An Overview of VeryIDX - A Privacy-Preserving Digital Identity Management System for Mobile Devices

Federica Paci[1], Elisa Bertino[1], Sam Kerr[1], Anna Squicciarini[2], Jungha Woo[1]
[1] CERIAS and Computer Science Department, Purdue University
[2] Information Sciences and Technology, The Pennsylvania State University
Email: {paci,bertino,skerr,wooj}@cs.purdue.edu, acs20@psu.edu

***Abstract*— Users increasingly use their mobile devices to communicate, to conduct business transaction and access resources and services. In such a scenario, digital identity management (DIM) technology is fundamental in customizing user experience, protecting privacy, underpinning accountability in business transactions, and in complying with regulatory controls. Users identity consists of data, referred to as *identity attributes*, that encode relevant-security properties of the clients. However, identity attributes can be target of several attacks: the loss or theft of mobile devices results in a exposure of identity attributes; identity attributes that are send over WI-FI or 3G networks can be easily intercepted; identity attributes can also be captured via Bluetooth connections without the user's consent; and mobile viruses, worms and Trojan horses can access the identity attributes stored on mobile devices if this information is not protected by passwords or PIN numbers. Therefore, assuring privacy and security of identity attributes, as well as of any sensitive information stored on mobile devices is crucial. In this paper we address such problems by proposing an approach to manage user identity attributes by assuring their privacy-preserving usage. The approach is based on the concept of privacy preserving multi-factor authentication achieved by a new cryptographic primitive which uses aggregate signatures on commitments that are then used for aggregate zero-knowledge proof of knowledge (ZKPK) protocols. We present the implementation of such approach on Nokia NFC cellular phones and report performance evaluation results.**

***Index Terms*— digital identity management, identity attributes, privacy, mobile devices**

I. Introduction

The combined use of the Internet and mobile technologies (e.g. mobile devices, mobile and wireless communication) is leading to major changes in how individuals communicate, conduct business transactions and access resources and services. People are able to communicate anytime, anywhere with anyone. Technological advances as well as the increased number of mobile applications have resulted in new additions in end-user equipment. Smart mobile devices are equipped with various communication technologies, such as GSM/GPRS, 802.11-WLAN, Bluetooth, NFC and RFID chips as well as GPS for location awareness. Mobile devices today offer a broad spectrum of functions, including web browsers, operating systems (e.g Symbian), environments (e.g., Java virtual machine) for running mobile applications, and e-mail clients. In such a scenario, digital identity management (DIM) technology is fundamental in customizing user experience, protecting privacy, underpinning accountability in business transactions, and in complying with regulatory controls. Digital identity can be defined as the digital representation of the information known about a specific individual or organization. As such, it encompasses not only login names, but many additional information, referred to as *identity attributes*. The management of identity attributes on portable devices is however challenging. On one hand, identity attributes need to be shared to speed up and facilitate authentication of users and access control in a variety of contexts, including mobile environments. Users should be able to manage their identity attributes when carrying transactions or other interactions from portable devices such as cellular phones. On the other hand, the identity attributes must be protected as they may convey sensitive information about an individual and can be target of attacks.

Moreover, users should be able to control which service provider has access to information about their identity attributes. Assuring privacy and security of users' identity attributes, as well as of any sensitive information, in the context of mobile environments is further complicated by the fact that mobile devices are not secure. Recent statistics [12] show that millions of lost or stolen mobile devices which store users' sensitive data have been reported.

In addition to loss or theft, there are an increasing number of viruses, worms and Trojan horses targeting mobile devices. Moreover, recent attacks against Bluetooth and well-known WLAN and GPRS vulnerabilities show that it is very easy for attackers to compromise mobile devices [31].

To date there are no comprehensive solutions for handling identity attributes on mobile devices and even solutions for conventional non-mobile environments are still at a preliminary stage.

In this paper we make some steps towards such a solution and present a multi-factor identity attribute verification approach for mobile devices. By multi-factor verification we mean that whenever an individual presents an identity attribute for carrying on a transaction with a

doi:10.4304/jsw.4.7.696-706

party, such party may verify the right of this individual to use such identity attribute by asking him/her to present other identity attributes. The specification of which identity attributes have to be presented is stated by *verification policies*. Different parties in a distributed system may specify different policies. To assure that such an approach does not undermine privacy, we have developed a cryptographic protocol, referred to as *aggregate zero knowledge proof* [4]. Such a protocol allows a user to prove the knowledge of multiple secrets to a party without having to reveal them to this party. We have developed a version of such protocol for Near Field Communication (NFC) [21] enabled cellular phones. NFC is a standard-based, short-range ($\sim$ 15 centimeters) wireless connectivity technology supporting two-way interactions among electronic devices [21]. A NFC device embedded in the cellular phone 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.

The rest of the paper is organized as follows. Section II provides an overview of VeryIDX, our system for managing identity attributes. Section III introduces the basic notions on which the multi-factor identity verification is based. Section IV presents the protocols for securing, managing and using identity attributes on the cellular phone. Section V describes the system architecture. Section VI describes the implementation of the multi-factor identity verification protocol on Nokia NFC mobile phones. Section VIII presents experimental performance results. Section VIII overviews some extensions to the VeryIDX identity verification protocol. Section IX discusses related work. Finally, Section X concludes the paper and outlines some future work.

