Anonymous public-key certificates for anonymous and fair document exchange

N. Zhang, Q. Shi and M. Merabti

Abstract: Two protocols are presented for the issuing and identity tracing of anonymous public-key certificates, used by different parties to engage in an anonymous and fair document exchange without revealing their real identities while still being held accountable for their activities. The certificate issuing protocol allows a party to apply to certificate authorities for anonymous public-key certificates using a certificate already issued. The identity tracing protocol enables a legal authority to trace the pseudonym in an anonymous certificate back to the real identity of the corresponding party. The protocols are also analysed with regard to anonymity and accountability.

1 Introduction

Exchange of valuable documents between different parties is an important activity in electronic commerce, and its applications include certified mail, contract signing, valuable electronic goods exchange, and payment for receipt. Owing to the valuable nature of such documents (e.g. payments and signatures), the exchange must be fair to avoid a situation where some parties can receive their expected documents, while others cannot.

Like any other activities in electronic commerce, accountability is an essential property for fair document exchange. Without its enforcement, fair exchange over the Internet is unlikely to become a reality. In this paper, accountability is concerned mainly with non-repudiation of origin to prevent the sender of a message from falsely denying having sent the message.

Another important property of document exchange is anonymity, which protects the privacy of personal information such as identities and locations. For example, when an individual engages in exchanges of electronic goods with a number of vendors, he/she would like to conceal his/her identity from the vendors to prevent them from assembling a profile of his/her personal interests, lifestyles, whereabouts etc.

Clearly, anonymity is to hide a party's identity, whereas accountability is to expose the party's identity, thereby holding the party responsible for its activities. Therefore, effective solutions are needed to resolve the conflicts between these two properties, in order to achieve anonymous, accountable and fair document exchange between several parties.

Normally, the accountability of a message is achieved by a digital signature on the message produced by the message sender using its private key. To verify the signature, a message recipient ought to hold the sender's public key that must be certified, e.g. by a certificate authority (CA) [1], to provide a secure binding between the sender's identity and public key.

One way to prevent the recipient from knowing the sender's real identity is to use a CA for an anonymous public-key certificate based on the original certificate. The anonymous certificate includes a new public key and a pseudonym. Only the CA issuing the anonymous certificate can establish a link from the pseudonym to the real identity of the sender. In this way, not only can the sender sign their message using the private key associated with the new public key without disclosing their real identity, but also the recipient is still assured of the message authenticity by the anonymous certificate. Should any dispute about the message arise, the CA can trace the sender's pseudonym back to its real identity and hold the sender accountable for the message.

To date, little work has been done to address the issue of anonymous but traceable public-key certificates. The most relevant work [2] has proposed several schemes for a party to use its original certified public key to acquire an anonymous (public-key) certificate from an CA. Each anonymous certificate of the party includes a public key converted from the original one, and shares the same private key associated with the original public key. The main problems with the schemes are two-fold. First, an anonymous certificate issued by a CA can be linked to the original public key and real identity of the party if the CA is compromised. Secondly, if the party's private key is compromised, all the anonymous certificates of the party may be linked to its original public key and real identity.

The aim of this paper is to propose a scheme for a party to use its original public-key certificate to obtain anonymous certificates from multiple CAs, which are difficult to link together even if some of the CAs are compromised, or the private key associated with the original certificate is compromised. Such anonymous certificates can be used directly by existing fair exchange protocols to achieve anonymous and fair document exchange. The proposed scheme is implemented by two protocols. A certificate issuing protocol allows a party to apply to CAs for multiple anonymous certificates using a certificate already issued.

© IEE 2000

IEE Proc.-Commun., Vol. 147, No. 6, December 2000
An identity tracing protocol enables a legal authority to trace the pseudonym in an anonymous certificate back to the real identity of its associated party.

2 Anonymous public-key certificates

Assume that there exists a number of CAs $A_i$ for issuing public-key certificates. A certificate with the real identity of a party is called a real (public-key) certificate of that party. Assume also that every party possesses at least one real certificate, and that every party can request any CA to issue an anonymous certificate based on its real certificate or one of its anonymous certificates. This means that multiple anonymous certificates can be issued to a party based on its real certificate.

