Mobile Netw Appl (2007) 12:79–92
DOI 10.1007/s11036-006-0008-7

Privacy-enhanced, Attack-resilient Access Control in Pervasive Computing Environments with Optional Context Authentication Capability

Kui Ren · Wenjing Lou

Published online: 2 December 2006

Abstract In pervasive computing environments (PCEs), privacy and security are two important but contradictory objectives. Users enjoy services provided in PCEs only after their privacy issues being sufficiently addressed. That is, users could not be tracked down for wherever they are and whatever they are doing. However, service providers always want to authenticate the users and make sure they are accessing only authorized services in a legitimate way. In PCEs, such user authentication may include context authentication in addition to the entity authentication. In this paper, we propose a novel privacy enhanced anonymous authentication and access control scheme to secure the interactions between mobile users and services in PCEs with optional context authentication capability. The proposed scheme seamlessly integrates two underlying cryptographic primitives, blind signature and hash chain, into a highly flexible and lightweight authentication and key establishment protocol. It provides explicit mutual authentication and allows multiple current sessions between a user and a service, while allowing the user to anonymously interact with the service. The proposed scheme is also designed to be DoS resilient by requiring the user to prove her legitimacy when initializing a service session.

K. Ren (✉) · W. Lou
Department of Electrical and Computer Engineering,
Worcester Polytechnic Institute, Worcester,
MA 01609, USA
e-mail: kren@ece.wpi.edu

W. Lou
e-mail: wjlou@ece.wpi.edu

Keywords pervasive computing environment (PCE) · security · privacy · access control · context authentication

1 Introduction

Pervasive computing environments (PCEs) with their interconnected devices and abundant services promise great integration of digital infrastructure into many aspects of our lives [1, 33]. The huge number of communicating devices will provide seamless access to multiple dynamic networks at any location. Companies, organizations and individuals are increasingly depending on electronic means to process information and provide relevant services in order to take advantage of ambient computing intelligence in PCEs [2, 4–6]. Inevitably, many of these information transactions will be sensitive and critical [17] and thus, it is essential to enforce *access control* to prevent information leakage and service abuse, and to stop malicious attacks. In other words, dynamic access to services should be granted only based on pre-established (direct or indirect) trust between users and service providers. To this end, trust relationship by means of mutual authentication between users and service providers should be established *prior* to the access of services. Traditional authentication which focuses on identity authentication may fail to work in PCEs, partly because it conflicts with the goal of user privacy protection and partly because the assurance achieved by entity authentication will be of diminishing value [17]. For instance, a service provider may only concern whether the accessing user is authorized or not, but has limited interest in who she is in many non-critical scenarios. Meanwhile, services themselves should be authenticated to users. Users will only accept authenticated information from genuine services they intend to interact with to avoid potential deception and other malicious attacks. The importance of authenticating services is discussed in [17].

One big forthcoming challenge for actually deploying pervasive computing services on a significant scale is how to have adequate provision for handling *user privacy*, which is considered as one of the fundamental security concerns that are explicitly identified by a series of laws [3]. In environments with significant concentration of "invisible" computing devices gathering and collecting the identities, locations and transaction information of users, users should rightly be concerned with their privacy. At the same time, the physical outreach of pervasive computing makes preserving user's privacy a much more difficult task [10, 13, 34]. Some of the user privacy issues that should be treated in PCEs has been pointed out in [8], including location privacy, connection anonymity and confidentiality. We further clarify the scope of privacy in PCEs as follows.

Anonymity: The real identity of a user should never be revealed from the communications exchanged between the user and a server unless it is intentionally disclosed by the user. Different communication sessions between the same user and service should not be linkable. *Context privacy:* In principle, users' context information (e.g., location, duration, type of service request, etc.) should be protected against both outsiders and even service providers they interact with. Neither the service nor other users of the service should be able to learn the exact context information of a user, unless the user decides to disclose such information or the service require context authentication before service access is granted. *Confidentiality* and *integrity:* The interactions between a user and a service should have both confidentiality and integrity protections whenever such protections are required.

The quests for authentication/access control and user privacy protection conflict with each other in many aspects, and the problem is more complex in PCEs. On the one hand, the service generally depends on the user identity information and corresponding pre-established trust relationship as well as the service contract between them to accomplish user authentication and conduct access control; some times, services even require user context information authentication. For example, a printing service may require the actual physical presence of users to prevent potential service abuse. On the other hand, the users want keep their privacy as possibly as they can. They do not want to be tracked by the service for wherever she is and whatever she does. Hence, to achieve a good trade-off between the two poses a great challenge to security designers.

In this paper, we propose a user privacy preserving access control scheme at the application level to address the security and user privacy concerns in PCEs. The proposed scheme is implemented at the application level without relying on any underlying system infrastructure such as the *Lighthouse* or *mist routers* etc., required by many other approaches [2, 7, 8, 13]. The proposed scheme provides explicit mutual authentication between the two parties, while at the same time allowing the mobile user to interact with desired service anonymously without revealing her identity. The scheme seamlessly integrates two underlying cryptographic primitives, *blind signature* and *hash chain*, into a highly flexible and lightweight authentication and key establishment protocol. The scheme possesses many desirable security properties, such as anonymity, non-likability, non-repudiation, accountability, differentiated services access control, etc., with very low protocol complexity (refer to Section 5). In addition, the proposed scheme is designed to be DoS resilient by requiring the user to prove her legitimacy when initializing a service session. The proposed scheme also supports context authentication: user location information can be verified by the service as part of access control requirements.

The rest of this paper is organized as follows. In Section 2 we describe the system architecture of PCEs and introduce the cryptographic primitives used in our scheme. We present in detail the proposed scheme in Section 3. And in Section 4, several security enhancements to the basic scheme are given. Then we discuss the security features and the performance of the proposed scheme in Section 5. Finally, we review the related work in Section 6 and conclude the paper in Section 7.

