Institut EURECOM
Corporate Communications
2229, route des Crêtes BP 193
06904 Sophia Antipolis
(France)

Research Report[a] N^o 100 — RR-04-100

Combining History-based Trust Establishment with Distance-bounding Protocols

Laurent Bussard, Refik Molva, and Yves Roudier

April 5, 2004

[a] Institut Eurécom's research is partially supported by its members: Bouygues Télécom, Cegetel, France Télécom, Hasler Foundation, Hitachi, STMicroelectronics, Swisscom, Texas Instruments, and Thales.

Combining History-based Trust Establishment with Distance-bounding Protocols

Laurent Bussard, Refik Molva, and Yves Roudier

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

Abstract. History-based trust establishment aims at using unlinkable recommendations and proofs of context in order to build trust based on human notions of trust: when two persons meet for the first time, they exchange information on their acquaintances, interest and context in order to find some shared experience on which to build trust. This paper focuses on two central aspects of trust establishment: the history-based trust establishment mechanism and the proof of locality viewed as an important contextual attribute. We suggest a mechanism for proof of locality based on a distance bounding protocol. The resulting proof of locality as well as other attributes can then anonymously be combined with our history-based scheme in order for each party to prove his attributes without revealing his identity. The trust establishment scheme relies on an extension of group signatures and thus inherits various privacy attributes thereof.

Introduction

With the advent of self-organizing systems such as ad hoc networks or ubiquitous computing, security protocols have to meet new challenges for establishing trust between communicating parties. First, there are numerous cases where no trust information about other parties is available at all: for instance, a public key infrastructure [20], or a web of trust [21], or any other identity-based trust infrastructure is not available in many cases [32]. Trust establishment in this context calls for a brand new paradigm with respect to classical scenarios whereby entities build trust based on some existing security association.

This paper introduces a protocol through which an entity can give cryptographic evidence of the history of its past interactions with other parties. This history makes it possible to evaluate the past behavior of an entity, and accordingly consider it more or less trustworthy when performing other interactions. Privacy is an essential requirement for such a protocol that builds up trust at the expense of exposing intimate behavior of a user: for instance, an electronic discount coupon valid for various locations other than the shopping mall where the

coupon was generated should not enable the tracing of clients. The anonymity of the history proving entity and the unlinkability of its interactions was thus a design objective for the history proving protocol detailed in this paper, which extends a scheme proposed in [6]. Such privacy requirements are met by the group membership notion in group signature schemes [16] that proves the existence of a relationship with other members of the group while ensuring the anonymity of the signer. This paper extends the group membership concept and associates embedded attributes within a signature in order to prove that attribute without revealing the identity of the signer.

History can be defined as a set of items such as the context of an action [33, 27] (i.e. its time and location), the membership of a group (reporter, program committee member), or a recommendation (defined by Bob as a trusted party) [18]. We suggest in this paper a privacy preserving mechanism through which the user can get the proof of a context in terms of time and location. Based on this scheme, the user can choose the degree of accuracy at which he wants to reveal his contextual attributes, e.g. someone that can prove that he was in Paris on the 15^{th} of January might only want to reveal that he was in France in January. This paper describes how a proof of context may be elaborated based on an extension of group signatures as hinted above, and revealed with a selected level of granularity using a challenge-response protocol, making it possible for the history proving entity to disclose only part of its history attributes.

Section 1 details the security requirements that must be addressed to enable contextual history credentials. Section 2 introduces the concept of a proof of knowledge used in our protocol. Section 3 shows how it is possible to anonymously assert the proximity of some entity using a distance-bounding protocol. Section 4 describes the protocol that makes it possible, based on this assertion, to prove the contextual history of the entity. Section 5 explains how an entity's contextual history, that is the time and location of its past interactions, can be disclosed with a variable granularity to enhance its privacy. Section 6 discusses the security of the scheme proposed for establishing trust based on contextual evidence. Finally, Section 7 presents some related works.

1 Problem Statement

This section gives an overview of the interactions necessary to build a provable history and to use this for history-based trust establishment.

1.1 Principle

Users anonymously collect evidences of their activity and store it as a provable history. In Figure 1, a user gets a proof that he has been at a location. To ensure non-transferability of evidences, they are implemented as credentials attached to a valuable secret. Those credentials are stored on the holder's personal device (e.g. cell-phone) and the secret is protected by a tamper-resistant module (e.g. smart card). This paper focuses on location-and-time stamps but different types

of credentials can be implemented similarly: group membership, recommendations, etc.

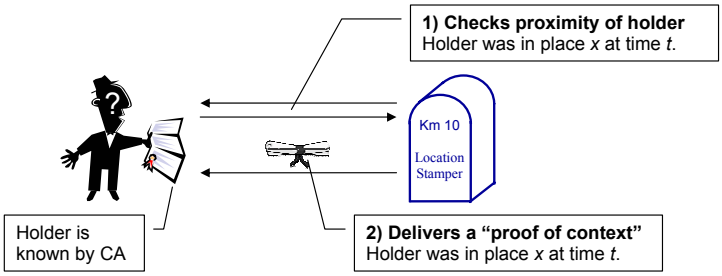

Fig. 1. Getting history items

During authentication, the author chooses some credentials in his history, modifies them, and proves the knowledge of those credentials. In Figure 2, a user is able to prove that he was at a location x at time t, that he is said reliable by some entity Z, that he is a member of group G, and that he has a given name and address (electronic id card). He chooses to be authenticated as *someone that was at location x at time t*. The authentication does not reveal more information on the user and it is even not possible to link two interactions of the same user. To ensure untraceability, it is necessary to avoid being too precise: it is indeed easier to identify a person that signed as having been in a given room at a precise time than to recognize this person based on the knowledge that he was in the building at some time [11].

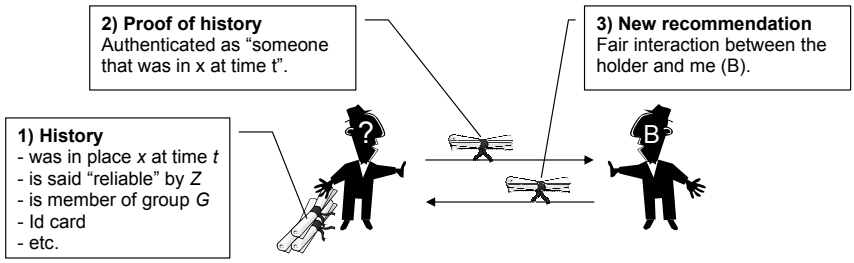

Fig. 2. History-based trust establishment

Credentials have to fulfill the following requirements to build a provable yet anonymous history:

- *Non-transferability*: credentials can only be used by the owner of some valuable secret (equivalent to the private key in public key infrastructures). This secret is critical and thus will not be transferred to another entity. As a result, credentials cannot be transferred.
- *Anonymity*: use of history-based credentials should not reveal the identity of the author.
- *Untraceability*: it is not possible to link different proofs of a same person even when the same credential is used.