The reasons for allowing one party to have multiple anonymous certificates are two-fold. First, different certificates of a party can be used for different sessions of document exchange. This prevents a malicious party from linking these sessions of exchange to the same party, as these certificates appear to have no connection. It is important in anonymous applications to avoid such associations. For example, when a party uses the same anonymous certificate for a number of sessions of document exchange with a vendor, the vendor can assemble the party's profile such as personal liking, based on those documents exchanged. If the vendor manages to associate the certificate to a real certificate of the party, then the party's profile can be linked to the party's real identity.

Secondly, anonymous certificates of a party can be issued sequentially (Fig. 1), to complicate association of an anonymous certificate with a real one. In Fig. 1, $C_{i,t}$ denotes a real certificate of a party $P_i$ which is issued by a CA $A_{i_0}$ according to $P_i$’s request, and $C_{i,t}(1 \leq t \leq n)$ is an anonymous certificate of $P_i$ which is issued by $A_{i_t}$ upon $P_i$’s request using certificate $C_{i,t-1}$. Note that some of these CAs may be identical. If we only allow the link from $C_{i,t-1}$ to $C_{i,t}$ to be established jointly by $A_{i,t-1}$ and $A_{i_t}$, then an adversary has to compromise all the CAs in the sequence in order to link $C_{i,t}$ to $C_{i,t}$. This is difficult to accomplish in practice.

![Fig. 1 Sequential use of anonymous certificates](image)

The basic idea of the sequential certificate issuing shown in Fig. 1 is analogous to that of untraceable authentication for mobile users [3]: a mobile user can request a recently visited domain to ensure their solvency to the domain which the user is currently visiting. This prevents an unauthorised party or a visited domain authority from tracing the mobile user’s home domain and identity. However, the use of anonymous certificates described in this paper has the following considerations.

- Control over the number of anonymous certificates issued: an unrestricted number of anonymous certificates for a party may impose heavy communication and processing loads on CAs, which could lead to degradation of their performance and problems with certificate management. Thus the number of anonymous certificates issued to a party should be limited.

- Certificate expiration times: to simplify certificate management and to increase certificate assurance, when a certificate of a party expires, it would be desirable for all the certificates, issued directly or indirectly using the expired certificate (i.e. all its down-stream certificates), to expire as well. If the same expiration time is used for different certificates of a party, then the time could be utilised by an adversary to link these certificates to the same party. Hence certain measures should be taken for determination of expiration times.

- Real identity tracing: when a dispute occurs about a message signed with the private key associated with an anonymous certificate of a party, a legal authority can be called to trace the anonymous certificate back to its corresponding real certificate, and thus hold the party accountable for the message. To fulfill such tracing, the legal authority must possess undeniable and verifiable evidence to prove the link from the anonymous certificate to the real one.

In the subsequent sections, we address these issues in detail, and formally present the protocols for anonymous certificate issuing and real identity tracing.

3 Notation

The notation to be used throughout this paper is summarised as follows:

- $(pk_{i,a}, sk_{i,a})$ is a pair of public and private keys for a party $P_i$ certified by a CA $A_{i_0}, (pk_{i,a}, sk_{i})$ with a single subscript is a pair of public and private keys owned by a CA $A_i$.
- $E_k(x)$ expresses the ciphertext of a data item $x$ encrypted with a key $k$. $E_k(x)$ is computed using a public-key cryptosystem if the corresponding decryption key is different from $k$, and using a conventional cryptosystem otherwise.
- $x \oplus y$ denotes the concatenation of data items $x$ and $y$.
- $h(x)$ is a one-way hash function with the following properties: for any $x$, it is easy to compute $h(x)$; given $h(x)$, it is difficult to compute $x$; and given $x$, it is difficult to find $x'$ such that $h(x') = h(x)$.
- $P_i \rightarrow m$ expresses that a party $P_i$ sends a message $m$ to another party $P_j$ through a normal communication channel.
- $P_i \rightarrow m$ signifies two cases. In the first case, $P_j$ sends $m$ to $P_j$ through an anonymous communication channel, so that $P_i$’s identity and address are hidden from $P_j$. This means that $P_j$ knows $P_i$’s address/identity, but $P_j$ does not know $P_i$’s’s identity. If $P_j$ requires $P_i$ to respond to its message, $P_j$ needs to send to $P_i$ $m$ together with an anonymous reply address. This is omitted from this notation for simplicity. The details of such anonymous communication are described elsewhere [4, 5].