2 System architecture and cryptographic primitives

Generally, a PCE consists of three type of entities: mobile users, services and back end authentication servers, in addition to the underlying wired and wireless communication infrastructures. To make the system architecture more scalable and flexible, a broker can be introduced between the user and service. Both users and services can interact with brokers to subscribe and distribute services. The system architecture is shown in Fig. 1. However, in this paper, we focus on the direct interactions between users and services for the sake of simplicity. Our proposed access control scheme is designed to secure the interactions among three types of

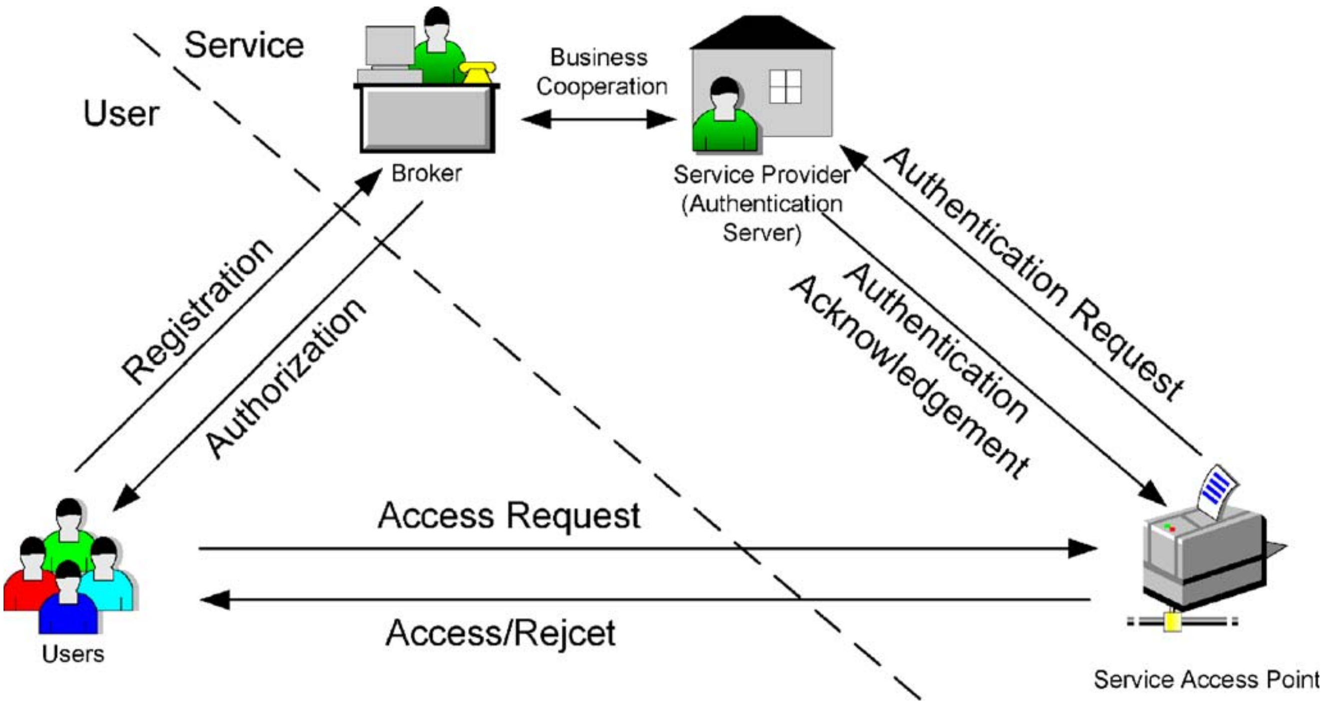


Figure 1 System architecture

entities, i.e., the user, the service and the authentication server.

The proposed scheme is based on two cryptographic techniques, blind signature and hash chain. A brief review of the two techniques is provided as follows.

Blind signature: Blind signature scheme [14] is a variation of digital signature scheme in which the content of a message is disguised from its signer. Blind signature schemes can be implemented based on a number of well known digital signature schemes, such as RSA [28]. To produce a signature on a message, a user first *blinds* the message with a *blinding function* f, typically by combining it with a random *blinding factor* k, and then forwards the blinded message to the signer A. The signer signs the blinded message using a standard signing algorithm, say $S_A(x)$ which denotes the signature of A on m, and sends the result back to the user, who then *unblinds* it with an *unblinding function* g to obtain the signer's signature on the original message. The algorithm is designed such that $g(S_A(f(m))) = S_A(m)$.

Blind signatures are used to provide non-linkability, which prevents the signer from linking blinded message it signed to the unblinded version that it may be called upon to verify. In this case, the signed, blinded value is unblinded prior to verification in such a way that the signature remains valid for the unblinded message. This can be useful in schemes where anonymity is required. Blind signature schemes find a great deal of use in applications where sender privacy is important. This includes various *digital cash* schemes and voting protocols [15, 16].

Hash chain: One-way hash function h is a powerful and yet computationally efficient cryptographic tool. By applying $h()$ repeatedly on an initial value m, one can obtain a chain of outputs $h^j(m)$. These outputs can be used in the reverse order of generation for the purpose of authentication: $h^{j-1}(m)$ can be proven to be authentic if $h^j(m)$ has been proven to be authentic due to the one-wayness property of hash function. Hash chains together with signatures are widely used in micro-payment schemes such as Payword, *i*KP and Netcard [27]. In such schemes, the effect of a digital signature is reused many times over subsequent messages (containing pre-images of a specific hash). The concept of hash chain was first proposed for use in authentication scheme by Lamport [30]. Recently, Weimerskirch et al. adopted hash chain technique for efficient user recognition based on weak authentication [31].

3 The proposed scheme

3.1 Design consideration

In this paper, we consider the scenario that a mobile user wants to be able to dynamically access a large

Table 1 Notation

U	A mobile user that is usually identified by her public key and can belong to some user group(s).
S	Service provider or its authentication server which is used to authenticate the user for the purpose of access control.
M	User group manager that can act as an agent for group members.
TTP	Trusted third party, an entity which is trusted by both the mobile user and the service provider.
SID	A service type identifier, which describes a selected subset of the available service pool that can be accessed by a particular mobile user, is identified by a unique public key. Different users may belong to the same service type.
P	Service access point. For wireless networking service, it represents the access point (AP).
$PubK_A$, $PriK_A$	Public and private key pair of entity A.
K_{AB}	Shared secret key between entities A and B.
m, X_m	Message m and its corresponding ciphertext.
(m_0, m_1)	Concatenation of two messages.
$\{m\}_{PubK_A}$	Encrypt message m with the public key of entity A.
$\{m\}_{PriK_A}$	Sign message m with the private key of entity A. If not otherwise stated, message m is recoverable.
$\{m\}_{K_{AB}}$	Encrypt message m by symmetric key algorithm with the secret key shared between entities A and B.
$h()$	A cryptographic secure one-way hash function, or one-way hash function in short, such as *MD5* [29].
$h_{K_{AB}}(m)$	A cryptographic secure MAC algorithm, computing the message digest of message m with key K_{AB}.
$h^j(m)$	Hash message m j times: $h^1(m) = h(m), h^j(m) = h(h^{j-1}(m)), j = 2, 3, 4, ...$
r_A	A nonce generated by entity A, usually it is 64-bit pseudo random number.
C^j, $j = 0, 1, 2, \ldots$	A series of authorized credentials used by an entity to obtain service access permission.
$CertA$	A certificate which binds entity A with her public key $PubK_A$.

number of different kinds of services available in PCEs in an efficient and secure manner. Therefore, the following design considerations should be achieved simultaneously: (1) Provide explicit mutual authentication between the mobile user and the service, and establish fresh session keys to secure the interaction if necessary; (2) Allow mobile users to anonymously interact with the services; (3) Enable differentiated service access control among different users and provide easy accountability; (4) Have high efficiency with respect to both communication and computation costs and management overheads; (5) Provide the service the ability to authenticate user location information as part of access control requirements, if mutual agreement on such context authentication has been achieved between the user and the service in priori; (6) Be DoS-resilient to protect the service from the attacks launched by malicious users.

