A flexible payment scheme and its permission-role assignment

Hua Wang [1] **Jinli Cao** [2] **Yanchun Zhang** [1]

[1] Department of Maths & Computing
University of Southern Queensland
Toowoomba, QLD 4350, Australia
[2] Department of Computer Science & Computer Engineering
La Trobe University
Melbourne, VIC 3086, Australia
Email: (wang, cao, zhang)@usq.edu.au

Abstract

A flexible payment scheme and its permission-role assignments are proposed in this paper. The scheme uses electronic cash for payment transactions. In this protocol, from the viewpoint of banks, consumers can improve anonymity if they are worried about disclosure of their identities. A role called anonymity provider agent (AP) provides a high level of anonymity for consumers. The role AP certifies re-encrypted data after verifying the validity of the content from consumers, but with no private information of the consumers required. With this method, each consumer can get a required anonymity level, depending on the available time, computation and cost.

There are two types of problems that may arise in permission-role assignments. One is related to authorization granting process. Conflicting permissions may be granted to a role, and as a result, users with the role may have or derive a high level of authority. Another is related to authorization revocation. When permission is revoked from a role, the role may still have the permission from other roles. To solve these problems, we first analyze the duty separation constraints of the roles and role hierarchies in the scheme, then discuss granting a permission to a role, weak revocation permissions and strong revocation permissions for the scheme.

1 Introduction

The Internet and WWW have changed the way of business organizations and business conduct, it provides an easy way to set up shops and conduct commerce without the limitation of locations. Vendors and customers can sell and buy goods through the Internet. While this brings convenience for both consumers and vendors, many consumers have concerns about security of their private information when purchasing over the Internet, especially with electronic payment or e-cash payment. Consumers often prefer to have some degree of anonymity when shopping over the Internet, as well as data security (Pointcheval D. 2000). The issues on ensuring secure transactions for electronic commerce have attracted many researchers. There are a number of proposals for electronic cash systems (Chaum D. 1983, Cox B., Tygar J.D., Sirbu M. 1995, MastercardVisa 1997). All of them lack flexibility in anonymity. For instance, David Chaum (Chaum D. 1983) first proposed an on-line payment system that guarantees receiving valid coins. This system provides some anonymity against collaboration between shops and banks. However, consumers have no flexible anonymity and banks have to keep a very big database for consumers and coins.

Systems mentioned above are on-line payment systems. On-line payment systems protect the merchant and the bank against customer fraud, since every payment needs to be approved by the customer's bank. This will increase the computation cost, proportional to the size of the database of spent coins. If a large number of people start using the system, the size of this database could become very large and unmanageable. Digicash (Chaum D. 1995) plans to use multiple banks each minting and managing their own currency with inter-bank clearing to handle the problems of scalability. It seems likely that the host bank machine has an internal scalable structure so that it can even be set up for a 1,000,000 consumer bank. However, under the circumstances, the task of maintaining and querying a database of spent coins is probably beyond today's state-of-the-art database systems (Peirce M. and O'Mahony. D 1995).

In an off-line protocol, the merchant verifies the payment using cryptographic techniques, and commits the payment to the payment authority later in an off-line batch process. Off-line payment systems were designed to lower the cost of transactions due to the delay in verifying batch processes. Off-line payment systems, however, may suffer from the potential double spending, whereby the electronic currency might be duplicated and spent repeatedly.

The first off-line anonymous electronic cash was introduced by Chaum, Fiat and Naor (Chaum D., Fiat A., and Naor M. 1990). The security of their scheme relied on some restricted assumptions such as requiring a function, which is similar to random oracle and has to map onto a special range. There is also no formal proof attempted. Although hardly practical, their system demonstrated how off-line e-cash can be constructed and lay the foundation for more secure and efficient schemes. In 1995, Chan, Frankel and Tsiounis (Chan A., Frankel Y., and Tsiounis Y. 1995) presented a provable secure off-line e-cash scheme that relied only on the security of RSA (Rivest R. L., Shamir A., and Adleman L. M. 1978). This scheme extended the work of Franklin and Yung (Franklin M., Yung M. 1993) who aimed to achieve provable security without the use of general computation protocols. The anonymity of consumers is based on the security of RSA and it cannot be changed dynamically after the system is established. In 2000, David Pointcheval (Pointcheval D. 2000) presented a payment scheme in which the consumer's identity can be found any time by a certification authority. So the privacy of a consumer cannot be protected.

Recently, role-based access control (RBAC) has been widely used in system management and operating system products. RBAC involves individual users being associated with roles as well as roles being associated with permissions (each permission is a pair of objects and operations). As such, a role is used to associate users and permissions. A user in this model is a human being (e.g. a staff member in a bank). A role is a job function or job title within an organization

associated with authority and responsibility (e.g. role BANK manages money for consumers). Permission is an approval of a particular operation to be performed on one or more objects. The user-role assignment relation UA and permission-role assignment relation PA are many-to-many relations between users and roles, and between roles and permissions as shown in Figure 1. Assigning permissions to roles is typically the province of application administrators. Thus a banking application can be implemented so credit and debit operations are assigned to a teller role. However, approval and funding operations cannot be assigned to a teller role since they are conflicting permissions. Users are authorized to use the permissions of roles to which they are assigned. This is the essence of RBAC (Ahn Gail J. and Sandhu R. 1999).

In 1993, the National Institute of Standards and Technology (NIST) developed prototype implementations, sponsored external research (Feinstein H. L. 1995), and published formal RBAC models (Ferraiolo D. F. and Kuhn D. R. 1992, Goldschlag D., Reed M., and Syverson P. 1999). Since then, many RBAC practical applications have been implemented (Barkley J. F., Beznosov K. and Uppal J. 1999, Ferraiolo D. F., Barkley J. F. and Kuhn D. R. 1999, Sandhu R. 1998). However, there has been little research done on the usage of RBAC in electronic payment management (Sandhu R. 2001).

As mentioned above, the on-line e-cash payments need more computing resources. The previous designed off-line schemes do not provide efficient anonymity for consumers. Hence, a new payment scheme for the purchases over the Internet with the following properties will be useful.

1. Off-line, it spends less computational resources and reduces network communication.
2. Flexible anonymity, it can provide a high level anonymity for some consumers with high secure requirements.
3. RBAC management, it can decrease the complexities of a system and improve system security such as integrity.

In this paper, we present a new electronic-payment model first, then propose a flexible payment scheme, in which the anonymity of consumers is scalable and can be done by consumers themselves. Consumers can get a required anonymity without showing their identities to any third party. Furthermore, to prevent fraud and errors by conflicting permissions, the essence of RBAC for the new payment scheme is discussed.

