Homomorphic Signature for Identity Authentication in Cloud Computing

Zhiwei Wang, Guozi Sun, and Danwei Chen

1.College of Computer, Nanjing University of Posts and Telecommunications,Nanjing, Jiangsu 210003, China
2.Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing, Jiangsu 210003, China
3.Key Lab of Broadband Wireless Communication and Sensor Network Technology (Nanjing University of Posts and Telecommunications),Ministry of Education Jiangsu Province, Nanjing, Jiangsu 210003, China
zhwwang@njupt.edu.cn

Abstract. In this paper, we introduce a new kind of homomorphic signature, which is suitable for identity authentication in cloud computing. User firstly gives his full signature (involves all his identity attributes) to the identity authentication server. During the valid period of his full signature, if the user wants to require a cloud service on a special identity (only involves part of identity attributes), he only needs to secretly send a $\{0,1\}^n$ vector to the identity authentication server. The identity authentication server who doesn't know the secret key can compute the partial signature on the special identity, and then signs it to the cloud server. We give a formal secure definition of this homomorphic signature, and construct a scheme from GHR signature. We prove that our scheme is secure under strong RSA assumption.

Key words: homomorphic signature; identity authentication; cloud computing; GHR signature; strong RSA assumption

1 Introduction

Homomorphic signature [18] is a cryptographic primitive where, given a set of signature $\sigma_1, \cdots, \sigma_n$ on messages $m_1, \cdots, m_n$ in message space $\mathcal{M}$, a function $f : \mathcal{M}^n \to \mathcal{M}$ from a set of "admissible" functions $\mathcal{F}$, anyone can compute a signature on the pair $(f, f(m_1, \cdots, m_n)) \in \mathcal{F} \times \mathcal{M}$. If the signature is valid, then it means that the claimed value equals $f(m_1, \cdots, m_n)$. If the system is secure, then any malicious adversary cannot compute a valid signature on (f, m^*) for any $m^* \neq f(m_1, \cdots, m_n)$.

Homomorphic signature was firstly proposed by Johnson et al. [18] In the past few years, there are about three classes research have touch on this area: **quoting/redacting signature**, **arithmetic signature**, **transitive signature**. Quoting/redacting signature [2–6] is that given Alice's signature on some message m, any one can derive Alice's signature on a substring of m. Quoting/redacting signature is specially applied to signed text message and signed

images. Arithmetic signature [7–11] is motivated by the application of secure network coding. In transitive signature[12–16], given Alice's signature on edges in a graph G, any one can derive Alice's signature on a pair of vertices u, v, if there exists a path from u to v in G.

In 2009, Gentry proposed a fully homomorphic encryption scheme to make it possible to analyze data without decrypting it [1]. As we all know, this property is very suitable for the encrypted date searching in cloud computing. Homomorphic signature also can be used to cloud computing, especially in identity authentication. **User requires cloud computing service through an identity authentication server. A user may have many identities, and require cloud service only by one identity each time. The users usually are the mobile phones, PDAs et al., which have limited computing power.**

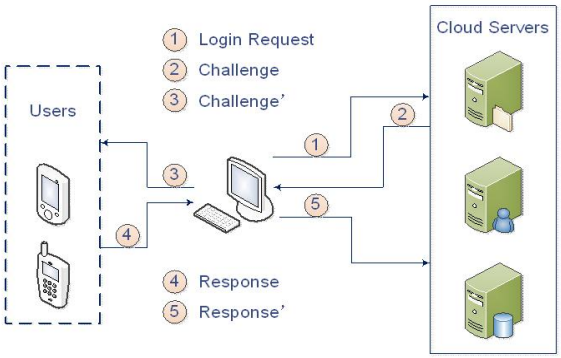

Fig. 1. Identity authentication in cloud computing

We assume that every user has a set of identity attributes $\mathbf{v} = \{v_1, \cdots, v_n\}$, and one identity ID is a subset of $\mathbf{v}$. Firstly, every user pre-computes his full signature $(\sigma_1, \cdots, \sigma_n)$ on the corresponding attributes $(v_1, \cdots, v_n)$, and sends them to the identity authentication server. Each full signature has a period of validity. During the valid period of his full signature, if the user wants to require a cloud service on a special identity ID, which is a subset of $\mathbf{v}$, he only needs to secretly send a vector $(c_1, \cdots, c_n)$ to the identity authentication server, where if $v_i \in ID$, then $c_i = 1$, otherwise $c_i = 0$. The identity authentication server can compute a partial signature $\hat{\sigma}$ on the ID by using the homomorphic signature algorithm, and then it sends $(\hat{\sigma}, ID)$ to cloud server for identification.