Note that we assume users are capable of manipulating the source addresses of the outgoing Medium Access Control (MAC) frames. This assumption is prerequisite for anonymous communication otherwise one can easily identify a user based on her unique MAC address. The notation used in protocol description is listed in Table 1.

3.2 The basic scheme

3.2.1 Phase I: user authorization

In Phase I, a mobile user first subscribes the service through explicit authentication between each other. This is typically done through some out-of-band non-cryptographic technique. The mobile user needs to register herself as a legal user of some service types the service provider provides. She then obtains the public keys of the services of which she is entitled to use. She also needs to obtain a certificate $CertU$ which binds her identity U to her public key $PubK_U$, signed by the private key of the service provider $PriK_S$. Next, the mobile user executes the user authorization protocol to obtain authorized service access credentials from the service, which are used as the security anchor in the subsequent mutual authentication processes whenever a mobile user attempts to access a service. The user authorization protocol is based on blind signature [14], which hides the association between the authorized credential and the mobile user's real identity. Therefore, the mobile user can anonymously access the services later. Moreover, to increase protocol efficiency, the user authorization protocol also enable a mobile user to obtain a series of authorized credentials through one protocol run by using hash chain technique.

More specifically, the proposed user authorization protocol contains two steps: (1) *credential generation*, and (2) *credential authorization*. In Step 1, a mobile user U performs as follows:

Step 1:

1. Computes the anchor value C^0 of the credential chain.
2. Computes the end value of the credential chain with length n: $C^n = h^n(C^0)$.

3. Blinds the end value C^n as $C_U = \{r'_U\}_{PubK_{SID}} \times C^n$, where r'_U is a fresh nonce.

Further, Step 2 operates as below:
Step 2:

1. U sends authorization request to service S: $U, C_U, CertU, SID$.
2. S verifies $CertU$ using $PubK_S$, and signs on C_U: $C_S = \{C_U\}_{PriK_{SID}} = r'_U \times \{C^n\}_{PriK_{SID}}$.
3. S replies authorization confirmation C_S to U.
4. U decrypts and verifies U, S.
5. U computes C_S/r'_U and obtains a valid signature pair $(C^n, \{C^n\}_{PriK_{SID}})$.

The mobile user first computes the anchor value C^0 of the credential chain. Next, the end value of the credential chain is generated via iterated hash operation. The length n can be adjusted to the proper value depending on the actual frequency of usage and storage capability. Then the mobile user blinds credential chain tail C^n by using blind signature technique. Next, the mobile user sends out the blinded C^n for authorization. And the authentication server signs C^n with the private key of the requested service type, and returns the signed credential back to the mobile user.

– Besides the general public key $PubK_S$ for user authentication purpose, the service provider maintains a pool of public keys corresponding to difference service types. We assume that the mapping between the service type identifier SID and its corresponding public key is clear to the mobile user. The authorized credentials of different service types are actually signed by the different public keys. This allows for differentiated access control in the subsequent stage. If the scope and the meaning of the service type is carefully defined and the services are therefore well classified, the combinational usage of several authorized credentials at the same time can further improve the ability to enable higher level differentiated service access control. This will also improve the flexibility and scalability of the proposed scheme.
– Once the signed credential is returned to the mobile user, the computation of C_S/r'_U indeed results in a valid signature on C^n due to the property of blinded signature. Therefore, after the protocol execution, the mobile user holds a verifiable authenticator—credential C^n and its signature. Although the authentication server doesn't know what the value of C^n is at the time it signs it, the authenticity of C^n can be verified by the signature. Therefore, once the authenticator is submitted to the authentication server, the authentication server will be able to verify and grant the service request. However, it still has no information about who the user is, except for her requested service type.
– Although the authentication server signs only the nth credential C^n, the remaining hash chain values through C^0 to C^{n-1} are authorized implicitly at the same time, due to the one-wayness nature of the hash function. Note that the values of the credential hash chain should never be revealed to any third party.
– The mobile user can also generate serval different credential hash chains at the same time, and get each C^n signed by the authentication server simultaneously in one protocol run. Hence, the protocol efficiency can be further improved, as well as the flexibility.

The user authorization protocol allows the mobile user to obtain the authorized credentials from service provider. Note that the user authorization protocol runs only when the mobile user's authorized credentials are exhausted or for the first time registration. The user authorization protocol is highly flexible. It can be accomplished via both online and off-line approaches. It can also be accomplished through the agent of the mobile users. We can easily imagine that the network manager or administration staff can acquire the authorized credentials from the service providers on behalf of the users in a company and then distribute them to the user. This delegable feature greatly improves the usage flexibility of the mobile users and allows dynamic authorization. It also significantly simplifies the management overheads at the service side. The authentication server is now able to manage only one certificate for each user group, instead of those of all group members.

3.2.2 Phase II: service access

The service access protocol allows a mobile user to safely enjoy different kinds of services she is authorized to in PCEs at anytime, from anywhere without disclosing any of her context information unless she agrees to do so. The service access protocol works as follows.

1. U generates a fresh nonce r_U, and computes $\{r_U, C^j\}_{PubK_S}$, $0 < j \le n$. U sends access request message: $\{SID, \{r_U, C^j\}_{PubK_S}\}$ to P.
2. P relays the received access request message to S.
3. Upon receiving the access request message, S decrypts r_U, C^j, and verifies C^j. Then, S replies access acknowledgement message $\{r_U, C^j\}$ to P.
4. Upon receiving the access acknowledgement message, P generates a fresh nonce r_P, and computes

$K_{UP} = h(C^j, r_P, r_U, 0)$ and $K'_{UP} = h(C^j, r_P, r_U, 1)$. P then sends its own access acknowledgement message: $\{r_P, \{r_U, P\}_{K_{UP}}\}$ to U.

5. Upon obtaining this message, U computes $K_{UP} = h(C^j, r_P, r_U, 0)$ and $K'_{UP} = h(C^j, r_P, r_U, 1)$. Then U decrypts and verifies r_U, C^j and P.
6. U encrypts the data m_0 as $X_{m_0} = \{m_0\}_{K'_{UP}}$, and computes the corresponding MAC as $h_{K_{UP}}(X_{m_0})$. And the final data traffic is $\{r_P, r_U, X_{m_0}, h_{K_{UP}}(X_{m_0})\}$.

