國立交通大學

資訊科學與工程研究所

碩 士 論 文

以可攜裝置達到的安全登入系統

A Secure Login System with Portable Devices

研 究 生：張君偉

指導教授：曾文貴　教授

中 華 民 國　九 十 五　年 八 月

以可攜式裝置達成的安全登入系統
A Secure Login System with Portable Devices

研 究 生：張君偉　　Student：Jun-Wei Zhang

指導教授：曾文貴　　Advisor：Wen-Guey Tzeng

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

August 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年八月

誌　　謝

首先要感謝我的指導老師曾文貴教授，在我碩士班的學習過程中，帶領我深入密碼學的領域，老師認真積極的教學態度，使我受益良多。另外，我要感謝口試委員，交大資工蔡錫鈞教授與中央研究院資訊科學研究所呂及人教授，在論文上給我許多建議與指導，讓我的論文更加完善。除此之外，我要感謝實驗室學長朱成康與學姊林孝盈的指導，實驗室同學陳冠廷、實驗室學弟周昆逸、陳仕烽及劉易儒的幫忙，和你們一起討論和學習是一件愉快的事。

最後，我要感謝我的父母和兄長，不論在精神或物質上都給我極大的支持，讓我在無後顧之憂的情況下可以順利完成學業。得諸人者太多，出諸己者太少，要感謝的人實在太多了，無法一一列舉，就感謝天罷。

以可攜式裝置達成的安全登入系統

學生：張君偉　　指導教授：曾文貴 博士

國立交通大學資訊工程與科學系

摘要

在這篇論文中，我們考慮在可攜式安全裝置的協助下，密碼驗證金鑰交換系統的表現。可攜式安全裝置可以是智慧型手機或個人行動助理等可安全存放資料及和個人電腦溝通的裝置。使用者攜帶其所有的可攜式安全裝置至公開電腦，便可自動且安全的進行身分驗證級金鑰交換的動作。

除了在一般密碼驗證金鑰交換系統所討論的安全性需求，好比身分驗證，金鑰的語意安全以及向前性安全。我們額外的考量了密碼保護以限制一個不完全信賴的公共電腦，在金鑰交換以及身分驗證的過程中得知使用者金鑰的可能性。

在實務上，使用者只需記憶自己的帳號和密碼。在沒有可攜式安全裝置的場合，使用者亦可在可信賴的電腦上，輸入帳號和密碼，來進行身分驗證以及金鑰交換。

關鍵字：密碼驗證，可攜式安全裝置，金鑰交換

Secure Login System with portable devices

Student: Jun-Wei Zhang Advisor: Dr. Wen-Guey Tzeng

Institute of Computer Science and Engineering

National Chiao Tung University

Abstract

We consider the password-based authenticated key exchange with help of the secure portable device. The secure portable device may be a smartphone or PDA which can store authentication information securely and communicate with computers. A user can bring his own secure portable device to some public computer and perform authentication and key exchange automatically and securely with his device. Beside the security requirements one usually consider in the password-based authenticated key exchange, such as the *authentication*, the *semantic security* of session keys and *forward security* of session keys, we additionally consider the *password protection* to against semi-trusted public computers from learning user's password. Users only need to have their password in hand and may perform a password authentication by inputing identities and passwords on computers. Our results hold in the random oracle model.

Keywords: Password Authentication, Key Exchange Protocol, Secure Portable Device

Contents

Chapter 1

Introduction

1.1 Motivation

Many servers on the Internet today rely on password authentication to verify the identity of a user. In particular, a user is asked to provide his identity and password to login the server before requiring services, such as ordering books or watching stream video. Usually, most users tend to memorize short and low entropy passwords such as their birthdays or favorite movie names. Those passwords come from a relative small dictionary. Thus if the adversary get some information which is enough to verify a password guess, he can perform a exhaustive search in the dictionary of possible passwords to determine user's password. Such attack is called offline dictionary attack. Many security experts researched associated with password authentication to against such attacks. On the other hand, as the growth of Internet services, there are many servers on the Internet, it's difficult for a user to remember a different password for each server. However, to use a common password between difference servers may cause security problem. For example, if a user chose

the same password for a bookstore server and a bank server, an adversary or a corrupted bookstore server may impersonate the user to the bank server. This is quite dangerous in practice.

Mobile devices, such as cellphones or personal digital assistants (PDA), are more and more popular day by day. Most people carry at least one such devices in their daily living. These devices are so useful since they can perform certain functions and store personal information for users. The functions may include to give a phone call or to browse on the Internet, and the personal information may be the address list or calendar. User's password is also one kind of personal information, one may ask that can these devices can memorize the passwords for users? If a user can put all his passwords into the device, then he can just remember the password for his device. Moreover, if the device is able to perform the password authentication on it's own, the user can just bring his device to some computer, press "login" button and all tasks, including the authentication of the user and the construction of the session key for later uses, are automatically done by the three-parties communication between the device, the computer and the server.

