CERIAS Tech Report 2008-10

Verification of Receipts from M-commerce Transactions on NFC Cellular Phones

by Jungha Woo, Abhilasha Bhagav-Spantzel, Anna Cinzia Squicciarini, Elisa Bertino
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

Verification of Receipts from M-Commerce Transactions on NFC Cellular Phones

Jungha Woo*, Abhilasha Bhagav-Spantzel[†], Anna Cinzia Squicciarini[‡], Elisa Bertino*

*Department of Computer Science
Purdue University
West Lafayette, IN
{wooj, bertino}@cs.purdue.edu

[†]Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA
abhilasha.bhargav-spantzel@intel.com

[‡]College of Information
Sciences and Technology
The Penn State University
University Park, PA
asquicciarini@ist.psu.edu

***Abstract*—A main challenge in mobile commerce is to make it possible for users to manage their transaction histories from both online e-commerce transactions and in-person transactions. Such histories are typically useful to build credit or to establish trust based on past transactions. In this paper we propose an approach to manage electronic receipts on cellular devices by assuring their secure and privacy-preserving usage. We provide a comprehensive notion of transactions history including both on-line transaction and in-person transactions. We apply cryptographic protocols, such as secret sharing and zero knowledge proofs, in a potentially vulnerable and constrained setting. Specifically, our approach supports flexible strategies based on Shamir's secret sharing to cater to different user requirements and architectural constraints. In addition, aggregate zero knowledge proofs are used to efficiently support proofs of various receipt attributes. We have implemented the system on Nokia NFC cellular phones and report in the paper performance evaluation results.**

I. Introduction

Mobile commerce (m-commerce) allows users to conduct business and service transactions over portable wireless devices. One main challenge in m-commerce transactions is providing users with the capability to prove their chosen past transactions to establish trust with the service providers (SPs for brevity). Such transaction history is used [18] to establish trust or provide history-based services. For example, users can prove their successful e-commerce transactions and qualify for rebates or discounts based on them. Because of the extensive commerce activities carried out by users both online and in-person, it is in the interest of the users be able to use information from their transaction history in various types of m-commerce transaction.

Ideally, transaction history should be easily accessible from any location, in a compact form which is acceptable by any SP. An approach to address such requirements is to support electronic receipts encoding relevant information about transaction history, and use such receipts from mobile devices, such as cellular phones.

Supporting such receipts in mobile environments is very challenging. First, it is not trivial to ensure the security and privacy of the receipts. By using technologies such as Bluetooth or RFIDs [1], a SP could retrieve information from the phones without user consent, which could result in privacy breaches. A second issue is related to the storage and computational constraints of most cellular phones [1] which require efficient receipt protocols. Furthermore, in order to maintain a comprehensive history, the transaction histories from online and in-person transactions should be integrated.

To support mobile history based transactions and address the mentioned issues, we propose a novel solution based on an existing federated digital identity management framework. Specifically, we cast our solution in the context of the VeryIDX framework [3]. A VeryIDX federation is composed of SPs, registrars, and users. SPs provide services to users as in conventional e-commerce environments. *Registrars* store and manage information related to users' identity attributes in identity records (IdR). (See Appendix A for a short overview of VeryIDX).

In this paper, we show how a Near Field Communication (NFC) [1] enabled cellular phone manages and use receipts for m-commerce transactions in a secure fashion. NFC is a standard-based, short-range ($\sim$ 15 centimeters) wireless connectivity technology supporting two-way interactions among electronic devices. A cellular phone having a NFC device is able to communicate not only with Internet via wireless connections but also with smart card readers.

In addition, the cellular phone applications, referred to as MIDlets, can access the phone's tag for reading and writing data. Furthermore, such cellular phones often have an extended memory which can be used for storing receipt information.

The main contributions of the paper are as follows: 1) An approach to store receipts in cellular devices and support their secure and privacy-preserving usage. 2) A comprehensive notion of history of transactions which include both on-line as well as in-person transactions. 3) Cryptographic protocols, such as secret sharing and zero knowledge proofs, in a potentially vulnerable and constrained setting. Specifically, our approach supports flexible strategies based on Shamir's secret sharing to cater for different user requirements and architectural constraints. In addition, our protocols use aggregate zero knowledge proofs to efficiently support the proof of multiple receipt attributes. 4) An evaluation of the performance of the proposed approach on actual cellular phones.

The rest of the paper is organized as follows. In Section II, we discuss key requirements and an example illustrating our approach. In Section III, we introduce preliminary notions and background information. In Section IV, we provide protocols for securing, managing and using receipts on the cellular phone. This is followed by the analysis of the approach in Section V. Finally in Section VII we outline some conclusions and future work.

II. Requirements

In this section, we discuss relevant security and privacy requirements concerning the use of receipts on cellular phones in the context of in-person transactions.

