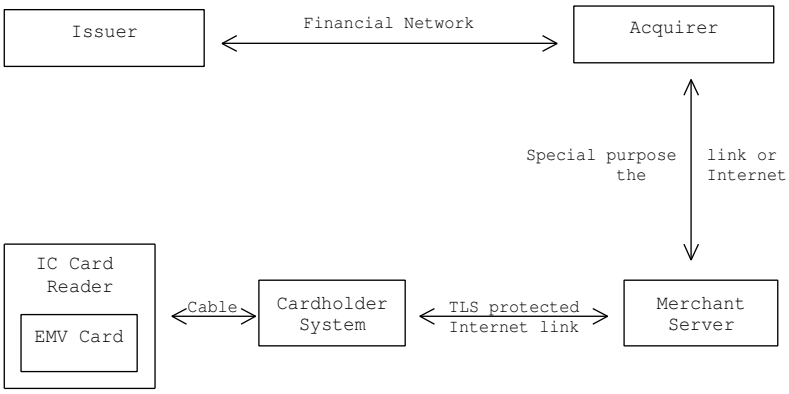


Figure 5.2: Remote EMV system architecture

5.3.1.1 EMV card and IC card reader

The tasks of the EMV card are the same as those given in the EMV Specifications. The card is assumed to be a completely 'standard' dynamic-data-authentication-capable EMV card — indeed, the scheme is designed so that existing EMV cards can be used to support e-commerce security without any modification. The EMV card interacts with a combination of system components, i.e. the card reader, the Cardholder System and the Merchant Server, just as it does with a merchant POS terminal.

The IC card reader, which can include a PIN pad, is required for interactions between the cardholder and the card, and between the card and the Cardholder System. Physical requirements for this device are similar to those in [13].

5.3.1.2 Cardholder System

The Cardholder System is the combination of hardware and software required to interact with the cardholder, the IC card, and the Merchant Server. The Cardholder System is assumed here to be a combination of a user PC and special purpose software which could, for example, be either a small program distributed with the IC card by the issuer or, to make system installation maximally transparent, a web browser applet. The source of cardholder system software is not an issue we address here, but it might be the card issuer, the card brand, or an associated party. The Cardholder System is jointly responsible, along with the IC card reader and the Merchant Server, for performing the tasks of the terminal defined in the EMV specifications.

5.3.1.3 Merchant Server and Acquirer

The Merchant Server is the component that interacts with the Cardholder System to support electronic payments. The Merchant Server and Cardholder System communicate using the Internet. In our protocol, we utilise the fact that in today's electronic transactions the Internet link is typically protected using SSL/TLS. Therefore, we assume throughout that SSL/TLS is used wherever necessary to provide merchant server authentication, data confidentiality and integrity for the Cardholder System/Merchant Server communication link.

The Merchant Server also interacts with the Acquirer. The choice of the communication link between the two is not an issue here. However, it could be the Internet or a special purpose link provided by the acquirer. As specified above, the Merchant Server, the Cardholder System and the IC card reader collectively fulfill the role of the EMV merchant terminal. The Acquirer interacts with the issuer via the financial network to support transaction authorisation. To support static and dynamic data authentication, the Merchant Server needs a trusted copy of the CA public key to enable it to verify issuer public key certificates.

5.3.2 Transaction processing procedures

In this section, we describe the processes necessary to complete a payment transaction. The protocol for using an EMV card for an e-commerce transaction is also described. The decision about which purchase to make is outside the scope of this thesis — we simply assume that the cardholder and the merchant wish to perform a specified transaction.

The transaction flow is shown in Figure 5.3. In the protocol description, $X \| Y$ denotes the concatenation of data items X and Y. Other terms are defined as they arise in the text below.

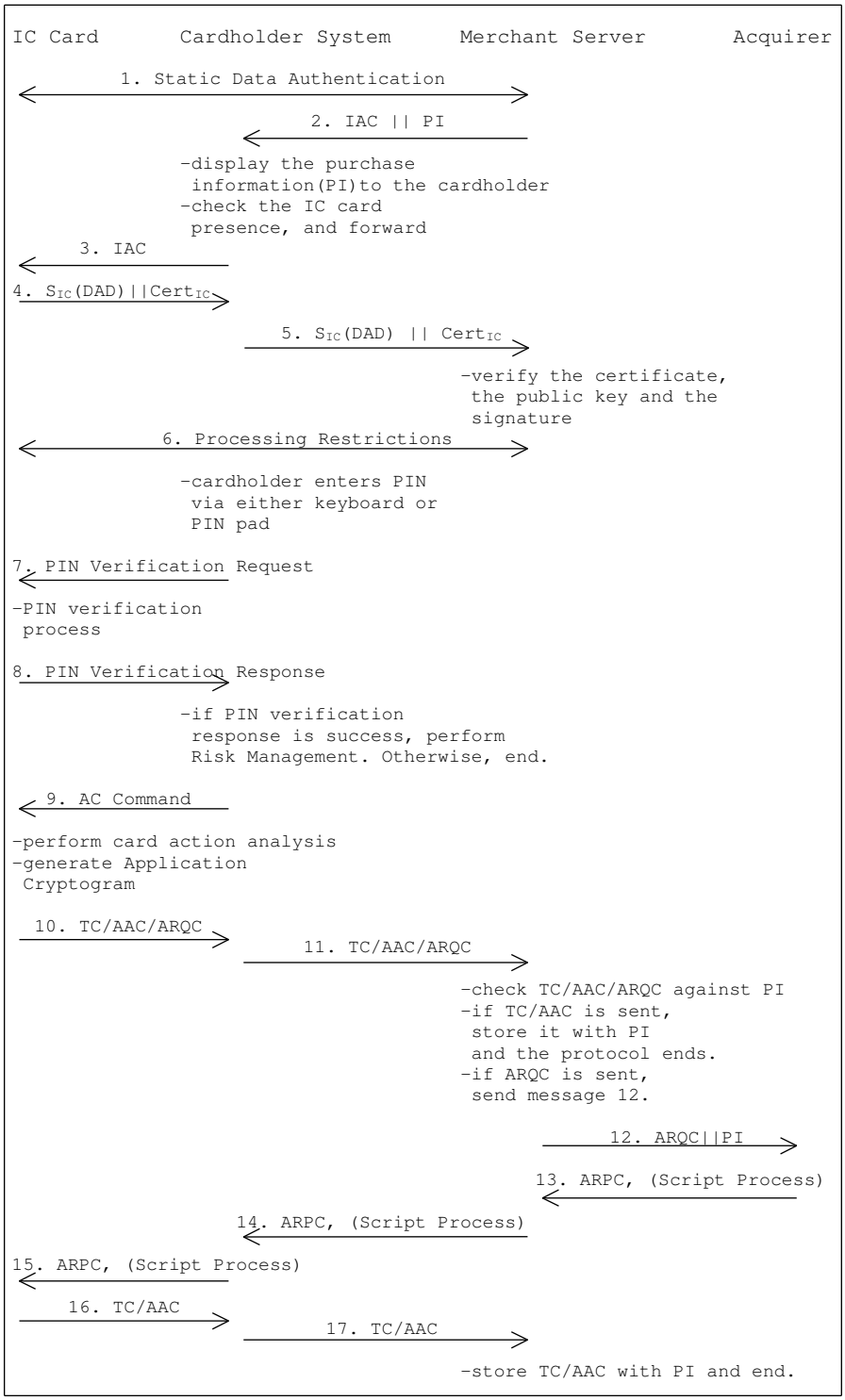


Figure 5.3: Transaction flow for remote EMV

5.3.2.1 Card authentication and processing restrictions

A transaction begins after the cardholder has decided to make a purchase. The Merchant Server and the EMV card first perform static data authentication (step 1 in Figure 5.3). In this process, an authentication request message is sent from the Merchant Server to the Cardholder System and thence to the IC card. A Static Application Data (SAD) message is subsequently sent back from the IC card to the Cardholder System and thence to the Merchant Server. The Merchant Server then verifies the issuer public key certificate using its copy of the CA public key. The issuer signature on the Static Application Data (SAD), sent by the IC card, is then verified. Data communicated between the Merchant Server and the card are sent and received via the Cardholder System. Note that the data are protected by SSL/TLS protocol while they are being transmitted over the Internet.

After successful static data authentication, the Merchant Server generates a random number, which is sent in the IAC, and constructs the purchase information (PI), which may contain a description of goods, the price, the date, and a transaction identifier. The Merchant Server then sends $IAC \| PI$ to the Cardholder System (step 2). The IAC is required to initiate the computation of the Signed Dynamic Application Data by the IC card. According to the EMV 2000 specifications [15], the command consists of the authentication-related data which is proprietary to an application. However, it is mandatory for the IAC to contain an unpredictable number [14].

Upon receipt of the above message, the Cardholder System displays the purchase information to the cardholder and forwards the IAC to the card (step 3). The IC card then computes the Signed Dynamic Application Data, $S_{IC}(DAD)$, where $S_A(X)$ represents the signature on data string X using the private key of entity A, and where DAD is the Dynamic Application Data containing the random number sent in the IAC. The IC card then sends the signature to the

Cardholder System along with the IC card's public key certificate Cert_{IC} (step 4), where Cert_A denotes a certificate for the public key of entity A. In step 5, the Cardholder System sends $S_{IC}(DAD)\|\text{Cert}_{IC}$ to the Merchant Server. Since the issuer public key is used and verified in the static data authentication process, the Merchant Server now only needs to verify the card certificate and then the signature $S_{IC}(DAD)$.

After successful card authentication, Processing Restrictions (step 6) is performed. Data Authentication and Processing Restrictions take place between Merchant Server and card, and the Cardholder System simply forwards messages. Note that these processes remain unchanged from the EMV specification.

5.3.2.2 Cardholder verification

The Cardholder System next requests the cardholder to enter the PIN. PIN verification can take place online or offline depending on the Cardholder Verification Methods (CVMs) specified in the card by the issuer. To perform offline verification, the Cardholder System sends a PIN Verification Request message to the IC card (step 7). The PIN does not need to be encrypted, since the environment is under cardholder control. On receipt of the PIN Verification Request, the EMV card returns a PIN Verification Response message (step 8) which indicates whether PIN verification has been successful. If so, the Cardholder System performs Terminal Risk Management; otherwise, the protocol ends.

5.3.2.3 Terminal risk management and action analysis

After successful cardholder verification, the Cardholder System performs Terminal Risk Management and Terminal Action Analysis. These two processes are as in the EMV specification. The Cardholder System then generates and

sends an AC Command to the EMV card (step 9).

5.3.2.4 Card action analysis

The IC card first performs its own risk management and then executes the Card Action Analysis to determine whether the transaction is to be approved offline, rejected offline, or processed online. In step 10, an AC will be generated by the IC card and sent to the Cardholder System where the nature of the AC will depend on the result of the Card Action Analysis (for details see below). The Cardholder System in turn forwards it to the Merchant Server (step 11) where the AC will be checked against the PI.

Offline approval. If the transaction is approved offline, the AC generated by the card is a TC, which is sent to the Cardholder System from where it is forwarded to the Merchant Server (steps 10/11). The TC will be held with the PI and acts analogously to the receipt in a conventional POS system. The Merchant Server can later send a batch of TCs, with the corresponding PIs, to the Acquirer to capture the payments. The issuer will verify the MAC in the TC and compare the information in the TC with that in the PI. If they match, the payment is accepted and processed.

Offline decline. If the transaction is declined offline, the AC generated by the card is an AAC which is sent to the Cardholder System, where it will be forwarded to the Merchant Server (steps 10/11). The merchant can store the AAC with the PI for management purposes, and the transaction now ends.

Online processing. If the IC card decides that online authorisation is needed, then the AC generated by the card is an ARQC which is forwarded to the Merchant Server via the Cardholder System (steps 10/11). The Merchant Server then sends the ARQC with the PI to the Acquirer and thence to the issuer (step 12). The issuer responds to the ARQC with an ARPC (step 13). The

Script Processing may also now be performed by the issuer to send command scripts to the IC card (steps 14/15). The transaction will be accepted or declined according to the ARPC. If it is accepted, the IC card generates a TC (step 16) and the process previously described under offline approval is performed. Similarly, if the transaction is declined, the IC card generates an AAC (step 16). The TC or AAC is then forwarded to the Merchant Server (step 17) and the transaction processing ends.

5.3.3 Security services

We now describe how the desired security services are provided in the proposed protocol.

5.3.3.1 Authentication

Cardholder and IC card authentication are provided in the same way as in 'standard' EMV. Cardholder authentication is based on knowledge of a PIN. IC card authentication uses static and dynamic data authentication. Merchant Server authentication, however, is not provided by the protocol. However, Merchant Server authentication is provided via the SSL/TLS session used to protect the Internet link.

5.3.3.2 Confidentiality and integrity

Although the entered PIN is not encrypted, its confidentiality and integrity are protected since it never leaves the environment over which the cardholder has control. If, however, PIN verification needs to be online, the entered PIN will be encrypted using a secret encipherment key shared between the IC card and the issuer. The PIN can also be encrypted between the PIN pad and the IC

card using the IC card's public key.

AC integrity is guaranteed by the use of MACs, as in the EMV specifications. Nevertheless, confidentiality of the AC is not provided by the protocol. However, confidentiality for the AC when it is transmitted over the Internet is provided by the use of SSL/TLS.

5.3.3.3 Non-repudiation

A measure of Cardholder non-repudiation is provided by the TC. The existence of a valid TC provides evidence that the genuine cardholder has implicitly consented, and hence the cardholder has consented to the transaction by entering his/her PIN. By contrast, merchant non-repudiation is not provided, although the value of such a service is unclear.

5.4 Threat analysis

In the protocol, there are five locations where the transaction data is at risk. These are the Cardholder System, the card reader, the link between the two, the Internet link between Cardholder System and Merchant Server, and the Merchant Server. Threats to the Internet link are addressed since we assume the use of SSL/TLS to protect the link. Threats to the Merchant Server-Acquirer link are outside the scope of this thesis, since such threats apply equally to conventional use of an EMV card, and we only consider here threats introduced by 'remote EMV'.

Therefore we divide our threat analysis into two parts, namely threats to the cardholder environment, and threats at the Merchant Server. In each case, the possible types of transaction data which may be at risk are considered, along with the entities which may pose a threat. There are six types of transaction

data that need to be examined, namely the Static Application Data (SAD), PI, IAC, $S_{IC}(DAD)$, PIN, and four ACs. Note, however, that integrity threats to $S_{IC}(DAD)$ and the four ACs are minimal since they are cryptographically protected using 'standard' EMV techniques. We thus focus most of our attention on the SAD, PI, IAC, and PIN.

5.4.1 Threats to the cardholder environment

The integrity of the Cardholder System may be at risk, since it is likely to be PC-based and is also connected to the Internet. There are thus many threats to its integrity, including the possible presence of malicious code. Of course, an integrity check of the Cardholder System could, in theory, be performed by the IC card at the start of the process — however, not only would this require modifications to the functionality of the EMV card (and hence would not really be practical), but there is also a limit to the degree to which an entity, no matter how powerful, can check the integrity of the system to which it is connected. More importantly, any PC system used to conduct e-commerce could be compromised, and this could put at risk both the confidentiality of the card details and the integrity of any transactions. There is no obvious solution to this problem, although the use of trusted components (e.g. a trusted card reader, as discussed in Section 5.5) may be of some help. For the purposes of this chapter we assume that the cardholder is responsible for ensuring the correct operation of the Cardholder System, and we only consider threats to the Cardholder System arising from the cardholder him/herself.

We refer to the combination of Cardholder System, card reader, and link between the two, as the cardholder environment. The SAD, PI, IAC, $S_{IC}(DAD)$, PIN, and four ACs pass through this environment. However, cardholder threats to the SAD, IAC, $S_{IC}(DAD)$, and PIN are not serious since the cardholder has the card and knows the PIN.

A malicious cardholder can modify the PI to make the Cardholder System send a smaller transaction value to the card than specified by the Merchant Server. However, the fraud will be detected as soon as the $S_{IC}(DAD)$ arrives and is verified at the Merchant Server. As a result, modifying the PI yields little to the cardholder. More seriously, because the Merchant Server does not have the MAC key necessary to verify the ACs, the Merchant Server cannot determine if the ACs received are authentic. Modifying the ARQC and the ARPC will yield no gain since they are sent online to the issuer, and altering the AAC is also unattractive for the cardholder because it yields nothing financially. However, an unscrupulous cardholder can modify or replace an offline-approved TC sent from the IC card, thereby causing the payment capture to fail at a later stage.

However, this risk also exists in the conventional EMV environment. Special equipment could be used to interfere with the communications between the POS terminal and the IC card. As in our scheme, the equipment could replace or modify the TC so that the MAC becomes invalid. A possible way to address this threat is to delay dynamic data authentication until after the TC has been generated, and then include the TC in the DAD signed by the card. The Merchant Server can verify the card public key and signature, using the issuer public key, thereby guaranteeing the authenticity of the TC. Including the TC in the DAD should not be a problem in the future, since such a process is supported by the latest version of the EMV specifications (see Section 6.6 of [14]) using what is known as Combined Data Authentication.

5.4.2 Threats at the Merchant Server

The transaction data available at the Merchant Server are the SAD, IAC, PI, and the ACs. The Merchant Server is either the legitimate recipient or the generator of the four types of data. There is thus no serious threat to data confidentiality. There is also no obvious financial gain for the merchant from

breaching data integrity. The ARQC and the ARPC will be sent online to the issuer, and the TC and the PI will also be checked by the issuer. If any modifications are made, the payment capture or authorisation process will fail. It is also clear that there is no point in modifying the SAD or the IAC since doing so will simply interrupt the transaction process.

5.5 Use of trusted card readers

The proposed protocol may be used in conjunction with 'trusted' card readers, i.e. those with a simple user interface such as a PIN pad and small display. How a cardholder obtains such a card reader is outside the scope of this thesis. However, one possibility is that it could be distributed by card issuing banks to their customers.

5.5.1 System architecture

If a trusted card reader is used in the proposed protocol, the system architecture will remain similar to that described in Section 5.3.1. However, note that here we suppose that the card reader has a PIN pad and a display. A trusted card reader may look like one of the examples shown in Figure 5.4.

In the protocol, the special purpose software required for the Cardholder System can be made simpler, since most of its tasks are now performed by the card reader and the Merchant Server. The remaining task of the Cardholder System is only to forward messages between the card reader and the Merchant Server. On the other hand, to perform the added tasks, the card reader and the Merchant Server now need different special purpose software from that described in Section 5.3.1.

Figure 5.4: Possible trusted card readers

5.5.2 Transaction process

The transaction processing procedure is similar to that described in Section 5.3.2. Figure 5.5 shows the modified transaction processing procedure in the protocol for the case where a 'trusted' card reader is used.