Moreover, when the credential is a proof of context, the server has to verify that the requester is really in front of the terminal. Indeed, it should not be possible for some remote user to pretend being present, even when some partner forwarding challenges and responses is indeed in front of the terminal.

2 Basic Mechanisms

This section presents the first group signature of [16] that will be modified in the sequel of this paper in order to define a history-based trust establishment scheme.

We define the following elements: $n = pq$ where p and q are two large primes; $\mathcal{Z}_n = \{0, 1, 2, \ldots, n-1\}$ is a ring of integers modulo n; $\mathcal{Z}_n^* = \{i \in \mathcal{Z}_n \mid \gcd(i, n) = 1\}$ is a multiplicative group; $G = \{1, g, g^2, \ldots, g^{n-1}\}$ is a cyclic group of order n; g is a generator of this group G; $a \in \mathcal{Z}_n^*$ is an element of the multiplicative group; and λ is a security parameter (see [16] for more details).

2.1 Interactive Proof of Knowledge

A *proof of knowledge* (PK) allows an entity to prove the knowledge of some secret without revealing this secret. For instance, the prover P claims to know the double discrete logarithm of z to the bases g and a. The verifier V tests if P indeed knows x. This is denoted PK$[\alpha \mid z = g^{(a^\alpha)}]$ or PKLOGLOG.

P sends a witness to V: $w = g^{(a^r)}$ where r is a random value and V returns a random challenge bit $c \in_R \{0, 1\}$. Finally P sends a response $s = r$ (if $c = 0$) or $s = r - x$ (if $c = 1$). The verifier checks that

$$\begin{aligned} c = 0 \ : \ w \stackrel{?}{=} g^{(a^s)} \quad &= g^{(a^r)} \\ c = 1 \ : \ w \stackrel{?}{=} z^{(a^s)} \quad &= \left(g^{(a^x)}\right)^{(a^s)} = g^{(a^{x+s})} = g^{(a^r)} \end{aligned}$$

This protocol has to be run l times where l is a security parameter.

2.2 Signature based on a Proof of Knowledge

A *signature based on a proof of knowledge* (or signature of knowledge) of the double discrete logarithm of z to the bases g and a, on message m, with security

parameter l is denoted $\mathrm{SPK}_l[\alpha \mid z = g^{(a^\alpha)}](m)$ or SPKLOGLOG. It is a non-interactive version of the protocol depicted in Section 2.1. The signature is an $l+1$ tuple $(c, s_1, \ldots, s_l)$ satisfying the equation:

$$c = \mathcal{H}_l(m \parallel z \parallel g \parallel a \parallel P_1 \parallel \ \ldots \ \parallel P_l) \quad \text{where } P_i = \begin{cases} g^{(a^{s_i})} & \text{if } c[i] = 0 \\ z^{(a^{s_i})} & \text{otherwise} \end{cases}$$

It is computed as following:

1. For $1 \leq i \leq l$, generate random r_i.
2. Set $P_i = g^{(a^{r_i})}$ and compute $c = \mathcal{H}_l(m \parallel z \parallel g \parallel a \parallel P_1 \parallel \ \ldots \ \parallel P_l)$.
3. Set $s_i = \begin{cases} r_i & \text{if } c[i] = 0 \\ r_i - x & \text{otherwise} \end{cases}$

2.3 Group Signature

The group signature scheme in [16] is based on two signatures of knowledge: one that proves the signer knows some secret and another one that proves this secret is certified by the group manager. The scheme relies on the hardness of computing discrete logarithm, double discrete logarithm and e^{th} root of the discrete logarithm.

The public key of a group is (n, e, G, g, a, λ) where e is chosen so that $\gcd(e, \phi(n)) = 1$ where $n = pq$. The private key of the manager is (p, q, d) where $de = 1 \bmod \phi(n)$. When Alice *joins* the group, i.e. becomes a member, she uses her secret x to compute a membership key (y, z) where $y = a^x \bmod n$ and $z = g^y$. A sends (y, z) to the group manager, proves that she knows x and receives a group certificate $(y+1)^d \bmod n$ corresponding to her secret x. In order to sign a message m, A chooses $r \in_R \mathcal{Z}_n$ and computes $\tilde{g} = g^r$, $\tilde{z} = \tilde{g}^y$ $(= z^r)$, and two signatures:

$$V_1 = \mathrm{SPK}[\alpha \mid \tilde{z} = \tilde{g}^{(a^\alpha)}](m)$$
$$V_2 = \mathrm{SPK}[\beta \mid \tilde{z}\tilde{g} = \tilde{g}^{(\beta^e)}](m)$$

V_1 is a signature of knowledge of a double discrete logarithm that can be computed when knowing some secret x. Similarly, V_2 is a signature of knowledge of an e^{th} root of the discrete logarithm that can be computed using the certificate $(y+1)^d \bmod n$. The group signature of message m is $(\tilde{g}, \tilde{z}, V_1, V_2)$.

The verifier checks that V_1 and V_2 are valid signatures of m. Both signatures together mean that $\tilde{g}^{(\beta^e)} = \tilde{z}\tilde{g} = \tilde{g}^{(a^\alpha+1)}$ and thus $\beta = (a^\alpha + 1)^d \bmod n$. The verifier knows that the signer holds a certified secret x. However, the verifier cannot get any information on x. In other words, the identity of the signer is preserved: this is a group signature.

3 Proof of Context

Proving the proximity of two entities is very difficult (related approaches are described in Section 7.2). It is indeed easy for an attacker to mount a mafia-fraud attack [4] against any authentication protocol that do not take that kind

of threat into account. The principle of such an attack is very simple: Alice stores her private key in a tamper-resistant device (e.g. her cell-phone) and uses a challenge-response protocol to open the front door of her house. Even when the authentication protocol is proven secure and the devices are tamper-resistant, it is easy for Eve to open the door. She will stay in front of the door, forward the challenge to her accomplice that follows Alice and get the correct response back. Two approaches can forbid this attack:

- Isolation [3] where some physical token is challenged in a Faraday cage so that any communication with a remote authorized entity is forbidden.
- Distance-Bounding Protocols [4] where the round trip time of challenge-reponse is evaluated to assure some proximity.

The former is used in ATMs but is not very practical for daily interactions because it imposes some Faraday cage to isolate one device. Here we study how distance-bounding protocols can be used in the context of history-based trust establishment. The main characteristic of a distance-bounding protocol is that it has to be very fast in order to provide a precise enough distance measurement. To achieve this goal, the following constraints are assumed during the rapid bit exchange: the challenge and the response can only be one bit messages; only Boolean operation can be done. For achieving required security properties, it is however necessary to have more complex interactions before and after a set of rapid bit exchange.

