Anonymous Password Based Authenticated Key Exchange with Sub-Linear Communication*

HSIAO-YING LIN AND WEN-GUEY TZENG
Department of Computer Science
National Chiao Tung University
Hsinchu, 300 Taiwan
E-mail: {lrain.cis92g@; wgtzeng@cs.}nctu.edu.tw

In this paper we propose a new anonymous password-based authenticated key exchange protocol. The communication cost of our protocol is sub-linear $O(\sqrt{N})$, which improves a previous one of $O(N)$ cost, where N is the number of users in the system. We show that the session key is secure against an active adversary in the random oracle model and identity anonymity is secure against a semi-honest adversary in the standard model.

Keywords: password based authentication, anonymous authentication, private information retrieval, authenticated key exchange, homomorphic encryption

1. INTRODUCTION

There are many websites that users can access via Internet to obtain information. Some websites would like to restrict access of their valuable data so that only authorized users can obtain them. Authentication is an essential part for a system to control access. A user's identity is authenticated by showing what secret he possesses. There are two types of secrets: long random string and password. The authentication system of using a long random string can be very secure if properly designed and used. However, it is impossible for a user to remember a long random string. Thus, a user needs an auxiliary device to store his long random secret and sometimes to provide computing capability. In practicality, users may prefer to use rememberable passwords. Thus, we focus on identity authentication based on passwords.

In some cases, a user is concerned about his privacy, such as, his interested information, access behavior, *etc*. Equivalently, the user does not want his identity to be known by a system when he retrieves data from the system. Identity anonymity and authentication seem conflicting since the system needs a user's identity for authentication and the user does not want to reveal his identity. However, an anonymous authentication protocol provides a solution.

In this paper we propose a new anonymous password-based authenticated key exchange (APAKE) protocol. The communication cost of our protocol is sub-linear $O(\sqrt{N})$, which is better than the $O(N)$ cost of a previous scheme [14], where N is the number of users in the system. In our protocol, the system and a user share a long-term password for

Received June 7, 2007; revised December 21, 2007; accepted March 20, 2008.
Communicated by Chi-Jen Lu.
* Research partially supported by the National Science Council of Taiwan, R.O.C. under grant No. NSC 95-2221-E-009-031 and Taiwan Information Security Center at NCTU (TWISC@NCTU No. 96-2219-E-009 013).

authentication. The system authenticates a user's identity without knowing it. Also, a session key is established between the system and the user for later secure communication. For security analysis, we show that the session key is secure against an active adversary in the random oracle model and identity anonymity is secure against a semi-honest adversary in the standard model.

Related work. Bellovin and Merrit proposed a PAKE protocol and proved its security against the dictionary attack [3]. Since then, many researchers proposed provably secure PAKE protocols [1, 2, 4, 6, 7, 12].

Viet, *et al.* [14] proposed the first APAKE protocol. The protocol is based on the oblivious transfer protocol of Chu and Tzeng [9] because the anonymity property is similar to choice concealment in the oblivious transfer protocol. The communication cost is $O(N)$. The security levels of session key and identity anonymity of the protocol are the same as ours.

2. PRELIMINARIES

There is a system server $\mathcal{S}$ which has a set of users $\{\mathcal{U}_0, \mathcal{U}_1, \ldots, \mathcal{U}_{N-1}\}$. Without loss of generality, we assume that the identity of user $\mathcal{U}_i$ is i. We arrange the users into an $L \times L$ rectangle such that user $\mathcal{U}_i$ is in row $i_x = \lfloor i/L \rfloor$ and column $i_y = i \bmod L$, where $N = L^2$. The server $\mathcal{S}$ shares a long-term password π_i with each user $\mathcal{U}_i$, $0 \le i \le N-1$. Assume that each password is drawn from a dictionary according to a distribution D_π. An APAKE protocol is executed by $\mathcal{S}$ and a user $\mathcal{U}_i$. Each execution of the protocol is called a *session*.

2.1 Adversaries

An adversary $\mathcal{A}$ is a probabilistic Turing machine. We consider two types of adversaries. For security of identity anonymity, the adversary is the server $\mathcal{S}$ who wants to figure out who he is talking to. This adversary is semi-honest, which means that it follows every step of the protocol, but does extra computation in order to retrieve the user's identity. For security of session keys, the adversary is an active outsider, who can remove, modify, or inject messages during an execution of the protocol.

The possible attacks of an adversary are modeled by the following three oracles.

- Execute($\mathcal{U}_i$, $\mathcal{S}$). This oracle models the eavesdropping (passive) attack. By issuing an Execute($\mathcal{U}_i$, $\mathcal{S}$) query, the adversary initiates a *new* and *genuine* session s between $\mathcal{U}_i$ and $\mathcal{S}$ and gets the transcripts of s.
- Send(s, $\mathcal{C}$, m). $\mathcal{C}$ is either the server $\mathcal{S}$ or a user $\mathcal{U}_i$. This oracle models an active attack. It returns the response message of $\mathcal{C}$ when $\mathcal{C}$ on session s receives message m.
  When the adversary wants to start a *fake* session s for $\mathcal{C}$, it executes Send(s, $\mathcal{C}$, "start") oracle and gets the first message responded by $\mathcal{C}$ for the session s.
- Reveal(s). This oracle models the misuse of the session key by some party in the session. It returns the session key of a genuine session s if the session key of s has ever been established.

We distinguish genuine sessions and fake sessions. A genuine session is established by the server $\mathcal{S}$ and a user $\mathcal{U}_i$ and a fake session involves an adversary who intends to impersonate. The adversary can inject false or modified messages into a fake session by Send queries. He tries to trick $\mathcal{S}$ or $\mathcal{U}_i$ to accept him, or to obtain information about the session key.

We require the Reveal oracle work for genuine sessions. The goal of a fake session is for impersonation. If the adversary successfully impersonates in a fake session, he has already had the session key. If the adversary fails to impersonate in a fake session, there will be no common session key between the adversary and the other party. Thus, there is no point to ask the other party to reveal his session key for this fake session.