As described in Section 5.3.2.1, the transaction begins with the static data authentication process. After successful static data authentication, the Merchant Server generates the IAC and PI and sends the two data items to the Cardholder System which in turn forwards them to the card reader (step 2). The card reader now displays the purchase information to the cardholder, instead of sending it to the Cardholder System. The IC card then computes and sends the signature $S_{IC}(DAD)$ along with its public key certificate to the Merchant Server via the Cardholder System (step 3). After successful card authentication, Processing Restrictions (step 4) is performed.

The cardholder verification process is one of the main differences between the previously described protocol and this modified protocol. Instead of using the Cardholder System, the cardholder is now required to enter his/her PIN via the PIN pad of the card reader (step 6). The PIN is then verified by the IC card. After successful cardholder verification, the Merchant Server (not the Cardholder System) performs Terminal Risk Management and Terminal Action Analysis, generates and sends an AC Command to the card reader (step 7). The rest of the procedure remains the same as that described in Section 5.3.2.

5.5.3 Threat analysis

If a trusted card reader is used, it is possible for the cardholder to verify the value of the transaction being authorised by the card by checking the amount displayed by the card reader. This therefore reduces the trust in the cardholder PC. Moreover, the cardholder can prevent the authorisation of multiple transactions

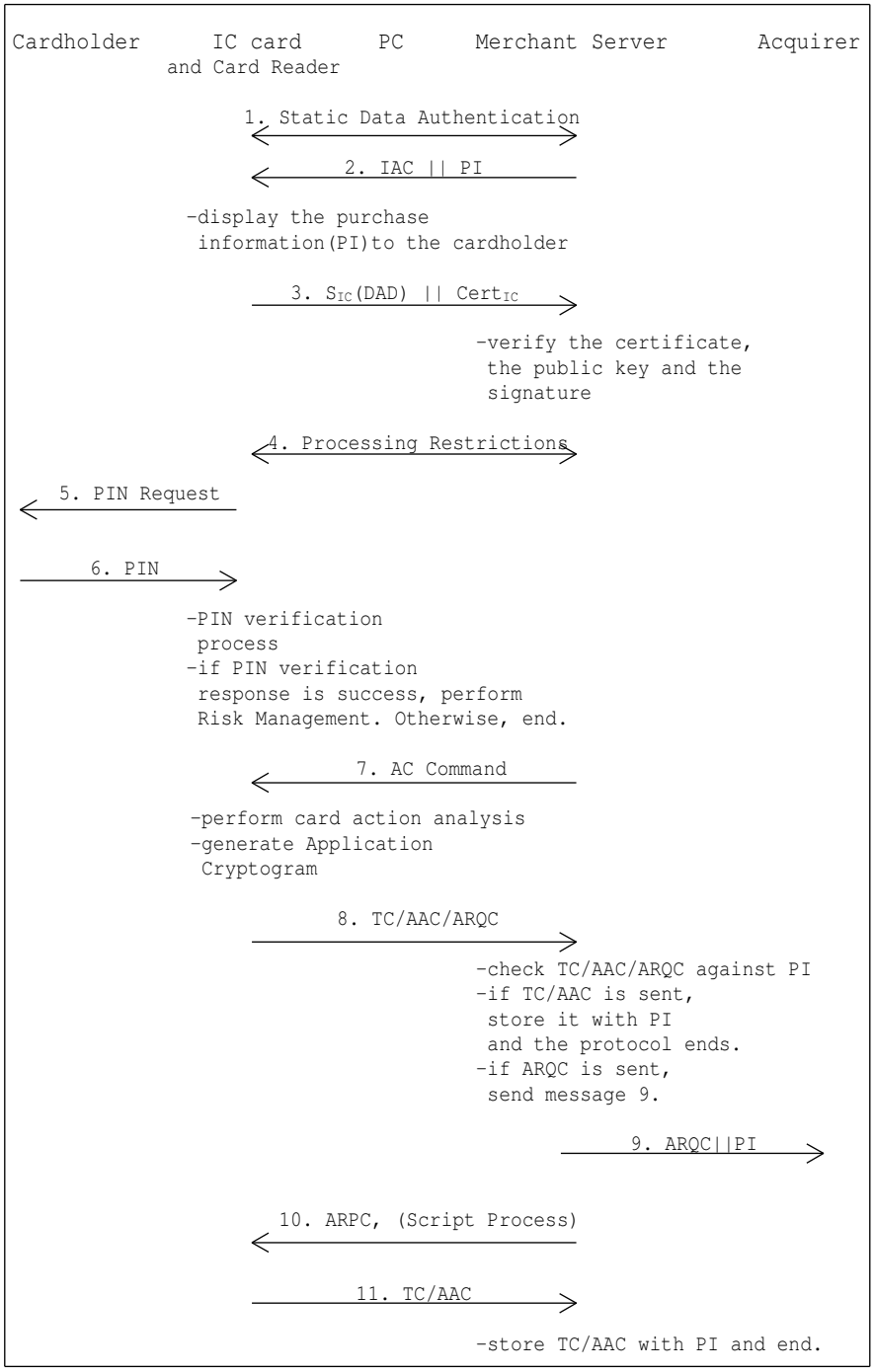


Figure 5.5: Transaction flow for remote EMV with secure card reader

by a manipulated cardholder PC. Using a trusted card reader also reduces the risk of malicious code spying on information entered via the user PC. The use of a trusted card reader therefore significantly reduces the necessary level of trust in the Cardholder System, and hence increases cardholder protection.

The fact that the Cardholder System software can be made much simpler also reduce the risks from malicious code present in the software, since it will be much easier to check the integrity of the program. However, as stated in Section 5.4, there is a limit to the degree to which the card reader and/or the IC card can check the integrity of the Cardholder System. For example, malicious code may modify the software in such a way that it replays the same checksum or issuer signature on the software. Nevertheless, there are ways to effectively check the integrity of a program using cryptographic techniques. These include the use of a MAC with 'random' information, such as that sent in the IAC, as key material. This prevents malicious code from replaying the integrity checksum. However, even if a MAC is generated on the concatenation of the code and a random value, this does not completely prevent malicious modification of code. It is still possible for the attacker to retain a copy of the unmodified code, and use it to compute the MAC which will clearly pass the MAC verification. Another disadvantage of introducing a cryptographic checksum is that the transaction process becomes more complex and hence loses the significant advantage of using the 'standard' EMV card capability.

5.6 Advantages and disadvantages

Using EMV cards to make a remote payment may compromise certain EMV security elements. A POS system has the advantage of face-to-face interactions as well as use of a tamper-resistant POS terminal. By contrast, Internet transactions involve no face-to-face interactions, and the terminal, here a combination of card reader, Cardholder System and Merchant Server, is clearly not tamper-

resistant. Indeed, in the proposed scheme, certain data which would be sent via internal communications in an EMV POS terminal are sent via the Internet.

The proposed protocol, however, enhances the security of existing Internet payment methods, which typically rely only on SSL/TLS for transaction security. There are known security risks with such an approach, including lack of cardholder authentication [82]. Our scheme provides cardholder authentication by using EMV PIN verification. The PIN is also associated with the IC card such that without the correct PIN no transaction can be made.

A major advantage of the scheme is that it uses existing technologies. This includes the SSL/TLS protocol, the EMV PKI established by the card brands and the issuers, and the EMV cards themselves. By comparison with the MasterCard SPA or Visa 3-D Secure scheme [90, 91], the proposed scheme offers the same security services, namely cardholder authentication and the associated non-repudiation. However, SPA and 3-D Secure both require real-time interactions with the issuer to perform cardholder authentication. By contrast, in the remote EMV scheme, the IC card can make decisions offline, thereby reducing communications overheads.

Using an IC card remotely does require special software (e.g. an applet) to be installed in the cardholder PC. An IC card reader, whether 'trusted' or not, is also needed. Nevertheless, use of the proposed scheme is 'light' compared to the SET initialisation process.

The protocol relies on Cardholder System integrity, since the Cardholder System could be modified by a malicious cardholder to send a bogus TC, or by an unscrupulous merchant to display different payment information to the cardholder from that sent to the IC card. The Cardholder System could also be compromised by viruses and trojan horses. Such threats, however can be significantly reduced by the use of 'trusted' card reader, and will in any case always exist for PC-based e-commerce solutions. The user must simply be made

aware of these threats.

Another weakness of the proposed protocol may be that confidentiality of the card details is not provided; the Cryptograms are transmitted over the Internet and hence may be intercepted. However, as described above, a secure channel, e.g. as provided by SSL/TLS, can be used in combination with the protocol to protect the transaction data en route between the Cardholder System and Merchant Server. Moreover, if the proposed protocol is used, it requires the presence of the IC card to make a transaction. Thus, if this protocol is universally used then knowledge of the card details becomes less useful.

5.7 Related work

The most closely related work is probably the scheme described in an annex to Book 3 of the EMV 2000 specifications [15]. This scheme combines SET with EMV-compliant IC cards to conduct Internet transactions. However, as discussed in Section 3.3.2, there remain serious obstacles to the use of SET.

Another related proposal is that described in a MasterCard white paper [63]. This white paper describes the use of the EMV card infrastructure in an authentication application. The protocol can be used to authenticate the presence of the cardholder in remote environments such as the Internet. Although the Mastercard protocol is very similar to our protocol in the way that they both use an EMV card, the former aims to provide only an authentication service. The protocol proposed in this chapter provides, in addition to cardholder authentication, payment authorisation.

GeldKarte [52] is an electronic cash card developed by the German banking industry. GeldKarte applications have also been extended to Internet uses, allowing the cardholder to use the value in the card to buy things from the

Internet as well as to enhance the security of Internet transactions. However GeldKarte is clearly different from the proposed protocol since it is an electronic cash scheme.

5.8 Conclusions

In this chapter, we have proposed a way to use an EMV-compliant IC card for e-commerce transactions. In the scheme, a user card reader and PC (the Cardholder System) together with the Merchant Server collectively take the role of the EMV Merchant Terminal. Most of the transaction procedures are similar to those in the EMV specification.

The use of 'trusted' card readers in conjunction with the proposed protocol is also examined in this chapter. If a trusted card reader is used, then confidentiality protection for the PIN can be enhanced. Moreover, the simpler software required in the Cardholder System makes it easier to perform an integrity check.

Although some of the EMV security requirements are affected, the proposed scheme is a step towards enhancing existing SSL/TLS based electronic transaction processing.

Chapter 6

Using GSM in e-commerce

Contents

6.1 Introduction 93
6.2 An overview of GSM security 94
6.2.1 Subscriber identity authentication 94
6.2.2 Data confidentiality 95
6.3 Using GSM authentication for e-commerce 95
6.3.1 System architecture 96
6.3.2 Transaction processing 98
6.3.3 Threat analysis 99
6.3.4 Advantages and disadvantages 105
6.3.5 Summary 107
6.4 Using GSM data encryption for e-commerce . . . 108
6.4.1 System architecture 108
6.4.2 Transaction processing 111
6.4.3 Threat analysis 114
6.4.4 Advantages and disadvantages 123
6.4.5 Summary 124
6.4.6 A comparison of the two proposed protocols 124
6.5 Related work 125
6.6 Extending the protocols to 3G/UMTS 126
6.6.1 3G/UMTS security 126
6.6.2 Protocol extension 127
6.7 Conclusions 128

The aim of this chapter is to propose two different ways of using the GSM security services to enhance the security of electronic transactions, as well as to increase the user's mobility when making an Internet transaction.

An overview of GSM security services which are particularly relevant to the proposed protocols is given in Section 6.2. The first protocol, in which the GSM subscriber identity authentication service is utilised, is described in Section 6.3, followed by a threat analysis, and a discussion of advantages and disadvantages. A description of the second protocol is given in Section 6.4, which is again followed by a threat analysis, and a review of advantages and disadvantages. In Section 6.4.6, the two proposed protocols are then compared. Related work is examined in Section 6.5, followed by a description of how the protocols can be extended to make use of the security features of UMTS instead of GSM (Section 6.6). The last section concludes the chapter.

It is important to note that much of the material in this chapter has previously been described in [57, 59].

6.1 Introduction

As stated in Section 4.3.3, SSL/TLS does not obligate client authentication. This, in turn, makes possible certain frauds during Internet payment processing. To provide client authentication using SSL/TLS, the user must first establish an asymmetric key pair. He/she will also need a secure place to store the private part of the key. Usually the key is stored in the user PC (password protected). As a result, the user has to use the particular machine every time a payment is to be made. Although a smart card could be employed to store the key and enhance mobility, not many user PCs are equipped with smart card readers. Hence, and as discussed previously, making use of SSL client authentication is not really a viable proposition for most users.

By contrast, very large numbers of users across the world now possess a GSM mobile phone. Therefore, in this chapter we propose two payment protocols which utilise the security services provided in the GSM air interface to support user authentication. The first protocol only provides user authentication while the second protocol also achieves card details' confidentiality.

The two protocols indirectly reduce the threat posed by the storage of unencrypted card numbers in a merchant server by reducing the value of stolen card numbers to a fraudster. This is achieved by requiring the user to possess both a debit/credit card and a GSM Mobile Station (MS), i.e. a GSM Mobile Equipment (ME) and a GSM Subscriber Identity Module (SIM), which must be registered under the same name as appears on the card. In short, the two protocols make use of MS portability and the GSM security mechanisms to provide user authentication and data confidentiality in a way that also supports user mobility.

In this chapter, an overview of GSM security is first provided followed by specifications of the two proposed protocols. A threat analysis, and the advan-

tages and disadvantages of each scheme are subsequently given, followed by a comparison of the two proposed protocols. A discussion of related work and a possible extension of the protocols to make use of the UMTS security features are also given.

6.2 An overview of GSM security

In this section, we describe how security services are provided in the GSM air interface protocol. GSM provides three main security services for the air interface, namely subscriber identity confidentiality, subscriber identity authentication, and data confidentiality. However, we will describe only the two latter security services, since they are the ones exploited in the protocols proposed here. Details of the first security service, together with other details of GSM security can be found in a number of places, e.g. [17, 18, 67, 88, 92].

6.2.1 Subscriber identity authentication

Within every SIM there exists a long-term secret key, K_i, which is unique and known only to the SIM and Authentication Centre (AuC) of the home network operator of the subscriber. The home network operator is the organisation with whom the subscriber has a contractual arrangement for the provision of service, and which the subscriber pays for this service.

To authenticate a SIM, the visited network needs a *triplet* which consists of a random number (*RAND*), the expected response (*XRES*), and a secret cipher key (K_c). The (*RAND*, *XRES*) pair enables the network to verify the authenticity of the SIM without having the key K_i, while K_c is used for encryption (see Section 6.2.2). To compute a triplet, the AuC generates a *RAND* and passes it with K_i as parameters to algorithms $A3$ and $A8$, which are specific to a network

operator. The outputs of $A3$ and $A8$ are $XRES$ and K_c respectively.

The AuC generates triplets as required, and passes them to whichever network needs them. When a SIM is requested to authenticate itself to a network, a $RAND$ from a triplet provided by the SIM's home network is sent from the network to the SIM. Since the SIM is equipped with the function $A3$ and the secret key K_i, it can generate the Signed Response ($SRES$) using $RAND$ and K_i as inputs. The SIM then sends the $SRES$ to the network where the $SRES$ is compared with the $XRES$. If they match, SIM verification is successful.

6.2.2 Data confidentiality

In GSM, the encryption of the data exchanged between the mobile phone and the base station is based on the secret session key K_c. As described in Section 6.2.1, the AuC generates a triplet, one element of which is K_c. The key is made available to the visited network by the subscriber's home network's Authentication Centre (AuC). The key K_c is computed within the SIM and made available to its host mobile telephone for data encryption (all data encryption is performed externally to the SIM).

A stream cipher algorithm $A5$ is used to encrypt the data sent across the radio path. Unlike $A3$ and $A8$, the algorithm $A5$ is not network specific but is defined in the GSM standards. In [18], several versions of $A5$ are specified, the choice of which can be negotiated between the SIM and the network.

6.3 Using GSM authentication for e-commerce

In this section, an e-commerce user authentication protocol which makes use of the GSM authentication service is described. In the proposed scheme, a consumer is required to have a GSM Mobile Equipment and a SIM registered

under the name that appears on the debit/credit card. It is important to note that the protocol does not need the SIM to be modified in any way. However, the ME does need to have the means to take a *RAND* value from a PC, pass it to the SIM, and pass the *SRES* value from the SIM back to the PC.

In this section, the system architecture for this scheme is first described, followed by the transaction processing procedure.

6.3.1 System architecture

Three main system components are involved in our payment protocol. These are a User System, a merchant server, and an AuC. The system architecture is shown in Figure 6.1.

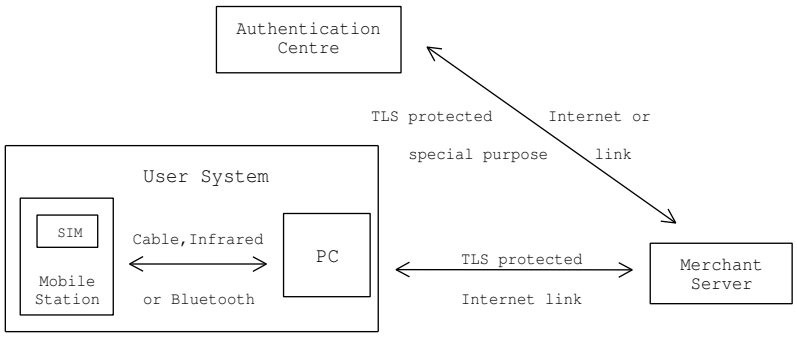


Figure 6.1: System architecture — GSM-based cardholder authentication.

6.3.1.1 User System

The User System consists of a GSM Mobile Station (MS), i.e. a GSM Mobile Equipment (ME) and a GSM Subscriber Identity Module (SIM), and a PC. The MS (in fact the SIM) is responsible for outputting the *SRES*. Therefore, although an ME is needed to interact with the SIM, the protocol can work without an ME if there is an alternative means for the SIM to communicate with the user PC. The means of communication used between the MS and

the user PC is not specified here. However, Infrared, a cable, or Bluetooth[1] could be employed for the purpose (such means of communication are becoming commonplace as mobile devices are increasingly being used for data transfer). In a recent version of the SIM Toolkit (U-SIM Application Toolkit) [93], there exists a command called 'AT command' which enables a U-SIM to tell an ME to open an infrared or bluetooth channel. The U-SIM Application Toolkit (USAT), therefore, could be used to implement the proposed protocol.

In the remainder of Section 6.3, the scheme is described in the context of a User System in which the PC provides the main platform for conducting user e-commerce, and the MS simply acts to support user authentication. However, in environments where the MS has sophisticated user interfaces and processing capabilities, e.g. a WAP device, the MS could take on some or all of the PC's tasks.

6.3.1.2 Merchant server and Authentication Centre

The merchant server is the component that interacts with the User System to support electronic transactions. The communication link between the Merchant Server and Cardholder System is the Internet. In our scheme, SSL/TLS is assumed to be used to provide merchant server authentication, and confidentiality and integrity for the information transmitted over this link. Indeed, the whole purpose of the scheme described here is to enhance the security provided by SSL/TLS rather than seeking to design a completely new and comprehensive security system. This is based on the belief that security for e-commerce must be introduced in ways which minimise the overheads for all parties, and in particular for the e-consumer.

