Wireless Wallet

Yannis Labrou, Jonathan Agre, Lusheng Ji, Jesus Molina, Wei-lun Chen
Fujitsu Laboratories of America
8400 Baltimore Avenue, Suite 302
College Park, Maryland 20740-2496, USA
{yannis,jagre,lji,jmolina,samchen}@fla.fujitsu.com}

Abstract

We present a framework for agreements in pervasive environments called the Universal Pervasive Transaction Framework (UPTF) for parties transacting in wireless insecure environments using mobile devices. We discuss one type of such agreement with commercial interest, namely mobile payments from a payer to a payee. We have implemented two complete systems for purchasing and payment with mobile devices utilizing UPTF. The first introduces a special purpose, new mobile device called the Universal Pervasive Transaction Device (UPTD) and the second utilizes J2ME-capable mobile phones.

1. Introduction

The future ubiquitous computing environment will consist of mobile users with information appliances (mobile devices), such as cellular phones or Personal Digital Assistants (PDA's), that will be wirelessly communicating and interacting with the varied services and devices encountered at any particular moment and place. Many applications that operate in such environments have been proposed from the research community, but there has not yet been a strong market pull for any particular one. It is apparent that a crucial enabler for ubiquitous computing to emerge into the marketplace is the ability to safely conduct financial transactions using mobile devices in this form of environment.

Our wireless wallet application is based on a general framework, called the Universal Pervasive Transaction Framework (UPTF), a generic architecture and a new security protocol for conducting secure multi-party agreements, using mobile devices over a wireless transport network. The framework is designed to address several key aspects specific to the envisioned pervasive environments:

- mobile devices present certain characteristics due to their limited capabilities (computation power, communication bandwidth, battery capacity, small display, limited keyboard, etc).
- a typical user operating a mobile device may not be technically savvy and should not be overly burdened.
- the wireless transport network should be deemed as insecure at the network layer, primarily because any users engaging in such agreements must be able to use the links without requiring prior registration, for example at a public hotspot.

Although, the basic application motivating our approach is the concept of mobile payments, i.e., users paying for goods and services using a mobile device, the framework is universal in the sense that many forms of agreements can be accommodated. There have been many approaches and solutions proposed for the m-commerce problem, some of which have already been deployed, although typically outside of the US. A survey and taxonomy of many of these approaches is presented in [4]. The approach discussed in this paper, referred to as *wireless wallet*, is built on the UPTF and is a holistic approach that has important advantages and benefits in terms of simplicity of design, low cost, ease of deployment, ease of use and straight-forward business model when considering the emerging ubiquitous environments.

The wireless wallet is intended as a solution that accommodates three classes of financial transactions, in a unified and simple manner:

1. Peer-to-Peer payments: A consumer can directly make an agreed upon payment to another consumer using mobile devices.
2. Web Store-Front Payment: A consumer pays for goods or services offered by a retailer that has an internet presence. Typically, the user browses the retailer's web pages (using the mobile device) to

identify the good or service to be purchased and then conducts payment. Examples of this case would be paying for a book or purchasing movie tickets.

3. Physical Point-of-Sale (POS) purchase: A consumer pays a retailer at a check-out station using the mobile device, such as when making a payment at a "brick and mortar" store, such as a restaurant, convenience store, etc..

Most existing solutions in m-commerce have approached each of these classes as a distinct scenario, both technically and in terms of an underlying business model. The boundaries between these categories are malleable and their common properties can be exploited using the transaction agreement point of view. The second category above, most resembles current practice. For web pages that are specially prepared for mobile devices, such as those that are WAP-enabled, one could use the mobile device to make a purchase as it is normally done in e-commerce transactions using a web browser on a personal computer. But, since payment typically requires logging in and typing a username and password, we believe that this approach is impractical and inefficient on a mobile device, even if the transaction uses WAP and has occurred through a secured network link such as through https. The physical POS case is typically the most complex to deploy from a business point of view because it frequently involves integration with the back-end store systems and some form of binding between the payer and the physical goods purchased.

This paper presents the evolution of the wireless wallet concept from the technical, business model and architectural point of view. The core technical idea, i.e., a computationally lightweight protocol for two party (easily extensible to multi-party) agreements between consumers operating mobile devices is described and its application to financial payment transactions is shown via two implemented prototype systems.