II. VeryIDX Overview

Our approach is based on an extended notion of federation. A federation is composed of the following entities: identity providers (IdPs), service providers (SPs), registrars and users [5]. SPs provide services to users as in conventional e-commerce and other federated environments. IdPs issue certified identity attributes to users and control the sharing of such information. The *registrars* store and manage information related to *strong identity attributes*, that is, identity attributes uniquely identifying an individual, as opposed to *weak identity attributes* which do not have such property. The information recorded at the registrar is used to perform multi-factor identity attribute verification. Note that, unlike the IdPs, the information stored at the registrar does not iclude the values of the strong identity attributes in clear. Instead, such information only contains the cryptographic semantically secure *commitments* of the strong identity attributes which are then used by the clients, running on behalf of users, to construct zero knowledge proofs of knowledge (ZKPK) [14] of those attributes. The key elements of our solution can be summarized as follows:

1) Whenever a party P presents a strong identity attribute to a SP in the federation, the SP requires additional proofs of identity according to its local verification policies. The submission of additional proofs of identity by P and the corresponding verification by the SP is executed through the use of our aggregated ZKPK protocols. By using such protocol the party can prove knowledge of any strong identity attributes efficiently. Since the actual values of the identifiers are not revealed to the SP, this approach preserves the privacy of the parties.
2) Each strong identity attribute used by a party P in a federation, either for direct use or just for identity proof, must be registered with a registrar that, upon registration, provides P with a signature on the commitment of the identifier. The management of the registered strong identity attributes is based on a *identity record* (IdR) created for each registering party. The identity record collects the commitments corresponding to the strong identity attributes.
3) To prevent a malicious party from registering with a federation a strong identity attribute owned by another individual, a duplicate detection protocol is run upon registration to determine whether the same strong identity attribute has already been registered by a different party.

Example 2.1: Consider a user Bob who is part of the E-Mall federation, that offers a safe environment for online shopping. Bob enrolls at registrar Reg_1 and registers his strong identifiers: his credit card number (CCN) and his social security number (SSN). The commitments values of CCN and SSN signed by the registrar are maintained in Bob's IdR. Bob now can use his CCN and SSN to prove his identity. Suppose then that Bob wants to buy a book from *e-Follets* SP. According to *e-Follets*'s policy, this store requires Bob's CCN along with a different form of identity verification for authentication. *e-Follets* thus challenges Bob's SSN. As such, Bob, in order to prove the ownership of CNN, downloads his IdR from the registrar Reg_1 onto his NFC cellular phone. The device retrieves the identity tuples corresponding to CCN and SSN specified in the SP-'s *e-Follets* policy and builds the aggregate proof of knowledge to be sent to *e-Follets*.

III. Preliminary Concepts

In this section we first introduce the cryptographic protocols used to implement our privacy preserving multi-factor identity verification approach. We first introduce the Pedersen commitments used to generate strong identity attributes secure commitments and the ZKPK protocol. Then, we describe the Shamir's secret sharing scheme, that is used to protect the secrets used to compute Perdesen commitments. Finally, we briefly present the Boneh's protocol [7] to generate aggregate signatures based on bilinear mappings.

a) Pedersen Commitment: Let g and h be generators of a group G of prime order q. A value m is committed

doi:10.4304/jsw.4.7.696-706

TAG	COMMITMENT	SIGNATURE	VALIDITY-ASSURANCE	OWNERSHIP-ASSURANCE	WEAK IDENTIFIER
CCN	329839797987493827983	7438726487979976668676989	A	B	Bob Smith
SSN	398723987479738294991	8887472472323098409236610	U	A	Bob Smith

Figure 1. Simplified graphical representation of an IdR

by choosing r randomly from $\mathbb{Z}_q$ and computing commitment $C = g^m h^r$. Commitment C is opened (or revealed) by disclosing m and r, and the opening is verified by checking that C is indeed equal to $g^m h^r$. A prover can prove by using a zero-knowledge proof that it knows how to open such commitment without revealing either m or r.

b) Zero-knowledge proof of knowledge: In our approach we use the techniques by Camenisch and Stadler [8] for the various ZKPK of discrete logarithms and proofs of the validity of statements about discrete logarithms. We also conform to the same notation [8]. For instance to denote the ZKPK of values α and β such that $y = g^\alpha h^\beta$ holds, and $u \leq \alpha \leq v$, we use the following notation:

$$PK\{(\alpha, \beta) : y = g^\alpha h^\beta \wedge (u \leq \alpha \leq v)\}$$

c) Shamir's secret sharing scheme: Shamir's (k, n) threshold scheme [29] is a method that divides a secret into n shares and allows the secret to be reconstructed if and only if any k shares are present. Here k and n are both positive integers and $k \leq n$. It is also called Shamir's secret sharing scheme.

The scheme works as follows. A trusted party, T, chooses a finite field $\mathbb{F}_p$ of p elements, with p large enough. Let the secret message S be encoded as an element $a_0 \in \mathbb{F}_p$. T randomly chooses $k-1$ elements $a_1, \ldots, a_{k-1} \in \mathbb{F}_p$, and constructs a degree $k-1$ polynomial $f(x) = a_0 + a_1 x + \ldots + a_{k-1} x^{k-1} \in \mathbb{F}_p[x]$. T chooses n elements $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}_p$, and creates the secret shares S_i as pairs

$$S_i = (\alpha_i, f(\alpha_i)), 1 \leq i \leq n,$$

where $f(\alpha_i)$ is the polynomial evaluation of f at α_i. Given any subset of k such shares, the polynomial $f(x)$, of degree $k-1$, can be efficiently reconstructed via interpolation. The secret S, encoded as the constant coefficient a_0, is thus recovered.

Shamir's (k, n) threshold scheme has many good properties. Most prominently, it is information theoretically secure, in the sense that the knowledge of less than k shares gives no information about the secret S better than guessing; and it is minimal, in that the size of each share does not exceed the size of the secret. Interested readers can refer to [29] for more details.

d) Bilinear maps: For a security parameter k, let q be a prime of length k, and G_1, G_2, G_T be groups of order q. Let $g_1 \in G_1$, $g_2 \in G_2$ be generators. Function e: $G_1 \times G_2 \rightarrow G_T$ is a bilinear mapping if it satisfies the following properties:

1) For all $u \in G_1$, $v \in G_2$ and a, $b \in Z$, $e(u^a, v^b) = e(u,v)^{ab}$.
2) $e(g_1, g_2) \neq 1 \in G_T$.
3) There exists a computable isomorphism φ from G_2 to G_1, such that $\varphi(g_2) = g_1$.

