NOTE TO USERS

Page(s) not included in the original manuscript and are unavailable from the author or university. The manuscript was scanned as received.

108-113

This reproduction is the best copy available.

UMI[®]

Secure Wireless Payment Protocol

By

Hong Wang

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of
Master of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Computer Science
Carleton University
Ottawa, Canada

May 15, 2002

© Copyright
2002, Hong Wang

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-79760-0

Canada

The undersigned hereby recommend to
the Faculty of Graduate Studies and Research
acceptance of the thesis,

Secure Wireless Payment Protocol

Submitted by

Hong Wang

In partial fulfillment of the requirements for
the degree of Master of Computer Science

Dr. Frank Dehne
(Director, School of Computer Science)

Dr. Evangelos Kranakis
(Thesis Supervisor)

Carleton University

May 15, 2002

Abstract

With the convergence of two network technologies, wireless data communication and the Internet, more and more Internet services are now being used in the wireless area. Mobile payment protocols are necessary for online transactions. A good payment protocol should balance the requirements of security and convenience.

WAP and WLAN are two prevalent wireless technologies. With the strong security demanded by banking services, WLAN is not adopted because of its deficiency in security; this was provided by WAP at the time this thesis began. WAP has failed to be widely used, because of the long latency caused by its circuit-switched network, but despite this, WAP has been embraced by the banking sector. WAP is therefore the protocol this thesis is based on.

WPP is a convenient lightweight protocol that supports both credit card and debit card transactions in wireless environment. However, security is not actually addressed. In order to make full use of the advantage of WPP and address its deficiency, we present the Secure Wireless Payment Protocol (SWPP) to address the security requirement in wireless transactions. SWPP draws upon WAP for WTLS (Wireless Transport Layer Security), WIM (Wireless Identity Module) and WMLScript to guarantee its security. Our work also provides a valuable sample of the implementation of SWPP.

Acknowledgement

I would like to express my sincere thanks to my supervisor, Professor Evangelos Kranakis, who has provided me with excellent advice and guidance in the process of my MSC program.

I am grateful to my family for their understanding and support. Special thanks go to my family and friends who are constantly there for me in all the other facets of my life.

Contents

Abstract

Acknowledgements

1. Introduction

1.1 Motivation	1
1.2 Thesis Contributions	3
1.3 Thesis Outline	5

2. Survey of Internet Payment Systems

2.1 Internet Payment Methods	7
2.1.1 Electronic Cash System	9
2.1.2 Credit Card Based System	9
2.1.3 Electronic Check System	11
2.1.4 Electronic Funds Transfer System	13
2.2 Internet Payment Protocols	14
2.2.1 SSL	14
2.2.2 SET	16
2.3 The need for a new payment Infrastructure	17

3. Survey of Mobile Payment Systems

3.1 Mobile Payment Methods	19
3.1.1 Jalda	19
3.1.2 Mint	20
3.1.3 Paybox	21
3.1.4 Conclusions	21

3.2 MeT Account Based System	22
3.2.1 System Reference Model	22
3.2.2 Shortcomings of MeT	24
3.3 Wireless Payment Protocol (WPP)	24
3.3.1 WPP Infrastructure	25
3.3.2 Comparison with SSL, SET	26
3.3.3 Criticism	28
4. Why Choose WAP	
4.1 Comparison between WLAN and WAP	29
4.1.1 WLAN	29
4.1.2 WAP	31
4.2 Why Choose WAP	33
5. Overview of Wireless Application Protocol	
5.1 Wireless Application Environment	37
5.2 Wireless Session Protocol	37
5.3 Wireless Transaction Protocol	38
5.4 Wireless Transport Layer Security	39
5.5 Wireless Datagram Protocol	40
5.6 WAP Gateway	40
6. Security Characteristics of WAP	
6.1 Wireless Identity Module	42
6.2 WMLSCrypt	43
6.3 WTLS	44

6.3.1	WTLS Implementation Classes	46
6.3.2	WTLS Handshake	47
6.3.3	Digital Certificate	49
6.4	Wireless Public Key Infrastructure	51
7.	Secure Wireless Payment Protocol	
7.1	Motivation of SWPP	53
7.2	Security Requirement	54
7.3	Security Addressed in SWPP	55
7.4	Architecture of SWPP	56
7.4.1	Description	57
7.4.2	Payment Transaction Flow	59
7.5	Secure WTLS Session	62
7.5.1	WAP Security Architecture	62
7.5.2	WTLS Handshake Process	63
7.6	End to end Security	71
7.7	Authorization by Customer	74
7.8	Exceptional Cases	75
8.	Implementation of SWPP	
8.1	Introduction	77
8.2	Developing Environment	79
8.3	Architecture	81
8.4	Data Flow	83
8.5	Program Structure	88

8.5.1	Server Authentication	88
8.5.2	Customer Authentication	89
8.5.3	Customer Authorization	91
8.6	Evaluation	93
8.7	Conclusion	97
9.	Comparison to WPP	
9.1	WPP Deficiencies	99
9.2	Security Characteristics of SWPP	101
9.3	WPP and SWPP	103
10.	Conclusions and Future Work	
10.1	Thesis Contributions	107
10.2	Conclusions	110
10.3	Future Work	110
	References	113

List of Figures

2.1 Electronic Cash	9
2.2 ECheck	10
2.3 Credit Card Payment	12
2.4 Electronic Funds Transfer	13
3.1 Reference Model of Account-based Payment	22
3.2 WPP Infrastructure and Payment Transaction Flow	26
5.1 WAP Programming Model	36
5.2 WAP Architecture	36
5.3 Different compositions of protocol stack	36
5.4 WTLS Layer	39
6.1 Full Handshake	48
6.2 Optimized Handshake	49
6.3 Comparison between X.509 and WTLS certificate	50
6.4 Major technical components and operational flow of WPKI	52
7.1 SWPP Infrastructure and Payment Transaction flow	57
7.2 Secure Payment Transaction flow	61
7.3 WAP Security Architecture	63
7.4 Full Handshake based on WIM	66
7.5 Optimized Handshake based on WIM	68
7.6 Secure Transport Session between Customer Agent and Bank Gateway	69
7.7 End-to-end Secure Transaction	72

7.8 Transport Layer end-to-end Security Architecture	73
7.9 Customer Authorization	74
7.10 Successful Transaction	75
8.1 SWPP Implementation Architecture	81
8.2 SWPP Workflow	83
8.3 Customer Authentication	89

Abbreviations

API	Application Programming Interface
CA	Certificate Authority
CAA	Customer Agent
eCash	Electronic Cash
eCheck	Electronic Check
EAP	Extensible Authentication Protocol
EE	End-entity Application
EFT	Electronic Funds Transfer
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IV	Initialization Vector
MA	Merchant Agent
MB	Merchant's Bank
MeT	Mobile electronic Transactions
PIN	Personal Identity Number
PTD	Personal Trusted Device
PKI	Public Key Infrastructure
PRF	Pseudo Random Function
RA	Registration Authority
SET	Secure Electronic Transaction
SMS	Short Message Service

SSL	Secure Socket Layer
SWPP	Secure Wireless Payment Protocol
TLS	Transport Layer Security
WAE	Wireless Application Environment
WDP	Wireless Datagram Protocol
WEP	Wired Equivalent Privacy
WIM	Wireless Identity Module
WLAN	Wireless Local Area Network
WPKI	Wireless Public Key Infrastructure
WPP	Wireless Payment Protocol
WSP	Wireless Session Protocol
WTLS	Wireless Transport Level Security
WTP	Wireless Transaction Protocol
WWW	World Wide Web

Chapter 1

Introduction

1.1 Motivation

Mobile Payment, the implementation of wireless payments, is becoming more and more important. Wireless services are increasing, along with the number of mobile users. Payment services are an indispensable prerequisite for these service offerings. Improved data transfer and the easier use of such services will also increase demand among end users. Using a mobile handset for on-line transactions is relatively more complex than using a fixed-location terminal, such as a desktop PC, for the same purpose because of the shortage of power and memory.

The mobile payment method has addressed challenges in cost, security and convenience. Security should be the utmost concern for customers. Customers would have little or no confidence in a payment method that cannot provide ways to ensure authenticity, confidentiality, integrity and non-repudiation. A recent study on wireless payment security uncovered the main challenge facing both wireless banking innovators and online banking establishments: nearly 85 percent of respondents are worried about online security transactions, including credit card purchases. More than 90 percent expressed concern about revealing personal information online, such as a credit card number. Over 90 percent of proprietary cryptography has been broken. Cell phone systems have been hacked [20].

WLAN and WAP are two different types of technology that have made an impact on today's wireless market. 802.11b has been agreed upon as the standard for WLAN. With

the ability to offer cheap, easy bandwidth, WLAN is considered to be a powerful, successful technology. One disadvantage to WLAN is its problems with regard to security. WEP, which is known as an encryption scheme in 802.11 that protects data packets, has some fundamental flaws. The weakness within WEP makes WLAN easily attackable. Although this problem had been solved by the end of 2001, it was still a problem when this thesis began.

WAP is another technology which is devoted to making full use of the Internet services already in existence. Having taken the limitations of the wireless environment into account, WAP is widely accepted as a de-facto standard. However, WAP has failed to some degree because of the latency resulting from the circuit-switched network. It's difficult to connect to WAP sites and complete data transactions. With many network operators upgrading their network to GPRS, the speed problem can be solved. Different security level solutions are provided by WAP according to the business requirements. Security problems caused by the WAP Gateway have been fairly well addressed by WAP 2.0, which will be put into practical use in the near future. One important reason we chose to use WAP here is political: WAP is embraced by the banking sector. According to Roy Smith, managing director at Brokat: "Banks are buying WAP servers despite not knowing what WAP is or how it will help their organization." [74]

Nowadays, there is no agreement regarding how mobile payment will work — particularly when the large number of different channels and payment methods that currently exist are taken into account. Numerous bodies are trying to work out a wireless payment standard which takes into account factors such as speed, convenience and security. Based on an analysis of the account-based payment architecture proposed by

MeT, the convenience issue is not well addressed at present as it embraces SET technology in the wireless environment. Another wireless payment protocol, WPP, is more convenient than MeT, as messages do not go through any intermediary. WPP achieves higher performance by reducing the number of messages, thereby lowering bandwidth and computation requirements. Lowering the processing time results in reduced communication costs. [22] Actually, WPP is not well designed and implemented, as security in the banking infrastructure is assumed to be sufficiently guaranteed by WTLS.

The main objective of this thesis is to establish a framework for a Secure Wireless Payment Protocol, based on the WAP framework. This solution will address both security and convenience, making the protocol practical to be used.

1.2 Thesis Contributions

The main contribution of this thesis is the design and implementation of a Secure Wireless Payment Protocol (SWPP). SWPP starts when the merchant sends an invoice to a customer and ends when the merchant receives a confirmation of payment from his or her bank. The message flow is similar to that in WPP. Details on how to build a secure SWPP between the end user and the service provider through a WAP gateway are given. To protect against network attacks, messages are encrypted using symmetric cryptography. The only way to break a symmetric cryptosystem is to use a brute-force attack, which involves trying all the possible keys that can be used for encryption. A 56-bit DES algorithm can be considered secure as there are 2^{56} possible permutations. It would take a supercomputer with the ability to test a million keys per second 2,285 years to try all the keys of a 56-bit key solution. However, there is a key distribution problem in

symmetric cryptosystems. Asymmetric cryptosystems, which are usually very computationally expensive and vulnerable to chosen plain text attacks, are used for digital authentication and to establish and distribute secret keys for symmetric cryptography. Symmetric and asymmetric cryptosystems are used in SWPP to prevent network attacks.

In this solution, SWPP relies upon a secure environment provided by the WAP specification. Confidentiality and integrity can be addressed through the use of Wireless Transport Layer Security (WTLS), while customer and server authentication are achieved by combining WTLS with the Wireless Identity Module (WIM). The WIM will also facilitate the use of digital signatures, which will help ensure non-repudiation. End-to-end security is achieved by combining TLS/SSL and WTLS. SWPP uses proxy technology, in conjunction with firewall technology, to define the boundary for the service domain. By owning the gateway in its private service network, the security problem caused by WAP is addressed. To provide a mechanism for the end user to navigate between different gateways, there are two kinds of gateway: the master WAP gateway and the subordinate WAP gateway. The master WAP gateway supplies the navigation documents while the subordinate WAP gateway supplies WAP services.

In order to implement this solution, a detailed SWPP design is given. Tools from Nokia company were used to simulate the payment process. Nokia Activ Server, with the support of Nokia Activ Security, simulates the Bank Gateway and Bank Service, so that end-to-end security is provided. The Nokia Toolkit simulates the end user. From the simulation, we found that SWPP can be a practical protocol that actually addresses both security and convenience. The performance varies when different algorithms are chosen.

1.3 Thesis Outline

This thesis describes an experimental implementation of the Secure Wireless Payment Protocol on a WAP terminal. It is organized as follows.

Chapter 2 provides a survey of Internet Payment Protocols and methods. From the survey, we know that both SSL and SET have shortcomings when used in a credit card-based system. SSL is less secure while SET is less convenient.

Chapter 3 provides an overview of mobile payment methods. There are several methods which provide the requirements for a mobile payment standard. MeT is one forum which is working out the standard. The system model of the MeT account-based payment system actually embraces the SET technology in the Internet. By changing data flow, the WPP protocol tries to embrace the SSL technology to achieve the same security requirement provided by SET. The pros and cons of MeT and WPP are discussed in this chapter.

Chapter 4 demonstrates why WAP is chosen instead of WLAN. Because WLAN's security issues had not been addressed when this thesis began, WAP was chosen instead. Although WAP has failed to live up to its expectations in the past because of the long latency brought about by its circuit-switched network, WAP has a big market as many network providers are updating to GPRS.

Chapter 5 gives an overview of the Wireless Application Protocol (WAP). WAP sounds like a simple version of the OSI standard. WAP is trying to address the deficiencies of the wireless environment and make full use of Internet services.

Chapter 6 provides an analysis of WAP security. WIM, WMLScript, WTLS and WPKI are the four components that apply security at the application, transport and management levels in the wireless environment.

Chapter 7 presents a detailed design of the Secure Wireless Payment Protocol. Security is achieved by encompassing WAP technology. This chapter includes details on how to build a WTLS session channel between Customer and Customer Gateway, how to build end-to-end security and how to process Customer Authorizations.

Chapter 8 shows the SWPP implementation through the use of Nokia's newest tools.

Chapter 9 gives a comparison between WPP and SWPP.

Chapter 10 concludes the thesis and presents a discussion of future research issues.

Chapter 2

Survey of Internet Payment Systems

2.1 Internet Payment Methods

In today's world, electronic messages moving through private networks have replaced paper for most of the transactions between banks on a daily basis. With the arrival of the Internet as a mass data network, technologies and business models are being developed to facilitate electronic transfers by ordinary customers. Electronic payments are generally representative of real-world exchanges. The Internet payment systems currently in use or under trial fall into two categories: account-based and token-based.

- **Account-based**

In account-based payment methods, every customer is associated with a specific account maintained by an Internet Payment Provider. In this type of system, a customer instructs his or her bank to debit or credit his or her account upon receiving the payment. Debits and credits are exchanged during a transaction. If pre-paid transactions are supported, this account can be linked directly to the customer's savings account, for example. The customer maintains a positive balance in this account that is the balance of the account to the Internet Payment Provider. Due to high administrative charges, traditional account-based payment methods are generally not suitable for handling transactions of very low value.