In the second case, $P_j$ responds to an anonymous message from $P_i$ with $m$ using the anonymous reply address defined in $P_j$’s message. This implies that $P_j$ knows $P_i$’s address/identity, but not conversely. The notation does not distinguish between the above two cases as they can be easily separated based on the context of the protocol presentation to be given.

$C_{i,b}$ is a (real or anonymous) certificate of a party $P_i$ issued by a CA $A_{i_0}$, which is denoted as $C_{i,b} = (A_{i_0}, I_{i_0}, pk_{i,b}, e_{i,b}, s_{i,b})$, where $s_{i,b} = E_{i_0}(h(A_{i_0}, I_{i_0}, pk_{i,b}, e_{i,b}))$ is $A_{i_0}$'s signature, and $I_{i_0}, pk_{i,b}$, and $e_{i,b}$ are a real identity or pseudonym of $P_i$ public key and certificate expiration time. We also assume that every party knows the public keys of all the CAs.

4 Anonymous certificate issuing

4.1 Certificate issuing protocol

To limit the number of anonymous certificates, we assume that each party is permitted to have a tree-like structure of certificates with one of its real certificates as the root.
Fig. 2 depicts such a certificate tree, where the root node is a real certificate of a party $P_i$ issued by CA $A$, and the others are anonymous certificates issued directly or indirectly based on the real certificate. For example, $C_{lb}$ in the Figure is the parent of $C_{ib}$, issued directly using $C_{ia}$ by $A_b$, and the grandparent of $C_{ib}$, issued indirectly through $C_{ia}$ by $A_b$. Clearly, if $P_i$ has $m$ real certificates, then there will be $m$ corresponding certificate trees for $P_i$.

![Certificate tree diagram]

For a certificate tree, we impose limitations on the number of its levels and the number of child certificates for each certificate. To fulfill this, we require that every CA $A_j$ specifies two constants $m_l$ and $m_c$, which remain unchanged to any certificate issued by $A_j$. $m_l$ states that for any real certificate issued by $A_j$, the number of levels for its corresponding certificate tree cannot exceed $m_l$. $m_c$ means that, for any real or anonymous certificate issued by $A_j$, if it is permitted to have any child certificates, then the number of its child certificates cannot exceed $m_c$. A certificate with no child is called a leaf of the tree.

For example, the certificate tree in Fig. 2 meets the limitations with regard to the numbers $m_l$ and $m_c$ ($j \in \{a, b, u, v\}$) shown. This is because the number of certificate levels is 3 ($= m_l$), and the number of child certificates for any certificate $C_{ia}$ is not greater than $m_c$. Note that numbers $m_l$ ($j \in \{a, b, u, v\}$) are omitted as they have no effect on the tree. Obviously, $C_{ia}$ and $C_{ib}$ are not allowed to have any child certificate although $m_c = 2$ and $m_c = 3$, as they are on the last level of the tree. $C_{ia}$ cannot have any more child certificates due to $m_c = 2$. $C_{ib}$ can have one more child certificate since $m_c = 3$, and $C_{ia}$ cannot have a child due to $m_c = 0$.

When a party $P_i$ requests a CA $A_j$ to issue a new anonymous certificate $C_{ib}$, based on an existing one $C_{ia}$, the expiration time $e_{ia}$ of $C_{ia}$ should not be later than that $e_{ia}$ of $C_{ib}$, as discussed in Section 2. To issue $C_{ib}$, $A_j$ asks the CA $A_j$, which has issued $C_{ia}$, to authenticate $P_i$’s request, as $A_j$ is not permitted to view $C_{ia}$. $A_j$ responds to $A_i$, with a guarantee of the authenticity and the latest expiration time $e_{ib}$ ($\leq e_{ia}$) of $C_{ib}$ without knowing $C_{ib}$ (as explained later). The main problem with $e_{ib}$ is that if $A_j$ is compromised and knows that very few certificates will expire before $e_{ib}$, then $A_j$ could identify $C_{ib}$ and link it to $C_{ia}$. Similarly, compromised $A_j$ could guess $C_{ib}$ if very few certificates expire after $e_{ib}$.

To avoid the problem by hiding $A_j$ from $A_i$, allowing the certificate tree to have a sufficient number of levels, and letting $P_i$ mainly use certificates on the last few levels of the tree. In this way, although an adversary could link $C_{ib}$ to $C_{ia}$ if the adversary compromises $A_j$, it would still be very difficult for the adversary to associate a certificate on one of the last few levels to its corresponding real certificate, as discussed in Section 2.