3.1 Distance-Bounding Proof of Knowledge

We define Distance-Bounding Proofs of Knowledge (DBPK) as interactive protocol that let some prover P prove to a verifier V that it is close by and that it knows a secret. Table 1 presents an example of DBPK of a double discrete logarithm $\mathrm{DBPK}_k[\alpha \mid z = g^{(a^\alpha)}]$ or DBPKLOGLOG where k is a security parameter. This is a modification of the scheme proposed by Chaum in [4].

The rapid bit exchange is done through a dedicated communication channel and only involves few logical gates in order to avoid delays. Like this it is the verifier measures a few nanoseconds round trip time between sending bit u_i and receiving bit v_i. A prototype, using fast logical gates has shown that two nanoseconds RTT is possible and thus, due to the speed of light, the accuracy is thirteen centimeters. This scheme is combined with a proof of knowledge. For instance, Table 1 defines a distance-bounding proof of knowledge of a double discrete logarithm. In other words, assuming that the signature and the generation of $\{\ldots, v_i, \ldots\}$ is done by a certified security module, the verifier knows that the double discrete logarithm of z is known by an entity that is physically very close.

This scheme as well as the original scheme [4] can be attacked by a remote prover colluding with some local proxy by providing each m_i. To avoid this, it is assumed that the distance bounding protocol is done by a certified tamper-resistant module, e.g. the prover's secret is protected by a smart card.

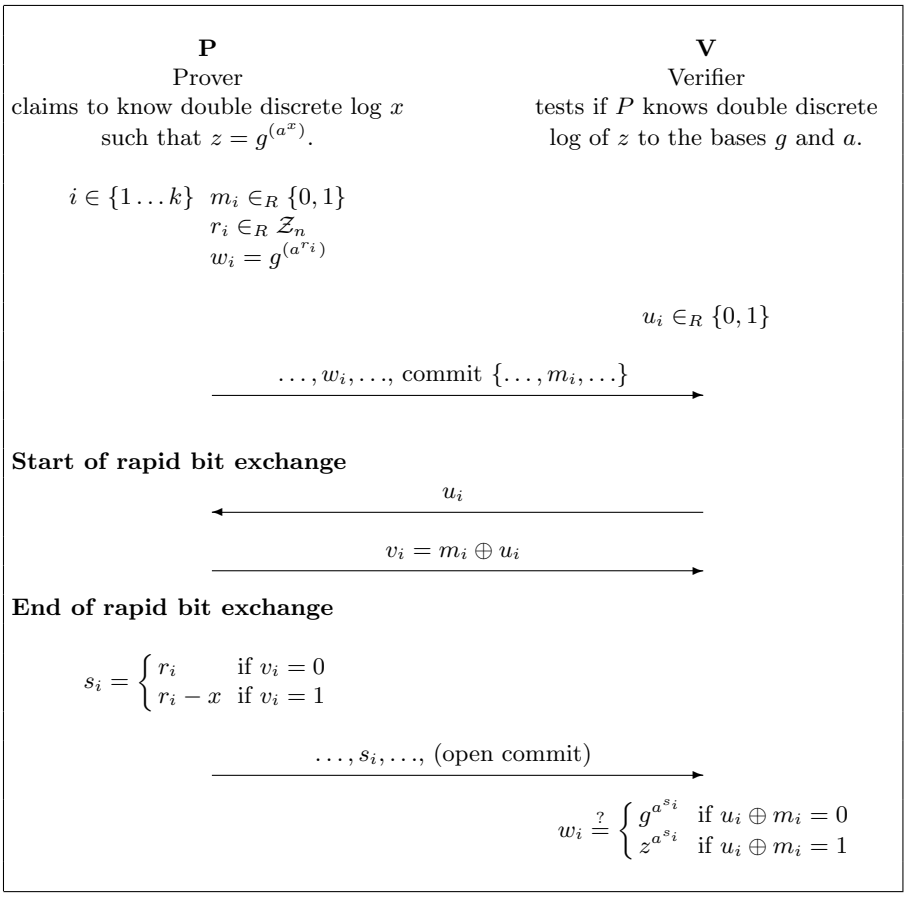

Table 1. distance bounding proof of knowledge of a double discrete logarithm

4 History-Based Trust Establishment

History-based trust establishment is an extension of the group signature scheme described in Section 2. Alice (A) is the signer. She collects some credentials to subsequently prove some history. For instance, A holds credentials to prove that she has been in some place. When A is traveling or visiting partners, she collects location stamps. A has credentials to prove some membership, e.g. employee of a company, member of ieee computer society, partner of some project, member of a golf club, citizen of some state, client of some bank, customer of some airline. A can show some recommendations: when she collaborates with other entities, she receives credentials. All those credentials define her provable history. Each credential can be used as a proof during a challenge-response protocol or as an attribute of a signature.

4.1 Certification

To initiate the system, each entity has to get some certificate proving that he/she has a valid secret, i.e. a secret linked to his/her identity. This part is similar to the join protocol of the Camenisch's scheme. But it is necessary to use a modified version because a coalition attack exists against the initial scheme [2, 29].

The secret x of A is generated by a tamper-resistant module (e.g. smart card) with a dedicated distance interface for distance-bounding protocols. This module can be certified by the manufacturer. Moreover, A receives from CA B a certificate on this secret x: $\text{cert}_{1b} = (a_b^x + 1)^{d_b} \bmod n_b$. Now, A is certified and can act anonymously as an entity certified by a given CA in order to get credentials and build a provable history.

4.2 Obtaining Proofs of Context or Recommendations

Once certified, A can visit different entities that will provide proofs of location, proofs of interaction, recommendations, etc. A provable history is a set of such proofs. Table 2 shows how A can get a credential from C. The identity of A is not known but C verifies that this entity is certified by some known CA or Group manager. It is always necessary to have some trust relationship with previous signers when providing credentials or when verifying history. In this example, C has to trust B otherwise the previous protocol has to be done once more. However, when an entity D needs to verify the signature of A on some document, D only has to know C.

Two proofs of knowledge are done in step 2.3). The first one is a distance bounding PK that proves that y_2 is based on some secret. Look at [14] for more details on how it is possible to prove relations among discrete logs from different groups. The second proof of knowledge shows that this secret has been certified by B. Indeed, $\tilde{z}\tilde{g_b} = \tilde{g_b}^{(\beta^{e_b})} = \tilde{g_b}^{(a_b^\alpha)}\tilde{g_b} = \tilde{g_b}^{(1+a_b^\alpha)}$ and thus $1 + a_b^\alpha = \beta^{e_b}$. It means that A knows $\beta = (1 + a_b^\alpha)^{d_b}$ that is a certification of α, which is also the discrete logarithm of y_2 to the base a_c. In other words, y_2 has been computed from the same secret x.