Security of information stored in cellular phones is a critical factor for m-commerce transactions [8]. The following requirements concern the security of the receipt attributes and the receipt usage. We focus on protection requirements for the mobile device and do not consider other security requirements concerning the SP and registrar since these are addressed by our previous work [3]. First the user should be able to prove ownership of the receipts recorded at the cellular phone. Second the receipt attributes should correspond to the transaction for which the receipt was issued. Additionally the integrity of the receipt attributes and other identity attributes stored in the receipt record should be assured. Finally, the secrets stored on the cellular phones should be trustworthy. Ensuring trustworthiness of secrets may require splitting them, or integrating independent parties or components on the cellular phones. It is important that the above requirements be addressed at the same time as assuring flexibility and receipt portability.

To preserve *privacy*, it is important that users be able to maintain some level of control over their personal information and other identity attributes while conducting transactions with any SP. More specifically we require that the retrieval of receipts should be driven by conditions based on the user preferences or the SP policies. In addition the user should explicitly consent or be aware of the data being released from the cellular phone. Finally, the user should be able to disclose to the SP only the information strictly needed in order to satisfy the SP service policies, thus satisfying the minimal disclosure principle.

In the following we introduce an example of a possible m-commerce application of our protocols.

Example 1: Consider a user Alice who conducts an e-commerce transaction with SP $e-$*Follets* to buy a book for the price of \$134.65 and then uploads the receipt corresponding to this transaction. As per the VeryIDX protocols Alice uses her set of receipt attributes to create a cryptographic commitment on the price of the receipt (see [3]) which would be needed to create proof of ownership of the receipt and prove receipt attributes.

Alice downloads a subset R of the receipts at the registrar to her NFC cellular phone. Alice then decides to use her receipts at a physical SP shop *Follets* to qualify for particular discount that requires her to show that she has performed a (possibly electronic) transaction buying an item from the *Follets* or $e-$*Follets* for more than 80\$, set by a SP. The device retrieves the appropriate receipts based on the conditions specified by the SP. All the receipt information is not passed to the SP without the explicit consent of Alice. Moreover to ensure minimal disclosure, Alice wishes to prove that the receipt from the $e-$*Follets* transaction is greater than 80\$ without showing the exact value in clear. If the proof is correct, *Follets* will offer the discount to Alice.

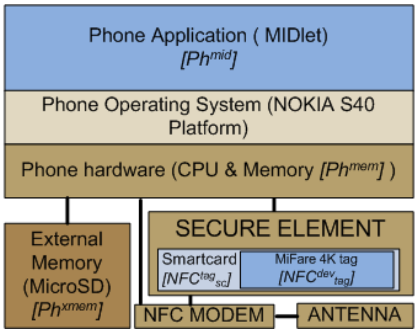
Phone Application (MIDlet)
[Ph^mid]
Phone Operating System (NOKIA S40 Platform)
Phone hardware (CPU & Memory [Ph^mem])
External Memory (MicroSD)
[Ph^xmem]
SECURE ELEMENT
Smartcard
[NFC^tag_sc]
MiFare 4K tag
[NFC^dev_tag]
NFC MODEM
ANTENNA

Fig. 1. Nokia NFC cellular phone components.

Follets, in turn, would like to make sure that the receipts are actually *owned* by the individual presenting the receipts. Using the cryptographic commitment stored in the cellular phone, which is signed by the registrar, Alice is able to prove to *Follets* that she owns the receipt containing the receipt attributes being presented. The integrity of the receipt attributes can also be verified by *Follets* by checking the registrar's signature on the receipt attributes.

III. Preliminary Notions

We begin with a brief overview of the key components of the cellular phone. We then illustrate the underlying cryptographic protocols adopted in the system, that is, the secret sharing protocols and the aggregate zero knowledge proof protocol.

A. NFC Cellular Phone Architecture

We employ the Nokia 6131 NFC cell phone (Ph^{NFC}) [1] for testing our portable receipts' protocols. We assume that the SPs have a NFC reader (denoted by NFC^{SP}_{reader}) which transmits and receives messages from the NFC cellular phone. The phone is integrated with a NFC device and thus contains both reader and writer for the embedded smart card and tags that directly communicate with SP's reader. Ph^{NFC} 's components are shown in Figure 1.

As shown, the main software component for managing and using receipts is the MIDlet suite. The MIDlet suite consists of a Java Application Descriptor (JAD) and a MIDlet. A MIDlet (denoted by Ph^{mid}) is a Java program that runs on the Java Virtual Machine (JVM) enabled mobile device. The JAD controls possible permissions that the MIDlet can have. A Ph^{mid} is installed onto a phone and operates in a sandbox [17] so different MIDlets are isolated from each other. The cellular phone has a secure element which can only be accessed by MIDlets signed by a trusted third party. Moreover these MIDlets should know the access key. The secure element consists of two main components, namely the Mifare tag (NFC_{tag}^{dev}) and Smartcard (NFC_{sc}^{dev}).