The paper is organized as follows. In the first two sections, some basic definitions and simple examples are reviewed. The payment model and the anonymity provider agent are described in section 3. An off-line electronic cash scheme, its security analysis and an example of the scheme are shown in section 4. The relation of roles, granting permission to a role and revocation permission from a role are discussed in section 5. Related work are compared in section 6 and conclusions and future work are included in section 7.

2 Some Basic Definitions

2.1 Hash functions

$H(x)$ is a hash function. For a given value W it is computationally hard to find an x such that $H(x) = W$, i.e. collisions are hard to find, where x might be a vector.

Hash function is a major building block for several cryptographic protocols, including pseudo-random generators (Bellare M., Goldreich O., and Krawczyk H. 1999), digital signatures (Canetti R., Goldreich O., and Halevi S. 1998), and message authentication.

2.2 DLA and ElGamal encryption system

Discrete Logarithm Assumption (DLA) is an assumption that the discrete logarithm problem is believed to be difficult.

The discrete logarithm problem is as follows: given an element g in a group G of order q, and another element y of G, find x, where $0 < x < q - 1$, such that y is the result of multiplying g with itself x times, i.e. $y = g^x$. In some groups there exist elements that can generate all the elements of G by exponentiation (i.e. applying the group operation repeatedly) with all the integers from 0 to $q-1$. When this occurs, the element is called a generator and the group is called cyclic. Rivest (Rivest R. T. 1992) has analyzed the expected time to solve the discrete logarithm problem both in terms of computing power and cost and shown that it is computational hard to get x from y.

For this reason, it has been used for the basis of several public-key cryptosystems, including the famous ElGamal encryption system. The ElGamal encryption system (ElGamal T. 1985) is a public key encryption scheme which provides semantic security. Let us briefly recall it.

Table 1 indicates that the message m can be obtained only by the person who has the secret key X.

2.3 Undeniable signature scheme and Schnorr signature scheme

The undeniable signature scheme, devised by Chaum and Antwerpen (Chaum D. and Van Antwerpen H. 1990), is a non-self-authenticating signature schemes, where signatures can only be verified with the signer's consent. Schnorr, for example, proposed an undeniable signature scheme in 1991 (Schnorr C. P. 1991). It is simply recalled in Table 2. The signer is not able to deny the signature $Sch = (e, y)$ because the signature can be produced by the owner of the secret key sk only.

2.4 Role based access control

Using RBAC to manage RBAC provides a convenient way for system administrators. ARBAC97, a comprehensive model for role-based administration of RBAC was introduced by Sandhu et (Sandhu R., Bhamidipati V., Coyne E., Canta S. and Youman C. 1997). There are three components in ARBAC97.

1. User-role assignment; specifying user authorizations to perform roles,
2. Permission-role assignment; specifying which permissions are granted to roles,
3. Role-role assignment; specifying relation between role and role.

Permission-role assignment is a particularly critical administrative activity. This is because conflict defined in terms of roles allows conflicting permissions to be assigned to the same role but conflict defined in terms of permissions eliminates this possibility. Therefore, this paper will also focus on permission-role assignments for the flexible payment scheme.

3 Basic model and new payment model

Electronic cash has sparked wide interest among cryptographers ((Chaum D. 1995, Wang H. and Zhang

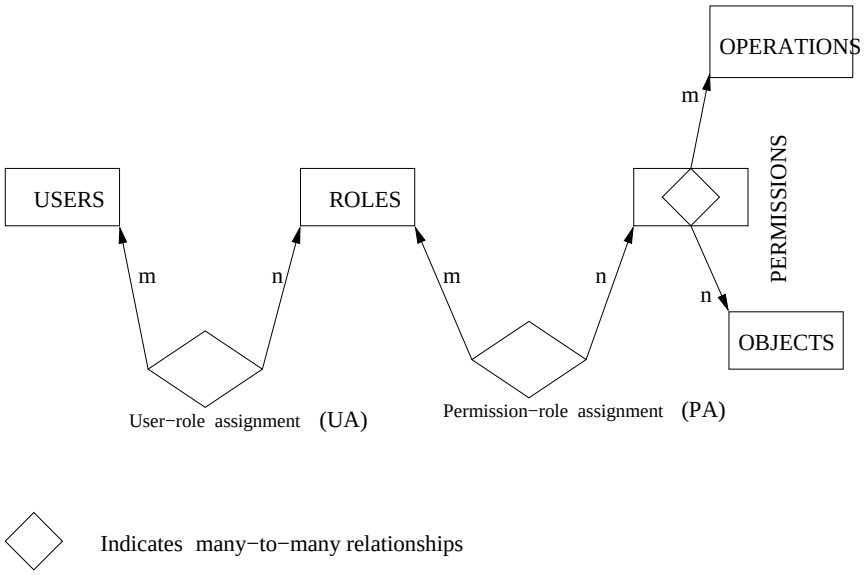

Figure 1: RBAC relationship

Step 1. The system needs a group G of order q, and a generator g. The secret key is an element $X \in Z_p = \{0, 1, ..., q-1\}$ and the public key is $Y = g^X$.
Step 2. For any message $m \in G$, the cipher text of m is $c = (g^r, Y^r m)$, for a random $r \in Z_p - \{0\}$.
Step 3. For any cipher text $c = (a, b)$, the message m can be retrieved by $m = b/a^X$.

Table 1: ElGamal encryption scheme

The system needs primes p and q such that q is divided by $(p-1)$, i.e. $q\|(p-1)$, $g \in Z_p$ with order q, i.e. $g^q = 1(modp)$, $g \neq 1$. A consumer generates by himself a private key sk which is a random number in Z_q. The corresponding public key pk is the number $pk = g^{-sk}(modp)$.
To sign message m with the private key sk the consumer performs the following steps: 1. Computes $x = g^r(modp)$, where $r \in Z_q$ is a random number. 2. Computes $e = H(x, m)$, where H is a hash function. 3. Computes $y = r + sk * e(modq)$ and output the signature $Sch = (e, y)$.
To verify the signature $Sch = (e, y)$ for message m with the public key pk a verifier computes $\overline{x} = g^y pk^e(modp)$ and checks $e = h(\overline{x}, m)$.

Table 2: Schnorr signature scheme

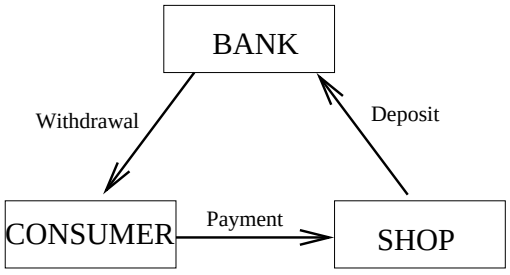

Figure 2: Basic processes of an electronic cash system

Y. 2001), etc.). In its simplest form, an e-cash system consists of three parts (a bank, a consumer and a shop) and three main procedures as shown in Figure 2 (Withdrawal, Payment and Deposit). In a coin's lifecycle, the consumer first performs an account establishment protocol to open an account with the bank.