2. Requirements and summary of our approach

Delivering a commercially viable solution requires meeting a combination of technical and business requirements. These will be briefly outlined; however, they are intentionally mixed because it is felt that they are inseparable. This is the intent of our holistic approach:

- All security methods for guaranteeing the authorization, authenticity, and accountability (AAA) of the transaction and the usual desirable properties (privacy, integrity, tamper-resistance, non-repudiation, protection against various types of attacks), should be delivered to the end user in a simple form, such as with a four digit Personal Identification Number (PIN). Even though a biometric method would be a nice security element, such features are not yet widely available on mobile devices and should not be required.
- Management of security should also be simple for the end-user, which means that the user should not have to understand or do anything other than remembering, but not sharing, the PIN. For example, the handling of multiple security certificates from many vendors is confusing, at best, and can be dangerous if left up to the users.
- The security solution should be computationally lightweight so that it does not introduce noticeable delay to the transaction using existing mobile devices.
- The solution should be deployable with existing mobile devices such as cell phones, PDA's and other portable communication devices and operate over non-secure links such as wireless local area networks (WLAN's) and cellular networks.
- Since mobile devices can be easily misplaced, lost or stolen, security should not be compromised in the case of such an event. In other words, the critical financial account, personal identification or other security information should be adequately protected if stored on the device, or not stored at all.
- The participants in an agreement should be able to authenticate the other parties in the transaction without exposing their financial accounts.
- Allow a user to choose to make payments from one of multiple accounts.
- Deployment of the solution should minimize the costs to deliver a transaction verification service that meets the desired levels of protection from fraud while maintaining the user's privacy.

Our UPTF and the wireless wallet application attempts to address these issues. We briefly summarize our approach. The payment transaction (transfer of funds) is thought of as an agreement between a payer and payee (e.g., a consumer and a merchant). In a payment transaction, the consumer and merchant will first exchange information about the details of a transaction. Then both parties submit their own views of the transaction (i.e., the request for payment) to a trusted third party called the Secure Transaction Server (STS). The trusted STS will process the request for payments and cause the transfer of funds, typically through another financial institution

or online payment service. The entire process is accomplished in a highly reliable and secure fashion, effectively preventing fraudulent transactions, guaranteeing the authenticity of the transactions and maintaining sufficient records to provide non-reputable evidence, if needed. In general, the environment is not a trusted one (primarily due to the wireless links) and the security properties are designed to operate in spite of this fact. In addition, the mobility aspects of the information appliances and devices necessarily limit the computational, storage, communication and power/battery capabilities of the devices and appliances and the algorithms take this into account. The wireless wallet scheme does not store any personal identifying information on the mobile device and requires that a user enter a short PIN prior to authorizing a payment. If the device is lost or stolen, it can not be used without the PIN. Since, every transaction must pass through the STS, it is straight forward to detect attempts to "guess" a PIN. Further, a simple mechanism for obtaining new PINs is incorporated in the system so that PINs are easily replaced. The details of the framework on which the wireless wallet is based are provided in the following section.

3. The Universal Pervasive Transaction Framework

UPTF Framework

The Universal Pervasive Transaction Framework (UPTF) defines a system architecture and a communication security protocol, called the Secure Agreement Submission (SAS) protocol. Essentially the UPTF offers a vessel, which is able to securely carry the individual views of a transaction agreement from each party involved in the transaction to a trusted third party for verification, using a communication network which may consist of insecure segments such as wireless LANs or cellular links. When used for financial applications such as the wireless wallet, the transaction parties are a customer and a merchant (or payer and payee), and a typical example of an "agreement" may read: "Party A will pay Party B $X for item Y."

The UPTF SAS protocol will encrypt the messages using a symmetric, shared-secret-key approach where the secret key is known only to an individual party and the trusted third party. The SAS insures that the authenticity of the parties is verified and during delivery, the privacy of the information is preserved, even when the parties distrust each other and the messages from one party may be forwarded by the other to the third party for verification. The UPTF also provides the mechanism for the trusted third party to verify that the independent views of the agreement are consistent with each other.

After the agreement data is extracted from the views received from the parties and the data is verified by the trusted third party, further actions may need to be taken to actually execute the agreement. This is realized, for example, by the trusted third party interacting with the financial institutions associated with the payer and the payee to cause the transfer of the specified funds between the customer and the merchant, in the case of a financial transaction.

Architecture

The UPTF system architecture for a financial transaction type of agreement is shown in Figure 1 and includes: a Payer operated device, a Payee operated device, a Secure Transaction Server (STS), a number of financial institutions and several communication channels. The payer operates a mobile computing device which interacts with the payee to determine the details of a purchase transaction and executes the Secure Agreement Submission (SAS) protocol and its corresponding security operations. The mobile device supports the wireless communication capability necessary for discovering the payees, communicating with the payee and communicating with the STS as necessary. It also has a user interface for interacting with the payee through some common application and the STS as needed. The payee also operates a device and is responsible for interacting with the payer, executing the SAS protocol and its corresponding security operations and interacting with the STS.