– In the access request message 1, the mobile user encrypts a fresh nonce r_U and authorized credential C^j with service's public key which is used for authentication purpose. The encryption operation has dual purposes: (1) keep the secrecy of r_U and C^j from eavesdropping; (2) service authentication, because only the user's intended legitimate service can decrypt the message correctly. The SID is provided to claim user's capability to access the targeted service.
– When the authorized credential chain is used for the first time, i.e., $C^j = C^n$, the mobile user should send both C^n and its signature for authentication. In this case, the access request message 1 would be: $\{r_U, C^n, \{C^n\}_{PriK_{SID}}\}_{PubK_S}$. Each credential is used exactly once, that is, used in only one session and is obsoleted afterwards. Hence, an authorized credential chain of length n can be used to access the services for n sessions before all credentials are exhausted. The use of a different credential for each session is necessary to defend against the replay attack and possible double spending problem (e.g., for accountability).
– The authentication of the submitted credential at the service side is as follows: If a credential is submitted together with its signature, that is, $(C^n, \{C^n\}_{PriK_SID})$, the authentication server verifies the signature using corresponding public key by referring to SID in the same message. A negative result will trigger an access denial message, sent to the service access point. A positive result confirms the validity of the submitted credential. A duplication check on C^n should be first executed before signature verification to prevent a potential double spending of C^n. Upon success of the verification, the authentication server saves C^n according to its service type. Recall that in the last subsection, we pointed out that each different public key is used to bind a particular service type. Thus, although the authentication server couldn't know who the user is, it does know this user's capability to access the services, that is, whether she is eligible for the requested service or not through the submitted credential. Hence, a differentiated service access control is easily realized.
If the submitted credential is a single value C^j, the back end server simply verify whether $h(C^j)$ matches the currently stored credential whose belonging hash chain is indexed by C^n. The authentication server then updates the currently stored C^{j+1} with C^j. The remaining operation is the same as above. Note that, for each different credential chain, the authentication server stores exactly two values: the signed C^n and the newest (current) C^j. This is for dual purposes: (1) for the ease of credential authentication on C^j, $j < n$; (2) prevent potential double spending of the credentials for all C^js.
– The traffic between the service access point and its back end server is assumed to be protected by private or previously established secure tunnel, which is beyond the design range of this protocol.
– The service access point has no responsibility for user authentication. It simply defers this job to its back end server. The computation and management overheads at the service access point are minimized and little storage capability is required: (1) no public key operations; (2) no long term key and certificate management; (3) session keys are discarded once the session is terminated; (4) hash and symmetric key operations only. Hence, it is very simple and efficient, which could greatly decreases its cost and helps wide deployments.
– The service access point and the mobile user compute the fresh session keys independently, and authentication server has no control over the computed session keys. The fresh nonce used in key generation guarantees the freshness of the session keys. Two fresh session keys are generated. One is for encryption and the other is for integrity protection, i.e., generating the message authentication code (MAC) [29].
– The fresh nonce r_P, r_U are then used by the mobile user and the service access point to identify the session between them, that is, binding the two communication parties and the exchanged traffic together. We can see that there is no way to identify the session between the two otherwise, because both two parties may interact with many other parties at the same time, especially for service access point.
– The one-time usage feature of the authorized credentials and its linkage with the service type provide effective accountability. Similar to the micropayment schemes, the accounting mechanism can be easily incorporated into the system in nearly the same manner. We point out that double spending

of the authorized credentials actually does not affect the system security as proved in Section 5. Hence, the choice between one-time or multiple usage of the authorized credentials can be a simple policy decision. It also can be dynamically switched according to real situations.

4 Protocol enhancements

4.1 DoS resilience

In the proposed basic service access protocol, the authentication server S is required to perform an expensive public key operation (i.e., a decryption operation) upon receiving an access request message. Then, a denial-of-service (DoS) attack is one in which an attacker (e.g., a malicious non-authorized user) sends a large number of access request message to S. The purpose is to exhaust its resources and render it less capable of serving legitimate users. The proposed basic service access protocol is vulnerable to this attack because a malicious user can send arbitrary access request messages without any cost on computation and S has no way to identify whether or not the received message is bogus.

Having observed this vulnerability, we approach this attack by requiring the user to prove its legitimacy when initializing a service session. At the same time, we allow the authentication server to be able to easily identify a bogus service request message. To this end, we introduce a public/private key pair $(\overline{PubK_{SID}}, \overline{PriK_{SID}})$, which is used to identify a service of type SID. $\overline{PriK_{SID}}$ is pre-distributed to the user during user authorization process. More specifically, the authorization confirmation message can be modified to include both C_S and $\overline{PriK_{SID}}$. Note that $(\overline{PubK_{SID}}, \overline{PriK_{SID}})$ is different from $(PubK_{SID}, PriK_{SID})$. We further make use of the beacon message which is periodically broadcast by a service access point for declaring its existence. The content of the beacon message is defined as below:

$$SID, P, \{SID, r'_P, P\}_{\overline{PubK_{SID}}},$$

where r'_P is a fresh random nonce. And we now require a valid service request message to include r'_P. That is, a service request message is modified as

$$\{SID, P, r'_P, \{r_U, C^j\}_{PubK_S}\},$$

instead of $\{SID, \{r_U, C^j\}_{PubK_S}\}$. Now a bogus service access message can be easily identified by checking on the value of r'_P at the service access point, even before it reaches the authentication server. This is because only the legitimate users of this service can decrypt the message and obtain r'_P. In fact, when a mobile user wants to access the service after receiving the beacon message, she has to perform the following operation: (1) decrypts beacon message using the corresponding private key $\overline{PriK_{SID}}$; (2) verifies whether or not SID and P are correctly included in the encrypted message; (3) obtains r'_P. Hence, before a valid service access message can be sent, the user has to prove her legitimacy. Note that beacon messages are periodically broadcasted, and the value of r'_P also changes accordingly each time. Thus, a replayed message could only possibly happen within one period. By controlling the duration of the period, the DoS attack caused by replaying messages can be greatly mitigated. Moreover, once an attack is identified, the beacon message broadcast period can always be dynamically shortened so that the replayed messages can be more efficiently rejected.

4.2 Support for context authentication

In PCEs, many services may require physical presence of the users when accessing the services. Therefore, the access control scheme should also be able to provide a mechanism to authenticate user's location information.