By combining the certificate tree and expiration time discussed above, we can define a granted right $gr_{ib} = (s_{ib}, k_{ib})$ for a CA $A_b$ to issue an anonymous certificate for a party $P_i$. This right states that $A_b$ can issue a certificate $C_{ib}$ for $P_i$ with an expiration time before $e_{ib}$, and permits $A_j$ to have a certificate sub-tree with $C_{ib}$ as the root and a maximum of $s_{ib}$ levels. Obviously, if $s_{ib} = 0$, $A_j$ is not allowed to issue any certificate for $P_i$. $gr_{ib}$ is granted by CA $A_j$, which has issued parent certificate $C_{ia}$, which is in turn granted by the CA that issued its parent certificate, and so on. When $C_{ia}$ is the real certificate, $gr_{ib}$ is $(s_{ib}, k_{ib})$ defined by $A_j$, $gr_{ia}$, and $gr_{ib}$ should meet the conditions $s_{ib} = s_{ia} - 1$ and $k_{ib} = k_{ib}$.

Fig. 2 shows an example of such rights where $e_{ib}, e_{ia}, s_{ib} < e_{ia}$ and $e_{ib}, e_{ib} < e_{ib}$. For example, $gr_{ia}$ states that the expiration time of $C_{ia}$ must be earlier than $e_{ib}$ and that the certificate sub-tree with $C_{ia}$ as the root is allowed to have a maximum of two levels due to $s_{ia} = 2$. Evidently, $gr_{ib}$ and $gr_{ia}$ do not permit $C_{ib}$ and $C_{ia}$ to have any child certificates because $s_{ib} = 1$ and $s_{ia} = 1$, respectively. $gr_{ia}$ is defined by $A_i$, with $s_{ib} = m_l - 3$, $k_{ib}$ is granted to $A_j$ by $A_i$, with $s_{ib} = s_{ia} - 1 = 2$, and $gr_{ib}$ is granted to $A_j$. By $A_i$, with $s_{ib} = s_{ia} - 1 = 1$. Similar explanation can be applied to the other certificates.

We can now describe the protocol for issuing an anonymous certificate, which consists of four stages. Suppose that a party $P_i$ requests a CA $A_j$ to issue a new certificate $C_{ib}$, based on an existing certificate $C_{ia}$ issued by a different CA $A_i$. In stage 1, $P_i$ sends $A_j$ its request, which includes identity $A_i$ and an item $x_{ib}$, verifiable only by $A_j$, over an anonymous channel without revealing its identity to $A_j$. In stage 2, $A_j$ simply forwards $x_{ib}$ along with other relevant items to $A_i$. In stage 3, $A_i$ verifies $x_{ib}$ to prove the authenticity of $P_i$’s request, grants $A_j$ a right for issuing a certificate for $P_i$ accordingly, and transfers the right to $A_j$. In stage 4, $A_i$ issues a new certificate $C_{ib}$ based on the right granted.

Table 1: Certificate issuing protocol and associated definitions

<table>
<thead>
<tr>
<th>Stage</th>
<th>Protocol</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>$P_i \rightarrow A_j: E_{ib}(A_j, pk_{ib}, s_{ib}, u_{ib}, x_{ib})$</td>
<td>$P_i$ randomly chosen by $P_i$</td>
</tr>
<tr>
<td>2.</td>
<td>$A_j \rightarrow A_i: x_{ib}$</td>
<td>$P_i$’s identity in $C_{ia}$ issued by $A_i$</td>
</tr>
<tr>
<td>3.</td>
<td>$A_i \rightarrow A_j: s_{ib}, E_{ib}(s_{ib})$</td>
<td>$A_j$’s public key to be certified</td>
</tr>
<tr>
<td>4.</td>
<td>$A_j \rightarrow P_i: C_{ib}$</td>
<td>$A_j$’s certificate issued by $A_j$</td>
</tr>
</tbody>
</table>

Table 1 illustrates this certificate issuing protocol, together with the definitions of the items used. The protocol is explained below.