The STS is the backend verification server on which both the payer and the payee have registered and provided identifying account information that is maintained in a secure STS database. The secret information used for encrypting the messages to/from each payer and payee are also stored in this DB. The STS receives the independently generated views from both the payer and the payee regarding the transaction conducted between them. The STS is able to decode both of the views using information from the messages and the information stored in the STS Database. Following successful decoding, the STS verifies that the view messages are original, authentic, involve the intended payer and payee and that the information fields in the agreement views are consistent with each other. The STS will maintain a log of all messaging activity for non-repudiation purposes.

In Figure 1, the generic set of communication channels are explicitly indicated. Channel A (Ch A)

represents the link between the payer and the payee. This link is used to negotiate the details of the payment. This aspect is application dependent and is not considered to be part of the UPTF framework. Ch A is typically a wireless channel to accommodate mobility and is not secure. Channels B and C, are the links between the Payer and the STS and the Payee and the STS, respectively. In most situations these are not direct links, but involve communicating through the Internet. In general, these are insecure channels. Channel D, from the STS to the Financial Institutions is a different type of channel and is assumed to be a highly secure communication path. In addition, STS itself is assumed to a housed in a protected facility so that its database is physically secure and inaccessible from the network.

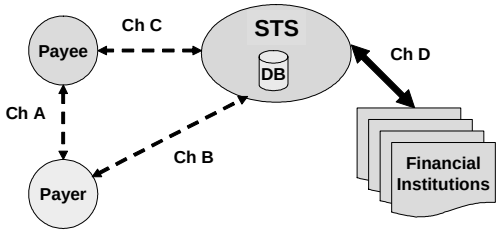

Figure 1 UPTF System Architecture

The steps involved in the typical transaction of Figure 1 are described. The payer initiates the SAS protocol through an explicit action and enters the PIN. This allows the payer to generate a view of the transaction and to encrypt this with its private key and then to send it as a message sent to the STS. Similarly, the payee (operator) enters its PIN and generates it's own view of the transaction, encrypts the view with its private key and then sends its view to the STS. The STS receives both encrypted views and verifies the views through a successful decryption. The STS then uses the secure back-channels to interact with the financial institutions of the payer and the payee for transferring the funds. The STS sends receipts (or failure notices) back to the payer and the payee to complete the transaction. The response messages are also encrypted by the STS for each intended destination.

The examples described in later sections are particular instances of this architecture. In particular, the physical POS deployment represents a common variation of this scheme. For this situation, the Payer is a customer device, the Payee is the merchant operated device. The merchant operated device is located at a fixed site and may be a more powerful computer and provide additional network services, such as an Internet connection. Ch B is not used, but rather the Payer messages are forwarded through the Merchant using Ch C, which can be a secure internet connection to the STS. The properties of the protocol prevent the Merchant from gathering personal information from the customer. Further details of the encryption procedures are described in the next section.

Secure Agreement Submission (SAS) Protocol

The SAS protocol is used for encrypting and submitting views of the desired transactions. The message structure and encryption mechanism of SAS are designed to provide the desired security properties in an insecure pervasive communication environment as discussed in Section 2.

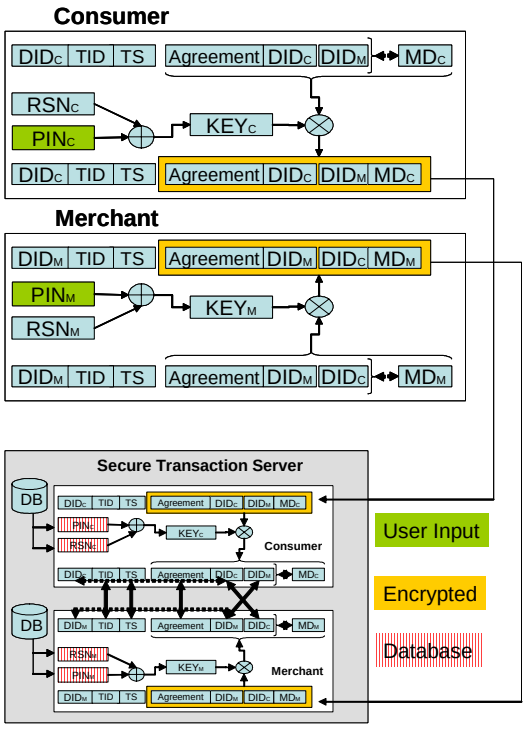

Figure 2 Consumer and Merchant message encryption and Secure Transaction Server Processing

The underlying SAS algorithms are well-suited for a system using low-cost user devices, which have limited computing resources, while minimizing the complexity for the user. In addition, some of the information necessary to use the SAS, such as the PIN, is not stored on the mobile device, so that if it is lost or stolen, it can not be used.

