Password Based Server Aided Key Exchange* **

Yvonne Cliff, Yiu Shing Terry Tin, and Colin Boyd

Information Security Institute, Queensland University of Technology
GPO Box 2434, Brisbane Q 4001, Australia.
y.cliff@isi.qut.edu.au, {t.tin, c.boyd}@qut.edu.au

Abstract. We propose a new password-based 3-party protocol with a formal security proof in the standard model. Under reasonable assumptions we show that our new protocol is more efficient than the recent protocol of Abdalla and Pointcheval (FC 2005), proven in the random oracle model. We also observe some limitations in the model due to Abdalla, Fouque and Pointcheval (PKC 2005) for proving security of such protocols.
Keywords: Key agreement, password authentication, three-party.

1 Introduction

A major goal of modern cryptography is to enable two or more users on an insecure (adversary controlled) network to communicate in a confidential manner and/or ensure that such communications are authentic. Symmetric key cryptographic tools are often used for such communications, due to their efficiency. However, due to the impracticality of every pair of users sharing a large secret key, public key and/or password based techniques are used to generate such a key when it is required. We focus on password-based key exchange, which is useful in situations where the secure storage of full length cryptographic keys is infeasible, such as in mobile environments. However, because of the short length of the password, special care must be taken when designing protocols to ensure that both the password and the key finally agreed remain secret.

One area of recent attention is password-based 3-party protocols with a formal security proof. These protocols enable two clients to exchange a secret key where each client shares a (different) password with a common server. Such protocols overcome the problem associated with 2-party password-based protocols (such as all of the password-based protocols being standardized in IEEE P1363.2 and ISO/IEC FDIS 11770-4) whereby a single user must hold as many passwords as there are parties with whom it wishes to communicate.

Although such protocols have received some attention in the literature, formal proofs have only recently been provided. Abdalla, Fouque and Pointcheval [AFP05] proved the security of a generic construction (called GPAKE) that uses

* Research funded by Australian Research Council through Discovery Project DP0345775.
** Extended abstract; for the full version see: http://sky.fit.qut.edu.au/∼boydc/papers

any two-party authenticated key exchange protocol as well as a three party key distribution protocol, and combines them with a Diffie-Hellman key exchange authenticated using a message authentication code (MAC). They proved this construction secure in a new model (which we call the AFP model) based on the models of Bellare et al. [BR93,BR95,BPR00]. However, protocols constructed according to this method can be quite inefficient.

The AFP model contains two variants. The first, called the find-then-guess (FTG) model, is similar to existing models, since it allows Reveal queries (to disclose the session key of a requested instance to the adversary) and only one Test query (where the adversary must guess whether it was told the actual session key of a session it selected). The other variant is called the real-or-random (ROR) model, and disallows Reveal queries, but allows multiple Test queries, where the keys returned by the test queries are either all real or all random. It is shown that the ROR model is stronger than the FTG model when password-based protocols are being studied. However, when high-entropy keys are used rather than passwords, protocols secure in one variant are secure in the other also.

The AFP model also defines a new notion, *key privacy*, which means that the server cannot deduce the value of the secret key shared between the clients. Key privacy may be proven separately to the protocol's semantic security. However, the AFP model does have the shortcoming of not allowing adaptive corrupt queries; corrupted parties are chosen statically at the beginning of a proof in the AFP model.

The GPAKE protocol was proven secure in the ROR variant of the AFP model, assuming that the two-party authenticated key exchange protocol used with it is also secure in the ROR model. Although most suitable password based protocols have been proven secure in the FTG model, it is claimed that most proofs, including the KOY one [KOY01], can be modified easily to meet the ROR model requirements.

Abdalla and Pointcheval [AP05] later proposed another 3-party password-based protocol, to which we refer as the AP protocol. It was proven secure using the FTG variant of the AFP model, using the random oracle (RO) model (note that earlier versions, including the conference version, have an error in the protocol description that leads to an attack [CBH05b]). The proof requires new and stronger variants of the Decisional Diffie-Hellman (DDH) assumption. The authors claim that their protocol is quite efficient, requiring 2 exponentiations and a few multiplications per party, or less than half the cost for the server compared with using GPAKE.

In this paper we propose another 3-party password-based protocol, proven secure using the Canetti-Krawczyk (CK) proof model [CK01]. This model allows the adversary to make adaptive corrupt queries. In contrast, the AFP model only allows static corrupt queries. We therefore select the CK model as it can model a wider variety of attack scenarios and allows the modular design of protocols by enabling key exchange and authentication mechanisms to be proven secure separately.

We regard it as a significant advantage that our proof is in the standard model, in contrast to the AP protocol which requires the RO model. We also examine the AP protocol efficiency claims more closely and claim that our new protocol can be more efficient with reasonable assumptions.

The rest of this paper proceeds as follows. Section 2 reviews the CK model, and is followed by a description of the protocol and its security proof in Section 3. Section 4 then discusses the efficiency, advantages and disadvantages of the proposed scheme in comparison to the AP and GPAKE protocols.

2 The Canetti–Krawczyk Model

In this section the CK approach is reviewed. Further details of the model can be found in the original papers [BCK98,CK01], a paper extending the model to justify optimization techniques [HBGN05], or the full version.

