Embedding Distance-Bounding Protocols within Intuitive Interactions

Laurent Bussard and Yves Roudier

Institut Eurecom
Corporate Communications
2229, route des Crêtes BP 193
06904 Sophia Antipolis
(France)
{bussard,roudier}@eurecom.fr

Abstract. Although much research was conducted on devising intuitive interaction paradigms with pervasive computing devices, it has not been realized that authentication, an important need in this context, has a strong impact on the ease of use. More specifically, distance-bounding protocols are necessary in some of the most interesting scenarios in pervasive computing. This article describes a drag-and-drop interaction paradigm that enables strong authentication by embedding such a protocol within personal authentication tokens. This article also discusses how this paradigm can be used as the basis for performing user-friendly pervasive multi-party secure interactions.

1 Introduction

The pervasive computing paradigm relies on the establishment of relations between several appliances embedding invisible electronic processors and human users authenticated through some digital identifier, potentially none of these entities trusting each other. Authentication requirements are thus clearly renewed in the pervasive computing world: it is the interaction of several entities, some virtual, and some real, that should be secured, rather than the identity of a single virtual party. In addition, the context, time, and location of an interaction is essential to some pervasive transactions.

Security properties like trust, non-repudiation, access control, or privacy have to be redefined in relation with authentication. We have shown in [2] that existing approaches like [7] and [4] are not sufficient when strong authentication is expected and we have proposed a solution based on distance-bounding protocols. This paper further investigates how such protocols may be embedded within an intuitive interaction paradigm.

Section 2 describes the security and usability requirements that are specific to pervasive computing, in particular the new authentication requirements in pervasive systems. Section 3 presents the distance-bounding protocols necessary for authentication and their implications for artifacts. Section 4 finally discusses how to devise a user-centric and user-friendly interaction embedding such a protocol and presents several security mechanisms adapted to pervasive security scenarios.

2 Application Requirements: Usability vs. Security

It is well known that security can have a strong impact on usability. Security leads to more complex protocols, requires users action such as entering a password, and sometimes relies on additional tokens such as smart cards. In pervasive computing, when interacting with artifacts, the situation is even worse: as shown in [2] and reminded in Section 3, touching artifacts during authentication is the only way to ensure strong yet flexible authentication. Transparently securing pervasive computing environments, which have important usability and security requirements, is challenging.

2.1 Usability of the Environment

Interactions between users and artifacts can be complex but have to stay as transparent and intuitive as possible. If some service requires the explicit interaction of users with real-world objects, this interaction should be rendered as intuitive as possible.

Discovery and advertisement [12] approaches impose the selection of virtual representations of surrounding devices: it is obviously neither transparent nor intuitive to select a printer in a list on one's PDA when it stands in front of the user and could be directly selected by touching it.

Physical contact with the device whose service is required may be constraining in the sense that it has to be within physical reach, but is extremely intuitive for the user; rooms containing devices out of reach might even be equipped with authentication switches alike light switches. However, plain physical contact lacks the details provided by a computer interface, which suggests that combining both approaches might be relevant.

Multi-party interactions involving multiple devices and/or multiple users should be possible in pervasive environments. Most paradigms essentially focus on two party interactions scenarios, but scenarios of sales typically involve two people and one device for instance.

2.2 Security Requirements

Securing interactions between users and artifacts is mandatory as soon as resources have to be protected. Access Control is necessary to verify the rights of users interacting with the environment. It can be based on access control lists or on capabilities (e.g. authorization certificates [10]).

Pervasive computing environments are shared by numerous users. It is the reason why it is necessary to account for their respective actions and to keep a proof that some critical interaction happened. In this context, non-repudiation can be required for establishing user liability.

Artifacts have an important role in pervasive computing and it is thus necessary to have a way to verify their characteristics. For instance, an interaction may require that the rights of an artifact or who its owner is be verified. The establishment of trust relationships has to be investigated as well. For instance, Public Key Infrastructures (PKI) are useless in open environments (i.e. without a priori trust).

In addition, standard security protocols that are used in the Internet, Intranets, or e-commerce rely on connected models where it is always possible to reach a Trusted Third Party (TTP) such as a Bank or a Certificate Revocation List (CRL) for security checks. In pervasive computing environments, some artifacts are put together and start collaborating over personal area networks. Personal devices as well as self-standing appliances do not necessarily have access to remote servers or trusted third parties. In this case, security checks have to be adapted.