1. In this stage, $P_i$ first decides a new pair of public and private keys ($pk_{ib}, sk_{ib}$), and chooses two random numbers $u_{ib}$ and $v_{ib}$. $v_{ib}$ is used as the current session number. $u_{ib}$ is a secret number used for the calculation of a session.
key $k_{ib}$. $P_i$ then computes $k_{ib}$, $a_{ib}$, and $x_{ib}$ defined in Table 1. Here, the encryption for $x_{ib}$ with $A_i$’s key $p_{ib}$ is to allow only $A_i$ to know $P_i$’s identity $I_{ia}$. The encryption for $a_{ib}$ with $P_i$’s secret key $s_{ia,b}$ is to allow $A_i$ to authenticate $P_i$. $x_{ib}$ allows $A_i$ to acquire from $A_i$ an approval of certifying public key $p_{ik,b}$ related to $k_{ib}$ without letting $A_i$ know $p_{ik,b}$. If necessary, a time stamp can be added into the message.

(2) In this stage, $A_i$ first decrypts $P_i$’s message with key $sk_i$ and verifies that $A_i$ is a valid CA. If the verification fails, $A_i$ terminates the protocol run. Note that, in the following, a failure of any verification will lead to such protocol termination without being explicitly mentioned. If the verification succeeds, $A_i$ saves all the items and forwards $x_{ib}$ to $A_i$.

(3) In this stage, $A_i$ decrypts $x_{ib}$ with $sk_i$, and verifies that identity $I_{ia}$, $x_{ib}$ has a certificate $C_{ib}$ issued and held by $A_i$, which has neither expired nor been revoked. If the verification is positive, $A_i$ decrypts $a_{ib}$ in $x_{ib}$ with key $p_{ik,b}$ in $C_{ib}$, to obtain $I_{ia}$, $I_{ib}$, $s_{ia,b}$, and $k_{ib}$. If $A_i = A_j$ and $I_{ia} = I_{ia}$, $A_i$ is convinced that the request originated from $P_i$.

Based on expiration time $e_{ib}$ in $C_{ib}$ and right $gr_{ib} = (st_{ia}, le_{ib})$, $A_i$ checks that

(a) $st_{ia} > 1$, i.e. $A_i$ is permitted to grant child certificates for $P_i$.
(b) the number of unexpired child certificates with regard to their latest expiration times, granted for $P_i$, by $A_i$, is less than $mc$, defined earlier, i.e. $A_i$ can grant another child certificate for $P_i$.

If both conditions (a) and (b) hold, $A_i$ chooses a future time $le_{ib} (e_{ib})$ that makes the number of existing certificates, issued by $A_i$, and to expire after $le_{ib}$ sufficiently large to avoid an adversary guessing $C_{ib}$ using compromised $A_i$, as discussed earlier. $A_i$ then assigns $A_i$ the right $gr_{ib} = (st_{ia} - 1, le_{ib})$. If either condition (a) or (b) is not met, the right $gr_{ib} = (0, 0)$.

Additionally, $A_i$ computes $s_{ib}$ (defined in Table 1), and sends $E_{A_i}(s_{ib})$ to $A_i$. Here, key $k_{ib}$ is used for the encryption $E_{A_i}(s_{ib})$, because $A_i$ does not know the identity of $A_i$.

Note that $A_i$ needs to store $a_{ib}$, $k_{ib}$, $e_{ib}$, $A_i$ when $st_{ia} = 1 > 0$. These are needed for identity tracing (present later).

(4) In this stage, $A_i$ decrypts $E_{A_i}(s_{ib})$ with $k_{ib}$ and then $s_{ib}$ with $p_{ik,b}$. $A_i$ examines whether $a_{ib}$ and $k_{ib}$ in $s_{ib}$ are correct, and that $s_{ib}$ in $gr_{ib}$ is greater than 0 ($st_{ia} > 0$). If the result is positive, $A_i$ generates a unique pseudonym $I_{ib}$, chooses an expiration time $e_{ib}$ ($e_{ib}$), and issues the following certificate:

$$s_{ib} = E_{sk_i}(h(A_i, I_{ia}, I_{ib}, p_{ik,b}, e_{ib}, e_{ib, b}))$$

$$C_{ib} = (A_i, I_{ia}, p_{ik,b}, e_{ib}, s_{ib})$$

If $A_i$ is not permitted to issue a certificate for $P_i$ (i.e. $st_{ia} = 0$), $A_i$ executes the following transaction:

$$A_i \rightarrow A_j : s_{ib} : E_{sk_i}(s_{ib}, \text{‘no certification’})$$

For intuitiveness, Table 2 lists the items stored for a certificate $C_{ib}$ by its corresponding CA $A_i$.