The internal structure and the generation process of a view message are shown in the above figure. Since the views from the payer and the payee are

symmetrical, we will only describe the payer's view. The symbols used in the figured are explained below:

- DID_C: device ID, a unique identifier for the Payer's device (the consumer, or the payment source).
- DID_M: device ID, a unique identifier for the Payee's device (the merchant, or the payment destination).
- RSN: random sequence number.
- TS: timestamp.
- TID: transaction ID, a unique identification number assigned to an agreement.
- MD: message digest
- PIN: Personal identification number, a user input secret alphanumeric string.

As shown in the Figure 2, a view includes a cipher text part (or encrypted part) and a plaintext part. Using the consumer view as example, the plaintext part includes the TID, the DID_C of the payee generating the view, and the local TS of that device. The TS is used to prevent transaction replay. The encrypted part includes two critical fields: the agreement data, and the DID_C of the payee's device involved in the agreement. The DID_m is the minimum necessary reference field in order to provide the desired verification properties of the SAS protocol.

First, the DID_C and the TS obtained from the device's local clock or provided as a part of the agreement data, are input to the device's pseudorandom number generator to generate a time-dependent RSN. The parameters of the generator are particular to each device. The encryption key K is then generated from the RSN and PIN. Firstly the RSN and PIN are combined using a function F and then a hash function H is applied to the result (typically a string):

$$K = H (F ((PIN, RSN))$$

A message digest function is applied to the agreement data, the DID_M, and DID_C to generate a MD of the view. The MD further strengthens the security by ensuring that no other party has tampered with or modified the contents of the view in any way. The encryption algorithm with the encryption key K is then applied to the MD, the agreement data, the DID_C, and the DID_M to generate the cipher text part of the view. For further protection, the SAS protocol uses random message padding in order to further prevent "known-text" attacks. In a version of the prototype wireless wallet, we used the AES for encryption, an HMAC-based scheme for random number generation, and the SHA1 for the hash function. The Payee view is generated in the same fashion.

The STS has sufficient prior knowledge of the functions and specific parameters used by each device in the encryption process, so that when combined with the plaintext portions of a message it is possible to decrypt the message by reversing the above process. From the plaintext part of the view, the STS recovers the DID_C and TS, which are used to look-up the customers PIN and other parameters of the RSN generator that were stored in the STS database. These are used to compute the RSN. The encryption key K can then be computed using the same method with which the customer UPTD generates the encryption key. The cipher text part of the view message is then decoded.

After all fields of the payer view are acquired, the STS locates the payee's view for the same transaction, using the DID_M and TID included in the previously decoded payer view. After going through a similar decryption process, the decoded fields of the agreement data of the payee are compared with the corresponding fields from the payer view. If all the corresponding fields match, the received views are considered verified. Further processing is then carried out and external executions are triggered as necessary.

Any responses from the STS to the payer or payee are encrypted by the STS using the same encryption methods and using the parameters for the destination device and the TS of the original transaction. Only the intended recipient can decrypt the response message, insuring privacy protection and authentication of the STS.

A SAS-based transaction requires a device, which provides device-specific parameters that determine a device-specific and time-specific key and an operator for the device who provides a fixed PIN that is only known to the STS and the operator. The combination of the two is required for an encrypted transaction request that can be validated by the STS. Intercepting one (or more) transaction message and successfully decrypting it would not be sufficient for purposes of inferring either the PIN, or the device-specific parameters employed in the key generation process. Moreover, a single, time-dependant key is not re-usable because of the pair-wise agreement notion of transactions processed by the STS .

4. Systems implementing UPTF

We next describe two complete, implemented systems that rely on UPTF to enable purchasing and payment using a mobile device. In the first case we designed and built a new, special purpose, mobile device and in the latter case we implemented a system where mobile phones are used for purchasing and payment.