Last but not least, finding a way to ensure security without bothering users is not trivial. Prompting the user for credentials such as passwords each time he interacts with his environment is not a credible solution and does not fit with the expected transparency or "pervasiveness". Section 4 proposes a new user-centric approach based on personal tokens with contact interface that are used to touch devices in order to dynamically and securely create relationships. In this solution, the user is authenticated and his rights are used during the operation.

2.3 Strong Authentication in Pervasive Environments

Pervasive computing environments seem to head for totally wireless interactions, supposedly simplifying the interactions between artifacts. Unfortunately, wireless networks make it easier to snoop on some protocol or to attack it by inserting malicious traffic. *Artifact authentication* is thus an essential security feature.

Two divergent points of view have been developed in former works. The first one is concerned with providing a unified and informative interface to all services and artifacts. It relies on a purely virtual representation, accessible on a PDA for instance, in which the user chooses the service or specific artifact that he wants to make use of. In such an approach, traditional techniques of authentication and key distribution apply quite straightforwardly. This approach however does not bridge the gap between the theoretical representation of the environment and the reality of it: no proof is given that the printer in front of the user is the one that the PDA says should be in this room.

The other approach attempts to authenticate physical artifacts in the user environment. The problem of this category of interaction thus becomes to know if the user is really dealing with the right artifact or if the real service is embedded in this artifact. Work on constrained channels [7] and the Smart-Its Friend project [4] proposed solutions for authenticating parameters that characterize a context of interaction. As acknowledged in these works however, those approaches are *"only usable in situations that do not require strong authenti-*

cation". Furthermore, the feedback provided to the user is very limited if only existent.

Depending on the criticality of the resources accessed or of the goods exchanged, it can be sufficient to base the access control or the transaction on weak authentication. However, pervasive computing will lead to numerous payment or micro-payment schemes and access to critical resources will take place in some applications. This paper aims at answering the following question: *Is it possible to define user-friendly interactions in pervasive computing environments requiring strong authentication?*

3 Pervasive Authentication and Distance

Common authentication and access control problems are not enough to deal with in the most open pervasive computing environments, where nothing is known about the surrounding devices. This section discusses the importance of determining the location together with classical authentication and the implication of this requirement in pervasive computing environments.

3.1 Attacking Location = Attacking Pervasive Authentication

Authenticating a user or finding his privileges commonly means deploying asymmetric cryptography and certificates, for instance, identity certificates and authorization certificates. Certificates contain the user's public key and some of his attributes, those data being signed by a trusted entity, the certification authority. When the user wants to access a resource, he presents his certificates and, thanks to a challenge-response protocol, proves that he knows the private key corresponding to the public key embedded in the certificate. In this scheme, protecting the private key is mandatory because anybody having access to it can impersonate the owner of this key i.e. use his identity and his rights. Figure 1 describes this attack: a token M_1 protects the private key K_{S-M_1} and is necessary to access resources. If K_{S-M_1} can be extracted, clones $(C_1 \cdots C_n)$ can be created and can access resources that M_1 is authorized to access.

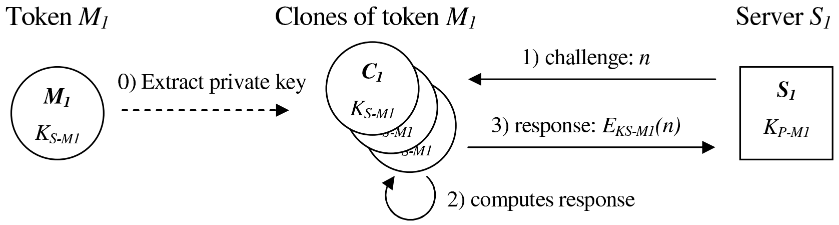

Fig. 1. Impersonation attack by stealing the private key

Generally, the private key is protected by a tamper-resistant token (e.g. a smart card) ensuring that the private key cannot be stolen. Moreover, it is necessary to generate the key pair within the token to ensure that only the public key can be known outside. Combining asymmetric cryptography and tamper-resistant modules ensures that a given physical token has to be involved during the certificate verification, i.e. during the challenge-response protocol, and ensures that the attack described in Figure 1 cannot occur.