The consumers and the shops maintain an account with the bank, while:

1. A consumer withdraws electronic coins from his/her account, by performing a withdrawal protocol with the bank over an authenticated channel.

2. The consumer spends a coin by participating in a payment protocol with a shop over an anonymous channel.

3. The shop performs a deposit protocol with the bank, to deposit the consumer's coin into its account.

The system is *off-line* if the shop does not communicate with the bank during payment. It is *untraceable* if there is no p.p.t. TM (probabilistic polynomial-time Turing Machine) that can identify a coin's origin even if one has all the information of withdrawal, payment and deposit transactions. It is *anonymous* if the bank, in collaboration with the shop, cannot trace the coin to the consumer. However, in the absence of tamper-proof hardware, electronic coins can be copied and spent multiple times by the consumer. This has been traditionally referred to as double-spending. In on-line e-cash, double-spending is prevented by having the bank check if the coin has been deposited before. In off-line e-cash, however, this solution is not possible; instead, as proposed by Chaum, Fiat and Naor (Chaum D., Fiat A., and Naor M. 1990), the system guarantees that if a coin is double-spent the consumer's identity is revealed with overwhelming probability.

There are also three additional processes: the bank setup, the shop setup, and the consumer setup (account opening). They describe the system initialization, namely the creation and posting of public keys and opening of bank accounts. Although they are certainly parts of a complete system, these are often

omitted as their functionalities can be easily inferred from the description of the three main procedures.

Besides the basic participants, a third party named anonymity provider (AP) agent is involved in the scheme. The AP agent will help the consumer to get the required anonymity but will not be involved in the purchase process. The model is shown in Figure 3. The AP agent gives a certificate to the consumer when s/he needs a high level of anonymity.

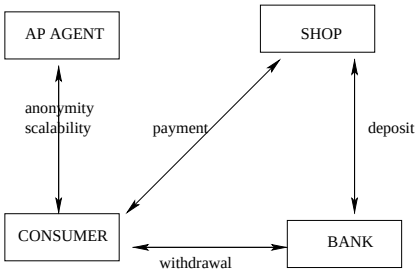

Figure 3: Electronic cash model

The AP agent cannot merge with the bank, otherwise the consumer is not able to get a coin with a high level of anonymity. We will discuss this case in detail in section 5.

3.1 Anonymity Provider Agent

Here we explain what the AP agent is. Assume a consumer owns a valid coin c with its certificate $Cert_c$, which guarantees correct withdrawal from the bank. Whether a coin is valid or not depends on its certificate. After the following processes with the AP agent, the consumer owns a new valid coin c' with its certificate $Cert_{c'}$.

1. The consumer re-encrypts the coin c into c'.
2. The consumer provides an undeniable signature Sch.
3. The consumer confirms the validity of this signature Sch to the AP agent.
4. The AP agent certifies the new coin c' and sends $Cert_{c'}$ to the consumer.

Indeed, after steps 2 and 3, the AP is convinced that the conversion has been performed by the owner of the coin c; c' is equivalent to c. The owner of c will not be able to deny Sch (the relation between c and c'). The AP agent only verifies the information of consumers. It does not need to know any private information about consumers.

3.2 Proof of ownership of a coin

Consumers have to prove the ownership of a coin to shops. Let us assume that Y is the public key of the bank, x_C is a secret key of a consumer, and $I = g^{x_C}$ the identity of a consumer. $H(x, y)$ is a hash function. A coin is an encrypted message of I: $c = (a = g^r, b = Y^r I^s)$ which is afterwards certified by the bank, where r, s are random numbers. With the certificate of the bank, one knows that the encryption is valid. Therefore, in order to prove his ownership, the consumer has just to convince of his knowledge of (x_C, r, s) such that $b = Y^r I^s$. This can be expressed as shown in Table 3.

Then, a scrambled coin is obtained by multi-plying parts of the old one, g and Y, both to the same random exponent ρ:

$$c' = (a' = g^\rho a, b' = Y^\rho b) = (g^{r+\rho}, Y^{r+\rho} I^s).$$

Then, if the owner of the old coin has certified the message $m' = h^\rho$, equivalence of both coins can be proven with the proof of equivalence of three discrete logarithms:

$$log_h m' = log_g(a'/a) = log_Y(b'/b).$$

Where h is a public variable.

4 A flexible anonymity payment scheme

In this section, we propose a scalable anonymity payment scheme. The new payment scheme has two main features, the first is that a consumer can have a high level of anonymity himself, the second is that the identity of a consumer cannot be traced unless the consumer spends the same coin twice.

The scheme includes two basic processes in system initialization (bank setup and consumer setup) and three main protocols: a withdrawal protocol with which a consumer withdraws electronic coins from a bank while his account is debited; a payment protocol with which the consumer pays the coin to a shop; and a deposit protocol with which the shop deposits the coin in the bank and has its account credited. If a consumer wants to get a high level of anonymity after getting a coin from the bank (withdrawal), s/he can contact the AP agent.

4.0.1 System Initialization

The bank setup and the consumer setup are described as follows. The details of the shop setup are omitted because the shop setup is similar to the consumer setup.

Bank setup: (Performed once by a bank)
Primes p and q are chosen such that $|p-1| = \delta + k$ for a specified constant δ, and $p = \gamma q + 1$, for a specified small integer γ. Then a unique subgroup G_q of prime order q of the multiplicative group Z_p and generator g of G_q are defined. Secret key $x_B \in_R Z_q$ for a denomination is created, where $a \in_R A$ means that the element a is selected randomly from the set A with uniform distribution. Hash function H from a family of collision intractable hash functions is also defined. The bank publishes p, q, g, H and its public key $Y = g^{x_B} (\mathrm{mod}\ p)$.

The secret key x_B is safe under the Discrete Logarithm Assumption (DLA) (EIGamal T. 1985). The hash function will be used in payment transactions.

Consumer setup : (Performed for each consumer)
The bank associates the consumer with $I = g^{x_C} (\mathrm{mod}\ p)$ where $x_C \in G_q$ is the secret key of the consumer and is generated by the consumer.

After the consumer's account and the shop's account are opened, the payment scheme can be described.

4.1 New off-line payment scheme

We now describe the new anonymity scalable electronic cash scheme, which includes withdrawal, anonymity scalability, payment and deposit.