However, there are many problems if we simply store user's password into the device and perform some password-based authenticated key exchange protocol in existence, between the device and the server. Consider the case if a user lost his device, he must change all his passwords stored in the device or someone may obtain those password from the device, this is very inconvenient

for the user. On the other hand, after the execution of the protocol, a session key should be shared between the server and the computer for later use, but notice that the protocol is performed by the device and the server, and one can not directly transfer the session key from the device to the computer since an adversary may eavesdrop the communication. Moreover, even if the computer can obtain the session key securely, we still need to prevent the computer to gain some useful information about user's password from the session key. For example, a common method to convert a password-based authenticated key exchange protocol in the symmetric password model to a password-based authenticated key exchange protocol in the asymmetric password model is to "*execute the protocol with verification data, and then sign the agreed session key with user's password*". But one can easily find that a converted protocol will suffer an offline dictionary attack since the computer know both the session key and the signature of the session key.

The device here is a helper but not in place of the user, we allow the user to login some server in the original way. In particular, if a user do not have the device or do not bring it with him, he can still perform the password authentication by inputing his identity and password on the computer. However in this case, some security requirements cant no be meet. For example, at least the computer can easily get user's password by a key logging.

Give a more precise problem description: We want to construct a three-parties (the device, the computer and the server) password-based authenti-

cated key exchange protocol, which can protect the session key, the password and the authentication. In particular, we consider the following security notions : the *semantic security of the session keys*, which we model by a extended game based on [CPP04]; the *authentication*, which restrict the probability that some adversary can impersonate a legal user; the *forward security of the session keys*, which entails that if the user leak his password accidentally, the session keys used before are still semantic security; the *password protection*, which means that the computer cannot learn any useful information about user's passwords from the execution of the protocol.

1.2 Our Contribution

Our contribution in this paper is a password-based authenticated key exchange protocol which satisfy the previous mentioned requirements. Our protocol is provably secure in the random oracle model, assuming the hardness of the Decisional Diffie-Hellman problem and the computation Diffie-Hellman problem.

Our protocol is named password-based authenticated key exchange with secure portable device (**PAKE-SPD**). **PAKE-SPD** protects the session keys, the authenticaion and the passwords according to formal security models described in chapter 3. It provides some additional properties, such as lost-free, which means that the lost of the device does not cause the user have to change his password, in particular, the only thing he needs to do is

to tell the servers to revoke his lost device. **PAKE-SPD** is convenience for use, if a user do not bring his mobile device with him, he can still perform the password authentication by inputing his identity and password on the computer.

PAKE-SPD is in the asymmetric password model, it means that the server only holds some transformed password that can be only used to verify user. Then if the same password is used between different servers, the server can not impersonate the user with the stored authentication information (transformed password) stored.

PAKE-SPD is easily for the implementation. The difference between two login modes, with or without device, is only the verification data used in the protocol. Thus, one can use a single flag to denote the mode and all processes are the same between two modes except taking different input data.

1.3 Related Work

The related study of password-based authenticated key exchange protocols which can resist to dictionary attacks started from Bellovin and Michael [BM92], [BM93]. In which they proposed *Encrypted Key Exchange* protocol (EKE). Suppose there are two parties A and B. The main idea of their protocol is that A generate a public/private key pair and then send this public key encrypted with the common password to B, where the encryption is done

by some symmetric encryption scheme. B can use the common password to decrypt the ciphertext and obtain the public key. Then he randomly choose a session key and encrypted this key with obtained public key. The resulting ciphertext is then re-encrypted with the common password and finally sent back to A. Now A can easily obtain the session key by decrypting with both the private key he generated and the common password.

Bellare, Pointcheval and Phillip proved the security of the *EKE* protocol in [BPR00]. However, the proof is going in the ideal-cipher model which is very strong (even stronger than the random oracle model). In the *EKE* protocol, one must be care about how to encrypt the public key with the common password such that the cleartexts encrypted should not have any redundancy. Otherwise one can apply the partition attack on it: One first guesses a password, decrypts the cihertext and check if the obtained cleartext has the same redundancy. If not it means that the guess is wrong and one candidate can be eliminated.

In [BMP00], Boyko, MacKenzie and Patel proposed a protocol named *PAK* which is secure in the random oracle model and a modification version named *PAK-X* in which only the client side stores a plaintext version of the password (and thus in asymmetric password model). In [MPS00], MacKenzie, Patel and Swaminathan modified *OKE* and protected-*OKE* to obtain a password-based authenticated key exchange protocol that can be proved secure in the random oracle. In [CPP04], Dario, David and Thomas

proposed a general way to uses any trapdoor hard-to-invert isomorphisms to construct a password-based authenticated key exchange protocol which is secure in the random oracle model. The protocols above are all proven secure in the random model. In [GL01], Goldreich and Lindell proposed a protocol which can be proved secure in the standard model, their proposal is based on sole existence of trapdoor permutations. In [KOY01], Katz, Ostrovsky and Yung proposed a protocol based on the Decisional Difie-Hellman problem. Their protocol requires only roughly 8 times more computation than standard Diffie-Hellman key exchange and can be proven secure in the standard model.