However, this solution relies on the assumption that the authenticated device is talking directly to the verifier. Man-in-the-middle attacks between virtual entities, as assumed in classical protocols, are usually prevented by encrypting communications based on the public key of the impersonated entity. In pervasive computing however, such attacks occur between physical entities. Applications that involve the delivery of goods or authenticating the location of a person or token are thus vulnerable to this class of attack, a scenario that is quite different from that of classical protocols. This type of attack thus happens to be a major threat in pervasive computing environments since even though the real artifact or token may be involved in the authentication process, its physical location is not assured at all.

Figure 2 shows that even with tamper-resistant modules, it is possible to implement proxies that forward challenges and responses between the token and the entity verifying its attributes.

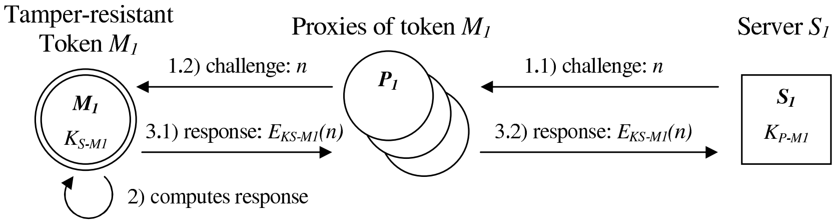

Fig. 2. Man-in-the-middle attack against a tamper-resistant token

Two solutions can be used to prevent proxy attacks and to offer strong authentication of artifacts: the artifact can be isolated when verifying that it knows a secret, or its vicinity can be determined with a distance-bounding protocol. The former approach is implemented to let automated teller machines (ATM) authenticate credit cards but it is not flexible enough to be used in most pervasive computing applications. The present work investigates how distance-bounding protocols can be integrated within simple interactions.

In many pervasive computing transactions, proximity and location verification are required during authentication of artifacts. For instance, a location stamping service can be necessary to prove the user or an artifact presence at a given place and time. If the location stamping service is implemented through a constrained channel, a peer-to-peer location sharing attack [2] can be mounted in order to obtain the stamp from anywhere: an accomplice of the attacker just needs to have access to the constrained channel (e.g. being in front of the location-stamp service) so that he can act as a proxy. Isolating the artifact is possible but may quite impede the user-friendliness or increase the cost of the solution. Distance-bounding protocols make it possible to prove that a given artifact is really present without resorting to its full isolation.

3.2 Distance-Bounding Protocols

A distance-bounding protocol associates a distance measurement and cryptographic properties in order to prove that a secret is known within a physical area. To ensure fast enough exchanges (i.e. a few nanoseconds), dedicated hardware interfaces and one-bit messages are required.

The artifact A wants to verify that the artifact B has some properties. B can show a certificate signed by a trusted certification authority (CA) which certifies that the owner of the public key K_{P_B} has some properties. The distance-bounding protocol is used to verify that the entity knowing the private key K_{S_B} is close enough and is not involved in a proxy attack.

The probability of successful attack can vary depending on the protocol used. In this paper, we will not use our initial proposal [2], but rather [1], which exhibits a lower probability of successful attack. This scheme, which was devised in the context of network authentication, can also be used for pervasive environments. One-bit challenges and one-bit responses being used, the probability of a successful attack during a single round is high: 1/2. Using N rounds ensures that the probability of successful attack is $(1/2)^N$. Table 1 show how artifact A can measure the distance to artifact B.