In order to do so, we embed a time-varying location ID (LID) information into the beacon message. Here, we assume that the beacon message is range limited and is broadcast through the techniques such as bluetooth and infrared channel. Therefore, only when a mobile user is physical nearby the service access point can she receive such a beacon message. After the location ID information is embedded, the beacon message is now as follows:

$$SID, P, LID, \{SID, LID, r'_P, P\}_{\overline{PubK_{SID}}}.$$

Consequently, a service access request message is now

$$\{SID, P, r'_P, \{r_U, C^j\}_{PubK_S}, MAC_U\},$$

where MAC_U is now a Message Authentication Code with LID as the key:

$$MAC_U = h_{LID}(SID, P, r'_P, \{r_U, C^j\}_{PubK_S}).$$

Now a service access request message is first checked at the service access point in the following sequence: (1) checking on the value of r'_P; (2) verifying the validity of MAC_U. Only when both of the checking results are positive, could this access request message be relayed to the authentication server for further checking. Then after the session is successfully initialized, both the user and the service access point periodically update their shared session keys K_{UP} and K'_{UP} as follows as the session continues:

$$K_{UP} = h_{LID}(K_{UP}),\ K'_{UP} = h_{LID}(K'_{UP}),$$

where LID is the location ID information contained in the current beacon message. These periodical updates on the session keys guarantee the continuous physical presence of the mobile user at the service access point. This is so because there's no way for a user to get an updated LID except for being around the service access point. Note that the update period of LID is independent to that of r'_P in our setting so that the system can more adequately react to the potential different types of attacks.

4.3 Dispute resolution

Since our protocol allows anonymous authentication of the mobile user, one concern from the service provider is that the service might be abused by the users beyond their control. For example, one might try to access the service using some stolen C^j. In the following, we present a dispute resolution protocol, which is used to prove a user's legitimacy while the service provider suspects that the user is potentially illegal by using some stolen credentials (e.g., the service provider has observed some abnormal access pattern of the user). The dispute resolution protocol allows the service provider to challenge the mobile user, and the mobile user responses to prove her legitimacy again without exposing her context privacy to the service provider.

Our approach is as follows. First, instead of using a random anchor value C^0, we embed unique information into it, so that only a particular user can generate it. We let

$$C^0 = h(r''_U, U, \{U, r''_U\}_{PriK_U}),$$

where r''_U is a fresh nonce generated by the mobile user. And she also signs her own identity together with the fresh nonce using her private key, that is, $\{U, r''_U\}_{PriK_U}$. Clearly, the signature contained in C^0 provides non-repudiation property. This is true because only the mobile user herself can generate it and the fresh nonce guarantees its freshness. Next, we introduce an off-line Trusted Third Party (TTP) for referee purpose. The idea is that when a dispute happens, the service provider submits the suspected credential to TTP, and the mobile user is requested to prove to TTP her ownership of the submitted credential. Technically, the proof of the ownership is done through showing the knowledge about the whole credential chain corresponding to the questioned credential. From our user authorization protocol, we know that the anchor value C^0 of the credential chain is computed over the mobile user's real identity and a corresponding signature signed over its identity and a fresh nonce. Therefore, only the mobile user herself can generate a valid pre-image value of C^0. Obviously, it is computationally impossible to forge such a message as long as the underlying cryptosystems is secure. So, if the mobile user proves that she could compute C^0 from the above identity information, it is indeed a proof of her ownership about the credential. TTP is therefore informs the service provider that the mobile user is indeed the authorized one, which otherwise would be a failure information. This accomplishes our protocol goal. We outline our dispute resolution protocol in Table 2.

Table 2 Dispute resolution protocol

S		TTP		U
	$\xrightarrow{C^j}$ 1) Resolution request			
			$\xrightarrow{C^j}$ 2) Proof request	
			$\xleftarrow{U, r''_U, \{U, r''_U\}_{PriK_U}, C^0, CertU}$ 3) Proof ack.	
		1. Verify $\{U, r''_U\}_{PriK_U}$. 2. Check $C^0 \overset{?}{=} h(U, r''_U, \{U, r''_U\}_{PriK_U})$. 3. Check $C^j \overset{?}{=} h^j(C^0)$.		
	$\xleftarrow{True/false}$ 4) Resolution ack.		$\xrightarrow{True/false}$ 4) Resolution ack.	

Note that in our dispute resolution protocol, the mobile user's privacy is well protected. The service provider only gets to know whether the questioned mobile user is an authorized one or not. Still the service provider has no clue about who the user is. At the same time, although TTP knows who the mobile user is, it has no information about user's service usage profile. We exclude the colluding possibility of TTP and the service provider, which is beyond the scope of this paper. The ability to resolve the disputes between the mobile user and the service provider also helps prevent potential illegal credential transfer among the users in some sense, as we will discuss further in Section 5.

4.4 Extension for out-of-order requests

Sometimes a mobile user might want to launch multiple sessions simultaneously. Note that if the multiple sessions are with respect to different service types, or if the multiple sessions are of the same service type but come to the authentication server in the same order as they were originated, the proposed protocol can handle them well. However, if the multiple concurrent sessions are with respect to a single service, but for some reasons (e.g., unexpected network problems, DoS attacks), the access request messages arrive out of order at the back end authentication server, one or more legitimate requests will be deemed illegal by the service access protocol we described above. In this subsection, we extend the service access protocol to deal with such out-of-order arrival of the access requests at the authentication server.

Our solution is a sliding window based extension to the credential authentication procedure at the authentication server. Recall that in the previous setting, each hash chain stores two values: C_n, used as the index of the hash chain, and C_j, the most recently used hash value. A submitted credential is hashed only once and compared with C_j. In the extension, the back end authentication server is allowed to store up to $k+1$ hash values for each hash chain: C_n and a window of up to k hash values. A submitted credential may be hashed up to $k-1$ times, once a hashed value is found equal to a value already in the window, the credential may be accepted. The extension works as follows. Obviously, at the beginning when all the requests come in order, there is only one value kept in the window—the most recently used value C_j. At this time, when the next credential C^{j_1} comes: (1) if it comes in order, namely $j_1 = j-1$ or $C^j = h(C^{j_1})$, the window will move forward by one place, which makes C^{j_1} the first value in the window while the rest of the places in the window remains empty; (2) if $j-k < j_1 < j-1$, which means that C^{j_1} is hashed more than once to be equal to C^j, the window remains where it is and the value C^{j_1} is saved at the corresponding location, i.e., $(j-j_1+1)$th place in the window; (3) otherwise, if a match is not found after k hashes, the credential is rejected. Similarly, when there are multiple credentials in the window, the following process applies for the next arrived credential C^{j_2}: (1) if C^{j_2} equals to any of the existing credentials, it is a double submission and the corresponding session should be rejected; (2) if $j_2 = j-1$,[1] the credential is valid and the window moves forward by one place which makes C^{j_2} the first credential in the window. Further, if its next position (j_2-1) is not empty, the window will continue to move forward until it reaches a value C^{j_3} whose next position (j_3-1) is empty. This operation will ensure

[1]We still assume C^j is the first credential in the window.

