Identity-Based and Inter-Domain Password Authenticated Key Exchange for Lightweight Clients

Ford Long Wong
Security Group & DTG, Computer Laboratory
University of Cambridge
fw242@cam.ac.uk

Hoon Wei Lim
Information Security Group
Royal Holloway, University of London
h.lim@rhul.ac.uk

Abstract

We propose a four-party password authenticated inter-domain key exchange protocol which makes use of properties of identity-based cryptography and secret public keys. Being password-based and certificate-free, our protocol is lightweight and is suited to lightweight computing environments, such as pervasive computing. Apart from resistance against offline and active attacks, our protocol additionally provides perfect forward secrecy. We provide heuristic analysis of various security properties. Performance comparisons against other related protocols show that our protocol is efficient.

Keywords: Inter-domain authentication, identity-based cryptography, password, secret public key.

1 Introduction

The classic mutually authenticated key exchange between two communicating parties aims to confirm that they each know who the other party is, and that they share a session key at the end of a key exchange protocol. In this paper, we study the case of inter-domain authenticated key exchange between lightweight devices, such as pervasive computing devices with user input interfaces. Succinctly, the goal of an inter-domain authenticated key exchange protocol is to address cross-domain authentication and key establishment between two users registered under two distinct authentication servers.

For example, let's suppose that each hospital has its own authentication domain, under which all its staff are registered. A medical consultant (i.e. entity A), working in Hospital X, visits Hospital Y carrying a PDA. He speaks to a surgeon (i.e. entity B) in Y, on his way from an operating theatre, who is also carrying a PDA, and they decide they need to exchange some clinical information quickly. We assume a path exists for A to access his own authentication server S_A through Y's wireless network. The entities have not met *a priori*. They do not know whether the other is accredited with an authentication server which their own authentication server recognizes. A needs to initiate a protocol, which when completed successfully, would indicate to A that B is properly accredited by a password to his authentication server S_B, and that S_B is in fact a server that is recognized and trusted by S_A. Currently, this type of key exchange appears to be under-researched.

Related Work. The recent work of Yeh and Sun [12] reminds us of the relevance of inter-domain authentication protocols. They proposed two four-party password-based authenticated key establishment protocols, which are based on key transport and key agreement techniques, respectively. While the proposals attempt to address issues of inter-domain authentication, they suffer from some limitations. Firstly, their proposals were based on the assumption that the users have access to their respective authentication servers' public keys. This implies the need for a public key infrastructure (PKI) to distribute and verify the servers' public keys for the clients. This is a significant requirement for standard password-based authentication protocols which may be acceptable for certain networked applications, but less desirable for lightweight computing environments. Secondly, Yeh and Sun claimed that their protocols satisfy the property of forward secrecy. However, they have not taken the authentication servers' long-term private keys into consideration. The exposure of an authentication server's long-term private key could trivially reveal its users' passwords, and for their KTAP protocol (derived from the key transport technique), even past session keys.

Kerberos [10] is another solution to inter-domain password-based authentication. It is known for its efficiency since it employs symmetric cryptographic techniques. However, purely symmetric key management for inter-domain secure communications is non-trivial and not scalable. In [13], a PKI-supported initial authentication

in Kerberos was proposed to improve the scalability of Kerberos. However, deployment of PKI at the client side within lightweight environments is, again, not desirable.

In this paper, we investigate the potential roles of identity-based cryptography (IBC) [2, 11] which can be exploited to overcome the aforementioned issues. In particular, we extend the recent proposal of identity-based secret public keys[1] (ID-SPK) by Lim and Paterson [9] to devise an identity-based four-party password authenticated key exchange (ID-4-PAKE) protocol. The concept of identity-based secret public keys, which was descended from Gong *et al.*'s work [5] on secret public keys, combines the use of passwords and identifiers in the IBC setting. Hence, an identity-based secret public key can only be constructed by a party who knows the associated password. Our contributions can be summarized as follows:

- **Functionality**: We present an identity-cum-password-based inter-domain key exchange protocol. This is a novel application of IBC. It requires only minimal communication bandwidth, because IBC is certificate-free, and small key sizes can be used.
- **Technical Novelty**: The deployment of an identity-based cryptographic scheme *generally* requires distribution of system parameters, and thus an infrastructure such as a PKI at the client side, is required for the users to authenticate these parameters. We show that our protocol overcomes this requirement, i.e. a client-side PKI is not required in our protocol. We achieve this by masking authentication servers' ephemeral Diffie-Hellman (DH) values with user passwords in a protocol run; these DH values are then extracted by clients and used to construct identity-based secret public keys (ID-SPKs) [9]. The messages encrypted using these identity-based secret public keys can be decrypted by the intended authentication servers only if they hold the correct user passwords. Due to this observation, the servers' public parameters and the ephemeral DH values need not be authenticated before use in our protocol setting.
- **Usability**: Our protocol requires users to remember only their respective passwords. Hence, it is PKI-free at the client end. It is convenient and user-friendly because our clients do not have to obtain and verify public key certificates of their respective authentication servers.
- **Improved Security**: Unlike the Yeh-Sun proposals which do not provide the property of forward secrecy, we show that it is possible to retain such forward secrecy in an inter-domain authenticated key exchange protocol. In our protocol, the compromise of a server's long-term secret does not reveal the user password nor past session keys. We also provide heuristic security analyses to demonstrate that our protocol possesses various standard security properties.

[1] A secret public key is no different from a conventional public key except that it is only known among the intended parties.

Organisation. The paper is organized as follows. In Section 2, we review the basic concepts of identity-based cryptography. In Section 3, we describe the architecture required to support our protocol, which is then presented in Section 4. In Section 5, we give some security analyses of our proposal. In Section 6, we compare our proposal to related protocols.

2 Identity-Based Cryptography

Identity-based cryptography (IBC) was first introduced by Shamir [11]. Recently, there has been an increased intensity in research on IBC. This was mainly due to the seminal discovery of a practical and secure identity-based encryption (IBE) scheme by Boneh and Franklin [2]. Their scheme uses pairings over elliptic curves. In the identity-based setting, a user's public key can be constructed based on an identifier, such as the user's identity or email address, and the matching private key can be obtained from a trusted third party called the private key generator (PKG).

In what follows, we provide more details of pairings. We also sketch the Boneh and Franklin IBE scheme of [2], which we will use in our proposal.

Pairings. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of order q for some large prime q, where $\mathbb{G}_1$ is an additive group and $\mathbb{G}_2$ denotes a related multiplicative group. A pairing in the context of IBC is a function $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ with the following properties.

- *Bilinear*: Given $P, Q, R \in \mathbb{G}_1$, we have

$$\hat{e}(P, Q + R) = \hat{e}(P, Q) \cdot \hat{e}(P, R) \text{ and}$$

$$\hat{e}(P + Q, R) = \hat{e}(P, R) \cdot \hat{e}(Q, R).$$

  Hence, for any $a, b \in \mathbb{Z}_q^*$, $\hat{e}(aP, bQ) = \hat{e}(abP, Q) = \hat{e}(P, abQ) = \hat{e}(aP, Q)^b = \hat{e}(P, Q)^{ab}$.
- *Non-degenerate*: There exists a $P \in \mathbb{G}_1$ such that $\hat{e}(P, P) \neq 1$.
- *Computable*: If $P, Q \in \mathbb{G}_1$, $\hat{e}(P, Q)$ can be efficiently computed.

For any $a \in \mathbb{Z}_q^*$ and $P \in \mathbb{G}_1$, we write aP as the scalar multiplication (or point multiplication) of group element P by integer a. Typically, $\mathbb{G}_1$ is obtained as a subgroup of

the group of points on a suitable elliptic curve over a finite field, $\mathbb{G}_2$ is obtained from a related finite field, and $\hat{e}$ obtained from the Weil or Tate pairing on the curve. Note that a scalar multiplication aP can be computed very efficiently. However, the problem of finding a when given aP is believed to be intractable, when the curve is appropriately chosen. This problem is known as the elliptic curve discrete logarithm (ECDL) problem.

The Boneh-Franklin IBE Scheme. The following four algorithms underpin Boneh and Franklin's IBE scheme [2].

