Secure Mobile Agent using Strong Non-designated Proxy Signature

Byoungcheon Lee[1], Heesun Kim[2*], and Kwangjo Kim[1]

[1] IRIS (International Research center for Information Security)
ICU (Information and Communications University)
58-4, Hwaam-dong, Yusong-gu, Taejon, 305-732, Korea
{sultan,kkj}@icu.ac.kr
[2] Information Security Technology Division
ETRI (Electronics and Telecommunications Research Institute)
161, Kajong-dong, Yusong-gu, Taejon, 305-600, Korea
sezsez@etri.re.kr

Abstract. It is expected that mobile agent will be widely used for electronic commerce as an important key technology. If a mobile agent can sign a message in a remote server on behalf of a customer without exposing his/her private key, it can be used not only to search for special products or services, but also to make a contract with a remote server. To construct mobile agents, [KBC00] used an RSA-based undetachable signature scheme, but it does not provide server's non-repudiation because the undetachable signature does not contain server's signature.
Mobile agent is a very good application example of proxy signature, and the undetachable signature can be considered as an example of proxy signature. In this paper we show that *secure mobile agent* can be constructed using *strong non-designated proxy signature* [LKK01] which represents both the original signer's (customer) and the proxy signer's (remote server) signatures. We provide RSA-based and Schnorr-based constructions of secure mobile agent, and moreover we show that the Schnorr-based scheme can be used very efficiently in multi-proxy mobile agent situation.

Keywords. Secure mobile agent, strong non-designated proxy signature, multi-proxy signature.

1 Introduction

1.1 Mobile Agent

Mobile agents [FGS96,KKC99,LM99] are autonomous software entities that are able to migrate across different execution environments through network. The characteristics of mobile agents, mobility and autonomy, make them ideal for

* This work was done when she was with ICU.

electronic commerce applications because permanent connections between customers and servers are unnecessary and low-bandwidth connections and asynchronous communications are possible. Furthermore, they provide better support for heterogeneous environments. Mobile agents can be used for electronic commerce in many ways; search and buy special products or services on behalf of a customer, negotiate something with other entities, and sell products on behalf of a shopping mall server.

We consider a scenario that a mobile agent is ordered to search the price of a flight ticket and book it on behalf of a customer. If the mobile agent finds a proper bid presented by a server, the mobile agent will book it by digitally signing the server's bid and the customer's requirement with both customer's and server's keys. To make it possible, the mobile agent must carry in any form the customer's private key and compute with it.

However, mobile agents are vulnerable to several attacks, particularly by malicious hosts. Fundamental problems of executing mobile code in a remote host can be listed as follows [ST97]:

1. Code and execution integrity: Can a mobile agent protect itself against tampering by a malicious server?
2. Code privacy: Can a mobile agent conceal the program it wants to have executed?
3. Computing with secrets in public: Can a mobile agent remotely sign a document without disclosing user's private key?

There have been extensive researches to solve these problems. A reasonable and practical approach is to provide software-based mechanism to prevent any kind of vulnerability actively. Implementing any kind of secure function in mobile agent is difficult because all the code and data of mobile agent are exposed to remote server. One of the best ways to conceal customer's private key and keep the integrity of mobile code is to use cryptographic hard problems such as integer factorization problem or discrete logarithm problem. Undetachable signature scheme is an example.

1.2 Undetachable Signature Scheme

[ST97] introduced the concept of Computing with Encrypted Function (CEF) which tried to conceal the signature function by composing it with encryption function. [KBC00] implemented CEF using an RSA-based undetachable signature scheme. The customer signs his requirement information using RSA signature and builds up an encrypted signature function, and then gives it to mobile agent. Then the server can generate customer's signature on the bid information on behalf of the customer. Customer's private key is hidden in the encrypted signature function and its secrecy is based on the RSA assumption.