In the CK model a protocol π is modelled as a collection of n programs running at different parties, $P_1, \ldots, P_n$. Each program is an interactive probabilistic polynomial-time (PPT) machine. Each invocation of π within a party is defined as a *session*, and each party may have multiple sessions running concurrently. The communications network is controlled by an adversary $\mathcal{A}$, also a PPT machine, which schedules and mediates all sessions between the parties. Three different models exist:

- The *authenticated-links model (AM)* defines an idealized adversary, $\mathcal{A}$, that is restricted to delivering messages faithfully (but possibly out of order) between uncorrupted parties, if at all. $\mathcal{A}$ is not allowed to fabricate, modify, or replay messages of its choice except if the message is purported to come from a corrupted party.
- The *unauthenticated-links model (UM)* allows the adversary, $\mathcal{U}$, to fabricate messages and deliver any messages of its choice.
- The *hybrid model (HM)*, with adversary $\mathcal{H}$, combines the above models, and messages are marked by the sender as authentic (so that AM rules apply to the message) or unauthentic (so that UM rules apply). $\mathcal{H}$ may fabricate unauthentic messages.

Upon activation, the parties perform some computations, update their internal state, generate local output and may output messages. Local output records the occurrence of important, security-related events, such as key establishment (recorded by P_i as "Established (P_i, P_j, s, κ)" to denote that a key κ has been established with party P_j). The adversary's view consists of all parties' public authentication information, output messages and local outputs, except for the established keys (κ-values) of completed sessions. Two sessions $(P_i, P_j, s, role)$ and $(P_i', P_j', s', role')$ are said to be *matching sessions* if $P_i = P_j'$, $P_j = P_i'$, and $s = s'$, i.e. if their session-ids are identical and they recognized each other as their respective communicating partner for the session.

In addition to activating parties, $\mathcal{A}$ can corrupt a party to obtain its long term keys, request a session's session-key or session-state, request session-expiration to

erase the session key or select a test-session to receive either the real key or a random one with equal probability. A protocol is session key (SK) secure if uncorrupted parties who complete matching sessions output the same key and the probability of $\mathcal{A}$ guessing correctly whether it received the real key from its test query is no more than $\frac{1}{2}$ plus a negligible function in the security parameter.

Protocols that are SK-secure in the AM can be converted into SK-secure protocols in the UM by applying an *authenticator* to them. Authenticators can be constructed from *message transmission (MT) authenticators*, which authenticate each message almost independently of all other messages. To translate an SK-secure protocol in the AM to an SK-secure protocol in the UM an MT-authenticator can be applied to each message and the resultant subprotocols combined to form one overall SK-secure protocol in the UM. An MT-authenticator emulates the MT protocol, in which the sender A outputs 'A sent m to B.' and sends (A, B, m) to party B, and upon receipt of the authentic message, B outputs 'B received m from A.'

Constructing an authenticator by using an MT-authenticator for each message can lead to very inefficient protocols due to the large number of messages generated and the requirement that the session identifier be known before the protocol begins. Until recently, heuristic arguments were made as to why optimized versions of protocols where messages were shifted and nonces reused were secure. However, recent work [HBGN05] has provided a formal basis for such optimizations by showing and/or proving how to define a session identifier part way through the protocol, that more than one MT-authenticator may be used to construct an authenticator for an entire protocol, that certain *preamble* authenticator messages can be shifted to earlier points in the protocol if some conditions are met, that the message m being authenticated need not form part of every authenticator message, and that nonces used in some authenticators only need to be previously unused by that party in that authenticator and may be replaced with other values from the protocol. The techniques presented in that work will be used throughout this paper.

3 Conversion from Two-party to Three-party Protocol

In this section, we propose a new protocol, labelled 3DH, that uses a server's assistance to perform the Diffie-Hellman key agreement between two parties. The protocol's purpose is to enable the use of a password-based authenticator between each of the parties and the server, S. The server is a gateway responsible for connecting parties A and B faithfully and providing assurance of the identify of each party to the other.

Figure 1 shows an HM template describing the possible actions for party A, the initiator or responder of the protocol, and specifies which actions are prerequisites for others, and which may be performed in parallel. It also specifies which actions must be performed in a single activation. The use of such a template has been recommended [HBGN05] to clearly show the security requirements of a

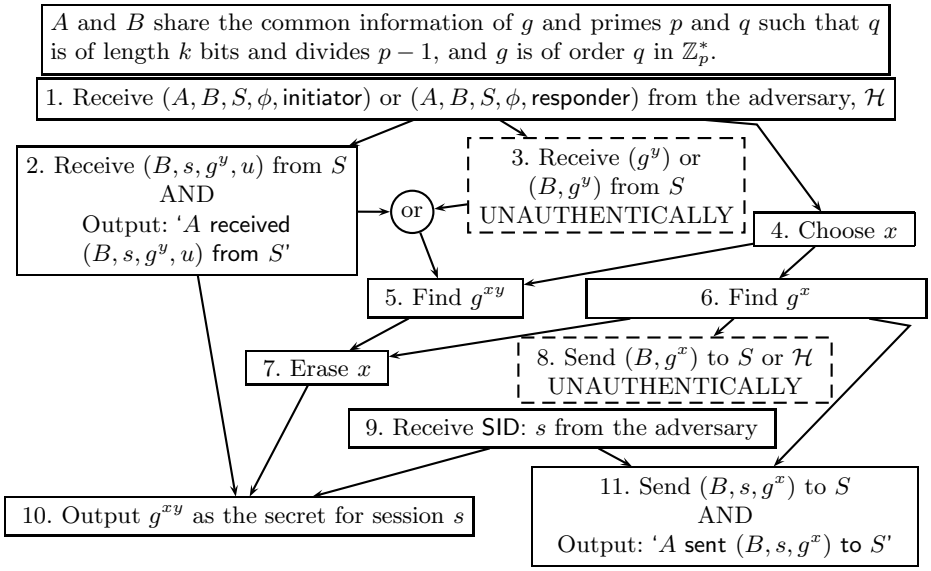

Fig. 1. Possible step order for receiver and responder in the 3DH protocol in the HM

protocol and yet allow it to be easily adapted to suit different authenticators or objectives, without breaking the security proof.