SETUP**:** Given a security parameter $k \in \mathbb{Z}^+$, the algorithm:

1. specifies two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of order q, and a pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$;
2. chooses an arbitrary generator $P \in \mathbb{G}_1$;
3. defines four cryptographic hash functions, $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$, $H_2 : \mathbb{G}_2^* \rightarrow \{0,1\}^n$ for some n, $H_3 : \{0,1\}^n \times \{0,1\}^n \rightarrow \mathbb{Z}_q^*$, and $H_4 : \{0,1\}^n \rightarrow \{0,1\}^n$; and
4. picks a master secret $s \in \mathbb{Z}_q^*$ at random and computes the matching public component as sP.

The system or public parameters are $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, sP, H_1, H_2, H_3, H_4 \rangle$.

EXTRACT**:** This algorithm extracts a private key $sH_1(\text{ID})$ when given an arbitrary identifier string $\text{ID} \in \{0,1\}^*$.

ENCRYPT**:** To encrypt a message $m \in \{0,1\}^n$ under an identifier ID, the public key used is $Q_{\text{ID}} = H_1(\text{ID})$. The algorithm selects a random $z \in \{0,1\}^n$ and sets $r = H_3(z, m)$. The resulting cipertext is then set to be $c = \langle U, V, W \rangle = \langle rP, z \oplus H_2(g^r), m \oplus H_4(z) \rangle$, where $g = \hat{e}(Q_{\text{ID}}, sP) \in \mathbb{G}_2$.

DECRYPT**:** To decrypt a ciphertext $c = \langle U, V, W \rangle$ encrypted using the identifier ID, the private key used is $sQ_{\text{ID}} \in \mathbb{G}_1^*$. If $U \notin \mathbb{G}_1^*$, reject the ciphertext. The plaintext m is then recovered by performing the following steps:

1. compute $V \oplus H_2(\hat{e}(sQ_{\text{ID}}, U)) = z$;
2. compute $W \oplus H_4(z) = m$; and
3. set $r = H_3(z, m)$, if $U \neq rP$, reject the ciphertext, otherwise accept m as the decryption of c.

The SETUP and EXTRACT algorithms are run by a PKG within a domain. As in all identity-based schemes and not just in the Boneh-Franklin IBE scheme, all the users within a domain are assumed to share the same system parameters, i.e. $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, sP, H_1, H_2, H_3, H_4 \rangle$. In the identity-based setting, each PKG must distribute its parameter set to its users *a priori*. While most of the components of these parameters can be fixed and made public, and thus require no further authenticity verification, there exists a component, sP, which is mathematically tied to the PKG's master secret s. The failure of authenticating a PKG's parameter set generally could allow a trivial man-in-the-middle attack. We will show that in our protocol, the server's public component does not need to be authenticated for resisting the man-in-the-middle attack.

3 Architecture

Here, we describe the architecture and trust hierarchy that we employ in our proposal. We assume that all the system parameters used in the Boneh-Franklin IBE scheme $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, H_1, H_2, H_3, H_4 \rangle$ except sP are fixed and bootstrapped in the system. All new users/devices are assumed to be initialized with these fixed parameters. This allows each authentication server to transmit only a server-specific value, i.e. sP, across the network (henceforth, we refer to a public component as a server-specific sP value). This represents a trade-off between savings in communication costs and lack of flexibility in supporting groups derived from different elliptic curves. The use of different curves and groups to achieve different levels of security is implementation-dependent, and thus will not be further discussed here.

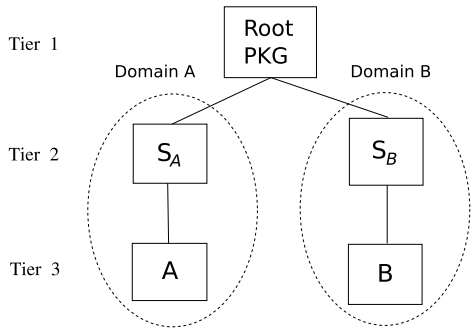


Figure 1. Architecture and trust hierarchy.

Our identity-based architecture consists of three tiers, as shown in Figure 1. We now briefly describe the key management aspect of our architecture.

- **Tier 1**: At this tier, there exists a root PKG which owns a public component s_0P, of which s_0 is the corresponding master secret. The root PKG issues daily private keys to authentication servers at tier 2 using the EXTRACT algorithm. These private keys correspond to public keys of the form $H_1(S_A \| \text{date})$ for authentication server S_A.
- **Tier 2**: An authenticated copy of the root PKG public component, s_0P, is made available to the authentication servers beforehand. If authenticity verification of the root PKG public component, and fine-grained revocation of the servers' public keys are required, then

an infrastructure, such as a PKI[2], would be required at the domain server tier.