There are two advantages for the above application of homomorphic signature. **Firstly, the identity authentication sever doesn't know any user's secret key, which can protect user's privacy. Secondly, during the identity authentication process, the user doesn't need to do any heavy computation, and only secretly send a $\{0,1\}^n$ vector to the identifica-**

tion sever. The pre-computation can be done with the help of other servers. Thus, it is very suitable that the users are devices with limited resource.

Our goal is to design a homomorphic signature scheme suitable for the above application. Our contributions can be listed as follows: 1) We give a security definition of the homomorphic signature suitable for identity authentication in cloud computing; 2) We construct such a homomorphic signature scheme based on the GHR signature [17] ; 3) We provide a security proof to our scheme without random oracles, using strong RSA assumption.

2 Preliminaries

2.1 Useful facts from number theory

Shamir showed that it is easy to compute $y^{1/a} \pmod N$ given $y^{b/a}$.

Lemma 2.1.[19] *Given $x, y \in \mathbb{Z}_N$ together with $a, b \in \mathbb{Z}$ such that $x^a = y^b$ and $gcd(a, b) = 1$, there is an efficient algorithm for computing $z \in \mathbb{Z}_N$ such that $z^a = y$.*

The algorithm is to use Euclid's algorithm to compute $f, g \in \mathbb{Z}$ such that $af + bg = 1$, and then compute $z = x^g y^f$.

Lemma 2.2. *The probability that two arbitrary integers m and n are relatively prime is $6/\pi^2$.*

Proof. Let m, n be the two arbitrary integers. Let p_i denote all the prime numbers $p_1 = 2$, $p_2 = 3$, $p_3 = 5$ etc. The probability that a number has a factor p_i is $1/p_i$. Thus, the probability that m and n do not both contain p_i at the same time is estimated as $1 - 1/(p_i)^2$.

Therefore, the probability that m and n are relatively prime, or do not both contain any of the p_i at the same time is product $((1 - 1/(p_i)^2), i = 1, ..., infinite)$. It is easy to prove that the reciprocal of this product equal to $\sum 1/k^2$, $(k = 1, ..., infinite)$, which is equal to $\pi^2/6$, thus, the probability equal to $6/\pi^2$.

2.2 RSA assumptions

Let $N = p \times q$ be a composite modulus, where p and q are two large prime numbers. We refer to the group $\mathbb{Z}_N^*$ as a RSA group. The two RSA computational problem can be defined as follows:

- **RSA Problem**: Given a tuple (N, g, e), where $g \xrightarrow{R} \mathbb{Z}_N^*$ and e is a randomly chosen integer less than and relatively prime to $\varphi(N) = (p - 1)(q - 1)$, to output a solution $g^{1/e} \in \mathbb{Z}_N^*$.

– **Strong RSA Problem**:Given a tuple (N, g), where $g \xrightarrow{R} \mathbb{Z}_N^*$, to output a solution $(z, e) \in \mathbb{Z}_N^* \times \mathbb{Z}$ such that $z = g^{1/e}$.

If $\mathcal{A}$ is an algorithm which can solve the RSA problem, we define $ADV_{\mathcal{A}}$ to be the probability that $\mathcal{A}$ outputs a solution. We say the RSA assumption holds if for all polynomial-time algorithm $\mathcal{A}$, $RSA-ADV_{\mathcal{A}}$ is negligible. We can define the strong RSA assumption analogously.

2.3 GHR signature

In this section, we will review the signature proposed by Gennaro, Halevi, and Rabin [17]. Let $H : \{0,1\}^{\lambda} \to \mathbb{Z}$ be a hash function. The GHR signature can be depicted as follows:

Setup(1^{λ}) This algorithm selects two large safe $\lambda/2$-bit primes p and q, and computes $N = pq$. Let $\mathbb{G}$ denote the subgroup of squares in $\mathbb{Z}_N^*$. Then, chooses a random $g \xleftarrow{R} \mathbb{G}$. The public key is $pk = (N, g)$ and the secret key is $sk = (p, q)$.
Sign(sk, m) This algorithm outputs signature $\sigma = g^{1/H(m)}$.
Verify(pk, m, σ) This algorithm outputs 1 if $\sigma^{H(m)} = g$, otherwise outputs 0.