2.2 Hard Problems

Let G be a cyclic group and g be a generator of G. The *CDH* problem is to compute g^{xy} from given g^x and g^y. Let $\mathsf{Succ}_{g,G}^{cdh}(t)$ be the best probability that a t-time bounded Turing machine solves the *CDH* problem. The *CDH* assumption is that the probability $\mathsf{Succ}_{g,G}^{cdh}(t)$ is negligible for any polynomial-time t.

Let G and G_1 be two cyclic groups of order $n = q_1q_2$, where q_1 and q_2 are large primes with bit length λ. Let g be a generator of G and e: $G \times G \to G_1$ be a bilinear map. The subgroup decision (SD) problem is to determine whether an element $x \in G$ is order q_1 or n without knowing the factoring of n. Let $\mathsf{Adv}_n^{sd}(t)$ be the greatest advantage among all t-time bounded Turing machines solving the SD problem. The SD assumption is that no probabilistic polynomial-time Turing machine can solve the SD problem with a non- negligible probability.

2.3 The BGN Cryptosystem

Our protocol uses the BGN cryptosystem [5], which is semantic secure if the SD assumption holds. The BGN cryptosystem has the properties of additive and one-time multiplicative homomorphisms. These properties are important for our construction. The following is a brief description of the BGN cryptosystem. It has two encryption functions E and E_1 that work over G and G_1 respectively.

– $\mathsf{KeyGen}(1^\lambda)$: Given a security parameter λ, the tuple (n, g, G, G_1, e) is generated such that the tuple is the parameters for the SD problem in section 2.2 and the length of q_1, q_2 is λ bits. We choose another generator g' of G and compute $h = (g')^{q_2}$. Then, the public key is $pk = (n, g, h, G, G_1, e)$ and the secret key is $sk = (q_1, q_2)$.
– $\mathsf{E}(pk, m)$: Choose a random number $r \in Z_n^*$ and compute the ciphertext $C = g^m h^r$, where $m \in \{0, 1, \ldots, q_2 - 1\}$. For practicality, the message space should be small enough.
– $\mathsf{D}(sk, C)$: Compute $C' = C^{q_1} = (g^m h^r)^{q_1} = (g^{q_1})^m$ and then exhaustively search m from g^{q_1} and C'.
– $\mathsf{E}_1(pk, m)$: Choose a random number $r \in Z_n^*$ and compute the ciphertext $C = e(g, g)^m \cdot e(g, h)^r$, where $m \in \{0, 1, \ldots, q_2 - 1\}$.
– $\mathsf{D}_1(sk, C)$: Compute $C' = C^{q_1} = (e(g, g)^m \cdot e(g, h)^r)^{q_1} = (e(g, g)^{q_1})^m$ and then exhaustively search m from $e(g, g)^{q_1}$ and C'.

This cryptosystem is additive homomorphic for both E and E_1. For E, we have

$$\mathsf{E}(pk, m_1) \cdot \mathsf{E}(pk, m_2) = g^{m_1}h^{r_1} \cdot g^{m_2}h^{r_2} = g^{m_1+m_2}h^{r_1+r_2},$$

which is an encryption of $m_1 + m_2$. Similarly, E_1 is additive homomorphic.

The encryption E is multiplicative homomorphic. However, the multiplication of two messages can be executed once only. The execution result of message multiplication is a ciphertext over G_1. Although we can continue to perform message addition over G_1, no more message multiplication is possible. The message multiplication is as follows. For two ciphertexts $C_1 = g^{m_1}h^{r_1}$ and $C_2 = g^{m_2}h^{r_2}$ over G, we compute

$$C = e(C_1,\ C_2) \times e(g,h)^r = e(g,g)^{m_1 m_2} \cdot e(g,h)^{\tilde{r}},$$

which is an encryption of m_1m_2 (over G_1), where r is randomly chosen from Z_n^* and $\tilde{r} = m_1r_2 + m_2r_1 + r_1r_2 \log_g h + r$.

Modification for our setting. We add one parameter $\omega = e(g, g)^{q_1}$ into the public key. We assume that this addition does not affect the security of the BGN cryptosystem.

3. SECURITY REQUIREMENTS

An APAKE protocol $\prod$ should meet the following security requirements for authentication, anonymity and session keys (AKE security).

- *Authentication* An adversary $\mathcal{A}$ successfully impersonates as a user $\mathcal{U}_i$ if the server $\mathcal{S}$ accepts a fake session s and establishes a common session key s with $\mathcal{A}$ for session s. Similarly, an adversary $\mathcal{A}$ successfully impersonates as the server $\mathcal{S}$ if any user $\mathcal{U}$ who interacts with $\mathcal{A}$ accepts and establishes a common session key for a fake session s. Let $\mathsf{Succ}^{S\text{-auth}}(\mathcal{A})$ ($\mathsf{Succ}^{U\text{-auth}}(\mathcal{A})$, resp.) be the probability that $\mathcal{S}(\mathcal{U}$, resp.) accepts and establishes a se be the advantage that $\mathcal{A}$ guesses $\mathsf{Reveal}(s)$ correctly (determining whe ssion key with $\mathcal{A}$. We say that $\prod$ is (t, ϵ)-secure for authentication if for any t-time bounded adversary $\mathcal{A}$, both $\mathsf{Succ}^{S\text{-auth}}(\mathcal{A})$ and $\mathsf{Succ}^{U\text{-auth}}(\mathcal{A})$ are less than ϵ.
- *Anonymity* Let $\mathsf{View}_{\mathcal{S}}(\mathcal{U}_i)$ be the view of $\mathcal{S}$ for an interaction session with user $\mathcal{U}_i$. We say that $\prod$ meets the security of anonymity if for any $0 \le i, j \le N - 1$, and any polynomial-time bounded server $\mathcal{S}$, $\mathsf{View}_{\mathcal{S}}(\mathcal{U}_i)$ and $\mathsf{View}_{\mathcal{S}}(\mathcal{U}_j)$ are polynomially indistinguishable.
- *AKE security* The AKE security is defined by the AKE game. The adversary is allowed to query $\mathsf{Execute}(\mathcal{S}, \mathcal{U}_i)$, $\mathsf{Send}(s, \mathcal{C}, m)$ and $\mathsf{Reveal}(s)$ oracles for collecting information. At the end, $\mathcal{A}$ queries the $\mathsf{Test}(s)$ oracle which returns either a session key of s or a random number with equal probability, where s is a fresh and genuine session between the server S and a user $\mathcal{U}_i$. By a *fresh s*, we mean that $\mathsf{Reveal}(s)$ is never queried by $\mathcal{A}$. Let $\mathsf{Adv}^{\text{ake}}(\mathcal{A})$ther it is a real session key or a random number.) We say that $\prod$ is (t, ϵ)-AKE-secure if for any t-time bounded adversary $\mathcal{A}$, $\mathsf{Adv}^{\text{ake}}(\mathcal{A}) < \epsilon$.