The merchant server also interacts with the AuC in order to retrieve values required in the user authentication process. The AuC is required to supply the

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

merchant server with values necessary for the GSM identity authentication process. It takes inputs from the merchant server and produces the values used for identity authentication. The choice of the communication link between the two is again not an issue here. However, it could be an SSL/TLS protected Internet session or a special purpose link provided by the mobile network operator.

As discussed in Section 6.3.3, we suppose that the integrity and confidentiality of the merchant server/AuC link is protected in some way, e.g. via encryption and MACs or signatures; however, the means by which this is achieved is outside the scope of the discussion here.

6.3.2 Transaction processing

The proposed payment protocol starts after a consumer has decided to make a payment. The decision about which purchase to make is outside the scope of this thesis — we simply assume that the consumer and the merchant wish to perform a specified transaction.

The consumer first fills in a typical Internet purchase form using the PC. In this protocol however, the form is required to contain a field for a mobile phone number. Upon receipt of the form, the merchant server extracts the mobile number from the form and the identity authentication process begins. The procedure is illustrated in Figure 6.2.

The merchant server first sends the consumer's mobile number to the AuC in order to retrieve three values: a random number (*RAND*), an expected response (*XRES*), and the subscriber name. This corresponds to message 1 in the figure.

Upon receipt of the merchant server request, the AuC generates the (*RAND*, *XRES*) pair using the key K_i of the requested mobile number with algorithm *A*3. It then sends the (*RAND*, *XRES*) pair, along with the name of the subscriber,

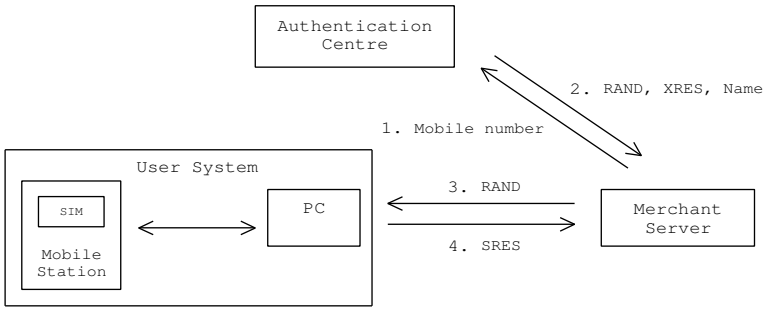


Figure 6.2: GSM-e-commerce identity authentication process.

to the merchant server, as shown in message 2 in the figure. Upon receipt of message 2, the merchant server first compares the name of the cardholder with the subscriber name received from the AuC. If they match, the *RAND* will be sent to the PC as in message 3 of the figure. Otherwise, the identity authentication process fails and the protocol ends.

After having received the *RAND*, the user PC forwards it to the ME. The ME then sends the *RAND* value to the SIM, just as it would if the *RAND* was sent via the radio interface by a GSM base station. The SIM now generates an *SRES* using the received *RAND* and its stored K_i as inputs to algorithm *A*3. The SIM then passes the generated *SRES* back to the ME, again just as it would normally (i.e. the SIM is not required to have any special functionality). The ME then sends the *SRES* to the PC, which forwards the value to the merchant server (message 4). At the merchant server, the *SRES* is compared with the *XRES*. If they match, the consumer is deemed to have been authenticated. The Internet transaction processing may now continue.

6.3.3 Threat analysis

In this section, we consider threats to the proposed protocol. The threats can be divided into three categories: threats to the User System, threats to the two

communications links (user system/merchant server and merchant server/AuC), and threats in the merchant server and the AuC.

6.3.3.1 Threats in the User System

As stated previously, the User System consists of a user PC and an MS. Since this protocol does not require the user PC to contain sensitive information, the threats arising from the PC are minimal. Although information that passes via the PC can be cached, this information is not confidential. A debit/credit card number can be cached and compromised but the protocol still requires a corresponding SIM to make an electronic transaction. In any event, such a threat would exist in any PC-based e-commerce protocol. The issue of protecting the user PC against threats to its integrity is addressed in more detail in Chapter 7.

Threats to the MS are divided into two scenarios, depending on the amount of information an attacker has. Clearly, if he/she has neither the SIM nor the card details, a transaction cannot be made and hence there is no threat. It should also be clear that if the attacker has both a complete set of card details and a stolen SIM for the cardholder, then the system cannot prevent an attack — unless, of course, the SIM has been reported stolen and blacklisted by the network, or the SIM has been PIN protected by its owner. We therefore consider the two main 'intermediate' scenarios.

Scenario 1: Attacker has a stolen SIM without the corresponding card details.

In this scenario, if an attacker has stolen a SIM and the subscriber name of the stolen SIM is unknown, although a valid *SRES* can be generated, he/she will not be able to create a matched cardholder name necessary to pass the authentication process.

By contrast, if the subscriber name is known to the attacker, it is possible

to complete the protocol successfully using a fabricated set of cardholder details as long as the fabricated details include a cardholder name corresponding to the subscriber name. However, the fraud will become clear soon after the merchant tries to charge the card. In the most typical case for an e-commerce transaction, the merchant will try to charge the specified payment card before the goods are dispatched. In such a case, the threat is therefore small. Nevertheless, the threat can be more serious if the goods are, for example, information or music which will be delivered instantly via the Internet. However, even in this case, the threat can be avoided if, as is often the case, the merchant server seeks payment authorisation before authorising delivery of the goods. If the card details are fabricated then the card issuer will, of course, reject the payment.

A possible way to prevent such attacks is for the SIM to be PIN-protected. It is also important that the PIN is never entered on an untrusted device.

Scenario 2: Attacker has stolen card details without the corresponding SIM.

If an attacker has only card details, without the SIM, it will not be possible to generate a valid *SRES*. This threat is therefore addressed by the scheme described above.

Thus, to be successful, an attack on the user system needs both the victim's SIM and the corresponding debit/credit card details to complete a fraudulent transaction.

6.3.3.2 Threats to the communications links

If any of the information transferred across either of the links is modified, then the protocol will fail. Hence, a theoretical denial of service attack exists, although there are many simpler ways to prevent the completion of a transaction. We now consider other threats arising to the two links.

Threats on the PC/merchant server link: The confidentiality and integrity issues apply to the payment information transferred across this link. However, as stated in Section 6.3.1, we assume that the Internet link between the PC and merchant server is protected using SSL/TLS throughout the transaction procedure.

Note that a possible alternative to the protocol described here would be to use GSM authentication to enhance the security of the SSL/TLS initialisation process. However, if such an approach is followed, it is not clear how to achieve the desired link between the GSM subscriber name and the cardholder name.

Threats on the merchant server/AuC link: Threats on this link can be further divided into two types, namely integrity threats and confidentiality threats.

- Integrity threats: There are a number of ways in which an attacker could manipulate this link in order to persuade the merchant server to accept an impostor. Perhaps the simplest method would involve the attacker using an arbitrary (valid) SIM and ME in combination with stolen card details (which, of course, will not match the GSM subscription name). In message 2 the AuC will provide a valid *RAND* and *XRES* for the attacker's SIM, and will return the name associated with the attacker's GSM subscription. An active attacker could change this name to the name associated with the stolen card details, and the merchant server will accept message 2. The remainder of the protocol will complete correctly, and the account for which the details were stolen will be charged for the transaction.

  An alternative attack, again using stolen card details, does not require the attacker to have a valid SIM at all. The attacker supplies an arbitrary (but valid) GSM number with the stolen card details. In message 2, the AuC will send a (*RAND*, *XRES*) pair for the arbitrarily chosen GSM subscription, along with the subscriber name. The active attacker can then

replace the contents of message 2 with the name for the stolen card details, along with an arbitrary (*RAND*, *XRES*) pair. The merchant server will accept message 2 because the names match, and will send the manipulated *RAND* to the attacker in message 3. The attacker simply returns the manipulated *XRES* value in message 4, and again the attack will succeed. The existence of these attacks means that it is vital that the integrity of the link between AuC and merchant server is protected.

- Confidentiality threats: There are also a number of serious confidentiality threats. First note that a passive eavesdropper can perform an attack similar to the second integrity attack described above. Suppose an attacker has a set of stolen card details and also knows the GSM number for the owner of the stolen card details. The attacker initiates the protocol using the stolen card details and the known GSM number. Message 2 will be accepted by the Merchant server because the GSM number belongs to the valid cardholder. However, if the attacker can intercept message 2, then the *XRES* value can be obtained. The attacker then simply inserts this value into message 4 and the protocol will complete successfully.

Also note that, in the absence of integrity and confidentiality, the merchant server/AuC protocol could also be used to find the subscriber name corresponding to any GSM number. This would be a significant breach of GSM subscriber confidentiality.

These attacks mean that it is important to provide both confidentiality and integrity for this link, and this is why we assume throughout this section that this link is both confidentiality and integrity protected.

6.3.3.3 Threats in the merchant server and the AuC

Since the merchant server is responsible for the identity authentication process, in particular the comparison of names and *XRES* with *SRES*, it is important to protect the server against any attack which might cause the protocol to be bypassed.

Over and above the integrity of the user authentication process, the merchant server will have access to large volumes of potentially sensitive subscriber information. As part of the user authentication process, the merchant server retrieves from the AuC the account holder name for any GSM telephone number. Not only is this a sensitive privacy issue, but requiring the AuC to supply such information may potentially be in breach of its licence and/or data privacy legislation. It is therefore vital that the merchant server be protected and trusted so that this information cannot be abused.

One way of mitigating this security issue is to make a slight modification to the protocol of Section 6.3.2. In the revised protocol, shown in Figure 6.3, in message 1 the merchant server supplies the cardholder name as well as the mobile number. The AuC is then required to compare the name supplied in message 1 with the name it has associated with the GSM number. If they do not match the protocol should not proceed. If they do match, in message 2 the AuC simply provides a (*RAND*, *XRES*) pair.

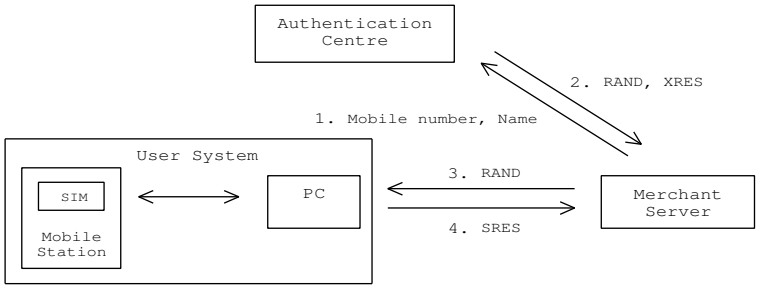


Figure 6.3: Revised protocol.

Another way to reduce this threat is for the merchant server to create and send a *RAND* to the User System and thence the SIM. Upon the receipt of the *RAND*, the SIM generates the *SRES* and sends it to the merchant server via the user PC. The merchant server subsequently sends the cardholder's name, his/her mobile number, the *RAND*, and the *SRES* to the AuC to verify. The protocol is shown in Figure 6.4.

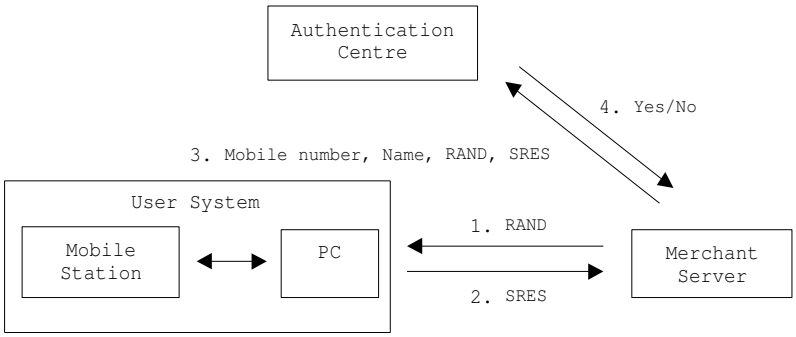


Figure 6.4: Another revised protocol.

These modified protocols have the advantage that the AuC retains control of sensitive subscriber information. However, it has the disadvantage of requiring additional processing by the AuC.

If the integrity of the AuC could be compromised, then there are possible attacks to the security of the user authentication process. However, in such an event there are also many other serious attacks to the security of the GSM network itself, and so we assume that the AuC is well-protected.

6.3.4 Advantages and disadvantages

In this section, the advantages and disadvantages of the proposed protocol are considered.

6.3.4.1 Advantages

The following advantages arise from the proposed use of GSM-based user authentication.

1. The protocol provides user authentication based on GSM subscriber authentication. As a result, stolen credit card details cannot be used to launch a successful e-commerce transaction.
2. Since stolen credit card details cannot be used to launch a successful e-commerce transaction, the threat arising from the storage of unencrypted credit card numbers in merchant servers is accordingly reduced.
3. The protocol supports user mobility. The user authentication process requires only the correct software to be loaded on the PC, and for there to exist a means to connect the MS to the PC. In the authentication process, the PC is simply responsible for forwarding messages between the MS and the merchant server. Moreover, since the protocol does not involve storing any secrets on the PC, the risks in using untrusted PCs are minimised.
4. The protocol can work with a 'standard' GSM SIM. It simply requires an appropriately equipped ME and a user PC.
5. From the merchant point of view, the protocol will reduce the number of fraudulent transactions and hence lessen the cost of 'card-not-present' chargebacks.

6.3.4.2 Disadvantages

The following disadvantages arise from use of the proposed GSM-based user authentication.

1. Prior agreement is required between the merchant and the mobile phone

service provider to support the protocol between the AuC and the merchant server. To avoid the need for many individual arrangements between merchants and mobile network operators, a Trusted Third Party (TTP) could be introduced to act as a 'broker' between the two parties. This broker could simply route messages between merchant servers and the AuC. In such a scenario, merchant servers and mobile network operators would only need to have a contractual agreement with the broker. One possible candidate for the broker would be the card brand.

2. Since it is possible for the merchants to collect subscriber identity information and phone numbers, the merchants must be trusted not to abuse such information.

3. Merchants may be charged for the AuC services. This cost therefore has to be weighed against the cost of 'card-not-present' chargebacks which may vary from merchant to merchant. Of course, this is not a disadvantage for the GSM network provider, who may find this a useful additional revenue stream.

4. If the U-SIM Application Toolkit [93] is to be used, the proposed protocol may require an ME and a SIM which support the functionality.

6.3.5 Summary

We have proposed a way in which GSM subscriber identity authentication can be used to enhance e-commerce security. The protocol provides user authentication and hence significantly reduces threats arising from misuse of misappropriated card details. It therefore also indirectly reduces the risk of storing card details in unencrypted form in merchant servers. The protocol works with a 'standard' GSM SIM and requires only an appropriate equipped Mobile Equipment and a user PC. It therefore imposes minimal overheads on the user, thus increasing the likelihood of successful use. The gains for the merchant in terms of reduced

chargebacks also appear significant, and the possibility of an increased revenue stream may also make the system attractive to the GSM operators.

6.4 Using GSM data encryption for e-commerce

In this section, we propose another protocol in which a consumer is again required to have a GSM Mobile Equipment (ME) and a SIM registered under the name that appears on his/her debit/credit card. It is important to note that, just like the previous scheme, the protocol does not need the SIM to be modified in any way. However, the ME does need some special capabilities, as described below.

In this section, the system architecture for this protocol is first described, followed by the transaction processing procedure.

6.4.1 System architecture

Five main system components are involved in this payment protocol. These are a User System, a merchant server, an acquirer, an issuer, and an AuC. The system architecture is shown in Figure 6.5.

6.4.1.1 User System

The User System consists of a PC and a Mobile System (MS) which includes a SIM and an ME. The MS (in fact the SIM) is responsible for outputting the key K_c. Therefore, although an ME is needed to interact with the SIM, the protocol can work without an ME if there is an alternative means for the SIM to communicate with the user PC.

As for the protocol described in Section 6.3, the means of communication

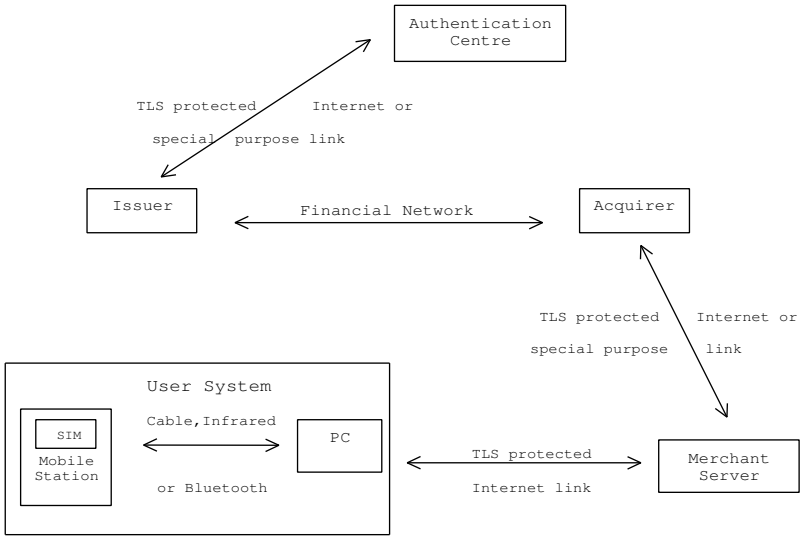


Figure 6.5: GSM-e-commerce payment system architecture.

used between the MS and the user PC is not specified here, and Infrared, a cable, or Bluetooth[2] could be employed.

In the remainder of Section 6.4, the scheme is described in the context of a User System in which the PC provides the main platform for conducting user e-commerce, and the MS acts to support the additional security functions. However, in environments where the MS has sophisticated user interface and processing capabilities, e.g. a WAP device, the MS could take on some or all of the PC's tasks.

Note that, in this protocol, we have proposed use of the key K_c for MAC computation, where this key is normally used for data encryption. This is a breach of key separation principles, although it may not be of significance here. However, if this does give rise to security concerns, then the key could be modified, e.g. passed through a one-way hash function, before being used to compute a MAC.

[2]http://www.bluetooth.com

6.4.1.2 Merchant server

The merchant server is the component that interacts with the User System to support electronic transactions. The merchant server communicates with the User System via the Internet. As for the scheme described in Section 6.3, we assume that the link is protected by SSL/TLS to support merchant server authentication and confidentiality and integrity protection for transferred data.

The merchant server also interacts with the acquirer to request a payment authorisation. The choice of the communication link between the Merchant Server and the Acquirer is not an issue here. However, it could be an SSL/TLS protected Internet session, or a special purpose link provided by the acquirer.

As discussed in Section 6.4.3.2, we suppose that the integrity of the merchant server/acquirer link is protected in some way, e.g. via MACs or signatures; however, the means by which this is achieved is outside the scope of the discussion here.