There are many studies of the security model of password-based authenticated key exchange. In [CBH05], Choo, Boyd and Hitchcock examined indistinguishability-based proof models and mentioned the difference between them. In [GL03], Gennaro and Lindell presented a general framework for password-based authenticated key exchange protocols, their procotol is based on a notion of smooth projective hashing [CS02]

On the other hand, many variants of password-based authenticated key exchange have been proposed. In [KR06], Vladimir and Charles assumed that there exist some long keys shared between the server and the client. They proposed a protocol that is strong enough to against denial of server attacks. In [MSJ02], MacKenzie, Shrimpton and Jakobsson proposed a password-based authenticated key exchange protocol in which a single server is re-

placed by a set of servers, such that the password is secure unless too many servers are compromised. Under a similar setting, Raimondo and Gennaro proposed a threshold password-based authenticated key exchange in [RG03]. In [ACFP05], Abdalla, Chevassut, Fouque and Pointcheval proposed a protocol in which they considered about the appearance of the gateway.

The most studies above were processing in the symmetric password model, in which we assume that the password is directly shared by A and B. In the asymmetric password model, only one party A may have his password and the other one, B holds only the transformed passwords such as the hashed value of user's password. In [BM92], the authors give a idea to convert a protocol in symmetric password model to a protocol in asymmetric password model: One can execute the original protocol with the transformed password in place of the original password. The resulting session key should be known only to A and B. Then A signs the session key with his original password and sends the signature to B. B then verifies the signature with the transformed password and concludes the protocol successfully only if the signature is correct.

Chapter 2

Preliminary

In this chapter, we discuss the original security model for password-based key exchange protocol and some of its variations. The variations involve gateway password-based authenticated key exchange **GPAKE** [ACFP05] and key exchange in the combined keys model [KR06].

2.1 Password-based Authenticated Key Exchange

The model described in this section is based on that in [CPP04]. The adversary A is defined to be a probabilistic machine that can control all communications between every parties. The goal of the adversary A is to break any protocol by attempting to impersonate a user or distinguish some agreed session key from a random one.

Initialization. We have fixed a set of protocol participants each of which is either a Client $C \in Client$ or a server $S \in Server$. For convenience we denote any participant as $P \in Client \cup Server$. Each $C \in Client$ holds some

password which is chosen from a relative small space of possible passwords. In a protocol of symmetric model, each $S \in Server$ holds the passwords of all users. The participants and corresponding passwords are well set in the initiation before execution of the protocol.

The protocol determines how participants behave in response to the input from their environment. Each participant may execute the protocol multiple times with different partners, this is modeled by allowing each participant an unlimited number of instance in which to execute the protocol. Let instance C^i (resp. S^i) denote instance i of client C (resp. S). The adversary A is defined to be a probabilistic machine that is in control of all communications between parties, which is formalized by allow A to ask the following queries

- $Execute(C^i, S^j)$: This query models passive attacks, where the adversary may eavesdrop all the communication between C^i, S^j. This oracle will execute the protocol between instance C^i and S^j, and outputs a transcript of this execution.

- $Send(P^i, M)$: This query models actives attacks, where the adversary may send a message M to instance P^i. This oracle computes a response according to the protocol and decides if P^i accepts or terminates. Then it outputs the response and the decision (if exists).

- $Reveal(P^i)$: This query models the lost of the session key by any instance P^i. This query is available only if the attacked instance P^i

accepts, which means it actually holds some session key. The output of this oracle is the session key.

Usually, there are two main security notions for authenticated key exchange protocols. The first one is the semantic security of the session key, which means that the agreed key should be unknown to anybody else than the participants instances. The second one is authentication, which means that there should always exist a partner instance for any terminated instance.

Semantic Security. The semantic security of the session key is modeled by an additional query $Test(P^i)$. $Test$ oracle is available to A only if the attacked instance is *Fresh*, where the freshness notion captures the intuitive fact that a session key is not obviously known to the adversary. Formally, we say an instance P^i is fresh if (a) P^i has accepted (b) neitherP^i nor its partner have been ask for a *Reveal* query. If P^i is fresh, the output of $Test(P^i)$ depends on a random bit z. When $z = 0$, the output is the session key which P^i holds. When $z = 1$, the output is a random key.

In the Bellare-Rogaway model [BPR00], we restrict the adversary A can query $Test$ oracle at most once. In the random-or-real model, the adversary A can query $Test$ oracle many times, the output of $Test$ is corresponding to the same random bit z. We say an adversary A is successful if he correctly guess the random bit z.