In step 2.4) A receives a new credential $\text{cert}_{2c} = (a_c^x + b_c^t)^{d_h} \bmod n_c$ from C that will be used to prove some history. b_c as well as a_c are elements of $\mathcal{Z}_{n_c}^*$, x prevents the transferability of credentials, and t is different for each credential to forbid a user from combining multiple credentials (see Section 6). The attribute value, be it a location or a recommendation, is defined using a technique that comes from electronic cash: $d_h = \prod_{i \in S} d_{c_i}$ where S is a set that defines the amount or any attribute. Construction of d_h is given in Section 5. Two other credentials can be provided: $\text{cert}_{1c} = (a_c^x + 1)^{d_c} \bmod n_c$ is a certification of the secret that can replace cert_{1b}. To avoid a potential attack (see Section 6), we add $\text{cert}_{3c} = (b_c^t + 1)^{d_c} \bmod n_c$.

4.3 Using History for Establising Trust

This section shows how Alice can prove that she holds of a set of credentials. A knows a secret x, the certification of this secret (cert_{1c}), and some credential

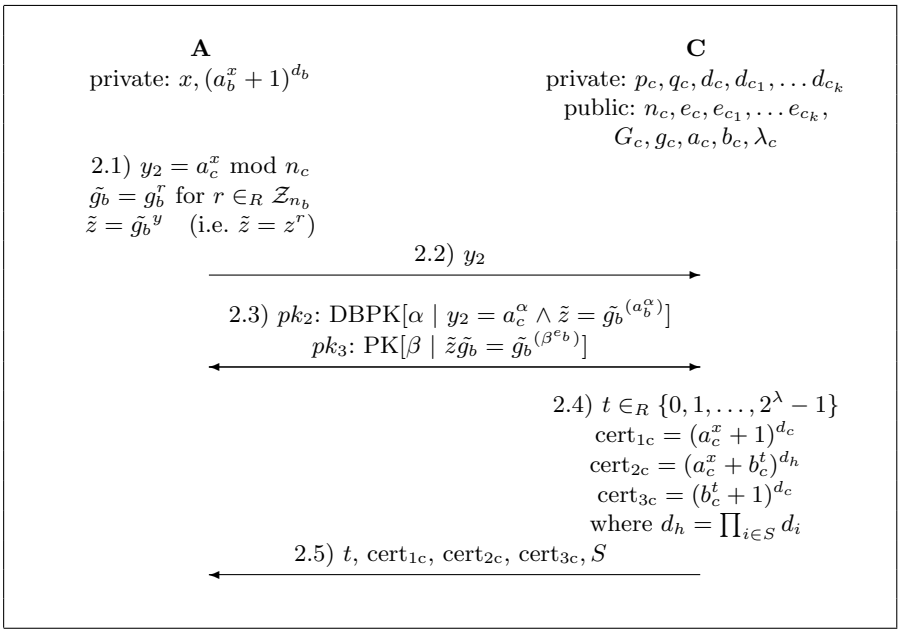

Table 2. Obtaining some credential to build history

that is part of her history (cert_{2c}). Using these credentials, she can compute a signature on some message m that is, for instance, a random challenge sent by some entity during a challenge-response protocol. A generates a random number $r_1 \in_R \mathcal{Z}_{n_c}$ and computes:

$$\hat{g_c} = g_c^{r_1},\ \hat{z_2} = \hat{g_c}^{y_2}, \text{ and } \hat{z_3} = \hat{g_c}^{(b_c^t)}$$
$$spk_1 = \text{SPK}[\alpha \mid \hat{z_2} = \hat{g_c}^{(a_c^\alpha)}](m)$$
$$spk_2 = \text{SPK}[\beta \mid \hat{z_2}\hat{g_c} = \hat{g_c}^{(\beta^{e_c})}](m)$$
$$spk_3 = \text{SPK}[\delta \mid \hat{z_3} = \hat{g_c}^{(b_c^\delta)}](m)$$
$$spk_4 = \text{SPK}[\gamma \mid \hat{z_2}\hat{z_3} = \hat{g_c}^{(\gamma^{e_{h'}})}](m) \quad \text{where} \quad e_{h'} = \prod_{i \in S'} e_i \text{ and } S' \subseteq S$$
$$spk_5 = \text{SPK}[\epsilon \mid \hat{z_3}\hat{g_c} = \hat{g_c}^{(\epsilon^{e_c})}](m)$$

The signature of message m is $\{spk_1, spk_2, spk_3, spk_4, spk_5, \hat{g_c}, \hat{z_2}, \hat{z_3}, S'\}$. The signatures of knowledge spk_1 and spk_2 prove that the signer knows cert_{1c}: $\beta = (1 + a_c^\alpha)^{d_c} \bmod n_c$. The signatures of knowledge spk_1, spk_3 and spk_4 prove that the signer knows cert'_{2c}: $\gamma = (a_c^\alpha + b_c^\delta)^{d_{h'}} \bmod n_c$. To avoid some potential attack (see Section 6), we added spk_5 to prove the knowledge of cert_{3c}. spk_3 and spk_5 prove that t was generated by C: $\epsilon = (1 + b_c^\delta)^{d_c} \bmod n_c$.

More subtle schemes can be defined to combine different credentials from multiple trusted third parties.

5 Encoding Attribute Values

In Section 4, the user receives cert_{2c} and signs with cert'_{2c} to hide part of the attributes when signing. This section presents a flexible mechanism for attribute encoding that allows the user to choose the granularity of attributes.

A straightforward solution to define attributes with various levels of granularity would be based on multiple credentials. For instance, a location-stamper would provide credentials defining room, building, quarter, town, state, etc. The holder would thus be able to choose the granularity of the proof of location. Unfortunately, this requires too much credentials when transversal attributes have different granularities (longitude, latitude, time, etc.).

5.1 Principle

Each authority that delivers certificates (time-stamper, location-stamper, group manager, etc.) has a public key: a RSA modulo (n), and a set of small primes $e_1, \ldots, e_m$ where $\forall i \in \{1, \ldots, m\} \mid \gcd(e_i, \phi(n)) = 1$. The meaning of each e_i is public as well. Each authority also has a private key: p, q, and $\{d_1, \ldots, d_m\}$ where $pq = n$ and $\forall i \in \{1, \ldots, m\} \mid e_i \cdot d_i = 1 \bmod \phi(n)$.

