Password Authenticated Key Exchange Using Quadratic Residues

Muxiang Zhang

Verizon Communications Inc.
40 Sylvan Road, Waltham, MA 02451, USA
muxiang.zhang@verizon.com

Abstract. This paper investigates the feasibility of designing password-authenticated key exchange protocols using quadratic residues. To date, most of the published protocols for password-authenticated key exchange were based on the Diffie-Hellman key exchange. It appears inappropriate to design password-authenticated key exchange protocols using other public-key cryptographic techniques. In this paper, we show that protocols for password-authenticated key exchange can be constructed using quadratic residues and we present the first protocol of this type. Under the factoring assumption and the random oracle model, we show that our protocol is provably secure against off-line dictionary attacks. We also discuss the use of cache technique to improve the efficiency of our protocol.

1 Introduction

Password-authenticated key exchange protocols allow two entities who only share a human-memorable password to authenticate each other and agree on a large session key between them. Such protocols are attractive for their simplicity and convenience and have received much interest in the research community. A major challenge in designing password-authenticated key exchange protocols is to deal with the so-called exhaustive guessing or off-line dictionary attacks [22], as passwords are generally drawn from a small space enumerable, off-line, by an adversary. In 1992, Bellovin and Merritt [2] presented a family of protocols, known as *Encrypted Key exchange* (EKE), which was shown to be secure against off-line dictionary attacks. Using a combination of symmetric and asymmetric (i.e. public-key) cryptographic techniques, EKE provides insufficient information for an adversary to verify a guessed password and thus defeats off-line dictionary attacks. Following EKE, a number of protocols for password-based authentication and key exchange have been proposed, e.g., [3-5,8-9,12,14-18,21]. A comprehensive list of such protocols can be found in Jablon's research link [13].

Unlike other public-key based key exchange protocols such as SSL, the EKE-like protocols do not rely on the existence of a public key infrastructure (PKI). This is appealing in many environments where the deployment of a public key infrastructure is either not possible or would be overly complex. Over the last decade, many researchers have investigated the feasibility of implementing EKE

M. Jakobsson, M. Yung, J. Zhou (Eds.): ACNS 2004, LNCS 3089, pp. 233–247, 2004.

using different types of public-key cryptosystems, e.g., RSA, ElGamel, and Diffie-Hellman key exchange. Nonetheless, most of the well-known and secure variants of EKE are based on Diffie-Hellman key exchange. It seems that EKE works well with Diffie-Hellman key exchange, but presents subtleties one way or the other when implemented with RSA and other public-key cryptographic systems. In their original paper [2], Bellovin and Merritt pointed out that the RSA-based EKE variant is subject to a special type of dictionary attack, called *e-residue attack*. In 1997, Lucks [17] proposed an RSA-based password-authenticated key exchange protocol (called OKE) which was claimed to be secure against the e-residue attack. Later, Mackenzie et al [18] found that the OKE protocol is still subject to the e-residue attack. In [18], Mackenzie et al proposed an RSA-based EKE variant (called $SNAPI$) and provided a formal security proof in the random oracle model. Although the SNAPI protocol only allows using a public exponent e which is larger than the RSA modulus n, it is interesting to see that secure password-authenticated key exchange protocols can be constructed based on a diverse of public-key cryptosystems.

In this paper, we investigate the feasibility of designing password authenticated key exchange protocols using quadratic residues. A nice feature of this type of protocols is that the overhead for the protocol setup is minimal, since entities only need to share a password in *advance*; they do not need to establish other common parameters such as a prime number p and a generator g of the cyclic group modulo p. Based on number-theoretic techniques, we show that password-authenticated key exchange protocols can be constructed using quadratic residues and we present the first protocol of this type. Our protocol, called $QR\text{-}EKE$, involves two entities (say, Alice and Bob) who share a short password and one of the entity (say, Alice) also possess a Blum integer $n = pq$, where p and q are distinct prime numbers each congruent to 3 modulo 4. Using quadratic residues of n, both entities perform authentication and key establishment without leaking useful information about the password. We show that our protocol $QR\text{-}EKE$ is secure against the residue attacks as described in [2]. We also provide a formal security analysis of $QR\text{-}EKE$ under the factoring assumption and the random oracle model.

To reduce the computational load on communication entities (i.e., Alice and Bob), we present a variant of $QR\text{-}EKE$, called $QR^c\text{-}EKE$. In the protocol $QR^c\text{-}EKE$, one of the entity, say Bob, caches a hashed value of the public parameter n used by Alice in previous sessions. In a new session, Bob checks if the same parameter n is used by Alice. If yes, Bob only needs to compute two quadratic residues in the current run of the protocol $QR^c\text{-}EKE$. If else, Bob executes exactly as in the protocol $QR\text{-}EKE$ and at the end of a successful protocol run, Bob updates the cache using the new public parameter. When Alice uses the same parameter n in multiple sessions, the computational load on Bob will be greatly reduced.

The rest of the paper is organized as follows. In Section 2, we review basic concepts of number theory used throughout this paper. We provide an overview of the security model for password-authenticated key exchange in Section 3. We

present the protocol $QR\text{-}EKE$ In Section 4 and investigate its security against residue attacks. In Section 5, we improve the efficiency of $QR\text{-}EKE$ using cache technique. In Section 6, we prove the security of $QR\text{-}EKE$ under the factoring assumption and the random oracle model.

2 Preliminaries

Let $\{0,1\}^n$ denote the set of binary strings of length n and $\{0,1\}^*$ denote the set of binary strings of finite length. Without confusion, we sometimes use s_1, s_2 to denote the concatenation of two strings s_1 and s_2. A real-valued function $\epsilon(k)$ of non-negative integers is called *negligible* (in k) if for every $c > 0$, there exists $k_0 > 0$ such that $\epsilon(k) \leq 1/k^c$ for all $k > k_0$.

For every positive integer n, $n > 1$, it is well know that n can be expressed as a product of nontrivial powers of distinct primes, i.e., $n = p_1^{a_1} p_2^{a_2} \dots p_r^{a_r}$, where $p_1, p_2, \dots, p_r$ are primes and $a_1, a_2, \dots, a_r$ are positive integers. Up to a rearrangement of the prime powers, this *prime-power factorization* is unique. Let $\mathbb{Z}_n$ denote the set of non-negative integers less than n and let $\mathbb{Z}_n^*$ denote the set consisting of integers in $\mathbb{Z}_n$ that are relatively prime to n. The number of integers in $\mathbb{Z}_n^*$ is equal to the *Euler phi-function* $\phi(n)$.