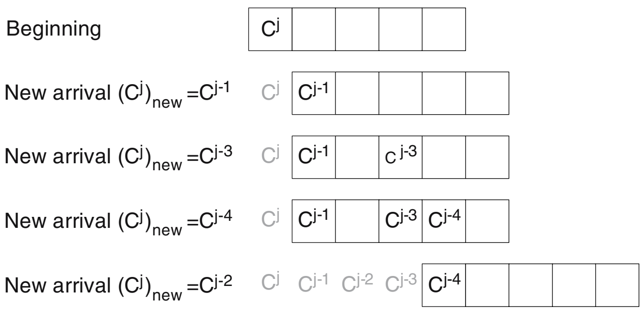

Beginning
C^j
New arrival $(C^j)_{new}=C^{j-1}$
C^j
C^{j-1}
New arrival $(C^j)_{new}=C^{j-3}$
C^j
C^{j-1}
C^{j-3}
New arrival $(C^j)_{new}=C^{j-4}$
C^j
C^{j-1}
C^{j-3}
C^{j-4}
New arrival $(C^j)_{new}=C^{j-2}$
C^j C^{j-1} C^{j-2} C^{j-3}
C^{j-4}

Figure 2 An illustration of the sliding window based credential authentication procedure (for $k=5$)

that the second location in the window is always empty; (3) if $j-k < j_2 < j-1$, the window remains unmoved and the value C^{j_2} is saved at the corresponding empty location in the window; (4) otherwise the credential is simply rejected. Figure 2 shows an example of how an out-of-order arrival of requests is handled by this extension.

Note that in this extension, k $(k << n)$ is the maximum distance between two access requests that go out of order. The requests are of the same type and from the same mobile user. Since the time needed to complete the whole authentication is in a scale of milliseconds or at worst seconds, it is very unlikely that a mobile user can launch a large number of sessions and have them go out of order. Therefore this number could be very small. Note that the number of concurrent sessions the user could have is not limited by k. A user may still have a large number of concurrent sessions as long as she initializes each of them with a small time separation.

In the proposed service access protocol, we assume that an incomplete session will always be due to the failure of the mobile user. The packets lost in the network can be resent by reliable link/transport protocols or failure notification will be returned to the source. If a mobile user gets no reply after sending an access request message, it would resend the same message again until obtaining a corresponding reply. However, to make the scheme highly robust to unexpected network problems, we may allow the window to move forward when the last position of the window is filled while the second position has yet not filled. On the other hand, as long as an access request message is successfully received and processed (replied) by the service access point (i.e., as long as an access acknowledgement message is sent back to the mobile user by the service side), the submitted credential is void thereafter even if the mobile user may fail to continue the subsequent session. Note that the access acknowledgement message may be resent many times before the session is aborted by the

Table 3 Simplified service access protocol

U		P		S
	$\xrightarrow{SID, C^j}$ 1) Access request			
			$\xrightarrow{SID, C^j}$ 1) Access request	1) Verify C^j.
			$\xleftarrow{\textit{Grant/deny}}$ 2) Access ack.	
	$\xleftarrow{\textit{Service/reject}}$ 2) Access ack.			

service access point. Since we generally consider micro-payment case, dropping a single credential occasionally does not affect much on accountability.

4.5 A simpler version

Sometimes the mobile user's targeted service is not related to the data traffic in PCEs. In this case, we have a much simplified and extremely efficient protocol as outlined in Table 3. Only hash and match operations are needed. Of course, if the credential chain is used for the first time, an additional signature verification is required. Also note that in such contexts, the authentication of the service may not be relevant, and thus is skipped in the protocol.

5 Analysis of the proposed scheme

5.1 Security related properties of the proposed scheme

The proposed scheme exhibit many nice security related properties as discussed below:

– *Mutual authentication:* In the proposed scheme, the mobile user is authenticated based on her authorized credential, in the sense that the service knows the user is indeed legal and authorized. The service authenticates itself to the user through its public key certificate and by showing its knowledge of the corresponding private key. The mutual authentication is highly necessary in the PCEs as discussed before to prevent potential malicious attacks from the both sides.
– *User context privacy:* The users' context privacy is well protected by the proposed scheme, only absolutely necessary information is known to the service, i.e., users' service type, in order to grant appropriate access. Through the blind signature technique, the mobile users could be authenticated anonymously without disclosing any other information. All the service side knows is that some legal users are accessing some particular services. The information is also protected against the outsiders. No third party has the ability to acquire the user's context information, as all the interaction traffic are well protected.
– *Dispute resolution (non-repudiation):* By carefully integrating the mobile user's fresh identity signature into the pre-image of the anchor value of the authorized credential chain, dispute resolution function is enabled in our proposed scheme. Because only the given mobile user can generate its fresh identity signature, and therefore, the authorized credential chain, explicit non-repudiation is also provided. This is a good resort to resolve the potential disputes arising between the two parties.
– *Non-linkability:* Ideally, non-linkability means that, for both insiders (i.e., service) and outsiders: (1) Neither of them could ascribe any session to a particular user; (2) Neither of them could link two different sessions to the same user [36]. In the proposed scheme, ideal non-linkability is achieved with respect to outsiders. Because the authorized credential is never transmitted in plaintext, and is always combined with fresh nonce in the message, an outsider cannot ascribe a session to a particular user, neither can he ascribe two sessions to a same user. Hence, users' transaction profiles are well protected. On the other hand, using hashing chain could allow the service provider to link up to n sessions using the hash values from the same chain to a same user, where n is the length of the hash chain. However, the service provider can not ascribe such information to a particular user due to the underlying blind signature technique used. In addition, such linkage is limited to n sessions only, there's no relationship among different hash chains. Therefore, there's no inter-hash-chain information can be accumulated by the service provider. Hence, users' transaction profile can still be well protected from the service provider.
– *Accountability and non-transferability equivalency:* In the proposed scheme, the credentials are authorized only when the mobile user is explicitly authenticated. The one-time usage property of the authorized credentials prevents double spending problem and further provides good accounting capability which allows the accounting function be

easily incorporated.[2] Furthermore, from the service point of view, the proposed scheme provides equivalent non-transferability, which means that even the credentials are delegated among users, no harm is done to the service provider in the sense that the authorized user is responsible for all the service received by her authorized credentials. This novel feature greatly reduces the service abuse problem worried by the service providers. Using blind signature [19] alone can not provide this property because there's no way to prevent the double spending problem and hence, no way to prevent service abuse problem.

– *Data traffic protection:* The user operation protocol generates fresh session keys to protect the interaction data traffic between the mobile user and the service. Data *confidentiality* and *integrity* can be provided based on the symmetric cryptography.
– *Differentiated service access control:* By classifying the mobile users into different service types, differentiated service access control is enabled in our scheme. Different mobile users are authorized accordingly based on their belonged service types. User authorization is therefore, accomplished in a differentiated way. Moreover, the combinational usage of the different credentials may help to provide high level differentiated service access control, which is beyond the scope of this paper.

We compare our scheme to other similar approaches that are intended to provide anonymous interactions between the users and the services in Table 4. The advantage of our scheme is obvious.