A Universal Pervasive Transaction Device

The universal pervasive transactions device (UPTD) was designed to be a wallet-sized device able to detect UPTF-enabled physical points of sale, to wirelessly connect to them using an 802.11b WLAN radio and to allow their owners to make purchases and or payments with them. For example, the UPTD could be used to place an order at a restaurant and subsequently make payment. The user experience is as follows: (1) the consumer, upon approaching a service spot (which could be a movie theater ticket booth, a gas station, etc), activates (powers up) the device, which automatically searches for merchants; (2) the device displays a listing of the available merchant offered services; (3) the consumer selects the service (e.g., ordering a meal, or payment at checkout station #12) with a simple keypad; (4) once a purchase amount is determined, the consumer presses a designated payment button on her device, which begins the payment stage and results in retrieval of the purchase order from the merchant (the agreement). It is important that the payment stage is explicitly initiated by the consumer, so that the consumer can not be spoofed into typing her PIN into an attacker-served page. In fact, in our implementation, pressing the payment button results in termination of the browsing application and the launching of a new application for user entry (albeit transparent to the user). As long as the user never entered her PIN without first pressing the payment button, we prevent hijacking of the PIN; following a visual inspection by the consumer she is requested to enter her security PIN (optionally selecting which account to use for payment[1]); (5) the consumer receives on the UPTD confirmation and a receipt if the transaction was successful (the merchant's equipment also receives a notification of the successful transaction) .

The architecture to support the described usage scenario can be seen in Figure 3 and a prototype implementation has been demonstrated. The merchant's physical POS is set up as a hotspot and the consumer and merchant will interact wirelessly using a WLAN. The merchant consists of one or more access points (APs) connected to a small computer (we use a laptop) that runs a lightweight DHCP server, a Lite HTTP server and the retail application (specific to the business of the merchant). The retail application implements the virtual store front (for ordering goods or retrieving the payment amount and is accessible through the web server) and the UPTF-related functions for effecting purchasing. The access points work in bridge mode, so any connection to them will be directly forwarded to the merchant's page of available services. Any connection by the consumer device is automatically redirected to the internal web server of the POS and then to the appropriate module. The merchant's equipment (merchant service spot) is also connected to the internet for communication with the remotely located STS. The STS has a secure connection to the Financial Institution for the actual transfer of funds.

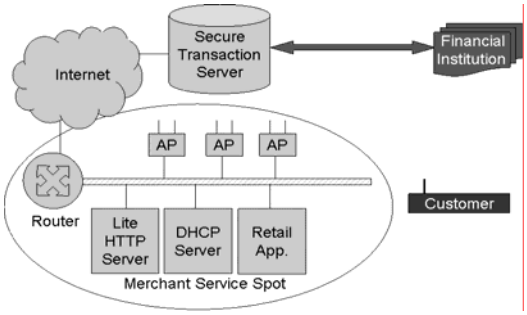

Figure 3 System architecture including a Physical Point of Sale and a customer-operated UPTD

A prototype UPTD was built based on an In-Hand [8] Single-Board Computer. The board featured an Intel Strong ARM SA-1110 running at 206 MhZ, with 16MB of flash memory and 32MB of SDRAM. The board also offered a Compact Flash slot which was used to attach a Symbol24 802.11b wireless card, as well as 10 inboard I/O pins to create a simple 5-button interface. The power supply could be driven either by a rechargeable 1600 mAH Lithium-Ion battery or a 5V DC charger. When the charger and the battery were connected at the same time, the circuit closed and the battery was charged. The power consumption could be reduced by down-clocking the processor, to a minimum of 59Hz and still accomplish the transactions. The board also provided an interface for a Seiko LCD display. The size of the board was 85 mm(L) x 55 mm (W), with a thickness of 9.5mm; dimensions very similar to a traditional wallet. With the casing, and the CF WLAN card, the total thickness of the UPTD was 15mm, mostly due to the design of the CF slot that added a few millimeters in total height and because of the prototype casing we manufactured. If the CF WLAN socket was custom made or, if the WLAN was integrated on the board and with a custom casing, we could have achieved a thickness between 10-11 mm, even with the use of off-the-self components. NEC recently announced a credit card form factor mobile phone with a thickness of 8.6mm, which was followed by a

[1] We often used PayPal as the payment service. Assuming that both merchant and consumer are PayPal users, we would actually transfer funds from payer to payee.

version that also includes a camera, at exactly the same form factor. We have no doubt that a production version of the UPTD could achieve a thickness of 8-9 mm, resulting in a pocket-sized device.

Figure 4 The prototype UPTD without casing and buttons; clockwise from the left there is 1) the battery, 2) the display, 3) the SBC and CF slot, 4) a credit card that indicates the UPTD dimensions.