Although the undetachable signature scheme of [KBC00] hides customer's private key successfully, it does not provide the fairness of contract. The basic requirement of fair contract is non-repudiations of both parties. The undetachable signature represents only customer's signature and it can be computed by any party, so the server can repudiate his signature generation later. After the booking process of the flight ticket is finished with customer's payment, the server can repudiate his signature generation and refuse to deliver the flight ticket.

A simple solution for this problem is that the server signs his final messages before giving them to the mobile agent, but this is not a neat solution. In Section 4, we propose an efficient strong proxy signature scheme which represents both the customer's and the server's signatures providing the fairness of contract.

The basic concept of undetachable signature scheme is very similar to the delegation of customer's signing capability to unspecified proxy signers. Hereafter we review proxy signature schemes briefly.

1.3 Proxy Signature

Proxy signature is a signature scheme that an original signer delegates his/her signing capability to a proxy signer, and then the proxy signer creates a signature on behalf of the original signer. When a receiver verifies a proxy signature, he verifies the signature itself and original signer's delegation together. The basic methodology of proxy signature is that the original signer creates a signature on delegation information (ID of the proxy signer, or any warrant information) and gives it secretly to the proxy signer, and then the proxy signer uses it to generate a proxy key pair. Because the proxy key pair is generated using original signer's signature on delegation information, any verifier can check original signer's agreement from a proxy signature.

[MUO96] firstly introduced the concept of proxy signature. They classified proxy signatures based on delegation type as full delegation (giving the original signer's private key itself), partial delegation (issuing a new key pair), and delegation by warrant (issuing a certificate stating the delegation information). Partial delegation is further classified as proxy-unprotected and proxy-protected according to protection of proxy signer. They provided various constructions of proxy signature schemes and their security analysis. [KPW97] extended them by using Schnorr signature and including warrant information in partial delegation schemes (partial delegation with warrant). [LKK01] provided several attacks against previous proxy signature schemes and introduced the concept of strong proxy signatures which represent both original signer's and proxy signer's signatures. They also introduced the concept of strong non-designated proxy signature where the original signer does not specify proxy signers in the delegation stage. It is useful when proxy signers cannot be determined in the delegation stage.

Mobile agent is one of the best application areas of proxy signature scheme, because the original signer (customer) has to delegate his/her signing capability to the mobile agent (and to the server) for it to execute any authentic operation on behalf of the original signer. [KBLK01] applied proxy signature scheme to mobile agent and introduced one-time proxy signature to guarantee one-timeness of signature generation. [OSM01] considered multi-proxy situation where plural customers delegate their signing capabilities to a mobile agent and proposed an efficient mobile agent scheme. Multi-proxy signature is also considered in

[YBX00]. But [OSM01] and [YBX00] have used weak version of proxy signature, so they cannot provide non-repudiation of the server.

1.4 Our Contribution

To provide strong undeniability, i.e., non-repudiation of the server, we construct Secure Mobile Agent (SMA) using the Strong Non-designated Proxy Signature (SNPS) [LKK01]. We provide two implementation examples of SMA. Firstly, we construct RSA-based SMA which is an extension of [KBC00] and show that it satisfies all the requirements of SNPS. Secondly, we construct Schnorr-based SMA using [LKK01,KBLK01] and show that it also satisfies all the requirements of SNPS. Moreover, we show that the Schnorr-based SNPS can be used very efficiently in multi-proxy situation providing efficiency in communication and computation.

In Section 2, we describe SNPS briefly with its security requirements. In Sections 3 and 4, we construct Schnorr-based SMA and RSA-based SMA, respectively. In Section 5, we describe multi-proxy SMA using multi-proxy signature. Finally, we conclude in Section 6.

2 Strong Non-designated Proxy Signature

[LKK01] has shown several attacks against previous proxy signature schemes [MUO96,PH97,KPW97]. There are possibilities of proxy signer's repudiation or misuse of the proxy key pair. They classified proxy signatures as strong and weak ones. Strong proxy signatures represent both original signer's and proxy signer's signatures, while weak ones represent only original signer's signature. In real situation, assuming the trustedness of original signer or proxy signer is difficult, specially in distributed environment as mobile agent. So weak versions of proxy signature cannot be used. If the proxy signature scheme is strong, it can be used without designating the proxy signer in delegation stage. We define the Strong Non-designated Proxy Signature (SNPS) as follows.