5.2 Performance of the proposed scheme

Despite the number of desirable security properties provided, the proposed scheme is extremely lightweight. We analyze the overheads introduced in this subsection.

– *Management Overhead:* The proposed scheme involves minimum management overheads (e.g., human interaction). The service provider needs to manage one certificate per user and the corresponding user profile. Due to the delegation property, this number can be significantly reduced to that of the user groups (i.e., one user per group). On the other side, each mobile user needs to manage the certificates of the service provider and the different service types she belongs to.

[2]For example, we can limit the amount of service one credential is entitled thus making the amount of service measurable.

Table 4 Protocol security features comparison

	This paper	[19]	[22]
Concrete protocol	Yes	Yes	No
Mutual authentication	Yes	Yes	No
User context privacy	Yes	Yes	Not to the services
Dispute resolution	Yes	No	Yes
Non-linkability	Yes	No	Not to the services
Non-transferability Equivalency	Yes	No	N/A
Data confidentiality	Yes	Easy to obtain	No
Message integrity	Yes	Yes	No
Accounting capability	Yes	No	Yes
Differentiated service Access control	Yes	No	Yes
DoS resilience	Yes	No	N/A

– *Storage Overhead:* While the protocol is running, the back end authentication server stores two values (C^j, C^n) for each currently in-use credential chain and one value (C^n) for each of the used but unexpired chain. The service access point maintains no permanent user information or key information. Each service access point only stores two session keys per session, besides two nonce to identify that session. The mobile user stores the two random nonce (r'_U, r''_U), and the credential chains authorized to her (e.g., $C^0, \ldots, C^n$ and signature of C^n). In addition, the mobile user should store two nonce and two session keys for each ongoing session. The method to store a hash chain can have a computation and storage trade-off. The mobile user can also choose to store the anchor value and the current value of the hash chain only and compute the needed value on-the-fly.
– *Communication Overhead:* The service access protocol requires two-way and four messages to accomplish mutual authentication and session key establishment between the user and the service. Note that two-way is the minimum number for any authenticated key establishment protocol to fulfill its goal. Therefore, the proposed scheme is highly efficient in the sense of communication overhead.
– *Computational overhead:* The mobile user performs one public key operation per session and all the remaining are hash and symmetric cryptographic operations. The public key operation can be done off-line. The authentication server also needs to do one public key operation per session, and one additional signature verification for each authorized

Table 5 Protocol computational overheads comparison

		Public key oper.	Sig. veri.	Nonce gen.	Hash oper.	Sym. key oper.
Ours	U	1(off-line)	0	1	2	3
	P	0	0	1	2	3
	S	1(online)	$1/n$	0	0	0
[19]	U	1(off-line)	0	0	1	1
	P	0	1(online)	0	1	1
	S	1(online)	1(online)	0	1	1

credential chain (which could be used for n sessions). The service access point is completely exempted from performing public key operations. We compare in Table 5 the computational overhead of the proposed scheme with the scheme proposed in [19]. It is observed that the proposed service access protocol is extremely lightweight.

Notice that our protocol is much more efficient than [19], despite of so many additional security features as discussed above. In [19], the authentication server needs to perform one signature verification every session in addition to one public key decryption. The server therefore, could be the bottleneck of the whole system, due to the potential large amount of concurrent transactions. Moreover, the service access point is required to perform one public key operation for each session, which is also a heavy burden to it. For instance, a wireless access point will have a great trouble to perform public key operations for every user in every session due to its constrained computation capability.

6 Related work

Recently, quite a few papers have been published to address the new security and privacy challenges in PCEs [7–9, 11, 13, 17, 19, 22–24, 26, 32, 34, 35]. However, most of these results fall in the scope of establishing general security framework and identifying general security requirements, without providing concrete security protocols.

Some of these efforts focused on designing specific security infrastructures to protect user context privacy like location information from service providers. The MIST system [7, 8] provides user anonymity through an overlay network assuming the existence of a *Lighthouse*, which keeps all information of all the users. In addition, performance degradation is unavoidable for systems that utilize MIX-network style approach [12]. A proxy-based scheme can be found in [11]. Another recent infrastructure based approach, LEXP, can be found in [26]. Some efforts try to maximize user privacy by restricting the access to users' context information. Hengartner et al. suggested an architecture to filter out user context information [20].

The remaining efforts mostly focused on identity manipulation approaches, with most of which originated from Chaum's anonymous ID based scheme in 1985 [16]. This general scheme allows users to interact with different services anonymously, using pseudonyms. Pseudonyms can not be linked, but are formed in such a way that a user can prove to one service about his relationship with another using a "statement." Such a statement is called credential. An in-depth description and analysis of different pseudonym systems can be found in [25]. Jendricke et al. [22] introduced an identity management system in PCEs where a user is issued multiple identities, and the user uses them depending on applications. The paper only presented a general framework of using virtual identities to protect user privacy while performing access control and authentication, but didn't give any concrete protocol. More recently, He et al. [19] presented a simple anonymous ID scheme for PCEs, which is a direct application of Chaum's blind signature technique [14]. However, the scheme suffers from several drawbacks as discussed in Section 5. In [32], we proposed our basic solution without considering DoS resilience and user context authentication, and proved the protocol correctness using BAN logic. Henrici and Muller [21] utilized hash functions to recompute identifiers of a RFID device every time it sends a request to service providers. Their intention were to protect the location privacy of RFID devices. Another approach proposed by Weimerskirch et al. uses hash function to realize efficient weak authentication [31]. In order to avoid leakage of user's MAC address or IP address at the lower levels, Gruteser et al. [18] came up with a method to hide user's MAC address with anonymous IDs so that the user can not be tracked in a wireless LAN environment.

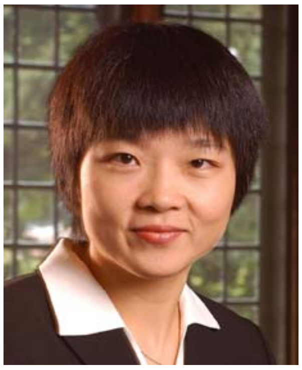
7 Conclusion

In this paper, we proposed a privacy preserving authentication and access control scheme to secure interactions between mobile users and services in PCEs. On the one side, the proposed scheme provides explicit mutual authentication between a mobile user and a service; on the other side, it allows the mobile user to anonymously interact with the service. Hence, the proposed scheme successfully satisfies concerns of both parties—security and privacy. The scheme integrates two cryptographic primitives, blind signature and hash chain, into a highly flexible and lightweight authentication and session key establishment protocol. The scheme enables both differentiated service access control and dispute resolution. The proposed scheme is also designed to be DoS resilient by requiring the user to prove her legitimacy when initializing a service session. The proposed scheme also supports context authentication: user location information can be verified by the service as part of access control requirements.