Authentication. Here we consider the unilateral authentication of a client instance. We say a client instance C^i authenticates with a server S^j instance

if S^j terminates (sets a terminate flag) with partner C^i and both instances have the same session key. The adversary A successfully impersonates a client instance if a server instance terminates without any accepted client instance which shares the session key with it. The unilateral authentication of a server instance can be defined vice versa. If a protocol can provide authentications of both clients and servers, this protocol then provides mutual authentication.

2.2 Gateway Password-based Authenticated Key Exchange

One variation of the original password-based authenticated key exchange is gateway password-based authenticated key exchange [ACFP05]. The major modification is on the participant settings. Beside clients and servers, it takes into account the presence of gateways (firewalls) when clients communicate with servers. This model is in the symmetric password model, the common password is shared between the client and the server. The gateway does not hold the authentication information and lies between the communication of the client and the server. The communication channel between the server and the gateway is assumed to be authenticated and private while the communication channel between the server and the client is insecure and under the control of an adversary.

The goal of the protocol is to establish an implicitly authenticated session key between the client and the gateway with the help of the server. Besides the semantic security of the session key, there are two additional security

notions to be considered in this model : Key privacy and Server Password protection.

The notion of key privacy is to capture the idea that the agreed session keys should only be known to the client and the gateway and not to the server. In order to meet this goal, one have to consider that the adversary can access to all secret information stored in the server and then show such adversary can not distinguish the real session key from a random one in a passive attack. Notice that in an active attack, an adversary can always get the session key by using the secret data to play the client's role in an execution of the protocol.

The notion of server password protection is to capture that the gateway should not be able to learn the password stored in the server. The gateway can know the agreed session key and some secret data he chose. We ask that after some executions of the protocol, it should be still hard for the gateway to gain some useful information about the password. There are two different behaviors assumptions about the gateway: a malicious gateway or a semi-trust gateway.

When we consider a malicious gateway, it means that the gateway can do what it want. So in each interaction, the adversary may be able to eliminate one candidate password from the dictionary by guessing a password and simulating the client. Then the security goal is to restrict the adversary can not do much better than. When we consider a semi-trusted gateway, it means

that the gateway may follow all protocol but try to obtain user's password from what he knowns and the randomness he choose in the communications.

2.3 Key Exchange in the Combined Keys Model

This is another variation of the original password-based authenticated key exchange. For each user, beside the low entropy password pw, we assume that the user may carry some storage device. The storage device can be a smart card or a storage card, which contains long and high entropy key l. Both long key l and user's password pw are required for the authentication. If the authentication failed because one provide a wrong long key, we say that a long key failure occurs. If the authentication failed because one provide a wrong password, we say that a password failure occurs. Since a password is shorter(lower entropy) than a long key, a password failure is more danger than a long key failure. In fact, password failures may correspond to the attempts of the dictionary attacks or denial of server attacks. Beside long keys and passwords, the client is assumed to have the public key of the server.

In this model, we will consider the following security requirement :

1. The adversary obtained the long key of the client, and attacks the server. The goal of the adversary is to distinguish a session key used by a fresh server instance, from a random key. This is a stronger than semantic security of the session key.

2. The adversary obtained both the password and the long key of the

client, and attacks the client. The goal of the adversary is to distinguish a session key used by a fresh client instance, from a random key.

3. The adversary obtained only the password of the client, and attacks the server. The goal of the adversary is to distinguish a session key used by a fresh client server, from a random key.

4. The adversary obtained only the password of the client, and attacks the server. The goal of the adversary is to cause a password failure. This may corresponding to the denial of access attacks, where the server suspend client's account because there are too many password failure.

5. The adversary obtained both the password and the long key of the client. The goal of the adversary is to cause any two honest partners output different session keys. This is about the implicitly authentication.

Chapter 3

Security Model

In this chapter, we present the security model we use to define the execution of our protocol for password-based authenticated key exchange with the secure portable device. One can see this model as a combination of gateway password-based key exchange and key exchange in the combined key model. In particular, We assume that there exits the secure portable device which contains some high entropy secret data. To login a server, the user carry his secure portable device to some public computer, then a device-computer-server connection will be constructed and the protocol will be executed over this connection. The goal of the protocol is to provide the authentication of the secure portable device (and thus the user) and the secure shared session key between the public computer and the server.

3.1 Overview

A password-based authenticated key exchange with the secure portable device is a three-party protocol among a secure portable device, a public com-

puter and a server. The purpose is to establish an session key between the public computer and the server, and to provide an authentication from the secure portable device to the server.

Since the connection between the public computer and the secure portable device is short and totally visible to the user. We assume that the communication channel between the public computer and the secure portable device is assumed to be authenticated but may loss some information to the adversary. The channel between the public computer and the server is insecure and under the control of the adversary.

The security requirements of our password-based authenticated key exchange with the secure portable device are some different from those models above. In particular, besides asking the semantic security, the authentication and the forward security, we also ask that the chances of the public computer learning some information on the password after some interactions should be negligible. Moreover, even if the adversary has stolen the secure portable device, it is still difficult for him to gain any information about password.