Definition 1 (Strong Non-designated Proxy Signature). *Let A be an original signer who has authentic key pair (sk_A, pk_A) and B be a proxy signer who has authentic key pair (sk_B, pk_B). Let m_w be A's warrant information for the delegation which does not specify a proxy signer. Let $\sigma_A = S(sk_A, m_w)$ be A's signature on warrant m_w using her private key sk_A. Then SNPS is constructed as the following three algorithms $(\mathcal{PKG}, \mathcal{PS}, \mathcal{PV})$.*

- *$\mathcal{PKG}$ is a proxy key issuing algorithm that takes original signer's signature σ_A and proxy signer's private key sk_B and outputs a proxy key pair (sk_P, pk_P). It is executed by the proxy signer.*

$$(sk_P, pk_P) \leftarrow \mathcal{PKG}(\sigma_A, sk_B).$$

- $\mathcal{PS}$ *is a proxy signing algorithm that takes proxy private key* sk_P *and message* m *and outputs proxy signature* σ_P. *It is executed by the proxy signer.*

$$\sigma_P \leftarrow \mathcal{PS}(sk_P, m).$$

- $\mathcal{PV}$ *is a proxy verification algorithm that takes* $(\sigma_P, m, m_w, pk_A, pk_B)$ *and outputs either accept or reject. It is executed by any verifier.*

$$\mathcal{PV}(\sigma_P, m, m_w, pk_A, pk_B) \stackrel{?}{=} accept \text{ or } reject.$$

SNPS should satisfy the following security requirements [LKK01].

R1. Verifiability: From a proxy signature a verifier can be convinced of the original signer's agreement on the signed message.
R2. Strong unforgeability: A proxy signer can create a valid proxy signature for the original signer. But the original signer and any third party cannot create a valid proxy signature with the name of proxy signer.
R3. Strong identifiability: Anyone can determine the identity of the corresponding proxy signer from a proxy signature.
R4. Strong undeniability: Once a proxy signer creates a valid proxy signature on behalf of an original signer, the proxy signer cannot repudiate his signature creation against anyone.
R5. Prevention of misuse: It should be confident that proxy key pair cannot be used for other purposes. In the case of misuse, the responsibility of proxy signer should be determined explicitly.

A proxy signature represents both the original signer's signature (by R1) and the proxy signer's signature (by R2, R3, and R4). Requirement R5 guarantees that the proxy key pair cannot be used for other purposes.

In mobile agent environment, the customer (original signer) cannot determine a proper server (proxy signer) in the delegation stage who will suggest a conforming bid. In this case mobile agent has the role of transferring customer's delegation information to possible proxy signers. To provide fairness of contract, proxy signature scheme should contain proxy signer's signature together with original signer's agreement. Therefore, SNPS is a perfect solution to construct SMA.

Because SNPS represents both the original signer's and the proxy signer's signatures, it can be considered as an efficient integration scheme of two related signatures. As stated in [MUO96] and [KPW97], partially delegated proxy signature is more efficient than that of delegation by warrant which is represented by two signatures. We will discuss the efficiency issue of proxy signatures in more detail in Section 5.

3 Schnorr-based SMA

We apply the SNPS of [LKK01] to mobile agent situation. Firstly we review Schnorr signature briefly. Let p and q be large primes with $q|p-1$. Let g be a generator of a multiplicative subgroup of Z_p^* with order q. $h()$ denotes a collision resistant cryptographic hash function. Assume that a signer A has a private key x_A and the corresponding public key $y_A = g^{x_A}$. To sign a message m, A chooses a random number $k \in_R Z_q^*$ and computes $r = g^k$, $s = x_A h(m,r) + k$. Then the tuple (m,r,s) becomes a valid signed message. The validity of signature is verified by $g^s \stackrel{?}{=} y_A^{h(m,r)} r$. Note that the verification of signature requires two modular exponentiations.