Few account-based solutions in use today have sufficient security to prevent fraud. In the Electronic Funds Transfer system, the encryption is done through a Secure Socket Layer

(SSL), which is convenient but not secure enough. Under this condition the use of SSL can be considered as sufficient since the customer's confidential data does not pass through any intermediary. The situation is different with credit card payment systems, where the data will pass through the merchant from the customer to the merchant bank. In Credit card payment systems, the credit card number and expiration date entered by the customer are sent to the online merchant in an encrypted format. Since digital goods can be downloaded almost immediately after purchase, fraudulent credit card use is more likely to happen over the Internet. To deal with security issues including fraud, SET, developed by Visa and Mastercard, is considered as an open technical standard for the commerce industry to facilitate secure payment card transactions over the Internet. However, SET has not been extensively implemented due to its high cost, although major banks and credit card companies support it.

Existing account-based electronic payment systems on the Internet are classified into electronic check systems, electronic fund transfer systems and credit card-based systems. Details of these systems are presented below.

- **Token-based**

Different from account-based systems, funds in token-based systems are not transferred from the buyer to the merchant but remain with the issuer until claimed by the merchant. A token here is defined as a medium of exchange representing some monetary value usually backed by a bank. In a token-based system, customers purchase electronic tokens from an issuer, who may be a bank or a non-bank. Electronic tokens serve as cash substitutes for transactions over the Internet. The value is then stored in digital form on a consumer's PC and the notational value is transferred over the Internet. The "coin" is

merely a notational series of numbers or other symbols that are transmitted over the Internet to a merchant. The merchant must then redeem the "coin" with an issuer – who will verify that the coin has not been spent previously. The issuer of the Internet "money" or "scrip" is obligated to redeem these payments when received from the merchant.

2.1.1 Electronic Cash System

Electronic Cash (eCash) is a token-based system. In the eCash system, the payer withdraws money from the issuer, and hands the payment tokens over to the payee, who in turn deposits the payment with its acquirer. The acquirer then asks the issuer for a settlement. The eCash system is suitable for small transactions called “micropayments”. As it has to keep track of a very large database of users and used coins, it is not suitable for large payments. Such payment systems usually involve either tamper-proof hardware like smart cards, or online validation by the issuer to protect against double spending of digital cash. [52] The data flow is shown in Fig 2-1 below.

Fig 2-1 Electronic Cash

2.1.2 Electronic Check Systems

An Electronic Check System (eCheck) is a type of account-based system. The eCheck is an electronic version of a paper check, except that it is initiated electronically. Using security techniques including authentication, public key cryptography, digital signatures, certificate authorities, duplicate detection and encryption, eCheck is the most secure payment instrument ever designed or developed. ECheck further enhances banking practices with added security so that even breaking the cryptographic protections would not necessarily allow a fraudulent transaction to be paid. ECheck works the same way a check does. [12]

1. The check writer "writes" the eCheck using one of many types of electronic devices and "gives" the eCheck to the payee electronically.
2. The payee "deposits" the eCheck, receives credit, and the payee's bank "clears" the eCheck to the paying bank.
3. The paying bank validates the eCheck and then "charges" the check writer's account for the check.

Figure 2-2 below shows how a payee can endorse and cash the eCheck at the payer's bank.

Fig 2-2 ECheck

(A certified check from the payer's bank to the payee through the payer can be sent directly from the payer to the payee)

The process of using checks or notes is very complex, while there are too many related laws and regulations for a general user to use a check or a note. It will not be addressed in detail here.

2.1.3 Credit Card Based System

The credit card system was designed to provide immediate gratification to customers by allowing them to purchase goods or services on credit. Once a merchant has had a purchase authorized by the card issuer over the private authorization network, the merchant is assured of payment and the card issuer assumes responsibility for billing the customer and collecting the money. In the credit card payment system, the payment amount range is very narrow. While the high transaction cost of a credit card makes it unsuitable for a lower bound, the effort to reduce risk to a credit card organization is the reason for the upper bound.

Since a credit card payment requires only the information on the face of the card, such as the credit card number, expiration date, and name on the card, the security of the transaction is very important, and is achieved by SET protocol. The purpose of the SET protocol is to use cryptography to (1) retain confidentiality of information, (2) ensure payment integrity, and (3) authenticate both merchants and cardholders. Fig 2-2 shows the typical processing flow for a credit card:

Fig 2-3 Credit Card Payment

1. The customer sends the payment information to the merchant server. The merchant agent will send and receive messages; encrypt and decrypt; store public and private keys; and request and receive certificates.
2. The merchant's software takes the payment information from the cardholder and sends it to the merchant's bank. The merchant's bank must receive and authenticate the payment information that the merchant received from the cardholder.
3. The merchant's bank sends an authorization request to the customer's bank over the interbank network. The customer's bank sends the authorization response to the acquirer.
4. The merchant's bank notifies the merchant about the status of the authorization; if the response is positive, the merchant fulfills the order.
5. The merchant presents the charge to his bank.
6. The merchant's bank sends a settlement request to the customer's bank.

7. The customer's bank charges the customer's credit card account and at regular intervals notifies customer of the transactions and accumulated charges.
8. The customer pays the charges to his bank. The merchant's bank credits the merchant's account.

2.1.4 Electronic Funds Transfer (EFT) Systems

Security First Network Bank (SFNB) is the first bank which opened on the Internet and provides the conventional bank's full service. As a banking service, SFNB provides EFT using the WWW. If a payer inputs a payment amount and a payee on the web, then the amount is debited to the payer's account and the payee can receive the amount. If a payee does not accept an electronic payment, SFNB writes a check from the payer's check book and mails it to the payee. [56]

Fig 2-4 Electronic Funds Transfer

Nowadays, we can transfer funds using ATMs, public networks, or packet networks. A paper receipt can be obtained when using ATM, but there is no receipt issued when

transferring funds over a network. The payer takes a risk by trusting the data in the bank's database.

In this system, the customer interacts directly with his or her bank in a secure manner, using protocols such as the SSL. SSL is sufficient for security since the customer's confidential data does not pass through any intermediary.

2.2 Internet Payment Protocols

The success of e-commerce lies with the protocols involved in exchanging money for goods and services. Internet Payment protocols define the specific message format and information flow for transferring information between parties in an e-commerce transaction. The payment protocols should balance security, convenience, performance, and cost. Web interactive and store-and-forward e-mails are the two major modes of interaction defined by the Internet. Credit and debit card payments are usually done interactively on the Web while e-checks are used with store-and-forward e-mail. The most commonly used payment protocols for secure online payment transactions are SSL and SET.

2.2.1 SSL

SSL is the standard for secure client-server communication on the Web. SSL protocol is a set of rules governing server authentication, client authentication, and encrypted communication between servers and clients. The two main advantages of SSL are simplicity and its pervasive implementation in browsers and servers. Because of its convenience and security, SSL is currently the most widely used method for performing secure transactions on the Web and is supported by most web servers and customers, including Netscape Navigator and Microsoft Internet Explorer.

On the Internet, SSL can be used at the front end between the customer's browser and the merchant or at the back end, between the merchant and the gateway. [40] Although SSL provides an encrypted transmission of data between the browser and the Web server, it has some limitations when used in the credit card system.

One limitation of SSL is that Merchants cannot later prove that a particular user authorized a transaction because SSL does not provide any provision for Customer Authorization. SSL secures the relationship between only two of the three participants (customer, merchants, and banks), such as customer-to-merchant or merchant-to-bank.

The other limitation of SSL is that it is limited in its use of certificates. In SSL, all the information sent between the customer and server is over an SSL protected channel. Server authentication is a must while customer authentication is an option.

Without customer authentication and customer non-repudiation, the customer may deny having made the payment and the merchant may not be able to prove that the transaction took place, even if it was legitimate. This causes a "charge-back" cost for honest merchants due to dishonest customers. Using the SSL protocol, dishonest merchants can also make illegal money. In order to initiate the authorization process after receiving the payment order from the customer, the merchant has to see the customer's account information, which results in the disclosure of sensitive account information. Furthermore, since the customer's bank has no way of checking the customer's intent to make the payment, a dishonest merchant is also able to use this account information later to make charges without the consent of customer. Although the customer can rightfully repudiate this bogus transaction, a dishonest merchant might benefit from a transaction being overlooked. To prevent credit card fraud, we have to use the SET protocol instead.

2.2.2 SET

The SET Specification is an open technical standard for the commerce industry to facilitate secure card payment transactions over the Internet. It was developed as a joint venture between Visa, Mastercard and others in an attempt to solve many security issues, such as fraud, by handling the entire purchase process. Different from the SSL protocol, which is suitable for peer to peer communication, the SET protocol defines the processes involved in credit card payment including purchase, payment authorization, and payment capture. [40] The shortcomings of SSL are addressed in SET.

The SET protocol relies on public key encryption that uses financial institution, merchant, and cardholder digital certificates. In the SET protocol, the cardholder must have a digital certificate issued by a trusted CA. To ensure the legitimacy of a cardholder's identity, the cardholder's public key must be certified via a digital certificate. A trust chain can therefore be created throughout the transaction using Digital Certificates to verify cardholder and merchant validity. The identities of all parties are verified before a transaction takes place.

To ensure the data is secure, the credit card number is encrypted separately from the order information when the order information is sent from the consumer to the merchant. In this way, only the information that is required by each party is actually sent to them. The order information can just be read by the merchant and the credit card information can just be read by the bank. A dual signature is required to secure a credit card number from merchant.

With Digital Certificates and Customer Authentication, both cardholders and merchants can feel safe when transferring payment information online. Unfortunately, this security

feature is obtained by sacrificing “convenience”. Financial Institutions are less than eager to deploy SET, for the reasons listed below.

- SET requires that banks register their customers. They need to have certificates in order to use the protocol. However, SSL-based solutions do not require such registration. [31]
- SET requires a PKI (Public Key Infrastructure), a complete system for certificates. The banks, the merchants and the end users come together in a hierarchical manner in this PKI. The certificates are issued by dependent CAs.
- SET is only for payment card (credit or debit)-based transactions. Other transactions, like electronic checks (e-checks), are not included in SET.

2.3 The Need for a New Payment Infrastructure

According to the analysis above, we know that convenience is sometimes more important than security in the end user’s eyes. Both the SSL and SET protocols have their own shortcomings when used in a credit card based system. The SSL protocol is convenient but lacks security. The SET protocol is secure but is short on convenience, making it not widely accepted.

Although security of an e-payment method is very important for all parties involved in a transaction, security alone does not guarantee success in the marketplace. An e-payment system must also be convenient. This requirement has different meanings for different parties. From the consumer’s point of view, “convenience” means to pay quickly and without an additional cost or too much effort. From the bank’s point of view, “convenience” means low deployment and operational cost. A good payment system should be both “secure” and “convenient”. [31]

On the other hand, SSL and SET, based on TCP/IP, do not address the limited resources of the wireless environment. A new payment protocol suitable for the wireless environment is required to make online banking services accessible to portable devices. Both convenience and security need to be addressed.

Chapter 3 will give an overview of mobile payments, and different payment methods will be discussed in section 3.1. A review on the MeT account-based payment is given in section 3.2. MeT is considered to be a payment standard. Another payment protocol, Wireless Payment Protocol (WPP), will be addressed in section 3.3. Both system models and their shortcomings will be analyzed in detail.

Chapter 3

Survey of Mobile Payment Systems

Although using mobile phones as a means of payment has been suggested a number of times in the past, to date there have been no agreements as to how this would work — particularly when the large number of different channels and payment methods currently in existence are taken into account. [38] Various initiatives and consortia are currently at work to help meet the challenges involved with working out a wireless payment standard, which include speed, convenience, and security. MeT (Mobile Electronic Transactions) is one of these, aimed at creating a trusted personal device through the integration of security and transaction applications into a mobile terminal platform.

3.1 Mobile Payment Methods

Mobile payment has already been used in some areas, including Europe and Asia. In North America, a series of trials were scheduled for late 2001. [53] Below is a list of some prevalent mobile payment methods that are in use nowadays.

3.1.1 Jalda

Jalda is developed by EHPT, a joint venture between Ericsson and Hewlett-Packard. [8] It is an account-based payment method consisting of open Application Programming Interfaces (APIs) and implements security based on RSA PKI encryption. The security mechanisms used in Jalda are GSM encryption, digital signature (PKCS#7), SSL, PIN, WPKI, and WIM.

The message flow in Jalda can be described as follows:

1. The user chooses to buy goods from an ISP.

2. The ISP receives an order.
3. The ISP sends a payment request to the payment provider using Jaldä.
4. The payment provider sends a digital contract as an SMS with a password request via the ISP to the user's phone.
5. The user accepts and confirms the payment by entering a password, which is sent via SMS to the mobile e-pay server.
6. The mobile e-pay server validates the password, generates a digital signature and signs the contract.
7. The payment server verifies the signature and sends an ok after approval to the ISP.
8. The payment server sends a receipt to the user's phone.

3.1.2 Mint

Mint was developed by Ericsson and has been in commercial operation in Stockholm since January 2001. It is an SMS-based system. The security mechanisms used in Mint are GSM encryption, secure links between terminals and the Mint server and a PIN.

The message flow in Mint can be described as follows:

1. The user selects goods at a "physical" shop.
2. The merchant enters the amount of the purchase into the Mint payment terminal.
3. The Mint terminal displays a telephone number (terminal specific number).
4. The user dials this number.
5. If the amount exceeds a predefined limit, the user has to enter a PIN; otherwise the user just accepts the transaction.

6. The merchant obtains a “payment accepted” or “payment failed” message on the payment terminal.
7. The user receives a receipt via SMS and / or e-mail.
8. Mint settles the accounts.

3.1.3 Paybox

Paybox is a payment system that uses the mobile phone as an authorization and confirmation device. The security mechanisms used in Paybox are GSM encryption, PIN and SSL for registration.

The message flow in Paybox can be described as follows:

1. The user chooses goods in a ‘real’ shop.
2. The user gives the merchant his or her Paybox account number (not the telephone number).
3. The merchant calls a special Paybox number and sends the amount and the user’s Paybox account number to Paybox.
4. Paybox calls the user and repeats the amount (here the user is called with “his own voice”).
5. The user inserts his or her paybox PIN to authorize the payment.
6. Paybox settles the accounts.

3.1.4 Conclusions

From the overview of the different mobile payment methods, we know that mobile e-commerce has to overcome a number of security issues. WAP provides the required security mechanisms to allow secure transactions, which form the basic requirement for mobile payments. There is still no standard for online account-based payments.

3.2 MeT Account Based Payment

The challenge in mobile payment lies mainly in the issue of security. Although there are many mobile payment methods currently in use, there exists no standard for online account-based payments. Here we will introduce an account-based payment standard proposed by MeT.

MeT, an initiative by Nokia, Motorola, and Ericsson, seeks to establish a framework to secure transaction in mcommerce. In MeT, confidentiality and integrity will be addressed by WTLS, while the WIM will ensure client and server authentication. The WIM will also facilitate the use of digital signatures, which will help ensure non-repudiation. The MeT account-based system can be considered as a standard for credit card based systems.

3.2.1 System Reference Model

The usage model includes the following components: user, PTD, merchant, acquirer and account issuer.

Fig 3-1 Reference model for account-based payments [37]

The **Acquirer** is an acquiring bank: a bank with a merchant relationship and which is responsible for capturing the transaction.

The **Issuer** is an issuing bank, i.e. the bank which has a business relationship with the Cardholder

A **PTD (Personal Trusted Device)** is used for making mobile electronic transactions. It contains a Security Element that is used to store cryptographic keys and perform operations using these keys.