A signature $SIGN_{(S,n)}(m) = m^{d_h} \bmod n$, where S is a set and $d_h = \prod_{i \in S} d_i$, can then be transformed into a signature $SIGN_{(S',n)}(m) = m^{d_{h'}} \bmod n$, where S' is a subset of S and $d_{h'} = \prod_{i \in S'} d_i$. The the attribute value is coded as a set S corresponding to its bits equal to one. This signature based on set S can be reduced to any subset $S' \subseteq S$:

$$SIGN_{(S',n)}(m) = \big(SIGN_{(S,n)}(m)\big)^{\left(\prod_{i \in \{S \setminus S'\}} e_i\right)} = m^{\left(\prod_{i \in S'} d_i \bmod \phi(n)\right)} \bmod n$$

Thus, an entity that received some credential cert_{2c} is able to compute cert'_{2c} and to sign a document with this new credential.

$$\text{cert}'_{2c} = (\text{cert}_{2c})^{\prod_{j \in \{S \setminus S'\}} e_j} = \left(\left(a_c^x + b_c^t\right)^{\prod_{i \in S} d_i}\right)^{\prod_{j \in \{S \setminus S'\}} e_j} = \left(a_c^x + b_c^t\right)^{\prod_{i \in S'} d_i}$$

This technique ensures that part of the signed attributes can be modified. For instance, the attribute value $v = 13_d$ is equivalent to the binary string 01101_b and can be encoded as $S = \{4, 3, 1\}$, i.e. 4^{th}, 3^{rd}, and 1^{st} bits set to one. $d_h = d_4 \cdot d_3 \cdot d_1 \bmod \phi(n)$. Knowing $\{e_i \mid i \in S\}$, the following transformations are possible: $S' \in \{\{4, 3, 1\}; \{3, 1\}; \{4, 3\}; \{4, 1\}; \{4\}; \{3\}; \{1\}\}$ and thus $v' \in \{13, 5, 12, 9, 8, 4, 1\}$. Any bit i equal to one can be replaced by a zero (by using e_i) but any bit j equal to zero cannot be replaced by a one (because d_j is private).

5.2 Possible Codes

Choosing different ways to encode data enables to define which transformations of the attribute values are authorized:

- *greater-or-equal*: values are encoded so that they can only be reduced. For instance, $v = 13_d \rightarrow 01101_b \rightarrow S = \{1, 3, 4\}$. Because bits equal to one can be replaced by zeros, it can be transformed into $v' \in \{13, 12, 9, 8, 5, 4, 1\}$.
- *less-or-equal*: values are encoded so that they can only be increased. For instance, $v = 13_d \rightarrow 10010_b \rightarrow S = \{2, 5\}$. It can be transformed into $v' \in \{13, 15, 29, 31\}$.
- *unary greater-or-equal*: the problem with binary encoding is that they cannot be reduced to any value. For instance, $7_d = 111_b$ can be shown as 7, 6, 5, 4, 3, 2, 1, or 0 but $6_d = 110_b$ can only be shown as 6, 4, 2, or 0. This limitation can be solved by using a binary representation of unary: $v = 6_d = 111111_u \rightarrow 0111111_b \rightarrow S = \{1, 2, 3, 4, 5, 6\}$ can be shown as $v' \in \{6, 5, 4, 3, 2, 1, 0\}$. The overhead is important (l bits data is encoded with 2^l bits) and thus unary has to be restricted to small values.
- *unary less-or-equal*: unary representation a similar approach can be used for less-or-equal too: $v = 2_d \rightarrow 1111100_b \rightarrow S = \{3, 4, 5, 6, 7\}$ can be transformed in $v' \in \{2, 3, 4, 5, 6, 7\}$.
- *frozen*: values are encoded so that they cannot be changed. In this case, the number of bits have to be larger: l bits becomes $l + \lfloor \log_2(l) \rfloor + 1$ bits. For instance, $13_d \rightarrow 0001101_b, c = 100_b \rightarrow 0001101|100_b \rightarrow S = \{7, 6, 4, 3\}$. The checksum c represents the number of bits equal to zero, any modification of the value increase the number of zero but the checksum can only be decreased. It is not possible to change frozen values.
- *blocks*: data are cut into blocks. Each block is encoded with one of the previous schemes.

5.3 Example: Location- and Time-Stamper

This section describes how the previous encoding schemes are used. A location- and time-stamper (LTS) certifies that some entity has been in a given place at a given time. The proof can be provided by a cell-phone operator that locates subscribers, by a beacon in a building, or even by using some distance bounding protocol. A LTS can define logical location (e.g. continent, country, department, town, quarter, building, room) or geographic location (longitude, latitude). We only focus on the latter case because it does not require the definition of a complex data structure.

A location- and time-stamper company can deploy a network of public terminals and sensors. When Alice plugs her smart card in a terminal or when she passes a wireless sensor, she receives a location-and-time stamp with the following attributes: time (UTC, date) and location (latitude, longitude). Table 3 shows an example of the attributes that could be delivered by some LTS in Eurecom Institute.

It can be represented by four attributes `[180432, 24112003, 436265, -0070470]` that can be divided into frozen blocks: `[18|04|32, 24|11|2003, 43|62|65, -007|04|70]` the meaning of each block is publicly known: LTS defines his public key as n and a set of e. For instance, e_1 is the least significant bit of the time in seconds (0-59 : 6 bits), e_6 is the most significant bit of the time

Value	Meaning
180432	UTC in hhmmss format (18 hours, 4 minutes and 32 seconds)
24112003	Date in ddmmyyyy format (November 24, 2003)
43.6265	Geographic latitude in dd.dddd format (43.6265 degrees)
N	Direction of latitude (N - North, S - South)
007.0470	Geographic longitude in ddd.dddd format (7.047 degrees)
E	Direction of longitude (E - East, W - West)

Table 3. Context data: location and time

in seconds, e_7 is the LSB of checksum of time in seconds, etc. If a location- and time-stamper provides the following credential to Alice:
`[18|04|32, 24|11|2003, 43|62|65, -007|04|70]`, she can sign a document with a subset of this credential.
`[18|XX|XX, XX|XX|XXXX, 43|62|65, -007|04|70]`, i.e. the document is signed by *someone that was in the building someday around six o'clock*. Or
`[XX|XX|XX, 24|11|2003, 43|XX|XX, -007|XX|XX]`, i.e. *someone who was in the South of France the* 24^{th} *of November*.

Hidden attributes are different than zero values (`XXX` $\neq$ 000). Indeed, `XXX` is represented as 000|00 and is not equal to 000 that is defined as 000|11. Thus it is not possible to convert 09:08:30 into 09:00:30. The only way to suppress minutes is to remove seconds as well: 09:`XX`:`XX`. This value does not mean that some action occurred at nine o'clock but that it occurred between nine and ten o'clock.

Similarly, a company can qualify customers as *Platinum, Gold, or Silver*; a state can provide digital Id cards to citizen to certify gender, name; a company can provide credentials that define role, access rights; and a partner can define recommendations. In all those cases, the ability of selecting which attribute is displayed is very important to protect privacy when enabling trust evaluation.

6 Security Evaluation

The security of the scheme is based on the assumptions that the discrete logarithm, the double discrete logarithm and the roots of discrete logarithm problems are hard. In addition it is based on the security of Schnorr and RSA signature schemes and on the additional assumption of [16] that computing membership certificates is hard.

Our proposal is based on the group signature scheme of [16], whose join protocol is subject to a collusion attack [2]. Modifications suggested in [29] and that prevent this attack have been taken into account. Even with this modification, there is no proof that the scheme is secure. The security does, however, rest on a well-defined number-theoretic conjecture.