1)	**Initialization**				
1.1)	A			Choose a N bits random number R_A	
1.2)	B			Choose a N bits random number R_B	
2)	**Rounds**			(for $i \in \{1 \cdots N\}$)	
2.1)	A			Start measuring round trip time	
2.2)	A	$\Rightarrow$	B	$R_A[i]$	
2.3)	A	$\Leftarrow$	B	$R_B[i]$	
2.4)	A			Verify round trip time	
3)	**Verification**				
3.1)	A	$\leftarrow$	B	$SIGN_B(R_A, R_B)$	where $SIGN_X(Y) = E_{K-S_X}(H(Y))$

Table 1. Distance-bounding protocol between artifacts A and B

During rounds $1 \cdots N$, the dedicated interface ($\Rightarrow$) is used and the response time is measured in order to evaluate the distance between A and B. Dedicated hardware and one-bit exchanges allow a few nanoseconds round trip time. If B cannot provide R_B to another artifact, the protocol can be used to prove that B is present. This protocol proves to A that it is in contact with an entity knowing K_{S-B}. If this is combined with tamper-resistant modules to protect the private key storage and use, A can be sure that it is in front of B. Within this paper, this protocol is symbolized by $A \rightsquigarrow B$. Mutual distance-bounding protocol ($A \leftrightsquigarrow B$) means that both A and B have verified that they have just been in contact.

To summarize, Security in Pervasive Computing relies on $Cert$ificates, Tamper-Resistant Modules, and Distance-Bounding Protocols:

$$SPC = CERT + TRM + DBP$$

3.3 Implementation Constraints

Different parameters have to be taken into account when implementing distance-bounding protocols. First, such protocols rely on a large number of simple rounds at each of which the probability of a successful attack is quite high. Such a protocol is thus difficult to be rendered fault-tolerant and has to be implemented on a noiseless channel. Broadcast media such as radio are thus seemingly out of question for implementing these protocols.

In addition to these technical problems, radio does not offer a way to precisely select a (physical) artifact, which is essential for the user who wants to authenticate a precise device and not any device that can listen and answer to his requests. Contact based approaches are better in this respect, except when a user may want to access a device fixed to the ceiling. Alternately, a directive technique like the use of a laser may allow contacting wirelessly a precise artifact. It should be noted however that the laser performs two different tasks at the same time. It first selects a precise artifact and measures its physical distance, like the plugging in a contact based approach. It is then used to implement a distance-bounding protocol, therefore checking that the logical entity is effectively embodied in that artifact through the measurement of the time taken to transmit a secret. Performing these two tasks suggests a difficult technological integration.

Finally, mutual authentication is often expected and, in this case, the selection process has to be bi-directional. Again, defining a wireless approach based on lasers seems difficult in that case. The easiest solution is to use contact based approaches, be they electrical or optical.

In the following sections, we assume that each artifact offers a dedicated contact based interface for distance-bounding protocols (look at [2] for more details). Users that want to authenticate artifacts can touch them with a trusted artifact (PDA, electronic ring, etc).

4 Pervasive Security Mechanisms

As explained in Section 3, an electric contact is the easiest if not the only reasonable way to reach a strong authentication of pervasive computing artifacts.

The requirement for a contact has a major impact on the possible secure interactions that can be defined. However, it fits well with scenarios involving a direct contact between users and artifacts. This section presents a solution that combines the use of tamper-resistant tokens and distance-bounding protocols in order to secure intuitive interactions between human beings and artifacts.

4.1 Presence of User

Distance-bounding protocols can be used to verify that an artifact is physically present. When it is necessary to check if a human being is present during a transaction or to deliver a proof of location, it is in fact the presence of his token that is verified, the token being for instance an electronic ring [9] that can be worn by the user. Tokens may potentially carry numerous rights such as accessing office, house, car, communication, and entertainment services, and may be used to sign documents, for payment, or to delegate rights. However, even such tokens can be stolen: directly authenticating the user of a token is thus critical. PIN codes are generally used to unlock tokens such as SIM cards. However, in pervasive computing, it is not possible to rely on passwords for each interaction because it suppresses intuitiveness. Two mechanisms can be proposed to diminish the threats on the personal token: when the interactions are done online, it is possible to create a token revocation list; when the interactions are done offline it is necessary to lock the token periodically. Both rely on surrounding artifacts: the former needs a terminal to add the token to the revocation list; the latter requires a way to delegate the right of unlocking the token to another artifact. For instance, an e-ring could be unlocked by entering a PIN code on a cell-phone or by touching a given finger print reader.

In the following, we will assume that each user U carries a personal token T_U that identifies him and that is used to interact with surrounding artifacts. We propose to implement tokens as tamper-resistant electronic rings with dedicated distance-bounding interface. They can be used for protecting the private key K_{S_U} of their owner and to select other artifacts by touching them.