Let a, b, and n be integers such that $n > 0$ and $\gcd(a, n) = c$. If $c \nmid b$, the congruence $ax \equiv b \pmod{n}$ has no solutions. If $c \mid b$, then $ax \equiv b \pmod{n}$ has exactly c incongruent solutions modulo n. Let x_0 denote one of the solutions, then the c incongruent solutions are given by

$$x = x_0 + j(n/c), \quad j = 0, 1, \dots, c-1. \tag{1}$$

Let g and n be positive integers relatively prime to each other. The least positive integer i such that $g^i \equiv 1 \pmod{n}$ is called the *order of g modulo n*. If the order of g is equal to $\phi(n)$, then g is called a *primitive root of n*. It is known (see [1,20]) that a positive integer n, $n > 1$, possesses a primitive root if and only if $n = 2, 4, p^t$ or $2p^t$, where p is an odd prime and t is a positive integer. When the positive integer n has a primitive root g, then the integers $g^0, g^1, g^2, \dots, g^{\phi(n)-1}$ form a cyclic group under the modulo n multiplication. Due to this fact, we see that if a is a positive integer relatively prime to n, then there exists a unique integer i, $0 \leq i \leq \phi(n) - 1$, such that $a = g^i \bmod n$. The integer i is called the *index of a to the base g modulo n*, and is denoted by $\mathrm{ind}_g a$. With this notation, we have $a = g^{\mathrm{ind}_g a} \bmod n$.

If n and e are positive integers and a is an integer relatively prime to n, then we say that a is a *e-th power residue of n* if the congruence $x^e \equiv a \pmod{n}$ has a solution. If a is a second power residue of n, it is also called a *quadratic residue* of n. If a is not a quadratic residue of n, it is called a *quadratic non-residue* of n. We use Q_n to denote the set of all quadratic residues of n. The set of all quadratic non-residue of n is denoted by $\bar{Q}_n$. Let p be an odd prime and a be an integer not divisible by p. The *Legendre symbol* $\left(\frac{a}{p}\right)$ is defined to be 1 if $a \in Q_p$; and -1 if $a \in \bar{Q}_p$.

3 Security Model

We consider two-party protocols for authenticated key-exchange using human-memorable passwords. In its simplest form, such a protocol involves two entities, say *Alice* and *Bob* (denoted by A and B), both possessing a secret password drawn from a small password space $\mathcal{D}$. Based on the password, Alice and Bob can authenticate each other and upon a successful authentication, establish a session key which is known to nobody but the two of them. There is present an active adversary, denoted by $\mathcal{A}$, who intends to defeat the goal for the protocol. The adversary has full control of the communications between Alice and Bob. She can deliver messages out of order and to unintended recipients, concoct messages of her own choosing, and create multiple instances of entities and communicate with these instances in parallel sessions. She can also enumerate, off-line, all the passwords in the password space $\mathcal{D}$. She can even acquire session keys of accepted entity instances. Our formal model of security for password-authenticated key exchange protocols is based on that of [5]. In the following, we review the operations of the adversary and formulate the definition of security. For details as well as motivations behind the model, please refer to [5].

Initialization. Let I denote the identities of the protocol participants. Elements of I will often be denoted A or B (Alice and Bob). We emphasis that A and B are variables ranging over I and not fixed members of I. Each pair of entities, $A, B \in I$, are assigned a password w which is randomly selected from the password space $\mathcal{D}$. The initialization process may also specify a set of cryptographic function (e.g., hash functions) and sets a number of cryptographic parameters.

Running the Protocol. Mathematically, a protocol Π is a probabilistic polynomial-time algorithms which determines how entities behave in response to received input. For each entity, there may be multiple instances running the protocol in parallel. We denote the i-th instance of entity A as Π_A^i. The adversary $\mathcal{A}$ can make queries to any instance; she has an endless supply of Π_A^i oracles ($A \in I$ and $i \in \mathbb{N}$). In response to each query, an instance updates its internal state and gives its output to the adversary. At any point in time, the instance may accept and possesses a session key sk, a session id sid, and a partner id pid. The query types, as defined in [5], include:

- $\mathsf{Send}(A, i, M)$: This sends message M to instance Π_A^i. The instance executes as specified by the protocol and sends back its response to the adversary. Should the instance accept, this fact, as well as the session id and partner id will be made visible to the adversary.
- $\mathsf{Execute}(A, i, B, j)$: This call carries out an honest execution between two instances Π_A^i and Π_B^j, where $A, B \in I, A \neq B$ and instances Π_A^i and Π_B^j were not used before. At the end of the execution, a transcript is given to the adversary, which logs everything an adversary could see during the execution (for details, see [5]).
- $\mathsf{Reveal}(A, i)$: The session key sk_A^i of Π_A^i is given to the adversary.

- $\mathsf{Test}(A, i)$: The instance Π_A^i generates a random bit b and outputs its session key sk_A^i to the adversary if $b = 1$, or else a random session key if $b = 0$. This query is allowed only once, at any time during the adversary's execution.
- $\mathsf{Oracle}(M)$: This gives the adversary oracle access to a function h, which is selected at random from some probability space Ω. The choice of Ω determines whether we are working in the standard model, or in the random-oracle model (see [5] for further explanations).

In additional to the above query types, we introduce another query type:

- $\mathsf{Impersonate}(A, i, \pi, paras)$: This replaces the password and the parameters of the instance Π_A^i by π and $paras$, respectively, where Π_A^i was not used before. After this query, the internal state of Π_A^i is visible to the adversary. Each query of this type is also called an *impersonation attempt*.

We use the Impersonate type to model an impersonation attack, which allows the adversary to test a guessed password on-line. In an impersonation attack, the adversary picks a password π as her guess and then impersonates as an instance Π_A^i to start the protocol towards another instance Π_B^j. By observing the decision of Π_B^j (i.e., accepts or rejects), the adversary can test the correctness of the guessed password π. Furthermore, by analyzing, off-line, the transcript of the execution, the adversary may be able to test passwords other than π. For a secure protocol, we expect that the adversary can only test a single password in each impersonation attempt. Certainly, the impersonation attack can be implemented by solely using the Send query type. The number of Send queries called by the adversary, however, may vary with different protocols. Using the Impersonate type, we can *explicitly* defines the number of impersonation attempts performed by the adversary. We assume that the adversary always use an impersonated instance to launch an impersonation attack.