4. OUR APAKE PROTOCOL

The main idea of our protocol comes from symmetric private information retrieval (SPIR) in which a user looks up some data from a server's database, but does not reveal which data he selects. On the other hand, the server does not want the user to learn other data.

In our protocol, a user uses his identity as the chosen index to retrieve the corresponding password stored in the server. The passwords stored in the server are re-arranged in an $L \times L$ table T, such that the password π_i of user $\mathcal{U}_i$ is in row $i_x = \lfloor i/L \rfloor$ and column $i_y = i \bmod L$, *i.e.* $T[i_x, i_y] = \pi_i$. The identity sent by the user is encrypted and the retrieved data is encrypted also (*i.e.* the corresponding password is in encrypted form). To prevent the user from obtaining a password that does not belong to him, the server adds a random value to the retrieved data before sending the encrypted data back to the user. Thus, only the valid and as-claimed user can use his password to get the random value from the retrieved data. This random value is needed for computing the session key.

Let ψ be a hash function from $\{0, 1\}^l$ to G_0, and H_0, H_1, H_2 be hash functions from $\{0, 1\}^*$ to $\{0, 1\}^l$ for some positive integer l, which are the public parameters of the system. The user $\mathcal{U}_a$ with the password π_a can be anonymously authenticated by $\mathcal{S}$ by executing our APAKE protocol described as follows.

1. $\mathcal{U}_a$ does the following:
- Generate the key pair (pk, sk) and give a zero knowledge proof ϕ_1 for validation of the key pair[1], where $pk = (n, g, h, \omega, G, G_1, e)$ and $sk = (q_1, q_2)$ such that every element in G_1 can be expressed as a l-bit string.
- Represent the identity a by two indicator bitstrings $(x_0x_1 \ldots x_{L-1})$ and $(y_0y_1 \ldots y_{L-1})$ of row position a_x and column position a_y respectively. Denote the representation as

  $$\beta_a = (x_0x_1 \ldots x_{L-1}, y_0y_1 \ldots y_{L-1})^2.$$

- Encrypt each bit of $(x_0x_1 \ldots x_{L-1}, y_0y_1 \ldots y_{L-1})$ as

  $$C_x = (\mathsf{E}(pk, x_0), \mathsf{E}(pk, x_1), \ldots, \mathsf{E}(pk, x_{L-1})),$$
  $$C_y = (\mathsf{E}(pk, y_0), \mathsf{E}(pk, y_1), \ldots, \mathsf{E}(pk, y_{L-1})).$$

  We use zero knowledge proofs ϕ_2 to guarantee the format of the encrypted identity representation[3].
- Compute $X = e(g, g)^{x'}$, where $x' \in Z_n$ is randomly selected.
- Send $(pk, C_x, C_y, X, \phi_1, \phi_2)$ to $\mathcal{S}$.

2. After receiving (pk, C_x, C_y, X) from $\mathcal{U}_a$, $\mathcal{S}$ does the following:
- Verify correctness of ϕ_1 and ϕ_2.
- Compute an $L \times L$ (filtering) table F, where $F[i, j] = e(\mathsf{E}(pk, x_i), \mathsf{E}(pk, y_j)) = \mathsf{E}_1(pk, x_iy_j)$, for $0 \le i, j \le L - 1$. We see that $F[i, j] = \mathsf{E}_1(pk, 1)$ if and only if both x_i and y_j are 1. There is exactly one $\mathsf{E}_1(pk, 1)$ in F.
- Compute $A = \prod_{i,j} F[i, j]^{T[i,j]} = \mathsf{E}_1(pk, T[a_x, a_y]) = \mathsf{E}_1(pk, \pi_a)$, for $0 \le i, j \le L - 1$.

[1] For the validation of the key pair, we consider that the validation of the RSA modulus n should be proved [8].

[2] For example, $N = 25$ the representation of $a = 18$ would be $\beta_{18} = (00010, 00010)$, where the row position $18_x = 3$ and column position $18_y = 3$.

[3] For each ciphertext, it should be proved that the plaintext is either 0 or 1 [10]. After all ciphertexts are proved as bit-encryptions, the product of all ciphertexts in C_x should be proved as an encryption of bit as well as the product of all ciphertexts in C_y. Totally, it costs $L + 2$ zero knowledge proofs for bit encryptions to prove the validation of the format. From [10], each zero knowledge proof for bit encryption consists 3 elements in G and the proof is non-interactive.

- Randomly select y', $r \in Z_n$ and compute $B = e(g, g)^r \cdot A = \mathsf{E}_1(pk, \pi_a + r)$, $R = \omega^r$ and $Z = \psi(R) \cdot e(g, g)^{y'}$.
- Compute $K_{\mathcal{S}} = X^{y'}$.
- Send $(\mathcal{S}, B, Z)$ to $\mathcal{U}_a$.

3. After receiving $(\mathcal{S}, B, Z)$ from $\mathcal{S}$, $\mathcal{U}_a$ computes $R' = B^{q_1}/e(g, g)^{\pi_a q_1}$, $Y' = Z/\psi(R')$, and $K_{\mathcal{U}} = Y'^{x}$. Compute the authenticator $Auth_{\mathcal{U}} = H_1(pk, C_x, C_y, X, \mathcal{S}, B', Z, \psi(R'), K_{\mathcal{U}})$. Send $Auth_{\mathcal{U}}$ to $\mathcal{S}$.