GHR signature is the first "hash-and-sign" signature proved secure without random oracles, using strong RSA assumption. Gennaro et al. [17] proved the following theorem:
Theorem 2.1. *If the strong RSA assumption holds and H is a collision-resistant hash function that outputs odd primes less than $2^{2\lambda-2}$, then GHR signature is unforgeable.*

3 Homomorphic signature for identity authentication in cloud computing

We denote the homomorphic signature for identity authentication in cloud computing as Σ **signature**. In a Σ signature scheme, each user can pre-compute his full signature $\bar{\sigma} = (\sigma_1, \cdots, \sigma_n)$ on all identity attributes $(v_1, \cdots, v_n)$. Every element of full signature has the same validity period. During the term of validity, according to the vector from user, identity authentication server can generate the partial signature $\hat{\sigma}$ on the special identity, which is a subset of $\mathbf{v} = \{v_1, \cdots, v_n\}$. Formally, we have the following:

Definition 3.1. A Σ signature scheme is a tuple of probabilistic, polynomial-time algorithms (**Setup, Sign, Verify, Eval**) as follows:

– **Setup**$(1^{\lambda}, n)$: Takes as input a security parameter λ and a maximum identity attributes set size n, outputs a public key pk and a secret key sk.

- **Sign**(sk, v_i): Takes a secret key sk, an identity attribute v_i, outputs a signature $\tilde{\sigma}_i = (\sigma_i, s_i, t)$, where s_i denotes a random number and t denotes a period of validity.
- **Verify**$(pk, ID, \tilde{\sigma})$: Takes a public key pk, an identity $ID \subset \{v_1, \cdots, v_n\}$, and a signature $\tilde{\sigma} = (\sigma, s, t)$, outputs either 0(reject) or 1(accept).
- **Eval**$(pk, (c_1, \cdots, c_n), \bar{\sigma})$: Takes a public key pk, a vector $(c_1, \cdots, c_n)$ corresponding to an identity $ID \subset \mathbf{v}$ (if $v_i \in ID$ then $c_i = 1$, otherwise $c_i = 0$), and a tuple of signatures (full signature) $\bar{\tilde{\sigma}} = (\tilde{\sigma}_1, \cdots, \tilde{\sigma}_n)$, outputs a partial signature $\hat{\tilde{\sigma}}$ on ID.

Let e_i be a unit vector e_i with a 1 in the ith place and 0 elsewhere. Informally, the correctness of our scheme are that (1) a signature generated by **Sign** on v_i verifies for e_i, and (2) a signature output by **Eval** on ID and input signatures produced by **Sign** verifies for the identity $ID \subset \{v_1, \cdots, v_n\}$. Formally, we require that for each (pk, sk) output by **Setup**$(1^\lambda, n)$, we have:

1. For all $v_i, i \in \{1, \cdots, n\}$, if $\tilde{\sigma} \leftarrow$**Sign**$(sk, v_i)$, then with overwhelming probability **Verify**$(pk, e_i, \tilde{\sigma})$=1.
2. For all vectors $(c_1, \cdots, c_n)$ corresponding to $ID \subset \{v_1, \cdots, v_n\}$, if $\tilde{\sigma}_i \leftarrow$**Sign**$(sk, v_i)$ for $i = 1, \cdots, n$, then with overwhelming probability

$$\textbf{Verify}(pk, ID, \textbf{Eval}(pk, (c_1, \cdots, c_n), \bar{\tilde{\sigma}})) = 1$$

3.1 Security

The goal of an adversary attacking a Σ signature scheme is forge a valid signature on an identity $ID^* \not\subset \{v_1, \cdots, v_n\}$ or out of the validity period of full signature. In our security model, the adversary is allowed to query one ID at a time and proceed adaptively.

Definition 3.2. A Σ signature scheme $\mathcal{S}$ =(**Setup, Sign, Verify, Eval**) is unforgeable against adaptive chosen message attack if for all q the advantage of any probabilistic, polynomial-time adversary $\mathcal{A}$ in the following game is negligible in the security parameter n:

Setup: The challenger runs $Setup(1^\lambda, n)$ to obtain (pk, sk) and sends pk to $\mathcal{A}$.
Queries: $\mathcal{A}$ queries on an identity $ID \subset \{v_1, \cdots, v_n\}$. The challenger then gives to $\mathcal{A}$ the signature $\tilde{\sigma} = (\sigma, s, t) \leftarrow Sign(sk, ID)$. This interaction is repeated a polynomial number of times, subject to the restriction that at most q identities can be queried.
Output: $\mathcal{A}$ outputs the forgery including an identity ID^* and a signature $\tilde{\sigma}^* = (\sigma^*, s^*, t^*)$.