### 4.2 Protocol analysis

We now analyse the protocol shown in Table 1 with regard to anonymity and accountability. The anonymity of the protocol is argued based on the following cases:

- Compromised $A_i$ is unable to link $C_{ib}$ to $C_{ia}$ when the number of existing certificates, issued by $A_i$ and to expire after $le_{ib}$, is sufficiently large. This is due to the following reasons. First, the sufficiently large number means that the probability for $A_i$ to guess $C_{ia}$ among these certificates is very low. Second, $P_i$ does not sign the message sent to $A_i$ and the communication between $P_i$ and $A_i$ is conducted over anonymous channels, and so $A_i$ does not know the address and identity of $P_i$. Finally, $gr_{ib}$ granted by $A_i$ does not involve any information on $C_{ia}$. When the number is small, $A_i$ might be able to link $C_{ib}$ to $C_{ia}$, which can be handled by the solution suggested in Section 4.1. A similar discussion can also be applied to the case of linking $C_{ib}$ to $C_{ia}$, by compromised $A_i$.

- $A_i$ and $A_j$ are able to associate $C_{ib}$ to $C_{ia}$ by collusion. This problem can be alleviated when the certificate tree of $P_i$ has a sufficient number of levels, as discussed in Section 2.

The accountability of the protocol is demonstrated by the following cases:

- $A_i$ cannot dishonestly issue $C_{ib}$ under $P_i$’s name for another party $P_j (\neq P_i)$. The reasons for this are that $au_{ib}$ in $C_{ib}$ is signed by $P_i$’s private key $sk_{ia}$, $sr_{ib}$ is $A_i$’s signature on granted right $gr_{ib}$, $k_{ib}$ in both $au_{ib}$ and $sr_{ib}$ prevents $A_i$ from changing key $p_{ib}$ in $C_{ib}$; neither $A_i$ nor $P_i$ knows private key $sk_{ib}$ associated with $p_{ib}$. Similarly, we can show that even by colluding with $A_i$, $A_j$ is still unable to fulfill such dishonest issuing, as $A_i$ and $A_j$ cannot forge $au_{ib}$ and $k_{ib}$. This also implies that $P_i$ cannot falsely deny its request for $C_{ib}$, and that $A_i$ cannot falsely deny granting $gr_{ib}$ for the issuing of $C_{ib}$.

- $C_{ib}$ can be traced back to its corresponding real certificate only by a legal authority with verifiable and non-repudiable evidence (described in Section 5).

- Where another party $P_j$ has compromised $P_i$’s private key $sk_{ib}$ associated with $C_{ib}$, $P_j$ may be able to acquire new certificates using $C_{ib}$, and to abuse them, with $P_i$ being accused for $P_j$’s misbehaviour. This problem can be avoided by allowing $P_i$ to have $m (> 1)$ real certificates and to request a new anonymous certificate using at least $l (l > m)$ existing certificates. In this case, $P_i$ has to compromise $l$ private keys of $P_j$ in order to obtain a new certificate linked to $P_j$, which is harder to fulfill. This implies that $P_i$ has a graph-like structure of certificates. The protocol given in Section 4.1 can be extended to deal with such a certificate structure (to be addressed in future work).

### 5 Real identity tracing

#### 5.1 Identity tracing protocol

When a dispute about a message signed with the private key associated with an anonymous certificate of a party $P_i$ occurs, a legal authority $LA$ can be called to trace the anonymous certificate back to its corresponding real certificate, thereby holding $P_i$ accountable for the message. Here we assume that $LA$ is a trusted authority and does not disclose the link between an anonymous certificate and its corresponding real certificate to any unauthorised party.