4. After receiving $Auth_{\mathcal{U}}$ from $\mathcal{U}_a$, $\mathcal{S}$ does the following
- Check whether

  $Auth_{\mathcal{U}} = H_1(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K_{\mathcal{S}})$. If they are matched, then $\mathcal{S}$ accepts $\mathcal{U}_a$ and computes the session key

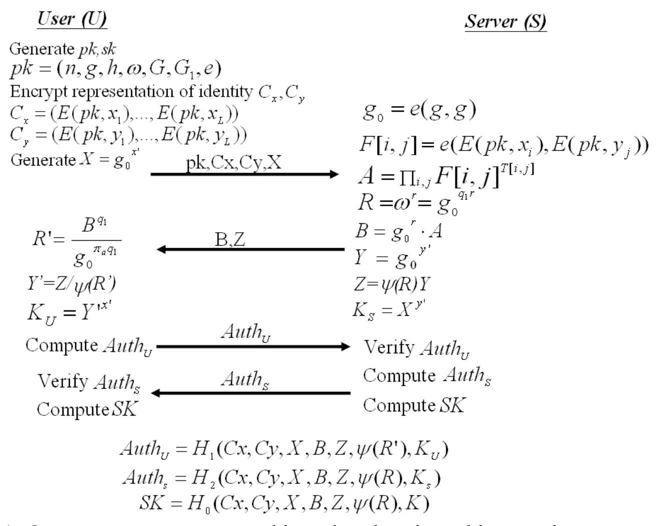
  $SK = H_0(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K_{\mathcal{S}})$.

  Otherwise, $\mathcal{S}$ rejects and terminates.
- Compute $Auth_{\mathcal{S}} = H_2(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K_{\mathcal{S}})$ and send $Auth_{\mathcal{S}}$ to $\mathcal{U}_a$

5. After receiving the authenticator $Auth_{\mathcal{S}}$, $\mathcal{U}_a$ does the following:
- Check whether

  $Auth_{\mathcal{S}} = H_2(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R'), K_{\mathcal{U}})$. If they are matched, set $SK = H_0(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K_{\mathcal{U}})$. Otherwise, $\mathcal{U}_a$ rejects and terminates.

To revoke a user $\mathcal{U}_i$, $\mathcal{S}$ resets the password π_i to a secret value $\perp$, which is unknown to all users. The revoked user has no valid password anymore. To add a user $\mathcal{U}_j$, $\mathcal{S}$ sets the corresponding entry as π_j in his id-password table T.



User (U)
Server (S)
Generate pk, sk
$pk = (n, g, h, \omega, G, G_1, e)$
Encrypt representation of identity C_x, C_y
$C_x = (E(pk, x_1), \ldots, E(pk, x_L))$
$C_y = (E(pk, y_1), \ldots, E(pk, y_L))$
Generate $X = g_0^{x'}$
pk,Cx,Cy,X
$g_0 = e(g, g)$
$F[i, j] = e(E(pk, x_i), E(pk, y_j))$
$A = \Pi_{i,j} F[i, j]^{T[i,j]}$
$R = \omega^r = g_0^{q_1 r}$
$R' = \frac{B^{q_1}}{g_0^{\pi_a q_1}}$
B,Z
$B = g_0^{r} \cdot A$
$Y = g_0^{y'}$
$Y' = Z/\psi(R')$
$Z = \psi(R)Y$
$K_U = Y'^{x'}$
$K_S = X^{y'}$
Compute $Auth_U$
$Auth_U$
Verify $Auth_U$
Verify $Auth_S$
$Auth_S$
Compute $Auth_S$
Compute SK
Compute SK
$Auth_U = H_1(Cx, Cy, X, B, Z, \psi(R'), K_U)$
$Auth_s = H_2(Cx, Cy, X, B, Z, \psi(R), K_s)$
$SK = H_0(Cx, Cy, X, B, Z, \psi(R), K)$

Fig. 1. Our anonymous password based authenticated key exchange protocol.

4.1 Correctness and Security

In this section, we show correctness and security of our protocol. First, correctness is shown in the following theorem.

Theorem 1 After execution of the protocol by both honest parties, a user $\mathcal{U}_a$ with the correct password π_a can establish a common session key SK with $\mathcal{S}$.

Proof: In the execution of the protocol, $\mathcal{U}_a$ has the password $\pi = \pi_a$, we can see that $R' = (B/(e(g, g))^{\pi})^{q_1} = R$ and $Y' = Z/\psi(R') = Z/\psi(R) = e(g, g)^y$. Thus $K_{\mathcal{U}} = Y'^x = e(g, g)^{xy} = X^y = K_{\mathcal{S}}$. Therefore, authenticators are matched and then $\mathcal{U}$ can establish the session key SK with $\mathcal{S}$. ❑

AKE security/Authentication Our protocol can achieve mutual authentication. Briefly speaking, only the user who knows the correct password can extract R out from B and then extract $e(g, g)^y$ from the message Z. Therefore, only the user can compute the same key $K_{\mathcal{U}} = (e(g, g)^y)^x$ as the one $K_{\mathcal{S}}$ computed by the server. On the other hand, only the server who knows $e(g, g)^{\pi}$ can compute the key $K_{\mathcal{S}}$ such that $K_{\mathcal{S}}$ is the same as $K_{\mathcal{U}}$.

For the requirements of AKE security and authentication, our protocol can against off-line dictionary attack. The number of Reveal-queries can be dominated by the number of Execute-queries, since the Reveal oracle only works for the genuine sessions.

The AKE security is independent of the security of the BGN's cryptosystem. For an adversary who is an outsider of a session, the AKE security still holds even if the adversary gets the secret key used in the session. Informally, the reason is that the adversary is without neither a password table nor the password of the known identity (he can compute by the secret key). Without the password for the identity and the password table, the adversary with sk can not compute R and also the key $K_{\mathcal{U}}$ or $K_{\mathcal{S}}$. For clarity, we put the proof details in the appendix.