We say that the adversary wins if $Verify(pk, ID^*, \tilde{\sigma}^*) = 1$ and $ID^* \not\subset \{v_1, \cdots, v_n\}$ or $t^* \neq t$. The advantage $\Sigma - Adv$ of $\mathcal{A}$ is the probability that $\mathcal{A}$ wins the above game. If $\Sigma - Adv$ is negligible, then the Σ signature scheme is secure.

4 A Σ signature scheme

We define a Σ **signature** scheme $\mathcal{S}$ from GHR signature scheme as follows:

S.Setup(1^λ,n): On input a security parameter λ and a maximum identity attributes set size n, do the following:

1. Select two large safe $\lambda/2$-bit primes p and q, and compute $N = pq$. Let $\mathbb{G}$ denote the subgroup of squares in $\mathbb{Z}_N^*$.
2. Choose random elements $h_1, \cdots, h_n, u \xleftarrow{R} \mathbb{G}$.
3. Choose $H : \{0,1\}^\lambda \to \mathbb{Z}$ as a hash function, and publish it.
4. Output the public key $pk = (N, \{h_i\}_{i=1}^n, u)$ and secret key $sk = (p, q)$.

The set of identity attributes $\{v_1, \cdots, v_n\}$ should be published, where each $v_i \in \mathbb{Z}_{|\mathbb{G}|}$ for $i = 1, \cdots, n$.

S.Sign(sk, v_i): On input a secret key sk and an identity attribute v_i, do the following:

1. Let `time` denote the period of valid, and compute $t = H(\texttt{time})$.
2. Choose $s_i \xleftarrow{R} \mathbb{Z}_{|\mathbb{G}|}$.
3. Compute $\sigma_i = (h_i^{v_i} \cdot u^{s_i})^{1/t} \pmod N$.
4. Output $\tilde{\sigma_i} = (\sigma_i, s_i, t)$.

S.Verify($pk, ID, \tilde{\sigma}$): On input a public key pk, an identity $ID \subset \{v_1, \cdots, v_n\}$, and a signature $\tilde{\sigma} = (\sigma, s, t)$, do the following:

1. Obtain the current time stamp $T \in \mathbb{G}$.
2. Compute $x \leftarrow \prod_{i=1}^n (h_i{}^{v_i})^{\tau_i} \cdot u^s \pmod N$, where $\tau_i = 1$ if $v_i \in ID$ otherwise $\tau_i = T$.
3. If $\sigma^t = x \pmod N$, output 1; otherwise output 0.

S.Eval($pk, (c_1, \cdots, c_n), \bar{\tilde{\sigma}}$): On input a public key pk, a $\{0,1\}^n$ vector $(c_1, \cdots, c_n)$, and a tuple of signatures (full signature) $\bar{\tilde{\sigma}} = (\tilde{\sigma_1}, \cdots, \tilde{\sigma_n})$, do the following:

1. Obtain the current time stamp $T \in \mathbb{G}$.
2. Compute a new vector $(\tau_1, \cdots, \tau_n)$, where $\tau_i = c_i$ if $c_i = 1$, otherwise $\tau_i = T$.
3. Compute $\sigma' \leftarrow \prod_{i=1}^n (\sigma_i)^{\tau_i}$, $s' \leftarrow \sum_{i=1}^n \tau_i s_i$.
4. Output $\tilde{\sigma'} = (\sigma', s', t)$.

Note. In the full signature $\bar{\sigma}$, all signatures $\tilde{\sigma_i}$ for $i = 1, \cdots, n$ have the same t, since they are all computed at the same time. Adding T is to against the replay attack.

Lemma 4.1. *The Σ signature scheme in section 4 satisfies the correctness properties of Definition 3.1.*

Proof. Suppose $(\sigma_i, s_i, t) \leftarrow Sign(sk, v_i)$. Note that the identity attribute v_i corresponds to the unit vector e_i with a 1 in the ith place and 0 elsewhere. First, since $x \leftarrow \prod_{i=1}^n (h_i{}^{v_i})^{c_i} \cdot u^s \pmod N$ implies $x = h_i^{v_i} u^{s_i}$ in this case, then **Verify**$(pk, e_i, \tilde{\sigma_i})$=1. Next, suppose $\bar{\tilde{\sigma}} = (\tilde{\sigma_1}, \cdots, \tilde{\sigma_n})$ is a vector of signatures (full signature) with $\tilde{\sigma_i} \leftarrow Sign(sk, v_i)$ for $i = 1, \cdots, n$, and let $\tilde{\sigma'} \leftarrow$