In the diagram, all messages are assumed to be authentic, unless otherwise specified, and actions that must be performed in the same activation are shown in the one box and joined with "AND." An arrow from one step to another indicates that the first step must be completed before the second is begun. Optional steps are shown using dashed boxes. The session identifier s received from the adversary must be the same as that received in the authentic message from B, otherwise the protocol halts without outputting a secret key. The template for party B is identical, except for the renaming of A to B, B to A, x to y and y to x. The value u received from S is not used in the template. Its purpose is described below.

In Figure 2, the possible interaction between the server and any party B is shown (i.e. B in Figure 2 may correspond to either A or B in Figure 1). In a single protocol run, such interaction may occur between the server and more than one party. The value u is included to ensure the requirement that all authentic messages are unique [BCK98, full version p.8, footnote 2] is met. Some authenticator proofs require this property.

An examination of Figures 1 and 2 shows that a number of steps can be performed if one party has possession of the other's unauthentic Diffie-Hellman value. Therefore, it seems logical to specify an HM protocol such as the one shown by Protocol 1 that allows a party to receive an unauthentic Diffie-Hellman value and then carry out as many actions as possible. Messages not labelled "unauth'c"

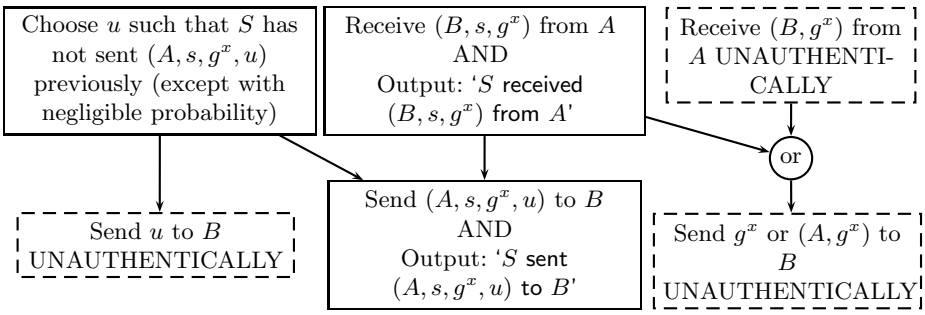

Fig. 2. Possible step order for the server of the 3DH protocol in the HM, with optional steps in dashed boxes

A		S (server)		B
$x \in_R \mathbb{Z}_q$		$u_A, u_B \in_R \{0,1\}^k$		$y \in_R \mathbb{Z}_q$
Find g^x	Unauth'c: $B, g^x \rightarrow$		Unauth'c: $A, g^x, u_B \rightarrow$	Find g^y
			$\mathcal{H}$ Unauth'c: $g^y \leftarrow$	
			$\mathcal{H}$ SID $= (g^x, g^y) \rightarrow$	$s = (g^x, g^y)$
			$\leftarrow A, s, g^y$	'B sent (A, s, g^y) to S'
		'S rec. (A, s, g^y) fr. B'		
	$\leftarrow B, s, g^y, u_A$	'S snt. (B, s, g^y, u_A) to A'		
'A rec. (B, s, g^y, u_A) fr. S'				
SID $= s = (g^x, g^y)$	$\leftarrow$ SID $= (g^x, g^y)$ $\mathcal{H}$			
$\sigma = (g^y)^x$; Erase x				
'Established (A, B, s, σ)'				
'A sent (B, s, g^x) to S'	$B, s, g^x \rightarrow$	'S rec. (B, s, g^x) fr. A'		
		'S snt. (A, s, g^x, u_B) to B'		
			$A, s, g^x, u_B \rightarrow$	
				'B rec. (A, s, g^x, u_B) fr. S'
				$\sigma = (g^x)^y$; Erase y
				'Established (B, A, s, σ)'

Protocol 1: A possible server aided 3DH HM protocol

are authentic. In this version, the adversary chooses the Diffie-Hellman values, (g^x, g^y), to be the session identifier, and the unauthentic message g^y from B to the HM adversary, $\mathcal{H}$, facilitates this. Later changes will allow the clients to choose the session identifier, so the messages in boxes will then be removed.

Our new protocol differs from the ordinary Diffie-Hellman key exchange in that protocol participants do not directly make contact with each other. Although password-based protocols where the participants communicate directly with one another do exist and have been proven secure in other proof models [KOY01,Mac02], they do not use a public key for the server and are not amenable to the CK-model, since their key exchange and authentication mechanisms cannot be separated [HK99]. Furthermore, such protocols require a party to maintain a list of passwords for each of the targets with whom it wishes to communicate. Thus the advantages of the modular approach used by the CK-

model cannot be realized by such protocols, and they are inappropriate when storage constraints exist, such as those in wireless networks.

The proposed protocol uses a common trusted server for authentication, a common practice in networking. The server is unable to calculate the session key; it can only be generated by two authenticated clients. Moreover, client authentication uses passwords which can be memorized by users, eliminating the need for shared secrets to be stored in the users' devices, and greatly reducing the potential security risk.

We disallow server corruption because there is no way of guaranteeing the establishment of secure session keys using a corrupted server controlled by the adversary. A corrupted server would effectively allow an adversary to inject authentic messages from the server into the network. In addition, session-state reveals on S have been disallowed to keep the partner and session identity definitions simple. However, if server session-state reveal queries were allowed, it would not affect the security of the protocol when used in conjunction with existing authenticators. This is because the server only forwards messages between the two clients. Since such messages are not secret, session-state reveals do not reveal any information that can be used effectively in an attack.