Let A be a customer who has an authentic key pair (x_A, y_A) and B be a server who has also an authentic key pair (x_B, y_B). Let ID_A and ID_B denote the identities of A and B, respectively. Let req_A be A's requirement for a purchase (any necessary information such as price range, date, delivery requirement, etc) and bid_B be B's bid information which conforms to req_A.

Preparing the agent (by the customer A):

A chooses a random number $k_A \in_R Z_q^*$ and computes $r_A = g^{k_A}$, $s_A = x_A h(req_A, r_A) + k_A$. The tuple (req_A, r_A, s_A) is A's Schnorr signature on req_A. A gives (req_A, r_A, s_A) to the mobile agent. Note that A does not specify any server in this stage. Mobile agent will migrate to servers through the network.

Executing the agent (by the server B):

B gets the mobile agent and tries to sell the product to A.

- B verifies the validity of the mobile agent by checking $g^{s_A} \stackrel{?}{=} y_A^{h(req_A, r_A)} r_A$.
- B generates a secure proxy key pair as

$$x_P = s_A + x_B, \quad y_P \equiv g^{x_P} = y_A^{h(req_A, r_A)} r_A y_B.$$

- B generates a bid information bid_B which conforms to req_A. He signs $m = (ID_A, req_A, ID_B, bid_B, r_A)$ with the proxy private key x_P to generate $\sigma_P = S(x_P, m)$ using the Schnorr signature scheme $S()$. He gives the following messages to the agent.

$$(ID_A, req_A, ID_B, bid_B, r_A, \sigma_P).$$

The mobile agent will get back to A with these messages as a receipt for her purchase.

Verifying the signature (by anyone):

When A receives $(ID_A, req_A, ID_B, bid_B, r_A, \sigma_P)$ from the mobile agent, she can verify the validity of her purchase as follows:

1) Verify the signature by $V(y_P, m, \sigma_P) \stackrel{?}{=} true$ where $y_P = y_A^{h(req_A, r_A)} r_A y_B$ and $m = (ID_A, req_A, ID_B, bid_B, r_A)$.
2) Verify the conformance of bid: $bid_B \stackrel{?}{\in} \{req_A\}$.

If the signature verification holds, it represents both the validity of signature itself and the authenticity of customer's delegation.

We show that the proposed Schnorr-based SMA satisfies all the security requirements of SNPS.

Theorem 1. *The proposed Schnorr-based SNPS is as secure as the Schnorr signature scheme.*

Proof. We consider two attack scenarios; the first case is that A tries to forge a SNPS with the name of B without B's agreement, and the second case is that B tries to forge a SNPS without A's delegation. Let $\sigma_P = (r, s)$ be a valid Schnorr-based SNPS for the message $m = (ID_A, req_A, ID_B, bid_B, r_A)$ generated by using the proxy private key x_P where $r = g^k$ for a random number $k \in_R Z_q^*$ and $s = x_P h(m, r) + k$. Note that x_P is not known to A and B in both attack scenarios.

1. Forgery by A**:** Assume that there is a SNPS breaker (oracle) which takes (m, k) and A's delegation as input and outputs a valid proxy signature (σ_P, r_A) which satisfies the verification equation. An attacker A chooses a random number k and computes $r = g^k$. She gives (m, k) and her delegation $s' = x_A h(req_A, r_A) + k_A$ to the SNPS breaker, then it will output a valid SNPS (σ_P, r_A) which satisfies the verification equation $g^s = (y_A^{h(req_A, r_A)} r_A y_B)^{h(m,r)} r$. Because of the group property of discrete logarithm problem,

$$\begin{aligned} s &= (x_A h(req_A, r_A) + k_A + x_B) h(m, r) + k \\ &= (s' + x_B) h(m, r) + k \end{aligned}$$

should hold. Then A can compute

$$x_B h(m, r) + k = s - s' h(m, r)$$

which is B's Schnorr signature on the message m. Using the SNPS breaker, A can forge B's Schnorr signature without knowing x_B.