$Eval(pk, (c_1, \cdots, c_n), \bar{\tilde{\sigma}})$. Since $(\tau_1, \cdots, \tau_n)$ be computed from $(c_1, \cdots, c_n)$, we have

$$\sigma' = \prod_{i=1}^{n} (\sigma_i)^{\tau_i} = (\prod_{i=1}^{n} (h_i)^{v_i \cdot \tau_i} \cdot u^{\sum_{i=1}^{n} \tau_i s_i})^{1/t} \pmod{N}.$$

Since $x = \prod_{i=1}^{n} ({h_i}^{v_i})^{\tau_i} \cdot u^{s'} \pmod{N}$ in this case, then $\mathbf{Verify}(pk, ID, \tilde{\sigma'}) = 1$.

5 Security analysis

In Definition 3.2, an adversary can break a Σ signature scheme by forging a valid signature $(ID^*, \tilde{\sigma}^* = (\sigma^*, s^*, t^*))$. This can be done in two ways:

1. *Type 1:* In the final forged signature $\tilde{\sigma}^* = (\sigma^*, s^*, t^*)$, t^* is *not* equal to t that is used in the former responses by the challenger. This forgery means that the adversary can forge a signature on ID^* outside of the valid period t.
2. *Type 2:* In the final forged signature $\tilde{\sigma}^* = (\sigma^*, s^*, t^*)$, t^* is equal to t that is used in the former responses by the challenger, but $ID^* \notin \{v_1, \cdots, v_n\}$

5.1 Type 1 forgery

Theorem 5.1. *If strong RSA assumption holds and H is a collision-resistant hash function that outputs odd primes less than $2^{2\lambda-2}$, then our Σ signature scheme is unforgeable against Type 1 forgery. Specially, let $\mathcal{A}$ be an adversary that attacks our Σ signature scheme and outputs a Type 1 forgery. Then there exists a challenger $\mathcal{C}$ that solves the strong RSA problem and a negligible value of ϵ such that*

$$\Sigma - Adv[\mathcal{A}] \leq \frac{\pi^2}{6} StrongRSA - Adv[\mathcal{C}] + \epsilon.$$

Proof. Our algorithm $\mathcal{C}$ will play the role of the attacker to strong RSA assumption and that of the challenger in the unforgeability game for our Σ signature scheme. Given a challenge (N, g), $\mathcal{C}$'s goal is to output a solution $(z, e) \in \mathbb{Z}_N^* \times \mathbb{Z}$ such that $z = g^{1/e}$. The algorithm works as follows:

Setup: 1. Let $\mathbb{G}$ be the subgroup of squares in $\mathbb{Z}_N^*$, and let χ be a $\mathbb{G}-$uniform distribution on $\mathbb{Z}_{|\mathbb{G}|}$.
2. For $i = 1, \cdots, n$, choose $\alpha_i \xleftarrow{R} \chi$ and set $h_i = g^{\alpha_i}$. Let $\bar{\alpha} = (\alpha_1, \cdots, \alpha_n)$.
3. Choose $\delta \xleftarrow{R} \chi$ and set $u = g^{\delta}$.
4. Send $\mathcal{A}$ the public key $pk = (N, \{h_1, \cdots, h_n\}, u)$.

Queries: When $\mathcal{A}$ makes a query for an identity $ID \subset \{v_1, \cdots, v_n\}$, do the following:
1. Let (z, t), where $z = g^{1/t}$, be the strong RSA problem's solution obtained from the challenger of strong RSA problem.

2. Choose $s \xleftarrow{R} \mathbb{Z}_{|\mathbb{G}|}$.
3. Obtain the current time stamp $T \in \mathbb{G}$
4. A n-dimensional vector $\bar{\beta} = (\beta_1, \cdots, \beta_n)$ can be derived from ID, where $\beta_i = v_i$ if $v_i \in ID$ otherwise $\beta_i = T \cdot v_i$.
5. Compute $\sigma = z^{<\bar{\alpha}, \bar{\beta}> + \delta s}$, and send the signature $\tilde{\sigma} = (\sigma, s, t)$ to $\mathcal{A}$.

Forgery: When $\mathcal{A}$ outputs a forgery (σ^*, s^*, t^*) on ID^*, $\mathcal{C}$ can also output a solution to strong RSA problem by using Lemma 2.1.

We analyze the simulation using a series of games. Let W_i be the event that $\mathcal{A}$ wins the unforgeability game in Game i.