Proving the security of 3DH requires the use of the decisional Diffie-Hellman (DDH) assumption [Bon98]. The definition and proof are based in $\mathbb{Z}_p^*$ but the protocol may also be run in an elliptic curve group where the elliptic curve version of the DDH assumption holds.

Assumption 1 (Decisional Diffie-Hellman (DDH)) *Let k be a security parameter. Let primes p and q be such that q is of length k bits and divides $p-1$, and let g be of order q in $\mathbb{Z}_p^*$. Then the probability distributions $Q_0 = \{\langle p, g, g^x, g^y, g^{xy}\rangle\} : x, y, \stackrel{R}{\leftarrow} \mathbb{Z}_q\}$ and $Q_1 = \{\langle p, g, g^x, g^y, g^z\rangle\} : x, y, z \stackrel{R}{\leftarrow} \mathbb{Z}_q\}$ of quintuples are computationally indistinguishable.*

Theorem 1. *Any Diffie-Hellman based HM protocol where the actions of each party satisfy the requirements specified in Figures 1 and 2 is SK-secure*[1]*, provided the DDH assumption holds.*

The proof is similar to Canetti and Krawczyk's Theorem 8 [CK01] and Hitchcock et al.'s Theorem 4 [HBGN05] and is provided in the full version.

Protocol 2 shows another version of the Diffie-Hellman protocol where the adversary no longer inputs the session identifier to the parties. In addition, messages containing the same term twice have had the second term removed, and the unauthentic message to the adversary has been removed. The proof of the following theorem is very similar to that of Theorem 5 of Hitchcock et al. [HBGN05] and may be found in the full version.

Theorem 2. *If Protocol 1 is secure then so is Protocol 2.*

[1] We use the 2-party SK-security definition, since S provides authentication only, cannot be corrupted or have its sessions revealed, and does not share the secret key.

A		S (server)		B
$x \in_{\mathrm{R}} \mathbb{Z}_q$		$u_A, u_B \in_{\mathrm{R}} \{0,1\}^k$		$y \in_{\mathrm{R}} \mathbb{Z}_q$
Find g^x	Unauth'c: $B, g^x \longrightarrow$			
			Unauth'c: $A, g^x, u_B \longrightarrow$	Find g^y
				Set SID $= s = (g^x, g^y)$
		'S rec. (A, g^x, g^y) fr. B'	$\longleftarrow A, g^x, g^y$	'B snt. (A, g^x, g^y) to S'
	$\longleftarrow B, g^x, g^y, u_A$	'S snt. (B, g^x, g^y, u_A) to A'		
'A rec. (B, g^x, g^y, u_A) fr. S'				
Set SID $= s = (g^x, g^y)$				
$\sigma = (g^y)^x$; Erase x				
'Established (A, B, s, σ)'				
'A snt. (B, g^y, g^x) to S'	$B, g^y, g^x \longrightarrow$	'S rec. (B, g^y, g^x) fr. A'		
		'S snt. (A, g^y, g^x, u_B) to B'		
			$A, g^y, g^x, u_B \longrightarrow$	
				'B rec. (A, g^y, g^x, u_B) fr. S'
				$\sigma = (g^x)^y$; Erase y
				'Established (B, A, s, σ)'

Protocol 2: Secure 3DH protocol suitable for optimization

We now focus on the authenticators to be used in conjunction with Protocol 2. The password-based authenticator, $\lambda_{\text{P-ENC}}$, shown in Protocol 3, has been chosen, so that clients do not need keep secret a large key. It has a proof [HTB+03] of password-based session key (PBSK-) security, when the encryption scheme is indistinguishable under chosen ciphertext attack (IND-CCA secure), $H(m) \stackrel{\text{def}}{=} m$, $\mathcal{E}_e$ denotes encryption with key e and $\mathcal{D}_d$ denotes decryption with key d. This means that if the server refuses to complete sessions with a client after γ unsuccessful login attempts for that client, then the adversary has an advantage negligibly greater than the advantage due to simply guessing a password and attempting to impersonate the user online. The nonce N_B may be any value of B's choice, including one chosen by the adversary, if it has not been used as a nonce in this authenticator before, except with negligible probability. Since N_B is a preamble message it may be sent before the first authenticator message.

In order to increase the efficiency of the authenticator (and hence the resulting protocols) we now alter the authenticator slightly, replacing the message m by $H(m)$. The authenticator is still PBSK-secure if H is chosen to be a collision resistant one-way hash function. This new authenticator is labelled $\lambda_{\text{P-ENC-H}}$. The proof is deferred to the full version, but proceeds by observing that $\lambda_{\text{P-ENC-H}}$ sends the same messages as would be required to authenticate $H(m)$ (rather than m) using $\lambda_{\text{P-ENC}}$. Therefore, breaking $\lambda_{\text{P-ENC-H}}$ involves breaking the hash function or else breaking $\lambda_{\text{P-ENC}}$.