2. Forgery by B**:** Assume that there is a SNPS breaker which takes (m, req_A, k) as input and outputs a valid proxy signature (σ_P, r_A) which satisfies the verification equation. An attacker B chooses a random number k and computes $r = g^k$. He gives (m, req_A, k) to the SNPS breaker, then it will output a valid SNPS (σ_P, r_A) which satisfies the verification equation $g^s = (y_A^{h(req_A, r_A)} r_A y_B)^{h(m,r)} r$. Because of the group property of discrete logarithm problem,

$$s = (x_A h(req_A, r_A) + k_A + x_B) h(m, r) + k$$

should hold. Then B can compute

$$x_A h(req_A, r_A) + k_A = (s - k)/h(m, r) - x_B$$

which is A's Schnorr signature on req_A. Using the SNPS breaker, B can forge A's Schnorr signature without knowing x_A.

Therefore the proposed Schnorr-based SNPS is as secure as the Schnorr signature scheme. □

From Theorem 1, the proposed Schnorr-based SMA satisfies all the security requirements of SNPS.

(i) *Verifiability*: A's agreement on req_A is included in y_P. If the proxy signature is verified to be valid, A's agreement is also verified explicitly.
(ii) *Strong unforgeability*: Anyone except the proxy signer B cannot generate a valid proxy key pair under the name of B because it contains proxy signer's private key x_B. Only the legitimate proxy signer can create a valid proxy signature.
(iii) *Strong identifiability*: Identity information of the proxy signer B is included explicitly in a valid proxy signature as a form of public key y_B. So anyone can determine the identity of the corresponding proxy signer.
(iv) *Strong undeniability*: Once the proxy signer B creates a valid proxy signature, he cannot repudiate it because the proxy key pair can be computed only by himself.
(v) *Prevention of misuse*: If the proxy signer B uses the proxy key pair for other purposes that are not specified in req_A, it is his responsibility because he is the only person who can generate it.

4 RSA-based SMA

In this Section, we propose an RSA-based SNPS scheme and apply it to construct SMA. It is an extension of [KBC00] scheme to include proxy signer's signature.

To generate RSA keys, each participant selects a modulus n which is the product of two large primes p, q and a number e, such that $1 < e < \varphi(n) = (p-1)(q-1)$ and $\gcd(e, \varphi(n)) = 1$. Let d be such that $de = 1 \bmod \varphi(n)$. Let $h()$ denote collision resistant cryptographic hash function.

Let A be a customer who has an authentic RSA key (n_A, e_A, d_A) and B be a server who has an authentic RSA key (n_B, e_B, d_B). Let ID_A and ID_B denote the identities of A and B, respectively. Let req_A be A's requirement for a purchase (any necessary information such as price range, date, delivery requirement, etc) and bid_B be B's bid information which conforms to req_A.

Preparing the agent (by the customer A):

A computes $k = h(ID_A, req_A)^{d_A} \bmod n_A$ which is her RSA signature on (ID_A, req_A). She gives (ID_A, req_A, k) to the mobile agent. Note that A does not specify any server (proxy signer) in this stage. Mobile agent will migrate to servers through the network.

Executing the agent (by the server B):

B gets the mobile agent and tries to sell the product to A.

- B verifies the validity of the mobile agent by checking

$$k^{e_A} \bmod n_A \stackrel{?}{=} h(ID_A, req_A).$$

- B generates a bid information bid_B which conforms to req_A and computes

$$x = h(ID_A, req_A, ID_B, bid_B)^{d_B} \bmod n_B$$

which is B's RSA signature on $(ID_A, req_A, ID_B, bid_B)$.
- B computes $y = h(ID_A, req_A)^x \bmod n_A$ and $z = k^x \bmod n_A$. He gives following messages to the mobile agent.

$$(ID_A, req_A, ID_B, bid_B, x, y, z).$$

The mobile agent will get back to A with these messages as a receipt for her purchase.