3.2 Security Model

Participants. As in [CPP04], we restrict there are only one secure portable device, one public computer, and one server in our model. But one can indeed easily extend this model, and corresponding proof, to the general case. Let D denote the secure portable device, PC denote the public computer and

S denote the server. Each of them may have several instance involved in the same time. Let D^i denote the i-th instance of D, PC^j denote the j-th instance of PC and S^k denote the k-th instance of S. For conveniency we denote by U when we consider about anyone of them.

Partnering. As in the most often assumption, we use the notion of partner based on the session identifications (sid), this means that a secure portable device instance D^i, a public computer instance PC^j and a server instance S^k are said to be partners if

1. D^i, PC^j and S^k are accepted.
2. D^i, PC^j and S^k share the same session identifications.

Where sid can be defined as the partial .transcript

Password. The user has a low-entropy secret pw which is chosen uniformly at random from a relative small space dictionary. f is some one-way function defined in the protocol which maps a password pw to a sign/verfiy key pair (sk, vk). The device D holds both keys (sk, vk) and the server S holds only verify key vk.

Semantic Security of the Session Key

Since we assume an authenticated channel between the public computer and the secure portable device, the security model is similar to 2.1. In particular, to the communication between the secure portable device and the public computer, the adversary can only do eavesdrop. Thus we can then restrict

the instances involved in oracle queries can only be PC or S, and the ability to eavesdrop is captured by adding some extra information in the $Send$ and $Execute$ queries.

On the other hand, we will combine the public computer with the secure portable device when we consider about the semantic security, the authentication, and the forward security.

The list of oracles available to the adversary are as follows:

- $Execute(PC^j, S^k)$: This query models passive attacks. Notice the output involves the transcript between the secure portable device and the public computer and the transcript between the public computer and the server.

- $Send(U^i, M)$: This query models active attacks. But here we restrict U can only be PC or S. When (PC^j, S^k) is queried, this oracle will output the response of PC on input M and the transcript between PC^j and its partner secure portable device.

- $Reveal(U^i)$: Here we restrict U can only be PC or S. This oracle will outputs the session key of U^i if U^i has accepted.

semantic security The semantic security of the session key is modeled by added an additional oracle $Test$. The $Test$ oracle is defined as follows:

- $Test(U^i)$: Since only the public computer and the server may have the session key, U can only be PC or S. If the session key for U^i is set, and

U^i is fresh, this oracle will choose a random bit z uniformly at random and return the session key for U^i if $z = 1$ or a random of key of the same size if $z = 0$.

Freshness The notation of freshness is the same as the one in [CPP04], that is, a instance U^i is fresh if

1. U^i has accepted.

2. Neither U^i nor its partners has been queried to *Reveal* oracle.

Formally, let $Succ^{ss}$ denote the event in which the adversary is successful. The advantage of the adversary A in violating the semantic security of the protocol $\mathbf{P}$ and the advantage function of the protocol $\mathbf{P}$ are denoted by

$$Adv^{ss}_{\mathbf{P}}(A) = 2Pr[Succ^{ss}] - 1$$

$$Adv^{ss}_{\mathbf{P}}(t) = max_A(Adv^{ss}_{\mathbf{P}}(A))$$

where the maximum is taken over all adversary A with time-complexity less than t.

Authentication.

One goal of the adversary is to impersonate a device (and thus a user). We consider only unilateral authentication of the device only. Let $Succ^{auth}$ denote the event that A successfully impersonates an device instance in an execution of $\mathbf{P}$, which means that there exists a server instance S^j which terminates but there does not exists partner instance for S^j. Formally,

we define the advantage of an adversary A as $Adv_{\mathbf{P}}^{auth}(A) = Pr[Succ_{\mathbf{P}}^{auth}]$ and the advantage function of the protocol $\mathbf{P}$ is defined as $Adv_{\mathbf{P}}^{auth}(t) = max_A(Adv_{\mathbf{P}}^{auth}(A))$, where the maximum is taken over all adversarys A with time-complexity less than t.

Forward Security.

In practice, the greatest threat to password-based authenticated key exchange scheme may simply be that a user leaks his password to the adversary . Once the password is exposed, future uses of it are compromised. We may hope that the user will change his password in time. However, the loss of the password may bring another danger. An adversary may eavesdrop and record the past transcript based on this leaked password. Now that he obtains the password, maybe he can gain some useful information about the past communications.

The security goal of forward security is to protect against this kind of threat. Even if the password is known to the adversary and the adversary holds all past transcript, the past communications, namely the session keys used before is still semantic security to the adversary.

Since the adversary holds the past transcript, he can not interact with party instance involved in the past communications. It means that the adversary can only do passive attack but not active attack. Thus, to give the security model for the froward security, we can modify the security model for the semantic security by disabling the *Send* oracle and providing the pass-

word used in *Execute* oracle to the adversary. The goal of the adversary is to distinguish the session key used by a fresh instance from a random key.