e) Bilinear aggregate signatures: The aggregate signature concept has been proposed by Boneh et. al [7]. We refer to such signature scheme as BGLS. Informally, an aggregate signature scheme allows multiple signatures to be aggregated into one signature with respect to the public keys of the signers and the signed messages. The BGLS scheme consists of five algorithms: *KeyGen*, *Sign*, *Verify*, *Aggregate*, and *AggVer*. Any principal P uses $KeyGen$ to generate the private and public key pair (χ,v) such that $v = g_2^\chi$ where $g_2 \in G_2$, χ is the private key and v is the public key. The $Sign$ algorithm computes the signature on input message m_i in G_1 by a full-domain hash function $h : \{0,1\}^* \rightarrow G_1$. The output $\sigma_i = h(m_i)^\chi \in G_1$ is the signature for m_i. The $Aggregate$ algorithm aggregates the signatures σ_1, σ_2, …, σ_t for t different messages m_1, m_2, …, m_t into one signature $\sigma = \prod_{i=1}^{t} \sigma_i$. The $AggVer$ algorithm verifies a signature and works like the $Aggregate$ signature algorithm. For a set $m_1, m_2, \ldots, m_t$ of different messages, and public keys $v_1, v_2, \ldots, v_t$ and a signature σ, the verifier checks that $e(\sigma, g_2) = \prod_{i=1}^{t} e(h_i, v_i)$, where $h_i = h(m_i)$ and e is the bilinear mapping.

IV. Protocols for the Multy-factor Verification of Strong Identity Attributes

In this section, we present the protocols for multi-factor strong identity attribute verification. We first introduce the notion of identity records (IdRs) that provide a representation of user identity attributes. Then, we introduce the protocol for strong identity attributes enrollment that consists of creating secure commitments and in signing them with the private key of the registrar. Finally, we present the protocol to create and verify the aggregate ZKPK of strong identity attributes' committed values.

A. Identity Records

As we mentioned, each principal P in a federation has associated one or more IdRs, each recorded at some

doi:10.4304/jsw.4.7.696-706

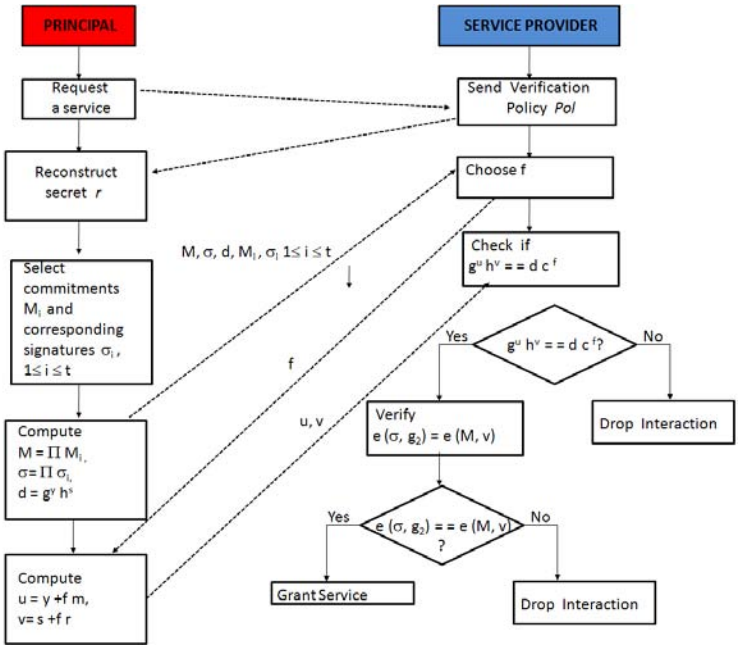

Figure 2. Protocol schema

registrar in the federation. Each IdR in turn consists of several identity tuples, denoted as τ_i, one for each principal's identity attribute m. An identity tuple $\langle \sigma_i, M_i, tag, validity\text{-}assurance, ownership\text{-}assurance, \{W_{ij}\}\rangle$ consists of tag, an attribute descriptor, the Pedersen commitment of m, denoted as M_i, the signature of the registrar on M, denoted as σ_i, two types of assurance, namely *validity assurance* and *ownership assurance* and a set of weak identifiers $\{W_{ij}\}$. M is computed as $g^m h^r$, where g and h are generators in group G of prime order q. G and q are public parameters of the registrar, and r is chosen randomly from $\mathbb{Z}_q$. Validity assurance corresponds to the confidence about the validity of the identity attribute based on the verification performed at the identity attributes original issuer. Ownership assurance corresponds to the confidence about the claim that the principal presenting a given identity attribute is its true owner. There are four levels of assurance: absolute assurance, tagged as A, corresponding to the absolute certainty about the claim; reasonable assurance, tagged as B, corresponding to case when one or more assertions from trusted parties exist regarding the certainty of the claim; unknown assurance, tagged as U, when there is no information to assert the certainty of the claim; and false assurance, tagged as F, denoting that the claim is incorrect. We assume that absolute validity of a given strong identity attribute can only be determined by authorities which have issued the strong identity attributes. This corresponds to value A of the validity-assurance of the associated strong identity attribute. Instead, we mark as B the validity assurance of a strong identity attribute the validity of which has been asserted by a principal, whose identity record has a validity assurance set to A. If no entity other than the principal supports the validity of the strong identity attribute, this attribute is marked with unknown assurance U.

With reference to Example 1, Figure 2 shows an example of an IdR. Here the principal is known as Bob@Registrar1 and has enrolled two strong identity attributes, namely a CCN and SSN.