Game 0. This is the real unforgeability game, where $\mathcal{A}$ interacts with a challenger $\mathcal{C}$ for our our Σ signature scheme, and therefore $Pr[W_0] = \Sigma - Adv[\mathcal{A}]$.

Game 1. This is the same as Game 0, except we choose h_i, u as described above instead of as random elements of $\mathbb{G}$. Since χ is a $\mathbb{G}$-uniform distribution, the values of h_i, u are statistically close to uniform in $\mathbb{G}$. It follows that $Pr[W_1] \geq Pr[W_0] - \epsilon$ for some negligible ϵ.

Game 2. To compute a signature on ID, we compute the σ as $\sigma = z^{<\bar{\alpha}, \bar{\beta}> + \delta s}$. It follows that

$$\sigma = (g^{1/t})^{<\bar{\alpha}, \bar{\beta}> + \delta s} = (\prod_{i=1}^{n} (h_i)^{\tau_i \cdot v_i} \cdot u^s)^{1/t},$$

where $\tau_i = 1$ if $v_i \in ID$ otherwise $\tau_i = T$. Obviously, this change is only conceptual. Thus $Pr[W_2] = Pr[W_1]$.

Game 3. This is the same as Game 2, except that $\mathcal{A}$ outputs a forgery $(\tilde{\sigma}^*, s^*, t^*)$ on ID^*, where $t^* \neq t$. If the forgery is valid, then

$$\sigma^* = (g^{1/t^*})^{<\bar{\alpha}, \bar{\beta}^*> + \delta s^*}.$$

If $GCD(<\bar{\alpha}, \bar{\beta}^*> + \delta s^*, t^*) = 1$, then it is easy to compute g^{1/t^*} by Lemma 2.1. Then, $\mathcal{C}$ can output a solution for the strong RSA problem. Since $t^* \neq t$, the values of t^* and $<\bar{\alpha}, \bar{\beta}^*> + \delta s^*$ are both uniform in $\mathbb{Z}_{|\mathbb{G}|}$. Thus, by Lemma 2.2, it follows that $Pr[W_3] \geq \frac{6}{\pi^2} Pr[W_2]$.

Obviously, W_3 be the event that $\mathcal{C}$ outputs a solution to the strong RSA problem, so $Pr[W_3] = StrongRSA - Adv[\mathcal{C}]$.

5.2 Type 2 forgery

We now consider the Type 2 forgery, in which the adversary outputs a signature on ID^* in the valid period of user's full signature ($t^* = t$), but ID^* is not a subset of $\{v_1, \cdots, v_n\}$.

Theorem 5.2. *If strong RSA assumption holds and H is a collision-resistant hash function that outputs odd primes less than $2^{2\lambda-2}$, then our Σ signature scheme is unforgeable against Type 2 forgery. Specially, let $\mathcal{A}$ be an adversary that attacks our Σ signature scheme and outputs a Type 2 forgery. Then there*

exists a challenger $\mathcal{C}$ that solves the strong RSA problem and a negligible value of ϵ such that

$$\Sigma - Adv[\mathcal{A}] \leq StrongRSA - Adv[\mathcal{C}] + \epsilon.$$

Proof. Our algorithm $\mathcal{C}$ will play the role of the attacker to strong RSA assumption and that of the challenger in the unforgeability game for our Σ signature scheme. Given a challenge (N, g), $\mathcal{C}$'s goal is to output a solution $(z, e) \in \mathbb{Z}_N^* \times \mathbb{Z}$ such that $z = g^{1/e}$. The algorithm works as follows:

Setup: 1. Let $\mathbb{G}$ be the subgroup of squares in $\mathbb{Z}_N^*$, and let χ be a $\mathbb{G}-$uniform distribution on $\mathbb{Z}_{|\mathbb{G}|}$.
2. Choose a generator $x \in \mathbb{G}$.
3. For $i = 1, \cdots, n$, choose $\alpha_i, \gamma_i \xleftarrow{R} \chi$ and set $h_i = x^{\alpha_i} g^{-\gamma_i}$. Let $\bar{\alpha} = (\alpha_1, \cdots, \alpha_n), \bar{\gamma} = (\gamma_1, \cdots, \gamma_n)$.
4. Choose $\delta \xleftarrow{R} \chi$ and set $u = x^{\delta} g$.
5. Send $\mathcal{A}$ the public key $pk = (N, \{h_1, \cdots, h_n\}, u)$.