6.4.1.3 Acquirer, issuer and Authentication Centre

The acquirer interacts with the issuer via the financial network to support transaction authorisation. However, in the proposed protocol, the issuer has the additional roles of authenticating the cardholder, decrypting the card details, and verifying the authenticity of the payment details and card details.

The issuer interacts with the AuC of the user's home network in order to retrieve values necessary to utilise the GSM security service. The choice of the communication link between the issuer and the AuC is again outside the scope of this thesis. However, it could be an SSL/TLS protected Internet session or a special purpose link provided by the mobile network operator. As discussed in Section 6.4.3.2, we assume that integrity and confidentiality protection for the

issuer/AuC link are provided by some means.

The AuC is required to supply the issuer with the values normally used for the GSM data confidentiality service.

6.4.2 Transaction processing

The proposed payment protocol starts after a consumer has decided to make a payment. The consumer first fills in a typical Internet purchase form (excluding card details). The form is also required to contain a field for a GSM phone number. Upon receipt of the form, the merchant server initiates the protocol. The procedure is illustrated in Figure 6.6. In this figure:

- *RAND* denotes a randomly generated 64-bit value,
- $e_K(M)$ denotes the encryption of message M (using symmetric encryption) with secret key K,
- K_c denotes a GSM cipher key used for encryption and MAC computation,
- 'CD' denotes card details entered by a consumer,
- $X \| Y$ denotes the concatenation of data items X and Y,
- $\text{MAC}_K(M)$ denotes a MAC computed on message M using the key K,
- 'PD' denotes payment details that must be unique per transaction (e.g. by containing a time stamp or a transaction ID),
- 'MN' is a GSM phone number, and
- 'NAME' is the subscriber name.

Upon receipt of the form, the merchant server generates and sends a random number (*RAND*) and the payment details (PD) to the user PC, as shown in

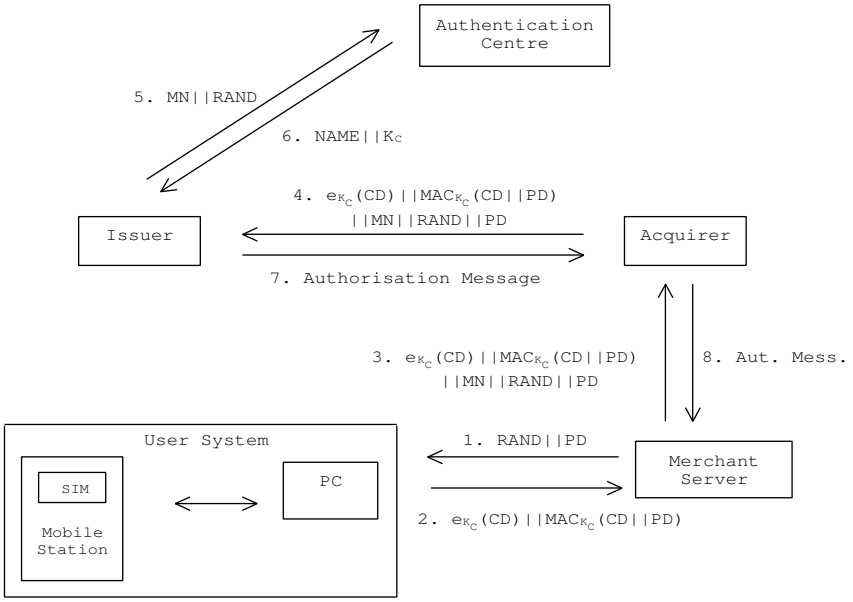


Figure 6.6: GSM-e-commerce payment protocol.

message 1. The user first checks the PD. If they are correct, the *RAND* is then forwarded to the SIM, which in turn calculates the key K_c using the received *RAND* and its stored key K_i as inputs to the key derivation algorithm shared with the AuC. The SIM then passes the generated K_c back to the ME, just as it would normally do in a GSM telephone (i.e. the SIM is not required to have any special functionality). The ME then forwards the encryption key to the user PC.

The user PC uses the key K_c to encrypt the card details entered by the user. Examples of card details include the account number, expiry date, issue number, and card verification code (CVC). In addition to the encryption, a MAC is computed on a concatenation of the card details (CD) and the payment details (PD), again using the key K_c, to protect the integrity of the information. The PD must include (but is not limited to) the transaction value, date, and merchant and transaction identification numbers. The enciphered information and the MAC are then sent to the merchant server as shown in message 2 in

the figure. Note that the encryption and MAC algorithms used here can be issuer-specific, and need bear no relationship to the GSM algorithms. The only requirement is that they are able to operate using a 64-bit GSM encryption key K_c. Note that, to avoid key separation issues associated with use of the same key for both encryption and MAC calculation, two different variants of the key can be used for the two purposes e.g. as calculated by applying a hash-function to the key. However, it is important to note that if one key can be derived from another, then there is a potential for security vulnerabilities to be introduced; hence this process needs to be designed with care.

The merchant server concatenates the received message with its own version of PD, the *RAND*, and the user mobile phone number (MN) extracted from the purchase form. The result is sent to the acquirer where it is forwarded to the issuer as shown in messages 3 and 4 respectively.

In order to decrypt the encrypted CD and verify the MAC, the issuer needs to contact the appropriate AuC to retrieve the key K_c. The issuer can either determine the identity of the user's home network (and hence the address of the AuC) from the mobile number, or, if necessary, an identifier for the AuC can be included in messages 2, 3 and 4. To enable the AuC to respond with the right information, the issuer sends the mobile number and the *RAND* to request the AuC to respond with the subscriber name and the cipher key. This corresponds to message 5 in Figure 6.6. The AuC then responds with message 6 containing the name and the key.

The issuer first decrypts the CD using the supplied key K_c. The issuer then verifies the MAC to check the integrity of both the CD and the information in PD, especially the transaction ID and merchant ID. The checking of PD is necessary in order to prevent replay attacks (see Section 6.4.3.3). The checking of the MAC is also important because, if the MAC is valid, then the user must possess the valid SIM. If, in addition, the subscriber name matches the

cardholder name, the cardholder is deemed to be the legitimate cardholder since he/she possesses the SIM.

If all these processes are successful, the issuer can now proceed with the 'normal' transaction authorisation. Otherwise, the transaction is declined. The decision of the issuer is reflected in the Authorisation Message (message 7) which is then sent to the acquirer where it is forwarded to the merchant server as shown in message 8. The protocol now ends.

Finally note that the protocol could be enhanced to ensure that a different key is used for every transaction, even if the merchant fails to generate a new $RAND$ every time. The user system could generate its own random number, $RAND^*$ say, and then derive a transaction key K_t as a one-way function of $RAND^*$ and K_c. The key K_t can then be used instead of K_c in the protocol. In this case however, the $RAND^*$ value must also be sent all the way to the AuC to enable it to compute the same key K_t.

6.4.3 Threat analysis

In this section, we consider threats to this protocol. The threats can be divided into four categories: threats to the User System, threats to the communications links (User System/merchant server, merchant server/acquirer, financial network, and issuer/AuC), threats in the merchant server, and threats in the acquirer, the issuer and the AuC.

6.4.3.1 Threats in the User System

As stated previously, the User System consists of a user PC and an MS. In this section, threats to the MS are first described followed by threats to the user PC.

Threats to the Mobile System: If an attacker has stolen a SIM, although a

valid cipher key K_c can be generated, he/she will not be able to complete a transaction. The attacker still needs matching card details, and, even if stolen card details are submitted, the fraud will be detected as soon as the transaction is processed by the issuer. This is because the card details, in particular the cardholder name, will not match the subscriber name sent by the AuC.

It is clear, however, that if the attacker has both a complete set of card details and a stolen SIM for the cardholder, then the system cannot prevent an attack — unless, of course, the SIM has been reported stolen and blacklisted by the network.

If an attacker has stolen an ME without a SIM, he/she will not be able to make a fraudulent payment, regardless of whether the corresponding card details have been obtained. The ME is only responsible for forwarding information between the SIM and the user PC. Without a SIM, a valid encryption key K_c cannot be generated, and hence stealing an ME does not enable a successful attack.

Threats to the user PC: Since the user PC does not contain sensitive information, the threats arising from the PC are minimal. Although information that passes via the PC can be cached and attacked, this information is not confidential. Debit/credit card details and the payment details can be cached and compromised, but the protocol still requires a corresponding SIM to make an electronic transaction. In any event, and as stated previously, any PC-based e-commerce protocol involves the same risk of credit card number compromise — for a more detailed analysis of the risks to a PC used for e-commerce see Chapter 7. The cipher key can also be compromised, but since it is only a transient key, and is a function of the *RAND* sent from the merchant server for each transaction, compromising this key is not a threat unless an attacker can impersonate a merchant server and force re-use of an old *RAND* value (and hence an old key K_c). This can be prevented by requiring the user system to

authenticate the merchant server and by the provision of integrity protection for this link (see threats to the communication link).

If the fact that the PC has access to the card details is an issue, alternative implementation scenarios are possible; in particular some of the functionality currently allocated to the user PC could be transferred to the ME. For example, if the ME has an appropriate user interface (and appropriate processing capabilities), the card details could be entered into the ME and encrypted there, denying the user PC any access to sensitive information.

6.4.3.2 Threats to the communication links

If any of the information transferred across any of the links is modified, then the protocol will fail. Hence, a theoretical denial of service attack exists, although there are many simpler ways to prevent the completion of a transaction. We now consider other threats arising to the three links (User System/merchant server, merchant server/acquirer, and issuer/AuC links). As stated above, the issuer/acquirer communication link is assumed to be provided by the card brand. Therefore, its security is assumed here.

Threats on the User System/merchant link: Threats on this link can be divided into two types, namely integrity threats and confidentiality threats. Each piece of information that is transmitted via this link, i.e. the *RAND*, the card details (CD) and the payment details (PD), are now considered in turn against both types of threat. However, threats to *RAND* will not be included, since modifying or eavesdropping on this value do not enable attacks to be launched. As a result, only threats to the card details and the payment details will be considered.

- Integrity threats: It is important to ensure PD integrity in order to prevent a malicious merchant from modifying it to gain financial advantage, such

as charging the consumer more than is agreed upon. The PD is protected against unauthorised modification using a MAC. Without the key K_c, it is hard to generate a valid MAC for a modified PD.

Although modifying the CD does not yield any gain to an attacker, in our protocol the information is included in the MAC computation to ensure its integrity. It is worth noting that including the CD in the MAC has no impact on the message length, and only creates a small extra computational requirement.

As stated previously, there are threats arising from forcing re-use of an old *RAND* for which the corresponding key K_c is known. In such a case, the MAC can be modified and/or the CD compromised. Although the likelihood of compromise of a key K_c by a malicious third party is relatively small, if this is a genuine possibility then integrity protection for this channel and merchant server authentication are required. This can be achieved using a secure channel such as is provided by SSL/TLS.

- Confidentiality threats: It is essential to ensure the confidentiality of sensitive information, i.e. the CD. In the protocol, this is provided by symmetric encryption.

  Unlike the CD, the PD contains no sensitive information and hence does not need protection against eavesdropping. Indeed, the PD is analogous to a Point of Sale (POS) receipt which typically contains only the store name, date, product description, transaction value and in some cases, the last four digits of the payment card used. Therefore, confidentiality of the PD is not provided. However, if both the PD and the MN can be intercepted, then this may pose a privacy threat to the consumer.

  However, as part of a purchase form, the consumer name along with other contact information, in particular his/her mobile number, will be entered. Consequently, confidentiality of the link is needed, otherwise it would be possible for an attacker to passively eavesdrop on the link and obtain

the (MN, consumer name) pair. Confidentiality protection for the User System/merchant server link can be provided using SSL/TLS, just as is normally the case for Internet transactions.

Threats on the merchant server/acquirer link: Threats to the card details (CD), the *RAND*, and the PD are similar to those previously described. We now consider the remaining information, i.e. the MN and the authorisation message, in terms of confidentiality and integrity threats.

- Confidentiality threats: By monitoring the link, a list of mobile phone numbers could be obtained. However, unlike the threat described in the previous section, the consumer name is not transmitted on this link. Therefore, having only a list of phone numbers without the corresponding names is not likely to be very valuable.

  An authorisation message may contain information similar to that in a normal receipt. However, it is clear that it does not need to contain any card details since such information is not necessary for the merchant to complete the proposed payment protocol. Therefore, the authorisation message is not sensitive and hence is not protected against eavesdropping in this protocol.

- Integrity threats: Modifying the MN can only make the protocol fail and does not yield gains to any party involved. On the other hand, the integrity of an authorisation message is important, since a malicious merchant could modify the authorisation message from reject to authorise, potentially causing a dispute. A way to prevent such a threat is to ensure the integrity of the message, e.g. to require the acquirer to sign or MAC protect message 8 in Figure 6.6 before sending it to the merchant.

Threats on the issuer/Authentication Centre link: Threats on this link can again be divided into two types, namely integrity threats and confidentiality

threats.

- Integrity threats: Modifying the *RAND* and the MN will only cause the protocol to fail. However, if the integrity of information sent via this link is not ensured, it would be possible for an attacker to manipulate this link in order to bypass the cardholder authentication check. The attacker could first use an arbitrary (but valid) GSM number and secret key to encrypt the details of a stolen card (which, of course, will not match the GSM subscription name and generate a MAC). In message 6 the AuC will provide a valid K_c and the name associated with the attacker's GSM subscription. An active attacker could then replace the contents of message 6 with the name associated with the stolen card details along with the arbitrary secret key he/she used for the encryption and MAC computation. The issuer will accept the cardholder authentication because the MAC will verify correctly and the decryption will yield the expected data. It will then proceed with the payment authorisation process. The remainder of the protocol will complete correctly, and the account for which the details were stolen will be charged for the transaction. The existence of this attack means that it is vital that the integrity of the link between AuC and issuer is protected.

- Confidentiality threats: As stated before, *RAND* is not sensitive and hence confidentiality threats to the data transmitted on this link are minimal. The key K_c is also not highly sensitive, although if the key can be intercepted, and if an attacker also has access to the encrypted card details, then it would be possible for them to decrypt the card details. However, having only the card details is not sufficient to make an electronic payment transaction in the proposed protocol.

  Confidentiality threats also arise from the fact that the mobile number and the corresponding subscriber name are sent across this link. Therefore, in the absence of confidentiality protection on the issuer/AuC link, an

eavesdropper could find the subscriber name corresponding to any GSM number. This would be a significant breach of GSM subscriber confidentiality. A list of matching names and mobile numbers could also be compiled, representing a further potential privacy threat.

These attacks mean that it is important to provide both confidentiality and integrity for this link, and this is why we assume throughout this section that this link is both confidentiality and integrity protected.

6.4.3.3 Threats to the merchant server

The merchant server does not have access to some of the sensitive information, in particular the CD, that it would in traditional electronic transactions, since the information is encrypted with a key that the merchant server does not have. The protocol therefore reduces the threat of storing unencrypted card details at merchant servers, which is one of the major security threats when SSL/TLS alone is used to protect electronic transactions.

The merchant server does have access to the *RAND*, PD, and the authentication message. However, this information is not sensitive. Therefore, there is no serious threat to data confidentiality in this system component.

A malicious merchant could replay message 3 in Figure 6.6 to re-capture a payment. However, recall that PD must contain the charging amount, date, and merchant and transaction ID (see Section 6.4.1.1). Therefore if, for example, an unscrupulous merchant tries to re-submit message 3 to the acquirer, the fraud will be detected as soon as the issuer performs the authorisation. The issuer will be able to detect that the transaction ID of a certain merchant matches a previously submitted transaction. If the issuer maintains a record of the *RAND* values used for each payment card account, a matching *RAND* in two different transactions can also be an indication of merchant fraud. This is because the

RAND must be re-used in the fraudulent transaction to enable the issuer to decrypt the replayed CD and hence be able to authorise the payment. The integrity of the CD and the PD is protected by use of the MAC.

Finally, the merchant server has access to large volumes of potentially sensitive GSM subscriber information. As part of the user authentication process, the merchant needs the user's mobile number. The merchant server also knows the name of the user since it is typically entered in the purchase form. As a result, the merchant server can collect mobile numbers and corresponding subscriber names. However, this is analogous to supplying personal contact information in a typical order form. Privacy laws then apply and may require order forms to contain a privacy statement or a section for user consent if their personal data is to be used for other purposes.

6.4.3.4 Threats to the acquirer and issuer

Threats to the acquirer are minimal, since it is responsible only for forwarding messages between the issuer and the merchant server. Moreover, the information that is transmitted via the acquirer is not sensitive.

Since the issuer is responsible for the identity authentication process, in particular the comparison of the names, it is important to protect the issuer against any attack which might cause the cardholder authentication process to be bypassed.

In the protocol, the issuer retrieves the account holder name for any GSM telephone number from the AuC. As a result, the same user privacy issue described in the previous section also exists here. As for the protocol described in Section 6.3, not only is this a sensitive privacy issue, but requiring the AuC to supply such information may potentially be in breach of its licence and/or data privacy legislation. It is therefore vital that the issuer is protected so that

this information cannot be abused.

One way of mitigating this security issue is to make a slight modification to the protocol of Section 6.4.2. In the revised protocol, shown in Figure 6.7, in message 6 the AuC supplies only the encryption key K_c. Subsequently, two more messages are required in the protocol. After successfully decrypting the CD and verifying the MAC (using K_c), the issuer sends message 7 which contains the cardholder name as well as the mobile number. The AuC is then required to perform the matching between the name supplied in message 7 with the name it has associated with the GSM number. The AuC finally sends the result of the matching to the issuer (message 8).

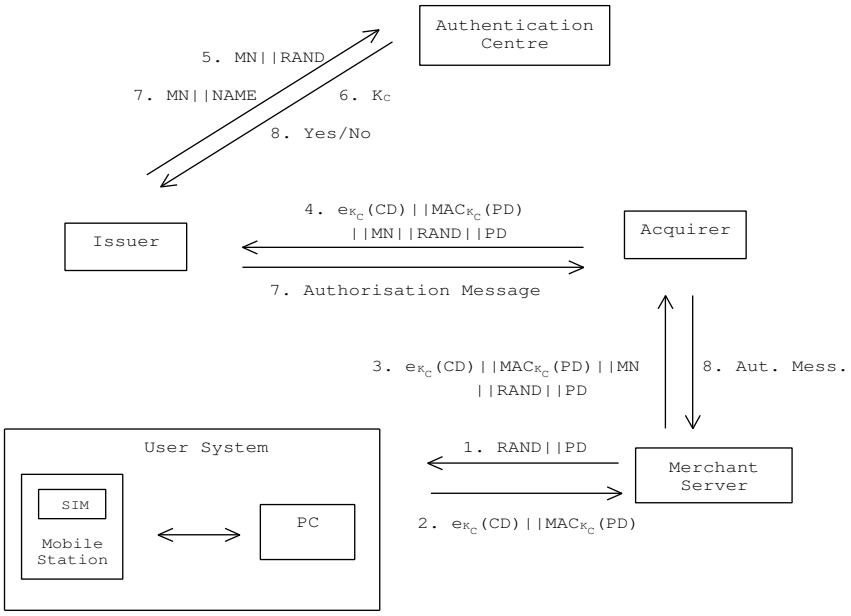


Figure 6.7: Revised protocol.