Each artifact A_i has its own private key $K_{S_{A_i}}$. It is possible to provide rights to an artifact by defining an authorization certificate. The features and the owner of an artifact may also be defined by attribute certificates.

4.2 Seamless Trust

Trust relationships are bi-directional and often asymmetric. When using an artifact, it is necessary to verify whether it can be trusted. And, when lending an artifact or providing some service, abusive usage must be prevented.

Pervasive computing requires the most transparent interaction semantics in order to remain intuitive: touching a device holds such a promise. For instance, a user may plug his ring into an ATM in order to be authenticated and retrieve money. If he wants to prevent proxy attacks, the setup will be slightly more complex so that, for instance, he gets warned with a blinking led on his ring that the device he believes to be an ATM is potentially a fake because it cannot associate a certificate issued by a bank with a distance-bounding protocol. In addition, interesting interactions often involve three or more artifacts. For instance, the user can have to connect two artifacts using his ring.

This paper focuses on scenarios in which there exists a form of *a priori* trust that can be based on a public key infrastructure or a web of trust. For instance, interactions with ATMs rely on the knowledge of the bank's public key; working with artifacts owned by another company requires the prior establishment of relationships.

The following subsections describe how the *drag-and-drop* metaphor can be recycled to implement several security mechanisms that often come up in pervasive scenarios: how to be sure that an artifact conforms to some specification? How to enable an artifact to perform some access on behalf of a user? How to provide some proof about an operation? Finally, how to be sure of the ownership of an artifact and how to transfer it?

4.3 Verifying Attributes of Artifacts

Data are often associated with a physical object, be it a comment associated to paintings in a museum or the expiring date of some food. Protecting those data against forgery requires certifying and linking them to the artifact thanks to a distance-bounding protocol. For instance, when someone buys an artifact, let's say a pervasive Swiss watch or a pervasive box of pills, it is necessary to verify that the manufacturer did actually certify this artifact. Mechanisms ensuring that artifacts (or at least the chips associated to these artifacts [7]) cannot be cloned and that they are physically present are required. Certification can be based on a PKI, a web of trust, or ad-hoc mechanisms.

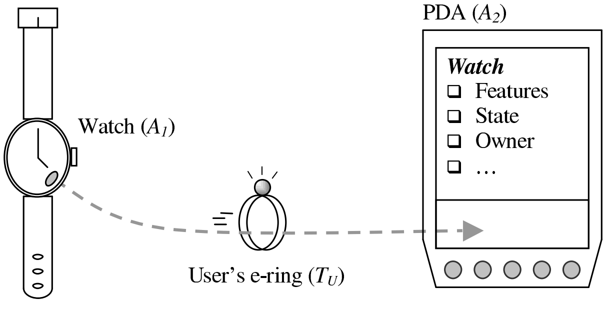

Fig. 3. Drag-and-drop to show attributes of an artifact

Requirements This approach relies on the following hypotheses:

– Tamper-resistant hardware are available

- Hardware interfaces dedicated to distance-bounding protocols are deployed.
- Everybody has a unique personal token (e-ring) that is used for authentication and interactions.

A tamper-resistant token takes care of its owner's private key. Each artifact has its own asymmetric key pair that allows their certification (features, rights, owner). Strong authentication of users and artifacts is ensured by asymmetric cryptography and a distance-bounding protocol. Figure 3 shows how a drag-and-drop mechanism can be used to intuitively display the characteristics of an artifact. This protocol ensures that the displayed data correspond to the artifact that has been touched (e.g. the watch). The user trusts his token to perform a distance-bounding protocol with the artifact. The artifact can be verified anonymously using the token, hence protecting the user privacy. Alternately, the artifact can require the user identity or an authorization to control access to its services or resources.

Protocol Description Table 2 describes how a drag-and-drop protocol can be used between two artifacts in order to verify attributes of the first one.