Each domain server[3] holds a copy of the passwords of the users in its respective domain. The domain servers also act as the domain PKG, in that they own a master secret (s_A and s_B, respectively) which is used to extract decryption keys during a protocol run with their respective domain users. The associated server public components are s_AP and s_BP, respectively.

- **Tier 3**: At the bottom tier, each user holds a password which he shares with his domain server. We will explain how this is defined and derived in Section 4.

4 Protocol

In our identity-based setting, a user A holds a low-entropy secret, the password PW_A and her authentication server S_A holds the matching image $PW_{S_A}[A]$, as defined in [1]. In our protocol, we assume $PW_{S_A}[A] = PW_A$, although they may be different in actual protocol implementations. We then set the transformed password as $\pi_A = H_1(A\|S_A\|PW_A)$, where H_1 is a full-domain hash function from $\{0,1\}^*$ into $\mathbb{G}_1^*$ (as defined in Section 2). We use $\{\cdot\}_{\pi_A}$ to denote a password-based mask generation function [1] under a password π_A (henceforth, we refer to a password as a transformed password using a full-domain hash of the password). For instance, $\{aP\}_{\pi_A}$ denotes encrypting a Diffie-Hellman (DH) value aP with a password π_A, which in turn, implies calculating the addition of aP and π_A. To decrypt and recover aP, one can simply subtract π_A from $\{aP\}_{\pi_A}$.

We use $\hat{PK}$ and PK to represent a secret public key [9] and a standard public key, respectively. We use the notation $Enc_A(\cdot)$ to indicate asymmetric encryption with A's public key and based on the Boneh-Franklin IBE scheme.

Our identity-based four-party password authenticated key exchange (ID-4-PAKE) protocol, as depicted in Protocol 1, can be described as follows:

1. $A \rightarrow B : A, B, S_A, aP$
   To begin, A sends an initiating message to B. The message contains the identities of: (i) initiator, (ii) recipient, and (iii) initiator's authentication server. A also includes an ephemeral DH value aP, where $a \in \mathbb{Z}_q^*$ is a randomly selected secret value.
2. $B \rightarrow S_B : B, A, S_B, S_A, bP, aP$
   In step (2), upon receiving the initiating message from A, B randomly selects a secret value $b \in \mathbb{Z}_q^*$ and computes his DH value bP. B then forwards this value and the original message that he received from A to his authentication server S_B.
3. $S_B \rightarrow S_A : B, A, S_B, S_A, Enc_{S_A}(B, A, S_B, S_A, byP, n_B), aP$
   When S_B receives the message in step 2 from B, it identifies the intended communicating target (A) and the corresponding authentication server (S_A). Subsequently, S_B randomly chooses a secret value $y \in \mathbb{Z}_q^*$ and computes byP. S_B also chooses a nonce n_B. The values of byP and n_B, and the identities of A, B, S_A and S_B are then encrypted using a public key computed from a current date and S_A's identifier. The resulting ciphertext and other information, such as S_B's identity and A's chosen DH value aP, are sent to S_A.

   $S_B \rightarrow B : B, A, S_B, S_A, \{yP + s_BP\}_{\pi_B}, s_BP$
   In parallel with[4] the previous message from S_B to S_A, S_B computes its DH value yP which is then sent to B along with S_B's public component s_BP. Note that yP is added to s_BP, and encrypted under B's password π_B because the DH value will be used later for both S_B and B to authenticate each other. The rationale for adding yP and s_BP before their sum is encrypted using π_B is to resist active insider attackers; this will become clearer in Section 5.
4. $S_A \rightarrow S_B : Enc_{S_B}(A, B, S_A, S_B, axP, byP, n_A, n_B)$
   As with what S_B did in the previous step, S_A randomly selects a secret value $x \in \mathbb{Z}_q^*$, and then computes a composite DH value axP. S_A also selects a nonce n_A. The message $(A, B, S_A, S_B, axP, byP, n_A, n_B)$, encrypted under S_B's daily public key, is forwarded to S_B. Note that S_A includes the DH value byP and the nonce n_B, in the message to authenticate itself to S_B.