Withdrawal: (Consumers withdraw coins from the bank)
Usually, an anonymous coin is a certified message, which embeds the public key of a consumer. In our scheme, the message is an encryption of this consumer's public key, using the public key Y of the bank.

The consumer $I = g^{x_C}$ constructs a coin $c = (a = g^r, b = Y^r I^s)$ using the public key Y of the bank, where s is the secret key of the coin, which is kept by

1. Consumers choose a random $k \in Z_p$, then compute $t = Y^k g^s (mod\, p)$ and $e = H(m, t)$ where m is a mixed message of c, current time etc, 2. Then compute $u = k - re$, $v = s - x_C e$, and $t_1 = g^{(s-1)x_C e} (mod\, p)$, 3. The signature finally consists of (e, u, v, t_1), 4. In order to verify it, one has just to compute $t' = Y^u g^v b^e (mod\, p)$ and check whether $t' = tt_1 (mod\, p)$ and $e = H(m, t'/t_1)$.

Table 3: Proof of validity of a coin $c = Y^r I^s$

the consumer and r is a random number in Z_q. Using the private key x_C, the consumer signs a Schnorr signature $Sch1$ on the message of c together with the date etc. She/He sends $(c, Sch1)$ to the bank together with r, I. Then the bank can check the correct encryption. With the signature of the coin and the date, only the legitimate consumer could have done it. After having modified the consumer's account, the bank sends back a certificate $Cert_c$.

The consumer can use the coin now without a high anonymity since the bank can easily trace any transaction performed through the coin. This is because information of the consumer such as $I, Cert_c$ are known by the bank. To solve this problem, a member in the AP agent is established to help the consumer to achieve a high level of anonymity: the consumer can derive a new encryption of his identity in an indistinguishable way. However, the consumer will need a new certificate for a new issued cipher text. The AP agent can provide this new certificate. Before certifying, the consumer requires both the previous coin $(c, Cert_c)$ and the proof of equivalence between the two cipher texts. Details are described below.

Anonymity scalability: (Performed between consumers and the AP agent)

The consumer contacts the AP agent if s/he needs to get a high level of anonymity. The consumer chooses a random ρ and re-encrypts the coin:

$$c' = (a' = g^\rho a, b' = Y^\rho b).$$

1. The consumer generates a Schnorr signature $Sch2$ on $m = h^\rho$ using the secret key x_C. Because of $Sch2$, the consumer will not be able to deny his knowledge of ρ later. Furthermore, nobody can impersonate the consumer at this step, since the discrete logarithm x_C of I is required to produce a valid signature. So there is no existential forgery.

2. The consumer also provides a designated -verifier proof of equality of discrete logarithms

$$log_h m = log_g(a'/a) = log_Y(b'/b). \qquad (1)$$

3. The consumer finally sends $c, c', Sch2, m$ to the AP agent.

4. The AP agent checks the certificate $Cert_c$ on c, the validity of the signature $Sch2$ on the message m, then certifies c' and sends back a certificate $Cert_{c'}$ to the consumer.

After these processes the consumer gets a new certified coin $c' = (a' = g^\rho a, b' = Y^\rho b)$ and a new certification $Cert_{c'}$ which is now strongly anonymous from the point of view of the bank. The AP agent has to keep (c, c', m, S) to be able to prove the link between c and c', with the help of the consumer.

Payment: (Performed between the consumer)
When a consumer possesses a coin, s/he can simply spend it at shops by proving knowledge of the secret key (x_C, s) associated with the coin c or c'. This proof is a signature $Sig = (e, u, v, t_1)$ of the new certificate $Cert_{c'}$, purchase, date, etc with the secret key (x_C, s) associating the coin to the receiver (which is later forwarded to the bank).

Deposit: (The receiver deposits a coin to a bank)
The shop will send the payment transcript to the bank. The transcript consists of the coin c or c' (if the consumer applies a higher level of anonymity), the signature and the date/time of the transaction. The bank will verify the correctness of payment and credit the coin into the shop's account.

4.2 Security Analysis

An off-line e-cash scheme is secure (Franklin M., Yung M. 1993) if the following requirements are satisfied:

1. *Unreusable*: If any consumer uses the same coin twice, the identity of the consumer can be computed.

2. *Untraceable*: With n withdrawal processes, no p.p.t. (Probabilistic polynomial time) Turing Machine can compute $(n+1)$th distinct and valid coin.

3. *Unforgeable*: With any number of the customer's withdrawal, payment and deposit protocols, no p.p.t. Turing Machine can compute a single valid coin.

4. *Unexpandable*: With any number of the customer's valid withdrawal, payment and deposit protocols, no p.p.t. Turing Machine can compute a legal consumer's identity.

The security in the e-cash scheme is based on the hardness of Discrete Logarithms (Yiannis T., Yung M. 1998) and hash functions. The system preserves the above four requirements.

Unreusable: When a consumer spends a coin, s/he hands over the coin together with a signature $S = (e, u, v, t_1)$ to a shop. If the consumer uses a coin twice, then we have two signatures $S_1 = (e_1, u_1, v_1, t_{11})$ and $S_2 = (e_2, u_2, v_2, t_{12})$, where

$$u_1 = k_1 - re_1 (modq),\ v_1 = s - x_C e_1 (modq).$$

$$u_2 = k_2 - re_2 (modq),\ v_2 = s - x_C e_2 (modq).$$

Then $(v_2 - v_1)/(e_1 - e_2) = x_C$, this is the secret key of the consumer I. So, a coin in the new scheme cannot be reused.

Untraceable: When a consumer constructs a coin, s/he uses the secret keys x_C and s, both of which are not shown to any other parts in the purchase process. So no one can trace the consumer and the coin.

Unforgeable: We first discuss whether the bank and the AP agent can forge a valid coin or not. To produce a valid coin, the first requirement is making a encryption $c = (a = g^r, b = Y^r I^s)$ of I. The second requirement is using the secret key x_C of the consumer to sign a Schnorr signature for c together with the current time. The bank can do the first step but cannot do the second step since it does not know the secret key x_C. This means the bank cannot forge

a valid coin. Similarly, the AP agent cannot forge a valid coin either. It should be noted that even though both the bank and the AP agent know a valid coin, they still couldn't use it. This is because there is a signature in payment process, which can only be done by the consumer.

As already seen, the secret key x_C of a consumer is never revealed; only used in some signatures. A consumer is therefore protected against any impersonation, even from a collusion of the bank, the AP agent, and the shop. Only the consumer can construct a valid coin since there is a undeniable signature embedded in the coin. To prevent the bank from framing the consumer as a multiple spender in the scheme, we use digital signature I^s for s which is known only by the consumer. Then the system is unforgeable.