Another authenticator is required for messages from the server to the clients. The only suitable existing authenticators are the signature based and the encryption based authenticators [BCK98] (denoted λ_{SIG} and λ_{ENC}), as other available authenticators would require the clients to hold secret keys. If signature and encryption schemes are chosen that have proofs of security in the standard model, the encryption and signature schemes have about the same efficiency. Therefore,

A (Client)	$\lambda_{\text{P-ENC}}$	B (Server)
Known: Password, π Public key of B, e_B		Known: Password of A, π Public/private keys, (e_B, d_B)
	$\xrightarrow{m}$	$N_B \in_R \{0,1\}^k$
	$\xleftarrow{m, N_B}$	
$c = \mathcal{E}_{e_B}(H(m), N_B, A, \pi)$	$\xrightarrow{m, c}$	$v = \mathcal{D}_{d_B}(c)$
		Check $v \stackrel{?}{=} (H(m), N_B, A, \pi)$

A (Server)	λ_{ENC}	B (Client)
Known: Public/private keys, (e_A, d_A)		Known: Public key of A, e_A
	$\xrightarrow{m}$	$r_B \in_R \{0,1\}^k$
$r_B = \mathcal{D}_{d_A}(c)$	$\xleftarrow{m, c}$	$c = \mathcal{E}_{e_A}(r_B)$
$z = \mathcal{M}_{r_B}(m, B)$; Erase r_B	$\xrightarrow{m, z}$	Check $z \stackrel{?}{=} \mathcal{M}_{r_B}(m, B)$
		'B received m from A'

Protocols 3 and 4: Password and encryption based authenticators

we use λ_{ENC}, shown in Protocol 4, to minimize the number of separate cryptographic primitives that must be implemented. We observe that c is a preamble message, but r_B must be erased in the same activation as c is decrypted for the authenticator to be secure [CBH05a]. Further details of the efficiency of these authenticators and the associated signature and encryption schemes having proofs in the standard model are in the full version.

Protocol 5 shows the result of applying $\lambda_{\text{P-ENC-H}}$ and λ_{ENC} to Protocol 2. S has two encryption keys to keep the state of each authenticator independent, as required by the proof that two or more authenticators may be applied to the one protocol [HBGN05].

We can begin to improve Protocol 5 by removing the authentic message (i.e. "m" in the authenticator description) from the first and second messages of each authenticator. Since the authentic message is still being delivered in the last message of each authenticator, this is not a problem [HBGN05]. The parts to be deleted have been boxed in Protocol 5.

The optimization process can now be completed by replacing a and b with u_A and u_B respectively to reduce the number of values generated and transmitted, and by moving messages and piggybacking them together. Values a and b may be replaced since they are only required to be not previously used as a nonce with $\lambda_{\text{P-ENC-H}}$. Since u_A and u_B have negligible probability of being generated previously, they may be used in place of a and b [HBGN05]. The four messages shifted to earlier points in the protocol (those containing only b, c_{A2}, a and c_{B2}) are all preamble messages. The values u_A and u_B are already known to the adversary at the time they are used in place of a and b, so this requirement for shifting and replacing the nonces is also met. The final result is shown in Protocol 6.

4 Comparison of Protocols

This section provides a comparison of the proposed, AP and GPAKE protocols. It assumes there is only one server (so the server's identity or public key may be used in offline computations), the password may not be used in offline computations, and the time for hashes and MACs is generally ignored. Exponentiations use a 1024 bit modulus with a 160 bit exponent unless otherwise specified. If the shared secret DH value generated by using the key agreement protocol can be used directly as a key, then k in Assumption 1 would typically be 160 bits for 80-bit security. However, if a uniformly distributed key (e.g. a 128-bit key) is to be derived from the DH value, k would need to be larger unless a random oracle model is used to extract randomness from the DH value (e.g. using the left over hash lemma, $k = 160 + 128 = 288$ bits for 79-bit security when the key is derived using a universal hash function). However, the exponents may remain at 160 bits if the 160-Discrete Log Short Exponent (DLSE) assumption is made. Further details are available in works by Dodis et al. [DGH$^+$04] and Gennaro et al. [GHR04].

Abdalla and Pointcheval claim that the AP protocol only requires 2 exponentiations and a few multiplications per party. However, this figure does not take into account the G_1 and G_2 hash functions used by the protocol, which are modelled as random oracles in the proof, and map into the group, G, in which the protocol operates. The algebraic setting for the AP protocols is not specified, but a typical choice for G would be a subgroup of order q (where q is 160 to 288 bits) in $\mathbb{Z}_p^*$ where p is a 1024 bit prime. How should G_1 and G_2 be implemented in such a setting? The most natural choice seems to be that used by MacKenzie [Mac02] for his suite of PAK protocols, and by IEEE P1363.2 [IEE06] in the DLREDP-1 (Discrete Log Random Element Derivation Primitive), in which the functions are implemented as a hash of the inputs raised to a power to map the hash output into the subgroup. This power will be quite large ($1024 - 288 = 736$ bits long), and the computation would take the time of about 4.6 exponentiations with 160 bit exponents. Table 1 shows the equivalent number of exponentiations for this option. We should note, however, that a more efficient implementation may be possible when using DLREDP-2 from IEEE P1363.2 [IEE06] (but this requires adding two extra values to the domain parameters) or when $\frac{p-1}{q}$ is small or G is an elliptic curve group (since the co-factor in these cases is much smaller).

The efficiency of GPAKE mainly depends on which 2-party password based key exchange protocol is used by it. Use of the KOY protocol [KOY01] is assumed here, since it is in the standard model and was suggested as a suitable protocol in the GPAKE proposal. (Other protocols recommended, although faster, were in the RO model.) The equivalent number of exponentiations for the KOY protocol, excluding those for the one-time signature, is shown in Table 1. (Some exponentiations may be performed more efficiently together, and we have accounted for this.) The Bellare and Rogaway [BR95] key distribution scheme was suggested for use with GPAKE, and requires 1 symmetric decryption for each client, and 2 symmetric encryptions for the server. The GPAKE protocol itself requires 2 exponentiations for each client.