Queries: When $\mathcal{A}$ makes a query for an identity $ID \subset \{v_1, \cdots, v_n\}$, do the following:
1. Given (N, x), Let (y, t), where $y = x^{1/t}$, be the strong RSA problem's solution obtained from the challenger of strong RSA problem.
2. obtain the current time stamp $T \in \mathbb{G}$.
3. A n-dimensional vector $\bar{\beta} = (\beta_1, \cdots, \beta_n)$ can be derived from ID, where $\beta_i = v_i$ if $v_i \in ID$ otherwise $\beta_i = T \cdot v_i$.
4. Set $s =< \bar{\gamma}, \bar{\beta} >$.
5. Compute $\sigma = y^{<\bar{\alpha}, \bar{\beta}> + \delta s}$, and send the signature $\tilde{\sigma} = (\sigma, s, t)$ to $\mathcal{A}$.

Forgery: When $\mathcal{A}$ outputs a forgery (σ^*, s^*, t^*) on $ID^* \not\subset \{v_1, \cdots, v_n\}$ with $t^* = t$, $\mathcal{C}$ can also output a solution (z, e) to strong RSA problem by using Lemma 2.1.

We analyze the simulation using a series of games. Let W_i be the event that $\mathcal{A}$ wins the unforgeability game in Game i.

Game 0. This is the real unforgeability game, where $\mathcal{A}$ interacts with a challenger $\mathcal{C}$ for our our Σ signature scheme, and therefore $Pr[W_0] = \Sigma - Adv[\mathcal{A}]$.

Game 1. This is the same as Game 0, except we choose h_i, u as described above instead of as random elements of $\mathbb{G}$. Since χ is a $\mathbb{G}$-uniform distribution, the values of h_i, u are statistically close to uniform in $\mathbb{G}$. It follows that $Pr[W_1] \geq Pr[W_0] - \epsilon$ for some negligible ϵ.

Game 2. To compute a signature on ID, we compute the σ as $\sigma = y^{<\bar{\alpha}, \bar{\beta}> + \delta s}$. Since $s =< \bar{\gamma}, \bar{\beta} >$, it follows that

$$\sigma = (x^{1/t})^{<\bar{\alpha}, \bar{\beta}> + \delta s} \cdot (g^{1/t})^{s - <\bar{\gamma}, \bar{\beta}>} = (\prod_{i=1}^{n} (h_i)^{\tau_i \cdot v_i} \cdot u^s)^{1/t},$$

where $\tau_i = 1$ if $v_i \in ID$ otherwise $\tau_i = T$. Obviously, this change is only conceptual. Thus $Pr[W_2] = Pr[W_1]$.

Game 3. This is the same as Game 2, except that $\mathcal{A}$ outputs a forgery $(\tilde{\sigma}^*, s^*, t^*)$ on $ID^* \not\subset \{v_1, \cdots, v_n\}$, where $t^* = t$. A n-dimensional vector $\bar{\beta}^* = (\beta_1^*, \cdots, \beta_n^*)$ can be derived from ID^*. If the forgery is valid, then $\mathcal{C}$ computes $a =< \bar{\alpha}, \bar{\beta}^* > +\delta s^*$ and $z = \sigma^*/x^{a \cdot 1/t^*}$. It follows that

$$\begin{aligned} z &= \sigma^*/x^{a \cdot 1/t^*} \\ &= (\frac{x^{<\bar{\alpha},\bar{\beta}^*>+\delta s^*} \cdot g^{s^*-<\bar{\gamma},\bar{\beta}>}}{x^{<\bar{\alpha},\bar{\beta}^*>+\delta s^*}})^{1/t} \\ &= g^{\frac{s^*-<\bar{\gamma},\bar{\beta}>}{t}} \end{aligned}$$

If $s^* =< \bar{\gamma}, \bar{\beta} >$, then aborts, otherwise if $GCD(s^* - < \bar{\gamma}, \bar{\beta} >, t) = 1$, then it is easy to compute $g^{1/t}$ by Lemma 2.1. Then, given (N, g), $\mathcal{C}$ can output a solution for the strong RSA problem. Since $t^* = t$ is an odd prime, and the values of $s^* - < \bar{\gamma}, \bar{\beta} >$ is uniform in $\mathbb{Z}_{|\mathbb{G}|}$, the probability of $t|s^* - < \bar{\gamma}, \bar{\beta} >$ is negligible. It follows that $Pr[W_3] \geq Pr[W_2]$.

Obviously, W_3 be the event that $\mathcal{C}$ outputs a solution to the strong RSA problem, so $Pr[W_3] = StrongRSA - Adv[\mathcal{C}]$.