Theorem 2 Suppose that our APAKE protocol is run with the password distribution $\mathcal{D}_{\pi}$. For any adversary $\mathcal{A}$ with a time bound t with less than q_s Send-queries, q_e Execute- queries, q_g hash queries to ψ, and q_h hash queries to $\{H_0, H_1, H_2\}$, we have

$$\mathsf{Adv}^{ake}(\mathcal{A}) \le \frac{4q_s}{2^l} + 18\mathcal{D}_{\pi}(q_s) + 24q_h^2 \times \mathsf{Succ}^{cdh}_{g_0,G_1}(t+2\tau_e) + \frac{6(q_s+q_e)^2}{n} + \frac{6(q_g)^2}{n},$$

$$\mathsf{Succ}^{S\text{-}auth}(\mathcal{A}) \le \frac{q_s}{2^l} + 3\mathcal{D}_{\pi}(q_s) + 4q_h^2 \times \mathsf{Succ}^{cdh}_{g_0,G_1}(t+2\tau_e) + \frac{(q_s+q_e)^2}{n} + \frac{(q_g)^2}{n},$$

$$\mathsf{Succ}^{C\text{-}auth}(\mathcal{A}) \le \frac{q_s}{2^l} + 3\mathcal{D}_{\pi}(q_s) + 4q_h^2 \times \mathsf{Succ}^{cdh}_{g_0,G_1}(t+2\tau_e) + \frac{(q_s+q_e)^2}{n} + \frac{(q_g)^2}{n},$$

where τ_e is the computational time for an exponentiation in G_1, $g_0 = e(g, g)$ is a generator in G_1 and $\mathcal{D}_{\pi}(\tau)$ is denoted for the probability of the most probable set of τ passwords.

Anonymity Here we show that the server cannot distinguish the views of interacting with different users.

Table 1. Comparison.

	Computation of user	Computation of server	Communication	round
Ours	$(4\sqrt{N}+4)\rho$	$(N+3)\rho+N\delta$	$O(\sqrt{N})$	4
[14]	4ρ	$(2N+2)\rho$	$O(N)$	3

* Computation cost is measured in the number of modular exponentiation and the number of pairing denoted as ρ and δ respectively.

Theorem 3 Our APAKE protocol is anonymous against the semi-honest server if the BGN' cryptosystem is semantic secure.

Proof: Assume that the server $\mathcal{S}$ can distinguish two transcripts from two users $\mathcal{U}_a$ and $\mathcal{U}_b$ with advantage ϵ. Then we can use $\mathcal{S}$ to break the semantic security of the BGN's cryptosystem with advantage ϵ.

Let $\{pk_a, Cx_a, Cy_a, X', Auth_{\mathcal{U}_a}\}$, $\{pk_b, Cx_b, Cy_b, X'', Auth_{\mathcal{U}_b}\}$ be transcripts of $\mathcal{U}_a$, $\mathcal{U}_b$. Since both users have the correct passwords and $Auth_{\mathcal{U}_a}$, $Auth_{\mathcal{U}_b}$ are generated by the random oracle H_1, the server $\mathcal{S}$ can not gain information from the authenticators. The distributions of X' and X'' are identical. Since pk is chosen randomly and independent of the users, pk_a and pk_b are identically distributed. The only place where can be used for distinguishing is the encrypted identity representation. If the server can distinguish the encrypted identity representation, we can use the server to break the semantic security of the BGN's cryptosystem as follows.

Let the ith, jth bits of indicator bitstrings of row positions a_x and b_x be different, and the kth, sth bits of the indicator bitstrings of column positions a_y and b_y be different. That is,

$$\begin{aligned}\beta_a &= (x_0 \dots x_{i-2}\mathbf{u}x_i \dots x_{j-2}\bar{\mathbf{u}}x_j \dots, y_0 \dots y_{k-2}\mathbf{v}y_k \dots y_{s-2}\bar{\mathbf{v}}y_s \dots),\\ \beta_b &= (x_0 \dots x_{i-2}\bar{\mathbf{u}}x_i \dots x_{j-2}\mathbf{u}x_j \dots, y_0 \dots y_{k-2}\bar{\mathbf{v}}y_k \dots y_{s-2}\mathbf{v}y_s \dots),\end{aligned}$$

where $\bar{u} = 1-u, \bar{v} = 1-v$ for $u, v \in \{0, 1\}$. Set $m_0 = u, m_1 = \bar{u}$ as the messages. The challenge ciphertext is $C = \mathsf{E}(pk, m_c)$ where $c \in \{0, 1\}$. If $u = v$, we set the encrypted identity representation (C_x, C_y) as

$$(\mathsf{E}(pk, x_0), \dots, \mathsf{E}(pk, x_{i-2}), \mathsf{E}(\mathbf{pk}, \mathbf{m_c}), \mathsf{E}(pk, x_i), \dots, \mathsf{E}(pk, x_{j-2}), \mathsf{E}(\mathbf{pk}, \bar{\mathbf{m}}_\mathbf{c}), \mathsf{E}(pk, x_j), \dots,$$
$$\mathsf{E}(pk, y_0), \dots, \mathsf{E}(pk, y_{k-2}), \mathsf{E}(\mathbf{pk}, \mathbf{m_c}), \mathsf{E}(pk, y_k), \dots, \mathsf{E}(pk, y_{s-2}), \mathsf{E}(\mathbf{pk}, \bar{\mathbf{m}}_\mathbf{c}), \mathsf{E}(pk, y_s), \dots).$$