Protocol	Client Efficiency			Server Efficiency		
	Operation	Equivalent Exp.		Operation	Equivalent Exp.	
		Offline	Online		Offline	Online
Proposed	1 offline exp.	1				
	1 online exp.		1			
	1 online PK enc.	3	1	4 online PK dec.		12
	1 offline PK enc.	4				
	Proposed Total:	8	2	Proposed Total:		12
GPAKE						
... KOY	One-time signature key gen.			Per KOY key exchange:		
	One-time signature			One-time verification		
	Validity check		5	Validity check		5
	2 offline exp.	2		3 offline exp.	3	
	2 double exp.		2.6	1 double exp.		1.3
	2 multi-exp.	2	3.3	2 multi-exp.	2	3.3
	KOY Total:	4	11	KOY Total per key:	5	9.6
... key dist.	1 symm. dec.			2 symm enc.		
... other	2 exp.	1	1			
	GPAKE Total	5	12	GPAKE Total:	10	19.3
AP	2 exp.	1	1	2 exp.		2
	1 G_1 evaluation		4.6	2 G_1 evaluations		9.2
	1 G_2 evaluation		4.6	2 G_2 evaluations		9.2
	AP Total:	1	10.2	AP Total:		20.4

Table 1. Protocol Efficiency Comparison

The proposed protocol requires 1 offline and 1 online public key encryption, and 1 offline and 1 online exponentiation for each client, as well as 4 public key decryptions for the server.

From the summary in Table 1, we see that the GPAKE protocol is the least efficient of all the schemes, especially if offline computations are considered. Including G_1 and G_2 evaluations in the efficiency analysis of AP makes its online efficiency similar to that of GPAKE, although it requires fewer offline computations. In comparison with these two schemes, our scheme performs quite favourably. However, some symmetric operations required for the public key encryption and decryption operations in the proposed scheme have been omitted from the analysis; symmetric operations have not been included in the GPAKE totals, either. In addition, it may be possible to optimize these schemes with precomputations to make the exponentiations run faster.

Another consideration in the choice of a protocol is the number of rounds it requires. The protocol proposed here seems, at first, inefficient in the number of rounds, as it requires six. In comparison, the GPAKE protocol requires 3 rounds for the KOY protocol, 3 rounds for the key distribution (although a small change to the key distribution protocol would result in 2 rounds), and 1 round for the exchange of the DH and MAC values, making a total of 6 or 7 rounds. The AP protocol requires only two rounds. However, by reordering the messages of the proposed protocol, the number of rounds can be substantially reduced to

only three rounds, making its round complexity slightly worse than that of the AP protocol, but better than that of the GPAKE protocol. The details of this optimization are in the full version.

In addition to the above efficiency considerations, there are a number of other points to consider when selecting a protocol:

Random oracle v. standard model. Our proof is in the standard model, whereas that of the AP protocol is in the RO model. The proof of GPAKE may be in either model, depending upon the components used with it.

Corrupt queries. The CK model, which is used for our proof, allows the use of adaptive Corrupt queries to model a malicious insiders. This is contrary to the AFP model. In fact, an attack on the AP protocol has been described [CBH05b] that uses a corrupt query. The original proof did not rule out this attack because corrupt queries were not allowed in the proof model. Therefore, we can rule out a wider range of attacks on our protocol because of the more general model.

Session-state reveal queries. The CK model also allows session-state reveal queries. These queries model the exposure of data that must be kept between activations, and the requirement that exposure of such data in one session should not compromise other sessions. However, the AFP model does not allow such queries, and the AP protocol would be insecure in the presence of such queries. This is because an adversary could find a party's secret Diffie-Hellman exponent using a session-state reveal query, and then use the exponent to find the secret password from the party's first protocol message.

Key privacy. The AFP model includes the key privacy requirement, but the CK model does not. However, the proposed protocol fulfills the requirement, since the server only forwards authentic messages in Protocol 1, and does not create any messages with new content. These messages are also known to the adversary, who cannot find any information about the key. Since the server knows no more than the adversary, it is also unable to deduce any information about the secret key. However, the CK model may need some modification to include the requirement if needed for other protocols.

Concurrency. The proof of the AP protocol does not allow concurrency, so that only one instance of a player can exist at a time. On the other hand, the proposed protocol allows full concurrency. Although instructions are given on how to modify the AP protocol to achieve partial and full concurrency, the correctness of these instructions is not proven. In addition, provision of full concurrency requires two extra message flows from the server at the beginning of the protocol.

Assumptions. The AP protocol proof requires the use of some non-standard assumptions based on the Diffie-Hellman problem. Our proposed protocol only requires the DDH assumption, a collision-resistant one-way hash function, and an IND-CCA secure encryption scheme. The use of standard assumptions may be less risky than the use of new assumptions that have not been extensively studied.

Online attack detection. It is always possible for the adversary to make an online attack by guessing a party's password and running the protocol using

the guessed password. If the final key is correct, then the password must have been correct. Otherwise another password may be guessed and the protocol run again until the correct password is found. To prevent such attacks, the definition of PBSK-security for the CK model requires the server to refuse to run the protocol with any party who has made more than a certain number of incorrect guesses of the password. However, the server in the AP and KOY protocols cannot detect whether a client has made an incorrect password guess, and although the proofs differentiate active and passive attacks, this cannot be done in practice. In addition, such attacks are tolerated by the AFP security definition where the problem is that the adversary's advantage must only be less than $O(n/|\mathfrak{D}|)$ plus a negligible function, where n is the number of active sessions and $|\mathfrak{D}|$ is the size of the password dictionary. No small bound is placed on the number of (unsuccessful) online attacks; it may be the same as the maximum number of online sessions.

In conclusion, we have proposed a new password-based three-party protocol, one of the few such protocols with a security proof. Our proof is in the standard model, in contrast to a recently proposed protocol with a proof in the RO model, and has a number of other advantages that have been discussed above.