A **Merchant** can be a physical retail store or a Web shop.

A detailed description of the message flow is given below.

1. The user activates the PTD and establishes a secure WTLS link to the merchant server.
2. A shopping phase (not specified) and a payment contract are prepared by the merchant.

(The payment contract contains the payment amount, merchant's description, date, time and transaction ID).
3. User authorization
4. Upon receipt of the signed payment contract, the merchant sends an authorization request message to the acquirer, using a backbone network not defined in MeT.
5. The acquirer forwards the signed payment contract to the issuer to transfer funds.
6. The issuer identifies the user's account from the certificate and, if the funds are sufficient, authorizes the funds transfer.
7. The issuer returns a response to the acquirer and the acquirer sends a response to the merchant. Based on the contents of the authorization response, the merchant sends the user a receipt message.

Transaction security will be increased as payments are agreed based solely on the user's digital signature, complemented by a secret PIN. The level of convenience to the user is increased as a single set of key pairs (typically one pair is used for the authentication and another pair for the signature) can be used for multiple accounts (through multiple certificates applied to the same set of keys). [37]

3.2.2 MeT's shortcomings

MeT Account Payment is a core usage model employed in online transactions. MeT is quite similar to SET, except that it is based on the WAP architecture. Security is based on a secure link between a user's PTD and a WAP content server, using WTLS for the transport layer security and WMLScript signText for the application layer user authorization. The data flow in MeT is the same as that in SET, using the merchant to make contact with the acquirer. As mentioned in Chapter 2, this is not very convenient as a dual signature is required for messages between the customer and the acquirer, passing through the Merchant.

3.3 Wireless Payment Protocol (WPP)

The Wireless Payment Protocol (WPP) is a standard payment protocol that supports both credit and debit card payments over wireless networks in a secure and efficient manner.

The roles of the six entities included in WPP can be summarized as follows:

1. *Customer Agent (CAA):*

The CAA is a WAP browser for viewing WML pages and executing WML Scripts. It is used as an interface between the WAP scripting language and the Smart Card (SC).

2. *Merchant Agent (MA):*

The MA represents a web-based application executing on the merchant's server. It can provide online web services to the customer.

3. *Customer's Bank (CB):*

The CB is responsible for issuing encrypted banking profiles of customer. It can process requests from the customer, withdraw money and transfer it to the Merchant's Bank.

4. *Merchant's Bank (MB):*

The MB is responsible for issuing the merchant's encrypted banking profiles. It can process requests from the customer's bank and deposit the money to the merchant's account.

5. *Customer's Smart Card:*

Provides a static storage for the encrypted customer's bank information

6. *Merchant's Smart Card:*

Provides a static storage for encrypted Merchant Bank's Information

3.3.1 WPP Infrastructure

A detailed description of the WPP payment transaction flow is presented in Fig 3-2.

The first step in WPP is when the Customer Agent confirms the invoice from the Merchant. Like the SET specifications, the interaction between banks in the settlement network is not part of WPP. Security in the banking infrastructure is considered to be sufficient. The web services supplied by the merchant's and the customer's bank are made available to customers through a WAP Gateway that converts html to a WML page.

The payment transaction flow can be explained as follows:

Fig 3-2 WPP Infrastructure and Payment Transaction Flow

1. The Merchant Agent prepares an invoice and sends it to the Customer Agent. (As in the SET protocol, this process is not specified in the WPP protocol. The WPP protocol only details what happens once the customer's software responds to this invoice.)
2. The customer confirms the accuracy of the invoice and the Customer Agent prepares a payment request to be sent to the customer's bank.
3. The customer's bank fulfills the request for payment by the customer and forwards a response to the merchant's bank.
4. The merchant's bank forwards a payment confirmation response to the Merchant Agent and the customer's bank forwards a notification of payment to the Customer Agent.

3.3.2 Comparison with SSL, SET

As indicated in Fig 3-2, WPP is quite different from MeT, which embraces the SET standard. WPP uses a Customer Agent to interact directly with the Customer's bank, using a simple protocol such as SSL, in which way it addressed the requirement of

“convenience”. The WPP protocol tries to combine the advantages of SSL with that of SET. The comparison can be summarized as follows:

- It is a well-known fact that credit card fraud is a serious problem on the Internet. Although the SSL protocol is widely accepted by service providers because of its simplicity and convenience, most users of the on-line payment program are afraid to give their credit card's Personal Identification Number (PIN) to anyone else without an end-to-end security guarantee.

By altering the direction of the transaction flow in WPP, the credit card information can only be given to the customer's bank. In this way, WPP eliminates the source of fraud since the customer's confidential data does not pass through an intermediary. WPP is convenient, with the addition of a strong security element.

- Unlike the SET protocol, a dual signature is not required in WPP. SET's dual signature is a digest of the concatenation of the order information and payment information message digests, encrypted with the customer's private key. This allows both the merchant and the merchant's bank to independently verify that the customer's payment instructions and order information belong together, without requiring either processor to see the other's data.

In WPP, a dual signature is not required since customer's payment instructions are sent directly to the Customer's Bank. The merchant's banking information (previously encrypted by the bank) is sent to the customer and then forwarded to the customer's bank.

- WPP makes use of Smart Cards to store encrypted banking information. It can also store Personal Identification Numbers (PINs) so that credit card payments can incorporate other types of payments, including debit card payments. Because PINs are used strictly for authorizing access to the Smart Cards and are not transmitted over the network, the security of the PIN is preserved.
- End to end security is guaranteed through a Gateway as a bridge between the SSL and WTLS protocols.

3.3.3 Criticism

From the analysis above, WPP protocol has the security of SET and the convenience of SSL. Compared with MeT, WPP is more convenient and provides the same level of security. The transaction flow defined in MeT is from the customer to the merchant, then to the bank. In WPP, the transaction flow is from the customer to the bank, then to the merchant. WPP can be considered to be a good payment system as it has been designed to balance the requirements of security and convenience.

However, in the WPP protocol security is actually not addressed. It is assumed that the WTLS protocol will provide the confidentiality and integrity of all messages exchanged between all participants of the protocol. WPP must be redesigned to fulfill its security requirements.

Both MeT and WPP are based on the Wireless Application Protocol. A comparison between WAP and WLAN will be given in Section 4.1 of Chapter 4. Section 4.2 will contain a detailed examination of why WAP was chosen.

Chapter 4

Why Choose WAP?

In this chapter, we will present two different technology types that made an impact on the wireless market in 2001: short-range wireless LANs and wide area wireless networks delivered via WAP, emphasizing the security issues involved in both. In addition, reasons will be given as to why WAP was chosen instead of WLAN for this thesis.

4.1 Comparison between WLAN and WAP

4.1.1 WLAN

WLANs are already powering successful businesses like UPS, Federal Express, and GM. These companies are using wireless LANs to offer worldwide customer relationship management solutions that allow customers and employees access to real-time information on the status of supplies. What is more interesting is the prediction that, in the future, WLANs will begin to be used outside of the office. WLANs may be used more for remote access for mobile workers via "public hotspots" in places like airports and hotels. Earlier this year, Gartner predicted that by 2006 there will be more than 38,000 wireless public LANs in North America alone. [18] This means that WLANs are well positioned to be a disruptive technology, and that is compelling

WLAN's security flaws

The most popular WLAN deployed is the 802.11b network. The access points are widely available and the 802.11b WLAN NIC cards which fit into laptops are reasonably priced.

However, despite their low price and easy installation, these networks have two critical security flaws: poor data protection and authentication mechanisms.

The encryption scheme in 802.11 wireless LANs that protects data packets is known as the Wired Equivalent Privacy, or WEP. However, due to some fundamental security flaws and the fact that most enterprises do not turn WEP on, it might be more infamously remembered as the "Why Encrypt Packets" protocol. [19] There are two basic problems in WEP that expose its vulnerabilities: the limitations of the initialization vector (IV) and the use of static WEP keys resulting in a high likelihood for collisions. IV collisions produce so-called "weak" WEP keys when the same IV is used with the same WEP key on more than one data frame. WEP can be attacked to expose the shared secret when a number of these weak keys can be analyzed.

Network security actually depends on the authentication system it is based on. WEP allows client authentication to take place via a password-based system, but leaves the system vulnerable to attacks.

Problems addressed

A lot of work had already been done to solve 802.11's security problems by the end of 2001. RSA Security, Microsoft, and Cisco offered a proposal for the Protected Extensible Authentication Protocol (Protected EAP) to the IETF, which outlines how to achieve authentication in a roaming WLAN environment. [18] This proposal was also accepted by the 802.1x committee so that it can be standardized in future WLAN products.

The second problem that needs to be tackled is the WEP encryption standard upon which all 802.11 networks are based. Unless that standard is fixed, there will be no privacy in the network. There is more good news here. Late in December, RSA Security and Hifn

announced that the IEEE committee had approved their fix for the broken WEP standard. This allows all vendors of existing 802.11b access points to write a software patch that can be applied to update WEP security in all the products already being used by end customers.

The third problem will be the transition from the 802.11b network to the higher bandwidth 802.11a network. In September 2001, Gartner predicted that WLAN chipset shipments overall will grow at a compound annual growth rate of 52% through 2005, thereby driving down prices and making WLAN technology competitive in the marketplace. [18]

4.1.2 WAP

Outside of Japan, WAP has failed to live up to its expectations because of long latency caused by the circuit-switched network upon which it is based. It has been impossible for WAP to live up to the hype generated around it in 1999, when one projection asserted that as many as 45 million people would soon be using the technology in Western Europe alone by 2002. While 50 million WAP-enabled phones have hit the market to date, only 18 million users are WAP subscribers. And, according to other reports, only a few of those subscribers actually make use of WAP's capability to access the Internet. A Meta Group poll conducted in June 2001 found that 90 to 95% of corporate users do not even use the data functions on their WAP phones. [18]

WAP inconvenience

The main problem of WAP is its inconvenience to end users. The latency resulting from the circuit-switched network makes it difficult for a WAP-enabled browser to connect to WAP sites and complete data transactions. Another problem lies in the fact that many

screens are required to complete a single transaction, such as buying stocks or booking a hotel room. Describing the inefficiencies of WAP, Jack Gold of the Meta Group claimed, "The amount of information was minuscule compared to the pain of getting it." His statement was based on reports of WAP users going through as many as 6 screens -- each screen with a latency of 45 seconds to 2 minutes -- to complete one stock transaction, or 37 clicks to make a hotel reservation. [18]

WAP insecurity

To address the problem of how to bring content to constrained devices with limited RAM running on slow and unreliable circuit-switched networks, WAP 1.x created a security gap at the gateway. During the translation at the gateway, it sits in plain text for a period of time. Although the amount of time this data is unencrypted is minimal, it is a high risk for organizations with strict security needs to bear. The gateway serves not only as a point of translation but a vulnerable point in the network -- a sitting target waiting to be attacked.

Problems Addressed

It is the latency problem caused by the circuit-switched network that has crippled WAP's success. However, this can be settled as many operators are now focusing on upgrading their networks to 2.5G GPRS, which will allow new WAP-enabled GPRS phones to work in a packet-switched, always-on environment. After upgrading the network, users can access more data faster and applications can send real-time information to phones.

There are many solutions to the "WAP gap". Actually the implementation depends on the agreement between customer (banks and enterprise) and operator. The real security challenge is a political one. Would a bank or enterprise deploy applications that collect

high-value transactions if those applications would have to run through the operator's gateway? Would companies be satisfied using a WAP gateway which resides at the operator's network? How much would it cost for a customer to deploy its own WAP gateway or lease bandwidth? If you launched your own gateway, how would customers find your WAP application? All things considered, the cost, complexity, and lack of trust more than canceled out the value of launching such applications. [18]

The new WAP 2.0 standard that was written and approved in August 2001 changes all of this. It has come up with a way for WAP 2.0-enabled phones to interoperate more securely. The WTLS security standard previously used in the WAP security layer has been replaced by Internet-standard TLS for secure end-to-end data transport. Although this is a big change and will take some time to roll out in practice, it will provide the shift in direction that will ultimately make WAP phones more interoperable with the Internet than ever before.

4.2 Why Choose WAP?

Roy Smith, Managing Director at Brokat, sees the reason why WAP is the chosen protocol for banking services as follows: "Banks are buying WAP servers despite not knowing what WAP is or how it will help their organization." He adds that banks do not want to make the same mistakes as in the past with the Internet – being in the game too little and too late. [74]

On the other hand, WLAN was not considered secure enough for banking services when I began this thesis. WAP is not a bad choice based on the security requirements and considerations. The WAP situation will change to a great extent based on the GPRS network. As WAP 2.0 had been approved in August 2001 but still not put into practice,

we chose WAP 1.2.1, the June 2000 conformance release, which has been widely implemented.

Chapter 5 will contain an overview of Wireless Application Protocol (WAP), the protocol on which this thesis is based.

Chapter 5

Overview of Wireless Application Protocol

In the wireless environment, wireless units have limited power and they tend to have less bandwidth, more latency, and less connection stability. Because handheld devices are nowhere near as powerful as computers, ordinary web technology such as TCP/IP or HTTP is not suitable. WAP is designed to address the constraints of the wireless environment, and adapt existing Internet technology to meet these constraints.

WAP is considered as the de-facto world standard developed by the WAP Forum, a group founded by Nokia, Ericsson, Phone.com and Motorola, with the aim of establishing a common format for Internet transfers to mobile telephones, without having to customize Internet pages for the particular display of every different mobile telephone or personal organizer. The key elements of the WAP specification were the WAP Programming Model (Fig 5-1) and the lightweight version of the TCP/IP protocol stack streamlined to minimize bandwidth requirements. WAP is the name of a whole suite of protocols which cover basically all layers of the ISO-OSI reference model. The WAP layers are pictured in Fig 5-2.

The WAP stack is basically divided into five layers including WAE, WSP, WTP, WTLS and WDP. We can take any subset of WAP layers and use them in an already existing framework (Fig 5-3). The end user can choose to use one of the protocol stacks below to build a connection with its gateway.

Fig 5-1 WAP Programming Model

Fig 5-2 WAP Architecture

Fig 5-3 Different compositions of protocol stacks

5.1 Wireless Application Environment

WAP is designed to resemble the World Wide Web and relies on its URL to get the resource it requires. WAE holds the tools used by wireless Internet content developers.

- **Wireless Markup Language** — This language is used for writing WAP browser pages.
- **WMLScript** — A scripting language very similar to JavaScript
- **Wireless Binary XML** — Specifies how WML-decks are compiled into a binary compact representation.
- **Wireless Telephony Application** — Describes a way to get hold of some phone specific functions, such as making calls or editing the phone book.

The main objectives of WAE are to:

- Minimize data sent over the air
- Minimize the computational energy required by the customer to process the data.

5.2 Wireless Session Protocol

The Wireless Session Protocol (WSP), designed to implement a request-response protocol much like the Hypertext Transfer Protocol (HTTP), is the session layer of the WAP suite. Using this protocol, the customer makes a request and the server answers with a reply.

WSP determines whether a session between the device and the network will be connection-oriented or connectionless. This means that WSP decides whether or not the device needs to talk back and forth with the network during a session. A connection-oriented session means that the packet will be sent from WSP to WTP, then the packet will be passed both ways between the device and the network. The session can be

suspended and resumed later. If the session is connectionless, the WSP will redirect the packet to the WDP, not the WTP. A connectionless session is commonly used when information is being broadcast or streamed from the network to the device.