Otherwise, we set the encrypted identity (C_x, C_y) representation as

$$(\mathsf{E}(pk, x_0), \dots, \mathsf{E}(pk, x_{i-2}), \mathsf{E}(\mathbf{pk}, \mathbf{m_c}), \mathsf{E}(pk, x_i), \dots, \mathsf{E}(pk, x_{j-2}), \mathsf{E}(\mathbf{pk}, \bar{\mathbf{m}}_\mathbf{c}), \mathsf{E}(pk, x_j), \dots,$$
$$\mathsf{E}(pk, y_0), \dots, \mathsf{E}(pk, y_{k-2}), \mathsf{E}(\mathbf{pk}, \bar{\mathbf{m}}_\mathbf{c}), \mathsf{E}(pk, y_k), \dots, \mathsf{E}(pk, y_{s-2}), \mathsf{E}(\mathbf{pk}, \mathbf{m_c}), \mathsf{E}(pk, y_s), \dots),$$

where $\mathsf{E}(pk, \bar{m}_c) = \mathsf{E}(pk, 1)/\mathsf{E}(pk, m_c)$. If the server answers that the user is $\mathcal{U}_a$, we output 0 as the answer to the semantic security game. If the server answers that the user is $\mathcal{U}_b$, we output 1 as the answer to the semantic security game. If $c = 0$, $C = \mathsf{E}(pk, m_0) = \mathsf{E}(pk, u)$. Therefore, (C_x, C_y) is the encrypted identity representation of β_a. If $c = 1$, $C = \mathsf{E}(pk, m_1) = \mathsf{E}(pk, \bar{u})$. (C_x, C_y) is the encrypted identity representation of β_b. Since the server has probability 1/2 + ϵ to give a correct answer, we have probability 1/2 + ϵ to output c correctly. The advantage of breaking the semantic security is thus ϵ. ❑

4.2 Efficiency

The communication complexity of our protocol is dominated by $O(\sqrt{N})$ ciphertext size. Each encryption needs 2 modular exponentiations. The computation cost of a user is $O(\sqrt{N})$ encryptions and 4 modular exponentiations in G_1. The computation cost of the server is $N + 3$ modular exponentiations in G_1 and N pairing computation. The round complexity of our protocol is 4.

We compare our result with the mutual authenticated version of [14] in Table 1.

5. DISCUSSION AND CONCLUSION

Our protocol is secure against the off-line dictionary attack. For the on-line dictionary attack, the general solution is to limit the number of trials from a particular source, such as IP.

In this paper, we propose an anonymous password based key exchange protocol with sub-linear communication complexity, which is better than the previous result. It is interesting that how to achieve anonymity against the malicious server or how to achieve AKE security without the random oracle.

A Proof of Theorem 2

Proof: We define a sequence of games from the real AKE game $\mathcal{G}_0$ to the final game $\mathcal{G}_5$. To bound the probability of the adversary winning in the real AKE game, we bound the probability in one of those games at first (specifically, game $\mathcal{G}_3$). Then we analyze the differences between successive games.

Game $\mathcal{G}_0$: This is the real protocol in the random oracle model. We consider the following three events"

- S_0 (for semantic security), which occurs if the adversary can correctly guess the bit used in the Test-query.
- $\mathcal{S}A_0$ (for $\mathcal{S}$-authentication), which occurs if $\mathcal{U}$ accepts for a fake session.
- $\mathcal{U}A_0$ (for $\mathcal{U}$-authentication), which occurs if $\mathcal{S}$ accepts for a fake session.

$$\mathsf{Adv}^{\mathrm{ake}}(\mathcal{A}) = 2\Pr[S_0] - 1$$
$$\mathsf{Succ}^{\mathrm{S\text{-}auth}}(\mathcal{A}) = \Pr[\mathcal{S}A_0]$$
$$\mathsf{Succ}^{\mathrm{U\text{-}auth}}(\mathcal{A}) = \Pr[\mathcal{U}A_0]$$

In each game $\mathcal{G}_i$, we analysis the events $\mathcal{S}A_i$, $\mathcal{U}A_i$, and the restricted event $RS_i = S_i \wedge\neg\, \mathcal{S}A_i \wedge\neg\, \mathcal{U}A_i$.

Game $\mathcal{G}_1$: In this game, we simulate the hash oracles (ψ, H_0, H_1, H_2) by maintaining lists Λ_ψ, Λ_H, Λ'_H (where the hash functions H'_0, H'_1, H'_2 will appear in the Game $\mathcal{G}_3$). We also simulate all real players would do for all queries (the Send-queries, the Execute-queries, the Reveal-queries and the Test-queries). From this simulation, the game is perfectly indistinguishable from the real one.

Game $\mathcal{G}_2$: We cancel some games:

- collisions on the partial transcripts $((pk, C_x, C_y, X), (\mathcal{S}, B, Z))$. There is at least one honest party involved in the transcription. Thus one of X or (B, Z) is truly uniformly distributed.
- collisions on the output of ψ.

We denote the above events as $Coll_2$ and bound the probability by the birthday paradox:

$$\Pr[Coll_2] \le \frac{(q_e + q_s)^2}{2n} + \frac{(q_g)^2}{2n}.$$

Game $\mathcal{G}_3$: We compute the session key SK and the authenticators $Auth_\mathcal{S}$, $Auth_\mathcal{U}$ using the private oracles H'_0, H'_1, H'_2 respectively.

$$Auth_\mathcal{S} = H'_1(pk, C_x, C_y, X, \mathcal{S}, B, Z)$$
$$Auth_\mathcal{U} = H'_2(pk, C_x, C_y, X, \mathcal{S}, B, Z)$$
$$SK = H'_0(pk, C_x, C_y, X, \mathcal{S}, B, Z)$$

Since the computations of SK and authenticators are without $K_\mathcal{S}$ and $K_\mathcal{U}$, and the password is not used anymore, we can simplify the rules as follows:

- In step 2, $\mathcal{S}$ does the follows:
  - randomly select $r, \tilde{r} \in Z_n$ and compute $B = e(g, g)^r e(g, h)^{\tilde{r}}$,
  - randomly select $r_z \in Z_n$ and compute $Z = e(g, g)^{r_z}$.