B. Cryptographic protocols

a) Shamir's Secret Sharing Scheme: Secret sharing refers to methods for distributing a secret amongst a group of participants, each of which is allocated a share of the secret. The secret sharing scheme that we adopt is the (k,n) threshold scheme by Shamir [15]. Such scheme splits a secret S into n partial secrets so that k, with $k < n$, partial secrets are required to reconstruct S. The scheme works as follows. First, $(k-1)$ random coefficients $\{a_1, \ldots, a_{k-1}\}$ are chosen. Then, a polynomial $f(x) = a_0 + a_1x + a_2x^2 + \ldots + a_{k-1}x^{k-1}$ is built, with with $a_0 = S$. Based on $f(x)$, n shares are constructed. Each share is of the form $(i, f(i))$ where i denotes the input to the polynomial and $f(i)$ the output. Given any subset of k of these pairs, the coefficients of the polynomial can be evaluated using interpolation. The secret, that is, a_0 can thus be determined.

b) Aggregate Zero Knowledge Proof of Knowledge (AgZKPK): To be able to create proof on an identifier m, the user[1] creates a Pedersen's commitment of the form $M = g_1^m h_1^r$ where r is a random value chosen by the user and g_1 and h_1 are public parameters of the registrar. Such commitment is enrolled at the registrar. At enrollment the registrar signs the commitment M to generate $\sigma = M^\chi$ as the signature where χ is the secret key of the registrar.

The verification works as follows. Let $\sigma_1, \sigma_2, \cdots, \sigma_t$ be the signatures corresponding to the identifiers that need to be proved by the user to the SP. The user aggregates the signatures into $\sigma = \prod_{i=1}^{t} \sigma_i$, where σ_i is the signature of committed value $M_i = g_1^{m_i} h_1^{r_i}$. It also computes $M = \prod_{i=1}^{t} M_i = g_1^{m_1+\cdots+m_t} h_1^{r_1+\cdots+r_t}$. The user sends σ, M, M_i, $1 \le i \le t$ to the verifier.

As a next step the user and the verifier SP carry out the following ZKP protocol[2]:

$$PK\left\{(\alpha, \beta) \colon M = g_1^\alpha h_1^\beta,\ \alpha, \beta \in \mathbb{Z}_q\right\}$$

After the verifier SP accepts the zero-knowledge proof of the commitments, it checks if the following verifications succeed:

$$M = \prod_{i=1}^{t} M_i \qquad \text{and} \qquad e(\sigma, g_2) = e(M, v)$$

where g_2 is a public parameter, v is the public key of the registrar and e is a bilinear mapping. If the last step succeeds then the verifier accepts the ZKPK of the signed commitments.

IV. Protocols for Cellular phones

In this section, we present the approach for securing the secrets associated with the receipt attributes and introduce protocols to execute AgZKPK protocols on the cellular phone.

A. Sharing secrets used in cellular phone

Receipts are stored in the external memory Ph^{xmem} or other memory devices, as per the user choice and Ph^{NFC}'s storage availability. The protocols for receipt usage are independent of how these receipts are stored. The cryptographic secret and random value *r* used to create a Pedersen commitment are associated with a receipt attribute. Such secret is used to generate an AgZKPK. We split this random value *r* into several different locations to increase security. Thus, only when all of the partial secrets are merged, *r* can be correctly reconstructed.

1) Secret sharing Options: A key feature of our approach is to allow the user to select different security levels by choosing different secret splitting levels. Specifically, our protocols support four specific *secret splitting levels*, namely *low*, *medium*, *medium-high*, *high*. The secret splitting level depends both on n and k parameters of Shamir's (k,n) secret sharing scheme. In this paper we consider $n = 4$ which denotes the maximum number of secret shares. According to the desired security splitting level, the secret is split in k shares. Depending on k, secret shares are stored in different components and devices, namely: 1) the NFC cell phone's internal memory, denoted as Ph^{mem}; 2) the NFC cell phone's external memory, denoted as Ph^{xmem} which is protected by a user secret PIN; 3) the NFC cell phone's smart card NFC_{sc}^{dev}; and 4) the user remote personal computer (PC). The user chooses the threshold value k that is the minimum number of partial secrets for reconstructing the original secret. For example, if user chooses *medium-high*, corresponding to *k* = 3, the partial secret would be in any of three combinations of the four different locations where the secret can be stored. When the user first saves the secret on the cell phone, he/she makes the choice on the security level. As such the choice of the secret sharing level depends on user requirements and architectural constraints. Specifically the low level sharing provides maximum user convenience as the secret does not need to be combined at the time of use. If the user wants to moderately increase security, then, as in the medium secret sharing level, the user provides a PIN needed to retrieve the secret share. Additionally, storing a secret share in the external memory makes it possible to port the secrets shares to other cellular phones. If the user's cellular phone allows storage in the smart card secure component, as in the case of medium-high level, the user stores part of the secret in the smart card. This option may not be always available because of the storage limitations of smart cards; however it enables higher security in that smart cards can maintain integrity and confidentiality of the data they store. The users can in fact destroy the contents remotely- by means of the wireless cellular phone service- in case their cellular phone is lost or stolen [14]. Finally, in case of high secret sharing level, the user must run a service from its PC, to retrieve the secret share stored at the PC. In this case, even if the cellular phone is stolen which might result in the secret shares stored in the cellular phone to be compromised, the actual secret is not compromised because it still requires

[1] Here and in what followings the term "user" denotes an agent running on behalf of the real user.