Unexpandable: For a legal consumer and a valid coin, the secret key x_C and the random number s are never shown to others at anytime. Furthermore, usually the random number s will be changed for different coins. With n withdrawal proceedings, the random number s will be changed n times. Then, no one can compute the $(n+1)th$ distinct and valid coin even if they see n proceeding withdrawals.

4.3 An example

We give a simple example of the flexible payment scheme that will show the main steps in the processes. We omit the details of two undeniable signatures in the withdrawal and in the scalable anonymity process, because they are only used for verifying the consumer.

Bank setup

Suppose $(p, q, \gamma, k) = (47, 23, 2, 4)$, then $G_q = \{0, 1, 2, ..., 22\}$ is a subgroup of order 23. $g = 3$ is a generator of G_q. The bank's secret key $x_B = 4$ and hash function $H(x, y) = 3^x * 5^y$. The bank publishes $H(x, y)$ and $\{p, q, g\} = \{47, 23, 3\}$. The public key of the bank is $Y = g^{x_B}(modp) \equiv 34$.

Consumer setup

We assume the secret of a consumer is $x_C = 7$ and the consumer sends $I = g^{x_C}(modp) \equiv 32$ to the bank. After checking some identifications like social security card or driver's license, the bank authorizes the consumer with I.

After the bank setup and the consumer setup, the consumer can purchase.

Withdrawal

The consumer chooses $(r, s) = (2, 3)$ and computes $c = (g^r(modp), Y^r I^s(modp)) \equiv (9, 2)$, then signs a Schnorr signature $Sch1$ for the message (c, t), where t is the current time. The consumer sends $c = (9, 2)$ and $Sch1$ to the bank, the latter sends back a certificate $Cert_c$.

The coin c and its certificate $Cert_c$ are known by the bank. Therefore the consumer can be traced by the bank if s/he uses the coin directly. S/He contacts the AP agent to achieve a high level of anonymity. The consumer and the AP agent follow the processes below. We suppose $h = 37$ is a public number.

Anonymity scalability

The consumer re-encrypts the coin c, chooses $\rho = 4$ and computes $c' = (a' = g^\rho a(modp), b' = Y^\rho b(modp)) \equiv (24, 14)$ and signs a Schnorr signature $Sch2$ on $m1 = h^\rho(modp) \equiv 36$. Finally, the consumer sends $(c, c', Sch2, m1)$ to the AP agent. The latter verifies the Schnorr signature $Sch2$ and the equation (1), and sends a certificate $Cert_{c'}$ to the consumer if they are correct.

Since the new coin $c' = (24, 14)$ and its certificate $Cert_{c'}$ have no relationship with the bank, the consumer gets a level high of anonymity.

Payment

The consumer signs a signature $S = (e, u, v, t_1)$ of a message $m2$ which includes $c', Cert_{c'}$ and purchase time etc to prove the ownership of the new coin. For convenience, we assume $m2 = 11$. The consumer chooses $k = 5$ then computes $t = Y^k g^s(modp) \equiv 19$, $e = H(m2, t)(modp) \equiv 40$, $u = 18, v = 5, t_1 = 28$.

The shop who is convinced that the consumer is the owner of the coin computes $t' = 15$ if the equation of $t' = tt_1$ and the signature S are successful. She/He does not know who the consumer is.

Deposit

The bank will put the money into the shop's account when the checking of the coin $c' = (24, 14)$ and the signature $S = (e, u, v, t_1) = (40, 18, 5, 28)$ are correct. The shop can also see that the money in his account is added.

There are four roles in the scheme, BANK, SHOP, CONSUMER and AP that are acted by a bank, a shop, a consumer and an AP agent respectively. The coin c and its certificate $Cert_c$ are known by role BANK while the new coin c' and its certificate $Cert_{c'}$ are known by role AP. BANK will get c' and $Cert_{c'}$ if merging the AP agent with the bank, and then the consumer cannot get a high level of anonymity. Therefore, BANK and AP are mutually exclusive roles and they cannot assign to an individual. The concept of conflicting permissions defines conflict in terms of permissions rather than roles. For example, the permission to make a credit operation in the bank and the permission to provide a certification of high level anonymity in the AP agent are conflicting, irrespective of the roles to which they are assigned. It shows that permission-role assignment is an important issue for electronic commerce scheme.

5 Permission-role assignments

Sandhu and Bhamidipati developed an oracle implementation for permission-role assignment (Sandhu R. and Bhamidipati V. 1998). It does not discuss permission-role assignments for electronic commerce. We will analyze permission-role assignment for the new scheme in this section. Before analyzing permission-role assignment of the scheme, we need to consider the relationships of roles.

5.1 Duty separation constraints

Duty separation constraints are role-role associations indicating conflicts of interest:

c1. Static separated duty (SSD) – a constraint specifying that a user cannot be authorized for two different roles;

c2. Dynamic separated duty (DSD) – a constraint specifying that a user can be authorized for two different roles but cannot act simultaneously in both.

Role hierarchies specify which role may inherit all of the permissions of another role. In Figure 4, for example, since all staff in the AP agent, the bank and the shop are employees, their corresponding roles inherit the role EMPLOYEE. Role AP, SHOP and BANK have DSD relationships with role CONSUMER. This indicates that an individual consumer cannot act the roles of AP, SHOP or BANK simultaneously. The staff in these three participants has to first log out if they want to register as consumers. For example, a consumer, who is a staff member of the AP agent and is able to act the role AP, can ask the AP agent to help him to get a coin with a high level anonymity. But as a consumer, s/he cannot give herself/himself a new certificate $Cert_{c'}$ of a coin when

s/he works for the AP agent. Another staff member of the AP agent should do the job for this person.

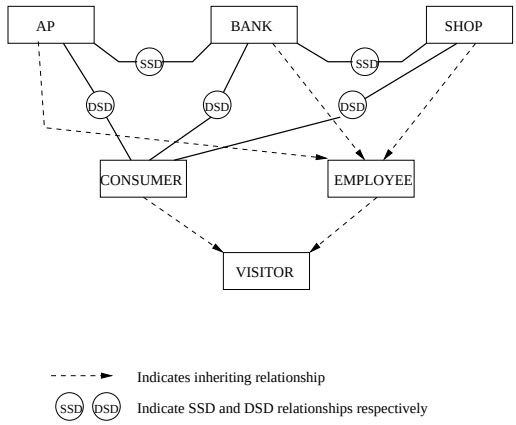

Figure 4: The relationships of the roles in the scheme

AP has an SSD relationship with BANK. This is because the duty of AP is to help a consumer to get a coin with a high level of anonymity. BANK knows the old coin $c = (g^r, Y^r I^s)$ and its certificate $Cert_c$. AP sends the new certificate $Cert_{c'}$ of the new coin $c' = (g^{r+\rho}, Y^{r+\rho} I^s)$ to the consumer. BANK will know the new certificate $Cert_{c'}$ and new coin c' if the same staff member from the AP agent and the bank processed the coin for the consumer. If this occurs, the consumer cannot have a coin with the required anonymity because BANK has known the new coin. SHOP also has an SSD relationship with BANK since BANK verifies the payment as well as depositing the coin to the shop's account. The SSD relationship is also a conflict of interest relationship like the DSD relationship but much stronger. If two roles have a SSD relationship, then they may not even be authorized to the same individual. Thus, the role AP, BANK, and SHOP may never be authorized to the same individual.