To trace an anonymous certificate $C_{ib}$ of $P_i$ back to its corresponding real certificate $C_{ia}$ containing the real identity of $P_i$, a process can be devised to produce the following certificate sequence from $C_{ib}$ to $C_{ia}$ in a bottom-up way:

---

**Table 2: Items stored for certificate $C_{ib}$ by CA $A_i$**

<table>
<thead>
<tr>
<th>Certificate</th>
<th>Stored items</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_{ib}$</td>
<td>$s_{ib}$, $C_{ib}$, $A_i$, $st_{ia}$, $le_{ib}$</td>
</tr>
<tr>
<td>Child 1: $C_{ia}$</td>
<td>$sr_{ia}$, $k_{ia}$, $le_{ia}$, $au_{ia}$</td>
</tr>
<tr>
<td>Child 2: $C_{ib}$</td>
<td>$sr_{ib}$, $k_{ib}$, $le_{ib}$, $au_{ib}$</td>
</tr>
</tbody>
</table>

---

*IEEE Proc.-Commun., Vol. 147, No. 6, December 2000*
This means that, A₀ passes the non-repudiable evidence \( A_{\text{IV}} \) to A₀, which issues C_{\text{IV}}. A₀ then sends its evidence about C_{\text{IV}}'s session number \( n_{\text{IV}} \) and the session number \( n_{\text{IV}} \) to A₀, which issued C_{\text{IV}}. This process continues until A₀, associated with C_{\text{IV}}, has received the evidence from A₀, which issued C_{\text{IV}}. A₀ finally transfers its evidence along with that received to LA. This enables LA to obtain all the evidence about the issuing and granting of the certificates in the sequence, and to assess the authenticity of the link between C_{\text{IV}} and C_{\text{IV}} based on the evidence. Note that the evidence is verifiable only by LA.

Table 3 shows such a protocol for real identity tracing, along with the definitions of the new items used.

<table>
<thead>
<tr>
<th>Table 3: Protocol for real identity tracing and associated definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>T1.</strong> ( \text{LA} \rightarrow A_0: E_{\text{pk}<em>C}(C</em>{\text{LA}}, s_{\text{LA}}, m, \text{id}_{\text{LA}}) )</td>
</tr>
<tr>
<td><strong>T2.</strong> ( A_0 \rightarrow A_0: E_{\text{pk}<em>C}(C</em>{\text{LA}}, s_{\text{LA}}, m, \text{id}<em>{\text{LA}}, d</em>{\text{LA}}) )</td>
</tr>
<tr>
<td><strong>...</strong></td>
</tr>
<tr>
<td><strong>T2.</strong> ( A_0 \rightarrow \text{LA}: E_{\text{pk}<em>C}(C</em>{\text{LA}}, s_{\text{LA}}, m, \text{id}<em>{\text{LA}}, d</em>{\text{LA}}) )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Item</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n_{\text{LA}} )</td>
<td>session number chosen by LA</td>
</tr>
<tr>
<td>( t_{\text{LA}} )</td>
<td>time stamp</td>
</tr>
<tr>
<td>( s_{\text{LA}} )</td>
<td>( E_{\text{pk}<em>C}(n</em>{\text{LA}}, t_{\text{LA}}) )</td>
</tr>
<tr>
<td>( s_{\text{LA}} )</td>
<td>( E_{\text{pk}<em>C}(h(n</em>{\text{LA}})) )</td>
</tr>
<tr>
<td>( e_{\text{LA}} )</td>
<td>( E_{\text{pk}<em>C}(C</em>{\text{LA}}, s_{\text{LA}}, m, \text{id}<em>{\text{LA}}, d</em>{\text{LA}}) )</td>
</tr>
<tr>
<td>( c_{\text{LA}} )</td>
<td>( h(k_{\text{LA}}, \text{id}<em>{\text{LA}}, s</em>{\text{LA}}, m, \text{id}<em>{\text{LA}}, d</em>{\text{LA}}) )</td>
</tr>
<tr>
<td>( e_{\text{LA}} )</td>
<td>( E_{\text{pk}<em>C}(C</em>{\text{LA}}, s_{\text{LA}}, m, \text{id}<em>{\text{LA}}, d</em>{\text{LA}}, a_{\text{LA}}) )</td>
</tr>
<tr>
<td>( a_{\text{LA}} )</td>
<td>( h(e_{\text{LA}}) )</td>
</tr>
</tbody>
</table>

The protocol is explained in detail below:

1. In transaction T1, C_{\text{LA}} is LA’s public-key certificate, and C_{\text{LA}} is the certificate to be traced. S_LA is LA’s signature on the current session number \( n_{\text{LA}} \), which will be passed to all relevant CAs. S_LA is LA’s signature on C_{\text{LA}}.

2. Having received the message from LA, A₀ decrypts it with key \( \text{sk}_{\text{LA}} \) and verifies that C_{\text{LA}} is issued by A₀.