6.1 Unforgeability of Signature

The signature produced by the above protocol is not forgeable. Specifically, only an entity having received a given credential could have issued this signature. This holds because, in the random oracle model, spk_1 proves that the signer knows his secret, spk_3 proves that the signer knows a credential's secret, and spk_4 proves that the signer knows a credential corresponding to both secrets. That is, spk_1 and spk_3 respectively show that

$$\hat{z_2} = \hat{g}^{(a^\alpha)} \quad \text{and} \quad \hat{z_3} = \hat{g}^{(b^\delta)}$$

and therefore:

$$\hat{z_2}\hat{z_3} = \hat{g}^{(a^\alpha + b^\delta)}$$

Whereby integers α and δ are known by the signer. On the other hand, spk_4 proves that

$$(a^\alpha + b^\delta) = \gamma^{e_{h'}}$$

for some γ that the signer knows. Under the hardness assumption on the unforgeability of credentials, this can only happen if the signer received a credential.

6.2 Unforgeability and Integrity of Credentials

In order to code attribute values, a set of different e_i and d_i are used with the same modulo n. However, the common modulus attack does not apply here because each d_i is kept secret and each modulo n is known by a single entity as with the standard RSA. Because there are multiple valid signatures for a given message, this scheme seems to make easier brute force attacks that aim at creating a valid signature for a given message: an attacker can choose a message m and a random $d_R \in_R \mathcal{Z}_n$ and compute a signature $m^{d'} \bmod n$. If e_i and d_i are defined for $i \in \{1, \ldots, k\}$, there are 2^k valid $d = \prod_{i \in S' \subseteq S} d_i$. The probability that a random d_R be acceptable is 2^k times higher than with standard RSA where $k = 1$. However, even if the number of possible signatures for a given message increases, it is necessary to find out the set S corresponding to the randomly chosen signature. In other words, the attacker has to test whether $\forall S' \subseteq S \mid m \stackrel{?}{=} (m^{d'})^{\prod_{i \in S'} e_i} \bmod n$. There are 2^k possible sets S' to check and thus the security of this scheme is equivalent to RSA.

In some cases, the signature scheme can allow combining attributes of two credentials in order to create a new one: naive credentials $(a^x + 1)^{d_{h_1}}$ and $(a^x + 1)^{d_{h_2}}$ could be used to create $(a^x + 1)^{d_{h'}}$ where $S' \subseteq S_1 \cup S_2$. If h_1 states that Alice was present from 8 a.m. to 10 a.m. and h_2 states that she was present from 4 p.m. to 6 p.m., it is necessary to forbid that Alice could create a h' stating that she was present from 8 a.m. to 6 p.m. To avoid this attack, a unique secret t is associated to each credential. Hence $(a^x + b^{t_1})^{d_{h_1}}$ cannot be combined with $(a^x + b^{t_2})^{d_{h_2}}$.

6.3 Non-Transferability of History

Even when the signature of a message cannot be forged, a desirable goal is to be able to assure that it is not possible to find another message with the same signature. Violation of this property with our protocol would require the generation of two pairs (x, t) and (x', t') so that $a^x + b^t = a^{x'} + b^{t'}$. In order to prevent transferability based on such generation of equivalent pairs, cert_{3c} and spk_5 were included in the protocol. Computing (x', t') from a credential based on (x, t) would thus require computing $x' = \log_a(a^x + b^t - b^{t'})$ which is equivalent to solving the discrete logarithm problem. Our protocol thus assures that the credential received as a proof of context or as a recommendation cannot be transferred. A proof that the generation of equivalent pairs is equivalent to a difficult problem (e.g. the discrete logarithm problem) would allow for important simplifications of the history-based trust establishment scheme.

6.4 Distance-bounding

The security of distance-bounding protocols relies on the previous assumptions and the fact that an attacker cannot forward rapid bit exchange (u_i, v_i) without being detected. Indeed, any modification of the communication channel or any new logical gate increases the round-trip time by few nanoseconds that are detected. The commitment on $\{\ldots, m_i, \ldots\}$ is necessary to ensure that response bits depend on the the challenge bits. More details on the security of such schemes can be found in [4].

7 Related Work

7.1 Unlinkable Credentials

Protecting privacy when revealing location information is a difficult concern. In [11], a solution for defining privacy-aware location system is proposed. The goal of this work is however not to provide proof of location.

Common attributes certificates such as X.509 and SPKI [20] are based on public key infrastructure and thus cannot be made unlinkable because the public key can be recognized.

Some works [13, 8] already allow for privacy-preserving attribute verification. However, the target of those works is anonymous attribute certificates and untraceable access control. Credentials defined in [13] rely on pseudonyms and thus it is necessary to know the verifier before starting the challenge-response protocol. Credentials defined in [8] do not ensure non-transferability and have to be used only once to ensure untraceability. The one-time property of these credentials also does not suit multiple interactions as required by our scenario. Table 4 compares different approaches with our scheme. All approaches provide a way to prove some attributes without being traceable. Non-transferability means that a user cannot let another person use one of his credential without revealing some valuable data: private key in public key infrastructures or all pseudonyms and

credentials in "all or nothing" (AoN). Idemix is a pseudonym scheme and thus cannot be used to sign documents, i.e. off-line verification by unknown parties. Finally only our scheme is dedicated to location and has been integrated with a distance-bounding protocol.

Properties	**Group signatures**	**Brands' credentials**	**Camenisch Idemix**	**History-based Sig**
Unlinkable	X	X	X	X
Modify attributes	-	X	X	X
Non-transferable	(PKI)	-	AoN or PKI	PKI
Multi-use	∞	1	1 or ∞	∞
Anonymity Revocation	-	-	X	-
Combine Credentials	-	-	X	X
Signature Scheme	X	X	-	X
Integration with DBP	-	-	-	X

Table 4. Comparison of different schemes

7.2 Location Systems

Using information on the user's context to evaluate trust or define rights is not new: [17] proposes a generalization of the role-based access control paradigm taking into account contextual information. Location verification techniques range from ultrasound-based challenge response [33] to distance bounding protocols [9].

There are many techniques to get one's location or to locate devices [24], the global positioning system (GPS) being one of the most widely deployed. Cellphones operators can locate phones knowing the cell and using triangulation. Like this they can propose location service to their customers. Active Badge [34] has been the first indoor location system. Badges used infrared to emit identifiers that were collected by fixed infrared sensors and transmitted to a server. Active Bat [23] use ultrasound time of flight to provide more accurate physical positioning than Active Badge. Users and objects carry Active Bats tags. A controller sends a short range radio signal that synchronizes sensors and makes the bat emits an ultrasonic pulse that is detected by surrounding sensors. Each sensor computes the distance to the bat and sends it to a server that find out the location of the bat. Cricket [28] proposes a passive for indoor location similar to GPS: a set of ultrasound emitters sends signals to objects that perform their own triangulation to know their location. Radar [7] is a building-wide tracking system based on WLAN. Base stations use the signal strength of wireless devices to evaluate their distance. Magnetic tracker generates axial magnetic-field pulses so that receivers can compute their orientation and location. Authors of [22] propose the combination of different location techniques. Works on location in ad-hoc and sensor networks [5] are increasing.