Despite the difference between connection-oriented and connectionless, both services are designed to provide:

- HTTP/1.1 functionality and semantics in a compact way

WSP is designed to include the same functionality as the HTTP protocol, but with a shorter syntax.

- Unreliable data push

The push facility in the WSP protocol does not have a counterpart in the HTTP protocol. A push is what is performed when a WSP server sends information to a customer without a preceding request from the customer. A connection-oriented WSP can provide a reliable data push.

5.3 Wireless Transaction Protocol

The Wireless Transaction Protocol (WTP) provides efficient, reliable data transfer through retransmission, segmentation and regrouping, based on a request/reply paradigm. To match the different requirements of different types of applications, three classes of transport are defined:

Class 0 provides an unreliable, stateless connection, in which there is no result message and no abort function. This is much like UDP in the fixed-wire world.

Class 1 defines a reliable stateful transport. It supports retransmission and abort functions with no result message. This is used in the push service.

Class 2 defines a reliable stateful transport. It supports retransmission with a result message. It is used for the basic request/answer transmission service.

5.4 Wireless Transport Layer Security

Wireless Transport Layer Security (WTLS), derived from SSL, provides secure transmission of data between the WAP client and the WAP gateway, and supports privacy, message integrity and authentication functions. WTLS includes several key exchange suites, a number of symmetric and asymmetric encryption ciphers and signature functions. Because much of its authentication is certificate-based it can be integrated with the Public Key Infrastructure (PKI).

In contrast to SSL, which is built on top of an IP protocol, the WTLS, on top of the WDP, incorporates new features such as datagram support, optimized packet size and handshake, and dynamic key refreshing. It has been optimized for low-bandwidth bearer networks with relatively long latency.

The security layer may be divided into two sub-layers.

Fig 5-4 WTLS layer

- Handshake protocol

The handshake protocol allows customers and servers to agree on cryptographic algorithms, exchange random values and exchange the same security parameters, as well as ensure that the handshake occurred without being tampered by an attacker.

- Record protocol

The record protocol is responsible for the encryption and decryption of messages. If a message is to be transmitted, the record protocol layer will compress it (optional), encrypt it and send it to the lower layers for transmission. On the other hand, if a message is received instead, the record protocol will decrypt and decompress the message before sending it to the upper layers.

5.5 Wireless Datagram Protocol

Since IP is not supported by many of the mobile networks at this time, the Wireless Datagram Protocol (WDP) is used instead of UDP over IP. WDP provides a connectionless unreliable transport protocol with support for message segmentation and reassembly, and UDP-based port numbers. It also provides a consistent interface for all of the underlying over-the-air bearers. This means that WAP is completely independent from any network operator. The transmission of SMS, USSD, CSD, CDPD, IS-136 packet data and GPRS is supported. The Wireless Control Message Protocol (WCMP) is an optional addition to WAP which will inform users of any errors which may have occurred.

5.6 WAP Gateway

WAP is a standard which makes full use of existing resources on the Internet. Web servers which only support the HTTP protocol and which are used to store information which the client can download, cannot process requests from the WAP Client which use WSP instead. In order to retrieve information from the web server, a request in WSP format has to be translated into HTTP format. A proxy server called a WAP gateway is introduced to support the translation.

There are three separate parts to the WAP programming model as shown in Fig 5-1: client, gateway and server. The client runs a WAP user agent such as a browser, and the server runs some web server software. The gateway translates between the WSP and HTTP protocols. It also encodes the contents of a message into a form more suitable for the Client. As a WAP Client only understands a binary encoding of WML, the gateway compiles the WML page into binary WML.

This chapter has provided only a brief overview of the WAP architecture. More important for the purposes of this thesis are the security characteristics supported by WAP. These will be discussed in chapter 6.

Chapter 6

Security of WAP

One of the most important reasons WAP was chosen for the purposes of this thesis is due to its strong security. Internet business would be untenable without security, no matter how convenient it is. WAP encompasses WIM, WMLScript, WTLS and WPKI, which all apply security at the application, transport and management levels in the wireless environment.

6.1 Wireless Identity Module

The WAP Identity Module (WIM) is used to store and process information needed for user identification and authentication such as certificates and keys. It is also used in performing WTLS and application level security functions. WIMs are most commonly implemented using smart card chips that optionally reside in the WAP device. Smart card chips have memory and storage for data and programs. Being separated from the handset, users are able to change handsets without losing personal information and settings. WIM can either be incorporated with a SIM or it can be a separate card.

There are two PINs, each associated with a different private key. The access PIN protects the authentication key and session keys, while the signature PIN protects the signature key (non-repudiation key). The initial PIN is delivered to the user in the similar way in which credit card/bank card PINs are delivered, e.g. via surface mail to the home address. End user has the capability to change the PIN (on providing the current PIN). Two certificates correspond to their keys are also stored in the WIM. [37]

In order to be able to store certificates, keys and PINs, WIM has to guarantee the security of the key pair, including:

- A good quality key pair
- No copies of the private key left outside the device
- Inability to obtain the private key from the device afterwards
- PINs protecting the usage of the private key are well managed

6.2 WMLSCrypt

Although WTLS defines Server Authentication and Client Authentication, it does not define Client Authorization, or non-repudiation. Authentication is always required when a transaction begins, in order to confirm the identity of the person or entity. Authorization is always required when a decision has to be made by the customer, whether or not the server is allowed to do or have something in the transaction. In this way, no party in the transaction can deny that the transaction took place. This problem can be solved at an application level by digitally signing the transaction. A digital signature is the electronic equivalent of signing a receipt.

The WMLScript Crypto Library Specification provides cryptographic functionality for message signing. The WAP WMLScript `signText` function provides digital signatures in WAP-compliant customer devices. The operation begins at the server, which wishes to have something signed, when it sends a signature request, along with the piece of text it wants signed, to the customer. The customer signs with his or her private key and returns the signature to the server. The server then uses the customer's public key to verify that the signature has been signed by the customer.

WMLScript can utilize an underlying WIM Module to provide crypto functionality including:

- a) Generate key pairs
- b) Store keys and other personal data
- c) Control access to stored keys and data
- d) Generate and verify digital signatures
- e) Encrypt and decrypt data

6.3 WTLS

WTLS is a security protocol originated from TLS/SSL, and takes into account the specific features of the wireless environment.

- High communication cost: As propagation delay is much more significant than transmission rate in the wireless environment, WTLS must reduce the communication cost of security processing.
- Weak reliability: Wireless communication links have weaker reliability than wired links. WTLS must have mechanisms to allow a certain extent of unreliability while still guaranteeing security.
- Limited computation capability and lower transmission rate: As opposed to SSL, which requires high computation capability to encrypt or decrypt data, WTLS must redefine the cypher suites to fit this requirement.
- Lower power and less memory:

In order to be used in wireless applications, WTLS has a number of additional characteristics which SSL lacks, such as compact coding, datagram support, optimized handshake, fast encryption and decryption algorithm, etc.

- *Compact coding.*

Compact coding is a method of offsetting the speed limitations imposed by the high latency and low bandwidth of wireless networks. In this way, WTLS can keep messages as short as possible and minimize the time and processing power required by the customer device to interpret and transmit them. This also compensates for the low power and computational resources of mobile devices.

- *Datagram support.*

WTLS operates directly above WDP, and therefore needs to accommodate the unreliability and unpredictability of connectionless datagram communication.

- *Optimised handshakes.*

WTLS allows the WAP gateway, which acts as a server to the mobile WAP customer, to authenticate the customer by obtaining the customer's digital certificate from an external source, rather than from the customer itself. This reduces the processing and memory burden on the customer.

- *Dynamic key refreshing.*

As an extra security measure against eavesdropping, WTLS allows for the symmetric session key to be changed regularly over the course of the session, without the need for a clean handshake.

- *Fast encryption and hashing algorithms.*

WTLS uses the quickest, most efficient algorithms available for hashing and encryption, so that the customer's processing time and power consumption are kept within reasonable limits.

- *Customer-gateway rather than Customer-server coverage.*

Unlike SSL, WTLS does not span the whole of the communication channel from the customer to the content provider's HTTP server, but only communication over the mobile telephone network between the customer and the WAP gateway. If security is also required between the gateway and the HTTP server, it must be implemented using SSL.

6.3.1 WTLS Implementation Classes

There are three levels of security provision at various stages of adoption. WTLS Class 1 provides confidentiality and data integrity between the wireless device and the WAP gateway. Class 2 adds the authentication of the WAP gateway to the security services provided by Class 1. Finally, Class 3 builds on Class 2 by adding support for the authentication of the wireless customer.

WTLS implementations may have support for various features. A class may have mandatory (M) or optional (O) support for a certain feature.

Feature	Class 1	Class 2	Class 3
Public-key exchange	M	M	M
Server certificates	O	M	M
Customer certificates	O	O	M
Shared-secret handshake	O	O	O
Compression	-	O	O
Encryption	M	M	M
MAC	M	M	M
Smart card interface	-	O	O

6.3.2 WTLS Handshake

The WTLS Handshake is very similar to the SSL handshake. The handshaking protocol is to establish a secure session between a WAP Customer and a WAP gateway. The session will be suspended when there is a WAP customer power-off, wireless link break-up, etc. It can be resumed later when the customer supplies a session id.

Session Resume is necessary in the WTLS protocol because it can reduce the expensive communication cost of handshaking, which involves several rounds of signal information exchange. Since both customer and gateway will keep the agreed security parameters once the secure session is established, the security parameters established previously can be used when the customer resumes the session. Every session has a unique ID specified by the gateway, representing the agreement between the customer and the gateway, which can be resumed later. Session ID is only used by the customer and the gateway to resume the connection.

The handshake process is very important for agreeing on all the security-related parameters. These parameters include attributes such as protocol versions, cryptographic algorithms, and information on the use of authentication and public key techniques to generate a shared secret. Based on the WTLS specification, a full handshake creates a new secure session between two peers. The full handshake includes parameter negotiation and the exchange of public-key information between the Customer and server. The flow chart of the full handshake is depicted in Fig 6-1 below.

Fig 6-1 Full Handshake

Unlike in the full handshake, in the optimized handshake (Fig 6-2) the server looks up the customer certificate from its own source without requesting it over the air from the customer.

To accommodate the unreliability and unpredictability of connectionless datagram communication, messages are always packed as one Record Protocol packet when sent in one direction, to ensure that they are either received or lost on the other side.

Fig 6-2 Optimized Handshake

6.3.3 Digital Certificate

A Digital Certificate is very important for customer authentication and non-repudiation. The X.509 Certificate is the most widely accepted Internet standard. However, X.509 is not supported by the current generation of WAP Customer devices, as they are marked by limited capacity. The WTLS certificate is similar to the X.509 certificate but is coded more compactly, and satisfies the high latencies and low bandwidth of wireless networks, as well as the limited processing resources of WAP Customer devices. WTLS certificates omit some of X.509's non-essential fields.

With the requirement of customer certificates for Class 3 authentication, the key pairs have to be safeguarded and managed until the certificate expires. Customer certificates need not be retained on the handheld device. The customer may refer the WTLS gateway to a directory for its certificate during negotiation that might improve performance by saving the bandwidth needed to send the Customer certificate over the air. The key pair associated with the Customer must reside on the Customer only.

X.509 Certificate	WTLS Certificate
<ol style="list-style-type: none"> 1. The subject's name 2. The issuing CA's name 3. The certificate's validity period 4. The asymmetric and symmetric algorithms used for key exchange 5. The subject's public key 6. The digital signature of the issuing CA 7. Alternative names for the subject (optional) 8. Allowed key usage - for example, whether the subject's public key may be used for encryption, server authentication, signing other certificates, and so on (optional) 	<ol style="list-style-type: none"> 1. The subject's name 2. The issuing CA's name 3. The certificate's validity period 4. The asymmetric and symmetric algorithms used for key exchange 5. The subject's public key 6. The digital signature of the issuing CA

Fig 6-3 Comparison between X.509 and WTLS Certificates

A certificate can be revoked by the CA before the end of its validity period. Since it is impracticable for the customer to check a revocation list every time it downloads a gateway's certificate using a small Customer device, short-lived certificates are issued by gateway.

6.4 Wireless Public Key Infrastructure

Similar to the IETF PKI standards that are most commonly used in wired networks, WPKI standards are the most commonly used in wireless networks. WPKI, an extension of traditional PKI, is used to leverage security features including WIM, WMLSCrypt and WTLS. The key elements of WPKI are:

- Mobile phones with WIM support
- WAP Gateway-enhanced with certificate-based identity validation capability
- Registration Authority (RA) for certificate enrollment
- Back-end PKI infrastructure with access to Certification Authority

It enforces security policy by managing relationships, keys and certificates. Like all security and application services within the WAP environment, WPKI must be optimized, using more efficient cryptography and data transport techniques, in order to work with personal wireless devices and the narrow-band wireless networks. WPKI has optimized PKI protocols, certificate format, cryptographic algorithms and keys.

There are four components required in WPKI, like that in traditional PKI. They are: End-Entity Application (EE), Registration Authority (RA), Certification Authority (CA) and PKI Directory. EE and RA are implemented a bit differently from traditional PKI, and a new component, PKI Portal, is also required.

The EE in WPKI is implemented as optimized software that runs in the WAP device. It relies on the WMLSCrypt API for key services and cryptographic operations. It is responsible for the same functions as the EE in traditional PKI including

- generate, store and allow access to a user's public key pair
- complete, sign and submit first-time certificate applications

- complete, sign and submit certificate-renewal requests
- complete, sign and submit certificate-revocation requests
- search for and retrieve certificates and revocation information
- validate certificates and read the certificate contents
- generate and verify digital signatures

The PKI Portal logically functions as the RA and is responsible for translating requests made by the WAP client to the RA and CA in the PKI. The PKI Portal will typically embed the RA functions and interoperate with the WAP devices on the wireless network and the CA on the wired network.

A detailed data flow between these components is presented in Fig 6-4 below.

Fig 6-4 Major technical components and operational flow of WPKI [7]

With the security characteristics provided by WAP, we can now implement a secure wireless payment protocol (SWPP). A detailed description of SWPP is given in Chapter

Chapter 7

Secure Wireless Payment Protocol

From the discussion in Chapter 5 and 6, we know that WAP provides an integration point for securing wireless access to sensitive content and transactions. From the survey in Chapter 3, WPP is a wireless payment protocol which is more convenient than MeT. In WPP security is actually not well addressed, based on the assumption of security function supplied by WTLS. SWPP is a protocol redesigned for WPP, aiming at its security requirement. In the following section, we will present our motivation on SWPP and the design of SWPP.

7.1 Motivation

Convenience and ease of use is the most vital component of any mobile payment solution. Customers will not take advantage of the mobile payment system if the execution of a mobile transaction of any kind is not simple and straightforward to complete. That is what WPP does. WPP is a secure protocol which is more convenient than the MeT protocol as it masters the payment transaction between the customer and the customer's bank. A dual signature is not required in WPP as messages are not going to pass through any intermediary.

However, convenience does not make sense without strong security. With a good payment system, there is always a tradeoff between convenience and security. WPP actually does not address security as it assumes that the WTLS protocol will provide the confidentiality and integrity of all messages exchanged between all participants of the protocol. SWPP is proposed here to make up for WPP's security deficiencies. SWPP

draws upon WAP, which is designed to make full use of Internet services, for WTLS, WIM, WMLScript and WPKI to guarantee the security requirement presented in WPP. Details on SWPP are presented below.

7.2 Security Requirement

There are five common security requirements, which have been well documented for pervasive e-business ever since the Internet became popular. This is the same for m-business.