1)	**Drag**			
1.1)	T_U	$\rightarrow$	A_1	<Get description>
1.2)	T_U	$\leftarrow$	A_1	$CERT\text{-}A_1 = CERT_{CA}(K_{P_{A1}}, attributes)$
				where $CERT_X(Y) = Y, SIGN_X(Y))$
1.3)	T_U	$\rightsquigarrow$	A_1	Distance-bounding protocol
2)	**Drop**			(before timeout)
2.1)	T_U	$\rightarrow$	A_2	<Put description>:
				$CERT_{TU}(T_U \text{ touched } A_1, CERT\text{-}A_1)$
2.2)	A_2			Display description of A_1

Table 2. Basic drag-and-drop protocol between two artifacts

The user touches the two artifacts in succession with his token. The protocol is described in two parts: drag, which is the interaction between the token T_U and the artifact A_1, and drop, which is the interaction between the token and another artifact A_2. In 1.2, T_U receives a certificate describing A_1 signed by a Certification Authority (CA). In 1.3, T_U verifies that the touched artifact knows the private key $K_{S_{A1}}$ corresponding to the certified public key $K_{P_{A1}}$. As a result of the drag operation, the token has the proof that *it touched an artifact knowing* $K_{S_{A1}}$ and that *CA certifies that the entity knowing* $K_{S_{A1}}$ *has some attributes.* In 2.1, T_U drops the certificate of A_1 to A_2. It is certified by T_U.

On a side note, the distance-bounding protocol, for it involves nonces, provides a one-time proof that the token is in contact with the owner of the secret key each time it is performed. No additional nonce or timestamp is required in the derived protocol. In any case, a timer should be available in the token to cancel operations if the drag is not performed after some timeout.

This scheme can be extended. For instance, in 1.2, mutual authentication could be required in order to provide access control so that A_1 only delivers description to authorized tokens. In this case, there is an obvious tradeoff between access control and privacy, that is, can anonymous drag be used or not.

4.4 Pervasive Access Control

In pervasive computing environments, the initialization of services has to be as transparent as possible. Figure 4 shows how an MP3 player should be connected to a headset: the user drags the song from one artifact to the other and dynamically creates a new relationship. In this context, secure pairing of artifacts can be achieved thanks to the drag-and-drop metaphor.

However, access control can be required and it can be necessary to know who is performing the drag-and-drop in order to verify his rights and possibly to charge him. Artifacts receive rights, for instance, an MP3 player can be allowed to pay up to a given amount when downloading a song. Bob can authorize Alice to drive his car or enter his house by providing a credential to her token.

Figure 4 shows how artifacts can be dynamically associated by a user. It is necessary to take into account the rights and ownership of the user and of the involved artifacts to define the rights of the artifact, how long they will remain in effect, etc.

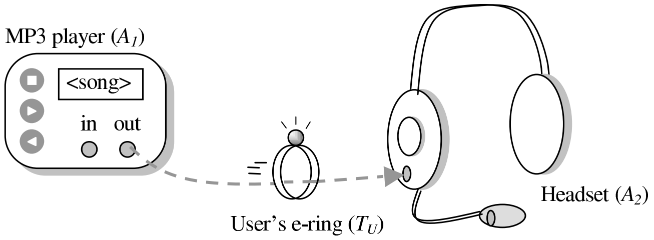

Fig. 4. Drag-and-drop to initiate a new relationship

The MP3 player and the headset know that a given user touched them one after another. They can then use a wireless medium to establish a connection and enable the headset to play some music according to the rights of this user.

Requirements It is necessary to use a distance-bounding protocol to ensure that only the touched artifact (i.e. the headset) will receive the access to the service delivered (i.e. music delivered by the MP3 player). When the interaction leads to more strategic access rights (i.e. confidential data or physical access), it becomes mandatory to verify who receives the authorizations. The protocol is similar to Section 4.3. However, after the drop operation, A_2 is able to contact A_1 through a wireless medium such as Bluetooth in order to establish the service.

Protocol Description Table 3 is an extended version of the basic drag-and-drop protocol defined in Table 2. It ensures mutual authentication and mutual proof of contact.