References

[AFP05] M. Abdalla, P.-A. Fouque, and D. Pointcheval. Password-based authenticated key exchange in the three-party setting. In *Public Key Cryptography—PKC 2005*, volume 3386 of *LNCS*, pages 65–84. Springer, 2005.

[AP05] M. Abdalla and D. Pointcheval. Interactive Diffie-Hellman assumptions with applications to password-based authentication. In *Financial Cryptography and Data Security—FC 2005*, volume 3570 of *LNCS*, pages 341–356. Springer, 2005. Full version: http://www.di.ens.fr/∼pointche/pub.php?reference=AbPo05.

[BCK98] M. Bellare, R. Canetti, and H. Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 419–428. ACM Press, 1998. Full version: http://www-cse.ucsd.edu/users/mihir/papers/key-distribution.html.

[Bon98] D. Boneh. The decision Diffie-Hellman problem. In *Proceedings of the Third Algorithmic Number Theory Symposium*, volume 1423 of *LNCS*, pages 48–63. Springer, 1998.

[BPR00] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In *Advances in Cryptology – Eurocrypt 2000*, volume 1807 of *LNCS*, pages 139–155. Springer, 2000.

[BR93] M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Advances in Cryptology – CRYPTO'93*, volume 773 of *LNCS*, pages 232–249. Springer, 1993. Full version: www-cse.ucsd.edu/users/mihir.

[BR95] M. Bellare and P. Rogaway. Provably secure session key distribution – the three party case. In *Proceedings of the 27th ACM Symposium on the Theory of Computing*, pages 57–66. ACM Press, 1995.

[CBH05a] K.-K. R. Choo, C. Boyd, and Y. Hitchcock. Errors in computational complexity proofs for protocols. In *Advances in Cryptology—Asiacrypt 2005*, volume 3788 of *LNCS*, pages 624–643. Springer, 2005.

[CBH05b] K.-K. R. Choo, C. Boyd, and Y. Hitchcock. Examining indistinguishability-based proof models for key establishment protocols. In *Advances in Cryptology—Asiacrypt 2005*, volume 3788 of *LNCS*, pages 585–604. Springer, 2005.

[CK01] R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *Advances in Cryptology – Eurocrypt 2001*, volume 2045 of *LNCS*, pages 453–474. Springer, 2001. http://eprint.iacr.org/2001/040.ps.gz.

[DGH+04] Y. Dodis, R. Gennaro, J. Håstad, Krawczyk H., and T. Rabin. Randomness extraction and key derivation using the CBC, cascade and HMAC modes. In *Advances in Cryptology — CRYPTO 2004 Proceedings*, volume 3152 of *LNCS*, pages 494–510. Springer, 2004.

[GHR04] R. Gennaro, Krawczyk H., and T. Rabin. Secure hashed Diffie-Hellman over non-DDH groups. In *Advances in Cryptology — EUROCRYPT 2004 Proceedings*, volume 3027 of *LNCS*, pages 361–381. Springer, 2004. Full version in: Cryptology ePrint Archive (http://eprint.iacr.org/2004/099), Report 2004/099.

[HBGN05] Y. Hitchcock, C. Boyd, and J. M. González Nieto. Modular proofs for key exchange: rigorous optimizations in the Canetti-Krawczyk model. *Applicable Algebra in Engineering, Communication and Computing (AAECC) Journal*, 2005. Special issue on Mathematical Techniques in Cryptology; http://dx.doi.org/10.1007/s00200-005-0185-9.

[HK99] S. Halevi and H. Krawczyk. Public-key cryptography and password protocols. *ACM Trans. on Information and Systems Security*, 2(3):230–268, 1999.

[HTB+03] Y. Hitchcock, Y. S. T. Tin, C. Boyd, J. M. González Nieto, and P. Montague. A password-based authenticator: Security proof and applications. In *4th International Conference on Cryptology in India – INDOCRYPT 2003*, volume 2904 of *LNCS*. Springer, 2003.

[IEE06] IEEE (Institute of Electrical and Electronics Engineers, Inc.). P1363.2: Standard specifications for password-based public-key cryptographic techniques (draft version d23), 2006. http://grouper.ieee.org/groups/1363/passwdPK/draft.html.

[KOY01] J. Katz, R. Ostrovsky, and M. Yung. Efficient password-authenticated key exchange using human-memorable passwords. In *Advances in Cryptology—EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 475–494. Springer, 2001.

[Mac02] P. MacKenzie. The PAK suite: Protocols for password-authenticated key exchange. Technical Report 2002-46, DIMACS, 2002. ftp://dimacs.rutgers.edu/pub/dimacs/TechnicalReports/TechReports/2002/2002-46.ps.gz.