5.2 Granting and revocation models

RBAC administration encompasses the issues of assigning users to roles, assigning permissions to roles, and assigning roles to roles to define a role hierarchy. These activities are all required to bring users and permissions together. In many cases, they are best done by different administrators. To analyze granting and revocation models, we add a manager role (M1) etc in an AP agent, a manager role (M2) etc in a bank, a manager role (M3) etc in a shop and some administrative roles Senior Officer(SSO) etc in the system as shown in Figure 5 and Figure 6. A hierarchy of roles and a hierarchy of administrative roles are also shown in these two Figures. Senior roles are shown towards the top of the hierarchies and junior are to the bottom. Senior roles inherit permissions from junior roles. Permissions can be granted to or revoked from the roles in Figure 5 by the administrative roles in Figure 6.

Let $x > y$ denote x is senior to y with obvious extension to $x \geq y$. The notion of a prerequisite condition is used to restrict on what permissions can be assigned to a role.

Prerequisite condition is an expression using Boolean operators $\wedge$ and $\vee$ on terms of the form x and $\bar{x}$ where x is a role and $\wedge$ means "and", $\vee$ means "or". A prerequisite condition is evaluated for a permission p by interpreting x to be true if $(\exists x' \geq x), (p, x') \in PA$ and $\bar{x}$ to be true if $(\forall x' \geq x), (p, x') \notin PA$, where PA is a set of permission-role assignments. $\diamond$

For a given set of roles R let CR denote all possible prerequisite conditions that can be formed using the roles in R. Not every administrator can assign permission to a role. The relation of **can-assignp** $\subseteq AR \times CR \times 2^R$ provides what permissions can be assigned by administrators with prerequisite conditions, where AR is a set of administrative roles.

For example, the meaning of *can-assignp* (x, y, Z) is that a member of the administrative role x can assign a permission whose current membership satisfies the prerequisite condition y to be a member of roles in range Z.

Permission-role assignment (PA) is authorized by can-assignp relation. To identify a role range within the role hierarchy of Figure 5, we use the familiar closed and open interval notation.

$$[x, y] = \{r \in R | x \geq r \wedge r \geq y\}$$
$$(x, y] = \{r \in R | x > r \wedge r \geq y\}$$
$$[x, y) = \{r \in R | x \geq r \wedge r > y\}$$
$$(x, y) = \{r \in R | x > r \wedge r > y\}$$

The motivation behinds the can-assignp relation is in Figure 5 and Figure 6. Figure 5 shows that role E is junior-most to which all employees in the new system and role Director (DIR) is senior-most to all employees. Figure 6 shows the administrative role hierarchy which co-exist with the roles in Figure 5. The senior-most role is the Senior Security Officer (SSO). Our interest is in the administrative roles junior to SSO. These consist of three security officer roles (APSO, BankSO and ShopSO) with the relationships illustrated in the Figure 6.

Based on the role hierarchy in Figure 5 and administrative role hierarchy in Figure 6, we define the can-assignp relation shown in Table 4.

Admin.role	Prereq.Condition	Role Range
NSSO	DIR	[M1, M1]
NSSO	DIR	[M2, M2]
NSSO	DIR	[M3, M3]
APSO	FPS $\wedge\overline{OP}$	[QC, QC]
APSO	FPS $\wedge\overline{QC}$	[OP, OP]
BankSO	FPS $\wedge\overline{TE} \wedge \overline{AU}$	[AC, AC]
BankSO	FPS $\wedge\overline{TE} \wedge \overline{AC}$	[AU, AU]
BankSO	FPS $\wedge\overline{AU} \wedge \overline{AC}$	[TE, TE]
ShopSO	FPS $\wedge\overline{SALER}$	[AUDITOR, AUDITOR]
ShopSO	FPS $\wedge\overline{AUDITOR}$	[SALER, SALER]

Table 4: can-assignp

There are related subtleties that arise in RBAC concerning the interaction between granting and revocation of permission-role membership. A relation **can-revokep**$\subseteq AR \times 2^R$ shows which permissions in what role range can be revoked by administrative, where AR is a set of administrative roles. The meaning of can-revokep (x, Y) is that a member of the administrative role x (or a member of an administrative role that is senior to x) can revoke membership of a permission from any role $y \in Y$, where Y defines the *range of revocation*. Table 5 gives an example of the can-revokep relation.

Admin.role	Role Range
NSSO	[FPS, DIR)
APSO	[AP, M1)
BankSO	[Bank, M2)
ShopSO	[Shop, M3)

Table 5: can-revokep

A permission P is an *explicit member* of a role x if $(P, x) \in PA$, and P is an *implicit member* of role x if