1)	**Drag**	
1.1)	$T_U \rightarrow A_1$	<Get description>
1.2)	$T_U \leftarrow A_1$	<Require mutual authentication>
1.3)	$T_U \rightarrow A_1$	$CERT\text{-}T_U = CERT_{CA_1}(K_{P_U}, attributes)$
1.4)	$T_U \leftarrow A_1$	$CERT\text{-}A_1 = CERT_{CA_2}(K_{P_{A1}}, attributes)$
1.5)	$T_U \leftrightsquigarrow A_1$	Mutual distance-bounding protocol
2)	**Drop**	(before timeout)
2.1)	$T_U \rightarrow A_2$	<Put description> $CERT_{TU}(T_U$ touched $A_1, CERT\text{-}A_1)$
2.2)	$T_U \leftarrow A_2$	<Require mutual authentication>
2.3)	$T_U \rightarrow A_2$	$CERT\text{-}T_U$
2.4)	$T_U \leftarrow A_2$	$CERT\text{-}A_2 = CERT_{CA_3}(K_{P_{A2}}, attributes)$
2.5)	$T_U \leftrightsquigarrow A_2$	Mutual distance-bounding protocol
3)	**Service**	
3.1)	$A_2 \rightarrow A_1$	<Negociate service>

Table 3. Drag-and-drop protocol to start a service between two artifacts

After step 1, T_U knows which device has been touched by user U and artifact A_1 is waiting for a service call. After step 2, artifact A_2 knows that user U touched successively (i.e. within a given timeout) artifacts A_1 and A_2. In 2.1, A_2 received the attributes of A_1, including its network address. During step 3, A_2 gets in touch with A_1 to negotiate the service. Both have been touched by the same user and thus can use his rights or charge him when delivering the service.

4.5 User Centered Interactions

Previous sections focused on artifacts, but many pervasive scenarios rather emphasize user centered interactions, and in particular, multi-user interactions. Part of the trust and rights associated to a device relates in fact with its owner, and it is necessary to bridge the gap between the human being and his device. Because physical ubiquity is impossible, this task should also benefit from distance-bounding protocols as discussed in this section.

Non-repudiation Non-repudiation of interaction aims at proving that an interaction occurred between users and/or artifacts. Distance-bounding protocols are not sufficient to ensure non-repudiation. Indeed, after a distance-bounding protocol $A \rightsquigarrow B$, the artifact A verified that B is within some distance. However this information must be associated with a certain context: providing a non-repudiation service to a pervasive interaction may not only require certifying the identity of the involved parties, but also the time at which it occurred and the history of previous interactions, for instance. Different types of non-repudiation (e.g. order of interaction, initiator, target) can be integrated within the drag-and-drop protocol.

If two artifacts have to sign a proof of contact after performing a mutual distance-bounding protocol, it is necessary to ensure a fair exchange. Depending on the context, this issue can be solved directly by the tamper-resistant module within the involved artifacts or with a trusted third party.

Ownership Management A straightforward way to define trust levels is to base them on ownership. In that case, the level of trust depends on the type of an artifact and on its owner. Security policies can define whether a device owned by a friend or a partner company can be trusted enough to be involved in specific transactions. For instance, a corporate security policy could define that confidential data can be displayed on partners' displays but only the company's devices can be used to access secret data.

The management of ownership is also required to support the trading of artifacts. Ownership should be based on certificates too, the initial certificate being signed by the first owner, which could be the manufacturer. Such an imprinting could be similar to the approach proposed in [13]. When a user wants to trade an artifact, he has to use a distance-bounding protocol in order to verify that he becomes the owner of the device he is holding. It is possible to define intuitive interactions based on the drag-and-drop metaphor: for instance, the seller and the buyer could each touch the device in sequence and within a given timeout in order to change its ownership. When more complex exchanges are required (e.g. involving money), the semantics of the drag-and-drop is too poor. In this case, it may be necessary to extend these semantics with other artifacts able to display trading information (e.g. a PDA).

Proof of Context Accurate location and proximity information is very important to secure pervasive computing environments. For instance, an employee may obtain access rights whenever he enters the building of his corporation.

Proxy attacks should especially be expected in short range or personal area networks (e.g. W-LAN) which are notoriously insecure with respect to location. Protocols based on distance-bounding would be more than useful for instance for employees in a meeting room. Each of them could touch the meeting table with their e-ring to become part of a meeting group. Physically authenticated members would then receive a group key allowing them to share some data, for instance as long as they are together. Without the use of a distance-bounding protocol, one member of the group could act as a proxy and let someone, who is not in the meeting room, become a legitimate member of the group and have access to group identity based functions.