   $S_A \rightarrow A : A, B, S_A, S_B, \{xP + s_AP\}_{\pi_A}, s_AP$
   At the same time, S_A computes its DH value xP. The value xP is added to s_AP, and transmitted to A encrypted with A's password π_A. Other information such as S_B's identity and S_A's public component s_AP is also included in the transmission.

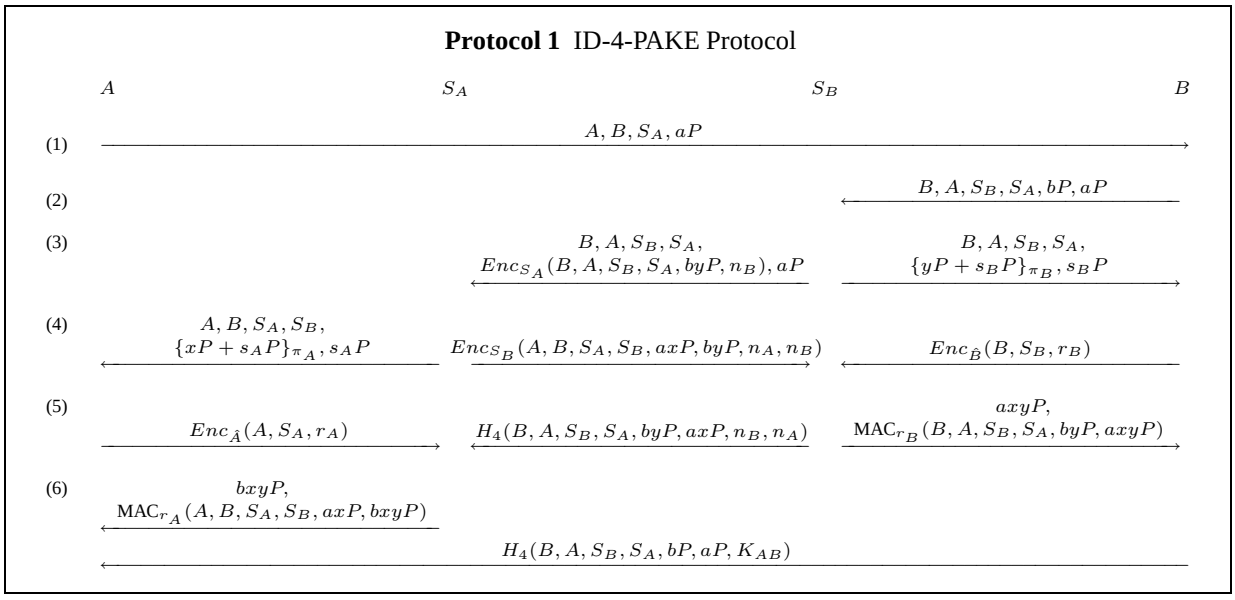
   $B \rightarrow S_B : Enc_{\hat{B}}(B, S_B, r_B)$
   B recovers yP using his password and by subtracting s_BP, and computes the composite DH value byP, which in turn is used to calculate a secret public key $\hat{PK}_B = H_1(B\|A\|\pi_B\|S_B\|S_A\|byP)$. This secret public key is then used to encrypt the identities of B and S_B, and a chosen random nonce r_B, and produce a ciphertext which could only be decrypted by a party who can extract the matching private key of $\hat{PK}_B$.
5. $A \rightarrow S_A : Enc_{\hat{A}}(A, S_A, r_A)$
   In this step, A encrypts a message that contains the identities of A and S_A, and a fresh random number r_A, with a secret public key $\hat{PK}_A = H_1(A\|B\|\pi_A\|S_A\|S_B\|axP)$. Note that $\hat{PK}_A$ can be computed by A only after she has successfully recovered xP obtained from S_A.

   $S_B \rightarrow S_A : H_4(S_B, S_A, byP, axP, n_B, n_A)$
   This hash value is generated by S_B to authenticate itself to

[2] It is worth noting that standard revocation techniques such as CRLs and OCSP can be adopted in the identity-based setting.

[3] We will use 'domain servers' and 'authentication servers' interchangeably.

[4] It makes sense that once y has been chosen, S_B can produce and send the relevant messages to S_A and B simultaneously.



S_A by proving to S_A that it has recovered the DH value axP and the nonce n_A successfully.