- **Authentication:** With any communication, the user wants to know who he is communicating with, and the other party needs to be certain that he is dealing with the person he thinks he's dealing with. Server Authentication ensures that the user is certain that the web site's server is who it claims to be. Customer Authentication is a means of establishing the user's identity. For high security, bi-directional authentication (combining both server authentication and customer authentication) is the best choice.
- **Confidentiality:** As a synonym for secrecy, confidentiality means protecting the data communication from eavesdropping.
- **Integrity:** Integrity means verifying that the data has not been altered in transit by a third party. This is to prevent forgery, tampering, and unauthorized alteration.
- **Authorization:** Authorization limits the improper use of data and services. Authorization limits what information the user is allowed to see or what actions one may take to minimize the chance of exposure of sensitive information to malicious attack or unauthorized alteration.

- **Non-repudiation:** Non-repudiation is a way of preventing parties in the data transaction from denying their actions after the transactions are done. This is to enforce accountability for electronic transactions.

Besides these five requirements, another important requirement for wireless security is the creation of a boundary for the service domain. This reduces the chance of attack by hackers from public networks. Firewalls are most commonly used to prevent such attacks, which may include denial of service, packet spoofing, and impersonation. [10]

7.3 Security Addressed in SWPP

SWPP is designed to help create a safe environment to support pervasive online mobile payment. It provides for centralized user authentication from limited points of entry. SWPP exploits the single sign-on for user-friendly implementation of credential sharing across the services hosted by the Nokia Activ Server. SWPP relies on a set of industry standards for security, such as TLS/SSL and WTLS, to achieve the security objectives for the service domain. SWPP uses proxy technology in conjunction with firewalls to define the boundary for the service domain.

Authentication

Both the bank server and the merchant server use the HTTP basic authentication process to authenticate the customer's information as it comes in from the mobile ends. The customer authentication function performed by the bank gateway is registered on the Nokia Activ Server. Both the clients and the gateway proxy for the clients must support HTTP.

To authenticate the merchant on the bank server, the message from the merchant has to be signed then verified on the bank server.

Confidentiality

SWPP utilizes a set of industry standard security technologies to implement confidentiality. Such technologies include the Secure Socket Layer (SSL) and Wireless Transport Layer Security (WTLS).

Authorization

There are two layers of authorization defined in SWPP.

The access control in SWPP is implemented using HTTP proxy technology. The implementation of finer levels of access control can be done on the Nokia Activ Server.

On authorization by the user, the bank server presents the payment transaction to the customer, which is then confirmed by the customer. This is implemented using the WMLScript signText function

Data integrity

SWPP provides data integrity using features such as message digest and certificates included in the security technologies SSL and WTLS.

Non-repudiation

Because SWPP utilizes SSL and WTLS with certificates, non-repudiation can be also implemented at the transaction level. SWPP provides non-repudiation using the WMLScript signText function, which authorizes the user at the same time.

7.4 Architecture of SWPP

A generic infrastructure of SWPP and its transaction flow is presented in Fig 7-1.

Fig 7-1 SWPP Infrastructure and Payment Transaction Flow

7.4.1 Description

The gateway in Fig 7-1 is used to locate the bank gateway or merchant gateway as requested by the customer. A security channel between the Merchant Agent and the customer is assumed here, as it is quite similar to that between the customer and the customer's bank. In this way, we do not have to take into account a Merchant Gateway between the customer and the Merchant Agent.

1. The merchant generates and sends the payment information to the Customer, along with the merchant's banking information encrypted by the customer's bank's public key. An invoice is presented to the customer to confirm.

With the assumption of a secure communication channel between the Customer Agent and the merchant, a payment order and the merchant's banking information,

encrypted by the merchant's private key, are sent to the Customer's Bank, encrypted by the public key of the Customer's Bank. The merchant's banking information, which was signed by the merchant bank's private key and encrypted using the Merchant's public key, can be retrieved from the merchant's Smart Card, which was decrypted using the Merchant's private key. The banking information can be retrieved from the Smart Card, which is protected by a PIN. An additional invoice, encrypted using the customer's public key, is sent to the customer to confirm.

2. The customer confirms the content of the invoice, prepares a payment authorization and sends it to the customer's bank.

The invoice information from the Merchant is decrypted using the Merchant's public key and presented to the customer. If the customer agrees with the invoice, it will forward the encrypted payment order and the merchant banking information to the customer's bank. The customer's bank account number can be retrieved from the banking information stored on his or her smart card, which is protected by a PIN. The banking information previously encrypted using the customer's public key can be decrypted using the customer's private key. Before the message is sent from the customer to the customer's bank, a secure WTLS session has to be established by mutual authentication. SSL is used between the bank's gateway and the customer's bank for end-to-end security between the customer and his or her Bank.

3. The customer's bank fulfills the customer's authorization request and forwards the response to both the customer and the merchant's bank

The customer's bank gateway decrypts the information using a session key and verifies the signature of the payment order and the merchant banking information.

The customer's bank will then generate some data for the Customer Agent to sign, thereby guaranteeing non-repudiation at the application layer. The Customer Agent can cancel the transaction at this time.

While authorization by the customer is taking place, its bank will process the request from the Customer and withdraw money from the customer's account, then forward the request to the merchant's bank. The transaction between the merchant's bank and the customer's bank is assured to occur on a secure internal bank network.

4. The merchant's bank sends a confirmation to the customer's bank and the Merchant Agent

The merchant's bank will process the request from the customer's bank, deposit the money to the merchant's account and send a confirmation to the customer's bank and the Merchant Agent. The customer's bank will then forward the confirmation to the customer.

7.4.2 Payment Transaction Flow

In SWPP, the customer has two private keys and certificates. Two PINs are required to get access to the keys. The private key must be stored with the customer in a tamper-resistant hardware token called a wireless identity module. This can be collocated with the existing SIM card to form a so-called 'SWIM' card. An alternative approach, which may be driven by the financial community, is to implement the WIM as a separate smart card. Here we use a SWIM card.

Customer SWIM: Provides static storage for personalized data such as certificates, keys, PINs and encrypted information concerning the Customer's Bank.

In the payment transaction flow shown in Fig 7-2, we use MBI for Merchant Bank Information, CBI for Customer Bank Information and PI for Payment Information. The bank gateway actually exists in the local intranet with the customer's bank. In Fig 7-2, we do not consider the merchant gateway, as there is an end-to-end security assumption between the customer and the merchant. The gateway is not taken into account either because its main function is to relocate the Gateway that provides a corresponding service to the customer. What will be discussed here is how a security channel is built between the customer and the customer's bank through the Bank Gateway in the middle. Details on how to build a WTLS session channel between the customer and the customer gateway will be given in section 7.5 of this chapter. End-to-end security is discussed in section 7.6. Details on how to process a customer authorization will be discussed in section 7.7.

Fig7-2 Secure Payment Transaction Flow

7.5 Secure WTLS Session

As the ordering and payment/settlement phases, as well as the recourse mechanisms, are outside the scope of this protocol, the SWPP process commences when the merchant sends an invoice to the customer and terminates when the merchant receives a confirmation of payment from its Bank. This section will contain a discussion on how the secure WTLS Session Channel is built between the customer and the bank gateway. We will talk about end-to-end security using SSL protocol between the gateway and the customer's bank in section 7.6.

7.5.1 WAP Security Architecture

In order to understand why there is no direct secure channel between the Customer Agent and the customer's bank, it is necessary to review WAP's security architecture.

Like any mobile Internet transaction, SWPP extends across both the mobile telephone network and the Internet, using WAP for the wireless portion and the Hypertext Transfer Protocol (HTTP) suite for the Internet portion, and a WAP gateway in the middle to translate between the two. The WAP gateway is an access point to the fixed Internet for wireless devices, converting both data and protocols from the wireless to the wired network. At this point, at the transport layer, there is a security protocol translation between WTLS and SSL. There is clearly vulnerability at the WAP gateway as encrypted data is deciphered to clear text before being re-enciphered using SSL. This will be addressed later.

Fig 7-3 WAP Security Architecture

First it is necessary to build a secure session between the Customer Agent and the WAP Gateway.

7.5.2 WTLS Handshake process

As we know, WTLS is composed of a Handshake protocol and a Record protocol. The Handshake protocol allows customers and servers to agree on cryptographic algorithms, exchange random values and exchange the same security parameters. In SWPP, the Customer Agent will negotiate the security related parameters with the Bank Gateway, based on the Handshake protocol with the help of a WIM card.

WIM is a tamper-proof card and plays an important role in the Handshake protocol. Besides storing certificates, keys and pins, which will later be retrieved by the Handshake protocol, WIM is also used to generate a high quality random number for the pre-master secret, and to encrypt the pre-master secret with the server public key, derive the master secret and calculate all values based on the master secret. These functions are required during the Handshake process. First, the functionality of WIM in SWPP will be discussed in detail.

The functionality of WIM in the full Handshake is presented in Fig 7-4. The process is explained as follows:

1. *Read Configuration:* Through this function, the Customer Agent ascertains the algorithms supported by the WIM, as well as the information on the keys and certificates stored in the WIM. The key information from the WIM is then sent to the bank gateway in the WTLS handshake's ClientHello message. The Customer Agent may have the configuration and other data in a cache, so that it is not read during every new handshake.
2. *Generate Random:* 12 random bytes are generated to be used in the ClientHello message later on.
3. *Read Customer Certificate:* During this process, the customer certificate is retrieved from the WIM card. This is required when the bank gateway is unable to obtain the customer certificate from its own sources. It then requests the customer to send its certificate.
4. *Verify Server Certificate:* This processes the certificate received from the bank gateway. The Customer Agent must verify the signed data in the certificate to provide Server Authentication.
5. *Establish pre-master secret:* The pre-master secret is the 20-byte value, consisting of the 1 byte customer version number and 19 random bytes appended with the server's public key. The WIM retains the pre-master secretly in its memory to be used during the next operation.
6. *Derive Master Secret:* A master secret is derived based on the pre-master secret established in the previous operation, and a seed value is received from the Customer Agent. The Pseudo Random Function (PRF), SHA-1 must be used. The WIM stores the master secret persistently, accessible only under a certain session/key id.

7. Sign $H(\text{handshake_messaging})$: The WIM signs a hash of handshake messages transmitted so far between the customer and the bank gateway.
8. *Calculate customer finished check*: Using the PRF, the WIM calculates the requested number of bytes based on the master secret, and a seed value received from the ME.
9. *Calculate server finished check*: This is quite similar to the Calculate customer finished check described above.

Calculate customer write block: Using the PRF, the WIM calculates the requested number of bytes based on the master secret, and a seed value received from the ME.
10. *Calculate server write key block*: Similar to the Calculate customer write key block, with a different seed.
11. *Write address and session data*: The Customer Agent stores all the information that is needed to resume the session later on. This covers address- and session-related data.

The master secret is handled internally by the WIM.

Fig 7-4 Full Handshake based on WIM

From the analysis above, the operations supplied by WIM should be executed following certain rules:

- The derive master secret operation must be preceded by an establish pre-master secret operation.
- All WIM-Phash operations must be performed after the derive master secret operation.
- The customer finished check must be calculated before the server finished check.
- The write session data operation must be performed only after the Server Finished message has been verified.

To support the Handshake protocol, the Customer Agent must be able to use CA certificates to verify server certificates and user certificates. The CA's root certificates are stored in the WIM. The Customer Agent should also store peer address and session data in the WIM in case the session is interrupted or cancelled. The WIM's functionality when resuming a session is explained in Fig 7-5.

Fig 7-5 Optimized Handshake based on WIM

In the optimized handshake, the WIM is used to calculate all values that are based on the master secret of the current session. The WIM operations used are similar to those in a full handshake. Read session data is used to read all address- and session-related data which are needed to resume the session. The master secret is handled internally by the WIM.

Using the storage and functionality of the WIM, as discussed above, the Customer Agent can now build a secure session with the bank gateway. The full process for building a secure session is presented in Fig 7-6.

Fig 7-6 Secure Transport Session between Customer Agent and Bank Gateway

The detailed process for building a secure transport session is explained as follows:

1. The customer is satisfied with the invoice from the Merchant Agent.

2. The customer activates a link on his WAP phone that sends a request for a secure Internet session to the bank gateway. The following information is appended to the request:
 - a. Information regarding the versions of WTLS supported by the customer's WAP browser, what encryption algorithms it supports, and so on.
 - b. Some randomly generated data. This will be used, along with other data, to generate the session key.
3. The gateway receives the customer's request for a secure session and sends back the following items:
 - a. Its digital certificate, including its public key. The certificate is signed by a trusted CA, using its private key.
 - b. Information concerning what versions of WTLS, encryption algorithms and so on are supported by the WAP gateway.
 - c. Some randomly generated data that will be used in the generation of the session key.
4. The customer sends its digital certificate to the gateway.
5. The customer validates the bank gateway's certificate. On the gateway side, it will validate the Customer's certificate.
6. The customer performs a series of operations on the random data which will be sent to the bank gateway, and the random data the bank gateway sent back, to produce the pre-master secret.
7. The customer encrypts the pre-master secret using the bank gateway's public key and sends it to the bank gateway.

8. The bank gateway receives and decrypts the pre-master secret.
9. The bank gateway and the customer simultaneously perform a series of operations to retrieve the master secret.
10. The customer and the bank gateway simultaneously perform a series of operations on the master secret to arrive at the session key, which will be used to encrypt the information they want to send to each other.
11. The customer sends the bank gateway two messages. The first confirms that all further messages from it will be encrypted using the session key. The second is an encrypted message that formally ends the handshake from the customer's side.
12. The gateway sends the customer two messages. The first confirms that all further messages from it will be encrypted using the session key. The second is an encrypted message that formally ends the handshake from the bank gateway's side.

7.6 End to end Security

From the WAP security architecture mentioned in Chapter 7.4.1, we know that the WTLS protocol cannot build end-to-end security between the customer and the customer's Bank. A WAP gateway must be used as a bridge between the different protocols. Not WTLS but SSL is supported when the WAP gateway makes the request to the origin server. As the data is decrypted and again encrypted at the WAP gateway, the gateway introduces a security hole which renders WAP unsuitable for any security-sensitive services.

In SWPP, with the guaranteed security required by the banking sector, the gateway is hosted by the content provider and placed behind the content provider's firewall. In Fig 7-1, the merchant's and the customer's bank have their own gateways in their own network. By placing a WAP gateway in their own network, the connection between the

customer and different services (including the merchant service and the bank service) is to be trusted as the decryption will not take place until the transmission has reached the service provider's own network, and not in the mobile operator's network. To provide the highest security solution, the functionality of the WAP gateway to the origin server can be included. This is the way it is used in our implementation. This set-up obviates both the WAP Gap and the need for SSL between the gateway and the HTTP server.

Fig 7-7 outlines an end-to-end secure transaction in SWPP, with the WAP gateway behind the content provider and firewall.

Fig 7-7 End-to-end Secure Transaction

Since both the merchant's and the customer's bank provide WAP services to the customer, they have their own gateway in their own network. With these two gateways in SWPP, the customer agent needs a mechanism to navigate between them. In Fig 7-3, a gateway is shown as the solution. The gateway acts as a master WAP gateway which

supplies a navigation document to the customer. The other two gateways are subordinate WAP gateways supplying WAP services.