Definition. Let Π_A^i and Π_B^i, $A \neq B$, be a pair of instances. We say that Π_A^i and Π_B^i are *partnered* if both instances have accepted and hold the same session id sid and the same session key sk. Here, we define the sid of Π_A^i (or Π_B^i) as the concatenation of all the messages sent and received by Π_A^i (or Π_B^i). We say that Π_A^i is *fresh* if: i) it has accepted; ii) it is not impersonated; and iii) a Reveal query has not been called either on Π_A^i or on its partner. With these definitions, we now define the advantage of the adversary $\mathcal{A}$ in attacking the protocol. Let Succ denote the event that $\mathcal{A}$ asks a single Test query on a fresh instance, outputs a bit b', and $b' = b$, where b is the bit selected during the Test query. The advantage of the adversary $\mathcal{A}$ is defined as $\mathsf{Adv}_{\mathcal{A}}^{ake} = 2Pr(\mathsf{Succ}) - 1$.

Definition 1. *A protocol Π is called a secure password-authenticated key exchange protocol if for every polynomial-time Adversary $\mathcal{A}$ that makes at most v impersonation attempts, the following two conditions are satisfied:*

1) *Except with negligible probability, each oracle call* $\mathsf{Execute}(A, i, B, j)$ *produces a pair of partnered instances Π_A^i and Π_B^j.*
2) $\mathsf{Adv}_{\mathcal{A}}^{ake} \leq v/|\mathcal{D}| + \varepsilon$, *where $|\mathcal{D}|$ denotes the size of the password space and ε is a negligible function.*

4 The Protocol

In this section, we present a new password-authenticated key exchange protocol called *QR-EKE*. In the protocol *QR-EKE*, there are two entities, Alice and Bob, who share a password w drawn at random from the password space $\mathcal{D}$ and Alice also possess a Blum integer $n = pq$, where p and q are primes of (about) the same size and $p \equiv q \equiv 3 \pmod 4$. Let A and B denote the identities of Alice and Bob, respectively. Before describing the protocol, let's review some of the facts about quadratic residues of Blum integers.

Let n be the product of two distinct primes p and q, $p \equiv q \equiv 3 \pmod 4$. Then for every quadratic residue a of n, i.e., $a \in Q_n$, the congruence $x^2 \equiv a \pmod n$ has four solutions x_1, x_2, x_3, x_4 in $\mathbb{Z}_n^*$. For any integer $\gamma \in \mathbb{Z}_n^*$, there is a unique square root $x_i, 1 \leq i \leq 4$, such that $x_i\gamma$ is also a quadratic residue of n, that is, $(\frac{x_i}{p}) = (\frac{\gamma}{p})$ and $(\frac{x_i}{q}) = (\frac{\gamma}{q})$. Moreover, the function $f : Q_n \to Q_n$ defined by $f(x) = x^2 \bmod n$ is a permutation. The inverse function of f is:

$$f^{-1}(y) = y^{((p-1)(q-1)+4)/8} \mod n. \tag{2}$$

It is clear that for every positive integer t, the function $f_t : Q_n \to Q_n$ defined by

$$f_t(x) = x^{2^t} \mod n \tag{3}$$

is also a permutation. For $z \in Q_n$, we can certainly compute the inverse $f_t^{-1}(z)$ by applying f^{-1} to z for t times. In fact, there is a more efficient algorithm (see [19]) for $f_t^{-1}(z)$.

Define hash functions $H_1, H_2, H_3 : \{0,1\}^* \to \{0,1\}^k$ and $H : \{0,1\}^* \to \mathbb{Z}_n$, where k is a security parameter, e.g., $k = 160$. Note that H can be implemented using a standard hash function $h : \{0,1\}^* \to \{0,1\}^l$, where l is the length of n, i.e., $l = \lceil \log_2 n \rceil$. On input x, $H(x) = h(x)$, if $h(x) < n$, and $H(x) = h(x) - \lceil n/2 \rceil$ if else. Assume that h is a random function, then for any integer $z \in \mathbb{Z}_n$, it can be proved that $|Pr(H(x) = z) - \frac{1}{n}| < 2^{-l}$; the bias is negligible. We will assume that H_1, H_2, H_3 and H are independent random functions.

The protocol *QR-EKE* is described in Fig. 1. Alice starts the protocol by sending her public parameter n and a random number $r_A \in_R \{0,1\}^k$ to Bob. Bob then verifies if n is an odd integer. If n is not odd, Bob rejects; otherwise, Bob computers an integer $t = \lfloor \log_2 n \rfloor$ and selects a random number $r_B \in_R \{0,1\}^k$. Bob also selects a quadratic residue α at random from Q_n. To do this, Bob may select a random number from $\mathbb{Z}_n^*$ and raise it to the power of 2. Bob then computes $\gamma = H(w, r_A, r_B, A, B, n)$ and checks if $\gcd(\gamma, n) = 1$. If yes, Bob assigns γ to the variable λ; otherwise, Bob assigns a random number of $\mathbb{Z}_n^*$ to λ. Next, Bob computes $z = (\lambda\alpha^2)^{2^t} \bmod n$ and sends z and r_B to Alice. Subsequently, Alice computes γ using her password w and checks if γ and n are relatively prime. If $\gcd(\gamma, n) \neq 1$, Alice assigns a random number of $\mathbb{Z}_n$ to the variable β. If $\gcd(\gamma, n) = 1$ and z is a quadratic residue, Alice sets $\beta = (\sigma\gamma^{-1})^{((p-1)(q-1)+4)/8} \bmod n$, where σ is a square root of $f_{t-1}^{-1}(z)$ such that $\sigma\gamma$ is a quadratic residue of n. Next, Alice and Bob authenticate each other using

α and β and generate the session key sk upon a successful authentication. In the protocol $QR\text{-}EKE$, both Alice and Bob intend to reject when they detect that $\gcd(\gamma, n) \neq 1$. To avoid leaking any information about this event, Alice and Bob use random numbers to compute their responses μ, z, and η. When p and q are large primes of about the same size, the probability of such an event is negligible.