The games $\mathcal{G}_3$ and $\mathcal{G}_2$ are indistinguishable unless some specific hash queries are asked, denoted as $AskH_3$, which can be split as the following events:

- $AskH1_3$: $\mathcal{A}$ queries $H_1(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K_\mathcal{S})$ or $H_1(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K_\mathcal{U})$ for some execute transcript $((pk, C_x, C_y, X, Auth_\mathcal{U})$ $(\mathcal{S}, B, Z, Auth_\mathcal{S}))$.
- $AskH2w1_3$: $\mathcal{A}$ queries $H_2(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K_\mathcal{S})$ or $H_2(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K_\mathcal{U})$ for some execute transcript $((pk, C_x, C_y, X, Auth_\mathcal{U})$ $(\mathcal{S}, B, Z, Auth_\mathcal{S}))$, where some user has accepted, but the event $AskH1_3$ didn't happen.
- $AskH0w12_3$: $\mathcal{A}$ queries $H_0(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K_\mathcal{S})$ or $H_0(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K_\mathcal{U})$ for some execute transcript $((pk, C_x, C_y, X, Auth_\mathcal{U})$ $(\mathcal{S}, B, Z, Auth_\mathcal{S}))$, where some user has accepted and the server has accepted in some session, but the events $AskH1_3$ and $AskH2w1_3$ didn't happen.

The authenticators are computed by the random oracles that are private to the adversary, then they can not be guessed by the adversary better than randomly guessing. In the similar way, the session key *SK* and a random value can not be distinguished by the adversary better than random guessing. That is

$$\Pr[\mathcal{SA}_3] \le \frac{q_s}{2^l},\ \Pr[\mathcal{UA}_3] \le \frac{q_s}{2^l},\ \Pr[RS_3] = \frac{1}{2}.$$

To bound Pr[$AskH_3$], We analyze the probability of the event $AskH_i$ happening in other game $Game_i$ and the differences of Pr[$AskH_i$] between those games.

Game $\mathcal{G}_4$: We embed a random Diffie-Hellman instance (P, Q) into the game. P is ${g_0}^{r_p}$ for some $r_p \in Z_n$, and Q is ${g_0}^{r_q}$ for some $r_q \in Z_n$, where $g_0 = e(g, g)$ is a generator of G_1. The computational Diffie-Hellman problem is to compute the value ${g_0}^{r_p r_q}$, denoted as $CDH_{g_0,G_1}(P, Q)$.

- In step 1, $\mathcal{U}$ computes X by randomly selecting $r_x \in Z_n$ and $X = P^{r_x}$.
- We modify the oracle ψ:

For queries to ψ, randomly select $r_g \in Z_n$ and output Q^{r_g}.
We excluded the cases $X = 1$ and the output of ψ is 1. Therefore,

$$\left|\Pr[AskH_4] - \Pr[AskH_3]\right| \le \frac{q_g}{n} + \frac{q_s + q_e}{n}.$$

Game $\mathcal{G}_5$: We cancel some games in which the following conditions hold. For the same partial transcript ((pk, C_x, C_y, X), ($\mathcal{S}$, B, Z)), there exist two distinct elements R_1, R_2, such that (C_x, C_y, X, $\mathcal{S}$, B, Z, $\psi(R_1)$, K_1), (C_x, C_y, X, $\mathcal{S}$, B, Z, $\psi(R_2)$, K_2) are both in Λ_H. We denote this event as $CollH_5$ and therefore, |Pr[$AskH_4$] − Pr[$AskH_5$]| ≤ Pr[$CollH_5$].

Lemma 1 If for some partial transcript (pk, C_x, C_y, X, $\mathcal{S}$, B, Z), there are two different π_1, π_2 such that the following conditions hold

- $R_1 = \dfrac{B^{q_1}}{g_0^{\pi_1}}$, $R_2 = \dfrac{B^{q_1}}{g_0^{\pi_2}}$.
- (pk, C_x, C_y, X, $\mathcal{S}$, B, Z, $\psi(R_i)$, K_i) are both in Λ_H for i = 1, 2.
- $K_i = CDH_{g_0,G_1}(X, Z/\psi(R_i))$ for i = 1, 2.

Then one can solve the computational Diffie-Hellman problem:

$$\Pr[CollH_5] \le g_h^2 \times Succ_{g_0,G_1}^{cdh}(t + \tau_e). \qquad ❑$$

Proof: Assume that there exist such π_1, π_2, then $K_1 = CDH_{g_0,G_1}(X, Z/\psi(R_1))$, $K_2 = CDH_{g_0,G_1}$ (X, $Z/\psi(R_2)$) where $\psi(R_1) = Q^{r_a}$, $\psi(R_2) = Q^{r_b}$. Note that

$$K_1 = CDH_{g_0,G_1}(X, Z/\psi(R_1)),\ K_2 = CDH_{g_0,G_1}(X, Z/\psi(R_2)).$$

As a result, we can have

$$CDH_{g_0,G_1}(P,Q) = (\frac{K_1}{K_2})^u$$

where $u = (r_x(r_b - r_a)^{-1}$. By random guessing two hash queries among q_h queries,

$$\Pr[CollH_5] \le g_h^2 \times Succ_{g_0,G_1}^{cdh}(t + \tau_e). \qquad \square$$

To analysis $\Pr[AskH_5]$, we divide $AskH1_5$ into three subcases.

- $AskH1\text{-}passive_5$: In this case, both parties are simulated. Thus we have the following lemma:

Lemma 2 If for some passive partial transcript $(pk, C_x, C_y, X, \mathcal{S}, B, Z)$, there exists $(pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K)$, in Λ_H, one can solve the computational Diffie-Hellman problem:

$$\Pr[AskH1 - passive_5] \le g_h \times Succ_{g_0,G_1}^{cdh}(t + 2\tau_e).$$

Proof: Since both parties are simulated, $X = P^{r_x}$, $Z = g_0{}^{r_z}$, $\psi(R) = Q^{r_g}$. Thus one can have

$$K = CDH_{g_0,G_1}(X, Z/\psi(R)) = P^{r_z}/CDH_{g_0,G_1}(P, Q)^{r_x r_g}$$

and compute

$$CDH_{g_0,G_1}(P, Q) = (P^{r_z}/K)^u$$

where $u = (r_x r_g)^{-1}$. $\square$

- $AskH1\text{-}WithS_5$: This event models the attack that an adversary wants to impersonate $\mathcal{U}$ to $\mathcal{S}$. To generate the authenticator, the adversary ask H_1 for some $((pk, C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K)$. Without any collision on ψ's output, R is determined uniquely. Since $B^{q_1} = R \times g_0^{\pi\, q_1}$, π is determined uniquely. Thus each authenticator sent by the adversary can only contain one password information.