Verifying the signature (by anyone):

When A receives $(ID_A, req_A, ID_B, bid_B, x, y, z)$ from the mobile agent, she can verify the validity of her purchase as follows:

1) Verify B's signature: $x^{e_B} \bmod n_B \stackrel{?}{=} h(ID_A, req_A, ID_B, bid_B)$.
2) Verify the validity of y: $y \stackrel{?}{=} h(ID_A, req_A)^x \bmod n_A$.
3) Verify A's signature: $z^{e_A} \bmod n_A \stackrel{?}{=} y$.
4) Verify the conformance of bid: $bid_B \stackrel{?}{\in} \{req_A\}$.

The proxy signature is valid only when all the verifications above are passed.

We show that the proposed RSA-based SMA satisfies all the security requirements of SNPS.

Theorem 2. *The proposed RSA-based SNPS is as secure as the RSA signature scheme.*

Proof. We consider two attack scenarios; the first case is that A tries to forge a SNPS with the name of B without B's agreement, and the second case is that B tries to forge a SNPS without A's delegation. Obviously the first attack cannot happen because a valid SNPS contains x which is B's signature for $(ID_A, req_A, ID_B, bid_B)$. Consider the second attack scenario where B tries to forge a SNPS without k.

Assume that there is a SNPS breaker (oracle) which takes $(ID_A, req_A, ID_B, bid_B, x)$ as input and outputs (y, z) which satisfy the verification equations. B prepares a warrant req_A and a conforming bid bid_B and generates his signature $x = h(ID_A, req_A, ID_B, bid_B)^{d_B} \bmod n_B$. He gives $(ID_A, req_A, ID_B, bid_B, x)$ to the SNPS breaker, then it will provide a valid (y, z). $y = h(ID_A, req_A)^x \bmod n_A$ can be verified from the known values (ID_A, req_A, x). To satisfy the third verification equation, the following equation should hold.

$$z = y^{d_A} \bmod n_A = h(ID_A, req_A)^{x d_A} \bmod n_A.$$

Then B can compute

$$z^{1/x} \bmod n_A = h(ID_A, req_A)^{d_A} \bmod n_A = k$$

which is A's RSA signature on message (ID_A, req_A). Using the SNPS breaker, B can forge A's RSA signature without knowing d_A. Therefore the proposed RSA-based SNPS is as secure as the RSA signature scheme. □

From Theorem 2, the proposed RSA-based SMA satisfies all the security requirements of SNPS.

(i) *Verifiability*: Original signer's agreement on the purchase can be verified by the third verification equation.
(ii) *Strong unforgeability*: Only the proxy signer B can generate a valid signature x satisfying the first verification equation.
(iii) *Strong identifiability*: Anyone can determine the identity of the corresponding proxy signer by the first verification equation.
(iv) *Strong undeniability*: Once B creates a valid proxy signature which passes all the verification equations, he cannot repudiate it later against anyone because a valid proxy signature can be generated only by himself.
(v) *Prevention of misuse*: k is A's signature on (ID_A, req_A) and it cannot be used for other purposes which are not stated in req_A. The proxy signature scheme is executed using B's signature x, so any possible misuse of k is B's responsibility.

5 Multi-Proxy Mobile Agent

In this Section, we propose an efficient mobile agent scheme when plural customers delegate their signing capabilities to a mobile agent. For example, we consider a situation that a mobile agent is ordered to book flight tickets for plural customers. Using the Schnorr-based SMA scheme where plural customers share the common system parameters p, q, and g, we can build an efficient mobile agent.

[OSM01] considered a similar application, but their scheme is based on the proxy signature of [MUO96] and customer's requirements are not used. So customers delegate their full signing capabilities to unspecified proxy signers and a server can sign any message on behalf of customers. [YBX00] also proposed proxy multi-signature scheme based on [MUO96]. We apply the strong non-designated proxy signature [LKK01] to multi-proxy mobile agent.