References

1. Microsoft Research. Easy living. http://research.microsoft.com/easyliving/
2. GAIA—active spaces for ubiquitous computing. University of Illinois, Urbana-Champaign, IL. http://choices.cs.uiuc.edu/gaia/
3. Location privacy protection act and other privacy related law. http://www.techlawjournal.com/cong107/Privacy
4. MIT project oxygen. http://oxygen.lcs.mit.edu/
5. National Institute of Standards and Technology (NIST), Pervasive Computing SmartSpace Laboratory. http://www.nist.gov/smartspace/
6. Georgia Institute of Technology. The aware home. http://www.cc.gatech.edu/fce/ahri/
7. Al-Muhtadi J, Campbell R, Kapadia A, Mickunas D, Yi S (2002) Routing through the mist: privacy preserving communication in ubiquitous computing environments. In: International conference of distributed computing systems (ICDCS 2002), Vienna, Austria
8. Al-Muhtadi J, Campbell R, Kapadia A, Mickunas D, Yi S (2002) Routing through the mist: design and implementation. Technical report UIUCDCS-R-2002-2267, March 2002
9. Al-Muhtadi J, Ranganathan A, Campbell R, Mickunas M (2002) A flexible, privacy-preserving authentication framework for ubiquitous computing environments, ICDCS Workshops 2002, Vienna, Austria, pp 771–776
10. Al-Muhtadi J, Ranganathan A, Campbell R, Mickunas M, (2003) Cerberus: a context-aware security scheme for smart spaces, PerCom, Fort Worth, TX, pp 489–496
11. Burnside M et al (2002) Proxy-based security protocols in networked mobile devices. In: ACM SAC 2002, Madrid, Spain
12. Camenisch J, Lysyanskaya A (2001) Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. In: Advances in cryptology, EUROCRYPT. LNCS 2045, Springer, Berlin Heidelberg New York, pp 93–118
13. Campbell R, Al-Muhtadi J, Naldurg P, Sampemane G, Mickunas M (2002) Towards security and privacy for pervasive computing. In: ISSS, Tokyo, Japan, pp 1–15
14. Chaum D (1982) Blind signatures for untraceable payments. In: Chaum D, Rivest RL, Sherman AT (eds) Advances in cryptology proceedings of crypto, vol 82. Plenum, New York, pp 199–203
15. Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2): 84–88
16. Chaum D (1985) Security without identification: transaction systems to make Big Brother obsolete. Commun ACM 28(10):1030–1044
17. Creese S et al (2004) Authentication for pervasive computing. In: Security in pervasive computing 2003. LNCS 2803, Springer, Berlin Heidelberg New York, pp 116–129
18. Gruteser M, Grunwald D (2003) Enhancing location privacy in wireless LAN through disposable interface identifiers: a quantitiative analysis. In: WMASH'03, San Diego, CA
19. He Q et al (2004) The quest for personal control over mobile location privacy. IEEE Commun Mag 42(5):130–136
20. Hengartner U, Steenkiste P (2003) Access control to information in pervasive computing environments. In: Proc. of 9th workshop on hot topics in operating systems (HotOS IX), Lihue, HI, May 2003
21. Henrici D, Muller, P (2004) Tackling security and privacy issues in radio frequency identification devices. In: PERVASIVE 2004, LNCS 3001. Springer, Berlin Heidelberg New York, pp 219–224
22. Jendricke U, Kreutzer M, Zugenmaier A (2002) Pervasive privacy with identity management. In: The 1st workshop on security, UbiComp 2002, Göteborg, Sweden
23. Jendricke U, Kreutzer M, Zugenmaier A (2002) Mobile identity management. In: The 1st security workshop, UBICOMP, Sep. 2002, Göteborg, Sweden
24. Langheinrich M (2002) A privacy awareness system for ubiquitous computing environments. In: UbiComp 2002. LNCS 2498. Springer, Berlin Heidelberg New York, pp 237–245
25. Lysyanskaya A, Rivest R, Sahai A, Wolf S (1999) Pseudonym systems. In: Proceedings of selected areas in cryptography 1999. Springer, Berlin Heidelberg New York, pp 184–199
26. Nakanishi K, Nakazawa J, Tokuda H (2003) LEXP: preserving user privacy and certifying location information. In: The 2nd workshop on security (Ubicomp2003)
27. Park D (2001) Cryptographic protocols for third generation mobile communication systems. PhD thesis, Queensland University of Technology, Australia
28. Rivest R, Shamir A, Adleman L (1978) A Method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21:120–126
29. Rivest R (1992) The MD5 Message Digest Algorithms. IETF RFC 1321
30. Lamport L (1981) Password authentication with insecure communication. Commun ACM 24(11):770–771
31. Weimerskirch A, Westhoff D (2003) Zero common-knowledge authentication for pervasive networks. In: Proceedings of selected areas of cryprotgraphy (SAC 2003), Ottawa, Ontario
32. Ren K, Lou W, Deng R, Kim K (2006) A novel privacy preserving authentication and access control scheme in pervasive computing environments. IEEE Trans Veh Technol 55(4):1373–1384, July
33. Weiser M (1991) The computer for the 21st century. Sci Am 265(3)94–104
34. Wu M, Friday A (2002) Integrating privacy enhancing services in ubiquitous computing environments. In: Workshop on security in ubiquitous computing, 4th international UBICOMP, Göteborg, Sweden
35. Zugenmaier A, Hohl A (2003) Anonymity for users of ubiquitous computing. In: Security workshop, UbiComp 2003, Seattle, October 2003
36. Xu S, Yung M (2004) k-anonymous secret handshakes with reusable credentials. In: Proc. of ACM conference on computer and communications security (CCS) 2004, pp 158–167

Kui Ren received his B.Eng and M. Eng both from Zhejiang University, China, in 1998 and 2001, respectively. He was a research assistant at Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences from Mar. 2001 to Jan. 2003, then in Institute for Infocomm Research, Singapore from Jan. 2003 to Aug. 2003, and in Information and Communications University, South Korea from Sept. 2003 to June 2004. Currently he is a PhD student at ECE department of Worcester Polytechnic Institute. His research interests include ad hoc/sensor network security, wireless mesh network security, Internet security, and security and privacy in ubiquitous computing environments.



Wenjing Lou is an assistant professor in the Electrical and Computer Engineering department at Worcester Polytechnic Institute. She obtained her Ph.D degree in Electrical and Computer Engineering from University of Florida in 2003. She received the M.A.Sc degree from Nanyang Technological University, Singapore, in 1998, the M.E degree and the B.E degree in Computer Science and Engineering from Xi'an Jiaotong University, China, in 1996 and 1993 respectively. From December 1997 to July 1999, she worked as a Research Engineer in Network Technology Research Center, Nanyang Technological University. Her current research interests are in the areas of ad hoc and sensor networks, with emphases on network security and routing issues.