B. Enrollment of Strong identity attributes

In order to use its identity attributes to prove its identity, a principal has to enroll them to a registrar. The enrollment of a principal at a registrar consists of the following steps:

1) *Registrar parameters.* The registrar runs parameter generation algorithm $GenKey$ that picks a prime q and three multiplicative groups G_1, G_2, G_T of prime order q. Also two generators g_1, h_1 in G_1 such that $log_{g_1} h_1$ and a G_2 group generator g_2 are returned by $GenKey$. Then, the registrar runs algorithm $KeyGen$ to generate the secret key χ, that is, a random number from $\mathbb{Z}_q$ and the public key $v = g_2^{\chi}$. The resulting set of parameters is (G_1, G_2, G_T,g_1,h_1,g_2,v).
2) *Commitment of a value* $m \in \mathbb{Z}_q$. The principal chooses a value $r \in \mathbb{Z}_q$, and computes $M = g_1^m h_1^r$.
3) *Zero-knowledge proof of the committed value.* The principal gives ZKPK of opening the commitment M to the registrar:

$$PK\{(m,r) : y = g_1^m h_1^r, m, r \in \mathbb{Z}_q)\}$$

4) *Signing of the committed value.* After performing the security checks on the committed value (namely the local consistency and federation duplicate detection), the registrar executes the $Sign$ algorithm on the commitment M to output M^{χ} as the signature where χ is the secret key of the registrar.

doi:10.4304/jsw.4.7.696-706

C. Secret Sharing on the Mobile Phones

Suppose that a principal requests a service from a SP which requires the principal to authenticate by proving the possession of identity attributes listed in SP's identity verification policy. To prove the possession of the identity attributes requested by SP, the principal shows that it knows how to open the corresponding commitments. In order for the principal to be able to carry out aggregate ZKPK protocol with SP, the principal needs the random secret r, used to compute the Pedersen commitments of principal's identity attributes. The security of the protocols strongly depends on r so it is necessary to protect it from unauthorized accesses that can occur on mobile devices. Mobile device security can be compromised if the device is lost or stolen, or due to the vulnerabilities of the communication network and/or the device software. To prevent these security threats, we adopt Shamir's secret sharing scheme that allows one to split a secret in n shares and then to reconstruct it if and only if k shares are present. The storage of the shares depends on the specific architecture of the mobile devices. Next we will focus on the Nokia NFC mobile phones that we have used in our implementation.

In our implementation the shares are stored on different mobile phone components and (possibly) on external devices such as a PC or an external storage unit. We split each random secret into four shares s_1, s_2, s_3 and s_4. The first share s_1 is stored in the internal memory of the mobile phone. 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 $P' \oplus P'' = s_2 \bullet P''$ is stored in the phone external memory. The third share s_3 is stored in the smart card integrated in the phone. Finally the fourth secret share s_4 is stored in the user's PC which has to be accessed remotely by the phone. We consider four levels of protection for the secret r that correspond to the number k of shares that are needed to reconstruct r. The possible levels of protection are *low*, *medium*, *medium-high* and *high*. The level of protection *low* requires no splitting of the secret r. In this case, r is stored in the phone smart card. The *medium* level corresponds to a value of k equal to 2. In this case the user has to retrieve two of the four shares s_1, s_2, s_3 and s_4 to obtain the secret r. If the *medium-high* level is chosen, three shares are needed while with level of protection *high*, all the four shares are needed to reconstruct the secret. The level of protection is set by the principal [1] once the principal computes the Pedersen commitments of its identity attributes by using the random secret r. Once set, the level of protection cannot be changed by the principal. When the principal has to prove the ownership of a set of identity attributes to the SP, r needs to be reconstructed. In order to do that, a number of shares according to the level of protection set up by the principal needs to be retrieved and then combined to obtain r.

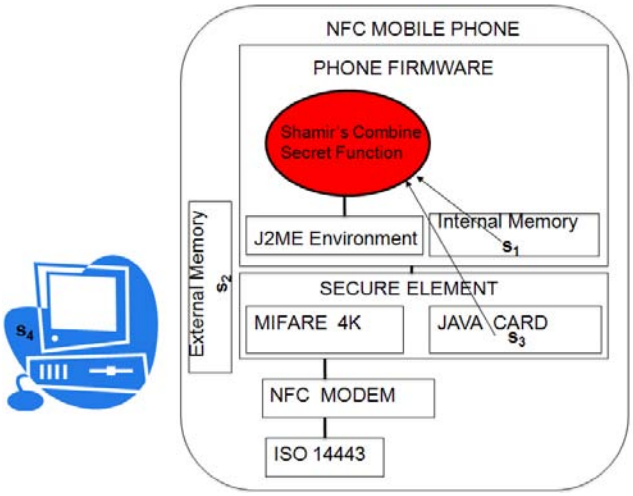

Figure 3. Random Secret Reconstruction

Example 4.1: Suppose that Bob has to prove the possession of credit card number and social security number to SP *e-Follet*. In order to accomplish that, Bob needs to reconstruct the secret r used to compute the Pedersen commitments of his credit card number and his social security number. Bob sets the security level for r to *high* and to retrieve each secret share he has to perform the following steps:

1) Bob retrieves s_1 from the phone internal memory.
2) To retrieve s_2, Bob inputs the secret PIN number P' using the phone keypad. P' is retrieved from the phone external memory and it is used to compute the second secret share $s_2 = P' \oplus P''$.
3) Bob retrieves the secret s_3 from the phone smart card.
4) To retrieve the secret share s_4 stored at the principal's PC, Bob connects to its PC by using the phone

By contrast if Bob sets up a medium security level, he has to retrieve only two shares to obtain the secret r. For example, Bob can decide to get shares s_1 and s_3 from the phone's internal memory and the phone smart card respectively without having to insert any PIN number (see Figure 3).

D. Aggregate zero-knowledge proof of knowledge (AgZKPK)

Once the principal has reconstructed the secret r, he can prove the possession of the identity attributes requested by the SP. The protocol that provides aggregate proof of knowledge of the commitments corresponding to π_{SP} is composed of the following steps:

1) *Principal's aggregation*. Let σ_1, ..., σ_t be the signatures corresponding to the strong identity attributes in SP's identity verification policy. The principal 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 $M = \prod_{i=1}^{t} M_i =$

[1] The specification of the security level and the entering of the PIN are the only steps that need to be carried by the actual principal. The security level can however be set as a default and the end-user does not need to enter it each time it enrolls an identity attribute.

doi:10.4304/jsw.4.7.696-706

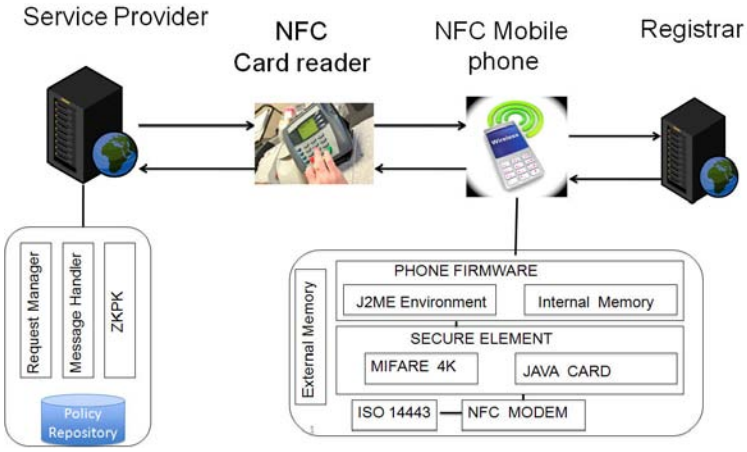

Figure 4. System architecture components.