5.1 Multi-Proxy Mobile Agent Scheme

Let A_i $(i = 1, ..., n)$ denote plural customers who have certified key pairs (x_i, y_i) and requirements req_i. They try to delegate their signing capabilities to unspecified servers through the mobile agent. Let B be a server who has certified key pair (x_B, y_B) and is willing to sell flight tickets to customers. He has to create a proxy signature on behalf of $\{A_1, ..., A_n\}$ under requirements $\{req_1, ..., req_n\}$.

Preparing the agent (by plural customers A_i):

Plural customers A_i $(i = 1, ..., n)$ choose random numbers $k_i \in_R Z_q^*$ and compute $r_i = g^{k_i}$, $s_i = x_i h(req_i, r_i) + k_i$. The tuple (req_i, r_i, s_i) is A_i's Schnorr signature on req_i. A_i gives (req_i, r_i, s_i) to the mobile agent. Mobile agent will migrate to servers through the network with this information.

Executing the agent (by the server B):
The server B gets the mobile agent and tries to sell the product to customers $\{A_1, ..., A_n\}$.

- B verifies the validity of the delegation information by checking $g^{s_i} \stackrel{?}{=} y_i^{h(req_i, r_i)} r_i$ for $i = 1, ..., n$.
- If this tests have passed, B generates a secure proxy key pair as
$$x_P = s_1 + \cdots + s_n + x_B, \quad y_P = g^{x_P}.$$
- B generates his bid bid_B which conforms to all req_i $(i = 1, ...n)$. He signs on $m = (req_1, \cdots, req_n, bid_B)$ with the proxy private key x_P to generate $\sigma_P = S(x_P, m)$ using the Schnorr signature scheme $S()$. The tuple
$$(bid_B, \sigma_P, req_1, r_1, y_1, ..., req_n, r_n, y_n, y_B)$$
is a valid proxy signature and represents valid flight tickets for $\{A_1, ..., A_n\}$.

Verifying the signature (by anyone):
When plural customers receives the tuple from the mobile agent, they can verify the validity of their tickets as follows:

1) Verify the signature by $V(y_P, m, \sigma_P) \stackrel{?}{=} true$ where
$$y_P = y_1^{h(req_1, r_1)} r_1 \cdots y_n^{h(req_n, r_n)} r_n y_B, \quad m = (req_1, \cdots, req_n, bid_B).$$
2) Check whether bid_B confirms to $\{req_1, \cdots, req_n\}$.

5.2 Comparison with multiple signatures

As stated in [MUO96], proxy signature schemes of partial delegation are more efficient than those of delegation by warrant. Consider a traditional approach of multiple independent signatures that plural customers A_i publish their signatures (req_i, r_i, s_i) and the server B just signs on bid_B with his certified key pair (x_B, y_B). The proposed multi-proxy signature scheme is more efficient than the traditional approach of multiple independent signatures in the following sense.

- A valid signature can be created by the proxy signer himself without any interaction with original signers, while traditional scheme requires n communications with original signers.
- Message size is reduced by $n|q|$ because $(s_1, ..., s_n)$ are not necessary in proposed scheme.

– Verification of signature is more efficient because proposed scheme requires only $n+2$ exponentiations (one signature verification and n exponentiations) while traditional scheme requires $2(n+1)$ exponentiation for $n+1$ signature verifications. Moreover, simultaneous multiple exponentiation with distinct bases can be computed very efficiently [MOV97].

Proposed scheme can be used in a very flexible way because the server can choose different combinations of delegations by himself among n delegations depending on the property of his bid. If he has only $l < n$ flight tickets to sell, he can sell them only to l customers of his choice.

6 Conclusion

We have pointed out the necessity of using SNPS to construct SMA. To provide the fairness of a purchase, the proxy signature should represent both customer's and server's signatures. The validity of bid information is verified by comparing it with customer's requirement. From the observation that the features of undetachable signatures are very similar to those of proxy signatures, we extended [KBC00] to provide an RSA-based SNPS scheme and applied it to mobile agent. Very similarly, we provided a Schnorr-based SMA scheme. In multi-proxy situation, Schnorr-based SNPS can be used in very efficient manner because plural customers can share the same system parameters.