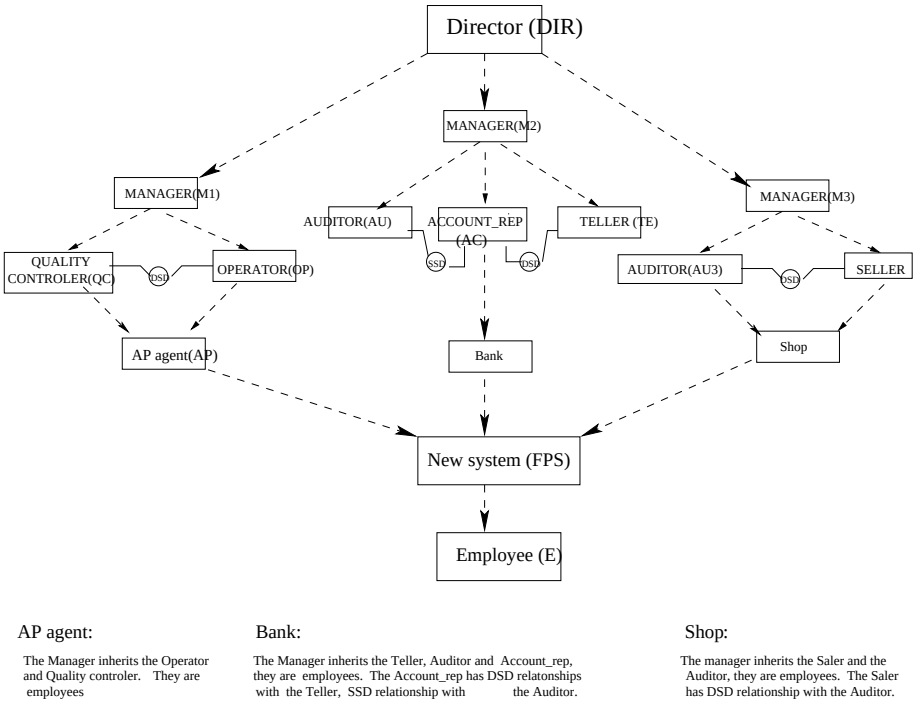

Figure 5: User-role assignment in the payment scheme

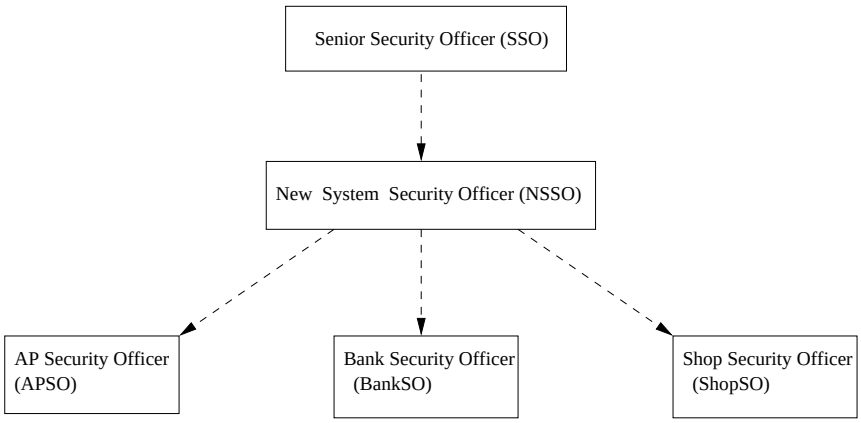

Figure 6: Administrative role assignment

for some role $x' < x, (P, x') \in PA$. Weak revocation has an impact only on explicit membership. For weak revocation, the membership of a permission is revoked only if the permission is an explicit member of the role. Therefore, weak revocation from a role x has no effect when a permission is not an explicit member of the role x. We show an example of weak revocation for the flexible scheme.

Suppose P is an explicit member of role M1, QC, AU, AP and FPS in the scheme. If Alice, with the activated administrative role APSO, can weakly revoke P's membership from QC, P continues to be an implicit member of QC since AP is junior to QC and P is an explicit member of AP. It is necessary to note that Alice should have enough power in the session to weakly revoke P's membership from explicitly assigned roles. For instance, if Alice has activated APSO and then tries to weakly revoke P from FPS, she is not allowed to proceed because APSO does not have the authority of weak revocation from FPS according to the can-revokep relation in Table 5. Therefore, if Alice wants to revoke P's explicit membership as well as implicit membership from QC by weak revocation, she needs to activate NSSO and weakly revoke P's membership from QC, AP and FPS.

Strong revocation requires revocation of both explicit and implicit membership. Strong revocation of a permission's membership in role x requires that the permission be removed not only from explicit membership in x, but also from explicit (implicit) membership in all roles junior to x. Strong revocation therefore has a cascading effect downwards in the role hierarchy.

In the scheme, for example, P is an explicit member of role M1, QC, AU, AP and FPS. If Alice, with the activated administrative role NSSO, strongly revokes P's membership from QC, then P is removed not only from explicit membership in QC, but also from explicit (and implicit) membership in all roles junior to QC. Actually, after the strong revocation from QC, P has been removed from FPS, AP as well as QC. However, P still has a membership of AU and M1 since they are not junior roles to QC based on the role hierarchy of Figure 5. This brings about the same result as weak revocation from QC, AP and FPS by NSSO. Note that all implied revocations downward in the role hierarchy should be within the revocation range of the administrative roles that are active in a session. For instance, if Alice activates APSO and tries to strongly revoke P's membership from QC, she is not allowed to proceed because FPS is junior to QC but it is out of the APSO's can-revokep range in Table 5.

6 Related work

Comparing with previous designed off-line payment schemes, the new payment scheme provides a flexible level of anonymity for consumers. This section will continue to discuss the related work on permission-role assignments. There are several other related works on permission-role assignments such as role-based access control models (Ahn G. J. and Sandhu R. 1999), an oracle implementation for permission-role assignment (Sandhu R. and Bhamidipati V. 1998).

A role-based separation of duty language (RSL 99) has been recently proposed (Ahn G. J. and Sandhu R. 1999). It has given a formal syntax and semantics for RSL99 and has demonstrated its soundness and completeness by using functions on conflicting permission sets. The proposal is different from ours in two aspects. First, It does not consider the case of the management for conflicting permissions. Therefore, there is no support to deal administrative roles with permissions in the proposal. By contrast, our work provides a rich variety of options that can deal the document of administrative roles with permissions. Second, the algorithm RSL99 does not provide access control models. It only gives separation of duty (SOD) policies. By contrast, we present a number of specialized authorization methods for access control that allow administrators to authorize permission to role or revoke permission from roles.

An oracle implementation for permission-role assignment has been proposed in (Sandhu R. and Bhamidipati V. 1998). In such a model, the difference between permission-role assignment (PRA97) and Oracle database management system was analyzed. Furthermore, through prerequisite conditions, the paper has demonstrated how to use Oracle stored procedures to implement it. However, our work substantially differs from that proposal. Differences are due to the consistency problem (Sandhu R. and Park J. S. 1998) may arise in (Sandhu R. and Bhamidipati V. 1998):

It is very difficult to keep the consistency by reflecting security requirements between global network objects and local network objects if there are hundreds of roles and thousands of permissions in a system.

This problem is in detail discussed for the flexible payment scheme in our paper because we focus on the conflicts between permissions. The granting model is used to find conflicts and prevent some secret information to be derived while the revocation model is used to check whether a role still has permissions of another role or not.

7 Conclusions and future work

In this paper, a flexible payment scheme and its role-based permission-role assignments are proposed. The new scheme provides different degrees of anonymity for consumers. Consumers can decide the levels of anonymity. They can have a low level of anonymity if they want to spend coins directly after withdrawing from the bank. Consumers can achieve a higher level of anonymity through the AP agent without revealing their private information. From the viewpoint of the bank, it is more secure because the new coin and its certificate come from the AP agent who is not involved in the payment process. Furthermore we have analyzed permission-role assignment for electronic payment. To address the problems arise in permission-role assignment, the duty separation constraints of the roles in the scheme and how to grant permission to a role associated with a can-assignp relation are discussed. Because of role hierarchies, a role may still have a permission which has been revoked by an administrative role. We have demonstrated this case in detail with weak revocation and strong revocation for the scheme.