Once the above verification process is successfully completed, A₀ locates items \( s_{\text{LA}}, A_{\text{IV}}, \text{se}_{\text{LA}}, \) and \( s_{\text{LA}} \) stored for C_{\text{IV}} (as illustrated in Table 2), drives key \( k_{\text{IV}} = h(A_{\text{IV}}, \text{se}_{\text{IV}}, s_{\text{IV}}) \), and computes \( e_{\text{IV}} \) and \( c_{\text{IV}} \) (defined in Table 3) to form its message in T2. Here, \( e_{\text{IV}} \) provides LA with the evidence that A₀ issued C_{\text{IV}} upon P’s request (detailed later) and that \( c_{\text{IV}} \) is a message checksum.

Upon the receipt of the message from A₀ through T2, A₀ decrypts it with key \( \text{sk}_{\text{IV}} \) and finds its related certificate C_{\text{IV}} and stored items \( k_{\text{IV}} \) and \( a_{\text{IV}} \). A₀ then examines the correctness of C_{\text{IV}}, S_LA, and \( c_{\text{IV}} \). If correct, A₀ locates another set of items \( s_{\text{IV}}, A_{\text{IV}}, \text{se}_{\text{IV}}, \) and \( s_{\text{IV}} \) saved for C_{\text{IV}}.

Similar to the operations performed by A₀, A₀ computes:

\[
\begin{align*}
k_{\text{IV}} &= h(A_{\text{IV}}, \text{se}_{\text{IV}}, s_{\text{IV}}) \\
e_{\text{IV}} &= E_{\text{pk}_C}(C_{\text{IV}}, e_{\text{IV}}) \\
E_{\text{pk}_C}(n_{\text{IV}}, \text{se}_{\text{IV}}, s_{\text{IV}}, a_{\text{IV}}, h(e_{\text{IV}})) \\
c_{\text{IV}} &= h(k_{\text{IV}}, C_{\text{IV}}, s_{\text{IV}}, m, \text{id}_{\text{IV}}, d_{\text{IV}}, e_{\text{IV}})
\end{align*}
\]

5.2 Protocol analysis

We now examine the anonymity and accountability of the protocol presented in Table 3. The protocol’s anonymity is obvious because information on the certificate sequence is encrypted with LA’s public key, i.e. only LA is able to link together the certificates in the sequence.

The accountability is demonstrated by the following cases:

- The certificate sequence from C_{\text{IV}} to C_{\text{IV}} is undeniable. As described in Section 5.1, for each certificate C_{\text{IV}} in the
sequence, $L_A$ can prove that $P_i$ did request $C_{ib}$ from $A_b$, using its parent certificate $C_{ib}$, that $A_d$ which issued $C_{ib}$ granted $A_b$ the right for the issuing of $C_{ib}$, and that $A_b$ did issue $C_{ib}$ in which the public key and expiration time are correct with respect to the granted right.

- A compromised CA could disrupt the tracing process by not executing transaction $T_2$. If such a case happens, $L_A$ can perform the protocol stepwise in such a way that $L_A$ sends its request to $A_b$ in transaction $T_1$. Instead of transferring evidence $e_{iv}$ to $A_d$, $A_b$ is required to send $e_{iv}$, $m_{ib}$, and $A_d$ directly to $L_A$ in $T_2$. $L_A$ repeats the same operations for every such CA $A_d$. In this way, $L_A$ is able to spot the compromised CA when it refuses to respond to $L_A$’s request, or provides incorrect evidence.

6 Conclusions

We have presented two protocols for anonymous certificate issuing and real identity tracing. The anonymity analysis of the protocols has demonstrated that it is very difficult for an adversary to link an anonymous certificate to its corresponding real certificate, even under the circumstances where the adversary is capable of compromising some of the CAs. The accountability analysis of the protocols has indicated that no CA could issue an anonymous certificate without a request from the certificate holder, and that a legal authority is able to collect verifiable and non-repudiable evidence to prove the link from an anonymous certificate to its associated real certificate.

Different anonymous certificates, coupled with anonymous communication, enable a party to operate existing fair exchange protocols [6–12] for different sessions of document exchange with other parties. Not only can this hide the party’s identity and location, while retaining its messages accountable, but it also prevents these sessions from being linked to the party unlawfully. Thus, the fair exchange protocols can be easily enhanced to achieve anonymity.

7 Acknowledgment

The authors would like to thank John Haggerty for his useful comments.

8 References