Fig 7-8 Transport Layer in an end-to-end Security Architecture

From Fig 7-8 above, there are three key components which provide total navigation for the Customer Agent:

- Navigation Document. This is the document that instructs a WAP customer on how to navigate to a subordinate WAP gateway. It is quite similar to a map with instructions on where to find a WAP service on the network.
- Master WAP gateway. This is a WAP gateway that is authorized to give navigation documents to a WAP customer. It can act as a subordinate WAP gateway at the same time.

- Subordinate WAP gateway. This is a gateway which wishes to establish an end-to-end secure link with the WAP customer. To do this, the subordinate gateway must be able to provide a navigation document to the master gateway over a secured link.

In the future, the WAP gateway might be discarded altogether, which would turn the WAP-enabled mobile phone into an ordinary Internet device. This thesis focuses mainly on the gateway between the customer and the customer's bank. The bank gateway acts as both master gateway and subordinate gateway. A navigation document is unnecessary in SWPP.

7.7 Authorization by Customer

Once the secure transport session is established, payment information and other relevant information are sent to the customer's bank via a secure channel which is built. After the customer's Bank decrypts the information from the Merchant and verifies its signature, the payment information is sent back to the user as a string to sign for signature. Customer Authorization is necessary for security on the application layer.

Fig 7-9 Customer Authorization

Customer Authorization in SWPP is implemented using the WMLScript signText function. The following transpires in the signText function:

- The customer's bank provides a text to be signed and an indication of the required user certificate (as a list of accepted certificate issuers) or key (as a key identifier)
- The customer confirms the text and certificate being used

- The customer is prompted to input a signature PIN
- WIM executes a sign function on the hashed text using a signature key

Fig 7-10 below shows the data flow.

Fig 7-10 Successful Transaction

7.8 Exceptional Cases

There are some situations where the transaction cannot be ended successfully, such as authentication failure, network error, and so on. Here we will discuss only two authentication errors on the customer side (one during handshake, one when the customer's bank requests the customer's signature). The customer can also cancel the transaction if he changes his mind.

- **Customer verification failure**

During the Handshake process, if the certificate and key that the customer attempts to authenticate to the bank are not valid for the chosen banking service, the customer will be informed that he or she cannot access the banking service.

- **Customer cancels transaction**

If the bank gateway requests a signature for the transaction contract using the WMLScript `signText` command, the user may choose to cancel the transaction for a variety of reasons: he may change his mind, or doubt whether the transaction to be signed accurately reflects the transaction requested. In this case the message flow is the same as Fig 7-10 until the bank requests the user to sign the transaction. Cancel information will be sent to the bank gateway and confirmation of the cancel process will be sent from the bank gateway to the user.

- **Customer signature invalid**

If the Customer signs the transaction, but the signature key used is not valid for signing transactions in this service, the message flow is the same as Fig 7-10 until the bank requests the user to sign the transaction. A Transaction Rejection will be sent from the bank gateway to the customer.

This chapter presented an overview of our solution to designing a secure payment protocol based on WPP. Details were given on the security components necessary for SWPP, including building a WTLS session between the customer and the bank gateway; end-to-end security between the customer and the customer's bank; and customer authorization using the `signText()` function defined in the WMLScript Library. Chapter 8 will provide a detailed description on how to implement SWPP's security character.

Chapter 8

Implementation of SWPP

This chapter will contain a discussion on SWPP implementation based on WAP and WPP. Since SWPP is a protocol redesigned for WPP, intended to balance the requirements of convenience and security, a great deal of attention will be paid to its security implementations, such as Server Authentication, Customer Authentication and Customer Authorization. These will be discussed in detail.

8.1 Introduction

SWPP is intended to bring Internet banking to the mobile device. Convenience and security are the two key points to be considered. Features like trusted user authentication, signed transactions, data integrity and encryption will provide SWPP end-to-end security for end-user transactions between the mobile device and the bank or merchant's back-end system.

SWPP is based on WAP infrastructure that provides a fast, secure, on-line and interactive connection method. Convenience is achieved through the use of WAP technology, combined with having the customer contact his or her Bank, rather than the Merchant. WAP products usually include a WAP gateway, an application developer's kit and mobile phones. The mobile phone used as the client must be WAP-enabled, and the website provided by the merchant's server must be in a WAP format.

With regard to security, WTLS - specified in WAP - is used in SWPP to guarantee end-to-end security. SWPP also supports established techniques for data integrity and encryption, including WPKI. WPKI consists of protocol extensions, together with

software and hardware additions to terminals and networks that extend traditional PKI to wireless networks. It is intended to enable the implementation of scalable application-, network-, and supplier-independent security solutions in mobile networks and terminals.

Emulators are used to simulate the mobile device during development. In accordance with WAP specifications, many companies, such as Nokia, Ericsson and Motorola, have developed their own products. With the SWPP security requirement, both the Gateway and the browser emulator should have support in WTLS. Until the implementation began, only the Nokia Company gateway supported WTLS Class 2. Its corresponding toolkit is probably one of the most prevalent products, with a user-friendly interface. It is also WTLS Class 3-supported. Both were supported by WAP in June 2000 (WAP 1.2.1), a widely accepted standard conformance release.

In this thesis, the Nokia Activ Server is used as the WAP gateway. Security is supplied by an added-on product, Nokia Activ Security, which is a product that ensures secure communication and supports WTLS all the way from the WAP terminal to the origin server. It encrypts the traffic between the WAP terminal and the Nokia Activ Server, and provides methods for certificate-based server authentication. A Key Exchange algorithm, Ciphersuites (which includes a bulk algorithm and an MAC algorithm) and server certificates can be deployed on the gateway.

A server certificate is a standard record that contains a description of the server owner and the Server's public key. The public key is used for server authentication and key exchange. The certificate used in SWPP is self-signed for test purposes. Server certificates can be bought from companies such as Verisign, Entrust, Baltimore, and so on.

8.2 Development Environment

SWPP is here developed under Windows 2000 (SP 2), using Nokia Toolkit and Nokia Activ Server, Nokia Activ Security, Lite Web Server (the pure Java web server with support for Servlet API 2.2, which is small in size and easy to use.)

With Nokia WAP Toolkit's integrated development environment, WML files, WMLScript files, and WBMP images can be created, and the creation can easily be viewed via the built-in emulator. The Toolkit can be used as a WAP June 2000 Simulator which supports most areas of functionality described in the June 2000 version releases of the WAP specifications, including Push, WTAI, WTLS, and WIM. Nokia SoftID is used to simulate the WIM device for developing security applications. It can save data such as Keys, PINs and Certificates.

The Nokia Activ Server, together with Nokia Activ Security, can be used as a gateway to an Internet web server over HTTP. It also supplies a development platform to implement stand-alone applications or connectors to various kinds of back-end systems. Java TM Servlets can be implemented on top of the Nokia Activ Server API (Application Programming Interface). Additionally, static WML or other types of content can be stored locally in the server.

Until now, Nokia Activ Security has only supported WTLS Class 2. SSL is supported to secure the communication between the Nokia Activ Server and the origin servers. For every request from an end user, the server supplies services such as URL Mapping, Content Encoding, Caching, Authentication and Access Control. Since WTLS Class 3 is not supported on the gateway, we combined WTLS Class 2 and an external authentication module that authenticates users of WAP services against the existing

security infrastructure. Considering the high security demand of mobile banking, we choose a 128-bit version that supports key lengths of 1024-bit for asymmetric and 128-bit for symmetric algorithms. It contains RSA and DH anon algorithms for key exchange as well as RC5 and DES, 3DES for bulk encryption. SHA-1 and MD5 algorithms provide data integrity.

In order to use existing technologies as much as possible, the content requested by the end user is normally placed on a web server. In this application, we put the Merchant's content directly onto the Lite Web Server. Servlets technology is used to generate a dynamic WAP application because its performance, consistency, portability and connectivity are better compared with the Common Gateway Interface (CGI).

From the analyses above, the environment used is based on Java-enabled technology.

Products used included:

- Nokia Activ Server 2.1
- Nokia Activ Security (128-bit)
- Nokia Toolkit 3.0
- JDK 1.3.1
- Java Runtime Environment 1.3.1
- Java Servlet Development Toolkit 1.2.2
- Lite Web Server 2.2.2

8.3 Architecture

The general SWPP application architecture is shown below:

Fig 8-1 SWPP implementation architecture

Based on the SWPP implementation architecture, there are four actors involved including the merchant, customer, merchant's bank and customer's bank.

- The merchant generates and sends the customer signed payment information, along with the signed merchant's banking information. An invoice is presented to the customer to confirm.

We used servlets on the Merchant Web Server to generate dynamic content to process requests from Customer, assuming the transaction between the merchant and the customer is secured.

- The customer confirms the invoice, sending signed payment information and the merchant's banking information to the customer's Bank after a secure channel has been established between the customer and the customer's bank.

We used a WAP J2K simulator to request information from the merchant and the customer's bank. The user agent is a WAP browser for viewing WML pages and executing WML Scripts.

- The customer's bank fulfills the customer's authorization request. After successfully processing the request, a response is forwarded to both the customer and the merchant's bank

We used the Nokia Activ Server to act both as gateway and content server, based on the end-to-end security requirement that the gateway and the content server should exist in the same intranet and architecture of the servlet architecture. An HTTP server is not required. In this way, the servlet can retrieve the WAP-specific user and protocol information. Customer Authentication can be implemented on the Nokia Activ Server.

- The merchant's bank processes the request from the customer's bank and sends a response to the merchant and the customer's bank.

A server socket is open and listening to the connection from the customer's bank.

During the implementation, a great deal of emphasis is placed on the security of the connection between the customer and his bank, while the security between the customer and the merchant are not really taken into consideration. One reason for this is that the security requirement between the merchant and the customer is not as high as that between the customer and his or her bank. The other reason is that security between the

merchant and the customer can be easily achieved after security has been implemented between the customer and his bank as they are based on the same theory.

As each bank has its own security infrastructure, we assume that security is guaranteed between the customer's bank and the merchant's bank.

8.4 Data Flow

To implement the architecture presented in section 8.2 of this chapter, their separate functions must be implemented as below:

Fig 8-2 SWPP Work Flow

Before the customer can request a banking service, a secure WTLS connection has to be established. With the WTLS class 2 supported by Nokia products, a Handshake message is exchanged between the gateway and browser automatically, and both sides agree on the same algorithm.

The messages and the data elements which accompany the functions are presented as follows:

1. Payment request from merchant to customer. (SallyJoeServlet)

Data	Description	Comments
SPaymentInfo: InvoiceNum Payer Payee Amount RequestDate	This information records the data for the invoice. InvoiceNum can uniquely identify the particular invoice while payee stands for the merchant's unique identifier.	SPaymentInfo is signed by merchant using its private key. Default RequestDate equals system date.
SMBankInfo: MName MProfile MBankURL MAccountType	Unique number identifying the merchant's bank and the location where the transaction is to be made.	SMBankInfo is signed by the merchant using its private key.
MCertificate	Merchant's certificate which is accompanied with its private key.	MCertificate is used to verify MbankInfo and PaymentInfo.
InvoiceInfo: InvoiceNum Items Price PaymentAmount	InvoiceInfo is presented to Customer to confirm.	

2. Information sent from Customer to Bank Gateway (SwppAuthenticatorServlet)

Data	Description	Comments
CustomerInfo: PIN (CProfile) Password	This information is input from the end user. PIN stands for the customer's account profile that can be retrieved from the customer's smart card.	Only an authenticated customer can have access to Bank system.

3. Payment request from the customer to the customer's bank. (Customer to BankServlet)

Data	Description	Comments
SPaymentInfo: InvoiceNum Payer Payee Amount RequestDate	This information records the data for the invoice. InvoiceNum can uniquely identify the particular invoice while payee stands for the merchant's unique identifier.	SPaymentInfo is signed by Merchant using its private key. Default RequestDate equals system date.
SMBankInfo: MName MProfile MBankURL	Unique number identifying the merchant's bank and the location where the transaction is to be made.	SMBankInfo is signed by Merchant using its private key.
MCertificate	Merchant's certificate, which is accompanied with its private key.	Mcertificate is used to verify MbankInfo and PaymentInfo.
CBankInfo: CName CProfile CBankURL CAccountType	Unique number identifying the customer's bank and the location where the transaction is to be made	CProfile equals the authenticated PIN. CBankURL is used to contact the customer's bank.

4. Information sent from the customer's bank to the customer for Authorization (SignPaymentServlet, CryptoScriptServlet)

Data	Description	Comments
PaymentInfo: InvoiceNum Payer Payee Amount RequestDate	Payment Info is presented to the payer to sign for authorization.	Authorization is to be done after payment info and MbankInfo are successfully verified based on the merchant's certificate.

5. Payment request from customer's bank to merchant's bank (This is for simulation only: it is out of definition in SWPP.) (ClientServlet)

Data	Description	Comments
TransferNum	Unique number to identify the transfer.	
PaymentInfo: InvoiceNum Payer Payee Amount RequestDate	This information records the data for the invoice. InvoiceNum can uniquely identify the particular invoice while payee stands for the merchant's unique identifier.	
MBankInfo: MName MProfile MBankURL MAccountType	Unique number identifying the merchant's bank and the location where the transaction is to be made.	

6. Payment notification from merchant's bank to customer's bank and merchant.
(MerchantBankServer)

Data	Description	Comments
RefNum	Unique number to identify the response.	
PaymentNum	Unique number to identify the deposit.	
InvoiceNum	InvoiceNum can uniquely identify the particular invoice.	
TransferNum	Unique number to identify the transfer.	
ResponseMessage		

7. Payment notification from customer's bank to customer. (BankServlet)

Data	Description	Comments
RefNum	Unique number to identify the response	
PymtReqNum	Unique number to identify payment request.	
InvoiceNum	InvoiceNumber can uniquely identify the particular invoice.	
TransferNum	Unique number to identify the transfer.	
ResponseMessage		

The SWPP work flow can be explained as follows:

1. From the customer to the merchant, the customer will login using the PIN he has previously registered. The merchant will present an invoice for the customer to confirm before the payment is made. The payment information and the merchant's bank information, signed using the merchant's private key, are sent to the customer once the customer has confirmed.
2. The customer will be authenticated by the bank gateway in order to build a secure connection between the customer and his or her bank using the signed payment information.
3. The signed merchant's bank information and payment information are sent on the secure channel from the customer to the customer's bank. The customer's bank will generate a pymtReqNum to identify the request and the bank will verify that the information received is signed by the merchant
4. Customer Authorization
 - 1) The customer's bank sends the payment information back to the customer to sign.

- 2) The signed data is sent back to the customer's bank
- 3) The customer's bank verifies the signature
5. The customer's bank withdraws the money from the customer's account and transfers the data to the merchant's bank to be processed.
 - 1) The customer's bank generates a transaction number.
 - 2) The customer's bank contacts the merchant's bank through the payee's bank URL
 - 3) The transaction number, payment information and the merchant's bank information are sent to the merchant's bank.
6. The merchant's bank deposits the amount to the merchant's account and generates a *payment number* to identify this deposit. The merchant's bank sends a notification to both the merchant and the customer's bank.
7. After a response has been successfully received from the merchant's bank, the customer's bank will send notification to the customer.

8.5 Program Structure

The main concern of this thesis is the security between the customer and the customer's bank, including Server Authentication, Customer Authentication and Customer Authorization.

8.5.1 Server Authentication

To authenticate the server on the user's side, the user should have the server's public key.

The following steps are required:

1. Create a self-signed WTLS certificate using the Nokia Activ Server certificate tool.

2. Import the Certificate to the Nokia Activ Server.
3. Import the Certificate to the Nokia Toolkit.
4. Secure a connection request from the Nokia Toolkit.
5. During the handshake, the server is authenticated.