The future work will include analyzing how to use RBAC management in generic electronic payment systems and implementation the system to convincing of its practicality.

References

Ahn Gail J. and Sandhu R. (1999), The rsl99 language for role-based separation of duty constraints, *in* 'The Fourth ACM Workshop on Role-Based Access Control, Fairfax, VA', pp. 43–54.

Peirce M. and O'Mahony. D (1995), Scaleable, secure cash payment for www resources with the payme protocol set, *in* 'The Fourth International World

Wide Web Conference in Boston, Massachusetts, USA.'.

Ahn G. J. and Sandhu R. (1999), The RSL99 Language for Role-Based Separation of Duty Constraints, *in* '4th ACM Workshop on Role-Based Access Control', Fairfax, VA, pp. 43–54.

Barkley J. F., Beznosov K. and Uppal J. (1999), Supporting relationships in access control using role based access control, *in* 'Third ACM Workshop on RoleBased Access Control', pp. 55–65.

Bellare M., Goldreich O., and Krawczyk H. (1999), Stateless evaluation of pseudorandom functions: Security beyond the birthday barrier, *in* 'Advances in Cryptology - Crypto 99', Vol. 1666 of *Lectures Notes in Computer Science*, Springer-Verlag.

Canetti R., Goldreich O., and Halevi S. (1998), The random oracle methodology, *in* 'Proceedings of the 30th ACM STOC '98', IEEE, pp. 209–218.

Chan A., Frankel Y., and Tsiounis Y. (1995), An efficient off-line electronic cash scheme as secure as RSA, Research report nu-ccs-96-03, Northeastern University, Boston, Massachussets.

Chaum D. (1983), Blind signature for untraceable payments, *in* 'Advances in Cryptology - Crypto 82', Plenum Press N.Y., pp. 199–203.

Chaum D. and Van Antwerpen H. (1990), Undeniable signatures, *in* 'Advances in Cryptology–Crypto89', Vol. 435 of *Lectures Notes in Computer Science*, Springer-Verlag, pp. 212–216.

Chaum D., ed. (1995), *An introduction to e–cash*, DigiCash, http://www.digicash.com.

Chaum D., Fiat A., and Naor M. (1990), Untraceable electronic cash, *in* 'Advances in Cryptology - Crypto 88', Vol. 403 of *Lectures Notes in Computer Science*, Springer-Verlag, pp. 319–327.

Cox B., Tygar J.D., Sirbu M. (1995), Netbill security and transaction protocol, *in* 'The first USENIX Workshop on Electronic Commerce', New York.

ElGamal T. (1985), 'A public key cryptosystem and a signature scheme based on discrete logarithms', *IEEE Transactions on Information Theory* **IT-31**(4), 469–472.

Feinstein H. L. (1995), Final report: Nist small business innovative research (sbir) grant: role based access control: phase 1. technical report, *in* 'SETA Corp.'.

Ferraiolo D. F. and Kuhn D. R. (1992), Role based access control, *in* '15th National Computer Security Conference', pp. 554–563.

Ferraiolo D. F., Barkley J. F. and Kuhn D. R. (1999), Role-based access control model and reference implementation within a corporate intranet, *in* 'TISSEC', Vol. 2, pp. 34–64.

Franklin M., Yung M. (1993), Secure and efficient off-line digital money, *in* 'Proceedings of the Twentieth International Colloquium on Automata, Languages and Programming', Vol. 700 of *Lectures Notes in Computer Science*, Springer-Verlag, pp. 265–276.

Goldschlag D., Reed M., and Syverson P. (1999), ' Onion routing for anonymous and private Internet connections', *Communications of the ACM* **24**(2), 39–41.

MastercardVisa, ed. (1997), *SET 1.0 - Secure electronic transaction specification*, http://www.mastercard.com/set.html.

Pointcheval D. (2000), Self-scrambling anonymizers, *in* 'Proceedings of Financial Cryptography', Springer-Verlag, Anguilla, British West Indies.

Rivest R. L., Shamir A., and Adleman L. M. (1978), 'A method for obtaining digital signatures and public-Key cryptosystems', *Communications of the ACM* **21**(2), 120–126.

Rivest R. T. (1992), 'The MD5 message digest algorithm', *Internet RFC 1321* .

Sandhu R. (1998), Role activation hierarchies, *in* 'Third ACM Workshop on RoleBased Access Control'.

Sandhu R. (2001), Future directions in role-based access control models, *in* 'MMS, 2001', http://www.list.gmu.edu/confrnc/misconf/.

Sandhu R. and Bhamidipati V. (1998), An oracle implementation of the pra97 model for permission-role assignment, *in* 'ACM Workshop on Role-Based Access Control', http://citeseer.nj.nec.com/27106.html, pp. 13–21.

Sandhu R. and Park J. S. (1998), Decentralized User-Role Assignment for Web-based Intranets, *in* '3th ACM Workshop on Role-Based Access Control', Fairfax, Virginia.

Sandhu R., Bhamidipati V., Coyne E., Canta S. and Youman C. (1997), The ARBAC97 model for role-based administration of roles: Preliminary description and outline, *in* 'The second ACM Workshop on Role-Based Access Control, Faixfax, Virginia', http://citeseer.nj.nec.com/sandhu97arbac.html, pp. 41–54.

Schnorr C. P. (1991), 'Efficient signature generation by smart cards', *Journal of cryptology* **4**(3), 161–174.

Wang H. and Zhang Y. (2001), Untraceable off-line electronic cash flow in e-commerce, *in* 'Proceedings of the 24th Australian Computer Science Conference ACSC2001', IEEE computer society, GoldCoast, Australia, pp. 191–198.

Wang H., Cao J. and Kambayashi Y. (2002), Building a consumer anonymity scalable payment protocol for the internet purchases, *in* '12th International Workshop on Research Issues on Data Engineering: Engineering E-Commerce/E-Business Systems', San Jose, USA.

Wang H., Cao J. and Zhang Y. (2002), Ticket-based service access scheme for mobile users, *in* 'Twenty-Fifth Australasian Computer Science Conference (ACSC2002)', Monash University, Melbourne, Victoria, Australia.

Yiannis T., Yung M. (1998), On the security of ElGamal-based encryption, *in* 'International Workshop on Practice and Theory in Public Key Cryptography (PKC '98)', Vol. 1346 of *Lectures Notes in Computer Science*, Springer-Verlag, Yokohama, Japan.