This modified protocol has the advantage that the AuC retains control of sensitive subscriber information. However, it has the disadvantage of requiring two more messages to be transferred, and also additional processing from the AuC.

Note that the revised protocol still allows the issuer and the acquirer to learn the phone number for the purchaser. To avoid this, the merchant server could send an encrypted MN to the acquirer (using a public encryption key for the AuC), although messages 2, 3 and 4 would then need to contain an identifier for the AuC of the user's home network.

6.4.3.5 Threats to the Authentication Centre

If the integrity of the AuC could be compromised, then there are possible attacks on the security of the user authentication and encryption processes. However, as stated in Section 6.3, in such an event there are also many other serious attacks to the security of the GSM network itself, and so we assume that the AuC is well-protected.

6.4.4 Advantages and disadvantages

In this section, the advantages and disadvantages of the protocol described in Section 6.4 are considered. The scheme has the same advantages and disadvantages as listed in Section 6.3.4, with the following additional advantages.

1. The protocol provides card details' confidentiality.
2. In the protocol, the merchant server has no access to the sensitive card details. As a result, the risks of storing unprotected card details in merchant servers are eliminated.

The disadvantages are essentially the same as those for the previous scheme, except that the issuer must establish a relationship with the mobile operator instead of the merchant (this may be rather simpler to achieve, since the number of issuers is much less than the number of merchants). As stated in Section 6.3.4,

to avoid the need for many individual arrangements between merchants and issuers, a Trusted Third Party (TTP) could be introduced to act as a 'broker' between the two parties.

6.4.5 Summary

We have proposed a second method of enhancing e-commerce transaction processing security by exploiting the existing GSM security features. The protocol provides user authentication and hence significantly reduces threats arising from misuse of misappropriated card details. It also eliminates the risk of storing card details in unencrypted form in merchant servers. As for the protocol described in Section 6.3, this protocol works with a 'standard' GSM SIM and requires only an appropriately equipped Mobile Equipment and a user PC. It therefore imposes minimal overheads on the user, thus increasing the likelihood of successful use. Again, the gains for the merchant in terms of reduced chargebacks and for the issuer in lessened card frauds also appear significant. The possibility of an increased revenue stream may also make the system attractive to GSM operators.

6.4.6 A comparison of the two proposed protocols

The protocol proposed in Section 6.3 makes use of the GSM authentication service while the protocol in Section 6.4 utilises the GSM data confidentiality service. The former provides only user/cardholder authentication whereas the latter provides also card details confidentiality and data integrity. In addition, the protocol proposed in Section 6.4 is a more complete payment protocol since it also offers transaction authorisation. However, by providing more security services and authorisation, the protocol is more complex, involving eight messages compared with four messages in the other protocol. Moreover, the second

protocol requires the card issuing bank and the acquiring bank to be involved in a card transaction, while the first protocol does not.

6.5 Related work

There exist other GSM-based payment systems which we now briefly review.

- The payment scheme proposed by Claessens et al. [9] provides user authentication using GSM. However, unlike the schemes discussed above, it makes extensive use of SMS messaging.
- The GiSMo (G i(nternet) S M o(pen)) scheme was developed by Millicom International Cellular in 1999. In this scheme, consumers must first open an electronic wallet over the Internet and supply their mobile phone number. Every Internet transaction is then validated with a password sent over the mobile phone using an SMS message. The GiSMo project, however, ended in 2001.
- Mint[3] and Paybox[4] are both GSM-based payment systems. They too require consumers to first open an e-wallet. Transactions in the two protocols involve either making or receiving calls using the designated mobile phone.
- The Visa 3-D Secure Protocol provides cardholder authentication for merchants using a card issuer server called the Access Control Server (ACS). As stated in Section 3.3.3 the cardholder must enroll before using this payment security scheme. The protocol can be extended to be used in mobile Internet devices such as WAP phones [89], in which case the transaction flow remains similar to that specified in [90].

[3] http://www.mint.nu
[4] http://www.paybox.co.uk

Broadly speaking, the other proposed GSM-based payment systems either use SMS messaging, require e-consumers to open an e-wallet, and/or require them to make or receive phone calls using a GSM phone. The protocols proposed here, however, do not impose any such requirements. They simply utilise the GSM subscriber authentication and data confidentiality services.

The Visa 3-D Secure Protocol is similar to the proposed protocols in that it also provides cardholder authentication. However, the Visa protocol requires, at minimum, the Visa Directory Server and the issuer ACS just to provide user authentication. The payment authorisation process then has to be performed separately. As a result, the two proposed protocols appear to have significant advantages over the 3-D Secure scheme. Similar remarks apply to MasterCard's SPA scheme.

6.6 Extending the protocols to 3G/UMTS

Since the third generation (3G) mobile communication system is now being implemented, in this section we describe a way in which the proposed protocols can be extended to utilise the security services offered by the third generation mobile system. Before doing so, however, it is necessary to briefly explain how the security mechanisms for the 3G mobile system operate.

6.6.1 3G/UMTS security

The Universal Mobile Telecommunications System (UMTS) is one of the 3G mobile systems and is the focus of this section. The 3G/UMTS security features outlined here are those given in the 3GPP specifications [1].

UMTS offers four main security services for the air interface. These are subscriber identity confidentiality, mutual authentication, data confidentiality,

and data integrity [1]. In this section, however, only the last three services will be discussed, since they are the ones relevant to our protocols. For more information on UMTS security, see, for example, [4, 5].

UMTS builds upon the GSM security features [5]. As in GSM, UMTS security is based on a secret key K shared between a User Services Identity Module (USIM), the UMTS equivalent of the GSM SIM, and the Authentication Centre (AuC). As in GSM, a data ciphering key CK is generated as a result of an authentication exchange between the ME and the base station. Again as in GSM, the parameters for this authentication exchange are generated in advance by the mobile user's home network. In addition, and unlike GSM, an integrity key, IK, is also generated as a result of this exchange.

6.6.2 Protocol extension

Since the subscriber authentication provided by UMTS is so similar to that of GSM, our first protocol can easily be extended without significant modification. The second protocol, however, can benefit from minor adaptation to take best advantage of the UMTS security features. Specifically, instead of using a variant of the encryption key K_c to compute the MAC, it is possible to use the integrity key IK, thereby avoiding any key separation issues.

UMTS divides its functional communications into four main strata, namely the application stratum, home stratum, serving stratum, and transport stratum. Therefore, it may be possible to place our protocols into the application stratum and configure the USIM and UMTS-capable ME to perform all the tasks of the user PC. If this is the case, the trust required in the user PC will be minimised or even eliminated.

6.7 Conclusions

In this chapter, two protocols are proposed to enhance e-commerce security by exploiting existing GSM security features. The two protocols provide card-holder authentication, and hence a measure of non-repudiation for the Internet payments, in a way that minimises the user overhead. The second protocol also provides confidentiality protection for the card details, including, for example, the account number. The proposed protocols can also be extended to use 3G security features with very minor modifications.

Chapter 7

The remaining threats

Contents

7.1 Introduction 131
7.2 Active content 131
7.2.1 Java applets 132
7.2.2 ActiveX controls 133
7.2.3 Security implications 134
7.3 Web browser flaws 136
7.4 Cookies 137
7.4.1 Monitoring user behaviour using cookies 138
7.4.2 Compromising confidentiality of cookie contents . . 139
7.4.3 Malicious cookies 140
7.5 Conclusions 141

The aim of this chapter is to identify threats which exist to all PCs when they are used for e-commerce. These threats exist regardless of the protection scheme employed for the transaction, and addressing them must be considered as part of the overall solution to e-commerce security. The threats are divided into three categories, namely those arising from active content, browser flaws, and cookies.

It is important to note that much of the material in this chapter has previously been given in [54].

7.1 Introduction

Sensitive information relating to e-commerce transactions is typically protected using SSL/TLS while it is being transmitted over the Internet between a consumer PC and a web server. Despite this, there remain significant threats arising from the use of a PC to conduct a transaction.

In the most typical case, an Internet user conducts an electronic payment transaction using a PC as a means to connect to the Internet. The security of an Internet transaction then also depends on how the user PC is configured. Most users, however, tend to use the default configuration. Therefore, in this chapter, we consider the remaining threats to SSL/TLS protected transactions based on the assumption that the user PC is in a 'standard' configuration.

There are numerous ways that a user PC can be attacked. However, in this chapter, we consider three main means that could be used to compromise the confidentiality, privacy, and/or integrity of the user's information in a web-based e-payment transaction scenario. These are through active content, browser vulnerabilities, and cookies.

7.2 Active content

Active content was introduced to overcome limitations of HTML, notably the lack of computational capabilities on the client side of a web session [27]. Java applets and ActiveX controls appear to be the most popular types of active content used for web-page design. Examples of less popular executable content include JavaScript, Telescript, and Word macros [64].

Although Java applets and ActiveX controls are similar in the way that they are used to add interactivity and animation to web pages, the security features

offered are somewhat different. In this section, Java security is first described followed by the security of ActiveX.

7.2.1 Java applets

Java was introduced in 1995. It is an object-oriented programming language. All Java programs use objects and every Java program is defined as a class, i.e. a collection of data, stored in named fields, and code, organised into named methods, that operates on that data [21].

Java programs can be run in two modes: in application mode and in applet mode [82]. There used to be no security restrictions for a Java application. In other words, Java applications had full access to system resources, just like any other programs. On the other hand, Java applets, as embedded in HTML documents, can be executed only from web browsers and must abide by the rules of a security mechanism called the Java sandbox [82]. This distinction between Java applets and Java applications has, however, become blurred over time. Indeed, in the most recent version of Java, users of Java applications are allowed to run an application within a sandbox that the user or system administrator has defined [69]. Therefore, to some extent the security of Java applications is now left to the user's or system administrator's discretion.

The Java sandbox, the core technology used to provide security for Java applets, consists of three main components, namely the bytecode verifier, the class loader, and the security manager [33, 64, 65, 69]. The bytecode verifier ensures that only legitimate Java applets that conform to the Java language specification, and that do not violate the Java language rules or name space restrictions, can be run [30, 33]. The class loader, on the other hand, is the component which is responsible for providing a particular class whenever it is needed. The class loader goes through certain steps to load and define a class. Details of such steps are outside the scope of this thesis and can be found, for

example, in [69]. The last component, the security manager, enforces access control to system resources and prohibits downloaded applets from executing arbitrary file input/output functions. For instance, an applet is prevented from reading from, or writing to, the file system of the computer on which it is running. An applet is also prevented from engaging in networking activities, such as listening to traffic or initiating network connections, apart from those back to the computer from which it originated [27].

While the Java sandbox has the advantage of preventing malicious applets from harming the user PC or accessing system resources, it also prevents them from doing many beneficial things. As a result, the concept of applet signing was introduced. A signed applet is then trusted and granted full access to system resources, just like a Java application. Clearly, signing applets cannot guarantee the safety of an applet [26]. Indeed, it only provides a way to determine the author of a malicious applet. Consequently, a recent version of Java (Java Developers Kit JDK 1.2) has a new security architecture which provides users or programmers with the ability to impose a fine-grained access control process, based on security policies and permissions [30].

7.2.2 ActiveX controls

An ActiveX control is an object that supports a customisable, programmatic interface[1]. Using the methods, events, and properties exposed by a control, it allows web authors to automate their HTML pages.

The security provisions for ActiveX controls are much simpler than those for Java executables. Indeed, ActiveX does not provide a security mechanism other than code signing. Just like a Java applet, a control can be signed using Microsoft's Authenticode [33]. A publisher can sign an ActiveX control by obtaining a Software Publisher Certificate (SPC) from a Certification Authority

[1]Details can be found at http://www.microsoft.com/com/tech/activex.asp

(CA). A signed control is then granted full access to system resources while access for an unsigned control is left to the individual. The security policy of ActiveX is therefore based on an all-or-nothing rule [26]. In other words, once download of a control is permitted, it either has complete access to the system resources or it is not run at all.

By comparison, a signed Java applet can be granted varying levels of privileges. This, however, does not necessarily imply that Java security is better than ActiveX. For Java applets, it is an individual decision whether or not to grant access to an applet and to decide the level of rights the applet is granted [27]. Consequently, Java content is more complex to manage, even though users have more control over downloaded applets.

7.2.3 Security implications

While SSL/TLS is typically used to protect Internet transactions, it cannot prevent a rogue applet from harming a user PC or compromising sensitive information. A user can still be lured to a malicious web site containing Java or ActiveX 'Trojan horses' which, for example, run a bogus dialogue box asking for a username and password, or record keystrokes. The site can even open an SSL/TLS connection with the user to make things appear more authentic and convincing.

A Java applet or an ActiveX control can be signed or unsigned. As stated above, the signatures on these applets provide no assurance that they will not behave maliciously. It is only the identity of the software developer which is verified, not the content itself. What the signature does provide is a method to support the generation of an audit trail since it enables the authentication of the author of the content. However, it may still be possible for a Java applet or an ActiveX control to conceal its activities by modifying the audit log after harming the machine. As a result, it becomes more difficult for the user to

identify which applet, and thence its author, has damaged their PC.

Various techniques have been proposed to control active content behaviour. One example is the set of mechanisms proposed by Hassler and Then [34], which are based on a method for monitoring and controlling Java applets and their threads running in a browser. They describe the implementation of a prototype monitoring applet, which, for convenience in constructing the prototype, was programmed as a normal applet, and hence has the same privileges as other applets. This limits the degree to which protection can be offered. Hence, and as described in [34], if this idea was used 'for real', then the monitoring applet would need to be implemented as part of the browser and given the highest privileges.

Another example is the Windows Personalisation technique, proposed by Tygar and Whitten [87]. This method is designed to counter attacks that imitate the visual appearance of a program that the user already trusts with sensitive information. The main idea is to use window appearances which are easily recognised by the user, but difficult to predict by an attacker. However, information for such personalisation still needs to be sent, and hence can be captured by a Java applet.

In [2] a technique called a Safe Interpreter is described which performs the functions of access control, independence of contexts and management of trust. The details of how the Safe Interpreter works are beyond the scope of this thesis. However, it is worth noting that, although the technique may be effective, it also increases system complexity and potentially damages performance.

In conclusion, dealing with the security issues raised by active content appears to be a problem to which a completely satisfactory solution does not yet exist. Whilst restricting the capabilities of ActiveX controls or Java applets will improve security, this is probably not a satisfactory solution for the majority of users, since it prevents the delivery of services which may be valued by users.

A possible way of solving such problem, however, may be to use a combination of two approaches: verifying a signature on the downloadable code, including trust chain verification, and monitoring the code as it executes. Nonetheless, the complexity of code monitoring must be limited to try to prevent serious damage whilst not dramatically degrading performance.

7.3 Web browser flaws

Web browsers are large and complex pieces of software and hence are likely to contain security flaws. From time to time, vulnerabilities in web browsers are exploited to compromise user PCs and/or associated security measures such as SSL/TLS. It is not the aim of this section to list all known web browser vulnerabilities. However, it is worth giving an example of such a flaw to show how they might pose threats to a user even when SSL/TLS is in use.

In May 2000, a report was published [8] indicating that Netscape browsers do not validate a SSL/TLS server certificate properly, allowing attackers to impersonate a site. According to the report, Netscape Navigator correctly checks the server certificate at the beginning of an SSL/TLS session. However, while the SSL/TLS session is active, all HTTPS connections to the server's IP address are assumed to be a part of this session, and therefore the certificate is not checked again. Instead of comparing host names to those of currently opened sessions, Navigator compares IP addresses. Given that IP spoofing is possible, and more than one host name can have the same IP address, there is potential for a security breach.

Most web browser security flaws are patched soon after they are discovered. However, whilst they are present, browser vulnerabilities can diminish or even disable essential functionalities of a security measure such as server authentication in SSL/TLS, as in the example above, or user authentication, as will be

seen in Section 7.4.2. It is also worth noting that there may be other security flaws in web browsers which are not generally known, and can be exploited by a malicious user. As stated above, web browsers are complex, and hence it is almost impossible to avoid the presence of some security flaws. Therefore, it is important that an Internet user should check and update his/her web browser as often as possible, to mitigate the risks of browser vulnerabilities being exploited.

7.4 Cookies

Cookies are pieces of information generated by a web server to be stored in a user's machine [62]. Their primary purpose is to enable clients to store protocol state, since HTTP operates in a stateless fashion. The information in cookies can be, for example, selected items in a user's shopping cart, authentication information used for accessing restricted pages, or account details. In a typical scenario, the first time a browser contacts a web server, a cookie is sent from the latter to the former. It is then stored in the user PC in a file called either cookies.txt, Cookies, or MagicCookie in the Windows, UNIX, and Macintosh operating systems respectively [62]. The next time the browser requests a web page from the web server, it sends the corresponding cookies (where cookies are indexed by the URLs of the remote servers which supply them).

Cookie-related security threats can be divided into three main categories, namely monitoring user behaviour using cookies, compromising confidentiality of cookie contents, and malicious cookies. In this section, we identify threats that cookies can pose to a user PC. Detailed analysis of cookie security is postponed until the next chapter.

7.4.1 Monitoring user behaviour using cookies

A particularly controversial issue concerning cookies relates to their possible use as tracking devices to follow user movements across the Internet [26]. Web-advertising agencies such as DoubleClick, Focalink, Globaltrack, and ADSmart run advertisement banners on various sites. Their clients add an <IMG> (image) tag to the client HTML page, pointing to a URL on the advertising agency's server [82]. When a web browser sees this <IMG> tag, it contacts the advertising agency server to retrieve the graphic. The first time the graphic is downloaded from the site, the user browser will receive a cookie containing a random ID. From then on, every time the browser connects to a site containing the agency's advertisement banners, it sends the cookie (the random ID) along with the URL of the page that is being read [82] back to the advertising agency server.

After a period of time, the advertising agency will be able to generate a user profile, revealing user browsing habits and interests. This might be used to improve advertising campaigns, to target advertisements to user interests, and to avoid repeatedly showing the same advertisements to a user [82]. The ability to track users is a potential violation of user privacy. It is also possible that advertising servers might share such information without user consent, although, at the time of writing, there is no strong evidence of such behaviour.

Even though using cookies as a tracking device may not reveal the actual identity of a user, the fact that an advertising agency server can maintain a list of URLs that a user has viewed can lead to a possible compromise of user personal details. In particular, if a web server uses the GET method to input data from an HTML form to a CGI script running on a remote server, the information will be sent as a part of the URL. Therefore, anyone who can read the URL will be able to obtain the information in the form.