Proxy signatures are very useful tools when one needs to delegate his/her signing capability to other party. But in distributed environment like the Internet, it is very difficult to assume the trustedness of original signer, proxy signer, and the proxy key issuing protocol between them. Because the delegation of signing capability to others can be risky, proxy signature schemes should be designed carefully such that proxy signer's responsibility is determined explicitly and any possibility of misuse is prevented. But if we can delegate signing capabilities safely using strong proxy signature schemes, many cryptographic applications in distributed environment such as electronic commerce and mobile agent can be implemented in more efficient and flexible way.

References

[FGS96] W. Farmer, J. Gutmann and V. Swarup, "Security for Mobile Agents: Authentication and State Appraisal", *Proc. of the European Symposium on Research in Computer Security (ESORICS)*, LNCS 1146, Springer-Verlag, pp.118-130, 1996.

[KBC00] P. Kotzanikolaous, M. Burmester and V. Chrissikopoulos, "Secure Transactions with Mobile Agents in Hostile Environments", *ACISP 2000*, LNCS 1841, Springer-Verlag, pp.289-297, 2000.

[KBLK01] H. Kim, J. Baek, B. Lee, and K. Kim, "Secret Computation with Secrets for Mobile Agent using One-time Proxy Signature", *Proc. of SCIS2001*, pages 845–850, 2001.

[KKC99] P. Kotzanikolaous, G. Katsirelos and V. Chrissikopoulos, "Mobile Agents for Secure Electronic Transactions", *Recent Advances in Signal Processing and Communications*, World Scientific and Engineering Society Press, pp.363-368, 1999.

[KPW97] S. Kim, S. Park, and D. Won, "Proxy Signatures, Revisited", *Proc. of ICICS'97,* Y. Han *et al*(Eds.), LNCS 1334, Springer-Verlag, pages 223-232, 1997.

[LKK01] B. Lee, H. Kim and K. Kim, "Strong Proxy Signature and its Applications", *Proc. of SCIS2001*, pages 603–608, 2001.

[LM99] S. Loureio and R. Molva, "Privacy for Mobile Code", *Proc. of Distributed Object Security Workshop OOPSLA'99*, 6 pages, 1999.

[MOV97] A. Menezes, P. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography,* pages 617 - 618, CRC Press, 1997.

[MS97] J. Merwe and S.H. Solms, "Electronic Commerce with Secure Intelligent Trade Agents", *Proc. of ICICS'97*, Y. Han *et al*(Eds.), LNCS 1334, Springer-Verlag, pp.452-462, 1997.

[MUO96] M. Mambo, K. Usuda, and E. Okamoto, "Proxy Signatures: Delegation of the Power to Sign Messages", *IEICE Trans. on Fundamentals*, Vol. E79-A, No. 9, Sep., pages 1338–1353, 1996.

[OSM01] R. Otomura, M. Soshi, and A. Miyaji, "On Digital Signature Schemes for Mobile Agents", *Proc. of SCIS2001*, pages 851–855, 2001.

[PH97] H. Petersen and P. Horster, "Self-certified Keys – Concepts and Applications", *Proc. Communications and Multimedia Security'97*, pages 102 - 116, Chapman & Hall, 1997.

[PS96] D. Pointcheval and J. Stern, "Security Proofs for Signatures", *Advances in Cryptology: Eurocrypt'96*, pages 387 - 398, Springer, 1996.

[ST97] T. Sander and C. F. Tschudin, "Protecting Mobile Agents Against Malicious Hosts", *Mobile Agent Security*, LNCS 1419, Springer-Verlag, pp.44-60, 1997.

[YBX00] L. Yi, G. Bai and G. Xiao, "Proxy Multi-signature Scheme: A New Type of Proxy Signature Scheme", *Electronics Letters*, Vol.36, No.6, pages 527-528, 16th March 2000.