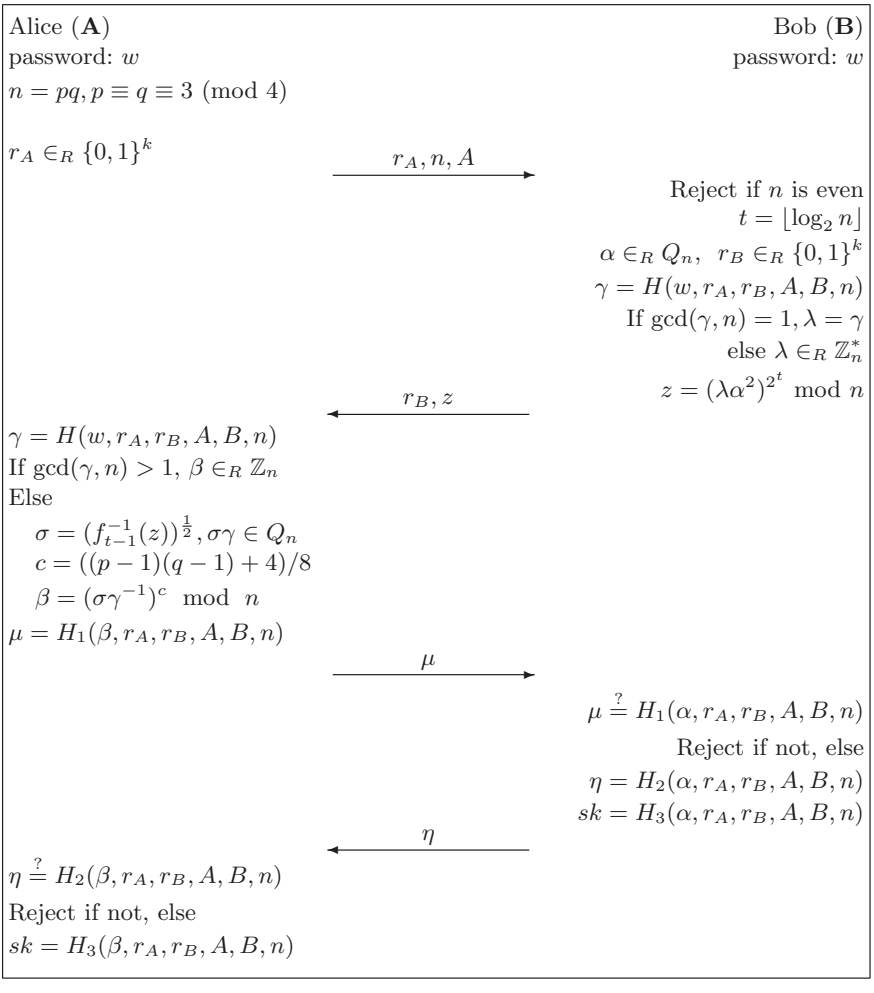

Fig. 1. The Protocol $QR\text{-}EKE$

Theorem 1. *Let n be the product of two distinct primes p and q, $p \equiv q \equiv 3$ (mod 4). Then for any integers $z \in Q_n, \gamma \in \mathbb{Z}_n^*$, and $t > 1$, the congruence $(\gamma x^2)^{2^t} \equiv z \pmod{n}$ has a unique solution in Q_n, which is given by*

$$\beta = (\sigma\gamma^{-1})^{((p-1)(q-1)+4)/8} \mod n,$$

where σ is a square root of $f_{t-1}^{-1}(z)$ such that $\sigma\gamma \in Q_n$.

Proof. Let z be a quadratic residue of n and t be a positive integer greater than 1. Since the function $f_{t-1} : Q_n \to Q_n$ as defined by $f_{t-1}(x) = (x)^{2^{t-1}} \bmod n$ is a permutation, there is a unique integer $v \in Q_n$ such that $z = (v)^{2^{t-1}} \bmod n$, i.e., $v = f_{t-1}^{-1}(z)$. For any integer $\gamma \in \mathbb{Z}_n^*$, there is a unique square root of v, denoted σ, such that $\sigma\gamma$ is a quadratic residues of n, or equivalently $\sigma\gamma^{-1} = \sigma\gamma \cdot \gamma^{-2}$ is a quadratic residue of n. By (2), $\sigma\gamma^{-1}$ has a unique square root in Q_n. Thus, the congruence $(\gamma x^2)^{2^t} \equiv z \pmod{n}$ has a unique solution in Q_n, which is given by β. □

Theorem 1 implies that when $\gamma = H(w, r_A, r_B, A, B, n)$ is relatively prime to n, Alice and Bob agree on a secret number $\beta = \alpha$ and can thus use the secret number to authenticate each other and establish a shared session key. Note that in the protocol *QR-EKE*, Bob only verifies that the integer n received from Alice is an odd number; he does not verify that n is the product of two distinct primes p and q and $p \equiv q \equiv 3 \pmod 4$. This may foster the so-called residue attack as described in [2]. In such an attack, an adversary, say, *Eva*, selects a password π_0 at random from $\mathcal{D}$ and an odd integer n which may not necessarily be a Blum integer. Then Eva impersonates as Alice and starts the protocol by sending r_E, n, A to Bob. After receiving r_B and z from Bob, Eva Computes μ and sends it back to Bob. If Bob accepts, then Eva has a successful guess of Alice's password. If Bob rejects, on the other hand, Eva excludes her guess (i.e., π_0) from the password space $\mathcal{D}$. Furthermore, Eva may exclude more passwords by repeating, *off-line*, the following three steps:

1) Eva selects a password π from $\mathcal{D}$.
2) Eva computes $\gamma = H(\pi, r_E, r_B, A, B, n)$.
3) Eva tests if $\gcd(\gamma, n) = 1$. If not, Eva returns to step 1; otherwise, Bob verifies if the congruence $(\gamma x^2)^{2^t} \equiv z \pmod{n}$ has a solution in Q_n. If the congruence has a solution, Eva returns to step 1. If the congruence has no solution in Q_n, then Eva is ensured that π is not the password of Alice. Next Eva excludes π from $\mathcal{D}$ and returns to step 1.

We say that Eva succeeds if she can exclude more than one password in each residue attack as described above. In the following, we show that our protocol *QR-EKE* is secure against residue attacks.

Theorem 2. *Let n, $n > 1$, be an odd integer with prime-power factorization $n = p_1^{a_1} p_2^{a_2} \ldots p_r^{a_r}$ and let $t = \lfloor \log_2 n \rfloor$. If z is a 2^t-th power residue modulo n, then for any $\gamma \in \mathbb{Z}_n^*$, the congruence $(\gamma x^2)^{2^t} \equiv z \pmod{n}$ has a solution in Q_n.*

Proof. To prove that $(\gamma x^2)^{2^t} \equiv z \pmod{n}$ has a solution in Q_n, we only need to prove that, for each prime power $p_i^{a_i}$ of the factorization of n, the following congruence

$$(\gamma x^2)^{2^t} \equiv z \pmod{p_i^{a_i}} \tag{4}$$

has a solution in $Q_{p_i^{a_i}}$.