[2] The convention is that Greek letters denote the secrets quantities, whereas all the other parameters are sent to the verifier [5].

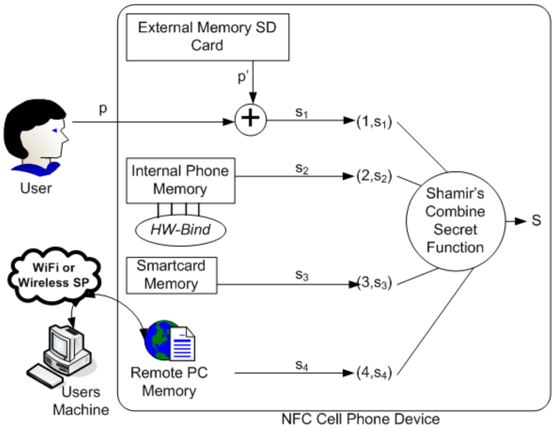

Fig. 2. Secret Splitting and Combination

the secret share in the PC.

In what follows we focus on the secret share management protocols as required by the highest level of security. The techniques to split and combine secret shares for the rest of the cases (that is low, medium, and medium high) can be derived from this case.

2) Creation and Distribution of Partial Secret Shares: Following Shamir's secret sharing technique we split the original secret S into secret shares s_1, s_2, s_3 and s_4. The first share s_1 is stored in the internal memory of the cell phone Ph^{mem}. The second share s_2 is further split into two secrets. A user chosen PIN number p and a number p' are selected such that their XORed value, $p \oplus p' = s_2$. p', is stored in Ph^{xmem}. The third share s_3 is stored in the NFC^{dev}_{sc}. Since the smart card offers a storage and computation unit, s_3 can potentially be encrypted and stored in the smart card integrated in the Ph^{NFC} [14]. Finally the fourth secret share s_4 is stored in the user's PC which has to be accessed remotely by the Ph^{NFC} when it is being used. As such, s_4 is accessed via a secure connection and running a remote service connection as elaborated later in this section.

3) Retrieval and Combination of the Partial Secrets: When the ZKPK proofs corresponding to the receipt attributes need to be created, k secret shares are combined to create the original secret in the users Ph^{NFC}. Shares are retrieved by the device according to their location. Specifically, the procedure to retrieve each secret share at the Ph^{NFC} (see Figure 2) is as follows:

1) s_1: Ph^{mem} is accessed directly by the secret combining function to retrieve s_1.
2) s_2: The user is required to input the secret PIN number p using the NFC cell phones keypad. p' which is stored in Ph^{xmem} is used to compute the second secret share $s_2 = p \oplus p'$.
3) s_3: The secret s_3, stored in the smart card, is read by the Ph^{NFC}
4) s_4: The secret share s_4 is stored at the user's PC which may not present in the vicinity of the Ph^{NFC} when it is being used. As a first step, the cellular phone connects to a service, e.g., the Windows NT Service which runs on the user's PC, by remote procedure calls in order to retrieve the secret shares stored at the PC. The activation requires the user to provide an authentication token. The authentication token may be automatically generated by the device, if the user configures its device to not require explicit authentication token input when the service activation occurs. Once the token is successfully verified, the $NFC^{user}_{Internet}$ queries the PC 'wallet' [3] containing the secret share. After the secret share s_4 is retrieved, the $NFC^{user}_{Internet}$ deactivates the query service.

The secret shares are retrieved and combined by a new MIDlet Ph^{midc} which runs in a protected domain with restricted permissions. In this way the secret reconstructed on the Ph^{midc} is isolated from other applications running on the Ph^{NFC} and can be used only as intended without requiring that a copy be stored at the Ph^{NFC}'s memory.

B. Managing receipts in the NFC cellular phone

In this section we present the protocol for providing receipts using Ph^{NFC}. We focus on the key steps specific to the NFC device itself, and omit the details on the cryptographic protocols involved for the correctness and integrity of the receipts. Adding receipts to the NFC device is straightforward as the user can select the digital receipts from its RREC which, in turn, is saved in Ph^{xmem}. More specifically, each time a user receives a receipt at a physical SP location, the phone can use standard digital data communication technology [1] to upload the receipt to the registrar. By contrast, the verification protocol has several interesting challenges. Recall that the protocol is carried out by the user to provide the proofs of receipt attributes required to satisfy the SP's trust establishment policies π_{SP}. Such policies specify the conditions the user needs to meet to qualify for a particular request (for example a discount).