Formally, let $Succ^{fs}$ denote the event in which the adversary is successful. The advantage of the adversary A in violating the forward security of the protocol $\mathbf{P}$ and the advantage function of the protocol $\mathbf{P}$ are denoted by

$$Adv_{\mathbf{P}}^{fs}(A(pw)) = 2Pr[Succ_{\mathbf{P}}^{fs}] - 1$$

$$Adv_{\mathbf{P}}^{fs}(t) = max_A(Adv_{\mathbf{P}}^{fs}(A(pw)))$$

where pw is the password and the maximum is taken over all adversarys A with time-complexity less than t.

Password Protection.

As a similar notions in [ACFP05], one of the security threat is that the public computer may learn the password during the execution of the protocol. Clearly, after a success execution of the protocol, the public computer will know the session key, the authenticator, some short-time secret and randomness used by the public computer. However, we ask the probability that the public computer can distinguish the true password from a random one in the dictionary should be only negligibly large then $O(1/N)$, where N is the size of the dictionary.

Since the public computer is semi-trust, it means that the public computer will follow the protocol $\mathbf{P}$ but may try to obtain user's password from the information he knows in the communication. We model this ability by

defining a oracle like $Execute$, but it additionally provide the adversary the ability to choose the randomness used by the public computer. Formally, we define

- $Semi-Execute(M)$: This oracle models the ability to choose the randomness. The input M should be a message for starting the execution of the protocol or the randomness used by the public computer. This oracle outputs the transcript between the three parities.

The adversary should output a guess password after some oracle queries, we say the adversary wins if he correctly guess the password used in $Semi-Execute$ oracle. Let $Succ_{\mathbf{P}}^{pp}$ denote the event in which the adversary is success. Let $Adv_{\mathbf{P}}^{pp}(A)$ be the advantage of the adversary A in violating the password protection and $Adv_{\mathbf{P}}^{pp}(t) = max_A(Adv_P^{pp}(A))$ be the advantage function of the protocol $\mathbf{P}$, where the maximum is taken over all adversarys A with time-complexity less than t.

Chapter 4

The Protocol

In this chapter, we will describe **PAKE-SPD**, the password-based authenticated key exchange protocol with secure portable device. **PAKE-SPD** supports two different environments : One environment is that a user bring his secure portable device to some public computer, and want to login the system and construct a secure channel automatically with the secret data stored in his device. The other is similar, but the secret data is from the identity and password that user key-in on some public computer.

Our **PAKE-SPD** is build based on previous password-based authenticated key exchange protocols in [CPP04].

4.1 Password Setting

Let p be a large prime number such that the discrete logarithm problem defined in Z_p^* is hard and let $\mathbb{G} \in Z_p^*$ be a cyclic subgroup of prime order q, where g is a random generator of $\mathbb{G}$. We assume that for each server, a user has a identity id and corresponding password pw chosen from the

dictionary. There exists a one-way function f which maps each password in the dictionary to a unique ElGamal signature key pair $(sk = x \in Z_q, vk = g^x)$, where sk is the sign key and vk is the corresponding verification key. We then use a randomness $t \in Z_q$ to mask the original $(sk = x, vk = g^x)$ and get $(sk = (x + t) \in Z_q, vk = g^x \cdot g^t)$ as the masked sign key and masked verification key.

In our protocol, each user only remember his identity id and password pw, and his secure portable device carries only the masked sign key sk' and the masked verification key vk'. The server has only the masked verification key vk' and the verification key vk

4.2 Password-base Authenticated Key Exchange with the Secure Portable Device

The description of our protocol is given in Figure 4.1, where

$$H_0 : ID \times ServerName \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}$$

$$H_1 : ID \times ServerName \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \rightarrow Z_q$$

$$H_2 : ID \times ServerName \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \rightarrow \mathbb{G}$$

are random oracles. One can notice that in the environment with the secure portable device, the execution of the protocol actually uses only the masked key pair. The password (and corresponding ElGamal key pair) is not used.