7.4.2 Compromising confidentiality of cookie contents

In order to allow users to browse among restricted pages without repeatedly identifying themselves, cookies are used to store authentication information such as user names and passwords [26]. Consequently, it is important to ensure the confidentiality and authenticity of cookies storing such information. Otherwise, anyone who can access such cookies can potentially impersonate the user. Although, in many cases, authentication information stored in cookies is in a server-specific format, and hence the contents are not immediately obvious to the reader, it is still possible for an attacker to simply replay an intercepted cookie and impersonate a user.

Implementation flaws, particularly in web browsers, can create security vulnerabilities. For example, a vulnerability in Internet Explorer Version 4 and 5 for Windows 95, 98, NT, and 2000, allowed any site to see the content of other sites' cookies [32]. This is because the browser could not cope a site having a long URL ending with the domain name of another server with that other server. Consequently, it was possible for a malicious web site to give itself a long URL ending with a sequence of characters identical to the URL of another site, and the malicious site was then able to access cookies stored by that other site. If a cookie contains personal information, e.g. confidential data such as account details, then the consequences of such a vulnerability can be significant. Although the web browser flaw has been fixed, it is possible that there are other undiscovered vulnerabilities that can pose security threats to users.

A configuration flaw can also pose a threat to the confidentiality of cookie content. An example of this lies in the way that Netscape Navigator folders are sometimes stored in a publicly accessible directory. In an environment where there are not as many computers as users, it is not unusual to provide public spaces for users to access their data from any computers within the environment. Such spaces are accessible to any users with a valid username and password.

For example, a college can provide a drive for student to store their home pages. In this drive, each student has his/her own directory. Any students with a valid username and password can access the drive, and hence other students' directories. An empirical study [53] showed that a number of cookie files could be found in this publicly accessible drive. This is because some users had stored their Netscape Navigator folder in a publicly accessible directory. Since Netscape Navigator stores user cookie files in the user's Netscape folder, this means that user cookies will also be stored in a publicly accessible location, i.e. accessible by any other users with a legitimate username and password. The study showed that it was possible to use this weakness to obtain personal details, ranging from user names to full user details including contact addresses and telephone numbers.

7.4.3 Malicious cookies

It is often rather misleadingly stated that cookies are just text files, and hence are harmless. Although cookies are application data files, they can include special tags that can introduce executable code; for example, Microsoft Office application files can contain Macro viruses.

In HTML, in order to distinguish text from 'markup' symbols, a set of characters such as '<', which typically indicates the beginning of an HTML tag, are defined as special. Tags can either affect the formatting of a web page or introduce a program that will be executed by web browsers. For example, a <SCRIPT> tag introduces code from a variety of scripting languages [7, 75]. Many web servers use information stored in cookies to create dynamic pages. Therefore, if a cookie includes those special tags, when a page incorporating this cookie is displayed a malicious program can be called and executed. The security effects of such a program can range from alterations of the submitted form, to bypassing an authentication process. However, what exactly can be

done by the called program depends on the language in which it is written, as well as the web server's security context configuration.

7.5 Conclusions

In this chapter, threats which exist to all PCs used to conduct Internet transactions are examined. It is important to consider such threats since they exist regardless of the protection scheme used for the transaction. In this chapter, we divide such threats into three categories, namely those arising from active content, browser flaws, and cookies.

Active content, especially Java applets and ActiveX controls, can pose a variety of threats to the user PC. These include showing bogus dialogue boxes prompting for username and password entry, and monitoring keystrokes. A number of techniques have been proposed to control active content behaviour. However, enhancing the security of active content leads to a difficult dilemma. Imposing more control over active content gives greater security but, at the same time, can block many useful functions that active content might perform. Although fine-grained control could be imposed on a piece of code so that it is allowed to perform only the functions it is authorised to, such a mechanism is likely to be complex and hence difficult to manage, at least for the majority of users.

From time to time, browser vulnerabilities have been discovered. Developing and patching browsers is a continuous process. Browsers continue to grow in complexity, in parallel with just about every other major application, and host operating system. It is therefore almost impossible to avoid the presence of vulnerabilities and flaws. As a result, it is important for the user to be aware of such threats and to keep their browser updated.

Threats which arise from the introduction of cookies can be divided into three types, namely monitoring user behaviour using cookies, loss of confidentiality of cookie contents, and malicious cookies. In the next chapter, we propose two cookie encryption protocols which can be used to enhance the security of cookies.

Chapter 8

Enhancing the security of cookies

Contents

8.1 Introduction 145
8.2 Security requirements 146
8.2.1 Cookie confidentiality 146
8.2.2 Cookie integrity 147
8.2.3 Cookie authentication 147
8.3 Meeting the security requirements 148
8.3.1 Browsers 148
8.3.2 Secure channels 148
8.3.3 Access control for user PCs 149
8.3.4 Cryptographic protection within cookie files 150
8.3.5 Summary 150
8.4 Server-managed cookie encryption 151
8.5 User-managed cookie encryption 153
8.5.1 Using symmetric cryptography 153
8.5.2 Using asymmetric cryptography 155
8.6 User-managed cookie protocols 156
8.6.1 Cookie encryption using symmetric cryptography . . 156
8.6.2 Cookie encryption using asymmetric cryptography . 158
8.6.3 Comparisons 160
8.6.4 Security services 161
8.7 Secure cookies vs. user-managed cookies 161
8.7.1 User authentication 162

8.7.2 Integrity and confidentiality 162
8.7.3 Cookie authentication 163
8.7.4 Other issues 163
8.7.5 Further development 163
8.8 Summary and conclusions 164

The aim of this chapter is to describe possible ways to enhance the security of cookies, i.e. to protect their confidentiality and integrity, and to enable their source to be authenticated. The chapter starts with Section 8.2, a review of the security services which are required to defeat the threats outlined previously in Chapter 7. In Section 8.3, various mechanisms which can be used to meet the security requirements are examined. An existing server-managed cookie encryption approach, the 'Secure Cookies' protocol [71], is subsequently described in Section 8.4, followed in Sections 8.5 and 8.6 by detailed descriptions of two new user-managed cookie encryption protocols. The two approaches are then compared in Section 8.7. The last section summarises and concludes the chapter.

Much of the material in this chapter has previously been described in [55].

8.1 Introduction

Today, browsing and on-line shopping are becoming increasingly convenient. A user can personalise a web page, have his/her own shopping cart, and be automatically authenticated to a web server without repeatedly entering his or her username and password. However, the stateless nature of the HTTP protocol does not support such features, which are instead supported by files called cookies, stored on the user's PC. Cookies were specifically introduced to enable web servers to maintain current session state and recognise individual users.

A cookie consists of six elements, namely Name, Expiration Date, the Domain name, Path, Secure, and String Data. The first part is the name of the cookie. The expiry date defines the cookie's lifetime. The domain name and path are used when a browser searches for a cookie corresponding to the host of the requested URL. The path attribute specifies the subset of the URLs to which the cookie belongs. The secure attribute indicates whether the cookie is transmitted in secure mode such as TLS and HTTPS. The String Data field is where all other information of the server's choice is stored. Detailed cookie specifications can be found in a variety of places, see for example [61, 62, 68].

While cookies are clearly very useful, they can also be abused to impersonate a user, compromise user privacy and, in some cases, reveal confidential user information (see Section 7.4). Although a number of papers, e.g. [3, 26, 31, 62, 80, 82], point out potential security threats, most of them focus on facts about cookies, such as what they are and how they are used, and do not appear to provide a satisfactory security analysis and solution.

In [71] Park and Sandhu propose the 'Secure Cookies' method, in which security measures are applied to cookies by a web server. Consequently, this approach allows web servers to control what, when and how the security proce-

dures will be performed.

Whilst a server-managed approach may be appropriate for many applications, in some environments users will wish to control the security of their own cookies. This chapter, therefore, examines possible user-controlled approaches to enhance cookie security.

In Section 8.2, the security requirements necessary to defeat the threats identified in Section 7.4 are outlined. Various options available to meet the security requirements are then examined in Section 8.3. The server-controlled approach is then outlined in Section 8.4 followed by two different methods of realising the user-controlled approach (sections 8.5 and 8.6). A comparison between the server-managed and user-managed approaches is subsequently given in Section 8.7. The final section summarises and concludes the chapter.

8.2 Security requirements

In this section, a number of security requirements are identified to deal with the security threats discussed in Section 7.4. Note that these security requirements do not address the threat that cookies could be used to monitor user behaviour. Such threats are typically tackled by using tools specially designed to monitor the activity of cookies, such as Cookie Pal[1] or Cookie Crusher[2].

8.2.1 Cookie confidentiality

As stated in Section 7.4.2, cookies can be used to store authentication data with which a client uses to authenticate him/herself to the remote web server, and personal details such as mailing addresses and credit card numbers. Therefore,

[1] http://www.kburra.com/cpal.html
[2] http://www.thelimitsoft.com/cookie/

it is important to provide confidentiality for cookies so that such information can be protected. There are two ways in which information in cookies might be revealed. Firstly, a cookie can be intercepted while it is being transmitted, and, secondly, a cookie can be disclosed while it is stored in a user's machine. We consider ways in which both threats can be addressed.

8.2.2 Cookie integrity

In order to prevent attacks such as cross-site scripting, i.e. where special tags are inserted into cookies as described in Section 7.4.3, maintaining the integrity of cookie data is vital. Moreover, if a cookie is used to authenticate a user to a remote web server and the content of the cookie is changed, then the authentication process will fail. An attacker could thus modify such a cookie, and hence prevent a legitimate user from accessing a service. The integrity of the domain name and Path in cookies is also important. If it is possible to change these elements, then cookies can be sent to an entity other than the owner.

8.2.3 Cookie authentication

Although the content of a cookie may be encrypted and protected from unauthorised modifications, there remains the possibility of an attack where one entity 'presents' a cookie copied from another party. Consequently, it is important to be able to verify that the entity supplying a cookie is the owner of that cookie and the cookie is authentic.

8.3 Meeting the security requirements

There are various ways of meeting the security requirements stated above. This section examines the possible options in more detail, and considers their advantages and disadvantages.

8.3.1 Browsers

Although browsers do not satisfy all the security requirements stated earlier, current browsers provide an option which seems to enable users to control the use of cookies. To be more precise, they offer the users the choice of accepting all cookies, refusing all cookies, or displaying a warning message before accepting a cookie. However, this is likely to be insufficient, since it is difficult for a user to make a decision as to which alternative to choose. Selecting the first option may not be a good idea, since in this case users will have no control over the use of cookies. On the other hand, disabling all cookies will deny access to their useful features and, as a result, browsing will become stateless. Therefore, the last option seems to offer an attractive middle path to a user. However, this option is very intrusive, since, whilst web browsing, users will very frequently be asked whether or not they wish to accept a cookie. Moreover, users tend to either accept or reject all cookies, because it is hard to identify which cookies should be accepted and which should not. Choosing this compromise option is then little different from the first or second option.

8.3.2 Secure channels

One way to secure cookies is to protect the channel via which they are transmitted, i.e. the link between a web browser and a web server, by using protocols such as SSL/TLS or HTTPS. These protocols provide confidentiality and in-

tegrity protection for transmitted data using an encrypted channel and MACs respectively. However, a secure channel only protects the information against eavesdropping and modifications en route. Once the cookie reaches its destination it is no longer protected by SSL/TLS, and hence this option provides only partial protection. Anyone who has access to the stored cookie file will be able to read, change, or replay it.

An advantage of this option, however, is that it is transparent to a user. Moreover, it makes use of existing capabilities, and therefore does not require modifications to web browsers. However, in order to send a cookie securely, the cookie's 'Secure' attribute must be set. Since cookies are generated by servers, it is completely in the server's hands whether the 'Secure' attribute is set. Given that most users are not aware whether or not cookies are sent via a secure channel, many web servers send them in clear.

Secure channels can provide user authentication; however, even in the unlikely event that this is used, this does not guarantee the origin of a cookie. In order to provide authentication for a cookie, there should exist some means to link the user authentication used in the secure channel establishment process with the cookie itself.

8.3.3 Access control for user PCs

Another way of providing security for cookies is to protect user PCs against unauthorised access. A user authentication technique, e.g. using passwords, can restrict access to a PC, thereby protecting stored cookies against unauthorised reading and modification. The main advantage of such an approach is its simplicity. It is relatively easy to implement since most users are accustomed to using passwords. There is also no need to modify web browsers.

A disadvantage of this mechanism is that it only protects cookies while they

are in a PC. Therefore, it may have to be employed with other mechanisms to enhance security. Furthermore, it does not provide a mechanism to prove the authenticity of a cookie, since a malicious user can still reuse a stolen cookie. The use of passwords may also require additional management to prevent dictionary attacks.

8.3.4 Cryptographic protection within cookie files

A combination of cryptographic techniques can be used to meet all three security requirements for cookies. Cookie encryption can be used for confidentiality. Both cookie integrity and cookie authentication can be provided by using a MAC or digital signature. As part of cookie authentication, cookie replay protection can be achieved by incorporating cookie transfer into an authentication protocol (e.g. using a time stamp).

Cookie encryption and integrity protection can protect a cookie both when it is stored and when it is transmitted, unlike secure channels which do not protect stored cookies (see Section 8.3.2). There is thus no need for additional access control to user PCs. However, a disadvantage of using cryptographic techniques is that keys are required. Key management issues, such as how to securely exchange the keys, where they should be stored, and who should keep them, have to be taken into account. Additionally, there is a possibility that web browser modifications or additional software will be required.

8.3.5 Summary

In practice, a system can combine some or all of the four methods described above to enhance security. However, the last technique, i.e. applying cryptographic protection to the cookie file itself, appears to offer the widest range of security services. For this reason, this approach is the focus of the remainder of

the chapter.

Applying cryptographic measures to cookie files can be performed by web servers or web browsers. Whichever does so will have control over what, when and how the measures are performed. In the remainder of this chapter, we examine the two approaches, and consider their respective merits.

8.4 Server-managed cookie encryption

Server-controlled cookie encryption has the major benefit of user transparency. If implemented appropriately, no changes to web browsers will be required. A disadvantage of this approach is obviously that users will have little control over the protection of their own cookies. An example of this approach is the Microsoft Passport scheme[3] which was introduced to provide an online user-authentication service where a user presents a cookie to authenticate him/herself to a remote web server. It employs encrypted cookies as a means of exchanging user-authentication information between a Microsoft Passport server and participating web sites. Another example of server-managed cookie encryption is the 'Secure Cookie' scheme proposed by Park and Sandhu [71].

Although these two schemes are similar in the way that they use encrypted cookies, the latter is more general in that it is a means of protecting all user cookies, and not just those generated by a single application. As a result, we use the Park and Sandhu 'Secure Cookie' scheme as the basis for a comparison with the new user-managed cookie security scheme proposed in Section 8.5. In the remainder of this section, we provide a brief overview of the Park and Sandhu scheme.

In this approach, web servers are required to use 'Secure Cookies' of specific kinds, each with a predefined type of content and protection. Examples include

[3]details are available at http://www.passport.com/

Name Cookies, Life Cookies, Key Cookies, and Seal Cookies. A Name Cookie, for example, contains a user name that can be used to authenticate a user to a server, and a Key Cookie contains an encryption key. The integrity of all cookies is protected by a Seal Cookie that holds either a MAC or a signed hash of the other cookies.

In order to have a set of Secure Cookies, a web browser needs to contact another server called the Cookie Issuer, which generates the Secure Cookies. The web browser then sends the cookies to the web server, which will verify or decrypt them as appropriate. Examples of Secure Cookies are listed in Table 8.1.

Table 8.1: Secure Cookie Components

Domain	**Flag**	**Path**	**Cookie_Name**	**Cookie_Value**	**Secure**	**Date**
acme.com	True	/	Name_cookie	Alice	False	12/31/2003
⋮	⋮	⋮	⋮	⋮	⋮	⋮
acme.com	True	/	Life_cookie	12/31/02	False	12/31/2003
acme.com	True	/	Pswd_cookie	Hashed password	False	12/31/2003
acme.com	True	/	Key_cookie	Encrypted key	False	12/31/2003
acme.com	True	/	Seal_cookie	Signed Message Digest of MAC	False	12/31/2003

This approach satisfies the security requirements of confidentiality, integrity, and user authentication, by using encryption, a signed message digest or MAC, and a digital signature respectively. However, it does not provide protection against replay attacks. A stolen Secure Cookie can be submitted to the web server.

The Key Cookie, as stated earlier, stores a session key that is used to encrypt and decrypt other cookies. The session key can be encrypted using either a server public key or a server secret key. In either case, web servers are responsible for key management.

While this approach may be appropriate in many applications, in some circumstances users may wish to read their own cookies and control their security. Consequently, in the next section we examine possible approaches that give

users more control.

8.5 User-managed cookie encryption

With a user-managed approach, users obviously have the benefit of control over what, when and how the security mechanisms should be applied. However, a special web browser or additional software is required in order to enable users to perform the security procedures. The client may also have to store cryptographic keys, which could be a security threat in some circumstances. As a result, there is a need for a key management system to support the use of cryptography.

In this section, two possible approaches, using symmetric and asymmetric cryptography, are described at a high level (in Section 8.6 these schemes are given in more detail). In order to provide cookie confidentiality, integrity and authentication, the schemes use encryption, MACs, signatures, and time stamps. The security mechanisms described below will be applied only to the cookie value, to minimise the complexity of the protocols.

8.5.1 Using symmetric cryptography

In this approach, a user selects cookie encryption by sending a request for cookie encryption to the web server. This will trigger a key establishment protocol. If a user chooses to encrypt cookies, he/she will be required to authenticate him/herself to his/her PC and the key management application, e.g. by using a password. This is required in order to prevent an unauthorised user, who may have access to the user's PC, from activating the security procedure and using cookies. Key establishment can be performed by sending the key via a secure channel, or by using other key distribution techniques such as those involving a trusted third party. After successful key establishment, the user and the server

then share a secret cookie key and the server will encrypt cookies with the key and send them to the user. The encrypted cookies are then stored locally in the user's PC.

The next time the web browser requests a web page from the site, it looks for corresponding encrypted cookies. A time stamp is generated and concatenated with the cookies, and a MAC is computed on this data. A page request, the time stamp, the encrypted cookies and the MAC are then sent to the server. On receipt of the request, the server verifies the MAC and checks whether the time stamp is within the acceptance window. The server can change information in the cookies whenever the user requests a page, and the new encrypted cookie will be sent back to the user with the requested page.

A time stamp and a MAC are included in order to prevent an intercepted cookie from being replayed. Without knowing the secret cookie key, an adversary will not be able to create a valid MAC. There are no cryptographic requirements for time stamp generation — the time stamp only needs to be within the acceptance window.

Given that symmetric cryptographic operations are typically simple to compute, the encryption operation will not significantly increase the server workload. Users only need to decrypt a received cookie if they want to see the content. However, this approach needs a secure means to distribute the secret key the first time the user and server communicate. Moreover, users and servers need to maintain the shared secret key. As the number of users (n) and servers (m) grows, the total number of keys will be bounded above by mn, and the task of key management will therefore become increasingly complicated over time. More generally, one effect of this is to make cookie management by servers a stateful process, i.e. servers are required to maintain state for web users.