$S_B \rightarrow B : axyP, \text{MAC}_{r_B}(B, A, S_B, S_A, byP, axyP)$
Here, S_B decrypts the ciphertext from S_A in step (4) and recovers axP. It then calculates a composite DH value $axyP$. Additionally, S_B generates a MAC value by taking as input r_B and the message $(B, A, S_B, S_A, byP, axyP)$. The $axyP$ value and the MAC value would be sent to B.

6. $S_A \rightarrow A : bxyP, \text{MAC}_{r_A}(A, B, S_A, S_B, axP, bxyP)$
In the final step, analogous to the message from S_B to B in the previous step, S_A computes the relevant composite DH value $bxyP$. The value of $bxyP$ and a MAC value derived from the relevant information, as specified above, are transmitted to A. The session key $K_{AB} = F(A, B, S_A, S_B, abxyP)$ is shared between A and B, where F is a key derivation function.

$B \rightarrow A : H_4(B, A, S_B, S_A, bP, aP, K_{AB})$
The above hash value is computed by B and sent to A to provide key confirmation. This signifies the completion of a successful run of Protocol 1.

5 Security Analysis

Mutual Authentication. In Protocol 1, each party contributes a DH component for the generation of a session key K_{AB}. The DH values chosen by the servers, xP and yP, are added to the respective servers' public components, s_AP and s_BP, and encrypted under the users' passwords, π_A and π_B, respectively. If S_A can successfully decrypt the ciphertext $Enc_{\hat{A}}(A, S_A, r_A)$ such that the identities of A and S_A are revealed in the resulting plaintext, A is authenticated to S_A. This is because A can only construct the correct $\hat{PK}_A = H_1(A\|B\|\pi_A\|S_A\|S_B\|axP)$ if she could recover the right xP from S_A using her password π_A, and thus generate the proper ciphertext for S_A.

On the other hand, S_A is authenticated to A if A can derive the same MAC value as what she received from S_A. This indicates that S_A has successfully extracted the matching private key of $\hat{PK}_A$ using its master secret s_A and subsequently recovered r_A chosen by A.

In a similar fashion between A and S_A, B and S_B authenticate each other using similar techniques.

The mutual authentication between S_A and S_B is straightforward. In step (3), S_B sends $Enc_{S_A}(B, A, S_B, S_A, byP, n_B)$ to S_A, encrypted under $PK_{S_A} = H_1(S_A\|date)$. The corresponding decryption private key has been obtained by S_A from the Root PKG at the start of each day. S_A decrypts the contents and recovers byP and n_B, which it would then encrypt together with axP and n_A, and send to S_B in step (4). If S_B recovers byP and n_B successfully by decrypting the message, S_A is authenticated to S_B. In a similar way, when S_A receives the hash value from S_B in step (5) and is able to compute the same hash value, it proves that S_B has decrypted the message from S_A, and S_B is authenticated to S_A.

We remark that the last message in step (6) from B to A is essential to confirm that B has authenticated himself to S_B and that he has calculated the same session key as A. This is because B would only receive the value of $axyP$ from S_B after he is authenticated to S_B, which will enable him to calculate the session key. As for A, she would receive the value of $bxyP$ from S_A after she has been authenticated to S_A. This in turn allows A to calculate the same session key and verify B's key confirmation message.

We remark that a client is clearly unable to mount a

successful insider attack to break the mutual authentication between two servers, as it does not have any server's decryption key, and thus cannot recover either of the nonces, n_A and n_B.

Offline Guessing. An adversary E cannot deduce any useful information by attempting to decrypt $\{xP+s_AP\}_{\pi_A}$ (resp. $\{yP + s_BP\}_{\pi_B}$) with a guessed password π'_A (resp. π'_B) and then subtract the resulting decryption by s_AP (resp. s_BP). This is because the use of any candidate password will result in a random point in $\mathbb{G}_1$. Similarly, since the Boneh-Franklin IBE scheme is probabilistic and secure against adaptive chosen ciphertext attacks (IND-ID-CCA) [2], E cannot learn any useful information from the ciphertext produced.