$$\Pr[AskH1\text{-}WithS_5] \le \mathcal{D}_\pi(q_s)$$

- $AskH1\text{-}WithU_5$: This event models the attack that an adversary wants to impersonate $\mathcal{S}$ to $\mathcal{U}$. To generate the authenticator, the adversary ask H_2 for some $((C_x, C_y, X, \mathcal{S}, B, Z, \psi(R), K)$. Each authenticator sent by the adversary can only contain one password information.

$$\Pr[AskH1\text{-}WithU_5] \le \mathcal{D}_\pi(q_s)$$

Now, we have

$$\Pr[AskH1_5] \leq g_h \times Succ^{cdh}_{g_0,G_1}(t + 2\tau_e) + 3\mathcal{D}_\pi(q_s).$$

For the event $AskH2w1_5$, we can analysis the two sub-events $AskH2w1\text{-}passive_5$ and $AskH2w1\text{-}With\ U_5$, since $\mathcal{U}$ has been authenticated by $Auth_\mathcal{U}$. The last event is $AskH0w12_5$, which is $AskH0w12\text{-}passive_5$ since $\mathcal{U}$ and $\mathcal{S}$ have been authenticated by $Auth_\mathcal{U}$ and $Auth_\mathcal{S}$ respectively. As a consequence, we have

$$\Pr[AskH_5] \leq 3g_h \times Succ^{cdh}_{g_0,G_1}(t + 2\tau_e) + 3\mathcal{D}_\pi(q_s).$$

Combining all the above equations, one can get the announced result.

Since the adversary is t time-bounded, the number of queries is polynomial bounded. Simulator simulates for hash queries, Send-queries, and Execute-queries. Answering the Execute-queries is the most time-consuming among all kind of queries since the simulator needs to simulate the whole execution of the protocol. However, it still takes polynomial time. Therefore, the simulation is in polynomial time. ❑

REFERENCES

1. E. Modiano, "An adaptive algorithm for optimizing the packet size used in wireless ARQ protocols," *Wireless Networks*, Vol. 5, 1999, pp. 279-286.
2. M. Abdalla and D. Pointcheval, "Simple password-based encrypted key exchange protocols," in *Proceedings of the 2005 Conference on Topics in Cryptology: The Cryptographer's Track at RSA*, 2000, pp. 191-208.
3. M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," in *Proceedings of Advances in Cryptology*, 2000, pp. 139-155.
4. S. M. Bellovin and M. Merritt, "Encrypted key exchange: Password-based protocol secure against dictionary attacks," in *Proceedings of the IEEE Symposium on Research in Security and Privacy*, 1992, pp. 72-84.
5. S. M. Bellovin and M. Merritt, "Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise," in *Proceedings of the 1st ACM Conference on Computer and Communications Security*, 1993, pp. 325-341.
6. D. Boneh, E. J. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in *Proceedings of the 2nd Theory of Cryptography Conference*, 2005, pp. 325-341.
7. V. Boyko, P. D. MacKenzie, and S. Patel, "Provably secure password authenticated key exchange using diffie-hellman," in *Proceedings of Advances in Cryptology*, 2000, pp. 156-170.
8. E. Bresson, O. Chevassut, and D. Pointcheval, "New security results on encrypted key exchange," in *Proceedings of the 7th International Workshop on Theory and Practice in Public Key Cryptography*, 2004, pp. 145-158.
9. J. Camenisch and M. Michels, "Proving in zero-knowledge that a number is the product of two safe primes," in *Proceedings of Advances in Cryptology*, 1999, pp.

107-122.
10. C. K. Chu and W. G. Tzeng, "Efficient *k*-out-of-*n* oblivious transfer schemes with adaptive and non-adaptive queries," in *Proceedings of the 8th International Workshop on Theory and Practice in Public Key Cryptography*, 2005, pp. 172-183.
11. J. Groth, R. Ostrovsky, and A. Sahai, "Perfect non-interactive zero knowledge for np," in *Proceedings of Advances in Cryptology*, 2006, pp. 339-358.
12. J. S. Hwu, R. J. Chen, and Y. B. Lin, "An efficient identity-based cryptosystem for end-to-end mobile security," *IEEE Transactions on Wireless Communications*, Vol. 5, 2006, pp. 2586-2593.
13. S. Jiang and G. Gong, "Password based key exchange with mutual authentication," in *Proceedings of 11th International Workshop on Selected Areas in Cryptography*, 2004, pp. 267-279.
14. V. Miller, "The weil pairing, and its efficient calculation," *Journal of Cryptology*, Vol. 17, 2004, pp. 235-261.
15. D. Q. Viet, A. Yamamura, and H. Tanaka, "Anonymous password based authenticated key exchange," in *Proceedings of the 6th International Conference on Cryptography in India*, 2005, pp. 244-257.

Hsiao-Ying Lin (林孝盈) is currently a Ph.D. student in Department of Computer Science, National Chiao Tung University, Taiwan. She received her B.S. degree in Computer Science and Engineering from National Sun Yat-Sen University, 2003; and M.S. degree in Computer Science from National Chiao Tung University, 2005. Her current research interests include applied cryptography and information security.

Wen-Guey Tzeng (曾文貴) received his B.S. degree in Computer Science and Information Engineering from National Taiwan University, Taiwan, 1985; and M.S. and Ph.D. degrees in Computer Science from the State University of New York at Stony Brook, USA, in 1987 and 1991, respectively. He joined the Department of Computer and Information Science (now, Department of Computer Science), National Chiao Tung University, Taiwan, in 1991 and works there till now. Dr. Tzeng's current research interests include cryptology, information security and network security.