The software on the UPTD delivered all the functionality for the secure payment transaction described earlier in this section. Our software used a version of Arm-Linux for the InHand SBC. The main reason for this choice was design flexibility, as our requirements were evolving; for example, access point discovery was not yet provided by other embedded operating systems, like Windows CE and we were considering biometric authentication in a version of the device. Since the user interface was supposed to be simple, we decided to use the Lynx text-based browser for purchasing and to create a custom user interface built using the *ncurses* library for the payment stage; interaction between the UPTD and the MTS during the payment stage was done over sockets. From the user's point of view, discovery of merchants, purchasing and payment appeared as a single application that started running whenever the device was activated or reset. For the core UPTF-related functions we used the *OpenSSL* library in order to support the different cryptographic primitives. To glue all the applications and to create the illusion of a single seamless user application, a custom boot routine was developed and several changes to the lynx browser were required. The support for access point scanning was added to the open source drivers for the Symbol WLAN Card.

The UPTD delivered the user experience we described earlier in a fast and intuitive manner. The implicit localization delivered by the WLAN allowed the device to discover the nearby merchants. The speed and its simple and unified interface were effective and with some practice, performing a transaction at a cashier-like POS could be accomplished in less than 12 seconds of user time. (Compare this to a typical cash transaction involving getting change!). We assumed that a consumer would purchase (or be offered) a UPTD that would be activated (associated with a payment account) on-line or over the phone. The consumer would obtain the PIN during the activation process. Merchants would purchase a small computing box to run the retail and purchasing applications. The box would also include a software WLAN AP, a display and a numerical keyboard for entering payment amounts (for the POS case). The box would operate stand alone or it would connect to the in-store back-end system (or to a hosted store on the network) for merchants with more substantial infrastructure.

A Wireless Wallet for mobile phones

While the prototype UPTD was met with excitement by attendees of our demonstrations who felt that production of a thinner version was feasible, the universal reaction was that in order to achieve commercial success, the first step would be to make the same functionality available on a mobile phone.

Delivering exactly the same capabilities on a mobile phone was infeasible because mobile phones are "closed platforms" controlled by the carriers. We decided that some adjustments were necessary. These adjustments were not targeted at the fundamental protocol, which was preserved, but at the development platform, the model for software distribution and the overall architecture. We next discuss these, although we would like to note first that we expect mobile phones in the very near future to offer WLAN[2], which would make it possible for the model described in the earlier sections to co-exist with the one described here.

The wireless wallet application is implemented as a J2ME application than can be downloaded and executed on the mobile phone and enables users to make purchases and payments leveraging the UPTF. Most of the phones offered by carriers in the US are J2ME enabled[3] and web-enabled. It is felt that the widespread availability of J2ME on mobiles enables new business models because it loosens the control of

[2] A few small manufacturers have offered WLAN-enabled mobile phones and both Motorola and Nokia have made informal announcements of plans to offer such phones in 2004.

[3] J2ME-enabled phones comprise anywhere from 50% to 100% of currently available phones in the US (except for Verizon that offers BREW rather than J2ME).

the carriers with respect to the delivery of content and services to mobile users. Currently, most carriers control the delivery of content and services to the user because they control the user interface (web browser) for delivering them and (especially in the US) they charge consumers for its use because they act as the intermediaries (for content and service delivery). With J2ME, anyone can develop downloadable J2ME applications for custom content or service delivery to the mobile. This is the case of the wireless wallet application, which offers the service of secure payments using mobile phones.

The J2ME wireless wallet application is a relatively small application (less than 90 Kbytes) which combines the functionality of a web browser (a cHTML web browser) and that of the purchasing application that implements the UPTF framework and security protocol. It allows mobile users to enable or disable the payment functionality and to store receipts of purchase. The wireless wallet method can be offered by any retailer or payment service in order for consumers to make payments with their mobile phone. It is a completely software solution to the problem of secure payments using a mobile device. If the provider of the wireless wallet is a web storefront retailer, such as Fandango for movie tickets, the wireless wallet application can be used to make payments with any of the accounts registered with the retailer. At the conclusion of payment we provide the mobile user with a receipt in the form of a barcode image (that can be scanned), accompanied by a receipt number; either (barcode or number) can be used to gain entrance, for example, at the facility that the service is associated with (Fandango allows consumers to purchase movie theater tickets over the web and retrieve them at movie theater-placed equipment).