The main steps of the VerifyReceipt protocol for executing such queries are given by Protocol 1. The protocol makes use of three functions, specifically: QueryRREC(RREC, $conditions$), UserInterface and CreateProof($ReceiptIDs$, $Commitments$, $Tags$). QueryRREC returns the receipts in the RREC which satisfy the conditions listed in $conditions$ specified by user preferences and/or the SP service policy. CreateProof creates Aggregated ZKPK on the receipt attributes indicated by $ReceiptIDs$ and $Tags$, along with the list of associated commitments stored. Finally, UserInterface is responsible for the user interaction interface involved in the selection and submission of receipt attributes.

As illustrated, user's Ph^{NFC} is initialized with a set of receipts $\mathcal{R}$ uploaded before carrying on the protocol. As a first step, the user's cell phone tag denoted by NFC^{dev}_{tag} captures π_{SP} sent by the SPs NFC^{SP}_{reader}[4]. NFC^{dev}_{tag} transfers this policy to the cell phones main memory Ph^{mem} in step 2.

[3] The wallet is cryptographic secure place where secrets are stored.

[4] NFC reader is a device that can transmit as well as receive data using NFC.

Subsequently, in step 3 NFC^{dev}_{tag} triggers an event to the cell phones computational unit to initiate the Ph^{mid} MIDlet to run the receipt queries. Ph^{mid} calls the function QueryRREC to evaluate the potential receipts in $\mathcal{R}$ which can satisfy the conditions in π_{SP}. As a result the eligible receipts $\mathcal{R}' \subset \mathcal{R}$ are retrieved from the Ph^{NFC} and displayed on the cell phones screen. In step 5, the Ph^{mid} calls the user interface related function UserInterface which allows the user to choose the receipt attributes from $\mathcal{R}'$ which the user wishes to show in clear (L_1) or create a ZKPK (L_2).

In the next step, the main Ph^{mid} initiates a new MIDlet called Ph^{midc} which runs in a protected domain with restricted permissions. This is because Ph^{midc} uses cryptographic secrets associated with the receipt attributes to create receipt proofs. The receipt proofs are created in an aggregated fashion using the function CreateProof in step 8. This results in the aggregated ZKPK called $AgProof$. Ph^{midc} sends the $AgProof$ to Ph^{mid}. The receipt attributes and proof are concatenated in step 9 to obtain the final token $\mathcal{F} := L_1||L_2||AgProof$ where: L_1 is a list of receipt id's, attribute values signed with registrar's key, and the corresponding attributes it wants to reveal in clear; and L_2 is a list of receipt id's, commitment values signed with registrars key, and of the corresponding attributes of which the user wants to prove ownership. Using the UserInterface function the user approves the disclosure of this information to the SP. Upon receiving user consent, $\mathcal{F}$ is sent via the NFC^{dev}_{tag} to be read by the NFC^{SP}_{reader}. If the receipt attributes and proofs provided satisfy the conditions defined in the SP's policy (π_{SP}), the user obtains the desired services.

V. Analysis

In this section we analyze the protocols used by the Ph^{NFC}, with respect to performance, security and privacy. We focus on the applications running on the phone executing the receipt usage protocol and discuss how we use specific capabilities of the phone to achieve desired requirements.

A. Performance

One main factor that could affect performance is the use of large numbers to perform the ZKPK computations. One may expect the time taken to compute the ZKPK to be proportional to the number of receipt attributes involved. However, our AgZKPK protocols takes almost constant time for the ZKPK generation even if the number of attributes being proven increases. The reason is that the AgZKPK only requires a constant number of exponentiations [2]. Such theoretical estimate is confirmed by experimental results. A graph showing the computation time for the aggregated proof for numbers of identifier in the range [1,. . .,50] is provided in Figure 3. The average time for a proof creation is 2.257 seconds. As expected, proof creation on the cellular phone takes longer than on a PC due to the phone's limited computing power. The experiments also show that, as expected, the increased number of identifiers being proven does not result in the increase in the computation time.

Protocol 1 VerifyReceipt : User providing receipt attributes to SP

```
Require: SP trust establishment policies $\pi_{SP}$, user receipts $\mathcal{R}$ on $Ph^{NFC}$.
Ensure: The user's $Ph^{NFC}$ and the SP's $NFC^{SP}_{reader}$are located at a close proximity.
1: $NFC^{SP}_{reader} \xrightarrow{M1} NFC^{dev}_{tag}$ $[M1 = \pi_{SP}]$
2: $NFC^{dev}_{tag} \xrightarrow{M2} Ph^{mem}$ $[M2 = \pi_{SP}]$
3: $NFC^{dev}_{tag} \xrightarrow{M4} Ph^{CPU}$[$M4$ =initiate $Ph^{mid}$ event]
4: $Ph^{mid}$[uncritical domain] executes QueryRREC($\mathcal{R}, \pi_{SP}$) $\leftarrow \mathcal{R}'$
5: $Ph^{mid}$[uncritical domain] executes UserInterface ($\mathcal{R}'$)
6: { User chooses $L_1 := \{R_i, a_k, attr_k\}$, a list of receipt id's, signed attribute values, and the corresponding attributes to reveal in clear
7: User chooses $L_2 := \{R_i, C_l, attr_k\}$, a list of receipt id's, signed commitment values, and the corresponding attributes to prove }
8: $Ph^{midc}$[critical domain] executes CreateProof ($L_2$) $\leftarrow$ $AgProof$
9: $Ph^{mid}$[uncritical domain] executes UserInterface to provide consent with final token $\mathcal{F} := L_1||L_2||AgProof$
10: $Ph^{mid} \xrightarrow{\mathcal{F}} NFC^{dev}_{tag}$
11: $NFC^{dev}_{tag} \xrightarrow{\mathcal{F}} NFC^{SP}_{reader}$
```