Those approaches do not address security. Location system can be attacked, e.g. GPS signal can be spoofed. Moreover, knowing its location is not sufficient to prove it. The remaining of this section presents different approaches that can be used to prove one's location. Few of those appoaches work when a colluder acting as a proxy is at the controlled location.

Location-Limited Channels Location-limited channels (or constrained channels [27]) aim at exchanging some secret between two physical entities and thus assure the proximity of two devices. An obvious implementation is to have a physical contact or a wire between both artifacts. This mechanism can be used permanently, e.g. smart card plugged in a point of sale terminal, or during some initialization or pairing protocol. In other words, two devices can be physically linked for exchanging their public keys or for sharing a secret key. Thanks to this initial key exchange, confidentiality, integrity, and authentication can be ensured on any wireless communication channel. In [30], Stajano proposed to initially share a secret through a contact channel in order to secure subsequent interaction based on wireless channels. This model was extended to address peer-to-peer interactions [31]. This basic concept has been extended in order to go without wire. Active badge were already using infrared as a local channel. In [10], IrDA is used as a location-limited channel and physical contact is also seen as an option. IrDA is interesting because it is short-range and directional enough to select a specific device. Limited-range radio channels allow local exchanges of secrets [15, 12].

Context Sharing A straightforward extension of location limited channels is to use some contextual data to initiate the key exchange: all devices within some space can observe the environment in order to share some local knowledge. For instance, a beacon can broadcast some secret to all devices within a room. This secret can be used to bootstrap some secure group among entities that are present. In [19] the signal of Global Positioning System (GPS) satellite is used. GPS signal of a set of up to twelve satellites is used to compute a local secret and exchange keys. Users can be involved to force some context sharing. Bluetooth, in its most secure configuration requires the user to enter a PIN into both devices in order to secure their first communication. The pairing mechanism of smart-its friends [25] also involve the user. He has to shake artifacts together in order to create a common movement pattern that is subsequently used to bootstrap the security of communications. Some secrets, especially GPS signal, can be computed by remote attackers and thus do not protect the initialization. However, it is possible to provide secure enough contextual secrets that are not available to attackers.

Certification of Fixed Location Another approach to verify proximity is to compare the location of two devices. The verifier is mobile and knows his location (e.g. GPS, cell phone network). The prover is fixed and is certified as standing in a given place. The verifier obtains his own location and gets the certificate of the server (e.g. ATM) that should be in front of him. The verifier checks the certificate, computes the distance and direction of the server, and displays it. The user can verify that he is in front of an appliance certified by some entity,

e.g. a bank. This solution avoids physical proxy attacks but impose that only fixed devices can be verified. In other words, the prover can prove his location to the verifier but the verifier cannot prove his location.

Isolation A widely deployed solution to check whether a physical entity knows a secret is to isolate it. To use a Faraday cage during identification is not new [3] and is used in ATM to check that a smart card embeds a private key. This approach forbids physical proxy attacks but is difficult to deploy. Indeed, it is possible to install a Faraday cage and to put two artifacts inside during the pairing process. However, it is not user-friendly enough.

Unforgeable Channel The goal of unforgeable channels is to use communications media that are difficult to create without knowing some secret. For instance, channel hopping or RF watermarking makes it difficult to transfer data necessary to create the signal in another place. Authors of [1] propose a new implementation of *Identification between Friend or Foe* (IFF). Since IFF systems were introduced during World War II, they become today an essential part of military equipment, be it a ship, an aircraft, or even a soldier. Authors propose to use channel hopping: tamper resistant devices secretly choose the next channel to use. A proxy has to listen to and forward all used channels because it does not know which channel is listened by the impersonated device. When numerous channels are available to a large set of devices, the bandwidth necessary to forward potential message becomes unaffordable. Another approach is to hide some geographical information in data packets (packet leashes [36]) or even in the radio frequency signal form. Like this, a proxy cannot forward the signal when each communicating artifact knows its location. Channel hopping seems very promising because it could be integrated within standard communication systems (GSM, UMTS, Bluetooth, 802.11, etc.). Unfortunately, when two given artifacts are communicating, it is easy to detect which channels are used. Channel hopping seems more appropriate when a large infrastructure exists, e.g. cell-phone network. In this case, the proxy cannot detect which channel is used by the infrastructure to communicate with the real device and thus has to forward all channels used by the infrastructure to communicate with the numerous surrounding devices.

Time of Flight Different solutions exists to evaluate the distance between two devices. This subsection describes various approaches that rely on the speed of sound and/or light. Sound and especially ultra-sound is interesting to measure distance because it is slow enough to authorize computation without reducing the accuracy of the measure. Authors of [33] go one step forward: the verifier sends a radio challenge to the prover that responds with an ultrasound response. The processing delay due to the computation of the response is shorter than the propagation time of the ultrasound media and only reduces the precision. This technique shows that a device or a collaborator, i.e. a proxy, has a presence around a verifier. It cannot be used to fight against mafia fraud attacks. Some works rely on the speed of light when measuring the round trip time of a message to evaluate the distance to the prover. However, one meter accuracy implies responding within few nanoseconds and thus it cannot be done through stan-

dard communication channels and cannot implies cryptography [35]. Distance-bounding protocols [4] are necessary in this case.

Conclusions and Future Work

This paper introduces a *history-based* trust establishment protocol that makes it possible to evaluate the trustworthiness of a previously unknown entity based on its past behavior. This protocol also preserves the privacy of users that are anonymous and untraceable and that may reveal a fuzzier set of information than carried by their history credentials. In this scheme, users collect credentials (proof of location, recommendation, etc.) in order to build a provable history. A large variety of attributes may be used for defining trust: recommendations, contextual proofs, reputation, and even hierarchical relationships.

This paper focuses on situations where the location of the user matters and should be asserted as a contextual proof by a trusted authority. Such authorities need not be anonymous and may be referenced in a classical public key infrastructure for instance. This assumes the availability of an infrastructure of authorities such as location- and time-stampers using distance-bounding protocols in order to verify that the requester is close enough before delivering a proof.

Some important work still has to be done. In particular, it is well-known that signatures based on the proof of knowledge of a double discrete logarithm are not efficient in terms of computational complexity and it would be important to find out if more efficient ways to define history-based schemes exist.

There is also a definite need for a history credential management layer able to decide, for instance, whether to glean and store a context proof in some location rather than another one, or how long should such a credential be stored. This layer should also be responsible for controlling the privacy level of the overall scheme when history credentials issued by the same authority or by different ones are submitted together to form a more complex proof, in order to be considered more trustworthy. Techniques like statistical disclosure control [11] should be studied in order to define if such combinations are possible without threatening one's privacy.