Active Attacks and Online Guessing. We observe that even though the servers' public components s_AP and s_BP are sent in the clear and unauthenticated, E cannot mount man-in-the-middle attacks by impersonating S_A or S_B. Suppose E tries to impersonate S_A by replacing the message $(A, B, S_A, S_B, \{xP + s_AP\}_{\pi_A}, s_AP)$ with $(A, B, S_A, S_B, \{x'P + s'_AP\}_{\pi'_A}, s'_AP)$, of which the master secret s'_A and the value $x'P$ are known to E, and π'_A is a guessed password from E's password dictionary. However, E cannot predict, in polynomial time, $\hat{PK}_A$ that A computes and thus extract the corresponding private key. The reason for this is that, assuming A recovers a DH value $x''P$ with the correct password π_A, the only way for E to correctly predict the value x'' (in order to compute $ax''P$) is to solve the ECDL problem.

Forward Secrecy. Based on similar reasoning as the previous, even if S_A's master secret (s_A) is exposed, the probability of guessing the correct password (π_A) or recovering a past session key appears to be negligible. The adversary is unable to verify a password guess because decrypting by any guess will result in a random point in $\mathbb{G}_1$. In trying to calculate a past session key, the adversary is hindered by his lack of knowledge of any past ephemeral DH (secret) values, which contributed to the session key. Thus we conjecture that Protocol 1 has the property of forward secrecy.

Insider Attacks by Weakly Honest Servers. We define a *weakly honest server* as a server, say S_B, that attempts to either impersonate a user A from another domain to the user's domain server S_A, or to guess A's password. Three related attacks are conceivable. In the first attack, S_B attempts to guess the password π_A. S_B swaps the s_AP value which S_A sends to A in cleartext in step (4) with his own chosen s'_AP (where he knows s'_A). A will be now manipulated to calculate a secret public key of the form $\hat{PK}'_A = H_1(A\|B\|\pi_A\|S_A\|S_B\|a(xP + s_AP - s'_AP))$, and to encrypt (A, S_A, r_A) under this key. The question is whether or not S_B can extract the corresponding decryption key with high probability by brute-forcing the password, since he holds the master secret s'_A. S_B has received axP from S_A in step (4), and he also knows $(s_AP - s'_AP)$. But he remains unable to construct the secret public key $\hat{PK}'_A$ because he is unable to obtain the value of $a(s_AP-s'_AP)$ to calculate $(axP+a(s_AP-s'_AP))$. The difficulty of finding the value is equivalent to solving the ECDL problem.

In the second attack, S_B attempts to impersonate A to S_A. S_B allows the message which S_A sends to A in step (4) to proceed unmodified. S_B intercepts the message from A to S_A in step (5), and substitutes it with his own message. The question is whether or not S_B can construct the secret public key $\hat{PK}_A = H_1(A\|B\|\pi_A\|S_A\|S_B\|axP)$. S_B has negligible probability of doing this correctly in one online guessing attempt, because he does not know π_A and s_A.

In the third attack, S_B attempts to perform an offline dictionary attack against π_A after obtaining the transcript of a successful protocol run. S_B knows the value of axP. The question is whether or not S_B can find the correct π_A and calculate the correct $\hat{PK}_A$. Since S_B does not hold the master secret s_A, he cannot extract a corresponding decryption key to verify a guess. The high entropy of r_A ensures that the attack based on matching encryptions under guessed $\hat{PK}_A$ values with that transmitted by A is resisted.

Likwise, the protocol resists attacks in which S_A instead of S_B is assuming the weakly honest server role.

6 Comparison

We now compare our protocol to other related protocols.

PKI-Kerberos. Kerberos can be used to achieve cross-realm authentication (PKCROSS) by using public key cryptographic techniques. The messages exchanged between two Key Distribution Centres (KDCs) closely follow the PKINIT specification [13]. Cross-realm KDC-to-KDC authentication is analogous to our ID-4-PAKE. But if a KDC's private key is compromised, then past keying material is exposed; PKCROSS does not fulfil our definition of perfect forward secrecy.

Yeh-Sun KAAP/KTAP. In [12], two protocols were proposed – a key transport version (KTAP) and a key agreement version (KAAP); we are primarily concerned with the latter. Like the PKI-Kerberos, the Yeh-Sun proposals require the clients to obtain the servers' static public keys, and hence a PKI which interacts directly with the clients is required. In both protocols, if the private key of a server is compromised, then the password can be found easily. Thus, these do not fully satisfy the property of perfect forward secrecy.