We also compared the time that the SP takes to verify such proofs at the server (which is an Intel Pentium D CPU 3.0 GHz and 1G ram and runs Windows XP Operating system) with the computation time for creating an Applet. The results are shown in Figure 4. On an average the SP takes 0.089 seconds to verify aggregate proofs of 50 identifiers. We notice that even if the number of exponentiations executed at the SP is constant, the amount of time taken for the verification increases sublinearly. The reason is that during the verification the SP is required to multiply all the commitments to verify the resultant aggregate signature.

B. Security and Privacy

Fundamental security and privacy properties, such as minimal information disclosure and non-replay of ownership proofs, are guaranteed because of the use of the AgZKPK protocols. Therefore, in the following we focus on the possible attacks to the MIDlets and on how our proposed protocols ensure user consent when releasing receipt attributes.

1) Security: The *integrity* of the MIDlet is assured by the use of code-signing certificates, and the trusted MIDlet suite for all applications running our protocols. The trusted MIDlet suite, which must be signed, ensures the integrity of the applications running on the Ph^{NFC}. An important aspect of MIDlets is that they run in a sandboxed environment [17], providing the necessary isolation of the memory usage between Ph^{mid} and Ph^{midc}. Therefore the computations in Ph^{midc} are not influenced by the computations performed by Ph^{mid} or other external applications.

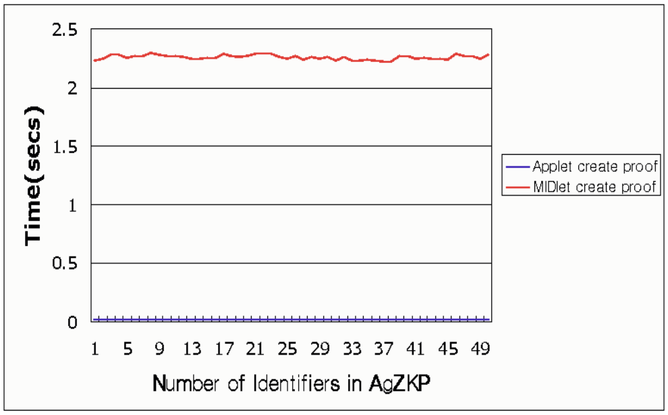

Fig. 3. Comparison of AgZKPK Proof Creation in the Midlet versus the Applet

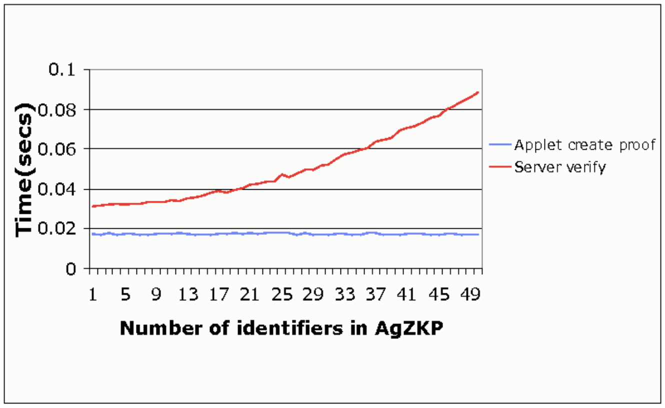

Fig. 4. Comparison of Aggregate Proof Create versus Verify

A cellular phone has a higher risk of physical *asset theft* as compared to ordinary desktop computers. The proposed use of secret sharing aims at protecting the secrets associated with sensitive identity information. More specifically, the secret sharing mitigates the threat of asset theft in a three fold manner. First, by storing a partial secret in a separate external memory that the user should keep separated from the cellular phone when the secrets are not needed. If the partial secret from the external memory cannot be retrieved, then the original secret cannot be reconstructed. The second measure is to associate the partial secret s_2 with a user pin p (see Figure 2) that is not stored on the phone. The third measure is to require the online connectivity during the use of a split secret. When a cellular phone is stolen, the user can revoke the secrets stored in the device by calling the cellular phone service provider. At the time of verification during a m-commerce transaction, the verifier checks for such flag and can invalidate a transaction if the use of revoked secrets is identified.

A potential threat is represented by *malware attacks* that may try to retrieve the partial secrets from the cellular phone. In our context, splitting the secret into secure components protects against the malware attacks since the malware cannot access secure element unless it knows the secret key which is required for access. For example, the secret share stored in the smartcard cannot be accessed by unsigned and potential malicious code.