8.5.2 Customer Authentication

Fig 8-3 Customer Authentication

The customer authentication is implemented using the Nokia Active Server Authentication API, which allows the user who is putting in a request in this session to be authenticated by an external Authentication Module.

1. The customer makes a request ® to the Nokia Activ Server.
2. If the customer is unauthenticated, the request is redirected to an authentication module.
3. The authentication module authenticates the customer.
4. If the authentication is successful, the original request ® is processed by the Nokia Activ Server, which then sends a response to the customer.

5. If the authentication fails, the Nokia Activ Server sends a response to the customer stating that the authentication has failed. If the customer makes another request, the process is restarted from the first step.
6. The Authentication Module is used along with Access Control. Every time the user accesses his bank's website, he is required to authenticate himself.

The SWPPAuthentication Module is what each authentication module must implement in order to be able to authenticate a user. It is composed of three components:

SWPPAuthenticationModule Interface

SWPPAuthenticationInfo Interface

SWPPAuthenticationException

The SWPPAuthenticationModule Interface is the entry point for the authentication process and defines one method.

```
public void authenticate (int authState, AuthenticationInfo authInfo)  
  
    throws AuthenticationException
```

SWPPAuthenticationModule.authenticate takes two parameters: an AuthenticationInfo object and an integer authState. The AuthenticationInfo interface is used to deliver information, including user requests and user credentials, to the Authentication Module.

```
Properties dialogSwpp = new Properties();  
  
dialogSwpp.setProperty ("DIALOGPROMPT", "Client Authentication");  
  
dialogSwpp.setProperty ("DIALOGTYPE", "USERNAMESECRET");  
  
dialogSwpp.setProperty ("USERNAMEPROMPT", "Account number:");  
  
dialogSwpp.setProperty ("USERSAMELENGTH", "3");  
  
dialogSwpp.setProperty ("USERSAMEFIELDTYPE", "NUMERIC");
```

```
dialogSwpp.setProperty ("SECRETPROMPT", "Password:");  
  
dialogSwpp.setProperty ("SECRETLENGTH", "3");  
  
dialogSwpp.setProperty ("SECRETFIELDTYPE", "NUMERIC");  
  
dialogSwpp.setProperty ("LINKPROMPT", "Sign In");  
  
authInfo.setUserDialogFlag (dialogSwpp);
```

Once the user's credentials have been obtained and validated, the SWPPAuthentication Module will inform the Nokia Activ Server that the user has been authenticated.

```
authInfo.setAuthenticationOK (strUser);
```

Once the authentication is successful, we set the expiration time to 20 seconds. In this way, every time the end customer re-enters the bank's WAP site, customer authentication is generally required.

```
authInfo.setExpiryLifetime(20);
```

8.5.3 Customer Authorization

In order for content providers to authorize a customer, they should have the ability to receive signatures sent from a mobile device and verify their validity. The tasks performed are:

- Generates signText messages
- Validates signText messages

Messages generated by the content provider are based on information received from the customer such as payment information. The customer will then choose a non-repudiation certificate to sign the text. This is implemented by calling `Crypto.signText()`, which is supported by Nokia Toolkit.

The following code is to call WMLScript.

```

out.println("<do type='options' label='Sign'>");

out.println("<go href='http://www.swpp.au/cryptoscript/' method='post'>");

out.println("<postfield name='data' value='$(data)'/>");

```

Here is the snippet from SignPayment.java to generate the WMLScript which will call Crypto.signText(),

```

String strData = request.getParameter("data");

response.setContentType("text/vnd.wap.wmlscript");

PrintWriter out = response.getWriter();

// WMLScript Content for signText()

out.println("extern function sign() {");

out.println("var signed;");

out.println("signed = Crypto.signText" + "(" + strData + "',5,0,");

out.println("WMLBrowser.setVar('signed',signed);");

out.println("WMLBrowser.refresh();}");

out.close();

```

The call to signText is generally presented as:

```
var signature = Crypto.signText(stringToSign, 5, 0, "");
```

Validates signText Messages:

To validate the signText Message, we first need to decode the base-64-encoded messages because the signed data returned by Crypto.signText() is a base-64-encoded signText.

```
byte[] signTextBytes = Base64Coder.decode(strSignedData.getBytes());
```

The signature can then be verified on the signTextBytes and return an array of bytes which contains the original data:

```
byte[] data = verifyAndReturnText(signTextBytes);
```

8.6 Evaluation

- i. Once the merchant's URL has been entered, server authentication is required as the Nokia Activ Server acts as a gateway between the merchant and the customer.

After the handshake with the customer has been completed successfully, a WTLS Class 2 is built between the gateway and the customer.

WTLS Connection Parameters	
Security Parameter	Value
Bulk Encryption Algorithm	RC5_CBC
MAC Algorithm	SHA
Key Exchange Suite	RSA
Key Size	Client Authentication Not Done
Server Certificate	
Certificate Field	Value
Subject Name	swpp; carleton; CA; 24.103.226.13
Issuer Name	swpp; carleton; CA; 24.103.226.13
Valid Not Before	Wed Apr 10 01:00:00 PDT 2002
Valid Not After	Wed Apr 11 01:00:00 PDT 2007

- ii. After entering the merchant's website, the customer has three options to choose from. To continue the payment, the customer has to confirm the invoice details.

(1)

(2)

(3)

(4)

(5)

iii. After pressing "Confirm", the customer is required to input a PIN for authorization.

(6)

(7)

iv. After validation, the customer can select a card before making payment request. The default card is a credit card.

(8)

(9)

(10)

v. When the customer sends a payment request to Customer Bank, Customer Authentication is required. The customer has to input an account number and a password.

(11)

- vi. After successfully authenticating the customer and verifying the merchant's signature, the customer enters his or her bank's website and is given five choices to choose from. To continue, the customer's bank requires authorization from the customer.

(12)

(13)

(14)

(15)

(16)

(17)

(18)

- vii. After the customer has verified the message, he can use its non-repudiation certificate stored in Nokia SoftID to sign the message. When he retrieves the non-repudiation certificate, a valid PIN is required.

(19)

(20)

(21)

(22)

(23)

(24)

- viii. On pressing "Process", the customer's bank will process the customer's request by withdrawing the amount from the customer's account and sending all the data to the merchant's bank. The merchant's bank will then deposit the same amount to the merchant's account and send a message back to the customer bank and a payment notification to the merchant. The customer's bank then forwards the message to the customer.

(25)

• Process Time

The total process time to finish the transaction based on SWPP is given below. Process time is different according to different algorithms being used. When using 1024-bit for asymmetric and 128-bit for symmetric algorithms, the total process time is around 8 seconds, which is about twice longer than that by using 768-bit for asymmetric and 56-bit for symmetric algorithms. Considering the strong security requirement of banking service, we suggest that we use 1024-bit for asymmetric and 128-bit for symmetric algorithms in SWPP.

Key Exchange:	RSA		RSA-768	
Cipher Suites:	RC5+SHA	RC5_56+SHA_80	RC5+SHA	RC5_56+SHA_80
Process Time:	8450ms	5900ms	6374ms	4376ms

8.7 Conclusion

SWPP is implemented to balance the requirements of security and convenience. Convenience is achieved based on the WAP infrastructure and by using the customer

instead of the merchant to request the payment process. With the combination of WPKI, WTLS and WMLScript technology, security is guaranteed between the customer and the customer gateway, and also between the merchant and the customer. By acting as a bank content provider at the same time as a WAP gateway, Nokia Activ Server guarantees end-to-end security for customers. Although only WTLS Class 2 is supported on the gateway, customer authentication is implemented through the use of external authentication, in addition to the existing security infrastructure. To authorize a customer, a message from the customer's bank is signed by calling the function: `Crypto.signText()`, using a non-repudiation key stored in Nokia SoftID. In this function the server sends a message to the client and the client replies by sending a digitally signed copy of that message, thereby authenticating itself.

Java is considered to be the main development language, especially in the wireless environment, because of its small memory requirement and automatic garbage collection. Servlet is used to generate dynamic web content, which can be implemented easily.

However, there is still some more work to do. During the implementation, we used a self-signed WTLS certificate instead of requesting a server certificate from the CA because of its price. Using a WTLS certificate instead of an X.509 certificate can facilitate its processing in resource-constrained handsets. We did not consider certificate revocation here because it was for test purposes only.

Since Class 3 is not supported on the gateway and no open interface is supplied, a client certificate is not sent to the customer's bank to be authenticated. The external Customer Authentication used here cannot achieve the same security level as that defined in WTLS Class 3.

Here files have been used to store data such as the account and bank information. Future improvements might make use of LDAP technology to import user data or retrieve data from a database to make this implementation more practical.

In conclusion, SWPP is implemented to satisfy requirements of security and convenience, but some improvement is required.

Chapter 9

Comparison to WPP

In this chapter, a comparison between WPP and SWPP will be made. Since SWPP is a protocol redesigned for WPP, the security deficiency in WPP is well addressed.

9.1 WPP Deficiencies

Convenience and ease of use is the most vital component of any mobile payment solutions. Customers will not take advantage of mobile payment if the execution of a mobile transaction of any variety is not simple and straightforward. The transaction flow in WPP is different from MeT: the transaction flow is from the customer to the bank, then to the merchant. WPP can be considered as a good payment system as it is designed to balance the requirements of security and convenience. A dual signature is not required in WPP as messages do not pass through any intermediary.

However, convenience is useless without strong security. For a good payment system, there is always a tradeoff between convenience and security. WPP actually does not address security, as it assumes that the WTLS protocol will provide confidentiality and integrity of all messages exchanged between all participants of the protocol.

In the technical definition of WPP, the following table is given to show how the security requirements are addressed.

Security Requirements	Mechanisms to address requirements
Confidentiality	Confidentiality of all data transmitted is maintained by encrypting all messages between any two peers. (<i><u>Handled by WTLS</u></i>)
Customer/Merchant Authentication	Customer and merchant authentication will be carried out by using certificates of the form X.509v3 (signature certificate) <i><u>once version 3 of WTLS specification has been finalized and implemented.</u></i> In the meantime, this requirement should be considered mandatory for versions 1 and 2.
Data Integrity	A hash is computed and appended to original message before being sent to the destination. (<i><u>Handled by WTLS</u></i>). In addition, payment requests sent to the bank are digitally signed by the CA at the application layer. (<i><u>Not implemented</u></i>)
Non-repudiation	Merchant profiles transmitted from the merchant to the customer's bank will be digitally signed. In addition, the PIN used by the customer and the digitally signed payment request will fulfill both authorization and non-repudiation requirements. (<i><u>Partially implemented</u></i>)
Customer Authorization	Payment authorization by customers will be based on a PIN number. The PIN number will not be transmitted over the open network. It will simply control access to the Smart Card, which will store the client's certificates and other banking information. (<i><u>Partially Implemented</u></i>)

Fig 9-1 Security Requirements and Implementation in WPP

From fig 9-1 above, only non-repudiation and authorization is partially implemented in WPP. “As the WTLS package (add-on to the WAP Server) could not be downloaded free of charge, an assumption was made that the package would provide the confidentiality and integrity of all messages exchanged between all participants of the protocol.” [22] WTLS is actually just assumed, without being tested in WPP. WPP needs to be redesigned to fulfill its security requirements.

9.2 Security Characteristics of SWPP

SWPP is designed to help create a safe environment to support pervasive online mobile payments. It provides for centralized user authentication from limited points of entry. SWPP exploits the single sign-on for user-friendly implementation of credential-sharing across the services hosted by the Nokia Activ Server. SWPP relies on a set of industry standards for security, such as TLS/SSL and WTLS, to achieve the security objectives for the service domain. SWPP draws upon WAP for WTLS, WIM, WMLScript and WPKI to guarantee the security requirement presented in WPP. SWPP uses the proxy technology in conjunction with firewalls to define the boundary for the service domain.

In SWPP, we use Nokia Activ Server 2.1, Nokia Activ Security (128-bit support) and Nokia Toolkit 3.0 to simulate wireless environment. Security requirements are well addressed based on WTLS, WIM, WMLScript and WPKI.

Security Requirements	Mechanisms to address requirements
Confidentiality	SWPP utilizes a set of industry standard security technologies to implement the goal of confidentiality. Such technologies include the SSL and WTLS.
Customer/Merchant Authentication	<p>Both the bank server and the merchant server use the HTTP basic authentication process to authenticate the customer coming from a mobile source. The customer authentication function performed by the bank gateway is registered on the Nokia Activ Server. Both the clients and the gateway proxy for the clients must support HTTP.</p> <p>To authenticate the merchant to the bank server, a message from the merchant has to be signed then verified on the bank server.</p>
Data Integrity	SWPP provides data integrity using features such as message digest and certificates included in the SSL and WTLS security technologies.
Non-repudiation	<p>Since SWPP utilizes SSL and WTLS with certificates, the objectives of non-repudiation can be also implemented at the transaction level. SWPP provides non-repudiation using the WMLScript signText function, which authorizes the user at the same time.</p>
Customer Authorization	<p>There are two layers of authorization defined in SWPP.</p> <p>The access control in SWPP is implemented using HTTP proxy technology. The Nokia Activ Server may be used to implement the finer levels of access control.</p> <p>In an authorization by the user, the bank server presents a payment transaction to the customer which is confirmed by the customer. This is implemented using the WMLScript signText function</p>

Fig 9-2 Security Requirements and Implementation in SWPP

9.3 WPP and SWPP

The following section provides a whole comparison of WPP and SWPP protocols based on selected criteria.

	WPP	SWPP
Transaction Flow	Customer-Bank-Merchant	
Security Mechanisms	Not actually implemented	Implemented at both the application layer and WTLS class 2/3
Number of certificates used	None	One
Server Authentication	None	Provided by WTLS class 2
Customer Authentication	None	Using Plug-in Authentication Module
Data Integrity	Implemented	Message from the merchant is signed using its private key. Message from the customer to the bank gateway is signed based on the definition of WTLS Class 2.
Customer Authorization	None	Uses Access Control provided by Nokia Activ Server and signText function defined in WMLScript.
Number of certificates used	None	Two. One for signature, one for encryption.
User of Gateway	Nokia Server is used to simulate merchant site	Nokia Activ Server is used as a gateway between customer and merchant. It also acts as a gateway and bank server to the customer.
End to end Security	None	By putting gateway function and bank server together on Nokia Activ Server
User of WIM Cards	None	Yes
Use of Smart Cards	Yes	

Fig 9-3 Key Differences

Transaction Flow

The transaction flow in SWPP basically follows that of WPP. The first advantage to this is that the number of messages exchanged between the participating agents is reduced. The overall processing time of wireless payment transactions, and subsequently communication costs for mobile users, can also be reduced. As the direction of the transaction flow from customer to the customer's bank is changed without an intermediary, the consumer's concerns regarding the transmission of private information to the merchant are addressed.

Security

Security in WPP is based on an assumption of WTLS. Actually it is not well addressed at all. SWPP, on the other hand, provides a definite architecture based on security requirements. It is also implemented using the advanced Nokia Tools.

As SWPP does not route payment transaction data via the Merchant, security implementation becomes less onerous. A dual signature is not required since the customer's payment instructions are no longer sent to the merchant and therefore cannot be altered by the merchant. Following the definition in WPP, it is the banking information of the merchant (previously encrypted by the bank) that is sent to the customer and then forwarded to the customer's bank.