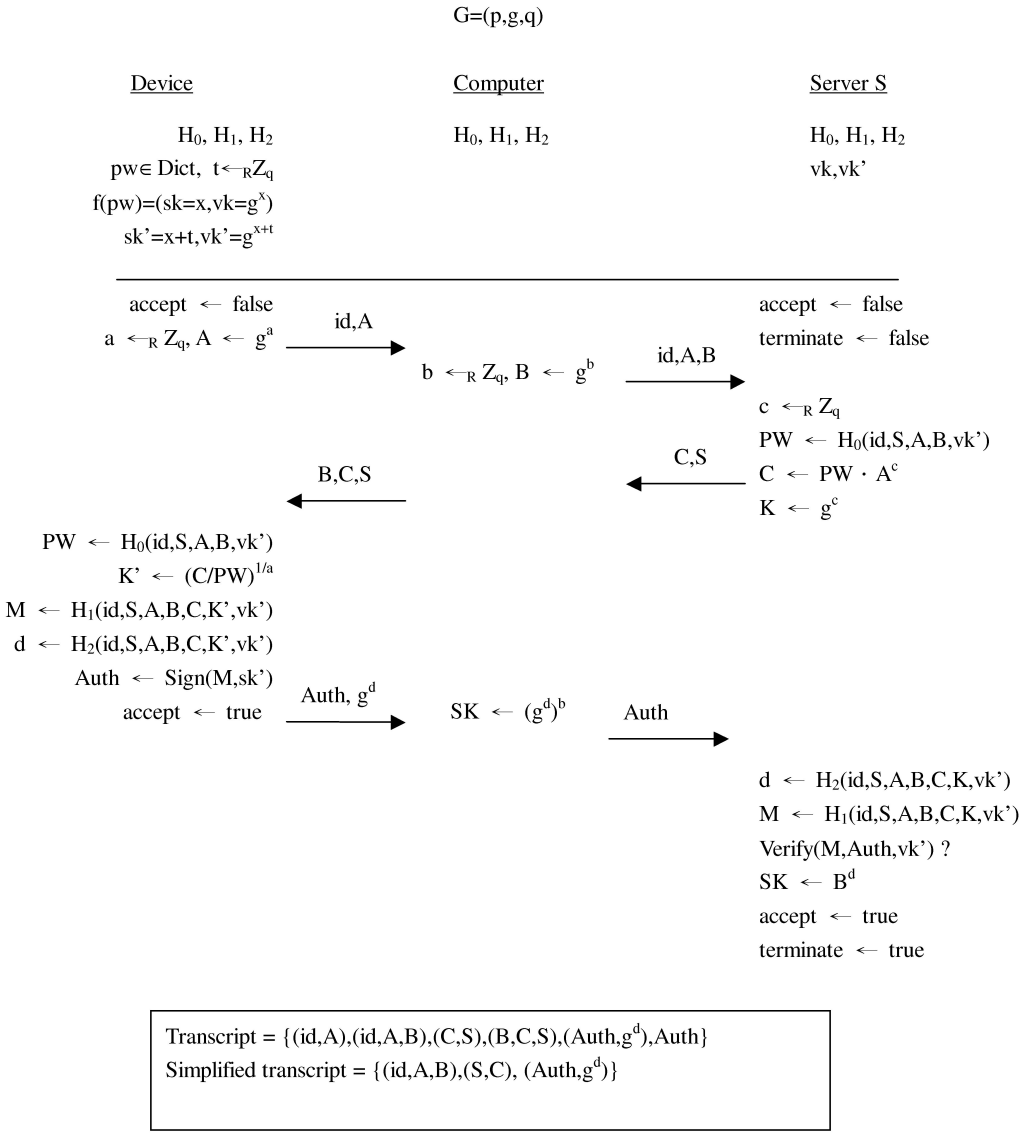


Figure 4.1: Password-base Authenticated Key Exchange with the Secure Portable Device

4.3 Password-base Authenticated Key Exchange without the Secure Portable Device

In the environment without the secure portable device, the user should key-in his identity and password on the public computer for login the server. Thus, the public computer can simulate the secure portable device by applying f on the password. The only difference is that in this environment, the execution of the protocol uses the original ElGamal key pair but not the masked one. The description of the protocol is given in Figure 4.2