$g_1^{m_1+m_2+...+m_t} h_1^{r_1+r_2+...+r_t}$. Finally, the principal sends σ, M, M_i, $1 \leq i \leq t$, to the verifier.

2) *Zero-knowledge proof of aggregate commitment.* The principal and SP carry out the following ZKP protocol:
   - the principal randomly picks y, s in $[1, ..q]$ and computes $d = g^y h^s \pmod p$;
   - the principal sends d, σ, M, M_i , $1 \leq i \leq t$, to the SP;
   - SP sends back a random challenge $f \in [1, .., q]$ to the principal;
   - the principal computes $u = y + fm \pmod q$ and $w = s + fr \pmod q$ where $m = m_1 + \ \ldots \ m_t$ and $r = r_1 + \ \ldots \ r_t$;
   - the principal sends u and w to SP;
   - SP accepts the aggregated zero knowledge proof if $g^u h^w = dc^f$. If this is the case, SP verifies the aggregate signature σ.
3) *Verification of aggregate signature.* After the verifier accepts the zero-knowledge proof of the commitments, it checks that the following verifications succeed:
   $M = \prod_{i=1}^{t} M_i$ and $e(\sigma, g_2) = e(M, v)$.

V. System Architecture

We have implemented our protocol on Nokia 6131 NFC [22] mobile phones. NFC enabled devices are gaining popularity because they are easy-to-use for ubiquitous accesses to systems and services. Based on a short-range wireless connectivity, the communication is activated by bringing two NFC compatible devices or tags within a few centimeters from one another.

The system architecture is shown in Figure 4. It consists of four main components: a SP application, an external NFC reader, the Nokia 6131 NFC [22] mobile phone and the registrar. The core architectural component is the NFC mobile phone. It consists of an `Antenna`, for detecting external targets such as tags, external readers, or other Nokia 6131 NFC mobile phones; an `NFC modem`, for sending and receiving commands between antenna, secure element, and phone firmware including J2ME environment; a `Secure element`, for enabling third-party application development using tag/card emulation; `Phone firmware`, for providing mobile phone functions with NFC features; a `SIM card`, for GSM subscription identification and service management; `J2ME environment` included in phone firmware, for enabling third-party application development using Nokia 6131 NFC features; and an `External memory`.

The `Secure element` within Nokia 6131 NFC can store information securely, which can be used for payment and ticketing applications or for access control and electronic identifications. `Secure element` is divided into two subcomponents, `Java Card` area (also referred to as smart card) and `Mifare 4K` area. `Mifare 4K` contains data, whereas smart card application contains an executable program. `Java Card` provides high security environment and executes code, which means it can be used for more complex applications. Therefore, we store in the `Java Card` some of the shares in which the random secret r is split because of the high security provided by `Java Card`. `Secure element` is accessible through `NFC modem` internally from MIDLets and externally by external readers. MIDLets are Java applications running in the J2ME environment. In the next section we describe in details, how we have implement our protocol to manage identity attributes by using MIDLets.

The NFC reader enables the communication between the SP application and the mobile phone. It transmits and receives messages from the NFC cellular phone. The SP application consists of three main modules: `Request Manager`, `Message Handler` and `ZKPK`. The `Request Manager` module parses principals requests and selects from a local repository the identity verification policy that applies to the request. The `Message Handler` module provides all functions supporting the communications between the SP application and the external NFC reader. The `ZKPK` module supports the verification of identity attributes by carrying out aggregate ZKPK. The registrar component provides the functionalities to store clients' identity records and to retrieve the public

doi:10.4304/jsw.4.7.696-706

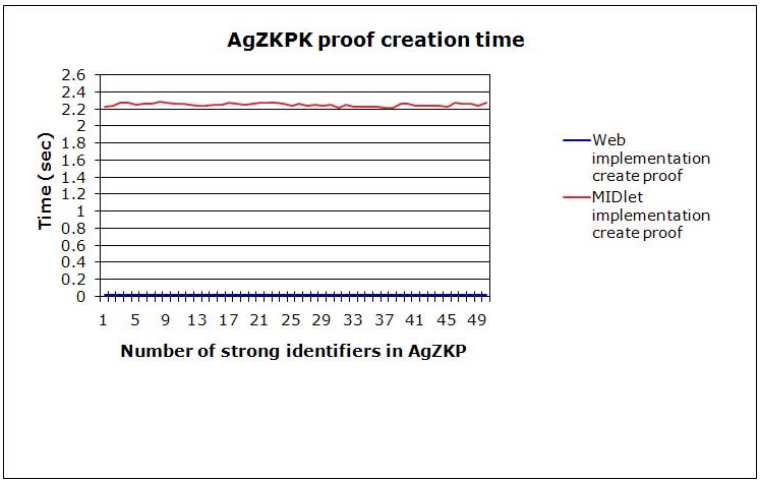

Figure 5. AgZKPK Creation on Midlet versus Web-based implementation

parameters required in the AgZKPK protocol.

VI. Implementation

In this section we describe how we have implemented the multi-factor identity attribute verification protocol on the Nokia 6131 NFC cell phone. We store the principals' IdR in the external phone memory, while the secret r used to compute the secure commitments is saved in `Mifare 4K`. We have implemented a MIDlet that creates the AgZKPK. The implementation of the secret sharing phase is under development.