SWPP draws upon WTLS, WMLScript, WIM and WPKI to implement its security requirement. A WIM card is used to store a certificate and private key, which are password protected. The WIM card can also provide the textSign function, defined in WMLScript for Customer Authorization.

A Smart Card is used for storing encrypted banking information, and a Personal Identification Number (PIN) permits us to incorporate other types of payments including debit card payments. PINs are used strictly for authorizing access to the Smart Cards and are not transmitted over the network, thereby preserving the security of PINs.

End-to-end security is guaranteed by combining the bank server with the Nokia Activ Server Gateway.

Performance

The issue of performance is very important to mobile users since improved performance (lower processing time) results in reduced communication costs. According to the WPP definition, the performance of a protocol is dictated by transaction flow, bandwidth requirements (number and size of messages) and computational requirements.

For the transaction flow, WPP is expected to provide a faster processing time per transaction than SET due to the reduced number of messages exchanged between agents participating in the protocol.

As far as bandwidth requirements are concerned, the exchange of multiple certificates and data in the SET protocol requires considerably more bandwidth than WPP. By keeping the size of messages to a bare minimum, the bandwidth requirement is reduced.

With regard to the computational requirement, WPP is optimized to operate more efficiently over wireless networks. This includes a limited number of security mechanisms (e.g. a digital signature) implemented at the application layer, as well as the use of Smart Cards to address the constraints of wireless devices.

The performance of SWPP is quite similar to that of WPP. Both performance and security requirement can be affected to a great extent through the choice of algorithm.

However this cannot be tested with algorithms supported by Nokia Toolkit.

As SWPP is based on WAP, the performance is mainly based on WAP. The latency problem caused by WAP has hampered its success, but this can be settled by upgrading circuit-switched networks to a packet-switched, always-on network.

NOTE TO USERS

Page(s) not included in the original manuscript and are unavailable from the author or university. The manuscript was scanned as received.

108-113

This reproduction is the best copy available.

UMI[®]

References:

1. Alligata Server, "Guide to Mobile Internet Security," 2001, <http://www.kannel.3glab.org/download/kannel-wtls-snapshot/wtls.html>.
2. M. M. Anderson, "The Electronic Check Architecture," Sept. 29, 1998, <http://echeck.commerce.net/library/wp/ArchitectualOverview.pdf>.
3. Arcstream Solutions, "A Primer on Mobile Application Security," May 2001, <http://www.arcstreamolutions.com/uploadfile/whitepapers/security%20white%20paper%20062501.pdf>.
4. P. Ashley, H. Hinton, M. Vandenwauver, "Wired versus Wireless Security: The Internet, WAP and iMode for E-Commerce," <http://www.acsac.org/2001/papers/61.pdf>.
5. A. Bevan, "Payment systems in a mobile/virtual world," 2002, http://www.mcommcentral.com/documents.asp?d_ID=412.
6. BizReport, "Online Fraud Loss 19 Times Offline's," March 02, 2002, http://www.bizreport.com/article.php?art_id=3074&PHPSESSID=d99.
7. Certicom, "Wireless Public-Key Infrastructure," 2001, <http://www.certicom.com>.
8. Y. C. Cheong and C. L. Tan, "Payments In Mobile Commerce," http://www.cgey.com/tmn/nmi/3g/downloads/3g_payments.pdf.
9. M. Christinat, and M. Isler, "WTLS – The security layer in the WAP stack," 2000, http://www.keyon.ch/en/publications/infosec_wtls_keyon.pdf.
10. T. Covalla, "IBM WebSphere Everyplace Suite v1.1 Security White Paper," 2001, http://www-3.ibm.com/pvc/products/pdf/wes_security.pdf

11. N. Cravotta, "Securing the wireless World Wide Web," *EDN*, vol. 45, no17 pp.89-100, Aug. 2000.
12. CommerceNet, "What is eCheck,"
<http://echeck.commerce.net/overview/what.html>.
13. M. Cutts, "Secure Wireless Application Protocol (WAP) on the Enterprise, Ready or Not?" Sept. 2000, http://rr.sans.org/wireless/WAP_enterprise.php.
14. J. Fallon and G. Singh, "Mobile Internet Security," 2001,
<http://www.morganstanley.com/waprap/presentations/000601.pdf>.
15. S. Farrell. "Outlining Wireless Public Key Infrastructure," July 10, 2000,
<http://download.baltimore.com/download/pdf/BaltimoreTelepathyWPKIWhitepaper.pdf>.
16. C. Fennelly, "Security in wireless - Let security hound you," May 2001,
<http://www-106.ibm.com/developerworks/library/wi-sec.html?dwzone=wireless>.
17. Fundamo Ltd, "Security Review of the Fundamo Product Architecture document," Dec. 2000, <http://www.fundamo.com/pdfs/security.pdf>
18. K. Getgen, "Securing the air: 2001 -- A security odyssey," Jan. 2002, <http://www-106.ibm.com/developerworks/library/wi-sec2.html>.
19. K. Getgen, "Securing the air: Don't let your wireless LAN be a moving target," Nov. 2001, <http://www-106.ibm.com/developerworks/library/wi-sec1/?dwzone=wireless>
20. R. Ghani, "The Future of Wireless Banking Implementation of Wireless Banking and Financial Systems," March, 2001, http://www-3.ibm.com/pvc/tech/pdf/The_Future_of_Wireless_Banking.pdf.

21. B. Güçer, “Wireless Application Protocol and Security”, Aug. 2000,
<http://rr.sans.org/wireless/WAP3.php>.
22. J. Hall, S. Kilbank, M. Barbeau and E. Kranakis. “WPP: A Secure Payment Protocol For Supporting Credit- And Debit-Card Transactions Over Wireless Networks” presented at IEEE International Conference on Telecommunications, Bucharest, June 2001. Available on line at:
<http://www.scs.carleton.ca/~barbeau/Publications/2001/ICT/hallkilbank.pdf>.
23. T. Halonen, “Authentication and Authorization in Mobile Environment,” 2000,
<http://citeseer.nj.nec.com/400921.html>.
24. J. F. Hampe, P. M. C. Swatman, P. A. Swatman. “Mobile Electronic Commerce: Reintermediation in the Payment System” presented at 13th International Bled Electronic Commerce Conference Bled, Slovenia, pp. 693-706, June 19-21, 2000.
25. S. Holtmanns, “Security of Mobile Payment Solutions,” Nov. 2001,
<http://www.cs.kau.se/~simone/Security-of-mobile-payment-systems.pdf>.
26. R. Jewson. “E-payments: Credit Cards on the Internet,” Oct. 2001,
http://www.aconite.net/Epayments_Whitepaper.pdf.
27. N. C. Juul and N. Jørgensen, “Security Limitations in the WAP Architecture,”
<http://www.dnainland.fi/oops/wap.pdf>.
28. N. C. Juul and N. Jørgensen, “WAP may stumble over the gateway (security in wap-based mobile commerce” presented at International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet, Veljko Milutinovic, Ed., L'Aquila, page 8, Italy, 2001.

29. M. Krueger, "The Future of M-Payments - Business Options and Policy Issues," Aug. 2001, <http://epso.jrc.es/Docs/Backgrnd-2.pdf>.
30. J. Lautmann and U. Chandrasekar "Wireless Web Access Survey Report," 1999, <http://www.engr.sjsu.edu/gaojerry/course/236/Ref/wireless.ppt>.
31. A. Levi and Ç. K. Koç, "CONSEPP: CONvenient and Secure Electronic Payment Protocol Based on X9.59, " presented at The 17th Annual Computer Security Applications Conference, New Orleans, Louisiana, pp. 286-295, Dec. 10-14, 2001.
32. R. MacMillan, W. Farrell, A. Dean, "WAP Rap Summary, June 1," June, 2000, http://www.morganstanley.com/waprap/archive/reports/000601_summary.pdf.
33. J. Malmberg, "Mobile Security and Payment Methods Panel Discussion Background Material," April 2001, http://www.commerce.net/events/mobileExpo/presentations/3-25/J_Malmberg_Panel_Debate.ppt.
34. D. E. Mansy, "E-Commerce vs. M-Commerce," http://www.pcworldegypt.com/mcommerce_nov2000.htm.
35. J. Messham, "M-Commerce Security," 2000, [http://www.tdap.co.uk/uk/archive/billing/bill\(fml_0012\).html](http://www.tdap.co.uk/uk/archive/billing/bill(fml_0012).html).
36. Mobey Forum. "The Preferred Payment Architecture Technical Documentation," July 2001, <http://www.mobeyforum.org/PPATechnical.pdf>.
37. Mobile electronic Transactions. (2001). "MeT Account-Based Payment," <http://www.mobiletransaction.org/pdf/MeT-Account-Based-Payment-20010221.pdf>

38. Mobile electronic Transactions, "MeT PTD Security Requirements," Feb. 2001,
<http://www.mobiletransaction.org/pdf/MeT-PTDSecReq-20010221.pdf>
39. B. Morgan, "Thinking about wireless security," Nov. 2001,
http://searchsystemsmanagement.techtarget.com/tip/1,289483,sid20_gci779983,00.html.
40. I. Naick and R. Gupta. "Electronic payment processing for Web businesses," Feb. 2002, <http://www-106.ibm.com/developerworks/ibm/library/i-money/>.
41. S. Naik and A. Varde, "Security in WAP," Fall 2000,
<http://www.cs.purdue.edu/homes/fahmy/cs590f00/projects/naik.ppt>.
42. M. Nardone, "Wireless PKI Model," 2001,
<http://www.eurescom.de/~pub/seminars/past/2001/SecurityFraud/10-Nardone/sld001.htm>.
43. Nokia, "Secure Corporate WAP Services," Nov. 2000,
http://www.nokia.com/corporate/wap/pdf/secure_corporate_WAP_services_n.pdf.
44. M. O'Connor, "WAP-Enabled Banking and Broking: A Case Study,"
<http://www.perfectxml.com/Conf/Wrox/Files/morgantext.pdf>.
45. D. Olson, "Wireless Networking Implementation and Applications," Dec. 2001,
http://www.more.net/strat_tech/reports/80211b/80211b.pdf.
46. Phone.Com, "Understading Security on the Wireless Internet. How WAP security is enabling wireless e-commerce applications for today and tomorrow," Jan. 2000, Phone.Com, White Paper WPSE-R4-001.
47. D. Ponce and M. Soriano, "Secure Intelligent Broker for m-Commerce," in *Proceedings of the SSGRR 2001, International Conference on Advances in*

- Infrastructure for Electronic Business, Science, and Education on the Internet*, Veljko Milutinovic, Ed., L'Aquila, Italy, Aug. 2001, Scuola Superiore G. Reiss Romoli (SSGRR), CD-ROM.
48. M. J. Saarinen, "Attacks against the WAP WTLS protocol," 1999, <http://www.cc.jyu.fi/~mjos/wtls.pdf>.
49. J. H. Saltzer, D. P. Reed, D. D. Clark, "End-To-End Arguments in System Design," *ACM Transactions on Computer Systems*, vol. 2, pp. 277-288, Nov. 1986.
50. M. Schuba and K. Wrona, "Mobile chip electronic commerce: enabling credit card payment for mobile devices," *eBiz-2000 Specialist Conference*, June 2000, Singapore. Available on line at: http://www.medialabeurope.org/people/k-wrona/pub/mskw_eBiz00.pdf.
51. S. Sengodan, "On Security in Wireless and IP Networks," http://www.iworldindia.com/speakers/Senthil_Sengodan.ppt.
52. V. R. Sharma (1999). "Features of an Electronic Payment Mechanism," <http://www.iitd.ac.in/~rajosi/ecashrep/node2.html>.
53. S. Singhal, T. Bridgman, L. Suryanarayana, D. Mauney, J. Alvinen, D. Bevis et al. "WAP - the Wireless Application Protocol Writing Applications for the Mobile Internet," Oct. 2001, ISBN: 0-201-70311-4.
54. M. A. Sirbu, "Credits and Debits on the Internet," 1997, <http://www.gsia.cmu.edu/afs/andrew/gsia/45-871/Readings/Spectrum/netp.html>
55. X. Song, "Mobile Payment and Security," 2001, <http://www.tml.hut.fi/Studies/T-110.501/2001/papers/xing.song.pdf>.

56. K. Sung, "Analysis and Design of the Internet Based Payment System,"
URL: <http://iis.kaist.ac.kr/~sky/thesis/thesis.html>
57. C. Testman and L. Eriksson, "Mobile e-commerce," Aug. 2002,
http://www.ericsson.com.au/MobilityWorld/demo/download/MobileE_CommerceWhitePaper.pdf.
58. J. Törnroth, "A Server Based Wireless Wallet Enabling Secure Payment Transactions," M.S. thesis, Royal Institute of Technology, Stockholm, Sweden, 2000.
59. N. T. Trask and S. A. Jaweed. "Adapting public key infrastructures to the mobile environment." *BT Technol J*, vol 19, pp 76-80, July 2001.
60. B. Walsh, "Understanding Internet Payment Protocols," May 1999,
<http://www.networkcomputing.com/shared/printArticle?article=nc/1009/1009ws3full.html&pub=nwc>.
61. G. Z. Wang, X. H. Chen, D. X. Lin, "An Integrated WAP-Enabled Application Platform for M-Commerce," 2000, <http://www.cis.nctu.edu.tw/~gis88518/wap1/pdf/An%20Integrated%20WAP%20Platform%20and%20M-Commerce.pdf>.
62. WAP Forum, "WAP Architecture: Wireless Application Protocol Architecture Specification," 1998, <http://www.wapforum.org>.
63. WAP Forum, "WAP TLS Profile and Tunneling WAP-219-TLS: Wireless Application Protocol TLS Profile and Tunneling Specification," 2001,
<http://www.wapforum.org>.
64. WAP Forum, "WAP WTLS: Wireless Application Protocol Wireless Transport Layer Security Specification," 2000, <http://www.wapforum.org>.

65. WAP Forum, "Wireless Application Protocol Identity Module Specification (WIM) part: Security," 2000, <http://www.wapforum.org>.
66. WAP Forum, "WAP Architecture: Wireless Application Protocol Architecture Specification," 2001, <http://www.wapforum.org>.
67. WAP Forum, "Wireless Application Protocol Identity Module Specification (WIM) part: Security," 2001, <http://www.wapforum.org>.
68. WAP Forum, "WAP Transport Layer End-to-end Security: Wireless Application Protocol," 2001, <http://www.wapforum.org>.
69. WAP Forum, "WMLScript Crypto Library: Wireless Application Protocol WMLScript Crypto Library Specification," 2001, <http://www.wapforum.org>.
70. WAP Forum, "WMLScript Crypto Library: Wireless Application Protocol WMLScript Crypto Library Specification," 1999, <http://www.wapforum.org>.
71. L. Wirzenius and K. Marjola, "Kannel: Open Source WAP and SMS Gateway 1.0.3 Architecture and Design,"
<http://www.kannel.3glab.org/download/1.0.3/arch-1.0.3/arch.html>
72. L. Wirzenius and K. Marjola, "Kannel 1.0.3 User's Guide Open Source WAP and SMS gateway," <http://www.kannel.3glab.org/download/1.0.3/userguide-1.0.3/userguide.html>
73. A. Wood, "Wireless Security Meta-FAQ," 2001, <http://wap.z-y-g-o.com/wsec.html>.
74. K. Young, "WAP fever – have you got it?" *The banker*, vol.150, pp.20-26, April 2000.

75. J. Zyren and A. Petrick "IEEE 802.11 Tutorial,"

http://www.wirelessethernet.org/downloads/IEEE_80211_Primer.pdf.