If the provider is an Online Payment Service (such as PayPal) the wireless wallet can be used to make payments to other PayPal users (peer-to-peer) or to "brick and mortar" retailers (POS)[4] that accept payment with this payment service. In the latter case, the merchant can use a merchant's version of the wireless wallet client to specify the payment amount; the problem of identifying what is being paid for by the consumer (a lesser problem in the version described in section 0 because of the immediate localization of the WLAN and the overall architecture) is addressed by having the consumer enter a receipt number (say printed in the receipt of the restaurant where the meal is paid for) or the phone number of the merchant on their wireless wallet application. Numerous variations of this idea exist; the goal is always to bind a virtual shopping cart that represents physical goods to a specific consumer.

As mentioned, we expect either retailers or payment services to be the providers of the wireless wallet software. We next discuss the software download and activation (see also Figure 5):

1. After the user logs into the Provider's secure web site using a personal computer, the option of enabling one's mobile phone for payments is offered and the user is re-directed to a page where one is asked for the phone number of the mobile phone to be used for such payments.
2. The Provider generates a UPTF_ID (random number) for the already captured username and password and sends to the STS the UPTF_ID and mobile phone number (this way the Provider does not have to share with the operator of the STS their real account information).
3. The STS then creates a new executable (with "fresh" initialization parameters per the UPTF requirements specifically for this device), a link to download this executable, an activation code and a PIN; the STS then sends all this information to the Provider.
4. The Provider receives the PIN and displays it to the user, along with the one time activation code.
5. The STS sends a Short Message Service (SMS) message to the previously entered mobile's phone number containing the download link
6. The mobile user downloads the wireless wallet software (the link can be easily followed directly from the SMS text itself using a button push) and activates it using the activation code.

Alternatively, the PIN and activation code can be communicated to the user through other channels than the provider. The activation code is a number (for easier user entry) and it is used as a one time password that encrypts the UPTF-related initialization parameters of the downloaded software, so that if a third party (attacker) intercepts the software while in transit (in step 6), the attacker can not have access to the device-specific initialization parameters. Even if the retrieved these parameters, she would still need the PIN to "replicate" a wireless wallet; one way to get this PIN would be to intercept an actual transaction from that mobile and armed with the initialization-specific parameters do a brute force attack.

[4] The virtual store of the merchant is hosted on the web in this case.

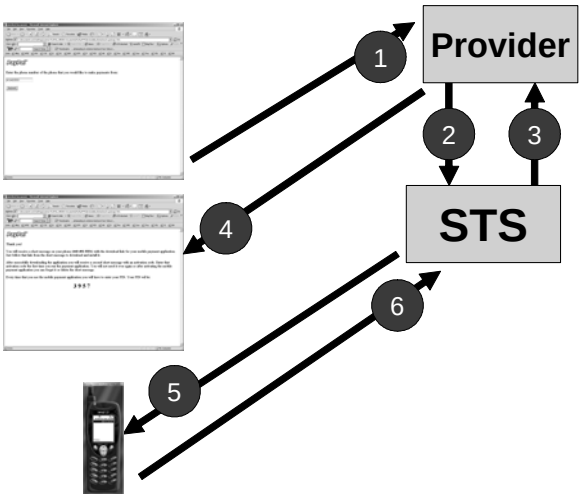

Figure 5 Steps of wireless wallet download and activation for a mobile phone user.

After these steps, the wireless wallet is ready for use; purchases can be paid for with any of the accounts registered with the wireless wallet provider. This scheme for distributing the software ensures a secure distribution of the software, on request, and a safe dissemination of the PIN. We have implemented this distribution model for downloading and installing the J2ME executable on the mobile phone. Operation of the software is very easy and transaction time largely depends on the speed of the carrier's network. We have fully tested the application on an emulator and on two mobile phones from two different US providers and we have found transaction time to vary between 30 and 45 seconds, the differences due to carrier's network-related delays. The traffic generated by the payment application (following the determination of what is purchased) is less than 1KByte (send/receive) for each transaction, as each message is smaller than 256 bytes. Communication between the mobile and the STS during payment was routed either through the merchant or the STS (depending on the type of financial transaction) and was carried over HTTP. Even though the J2ME executable (MIDP 1.0) should run on any J2ME phone in the market, we have not done exhaustive testing.

5. Evaluation

We stated earlier in this paper that our goal was to provide a complete deployable solution. A detailed comparison with other approaches that intend to provide a complete solution is beyond the scope of this paper, due to the enormous number of solutions that attempt to capture the mobile payments market and the need to compare across all aspects of the solution (technical, business and usability considerations). We have, to a large extent, done this kind of analysis in reference technical report [4] and we encourage the reader to also look at the ePSO database on e-payment systems [1].