The MIDlet execution is triggered when the principal's cell phone tag `Mifare 4K` captures the verification policy sent by the SP'NFC reader and the `Mifare 4K` transfers this policy to the cell phones main memory. The MIDlet retrieves from the phone external memory the commitments corresponding to the strong identity attributes requested by the verification policy. Then, the MIDlet runs a new MIDlet which is executed in a protected domain with restricted permissions. This is necessary because the new MIDlet uses cryptographic secrets associated with the strong identity attributes to create the aggregate zero knowledge proof AgZKPK. Once the AgZKPK is computed, the MIDlet sends it to the main MIDlet.

Upon receiving the AgZKPK, the main MIDlet transfers it to the `Mifare 4K` so that it can be read by the external NFC reader.

The MIDLets developed to generate the AgZKPK run on Java 2 Micro Edition (J2ME), a subset of Java 2 Standard Edition (J2SE), which provides environments and APIs for mobile and embedded devices. Since J2ME is aimed at hardware with limited resources, it contains a minimum set of class libraries for specific types of hardware. In our AgZKPK implementation on conventional non-mobile platforms, we used the java.math.BigInteger and java.security.SecureRandom class defined in J2SE to implement secure commitments, but both java.math and java.security package are not supported in J2ME.

Therefore, we have used the third-party cryptography provider BouncyCastle [1], a lightweight cryptography APIs for Java and C# that provide implementation of the BigInteger and SecureRandom classes.

In addition, because of the limited memory size of mobile phone, we reduced the MIDlets' code size by using code obfuscation techniques provided by Sun's NetBeans IDE. Code obfuscation allows one to reduce a file size by replacing all Java packages and class names shorter names, tipically consisting of meaningless characters. For example, a file of size 844KB can be reduced to a size of 17KB.

Moreover, the MIDlets must have read and write privileges on the principal's phone tag `Mifare 4K` in order to enable the communication with the SP's NFC reader. In fact, the SP's verification policy is saved in `Mifare 4K` and then passed to the MIDlet to create the proof. Then, the created AgZKPK is stored in `Mifare 4K` in order to be read by the SP's NFC reader. In order to allow the MIDlets to access `Mifare 4K`, the MIDlets must be signed. To sign the MIDlets we used the Carbide.j tool [2] provided by Nokia that requires a code signing certificate released by a certification authority (CA) to generate the signature. We have also implemented the SP component as a web application using Java and the Apache Tomcat Application Server and the registrar component as a JAVA servlet.

VII. Experimental Results

In this section we present the results of the tests we have performed to evaluate the performance of the multi-factor identity attribute verification protocol implementation on the mobile phone. An aspect that might influence the performance of our protocols is the number of strong identity attributes that are aggregated and verified. Therefore, we have considered the following test cases:

1) we have measured the execution time that the mobile client application takes for the generation of the AgZKP by varying the number of attributes being aggregated from 1 to 50;

doi:10.4304/jsw.4.7.696-706

2) we have measured the execution time the time that SP's interface takes to perform the verification by varying the number of strong identity attributes that are verified from 1 to 50.

We have compared the execution time to create the aggregate ZKPK on the mobile phone with the time to perform the same operation on the VeryIDX web-based implementation [3]. We have measured the execution time in CPU time (milliseconds). Moreover, for each test case we have executed twenty trials, and computed the average execution time over all the trial executions. To run the experiments, we have deployed the SP interface on a Windows XP Professional SP3, processor Intel Dual Core 2.33 GHz and a 2 GB 667 MHz DDR2 RAM.

Figure 5 reports the times required by the VeryIDX mobile phone implementation and by the web-based protocol implementation for generating the aggregate zero knowledge. In both cases, the AgZKPK protocol takes almost constant time for the ZKPK generation even if the number of identity attributes being proven increases. The reason is that the AgZKPK only requires a constant number of exponentiations [4]. Moreover, as expected, the time to create the proof on the mobile phone is higher than the time to perform the same operation on the web-based implementation due to the phone's limited computing power. The average time for the creation of an aggregate proof on the mobile phone is 2.257 seconds, while on the web-based application is around 0.02 seconds. Figure 6 reports the time that the SP application takes to perform the strong identity attributes verification. Notice that the verification time linearly increases with the number of strong identity attributes to be verified. The reason is that during the verification the SP is required to multiply all the commitments to verify the resulting aggregate signature.

VIII. Additional Features of VeryIDX

In addition to the basic protocol for the aggregate verification of identity attributes, a protocol supporting the privacy-preserving verification of conditions against identity attributes has been recently added to VeryIDX [25]. Such protocol uses the Oblivious Commitment-Based Envelope (OCBE) protocol, proposed by Li and Li [17].

The OCBE protocol allows a SP to send a user information encrypted based on the receiver's commitment, so that the user can only decrypt the information if and only if the committed value of the identity attribute verifies a predicate specified by the policy of the service provider. The OCBE protocol assures that the SP will not learn the value of the identity attribute of the user and not even whether the user identity attribute verifies the predicate or not. In our context, depending on the type of services provided, the SP may or may not need to know the outcome of predicate verification. For example, if the services only deal with content provisioning, the SP may encrypt the contents with one or more keys, and then transfer the keys to the user by using the OCBE protocol. If the user identity attribute verifies the condition, the user will be able to extract the keys and decrypt the contents; otherwise, the user will not be able to extract the keys, and thus be unable to decrypt the contents. In such context, the SP does not need to know whether the user's identity attribute verifies the predicate. For services different from content provisioning, the SP may have to know the outcome of the verification in order to provide the requested service and inferences on the actual value of the identity attributes may thus be possible. However because OCBE supports inequality conditions, the predicates may be written so that only minimal information about the identity attribute be inferred. For example, the fact that a user's age verifies a predicate of the form "age = 18" allows the SP to infer that the age of the user is indeed 18. However, a predicate of the form " age $>$ 18" leaks only a lower bound on the actual user age. In terms of performance, the experimental results show that OCBE protocol is quite efficient in verifying equality conditions on receipts; however, the performance of the inequality conditions' verification needs to be improved.