Let $n_i = p_i^{a_i}, 1 \le i \le r$. Then $\phi(n_i) = p_i^{a_i - 1}(p_i - 1)$. Since n is odd, p_i is an odd prime. Hence, the integer n_i possesses a primitive root. Let g be a primitive

root of n_i, then $g^{\phi(n_i)} \equiv 1 \pmod{n_i}$. Let $\gcd(2^t, \phi(n_i)) = 2^m, 1 \leq m \leq t$. We consider the following two cases:

(1) If $m = 1$, then $d = \phi(n_i)/2$ must be an odd integer. For any integer $a \in \mathbb{Z}_{n_i}^*$, $a^{2d} \equiv 1 \pmod{n_i}$, which implies that $a^d \equiv 1$ or $-1 \pmod{n_i}$. We claim that $a^d \equiv -1 \pmod{n_i}$ if and only if a is a quadratic non-residue of n_i. If $a^d \equiv -1 \pmod{n_i}$, it is obvious that $a \in \bar{Q}_{n_i}$. On the other hand, if $a \in \bar{Q}_{n_i}$, then there exists an odd integer s such that $a = g^s \bmod n_i$ since g is the primitive root of n_i. As the order of g is $2d$, not d, we have $a^d = (g^d)^s = -1 \bmod n_i$. Similarly, we can also prove that, if $a \in Q_{n_i}$, then the congruence $x^2 \equiv a \pmod{n_i}$ has two solutions, with one solution in Q_{n_i} and another in $\bar{Q}_{n_i}$. Hence, for any $\gamma \in \mathbb{Z}_n^*$, there exists a solution x_j, $0 \leq j \leq 1$, such that $x_j\gamma \in Q_{n_i}$, that is, $(x_j\gamma)^q = -1 \pmod{n_i}$. Following the proof of Theorem 1, it is clear that the congruence (4) has a solution in Q_{n_i}.

(2) Next, we consider the case that $2 \leq m \leq t$. Since z is a 2^t-th power residue modulo n, the congruence $x^{2^t} \equiv z \pmod{n}$ has solutions in $\mathbb{Z}_n^*$. By the Chinese Remainder Theorem, the following congruence

$$y^{2^t} \equiv z \pmod{n_i} \tag{5}$$

has solutions in $\mathbb{Z}_{n_i}^*$. Let $\mathrm{ind}_g z$ denote the index of z to the base g modulo n_i and let $y \in \mathbb{Z}_{n_i}^*$ be a solution of (5). Then, $g^{2^t \mathrm{ind}_g y - \mathrm{ind}_g z} \equiv 1 \pmod{n_i}$. Since the order of g modulo n_i is $\phi(n_i)$, it follows that

$$2^t \mathrm{ind}_g y \equiv \mathrm{ind}_g z \pmod{\phi(n_i)} \tag{6}$$

Also since $\gcd(2^t, \phi(n_i)) = 2^m$, equation (6) has exactly 2^m incongruent solutions modulo $\phi(n_i)$ when taking $\mathrm{ind}_g y$ as variable. This indicates that equation (5) has exactly 2^m incongruent solutions modulo n_i. Let y_0 be one of the solutions of equation (5), by (1), the 2^m incongruent solutions of (6) are given by

$$\mathrm{ind}_g y = \mathrm{ind}_g y_0 + j\phi(n_i)/2^m \mod \phi(n_i), \quad 0 \leq j \leq 2^m - 1.$$

For any $\gamma \in \mathbb{Z}_n^*$, we have

$$\mathrm{ind}_g y - \mathrm{ind}_g \gamma = \mathrm{ind}_g y_0 - \mathrm{ind}_g \gamma + j\phi(n_i)/2^m \mod \phi(n_i), \quad 0 \leq j \leq 2^m - 1.$$

Without loss of generality, let's assume that $\mathrm{ind}_g y_0 - \mathrm{ind}_g \gamma \geq 0$; otherwise we consider $\mathrm{ind}_g \gamma - \mathrm{ind}_g y$. Since $t = \lfloor \log_2 n \rfloor$ and $\phi(n_i) < n$, it is clear that $\phi(n_i)/2^m$ is an odd integer. Hence, there exist an integer $j, 0 \leq j \leq 3 \leq 2^m - 1$, such that

$$\mathrm{ind}_g y_0 - \mathrm{ind}_g \gamma + j\phi(n_i)/2^m \equiv 0 \pmod{4},$$

which implies that there exists an integer $y \in \mathbb{Z}_{n_i}^*$ such that $y^{2^t} \equiv z \pmod{n_i}$ and $y\gamma^{-1}$ is a 4-th power residue of n_i. Therefore, the congruence (4) has a solution in Q_{n_i}, which proves the theorem. □

Theorem 2 demonstrates that, by repeating the three steps, Eva could not exclude any password from the space $\mathcal{D}$. Hence, in each residue attack , Eva

could not exclude more than one password when Bob rejects. So, our protocol $QR\text{-}EKE$ is secure against the residue attack as described in [2]. In Section 6, we will provide a formal analysis of $QR\text{-}EKE$ within the security model described in Section 3. In each run of $QR\text{-}EKE$, Bob performs one gcd operation and $t+1$ squaring operations. As $t = \lfloor \log_2 n \rfloor$, the computation time for the $t+1$ squaring operations is $\mathcal{O}((\log_2 n)^3)$. It is easy to show that the computation time for Alice is about the same. When n is large, e.g., $n \approx 2^{1024}$, the computational load is high both on Alice and on Bob. In the next section, we describe an effective way to reduce the computational load on Alice and Bob.

5 Efficiency Improvement Using Cache

In practice, Alice (who may act as a server) would most likely use the same public parameter n in many sessions, although for perfect forward secrecy, Alice would need to select a new parameter in each session. Based on this observation, we let Bob cache a hashed value of Alice's public parameter n used in previous runs of $QR\text{-}EKE$, that is, $V = h(n, A)$. The initial value of V is set to be empty. Based on the cache, we describe a computationally-efficient variant of $QR\text{-}EKE$, which is called $QR^c\text{-}EKE$ and is described in Fig. 2.

In $QR^c\text{-}EKE$, Bob computes the hashed value $h(n, A)$ of the public parameter n received from Alice and compares it with the number in the cache. If they are equal, Bob is ensured that Alice's public parameter n has not changed. In this case, Bob sets $t = 1$ and computes the number z using only two squaring operations. If the hashed value $h(n, A)$ is not equal to the number in the cached, then Bob sets $t = \lfloor \log_2 n \rfloor$ and the protocol run is identical to that of $QR\text{-}EKE$. At the end of a successful run, Bob updates the cache using the hashed value $h(n, A)$. In $QR^c\text{-}EKE$, Bob also sends the number t explicitly to Alice. Alice performs the right computation for $f_{t-1}^{-1}(z)$ based on the received number t. To show that the protocol $QR^c\text{-}EKE$ works correctly, we have the following proposition. Its proof follows directly from that of Theorem 1 and is omitted.