2) Privacy: An example of attack which might breach user's privacy would be that a malicious SP accesses receipts and other identity attributes from the user's cellular phone without explicit consent of the individual. To mitigate this threat, ensuring user control is crucial [8]. The user consent is attained in steps 5 and 9 of Protocol 1. Internally, in the Ph^{NFC}, the UserInterface function of Ph^{mid} displays the potential receipt attributes the individual can use in a given transaction. Based on the individuals' choice, the list of attributes L_1 and L_2 are constructed. This is followed by cryptographic operations computed by the Ph^{midc} whose permissions are set which require user input (denoted by *User Permission* in [10]) to execute the CreateProof function. Finally, in step 9, before the receipt attributes and proofs are revealed to the SP, this information is confirmed using the UserInterface function of the Ph^{mid}. The consent thus helps the user in maintaining a level of control over which identity attributes to release to a given SP.

VI. Related Work

In this section we discuss related work on the use of cellular phones for e- and m-commerce transactions involving identity attributes and other recent developments in mobile identity management initiatives.

With the advent of high-speed data networks and feature-rich mobile devices, the concept of *mobile wallet* [12], [4] has gained importance. A seminal work introduced the concept of wallets with observers [6] enabling off-line digital cash and credentials to be used in commercial settings. Such project developed electronic wallet technology, using such technologies the transactions can be performed via a short range infrared link either directly with compliant cash registers and wallets held by other individuals, or over the Internet, or other SPs. A major difference of our approach is that it does not require an observer, as the integrity of the receipts is based on the signature of the registrar on the receipts. The addition of the observer would, however, be beneficial if the usage of the receipts were constrained for example by the number of times of use.

Other mobile identity management initiatives have gained importance with the rapid adoption of second-generation mobile telecommunication systems, leading to the growth of m-commerce [13], [9]. Two specific factors critical in this domain are usability and trust. Several approaches to enhance usability of mobile devices have been proposed [7]. Trust for mobile device includes several security and privacy properties such as confidentiality, integrity, user control and minimal disclosure of the identity data stored on such devices. One approach to mobile IdM is based on the GSM [13]. GSM based IdM uses the GSM infrastructure and the Subscriber Identity Module (SIM) as the underlying platform.

The Secure Electronic Transaction (SET)[11] protocol was developed to allow credit card holders to make transactions without revealing their credit card numbers to merchants and also to assure authenticity of the parties. SET deploys dual signature for merchant and payment gateway. Each party can only read a message designated for itself since each message is encrypted for a different target. To enable this feature, card holders and merchants must register with a Certificate Authority before exchanging a SET message. The SET protocols assure both confidentiality and integrity for the messages exchanged among card holders, merchants and payment gateway whereas our protocol is designed to assure integrity between SPs and registrars. SET authenticates the identity of the cardholder and the merchant to each other because both of them are registered with the same Certificate Authority. However, our protocols do not mandate this requirement. SET is considered to have failed because of its complexity. It requires that cardholders register in order to get a certificate. In our protocol, the users do not need to have certificates [5].

VII. Conclusion

This paper proposes protocols for managing electronic receipts in cellular devices and support their secure and privacy preserving usage. The flexible secret sharing strategies based on Shamir's secret sharing approach allow us to cater to different user requirements.

Currently, we are investigating what are the conditions under which VeryIDX could be deployed in large-scale. We have identified the following requirements. First, SPs should install the VeryIDX verification component. Such component, which is in charge of verifying the proofs by the client, is very small (around 51KB) and is modular. It can be therefore easily integrated with the rest of the SP systems. Second, the SPs should be able to connect to the registrar in order to retrieve the commitments used for the ZKP verification. Therefore, the SPs should have an on-line connection with the registrar during the execution of our protocols. Since we recognize that such a requirement may not be easily met and may slow down the response time of the protocols, we are developing a new version of VeryIDX which does not require the registrar to be on-line during the protocol execution. The approach that we will adopt is based on the issuance of special certificates, recording the commitments, by the registrar to the client. Such certificates are signed by the registrar to assure their integrity. An approach based on the Merkle hash technique will be used to support the selective release of receipts and still assure the integrity of the commitments stored on such certificates. The last requirement is that the SPs must be able to verify the signatures of the registrar. Therefore, the registrar and SPs must have a common PKI infrastructure. However users are not required to have such an infrastructure.

Our implementation on the Nokia NFC cellular phones shows that the approach has very good performance. An important issue is whether our approach can be easily extended to other brand's cellular phones. In theory, our approach should be portable of any NFC forum specification compliant cellular phone; the only architectural requirement is to have a NFC-enabled chip in the cellular phone. However, even though the NFC forum is releasing the specification of the NFC device, the software still needs to be written using vendor-specific proprietary APIs. Therefore, writing an application for a NFC cellular requires the use of proprietary development kits [16]. For example, to develop a NFC MIDlet on Samsung SGH x700, a Software Development Kit from Samsung, currently providing NFC APIs, is needed. Therefore the source code developed for Nokia phones cannot be directly re-used on the Samsung phones. We will investigate compatibility issues among the various NFC cellular phones as part of future work.