Another recent extension to VeryIDX is related to the problem of naming heterogeneity for identity attributes. Naming heterogenity occurs when service providers and clients use different vocabularies to denote identity attribute names. In particular, whenever a client sends a request for a resource or a service to a service provider, the client may not understand which identity attributes it has to provide to satisfy the service provider identity verification policy. Therefore, the client is not able to prove its identity and thus not able to access the service/resource. To address such problem, the VeryIDX identity verification protocol has been integrated with a protocol to match the identity attribute names referred by the service provider's policies and client's vocabularies. Such protocol uses look up tables, dictionaries, and ontology mapping techniques [24].

IX. Related Work

The most relevant proposals in the area of digital identity management are CardSpace [9], OpenID [23], Liberty Alliance [16], Shibboleth [30], WS-Federation [32], and Credentica [11]. We can classify these proposals into user-centric and federated-DIM frameworks. CardSpace, OpenID and Credentica are user-centric while like Liberty Alliance, Shibboleth, and WS-Federation are federated digital identity management frameworks. The main difference among such proposals is in the protocol they use to verify users' identity. In CardSpace, the user is presented with a set of information cards representing the digital identities that satisfy a SP's policy. When the user selects the information card to be presented to the SP, the IdP which has issued the card to the user, releases to the user a security token encoding claims corresponding to the selected information card. The user, then, passes such token to the SP. Credentica supports an identity verification protocol similar to the one by Card Space: the SP verifies the user's identity based on an ID Token issued by an IdP to the user encoding claims about the

doi:10.4304/jsw.4.7.696-706

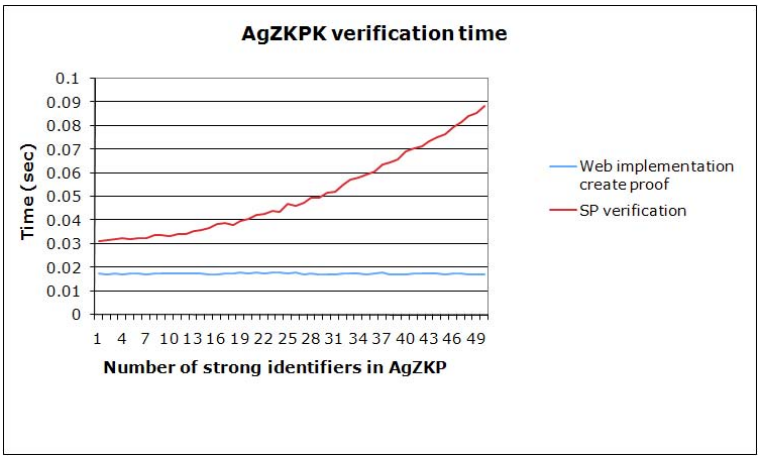

Figure 6. AgZKPK Verification versus Creation

identity the user has chosen to present to the SP. In Open ID, when a user accesses a SP's web site, he/she provides an OpenID that is the URL of a web page listing the user's IdPs. The SP selects an IdP and the browser is redirected to the IdP's web page. If the IdP successfully verifies the identity of the user, the browser is redirected to the designated return page on the SP web site along with an assertion that the user is authenticated. Our approach for digital identity verification is aligned with user-centric DIM frameworks because the user can view through the client interface which attributes are returned by the SP as result of the matching process.

Other relevant proposals, like Liberty Alliance, Shibboleth, and WS-Federation, are based on the notion of federated identity. Federations facilitate the use of users' attributes across trust boundaries to establish a federation context for the users. In Liberty Alliance, a federation is represented by a circle of trust that is constituted by service providers (SPs) and IdPs having mutual trust relationships. The circle of trust enables single sign-on (SSO) across different SPs' web sites. When the authentication of a user is requested by a SP, the IdP authenticates the user and then issues an authentication assertion. The SP validates the assertion issued by the IdP, and determines whether or not it should be accepted. Once a user has been authenticated, the user is able to sign-on on other service sites without having to be re-authenticated at each site. In [18], it is discussed how Liberty's open standard can enable secure delivery of mobile services.

Shibboleth is another initiative supporting cross-domain SSO. WS-Federation does not propose another identity verification protocol but specifies how to use WS-Trust, WS-Security, and WS-SecurityPolicy to provide mechanisms for identity brokering, attribute discovery and retrieval, authentication, and authorization claims between federation partners, and protecting the privacy of these claims across organizations. In our approach, we assume a "relaxed" notion of federation in that a trust relationship does not need to exist between all the SPs and IdPs.

Other identity management initiatives have gained importance with the rapid adoption of second-generation mobile telecommunication systems, leading to the growth of m-commerce [15], [26].

Rannenberg et al. [26] propose an approach to mobile DIM based on the GSM infrastructure and the information stored at the Subscriber Identity Module (SIM). Each SIM contains a secret unique symmetric key (specified as k) stored together with the ID of the subscriber. This key is only shared with the authentication centre (AuC) of that GSM network operator that issued the SIM. When a GSM subscriber tries to log on to the GSM network (usually when he switches on the phone) the SIM passes the subscribers ID to the AuC. The AuC then checks whether the SIM also knows the respective k: The AuC sends a random challenge message to the subscribers phone. The SIM in the phone has to encrypt that challenge message with the k and send it back to the AuC. The AuC encrypts the same message with the local copy of the k and compares the results. If they match, the subscriber is granted access to the GSM network.

Jendricke et al. [15] propose a system that selects the identity information a principal can use depending on the context (location etc) and warns the principal when he/she accidentally disclose sensitive identity information.

X. Conclusion

This paper proposes protocols for managing identity attributes in cellular devices and supporting their secure and privacy preserving usage. The protocols are based on aggregate zero knowledge proof and aggregate signature on strong identity attributes' commitments. We have implemented the protocols on the Nokia NFC cellular phones and we have shown that the execution time to create the aggregate proof of knowledge is almost constant with respect to the number of strong identity attributes being aggregated. We are currently completing the implementation of our prototype system by developing a MIDLet supporting the secret sharing phase of our protocol. We are working to improve the usability of the shares retrieval process to reconstruct the random secrets so that the users intervention is minimized.

doi:10.4304/jsw.4.7.696-706

ACKNOWLEDGMENT

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 intereted 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] Bouncy Castle Crypto APIs, http://www.bouncycastle.org/

[2] Development tools, http://www.forum.nokia.com/main/resources/tools_and_sdks/carbide/index.html

[3] Bhargav-Spantzel, A., Woo, J., Bertino,E.: Receipt management- transaction history based trust establishment, In Proceedings of the 2007 ACM workshop on Digital identity management, pp. 82–91, New York, NY, USA.