Proposition 1. *Let n be the product of two distinct primes p and q, $p \equiv q \equiv 3$ (mod 4). If z is a quadratic residue of n, then for any integer $\gamma \in \mathbb{Z}_n^*$, the congruence $(\gamma x^2)^2 \equiv z$ (mod n) has a unique solution in Q_n, which is given by*

$$\beta = (\sigma\gamma^{-1})^{((p-1)(q-1)+4)/8} \mod n,$$

where σ is a square root of z such that $\sigma\gamma \in Q_n$.

We need to point out that the use of cache in $QR^c\text{-}EKE$ is different than the use of *public password* (also called *hand-held certificate*) in the Halevi-Krawczyk protocol [10]. A public password, which is the hashed value of an public key, must be computed beforehand and will not be changed during its life time. The owner of the public password needs to either remember its value or carry it using a memory device. In the protocol $QR^c\text{-}EKE$, however, the cache does not need to be set beforehand; its initial value is empty. Moreover, Bob does not need to

remember the number in the cache, either; Bob can provision the cache using the protocol itself. When Alice does not change the public parameter n, Bob only needs to compute two squaring operations in each protocol run. In this case, the computation time for Bob is $\mathcal{O}((\log_2 n)^2)$, which is greatly reduced in comparison with that in *QR-EKE*.

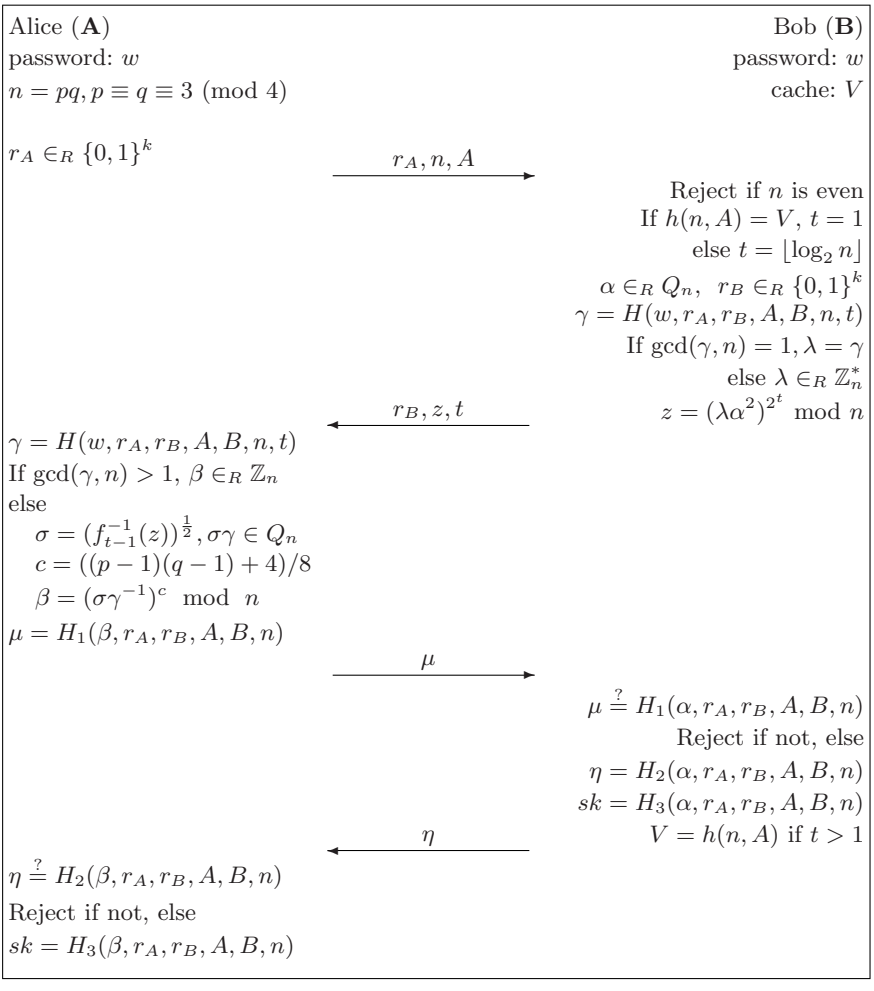

Fig. 2. The Protocol QR^c-*EKE* with a Cache

6 Formal Security Analysis

In this section, we analyze the security of *QR-EKE* within the formal model of security given in Section 3. Our analysis is based on the random-oracle model [6,7]. In this model, a hash function is modeled as an oracle which outputs a

random number for each new query. If the same query is asked twice, identical answers are returned by the oracle. In our analysis, we also assume the intractability of integer factorization.

Factoring Assumption: Let MG be a probabilistic polynomial-time algorithm that on input 1^ℓ returns a product of two distinct primes of length $\ell/2$. For any probabilistic polynomial-time algorithm C, the following probability

$$Pr(C(n) = (p, q), pq = n | n \leftarrow \mathsf{MG}(1^\ell))$$

is negligible (in ℓ).

Theorem 3. *Let $\mathcal{A}$ be a polynomial-time adversary who makes v impersonation attempts in attacking the protocol QR-EKE. Let $\pi_1, \pi_2, \ldots, \pi_v$ denote her guesses of the password (shared between A and B) in the v impersonation attempts and let E_v denote the event that one of her guesses, say π_i, is a correct guess. Under the condition that E_v is false, the adversary's success probability in attacking the protocol is equal to $Pr(\mathsf{Succ}|_{\neg E_v}) = 1/2 + \zeta$, where ζ is negligible.*

Proof. Assume that the adversary $\mathcal{A}$ makes a Test query on a *fresh* instance, which is either Π_A^i or Π_B^j, and succeeds with probability $Pr(\mathsf{Succ})$. Without loss of generality, we assume that random numbers generated by instances Π_A^i and Π_B^j and by random oracles H, H_1, H_2, H_3 never repeat. To prove that $Pr(\mathsf{Succ}|_{\neg E_v}) - 1/2$ is negligible, we consider the following two cases:

Case 1: Test query is called on Π_A^i. First, we show that, except with negligible probability, r_B, z and η could not be sent by an instance which is impersonated by $\mathcal{A}$. If r_B, z and η were sent by an instance Π_B^j which was impersonated by A, then the instance Π_B^j queried the oracle H_2 on the input $\alpha, r_A, r_B, A, B, n$ and obtained the answer η, where α is a random number selected by Π_B^j. Let λ' denote the answer of the oracle H on the input π, r_A, r_B, A, B, n, where π is the adversary's guess of the password of A. Under the condition that E_v is false, we have $\lambda' \neq \lambda$. Hence, the probability that $(\lambda'/\lambda)^{2^t} \equiv 1 \pmod{n}$ is negligible. Due to the uniqueness of the solution of $(\lambda x^2)^{2^t} \equiv z \pmod{n}$ in Q_n, it is clear that the probability $Pr(\beta = \alpha)$ is negligible also. Therefore, the probability that r_B, z and η were sent by Π_B^j is negligible.