6 Conclusions

In this paper, we define a new homomorphic signature that is suitable for identity authentication in cloud computing. In this system, user's full signature on all identity attributes is stored in the identity authentication server in advance, and during the valid period of full signature, user only needs to secretly send a $\{0,1\}^n$ vector for the cloud service, which denotes a special identity (only involves part of identity attributes). According to the user's instruction, the identity authentication server can compute the partial signature, and send it to the cloud server for identification.

We provide a secure model of this homomorphic signature, and design a scheme from GHR signature. We proved that our scheme is secure under strong RSA assumption.

Acknowledgments.

References

1. Craig Gentry. A fully homomorphic encryption scheme. PhD thesis, Stanford University, 2009.
2. Giuseppe Ateniese, Daniel H. Chou, Breno de Medeiros, and Gene Tsudik. Sanitizable signatures. In ESORICS 05, volume 3679 of LNCS, pages 159C177, 2005.
3. Kunihiko Miyazaki, Goichiro Hanaoka, and Hideki Imai. Digitally signed document sanitizing scheme based on bilinear maps. In ASIACCS 06: Proceedings of the 2006 ACM Symposium on Information, computer and communications security, pages 343C354, 2006.

4. Stuart Haber, Yasuo Hatano, Yoshinori Honda, William Horne, Kunihiko Miyazaki, Tomas Sander, Satoru Tezoku, and Danfeng Yao. Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In ASIACCS 08, pages 353C362, 2008.
5. Christina Brzuska, Marc Fischlin, Tobias Freudenreich, Anja Lehmann, Marcus Page, Jakob Schelbert, Dominique Schroder, and Florian Volk. Security of sanitizable signatures revisited. In Public Key Cryptography, volume 5443 of LNCS, pages 317C336, 2009.
6. Christina Brzuska, Marc Fischlin, Anja Lehmann, and Dominique Schroder. Unlinkability of sanitizable signatures. In Public Key Cryptography (PKC) 10, volume 6056 of LNCS, pages 444C461, 2010.
7. Dan Boneh, David Freeman, Jonathan Katz, and Brent Waters. Signing a linear subspace: Signature schemes for network coding. In Public-Key Cryptography PKC 09, volume 5443 of Springer LNCS, pages 68C87, 2009.
8. Rosario Gennaro, Jonathan Katz, Hugo Krawczyk, and Tal Rabin. Secure network coding over the integers. In Public Key Cryptography PKC 10, volume 6056 of Springer LNCS, pages 142C160, 2010.
9. Dan Boneh and David Freeman. Homomorphic signatures for polynomial functions. In Proc. of Eurocrypt, 2011. Cryptology ePrint Archive, Report 2011/018.
10. Dan Boneh and David Freeman. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In Proc. of PKC, volume 6571 of LNCS, pages 1C16, 2011. Cryptology ePrint Archive, Report 2010/453.
11. Lei Wei, Scott E. Coull, and Michael K. Reiter. Bounded vector signatures and their applications. In ASIACCS 11, pages 277C285, 2011.
12. Alejandro Hevia and Daniele Micciancio. The provable security of graph-based one-time signatures and extensions to algebraic signature schemes. In ASIACRYPT 02, volume 2501 of LNCS, pages 379C396, 2002.
13. Mihir Bellare and Gregory Neven. Transitive signatures: New schemes and proofs. IEEE Transactions on Information Theory, 51:2133C2151, 2005.
14. Siamak Fayyaz Shahandashti, Mahmoud Salmasizadeh, and Javad Mohajeri. A provably secure short transitive signature scheme from bilinear group pairs. In Security and Communication Networks, volume 3352 of LNCS, page 6076, 2005.
15. Xun Yi. Directed transitive signature scheme. In CT-RSA 07, volume 4377 of LNCS, page 129144, 2007.
16. Gregory Neven. A simple transitive signature scheme for directed trees. Theoretical Computer Science, 396(1-3):277C282, 2008.
17. R. Gennaro, S. Halevi, and T. Rabin. "Secure hash-and-sign signatures without the random oracle." In Advances in Cryptology Eurocrypt 1999, LNCS 1592 (1999), 123C139.
18. R. Johnson, D. Molnar, D. Song, and D. Wagner. "Homomorphic signature schemes." In Topics in Cryptology CT-RSA 2002, Springer LNCS 2271 (2002), 244C262.
19. A. Shamir. "On the generation of cryptographically strong pseudorandom sequences." ACM Transactions on Computer Systems 1 (1983), 38C44.