Users can also use proof of location to subsequently prove that they went somewhere or meet someone. The history of their actions may then serve to elaborate a key infrastructure or to deduct a trust level for their devices.

5 Conclusion and Future Work

Protection against impersonation attacks requires the association of asymmetric cryptography, tamper-resistant modules, and distance-bounding protocols. This combination makes it possible to verify whether one is really interacting with an identified appliance. The implementation of a distance-bounding protocol also implies the use of a small personal token able to perform cryptographic operations yet portable. However, authentication alone is not enough, especially because users manipulate appliances, not virtual services or cryptography.

The proposal made in this paper is to integrate the authentication phase performed with the personal token together with an intuitive model of interaction with appliances. This paper illustrates the use of the simple and familiar drag-and-drop paradigm to introduce security mechanisms specific to pervasive applications like managing the ownership of a device or manually performing access control between two artifacts.

The scenarios depicted in this paper essentially assume that every artifact has a network connection that can be used to connect to a trust infrastructure. Yet, the drag-and-drop approach developed can be useful to alleviate this limitation. For instance, personal token may be exploited as a repository for certificates: users would thus bring a small part of the public key infrastructure with them. It should be noted that in general, trust remains difficult to achieve in a decentralized situation where replay attacks on the services offered by an appliance are likely to happen because of the distributed nature of the system. We are currently developing one-time certificates to address this issue.

The *a priori* trust assumption that is made in the scenarios presented in this paper is correct in many business situations. However, open pervasive environments can be envisioned where this assumption does not hold because there is absolutely no prior relationship between users or appliances. Signing a transaction may as well be devoid of any legal liability in scenarios where the user travels to a country with a different legislation. We are investigating new mechanisms to deal with such open trust scenarios.

Finally, the protection of the user privacy [11] is a rather critical issue in pervasive computing [8] that may seem in contradiction with authentication. The one-time certificate mechanism mentioned above might just help dealing with an appliance without revealing one's identity.

References

1. Brands S. and Chaum D.: Distance-bounding protocols (extended abstract). EUROCRYPT 93, volume 765 of Lecture Notes in Computer Science, pages 344-359. Springer-Verlag, 1994, 23-27 (May 1993).
2. Bussard L., Roudier Y.: Authentication in Ubiquitous Computing, Ubicomp 2002, Workshop on Security in Ubiquitous Computing, (Sept 2002).
3. Covington, M.J.,. Moyer, M.J., and Ahamad, M.: Generalized Role-Based Access Control for Securing Future Applications. In 23rd National Information Systems Security Conference (2000).
4. Holmquist, L.E., Mattern F., Schiele B., Alahuhta P., Beigl M., and Gellersen H.W.: Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts, In Proc. of UBICOMP 2001, Atlanta, GA, USA (Sept. 2001).
5. Kagal L., Finin T., and Joshi A.: Trust-Based Security in Pervasive Computing Environments. In IEEE Computer Volume 24, Number 12, pages 154-157. (December 2001).
6. Kahn J.M., Katz R.H., and Pister K.S.J.: Next Century Challenges: Mobile Networking for 'Smart Dust'. In MOBICOM, pages 271-278, (1999).
7. Kindberg T., Zhang K., and Shankar N.: Context authentication using constrained channels. In Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), pages 14–21, (June 2002).
8. Langheinrich M.: Privacy by design- principles of privacy-aware ubiquitous systems. In ubicomp2001, volume 2201 of LNCS, pages 273-291, (2001).
9. Meloan S.: Inside the Java Ring event. http://java.sun.com/features/1998/07/ring-project.html
10. Network Working Group: Request for Comments 2693: SPKI Certificate Theory, (September 1999).
11. Pfitzmann A., Köhntopp M.: Anonymity, Unobservability, and Pseudonymity - A Proposal for Terminology. Workshop on Design Issues in Anonymity and Unobservability (2000).
12. Richard G.G.,: Service Advertisement and Discovery: Enabling Universal Device Cooperation, IEEE Internet Computing, vol. 4, no. 5, (September/October 2000).
13. Stajano F. and Anderson R.: The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks, 7th International Workshop on Security Protocols Proceedings, (1999).