Next, let us assume that r_B, z and η were sent by an instance Π_B^j which is not impersonated by $\mathcal{A}$. Then Π_B^j is partnered with Π_A^i. Under the assumption that random numbers generated by entity instances and by random oracles never repeat, it is clear that Π_B^j is the only instance partnered with Π_A^i. Thus, the session key sk_A^i could not be held by any instance other than Π_A^i and Π_B^j. Due to the randomness assumption of H_3, the session key sk_A^i is just a random session key for anyone without knowing β. To recover β, the only thing that the adversary could do is to perform off-line dictionary attacks, that is, the adversary selects a random password π, obtains a solution α of the congruence $(\lambda' x^2)^{2^t} \equiv z \pmod{n}$, and then tests the correctness of α using μ or η, where $\lambda' = H(\pi, r_A, r_B, A, B, n)$. Let E_β denote the event that the adversary $\mathcal{A}$ correctly recovers β in the off-line dictionary attacks. Assume that the probability $Pr(E_\beta)$ is non-negligible.

Then for any $\lambda \in \mathbb{Z}_n^*$ and $z \in Q_n$, the adversary $\mathcal{A}$ can obtain the solution of $(\lambda x^2)^{2^t} \equiv z \pmod{n}$ in Q_n with non-negligible probability. In this case, we can construct a factoring algorithm C for n as follows: the algorithm C selects $\rho \in \bar{Q}_n$ and $\lambda \in \mathbb{Z}_n^*$ and gives λ and $z = (\lambda\rho^2)^{2^t} \bmod n$ to $\mathcal{A}$. The adversary $\mathcal{A}$ solves the congruence $(\lambda x^2)^{2^t} \equiv z \pmod{n}$ and returns $\beta \in Q_n$. Under our assumption, it can be concluded that $Pr((\lambda\beta^2)^{2^t} \equiv (\lambda\rho^2)^{2^t} \pmod{n}) = Pr(E_\beta)$. Thus, $Pr(\beta^{2^{2^{t+1}}} \equiv \rho^{2^{2^{t+1}}} \pmod{n}) = Pr(E_\beta)$. Note that $f(x) = 2^{2^t} \bmod n$ is a permutation on Q_n. Hence, $Pr(\beta^2 \equiv \rho^2 (\bmod\ n)) = Pr(E_\beta)$. Since $\rho \in \bar{Q}_n$ and $\beta \in Q_n$, the algorithm C can find the factorization of n by computing $\gcd(n, \rho + \beta)$ and $\gcd(n, \rho - \beta)$, which contradicts the factoring assumption. Hence, $Pr(E_\beta)$ must be negligible.

Finally, let Auth denote the event that r_B, z and η were sent by an instance Π_B^j which is not impersonated by $\mathcal{A}$. Then $Pr(\neg\mathsf{Auth}|_{\neg E_v})$ is negligible. Moreover,

$$\begin{aligned}
Pr(\mathsf{Succ}|_{\neg E_v}) &= Pr(\mathsf{Succ}|_{\neg\mathsf{Auth}})Pr(\neg\mathsf{Auth}|_{\neg E_v}) + Pr(\mathsf{Succ}|_{\mathsf{Auth}})Pr(\mathsf{Auth}|_{\neg E_v}) \\
&\le Pr(\neg\mathsf{Auth}|_{\neg E_v}) + Pr(\mathsf{Succ}|_{\mathsf{Auth}}) \\
&= Pr(\neg\mathsf{Auth}|_{\neg E_v}) + Pr(\mathsf{Succ}|_{E_\beta})Pr(E_\beta) + Pr(\mathsf{Succ}|_{\neg E_\beta})Pr(\neg E_\beta) \\
&= Pr(\neg\mathsf{Auth}|_{\neg E_v}) + Pr(E_\beta) + 0.5(1 - Pr(E_\beta)) \\
&= 0.5 + Pr(E_\beta)/2 + Pr(\neg\mathsf{Auth}|_{\neg E_v})
\end{aligned}$$

which demonstrates that $Pr(\mathsf{Succ}|_{\neg E_v}) - 1/2$ is negligible.

Case 2: Test query is called on Π_B^j. Assume that the instance Π_B^j sent out r_B and z after receiving r_A, n, A in the first flow, where $z = (\lambda\alpha^2)^{2^t} \bmod n$, $\alpha \in Q_n$, and $\lambda = H(w, r_A, r_B, A, B, n)$. The instance Π_B^j accepted after receiving μ, which is equal to the value of $H_1(\alpha, r_A, r_B, A, B, n)$. As in Case 1, wee first show that, except with negligible probability, r_A, n, A and μ were not sent by an instance Π_A^i which is impersonated by $\mathcal{A}$.

If r_B, z and η were sent by an instance Π_A^i which is impersonated by A, then the integer n may not necessarily be a Blum integer. In addition, the adversary has knowledge of the factorization of n. Let π denote the adversary's guess of the password of B and let λ' denote the answer of the oracle H on the input π, r_A, r_B, A, B, n. Under the condition that E_v is false, we have $\lambda' \neq \lambda$. By Theorem 1, the congruence $(\lambda' x^2)^{2^t} \equiv z \pmod{n}$ has solutions in Q_n for every integer $\lambda' \in \mathbb{Z}_n^*$. Thus, the probability that the adversary could obtain α by solving the congruence is $1/|Q_n|$. Let $n = P_1^{a_1} P_2^{a_2} \dots P_r^{a_r}$ denote the prime-power factorization of n. By [11], $r \sim \ln\ln n$, which demonstrates that $1/|Q_n| \sim (\ln n)/\phi(n)$. On the other hand, it is known that (see [19]), for all integers $n \ge 5, \phi(n) > n/(6\ln\ln n)$. Hence, the probability that the adversary could recover α is negligible. Therefore, the probability that r_B, z and η were sent by Π_B^j is negligible. Next, following the analysis in Case 1, we can also show that $Pr(\mathsf{Succ}|_{\neg E_v}) - 1/2$ is negligible in Case 2. □