In addition, as a part of future work, we plan to extend our approach along the following directions. The first is related to the extension of the secret sharing approach to include other device ensembles. With the emerging personal computing ecosystem, the secret sharing scheme could leverage various other electronic devices (in addition to cellular phones) for storage and recreation of the secret shares. The second direction is related to enhancing the user interface to allow the user to select the receipt attributes for the zero-knowledge proof generation and to improve the usability of our approach.

VIII. Acknowledgments

This material is based in part upon work supported by the National Science Foundation under the ITR Grant No. 0428554 "The Design and Use of Digital Identities" and upon work supported by the U.S. Department of Homeland Security under Grant Award Number 2006-CS-001-000001, under the auspices of the Institute for Information Infrastructure Protection (I3P) research program. The I3P is managed by Dartmouth College. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security, the I3P, or Dartmouth College.

References

[1] Near field communication forum. http://www.nfc-forum.org.

[2] A. Bhargav-Spantzel, A. C. Squicciarini, R. Xue, and E. Bertino. Practical identity theft prevention using aggregated proof of knowledge. Technical report, CS Department, 2006. CERIAS TR 2006-26.

[3] A. Bhargav-Spantzel, J. Woo, and E. Bertino. Receipt management-transaction history based trust establishment. In *DIM '07: Proceedings of the 2007 workshop on Digital identity management*, 2007.

[4] J.-P. Boly, A. Bosselaers, R. Cramer, R. Michelsen, S. F. Mjolsnes, F. Muller, T. P. Pedersen, B. Pfitzmann, P. de Rooij, B. Schoenmakers, M. Schunter, L. Vallee, and M. Waidner. The ESPRIT project CAFE - high security digital payment systems. In *ESORICS*, pages 217–230, 1994.

[5] J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In *Advances in Cryptology – CRYPTO '97*, pages 410–424, 1997.

[6] D. Chaum. Security without identification: transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

[5] Only the SPs and registrars must have certificate.

[7] A. Dix, T. Rodden, N. Davies, J. Trevor, A. Friday, and K. Palfreyman. Exploiting space and location as a design framework for interactive mobile systems. *ACM Transactions on Computer Human Interaction*, 7(3):285–321, 2000.

[8] S. Duri, M. Gruteser, X. Liu, P. Moskowitz, R. Perez, M. Singh, and J.-M. Tang. Framework for security and privacy in automotive telematics. In *WMC '02: Proceedings of the 2nd international workshop on Mobile commerce*, pages 25–32, New York, NY, USA, 2002. ACM Press.

[9] U. Jendricke, M. Kreutzer, and A. Zugenmaier. Mobile identity management. Technical Report 178, Institut fr Informatik, Universitt Freiburg, October 2002.

[10] O. Kolsi and T. Virtanen. Midp 2.0 security enhancements. In *HICSS '04: Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 9*, page 90287.3, Washington, DC, USA, 2004. IEEE Computer Society.

[11] V. MasterCard. Set secure electronic transaction specification book 1: Business description. 1997.

[12] S. F. Mjolsnes and C. Rong. Localized credentials for server assisted mobile wallet. *ICCNMC'01: International Conference on Computer Networks and Mobile Computing*, 00:203, 2001.

[13] K. Rannenberg. Identity management in mobile cellular networks and related applications. Information Security Technical Report 9, Johann Wolfgang Goethe University Frankfurt, January 2004.

[14] S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady. Security in embedded systems: Design challenges. *Trans. on Embedded Computing Sys.*, 3(3):461–491, 2004.

[15] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[16] Timo. Developing nfc applications. 2006.

[17] A. Wolfe. Toolkit: Java is jumpin'. *Queue*, 1(10):16–19, 2004.

[18] G. Zacharia, A. Moukas, and P. Maes. Collaborative reputation mechanisms in electronic marketplaces. In *HICSS*, 1999.

Appendix

VeryIDX is a system for managing receipts and using receipts for on-line e-commerce transactions within federations. The VeryIDX federation is composed of SPs, registrars and users. SPs provide services to users as in conventional e-commerce and other federated environments. The *registrars* store and manage information related to users' identity attributes in an identity record (IdR). The information recorded at the registrar is used to perform multi-factor identity verification of identity attributes of users. Such information does not disclose the values of the strong user attributes[6] in clear. Instead, it contains the cryptographic semantically secure *commitments* [6] of the identity attributes which are then used by the users to construct zero knowledge proofs of knowledge (ZKPK) [2] of those attributes. All receipts are stored in the user Receipt Record (RREC for short) which is created for each registered user. Users who make purchases online enroll their electronic receipts at the registrar and, a later stage, can provide such receipts as a proof of transactions. To preserve privacy, the information which is critical for the transaction is extracted from the receipt and disclosed.

[6] Strong user attributes are those attributes which uniquely identify the user and therefore are considered sensitive.