Performance/Protocol	PKI-Kerberos	3-HK-PAKE	YS-KAAP	ID-4-PAKE
# message rounds	8	8	6	6
# asymmetric cryptographic operations	12	8	6	8
# asymmetric cryptographic operations per client	3	1	1	1

Table 1. Performance Comparison

Three 2-party Key Agreements. We consider a protocol derived from two 2-party password-authenticated key agreements using servers' static public keys, and one server-to-server 2-party key agreement. Surveying the literature on 2-party password-authenticated key agreement protocols [7], the most efficient ones have the minimum of three message rounds. Using the Halevi-Krawczyk scheme [6] (HK-PAKE), which is provably secure, as a building block and proceeding straightforwardly, we can derive a scheme which has a total message round number of 8, which also corresponds with that suggested by Yeh and Sun [12]. However, in HK-PAKE, if the server's long-term private key is compromised, then the user's password is exposed to dictionary attack.

It is possible to imagine a composed three 2-party key agreement protocol in which the two client-to-server key agreements are mediated by ephemeral public keys. This would confer the benefit of certificate-free operation at the client side. However, we note that the server-to-server key agreement would still need to rely on servers' authenticated public keys — implying an infrastructure would nevertheless be required at the server level. We conjecture that the composition would require at least the same number of message rounds as a straightforwardly composed 3-HK-PAKE protocol.

Performance Comparison. We consider the total number of message rounds for a protocol to run successfully and the incurred asymmetric cryptographic operations (i.e. encryption/decryption and signing/verification).

Table 1 compares the relative performance of various protocols. The table shows that our protocol is comparable to YS-KAAP and more efficient than the others. In addition, our protocol requires considerably minimal communication bandwidth because it is certificate-free. Moreover, users of our protocol do not rely on PKI when executing a protocol run, a significant advantage over the other protocols.

7 Conclusions and Future Work

We proposed a password-authenticated protocol for inter-domain key agreement using identity-based cryptography and the concept of secret public keys. We also presented heuristic security analysis of our protocol. Comparisons have been made with related protocols, revealing that our protocol is efficient and viable.

For future work, we will attempt to reduce the message complexity, and work on the formal security analysis of our protocol.

References

[1] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In B. Preneel, editor, *EUROCRYPT 2000*, pages 139–155. Springer-Verlag LNCS 1807, 2000.

[2] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *CRYPTO 2001*, pages 213–229. Springer-Verlag LNCS 2139, 2001.

[3] C. Boyd and A. Mathuria. *Protocols for Authentication and Key Establishment*. Springer-Verlag, Berlin, 2003.

[4] S.D. Galbraith. Supersingular curves in cryptography. In C. Boyd, editor, *ASIACRYPT 2001*, pages 495–513. Springer-Verlag LNCS 2248, 2001.

[5] L. Gong, T.M.A. Lomas, R.M. Needham and J.H. Saltzer. Protecting poorly chosen secrets from guessing attacks. *IEEE JSAC*, 11(5):648–656, 1993.

[6] S. Halevi and H. Krawczyk. Public-key cryptography and password protocols. *ACM TISSEC*, 2(3):25–60, 1999.

[7] IEEE. *IEEE P1363.2: Password-Based Public-Key Cryptography*, http://grouper.ieee.org/groups/1363/passwdPK/index.html.

[8] J. Kohl and C. Neuman. The Kerberos Network Authentication Service (V5). *IETF*, RFC 1510, Sep 1993.

[9] H.W. Lim and K.G. Paterson. Secret public key protocols revisited. In *Proceedings of the 14th International Workshop on Security Protocols 2006*, March 2006.

[10] B.C. Neuman and T. Ts'o. Kerberos: An authentication service for computer networks. *IEEE Communications*, 32(9):33–38, Sep 1994.

[11] A. Shamir. Identity-based cryptosystems and signature schemes. In G.R. Blakley and D. Chaum, editors, *CRYPTO'84*, pages 47–53. Springer-Verlag LNCS 196, 1985.

[12] H. Yeh and H. Sun. Password authenticated key exchange protocols among diverse network domains. *Computers and Electrical Engineering*, 31(3):175–189, 2005.

[13] L. Zhu and B. Tung. Public Key Cryptography for Initial Authentication in Kerberos (PKINIT). *IETF*, RFC 4556, June 2006.