Since the security of this approach depends on the secrecy of the shared key, it is vital to store the key securely. To meet this requirement, a user could store

the keys in a smart card or in a password-protected file on his/her PC. In the former case, the user would typically be required to authenticate him/herself to the card before the card will perform any computation using its stored keys.

8.5.2 Using asymmetric cryptography

In this approach, users are required to have an asymmetric key pair and a certificate for the public key from this key pair. If a user wishes to have his/her cookies encrypted, the first time the user requests a web page his/her certificate and the page request message will be signed and sent. The server then generates a secret key, encrypts the cookies with this secret key, encrypts the secret key with the user's public key, and sends the encrypted secret key with the encrypted cookies and the requested page. There is no need for the user to decrypt the cookie unless he/she wants to know the cookie content. The next time the user contacts the server, the encrypted cookie, a time stamp, and a MAC is sent with a web page request.

As for the symmetric technique, this approach allows users to decide if they want to encrypt the cookie or not. If the user sends a certificate, the server will know that the cookie must be sent encrypted. The user will also be required to enter a password for user authentication to the PC, since his/her private key may be stored locally in the PC. It is also possible for the private key to be stored in another more secure way, e.g. on a smart card. In such case, the user is still required to authenticate him/herself to the device to prevent unauthorised use. As for the symmetric cryptography approach, the time stamp is used to prevent replay of an intercepted cookie.

A drawback of this technique is that certificates and key pairs are required for the client. A Public Key Infrastructure (PKI) will also be needed to create and manage public key certificates. Again, as with the symmetric cryptography scheme, the protocol makes web serving stateful. However this would appear to

be unavoidable.

8.6 User-managed cookie protocols

In this section, two protocols for user-managed cookie security are described in detail, building on the general approaches described in the previous section.

8.6.1 Cookie encryption using symmetric cryptography

In this approach, a secure channel is employed to distribute a cookie key. How this secure channel is established is outside the scope of this paper, but it could, for example, be provided using protocols such as TLS and HTTPS.

The protocol is defined in Figure 8.1. In the protocol description:

- 'Client' can represent additional software, e.g. a modified web browser, a plug-in, or an applet which performs security procedures for users,
- 'Server' represents a web server,
- $X||Y$ denotes the concatenation of data items X and Y,
- K denotes a secret key used to encrypt cookies (the 'cookie key'),
- $e_K(M)$ denotes message M encrypted (using symmetric encryption) with key K,
- 'Key ID' identifies the cookie key K used to encrypt the cookie,
- T denotes a time stamp, and
- $\text{MAC}_K(M)$ denotes a MAC on message M using a variant of key K (note that it is important that the key used to compute the MAC is not precisely the same as the key used for encryption, particularly if the MAC is a

CBC-MAC based on the same block cipher as used to perform cookie encryption).

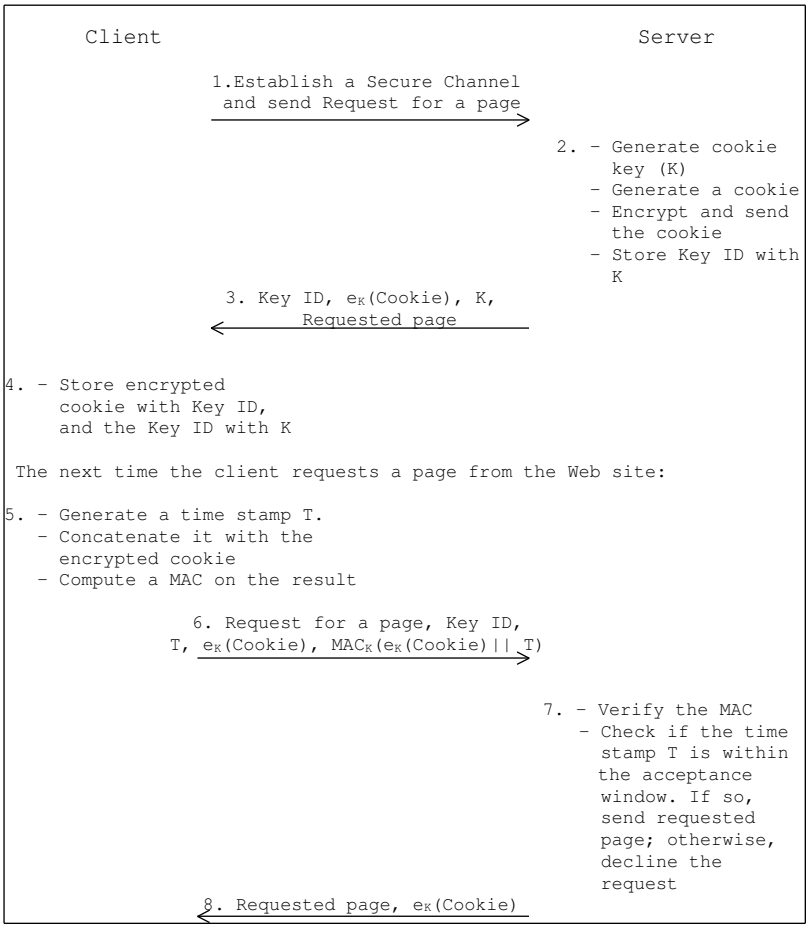


Figure 8.1: Cookie encryption using symmetric cryptography.

The protocol starts when a user chose to encrypt cookies. The client first sends a request for a page. Upon receipt of this request, the server generates the cookie key (K) and cookie(s) which is (are) subsequently encrypted using the key. It is important for the server to assign a key ID to each key so that the right key will be retrieved to decrypt the cookie in the future. The key ID is then stored by the server with the corresponding key. The cookie key, the key

ID, and the encrypted cookie are then sent to the user as illustrated in step 3 in Figure 8.1. Although the cookie key is provided by the server, the user does not need to decrypt the cookie to complete the protocol. The information received from the server only needs to be stored on the user PC.

The next time the client requests a page from the server, it first generates a time stamp (T), concatenates it with the encrypted cookie, and computes a MAC on the result. The request for a page, the key ID, the encrypted cookie, the MAC, and the time stamp are then sent to the server (step 6). Upon receipt of the message, the server first verifies the MAC and checks whether the time stamp is within the acceptance window. If both checks are positive, the requested page with updated encrypted cookie are sent to the client (step 8). Otherwise, the server rejects the client request and the protocol ends.

8.6.2 Cookie encryption using asymmetric cryptography

The public key based scheme is specified in Figure 8.2. In the protocol description, the following notation is employed (in addition to that used in Figure 8.1):

- $E_P(X)$ denotes data X encrypted (using asymmetric encryption) with public key P,
- $S_C(M)$ denotes the signature of the client on message M (computed using the client private key), and
- P_C denotes the public key of the client.

As in the symmetric approach, the protocol starts when the user chooses to encrypt cookies. The request for a page is then sent together with the user public key certificate and a signature on the request concatenated with the certificate. The signature is particularly important because it allows servers to detect an attack where a malicious user deletes the request for cookie encryption (the user

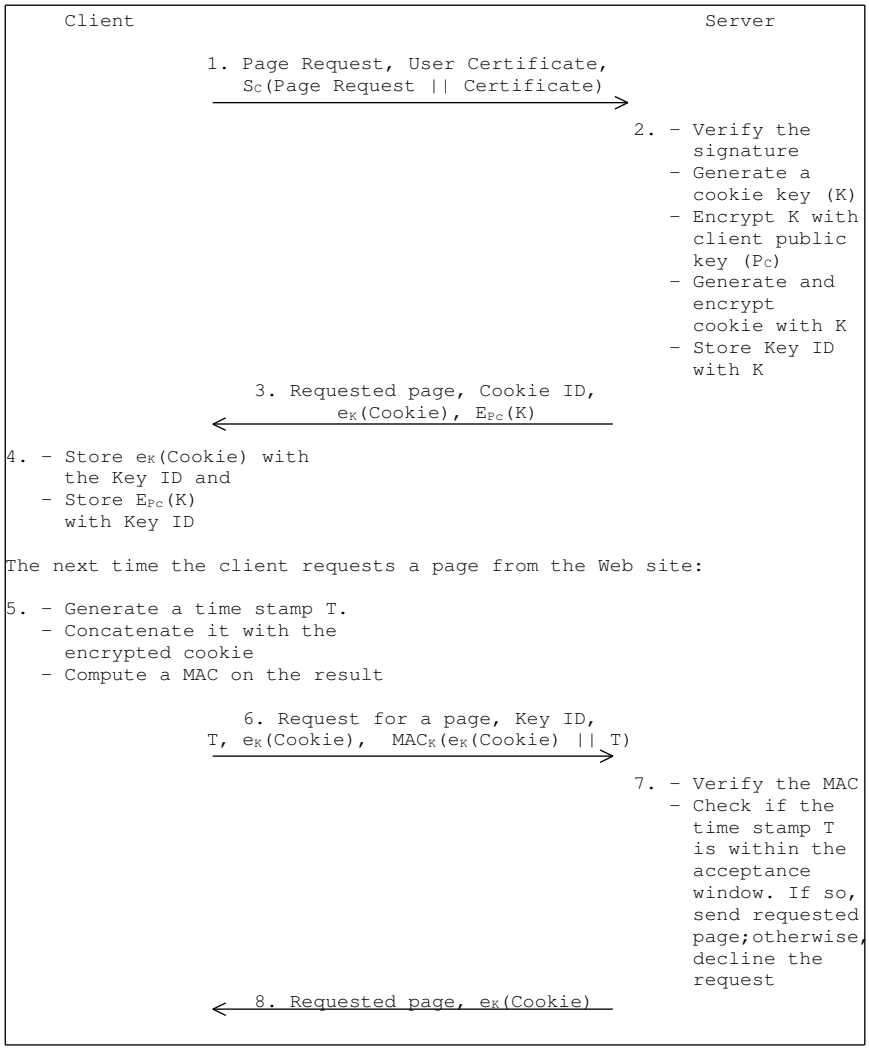


Figure 8.2: Cookie encryption using asymmetric cryptography.

certificate). If an attack is detected, the server can send a message to inform the user and ask if the user wants to try again. This, however, introduces a risk of denial of service where a malicious user keeps modifying the message causing the page request to fail. Moreover, the additional computation of the signing process can increase the user machine's workload and possibly slow the page request process.

Upon the receipt of message 1 in Figure 8.2, the server verifies the signature, generates a cookie key (K) and encrypts it with the client public key (P_C). It then generates and encrypts cookie(s) with the key K and assigns an ID to K. The key ID is subsequently stored with its corresponding cookie key.

The server constructs a message consisting of the requested page, the cookie ID, the encrypted cookie, and the encrypted key. It then sends the message, which corresponds to message 3 in Figure 8.2, to the client. Upon receipt of the message, the client stores the encrypted cookie with the key ID, and the encrypted key with the key ID. The next time the client contacts the server, the process resembles that in the symmetric approach.

8.6.3 Comparisons

The main advantage of the symmetric approach is convenience. It makes use of existing security protocols, i.e. SSL/TLS, to distribute the cookie key. However, doing so requires the establishment of such secure channel. Another drawback is that key management is relatively complicated, since there will be a large number of cookie keys for users to manage and store securely.

On the other hand, the main advantage of the asymmetric approach is that the key management task is not so complicated. Key distribution is also simpler than in the system based on symmetric cryptography. Moreover, cookie keys are stored encrypted. The only key a user has to keep secret is the private key

needed to decrypt the cookie keys.

8.6.4 Security services

The user-managed cookie approach satisfies all three security requirements. In addition to those described in Section 8.2, it also offers user authentication. The two proposed protocols provide the four security services in the following ways.

- **Confidentiality**: The two protocols proposed here use either symmetric or asymmetric encryption to provide cookie confidentiality.
- **Cookie Integrity**: The symmetric approach uses only a MAC to protect the integrity of the cookie. The asymmetric approach also requires use of a signature to ensure that a malicious party will not be able to delete the request for cookie encryption.
- **Cookie authentication**: The protocols both use a time stamp to provide protection against replay attacks.
- **User authentication**: The user-managed cookie approach provides this security service by requiring users to enter a password to activate the security protocol and use cookies.

8.7 Secure cookies vs. user-managed cookies

In this section, a detailed comparison between the server-controlled and user-controlled approaches is provided.

8.7.1 User authentication

The Secure Cookies method [71] offers three choices for a user to authenticate him/herself to a server, namely by IP address, by signature, and by password. The first alternative is not always appropriate since in some environments IP addresses are assigned dynamically. Moreover, it is prone to an IP spoofing attack.

The second user authentication mechanism is signature-based, and requires users to have a public key pair in order to sign a time stamp in a cookie. However, the signing process can increase the user workload, and hence possibly slow the web-browsing process.

Both Secure cookies and user-managed cookies employ user passwords. However, in the Secure Cookies technique the password must be sent via a secure channel to a web server for verification. Therefore, in addition to the cookie security process, a secure channel may be needed. By contrast, in the user-managed methods the password is verified locally, which lessens its exposure and avoids the need for secure channel establishment.

8.7.2 Integrity and confidentiality

The Secure Cookies scheme uses signatures and MACs. The user-managed cookie security schemes also use either a signature or a MAC. In both cases, an additional computation for the encryption process is required.

Both techniques offer a choice between symmetric and asymmetric encryption to provide cookie content confidentiality. However, one disadvantage of the user-managed scheme is that the required cryptographic keys will have to be stored by the clients.

8.7.3 Cookie authentication

In the Secure Cookies approach, there is no mechanism for replay protection. The user-managed cookie security schemes, however, use a time stamp to provide protection against replay attacks. However, a MAC must be included in order to prevent a malicious user from replaying a stolen cookie with a new time stamp, potentially increasing the user's workload.

8.7.4 Other issues

The main advantage of the user-managed approach over Secure Cookies is probably the fact that it allows users to have more control over the security of their own cookies. By contrast, with Secure Cookies, web servers have full control over cookie encryption. Although this may be appropriate for many applications, in some environments users may wish to have more control over their security — indeed, if users wish to read their own cookies, e.g. to deal with the threat of user tracking, then a user-controlled approach is essential. Moreover, the Secure Cookies method may be more complex since a set of cookies is required for each web site. There is also a need for an Issuing Cookie server to generate the Secure Cookies.

8.7.5 Further development

Recently another server-controlled protocol has been proposed by Yang and Rhee [94]. This protocol, however, is not entirely server-managed. Rather, it is a combination of user and server managed approaches. The authors propose use of a set of secure cookies similar to those used in the Park and Sandhu scheme. The cookies are then encrypted with a secret key which is sent encrypted with the server public key, as in the protocol described in Section 8.6.2. The Yang and Rhee protocol differs from the protocol in Section 8.6.2 in that the client

generates the secret cookie key rather than the server.

8.8 Summary and conclusions

Although cookies are a very useful mechanism for maintaining session state, as discussed earlier they can pose a number of security threats. As a result, three cookie security requirements, namely confidentiality, integrity, and authentication, can be identified.

Secure Cookies [71] and the protocol proposed by Yang and Rhee [94] are server-controlled approaches to cookie protection that satisfy some of these security requirements. However, in some environments users may want to control their own cookie security, especially if the user tracking threat is to be effectively combated. In this chapter, two user-controlled approaches, using symmetric and asymmetric encryption of cookies, are proposed. The main differences between how security services are provided in the server-managed 'Secure Cookies' and the user-managed approaches are summarised in Table 8.2.

Table 8.2: Comparisons of server-managed and client-managed cookies

Security Requirements	**User-Managed Cookie Encryption**	**Server-Managed Secure Cookies**
User Authentication	Password-based	IP Address, Password, Signature
Integrity	Signature, MAC	Signature, MAC
Confidentiality	Encryption	Encryption
Replay Protection	Time Stamp	N/A

The 'Secure Cookies' approach is potentially more flexible since it offers various options to provide each security requirement. It can also be made transparent to the user's web browser. However, it is relatively complex because it requires an additional server to generate Secure Cookies. Moreover, in most cases, it requires a number of cookies, while in the user-managed approaches a variety of information can be stored in a single cookie. Finally, the Secure

Cookies scheme has the potentially major disadvantage that it prevents users examining the contents of cookies stored in their own machine.

If a password scheme is used, the Secure Cookies method requires it to be sent via a secure channel. On the other hand, the user-managed techniques verify a password locally, and hence minimise its exposure. The user-managed approaches also provide additional protection against replay, while the Secure Cookies scheme does not.

Chapter 9

Conclusions and directions for future research

Contents

9.1 Summary and conclusions 167
9.2 Directions for future research 170

The aim of this chapter is to conclude and summarise the primary contributions of this thesis (Section 9.1). In Section 9.2, suggestions for future research are given.

9.1 Summary and conclusions

This thesis deals with security in electronic commerce transaction processing. The main focus has been on methods designed to enhance the security of SSL/TLS protected transactions made over the Internet.

As discussed in Section 3.2, there are five main security requirements for e-commerce transaction processing, namely confidentiality, authentication, non-repudiation, integrity and replay protection. By far the most widely used protocol to provide security for Internet transactions is SSL/TLS. However, the analysis in Chapter 4 reveals that the protocol provides confidentiality and integrity only while the information is being transmitted. Once the information has reached its destination, SSL/TLS offers no protection. Consequently, information can be compromised if either end of the communications link has been penetrated.

Moreover, SSL/TLS only obligates server authentication. Therefore, it is possible for anyone who has access to the client's PC to impersonate the client. SSL/TLS also does not protect any party against repudiation. SSL/TLS does provide partial protection against replay attacks; in particular it prevents a third party using an intercepted SSL/TLS message. Nonetheless, it does not prevent malicious merchants or clients from re-using a transaction.

Because of these shortcomings in the level of security provided by SSL/TLS, four schemes for enhancing the security of e-commerce transactions have been proposed; in each case the schemes build upon the level of security provided by SSL/TLS. In Chapter 5, a way to use an EMV-compliant IC card to enhance security of e-commerce transactions has been described. The scheme provides cardholder authentication, non-repudiation, confidentiality, and integrity. The use of 'trusted' card readers in conjunction with the proposed protocol is also examined. If a trusted card reader is used, then confidentiality protection for

the PIN is enhanced and cardholder control is gained over precisely which value of transaction is authorised. Moreover, the fact that the software required in the Cardholder System can be made simpler makes it easier to perform a correctness check on such software.

In Chapter 6, two protocols have been proposed in which GSM subscriber authentication and data confidentiality are utilised to enhance e-commerce transaction security. The two protocols provide cardholder authentication as well as a measure of non-repudiation. The protocol which exploits the GSM data confidentiality service also provides card details confidentiality and transaction authorisation protection. With some minor modifications, the two protocols can also be extended to use the 3G security features.

Even if the proposed schemes are used, threats still exist to all PCs used to conduct Internet transactions. In Chapter 7, we discussed three major categories of such threats, namely threats arising from active content, browser flaws, and cookies.