A		S (server)		B
Known: password, π_A Public keys of S : e_{S1}, e_{S2}		Known: passwords π_A, π_B Secret keys: d_{S1}, d_{S2}		Known: password, π_B Public keys of S : e_{S1}, e_{S2}
$x \in_{\mathrm{R}} \mathbb{Z}_q$; $r_A \in_{\mathrm{R}} \{0,1\}^k$ Find g^x	$\xrightarrow{B, g^x}$	$a, b, u_A, u_B \in_{\mathrm{R}} \{0,1\}^k$	$\xrightarrow{A, g^x, u_B}$	$y \in_{\mathrm{R}} \mathbb{Z}_q$; $r_B \in_{\mathrm{R}} \{0,1\}^k$ Find g^y
				Set SID $= s = (g^x, g^y)$
			$\xleftarrow{\boxed{A, s}}$	'B snt. (A, s) to S'
			$\xrightarrow{\boxed{A, s}, b}$	$c_{B1} = \mathcal{E}_{e_{S1}}(H(A,s), b, B, \pi_B)$
		$v_B = \mathcal{D}_{d_{S1}}(c_{B1})$	$\xleftarrow{A, s, c_{B1}}$	
		$v_B \stackrel{?}{=} (H(A,s), b, B, \pi_B)$ Erase v_B		
		'S rec. (A, s) fr. B'		
	$\xleftarrow{\boxed{B, s, u_A}}$	'S snt. (B, s, u_A) to A'		
$c_{A2} = \mathcal{E}_{e_{S2}}(r_A)$	$\xrightarrow{\boxed{B, s, u_A}, c_{A2}}$	$r_A = \mathcal{D}_{d_{S2}}(c_{A2})$		
		$z_A = \mathcal{M}_{r_A}(B, s, u_A, A)$		
$z_A \stackrel{?}{=} \mathcal{M}_{r_A}(B, s, u_A, A)$	$\xleftarrow{B, s, u_A, z_A}$	Erase r_A		
'A rec. (B, s, u_A) fr. S' Set SID $= s = (g^x, g^y)$ $\sigma = (g^y)^x$; Erase x 'Established (A, B, s, σ)' Let $t = (g^y, g^x)$				
'A snt. (B, t) to. S'	$\xrightarrow{\boxed{B, t}}$			
	$\xleftarrow{\boxed{B, t}, a}$			
$c_{A1} = \mathcal{E}_{e_{S1}}(H(B,t), a, A, \pi_A)$				
	$\xrightarrow{B, t, c_{A1}}$	$v_A = \mathcal{D}_{d_{S1}}(c_{A1})$		
		$v_A \stackrel{?}{=} (H(B,t), a, A, \pi_A)$ Erase v_A		
		'S rec. (B, t) fr. A'		Let $t = (g^y, g^x)$
		'S snt. (B, t, u_B) to. B'	$\xrightarrow{\boxed{A, t, u_B}}$	$c_{B2} = \mathcal{E}_{e_{S2}}(r_B)$
		$r_B = \mathcal{D}_{d_{S2}}(c_{B2})$	$\xleftarrow{\boxed{A, t, u_B}, c_{B2}}$	
		$z_B = \mathcal{M}_{r_B}(A, t, u_B, B)$		
		Erase r_B	$\xrightarrow{A, t, u_B, z_B}$	$z_B \stackrel{?}{=} \mathcal{M}_{r_B}(A, t, u_B, B)$
				'B rec. (B, t, u_B) fr. S'
				$\sigma = (g^x)^y$; Erase y
				'Established (A, B, s, σ)'

Protocol 5: Unoptimized authenticated server aided 3DH

A		S (server)		B
Known: password, π_A Public keys of S : e_{S1}, e_{S2}		Known: passwords pi_A, π_B Secret keys: d_{S1}, d_{S2}		Known: password, π_B Public keys of S : e_{S1}, e_{S2}
$x \in_{\mathrm{R}} \mathbb{Z}_q$; $r_A \in_{\mathrm{R}} \{0,1\}^k$ Find g^x $c_{A2} = \mathcal{E}_{e_{S2}}(r_A)$	$\xrightarrow{B, g^x, c_{A2}}$	$u_A, u_B \in_{\mathrm{R}} \{0,1\}^k$		$y \in_{\mathrm{R}} \mathbb{Z}_q$; $r_B \in_{\mathrm{R}} \{0,1\}^k$ Find g^y
			$\xrightarrow{A, g^x, u_B}$	Set SID $= s = (g^x, g^y)$ 'B snt. (A, s) to S' $c_{B1} = \mathcal{E}_{e_{S1}}(H(A, s), u_B, B, \pi_B)$
		$v_B = \mathcal{D}_{d_{S1}}(c_{B1})$	$\xleftarrow{A, s, c_{B1}, c_{B2}}$	$c_{B2} = \mathcal{E}_{e_{S2}}(r_B)$
		$v_B \stackrel{?}{=} (H(A, s), u_B, B, \pi_B)$ Erase v_B 'S rec. (A, s) fr. B' 'S snt. (B, s, u_A) to A' $r_A = \mathcal{D}_{d_{S2}}(c_{A2})$ $z_A = \mathcal{M}_{r_A}(B, s, u_A, A)$ Erase r_A		
	$\xleftarrow{B, s, u_A, z_A}$			
$z_A \stackrel{?}{=} \mathcal{M}_{r_A}(B, s, u_A, A)$ 'A rec. (B, s, u_A) fr. S' Set SID $= s = (g^x, g^y)$ $\sigma = (g^y)^x$; Erase x 'Established (A, B, s, σ)' Let $t = (g^y, g^x)$ 'A snt. (B, t) to. S' $c_{A1} = \mathcal{E}_{e_{S1}}(H(B, t), u_A, A, \pi_A)$				
	$\xrightarrow{B, t, c_{A1}}$	$v_A = \mathcal{D}_{d_{S1}}(c_{A1})$		
		$v_A \stackrel{?}{=} (H(B, t), u_A, A, \pi_A)$ Erase v_A 'S rec. (B, t) fr. A' 'S snt. (B, t, u_B) to. B' $r_B = \mathcal{D}_{d_{S2}}(c_{B2})$ $z_B = \mathcal{M}_{r_B}(A, t, u_B, B)$ Erase r_B		
			$\xrightarrow{A, t, u_B, z_B}$	Let $t = (g^y, g^x)$ $z_B \stackrel{?}{=} \mathcal{M}_{r_B}(A, t, u_B, B)$ 'B rec. (B, t, u_B) fr. S' $\sigma = (g^x)^y$; Erase y 'Established (A, B, s, σ)'

Protocol 6: Optimized authenticated server aided Diffie–Hellman protocol