Finally, collaborative services implemented on top of ad-hoc networks seem to make up for an interesting application for these mechanisms, in particular in the case of sparsely populated networks in which being able to self-carry one's history of past interactions would permit new security checks. In these architectures however, each entity acts as a signer and as a credential provider at the same time, contrary to an architecture in which authorities are distinguished as self-standing. It remains to be seen if the credential client and server roles can be combined or on the contrary if they should remain separate.

References

1. A. Alkassar and C. Stuble, *Towards secure IFF: preventing mafia fraud attacks*, in proceedings of MILCOM 2002, pages 1139–1144, volume 2, 2002.

2. G. Ateniese and G. Tsudik. *Some open issues and new directions in group signatures*. In Proceedings of Financial Cryptography99, volume 1648 of LNCS, pages 196211. Springer-Verlag, 1999.
3. S. Bengio and G. Brassard and Y. Desmedt and C. Goutier and J.J. Quisquater, *Secure Implementation of Identification Systems*, in Journal of Cryptology, volume 4, number 3, pages 175–183, 1991.
4. S. Brands and D. Chaum. *Distance-bounding protocols (extended abstract)*, in proceedings of EUROCRYPT 93, volume 765, pages 23–27, LNCS, 1993.
5. N. Bulusu, J. Heidemann, and D. Estrin, *GPS-less low-cost outdoor localization for very small devices*, IEEE Personal Communications, volume 7, number 5, pages 28–34, 2000.
6. L. Bussard, R. Molva, and Y. Roudier *History-Based Signature or How to Trust Anonymous Documents* in proceedings of Second International Conference on Trust Management (iTrust'04), Oxford, UK, March 2004.
7. P. Bahl and V.N. Padmanabhan, *RADAR: An In-Building RF-Based User Location and Tracking System*, in proceedings of INFOCOM'00, volume 2, pages 775–784, 2000.
8. S. Brands. *A technical Overview of Digital Credentials*. Research Report, February 2002.
9. L. Bussard and Y. Roudier, *Embedding Distance-Bounding Protocols within Intuitive Interactions*, in Proceedings of Conference on Security in Pervasive Computing (SPC'2003), Boppard, Germany, March, 2003.
10. D. Balfanz, D.K. Smetters, P. Stewart, and H. Chi Wong *Talking to strangers: Authentication in adhoc wireless networks* in proceedings of Symposium on Network and Distributed Systems Security (NDSS '02), 2002.
11. A. Beresford and F. Stajano, *Mix Zones: User Privacy in Location-aware Services*, in proceedigs of workshop on Pervasive Computing and Communications Security (PerSec'04). pages 127–131, 2004.
12. D. Caswell and P. Debaty, *Creating Web Representations for Places*, in proceedings of Handheld and Ubiquitous Computing: Second International Symposium, HUC 2000, LNCS 1927, 2000.
13. J. Camenisch and A. Lysyanskaya, *An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation*, LNCS 2045, 2001.
14. J. Camenisch and M. Michels *Separability and Efficiency for Generic Group Signature Schemes*, in Advances in Cryptology - CRYPTO '99, volume 1666 of LNCS, pages 106-121, Springer Verlag, 1999.
15. M. Corner and B. Noble, *Zero-interaction authentication*, in proceedings of Conference on Mobile Computing and Networking (MobiCom), 2002.
16. J. Camenisch and M. Stadler. *Efficient group signature schemes for large groups*. In Advances in Cryptology, CRYPTO '97 Proceedings, LNCS 1294, pages 410–424, Santa Barbara, CA, August 1997.
17. M.J.Covington, M.J.Moyer, and M.Ahamad, *Generalized Role-Based Access Control for Securing Future Applications*. In 23rd National Information Systems Security Conference (2000).
18. Nathan Dimmock. *How much is 'enough'? risk in trust-based access control*, In IEEE International Workshops on Enabling Technologies (Special Session on Trust Management), June 2003.
19. D.E. Denning and P. F. MacDoran, *Location-Based Authentication: Grounding Cyberspace for Better Security*, in Computer Fraud and Security, February 1996.
20. C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T.Ylonen. *Rfc 2693 spki certificate theory*, 1999.

21. Simson Garfinkel. *PGP : Pretty Good Privacy*. International Thomson Publishing, 1995.
22. D. Graumann, W. Lara, J. Hightower, and G. Borriello, *Real-world implementation of the Location Stack: The Universal Location Framework*, in proceedings of the 5th IEEE Workshop on Mobile Computing Systems & Applications (WMCSA 2003), pages 122–128, 2003.
23. A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. *The Anatomy of a Context-Aware Application*, In proceedings of Mobile Computing and Networking, pages 59–68, 1999.
24. J. Hightower and G. Borriello, *Location Systems for Ubiquitous Computing*. IEEE Computer, 2001.
25. L.E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, and H-W. Gellersen, *Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts*, in proceedings of UbiComp'01, 2001.
26. D. Ingram. *Trust-based filtering for augmented reality*. In Proceedings of the First International Conference on Trust Management, volume 2692. LNCS, May 2003.
27. T.Kindberg, K.Zhang, and N.Shankar, *Context authentication using constrained channels*, in Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), pages 14–21, June 2002.
28. N.B. Priyantha, A. Chakraborty, and H. Balakrishnan, *The Cricket location-support system*, in proceedings of Mobile Computing and Networking, pages 32–43, 2000.
29. Zulfikar Amin Ramzan. *Group blind digital signatures: Theory and applications*, Master Thesis, MIT, 1999.
30. F. Stajano and R.J. Anderson, *The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks*, in proceedings of Security Protocols Workshop, pages 172–194, 1999.
31. Frank Stajano, *The Resurrecting Duckling - What Next?*, in proceedings of Security Protocols Workshop, pages 204–214, 2000.
32. J.M. Seigneur, S. Farrell, C.D. Jensen, E. Gray, and Y. Chen *End-to-end Trust Starts with Recognition*, in Proceedings of Conference on Security in Pervasive Computing (SPC'2003), Boppard, Germany, March, 2003.
33. N. Sastry, U. Shankar, and D. Wagner. *Secure verification of location claims*, In Proceedings of the 2003 ACM workshop on Wireless security, 2003.
34. R. Want, A. Hopper, V. Falcao, and J. Gibbons. *The active badge location system*, ACM Transactions on Information Systems, volume 10, pages 91–102, 1992.
35. B. Waters and E. Felten, *Proving the Location of Tamper-Resistant Devices*, research report.
36. Yih-Chun Hu, A. Perrig, and D.B. Johnson, *Packet leashes: a defense against wormhole attacks in wireless networks*, in Proceedings of INFOCOM 2003. volume 3, pages 1976–1986, 2003.