[4] Bhargav-Spantzel, A., Squicciarini, A.C, Xue, R., and Bertino, E.: Practical Identity Theft Prevention using Aggregated Proof of Knowledge, Technical report CERIAS TR 2006-26, 2006.

[5] Bhargav-Spantzel, A., Squicciarini, A.C, Bertino, E.: Establishing and Protecting Digital Identity in Federation Systems. Journal of Computer Security, IOSPress, 14(3), pp. 269–300, (2006)

[6] Boly, J., Bosselaers, A., Cramer, R., Michelsen, R., Mjolsnes, S., Muller, F., Pedersen, T.P, Pfitzmann, B., de Rooij, P., Schoenmakers, B., Schunter, M., Vallee, L., Waidner, M.: The ESPRIT Project CAFE - High Security Digital Payment Systems, ESORICS, pp. 217–230,1994.

[7] Boneh, D., Gentry, C., Shacham, H., Lynn, B.: Aggregate and verifiably encrypted signatures from bilinear maps, In Proceedings of Advances is Cryptology , Eurocrypt'03, LNCS. Springer-Verlag, 2003.

[8] Camenisch, J., Stadler, M.: Efficient Group Signature Schemes for Large Groups, Advances in Cryptology, CRYPTO '97, pp. 410–424, 1997.

[9] D. Chappel. Introducing Windows CardSpace. http://msdn.microsoft.com/en-us/library/aa480189.aspx.

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

[11] Credentica. www.credentica.com.

[12] Help for lost and stolen phones. http://news.bbc.co.uk/1/hi/technology/4033461.stm.

[13] Dix, A., Rodden, T., Davies, N., Trevor, J., Friday, A., Palfreyman, K. :Exploiting space and location as a design framework for interactive mobile systems, ACM Transactions on Computer Human Interaction, 7(3), pp. 285–321, 200, New York, NY, USA.

[14] Fiege, U., Fiat, A., Shamir, A.: Zero knowledge proofs of identity, In Proceedings of the nineteenth annual ACM conference on Theory of computing, pp. 210–217, New York, NY, USA, 1987.

[15] Jendricke, U., Kreutzer, M., Zugenmaier, A., Mobile Identity Management, UBICOMP 2002: Workshop on Security in Ubiquitous Computing, 2002.

[16] Liberty Identity Web services Framework. http://www.projectliberty.org/libery/resource_center/specifications/liberty_alliance_id_wsf_1_1_specifications.

[17] J. Li and N. Li. OACerts: Oblivious attribute certificates. IEEE Transactions on Dependable and Secure Computing, 3(4):340–352, 2006.

[18] Liberty Alliance Project. Tier 2 Business Guidelines: Mobile Deployments. www.projectliberty.org/liberty/content/download/989/6958/file/LibertyMobileBusinessGuidelines1_2.pdf

[19] Mjolsnes, S.F., Rong, C.: Localized Credentials for Server Assisted Mobile Wallet, In Proceedings of International Conference on Computer Networks and Mobile Computing, Los Alamitos, CA, USA, 2001.

[20] Met initiative. http://www.mobiletransaction.org.

[21] Near Field Communication Forum, http://www.nfc-forum.org.

[22] Nokia Forum. Nokia 6131 NFC Technical Description. http://www.forum.nokia.com.

[23] Open ID. http://openid.net/.

[24] Paci, F., Ferrini, R., Musci, A., Steuer Jr K., Bertino, E. An Interoperable Approach to Multi-factor Identity Verification. To appear on IEEE Computer Special Issue April 2009 - Interoperable Identity Management Systems.

[25] Federica Paci, Ning Shang, Elisa Bertino, Kevin J. Steuer, Jungha Woo. Secure Transactions' Receipts Management on Mobile Devices. In Proceedings of Symposium on Identity and Trust on the Internet (IDtrust Symposiums), NIST, Gaithersburg, MD, April 14-16, 2009.

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

[27] SET Secure Electronic Transaction Specification Book 1: Business Description, 1997.

[28] Wolfe, A.: Toolkit: Java is Jumpin',Queue, 1(10), pp. 16–19, 2004

[29] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[30] Shibboleth. http://shibboleth.internet2.edu/.

[31] TechRepublic. Identify and reduce mobile device security risks. http://articles.techrepublic.com.com/5100-22_11-5274902.html.

[32] WS-Federation, Web services Federation Language. http://specs.xmlsoap.org/ws/2006/12/federation/.

Federica Paci is a post doc in the Department of Computer Science at Purdue University. Her research interests include access control for service oriented architectures, digital identity management and privacy in social networks. She received a PhD in Computer Science from the University of Milan in Febraury 2008.

Elisa Bertino is a full professor in the Department of Computer at Purdue University and Research Director of the Center for Information and Research in Information Assurance and Security. Her research interests include privacy techniques for databases, digital identity management, policy systems, and security for Web services. She received a PhD in Computer Science from University of Pisa.

doi:10.4304/jsw.4.7.696-706

Sam Kerr is an undergraduate student in the Department of Computer Science at Purdue University. His research interests include digital identity management.

Anna Squicciarini is an assistant professor at Penn States College of Information Sciences and Technology. Her main research interests include trust negotiations, privacy, and access control for grid computing systems. Currently, she is exploring security issues in the context of social networks and is developing trust negotiation protocols in peer-to-peer platforms. Squicciarini earned her Ph.D. in Computer Science from the University of Milan, Italy, in March 2006.

Jungha Woo is a graduate student in the Department of Computer Science at Purdue University. His research interests include digital identity management.

doi:10.4304/jsw.4.7.696-706