Active content, especially Java applets and ActiveX controls, can pose a number of threats to the user PC. A variety of techniques have been proposed to control active content behaviour. However, addressing the security threats posed by active content is not simple. While it is possible to impose more control over active content, doing so will limit many of its useful functions.

Browsers are complex pieces of software. Discovering and patching their vulnerabilities seems to be a continuous process. It would appear that it is almost impossible to avoid the presence of vulnerabilities in such complex pieces of software. The important issue is that the user must be made aware of such threats and keep their browser updated.

Threats arising from the introduction of cookies were examined. Such threats can be divided into three main types, namely monitoring user behaviour using

cookies, loss of confidentiality of cookie contents, and malicious cookies. The existence of these threats has motivated the design of two cookie encryption protocols.

First, the main security requirements for cookies, namely confidentiality, integrity and authentication, were identified. Two user-controlled cookie protection schemes, using symmetric and asymmetric encryption of cookies, are then proposed. The protocols have been compared with Secure Cookies, a server-controlled approach to cookie protection proposed by Park and Sandhu [71]. The Secure Cookies scheme satisfies some of the security requirements. However, it is a server-controlled approach and in some environments users may want to control their own cookie security. The novel protocols described in this thesis offer such control to users.

Although the 'Secure Cookies' approach is potentially more flexible than the user-controlled cookie security protocols, it is relatively complex since an additional server is required to generate Secure Cookies. Moreover, in most cases, it requires the generation of a number of cookies. In our approaches however, a variety of different types of information can be stored in a single cookie.

If a password is used to authenticate users, the Secure Cookies method requires it to be sent via a secure channel to the web server. By contrast, the techniques described in this thesis verify passwords locally, and hence minimise their exposure. The user-managed approaches also provide additional protection against replay, while the Secure Cookies scheme does not.

9.2 Directions for future research

Many research issues remain in the area of e-commerce security, and major new problems are likely to emerge with the growth in ubiquitous Internet access and mobile computing. Examples of research issues requiring attention include the following.

- Personal information processing devices, such as personal organisers and mobile phones, are becoming increasingly common. These devices may be used to enhance the security of Internet payments. For example, a mobile phone could be used in place of a user PC to conduct an electronic transaction. Whilst there are many proposed mobile commerce schemes e.g. those using dual slot mobile phones, none of them has been widely adopted. It is therefore interesting to investigate what the obstacles are to their adoption and to find suitable solutions.

- It is interesting to see how biometric user authentication techniques could be used to provide cardholder authentication. Since biometric schemes involve comparing a measurement of some human characteristic with a template, it is important to prevent both replay attacks and breaches of user privacy. If biometric techniques are used to enhance the security of e-commerce transactions, it becomes a matter of critical importance to decide where certain functions are implemented. For example, where should the comparison between the biometric measurement and the template take place? This has an effect on where templates are stored — a critical user privacy issue. Also, what measures are employed to ensure the 'liveness' of the biometric measurement? Complete solutions to these questions in the context of current and emerging e-commerce scenarios do not appear to be available.

- Active content is very useful and at the same time is very dangerous. As

> we have seen, existing approaches to controlling active content are not really very useable. For one reason or another, users typically configure their systems to accept all content or none. Code signing by itself is clearly not the answer, since it only guarantees the origin of the code and not its safety properties. New, more sophisticated, yet easily managed, approaches to mobile code authorisation are required. This is certainly an area of considerable current research interest.

In parallel with the development of new security schemes, there is a need to further evaluate and compare existing proposals. For example, it would be of interest to evaluate the security and efficiency of the schemes proposed in this thesis. There are various ways in which this might be done.

One option would be to attempt a mathematical proof of security properties of the schemes. However, this is likely to be a highly non-trivial exercise, as the protocols involve three or more parties, and the trust relationships and protocol objectives are far from simple. It is not clear whether any of the existing formalisms could capture the important security properties. A second, and rather less ambitious option, would be to prototype the schemes. However, it is far from clear what such an approach would achieve — prototyping would certainly give no guarantees about the security of the schemes, and unless performed very carefully might only provide very limited information about practicability.

In general, more fundamental research is required on appropriate methods for testing the security of electronic payment systems. This is, of course, a very large research area in its own right.

Bibliography

[1] 3rd Generation Partnership Project (3GPP). *3GPP TS 33.102: Security architecture*, December 2002.

[2] V. Anupam and A. Mayer. Security of web browser scripting languages: Vulnerabilities, attacks and remedies. In *Proceedings of the seventh USENIX Security Symposium*, pages 187–200. USENIX, January 1998.

[3] H. Berghel. Caustic cookies. *Communications of the ACM*, 44(5):19–22, May 2001.

[4] C. W. Blanchard. Wireless security. In R. Temple and J. Regnault, editors, *Internet and Wireless Security*, pages 147–162. The Institution of Electrical Engineers, London, 2002.

[5] K. Boman, G. Horn, P. Howard, and V. Niemi. UMTS security. *Electronics Communication Engineering Journal*, 14(5):191–204, October 2002.

[6] Bureau of Export Administration, Department of Commerce. Revisions to encryption items. *Federal Register*, 65(203), October 2000.

[7] The CERT Coordination Center. Malicious HTML tags embedded in client web requests, February 2000. Available at http://www.cert.org/advisories/CA-2000-02.html.

[8] The CERT Coordination Center. Netscape Navigator improperly validates SSL sessions, May 2000. Available at http://www.cert.org/advisories/CA-2000-05.html.

[9] J. Claessens, B. Preneel, and J. Vandewalle. Combining World Wide Web and wireless security. In B. De Decker, F. Piessens, J. Smits, and E. Van Herreweghen, editors, *Advances in Network and Distributed Systems Security*, Proceedings of IFIP TC11 WG11.4 First Annual Working Conference on Network Security, pages 153–171. Kluwer Academic Publishers, Boston, 2001.

[10] D. D. Clark and D. R. Wilson. A comparison of commercial and military computer security policies. In *Proceedings of the 1987 IEEE Symposium on Security and Privacy*, pages 184–194. IEEE Computer Society Press, 1987.

[11] T. Dierks and C. Allen. *The TLS protocol version 1.0 — RFC 2246*. IETF, January 1999.

[12] W. Diffe and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22:644–654, 1976.

[13] EMVCo. *EMV 2000 Integrated Circuit Card Specification for Payment Systems Version 4.0 — Book 1: Application Independent IC Card to Terminal Interface Requirements*, December 2000.

[14] EMVCo. *EMV 2000 Integrated Circuit Card Specification for Payment Systems Version 4.0 — Book 2: Security and Key Management*, December 2000.

[15] EMVCo. *EMV 2000 Integrated Circuit Card Specification for Payment Systems Version 4.0 — Book 3: Application Specification*, December 2000.

[16] EMVCo. *EMV 2000 Integrated Circuit Card Specification for Payment Systems Version 4.0 — Book 4: CardHolder, Attendant, and Acquirer Interface Requirements*, December 2000.

[17] ETSI. *Digital cellular telecommunications system (Phase 2+); Security aspects (GSM 02.09 version 8.0.1)*. European Telecommunications Standards Institution (ETSI), June 2001.

[18] ETSI. *Digital cellular telecommunications system (Phase 2+); Security related network functions (GSM 03.20 version 8.1.0)*. European Telecommunications Standards Institution (ETSI), July 2001.

[19] E. Felten, D. Balfanz, D. Dean, and S. Wallach. Web spoofing: An Internet con game. In *Proceedings of the twentieth National Information System Security Conference, Baltimore, Maryland*, pages 95–104. Computer Security Resource Center, October 1997.

[20] Federal Information Processing Standards (FIPS). *FIPS PUB 186-2: Digital Signature Standard*. Gaithersburg, MD, January 2000.

[21] D. Flanagan. *Java in a nutshell*. O'Reilly, 1999.

[22] W. Ford. *Computer communications security: Principles, standard protocols and techniques*. Prentice Hall, 1994.

[23] W. Ford and M. S. Baum. *Secure Electronic Commerce: Building the infrastructure for digital signatures and encryption*. Prentice Hall PTR, second edition, 2001.

[24] Electronic Frontier Foundation. *Cracking DES: Secrets of Encryption Research, Wiretap Politics, and Chip Design*. O'Reilly, Sebastopol, CA, 1998.

[25] A. O. Freier, P. Karlton, and P. C. Kocher. *The SSL protocol version 3.0*. Netscape, 1996.

[26] S. Garfinkel and G. Spafford. *Web Security & Commerce*. O'Reilly, 1997.

[27] A. Ghosh. *E-Commerce Security*. John Wiley and Sons, Inc., Third Avenue, New York, 1998.

[28] D. Gollmann. What do we mean by entity authentication? In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 46–54. IEEE Computer Society Press, 1996.

[29] D. Gollmann. *Computer security*. John Wiley and Sons, 1999.

[30] L. Gong. *Inside Java^{TM} 2 Platform Security: Architecture, API design, and implementation*. Addison-Wesley, 1999.

[31] B. Hancock. Security views: Some cookies are not tasty. *Computers & Security*, 17(5):374–376, 1998.

[32] B. Haselton and J. McCarthy. Internet Explorer open cookie jar. Available at http://www.peacefire.org/security/iecookies/, May 2000.

[33] V. Hassler. *Security Fundamentals for E-commerce*. Artech House Publishers, 2001.

[34] V. Hassler and O. Then. Controlling applets' behaviour in a browser. In *Proceedings of the 14th Annual Computer Security Applications Conference (ACSAC'98, Dec 7-11, 1998, Scottsdale, Arizona, USA)*, pages 120–128. IEEE Computer Society Press, 1998.

[35] Institute of Electrical and Electronics Engineers (IEEE). *IEEE P1363: Standard specifications for public key cryptography*, 2000.

[36] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC 11770-1: Information technology — Security techniques — Key management — Part 1: Framework*, 1996.

[37] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC 10118-3: Information technology — Security techniques — Hash Functions — Part 3: Dedicated hash functions*, 1998.

[38] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC 10118-4: Information technology — Security techniques — Hash Functions — Part 4: Hash-functions using modular arithmetic*, 1998.

[39] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC 14888-1: Information technology — Security techniques — Digital signatures with appendix — Part 1: General*, 1998.

[40] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC 14888-2: Information technology — Security techniques — Digital signatures with appendix — Part 2: Identity-based mechanisms*, 1999.

[41] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC 9797-1: Information technology — Security techniques — Message Authentication Codes (MACs) — Part 1: Mechanisms using a block cipher*, 1999.

[42] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC 10118-1: Information technology — Security techniques — Hash Functions — Part 1: General*, 2000.

[43] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC 10118-2: Information technology — Security techniques — Hash Functions — Part 2: Hash-functions using an n-bit block cipher*, 2000.

[44] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC). *ISO/IEC 9796-2: Information technology — Security techniques — Digital signature schemes giving message recovery — Part 2: Mechanisms using a hash-function*, 2002.

[45] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC). *ISO/IEC 9797-2: Information technology — Security techniques — Message Authentication Codes (MACs) — Part 2: Mechanisms using a dedicated hash-function*, 2002.

[46] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC FCD 18033-1: Information technology — Security techniques — Encryption algorithms — Part 1: General*, March 2003.

[47] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC FCD 18033-2: Information technology — Security techniques — Encryption algorithms — Part 2: Asymmetric ciphers*, March 2003.

[48] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC FCD 18033-3: Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers*, March 2003.

[49] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Geneva. *ISO/IEC FCD 18033-4: Information technology — Security techniques — Encryption algorithms — Part 4: Stream ciphers*, March 2003.

[50] ITU-T. *X.800 Data Communication Networks: Open System Interconnection (OSI); Security, structure and Applications — Security architecture for Open Systems Interconnection for CCITT Applications.* Geneva, 1991. Also published as ISO 7498-2.

[51] ITU-T. *Recommendation X.509, Information technology — Open Systems Interconnection — The Directory: Public-key and attribute certificate frameworks.* Geneva, March 2000.

[52] R. Keller, G. Zavagli, J. Hartmann, and F. Williams. Mobile electronic commerce: Research investigations into loading and payment functionality in wireless wallets, 1998. Available at http://citeseer.nj.nec.com/cs.

[53] V. Khu-smith. An implementation flaw concerning Netscape Navigator and cookies. Internal Report, Information Security Group, Royal Holloway, University of London, June 2000.

[54] V. Khu-smith. User's security risks: a comparison of SSL and SET. Internal Report, Information Security Group, Royal Holloway, University of London, June 2000.

[55] V. Khu-smith and C. J. Mitchell. Enhancing the security of cookies. In K. Kim, editor, *Proceedings of the 4th International Conference on Information Security and Cryptology – ICISC 2001, Seoul, Korea*, Lecture Notes in Computer Science 2288, pages 132–145. Springer-Verlag, December 2001.

[56] V. Khu-smith and C. J. Mitchell. Electronic transaction security: An analysis of the effectiveness of SSL and TLS. In *Proceedings of International Conference on Security and Management (SAM '02), Las Vegas, Nevada, USA*, pages 425–429. CSREA Press, June 2002.

[57] V. Khu-smith and C. J. Mitchell. Enhancing e-commerce security using GSM authentication. Technical Report RHUL-MA-2002-3, Mathematics Department, Royal Holloway, University of London, December 2002. Available at http://www.ma.rhul.ac.uk/techreports.

[58] V. Khu-smith and C. J. Mitchell. Using EMV cards to protect e-commerce transactions. In K. Bauknecht, A. Min Tjoa, and G. Quirchmayr, editors, *Proceedings of EC-Web 2002, 3rd International Conference on Electronic Commerce and Web Technologies, Aix-en-Provence, France*, Lecture Notes in Computer Science 2455, pages 388–399. Springer-Verlag, September 2002.

[59] V. Khu-smith and C. J. Mitchell. Using GSM to enhance e-commerce security. In *Proceedings of the Second ACM International Workshop on Mobile Commerce, Atlanta, Georgia, USA (WMC '02)*, pages 75–81. ACM Press, September 2002.

[60] H. Krawczyk, M. Bellare, and R. Canetti. *HMAC: Keyed-Hashing for Message Authentication — RFC 2104*. IETF, February 1997.

[61] D. Kristol and L. Montulli. *HTTP State Management Mechanism — RFC 2109*. IETF, 1997.

[62] S. Laurent. *Cookies*. McGraw-Hill, 1998.

[63] MasterCard International Incorporated. *A White Paper: Using EMV for Remote Authentication*, December 2001.

[64] G. McGraw and E. Felten. *Java Security: hostile applets, holes, and antidotes*. John Wiley and Sons, 1996.

[65] G. McGraw and E. W. Felten. *Securing Java: getting down to business with mobile code*. John Wiley and Sons, Inc., New York, 1999.

[66] A. F. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of applied cryptography*. CRC Press, 1997.

[67] C. J. Mitchell. The security of the GSM air interface protocol. Technical Report RHUL-MA-2001-3, Mathematics Department, Royal Holloway, University of London, August 2001. Available at http://www.ma.rhul.ac.uk/techreports.

[68] Netscape. *Persistent Client State HTTP Cookies*. Netscape, 1996.

[69] S. Oaks. *Java Security*. O'Reilly, 1999.

[70] D. O'Mahony, M. Peirce, and H. Tewari. *Electronic Payment System for E-Commerce*. Artech House Publishers, 2001.

[71] J. Park and R. Sandhu. Secure cookies on the web. *IEEE Internet Computing*, 4(4):36–44, July/August 2000.

[72] C. P. Pfleeger. *Security in computing.* Prentice Hall International, Inc., 1997.

[73] E. Rescorla. *SSL and TLS — Building and Designing Secure Systems.* Addison Wesley, 2000.

[74] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public key cryptosystem. *Communications of the ACM*, 21:120–126, February 1978.

[75] D. Ross, I. Brugiolo, J. Coates, and M. Roe. Cross-site scripting overview. http://www.microsoft.com/technet/security/, February 2000.

[76] SETCo. *Secure Electronic Transaction Standard — Book 1: Business Description*, 1997. Available at http://www.setco.org.

[77] SETCo. *Secure Electronic Transaction Standard — Book 2: Programmer's Guide*, 1997. Available at http://www.setco.org.

[78] SETCo. *Secure Electronic Transaction Standard — Book 3: Formal Protocol Definitions*, 1997. Available at http://www.setco.org.

[79] SETCo. *Secure Electronic Transaction Standard — Glossary*, 1997. Available at http://www.setco.org.

[80] E. Sit and K. Fu. Web cookies: Not just a privacy risk. *Communications of the ACM*, 44(9):120, September 2001.

[81] W. Stallings. *Cryptography and network security: Principles and practice.* Prentice Hall, 1999.

[82] D. Stein. *Web Security.* Addison-Wesley, Boston, 1998.

[83] D. R. Stinson. *Cryptography: Theory and practice.* CRC Press, 1995.

[84] S. G. Stubblebine, P. F. Syverson, and D. M. Goldschlag. Unlinkable serial transactions: protocols and applications. *ACM Transactions on Information and System Security*, 2(4):354–389, 1999.

[85] S. Thomas. *SSL and TLS Essentials — Securing the Web*. John Wiley and Sons, New York, 2000.

[86] E. Turban, J. Lee, D. King, and H. M. Chung. *Electronic Commerce: A managerial perspective*. Prentice Hall, 2000.

[87] J. D. Tygar and A. Whitten. WWW electronic commerce and Java trojan horses. In *Proceedings of the second USENIX Workshop on Electronic Commerce*, pages 243–250. USENIX, November 1996.

[88] K. Vedder. GSM: Security, services, and the SIM. In B. Preneel and V. Rijmen, editors, *State of the Art in Applied Cryptography*, Lecture Notes in Computer Science 1528, pages 224–240. Springer-Verlag, 1998.

[89] Visa International Service Association. *3-D Secure Protocol Specification: Extension for mobile Internet devices version 1.0.1*, November 2001.

[90] Visa International Service Association. *3-D Secure Protocol Specification: Core functions version 1.0.2*, July 2002.

[91] Visa International Service Association. *3-D Secure Protocol Specification: System overview version 1.0.2*, July 2002.

[92] M. Walker and T. Wright. Security. In F. Hillebrand, editor, *GSM and UMTS: The Creation of Global Mobile Communication*, pages 385–406. John Wiley & Sons Ltd., 2002.

[93] K. Woodsend. *Technical Specification Group Terminals; USIM Application Toolkit (USAT)*. 3rd Generation Partnership Project, June 2002.

[94] J. P. Yang and K. H. Rhee. The design and implementation of improved secure cookies based on certificate. In A. Menezes and P. Sarkar, editors,

Progress in Cryptology — INDOCRYPT 2002, Lecture Notes in Computer Science 2551, pages 314–325. Springer-Verlag, 2002.

[95] E. Z. Ye, Y. Yuan, and S. Smith. Web spoofing revisited: SSL and beyond. Technical Report TR2002-417, Department of Computer Science, Dartmouth College, February 2002.