Theorem 4. *The protocol QR-EKE is a secure password-authenticated key exchange protocol under the factoring assumption and the random oracle model.*

Proof. It is easy to verify that the protocol $QR\text{-}EKE$ satisfies the first condition of Definition 1. To prove that protocol $QR\text{-}EKE$ also satisfies the second condition of Definition 1, let us fix a polynomial-time adversary $\mathcal{A}$ who makes v impersonation attempts in attacking the protocol $QR\text{-}EKE$. Let $\pi_1, \pi_2, \ldots, \pi_v$ denote her guesses of the password in the v impersonation attempts. Let E_v denote the event that one of the guesses, say π_i, is a correct guess. Under the condition that E_v is true, it is clear that the adversary's success probability $Pr(\mathsf{Succ})$ in attacking the protocol $QR\text{-}EKE$ is equal to 1, i.e., $Pr(\mathsf{Succ}|_{E_v}) = 1$. Let $\neg E_v$ denote that event that E_v is false, i.e., $\pi_1, \pi_2, \ldots, \pi_v$ are incorrect password-guesses. By Theorem 3, $\zeta = Pr(\mathsf{Succ}|_{\neg E_v}) - 1/2$ is negligible. Hence,

$$\begin{aligned}
\mathsf{Adv}_{\mathcal{A}}^{ake} &= 2Pr(\mathsf{Succ}) - 1, \\
&= 2Pr(\mathsf{Succ}|_{E_v})Pr(E_v) + 2Pr(\mathsf{Succ}|_{\neg E_v})Pr(\neg E_v) - 1, \\
&= 2v/|\mathcal{D}| + 2(0.5 + \zeta)(1 - v/|\mathcal{D}|) - 1, \\
&= v/|\mathcal{D}| + 2\zeta(1 - v/|\mathcal{D}|).
\end{aligned}$$

which indicates that $QR\text{-}EKE$ satisfies the second condition of Definition 1 and thus is a secure password-authenticated key exchange protocol. □

Similarly, we can also prove that $QR^c\text{-}EKE$ is a secure password-authenticated key exchange protocol under the factoring assumption and random oracle model.

References

1. E. Bach and J. Shallit, *Algorithmic Number Theory*, vol. 1: *Efficient Algorithms*, MIT Press, 1997.
2. S. M. Bellovin and M. Merritt, Encrypted key exchange: Password-based protocols secure against dictionary attacks, *Proc. of the IEEE Symposium on Research in Security and Privacy*, Oakland, May 1992, pp. 72-84.
3. S. M. Bellovin and M. Merritt, Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise, *Proc. of the 1st ACM Conference on Computer and Communications Security*, ACM, November 1993, pp. 244-250.
4. V. Boyko, P. MacKenzie, and S. Patel, Provably secure password authenticated key exchange using Diffie-Hellman, *Advances in Cryptology - EUROCRYPT 2000 Proceedings*, Lecture Notes in Computer Science, vol. 1807, Springer-Verlag, 2000, pp. 156-171.
5. M. Bellare, D. Pointcheval, and P. Rogaway, Authenticated key exchange secure against dictionary attack, *Advances in Cryptology - EUROCRYPT 2000 Proceedings*, Lecture Notes in Computer Science, vol. 1807, Springer-Verlag, 2000, pp. 139-155.
6. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols, *Proc. First Annual Conference on Computer and Communications Security*, ACM, 1993, pp. 62-73.
7. M. Bellare and P. Rogaway, Entity Authentication and key distribution, *Advances in Cryptology - Crypto 93 Proceedings*, Lecture Notes in Computer Science Vol. 773, Springer-Verlag, 1994, pp. 22-26.

8. R. Gennaro and Y. Lindell, A framework for password-based authenticated key exchange, *Advances in Cryptology - Eurocrypt 2003 Proceedings*, Lecture Notes in Computer Science Vol. 2656, Springer-Verlag, 2003, pp.524-542.
9. O. Goldreich and Y. Lindell, Session-key generation using human passwords only, *Advances in Cryptology - Crypto 2001 Proceedings*, Lecture Notes in Computer Science Vol. 2139, Springer-Verlag, 2001, pp.408-432.
10. S. Halevi and H. Krawczyk, Public-key cryptography and password protocols, *Proc. of the Fifth ACM Conference on Computer and Communications Security*, 1998, pp. 122-131,
11. G. H. Hardy, *Ramanujan: Twelve Lectures on Subjects Suggested by His Life and Work*, 3rd ed., New York: Chelsea, 1999.
12. D. Jablon, Strong password-only authenticated key exchange, *Computer Communication Review, ACM SIGCOMM*, vol. 26, no. 5, 1996, pp. 5-26.
13. D. Jablon, http://www.integritysciences.com.
14. J. Katz, R. Ostrovsky, and M. Yung, Efficient password-authenticated key exchange using human-memorable passwords, *Advances in Cryptology – Eurocrypt'2001 Proceedings*, Lecture Notes in Computer Science, Vol. 2045, Springer-Verlag, 2001.
15. K. Kobara and H. Imai, Pretty-simple password-authenticated key-exchange under standard assumptions, *IEICE Trans.*, vol. E85-A, no. 10, 2002, pp. 2229-2237.
16. T. Kwon, Authentication and key agreement via memorable passwords, *Proc. Network and Distributed System Security Symposium*, February 7-9, 2001.
17. S. Lucks, Open key exchange: How to defeat dictionary attacks without encrypting public keys, *Proc. Security Protocol Workshop*, Lecture Notes in Computer Science, Vol. 1361, Springer-Verlag, 1997, pp. 79-90.
18. P. MacKenzie, S. Patel, and R. Swaminathan, Password-authenticated key exchange based on RSA, *Advances in Cryptology—ASIACRYPT 2000 Proceedings*, Lecture Notes in Computer Science, vol. 1976, Springer-Verlag, 2000, pp. 599–613.
19. A. Menezes, P. C. van Oorschot, S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.
20. K. H. Rosen, *Elementary Number Theory and Its Applications*, 4th ed., Addison Wesley Longman, 2000.
21. T. Wu, The secure remote password protocol , *Proc. Network and Distributed System Security Symposium*, San Diego, March 1998, pp. 97-111.
22. T. Wu, A real-world analysis of Kerberos password security, *Proc. Network and Distributed System Security Symposium*, February 3-5, 1999.