The suitability of the UPTF as a framework for mobile payments has been demonstrated by its flexibility for creating a unified architecture for all three types of payments. We argued against using a PKI-based solution on grounds of computational performance[5] and complexity of the user experience. There are many different ways PKI can be used for mobile payments. Perhaps the most comprehensive approach towards a PKI-based solution for mobile payments is that of the MeT (Mobile electronic Transactions) forum [2]. The idea is that the user signs a transaction (a purchase order) with a certificate that authenticates the identity of the user (it is unclear whether each user has a single such certificate or a variety of them, each for every eligible account) [5]. Since these certificates are stored on the device, the certificate store needs to be protected and "unlocked" on a per use basis, in accordance with [6,7]. If the certificate storage is implemented in software the key used to unlock the storage should be of sufficient length to protect this storage, or it can be instead implemented in hardware which would require the phone to be designed for this purpose. Such an approach requires an infrastructure for dissemination of certificates (including revocation), possibly specialized mobile phones and possibly some basic understanding by the user of certificates and their usage. A major counter-argument against this approach is that it could be also used in the desktop environment for making payments while conducting electronic commerce, without having to enter account information for each purchase. Even though a desktop or a laptop does not impose computational constraints for PKI, we have yet to see a solution where users type a 4 digit PIN (or some longer key) and then get to select any of their registered accounts, even though internet users have been conducting e-commerce for almost a decade now. We do not claim that there is a single reason for the unavailability of such a scheme for desktop-based e-commerce (usability, real security, perceived security and business model considerations are all contributing factors) but the simple fact it has not happened does not bode well for the deployment of a similar model with respect to mobile payments.

Our approach can be thought of as a server-side wallet to which access is controlled through a four digit PIN, with the UPTF notions of multi-party agreements and time-of-transaction dependent key

[5] Our timing tests were consistent with [3].

generation combining to provide the (usual) expected security properties. The combined solution does not require any storage of personal or account information data on the mobile device; it does not impose special hardware requirements and "reduces" security to a 4-digit PIN which is a major convenience for the user. We have found the solution to be computationally fast; on mobile phones the key generation and encryption (or decryption) takes 100 ms on the fastest of the two phones we use (500 ms on the slowest) using J2ME (probably not the most efficient implementation of cryptographic primitives), for 160-bit AES encryption for each message. Thus, the security-related computational time is non-noticeable with respect to the transaction time.

6. In Conclusion

We have described the Universal Pervasive Transaction Framework and two systems that utilize it to support mobile payments over insecure, possibly wireless networks. The wireless wallet will provide a convenient alternative to using a wallet full of cash and credit cards. It provides simple, uniform access to a set of accounts (credit cards and bank accounts) and the account to be used can be easily selected at time of payment. The time required to make a payment using the wireless wallet can be significantly less than making a payment with cash or by the traditional payment using a credit card. For example, consider the traditional process of paying with a credit card in a restaurant and contrast this with the few simple steps needed to pay by wireless wallet in a mobile phone. A wireless wallet also provides advantages to financial institutions by reducing losses due to lost or stolen cards. Lastly, it enables one to safely make or receive payments to/from individuals or merchants from any enabled location.

We expect the wireless wallet to eventually incorporate all the functions provided by physical wallets, such as identification, club cards, bank cards and credit cards, making all of that securely available for a variety of transactions (including payments) through a few key strokes. At this stage we are in the process of commercialization as we are seeking partners that would be interested in becoming providers of the wireless wallet on mobile phones.

7. References

1. http://epso.jrc.es/paysys.html, ePayment Systems Observatory
2. http://www.mobiletransaction.org/, Mobile Electronic Transactions Forum
3. M. Brown, D. Cheung, D. Hankerson, J. Hernandez, M. Kirkup, and A. Menezes., *PGP in constrained wireless devices*, in The 9th USENIX Security Symposium, USENIX, August 2000
4. Yannis Labrou, Lusheng Ji, Jonathan Agre, *A survey of the Universal Pervasive Transaction Framework (UPTF) and related systems*, Fujitsu Laboratories of America, Technical Memo, FLA- PCRTM03-03
5. *MeT Personal Trusted Device Definition*. V2.0, MeT
6. *PKCS #11 v2.11 Amendment 1* , RSA Laboratories, August 2002
7. *PKCS #11 v2.11 Rev.1: Cryptographic Token Interface*, RSA Laboratories, November 2001
8. http://www.inhandelectronics.com/, InHand Electronics