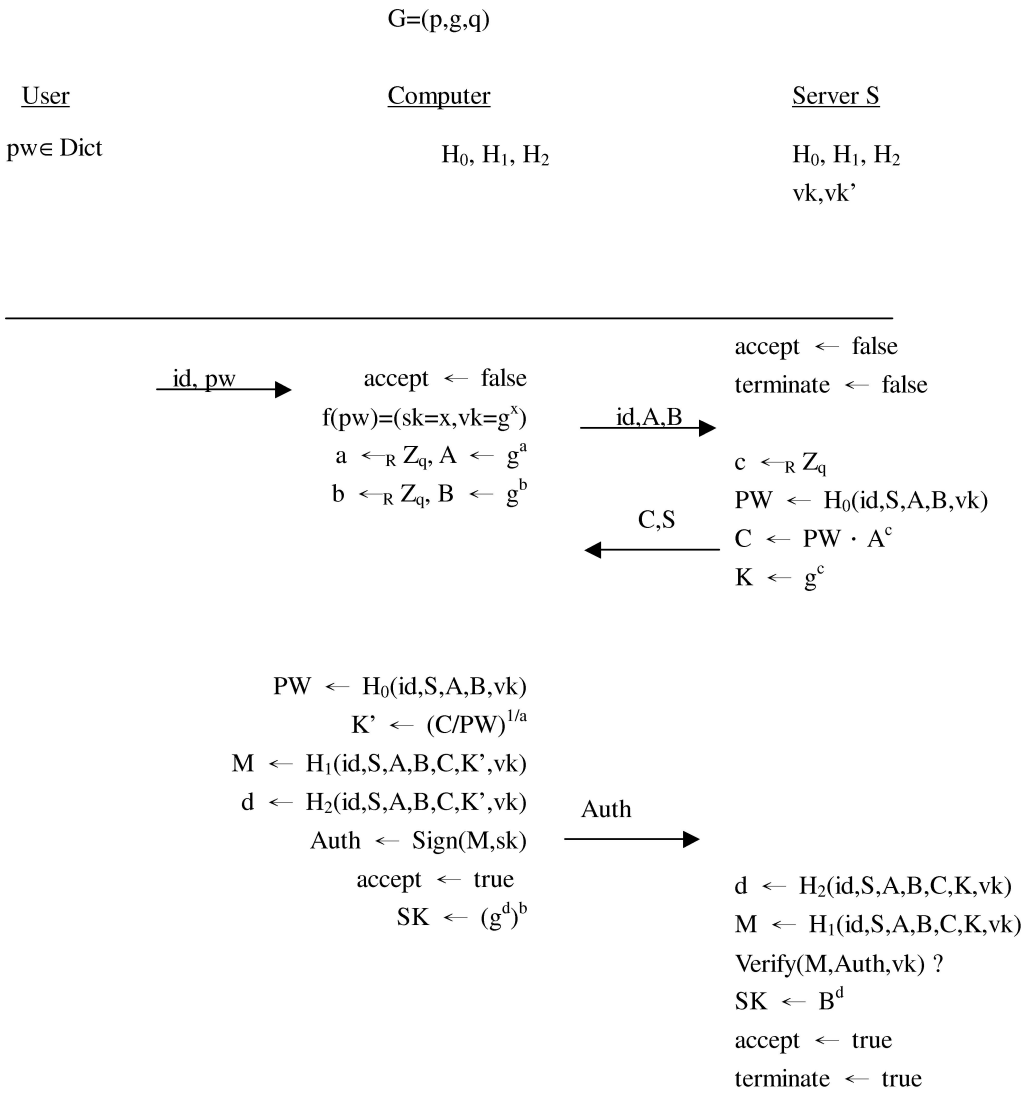


Figure 4.2: Password-base Authenticated Key Exchange without the Secure Portable Device

Chapter 5

Security Analysis

In this chapter we present the standard definition of Diffie-Hellman problem and give both decisional and computational Diffie-Hellman assumption. Then we show that our protocol provides an unilateral authentication for the device D, explicitly, an unilateral authentication for the identity id, while the agreed session keys are both semantically secure and forward secure. Moreover, we will show that even if the the adversary has corrupted the public computer, or the adversary has stolen the secure portable device, it is still hard for the adversary to gain any useful information about user's password. Our proof is in the random oracle, under DDH and CDH assumption.

For convenience, we prove the security of **PAKE-SPD** with the secure portable device, but one can easily extend this proof to the case for **PAKE-SPD** without the secure portable device, expect the password protection.

5.1 Security Assumption

Let p be a large prime number such that the discrete logarithm problem defined in Z_p^* is hard. Let $\mathbb{G} \in Z_p^*$ be a cyclic group of prime order q and g is a random generator of $\mathbb{G}$. x and y are two elements randomly chosen from $Z_q - \{0\}$.

The Computational Diffie-Hellman(CDH) problem can be defined as the task of computing g^{xy} given g, g^x, and g^y, and the Decision Diffie-Hellman(DDH) problem is defined as the task to distinguish g^{xy} from a random element in $\mathbb{G}$, given g^x, g^y, and g.

We say the Computational Diffie-Hellman assumption holds in $\mathbb{G}$ means that it is computationally intractable to compute g^{xy}. We can define this assumption more precisely by considering a experiment $Exp_{\mathbb{G}}^{CDH}(A, g)$, in which we choose two random elements x, y in $Z_q - \{0\}$, and then give both g^x, g^y to the adversary A. Let r be the output of A. Then, the experiment $Exp_{\mathbb{G}}^{CDH}(A, g)$ outputs 1 if $r = g^{xy}$ and 0 otherwise.

Define the advantage of A in violating the CDH assumption with respect to the generator g as $Adv_{\mathbb{G},g}^{CDH}(A) = Pr[Exp_{\mathbb{G}}^{CDH}(A, g) = 1]$, where the probability is taken over the random values x and y in $Z_q - \{0\}$ and the random bits A uses. The advantage function, $Adv_{\mathbb{G},g}^{CDH}(t)$, is defined as the maximum values of $Adv_{\mathbb{G},g}^{CDH}(A)$ over all A with time-complexity at most t, this means the maximal success probability over every adversary running within time t.

The Decision Diffie-Hellman assumption states that give two elements

g^x, g^y in $\mathbb{G}$, where x and y were chosen randomly from $Z_q - \{0\}$, it is computationally intractable to distinguish g^{xy} from a random element in $\mathbb{G}$. Consider a experiment $Exp_{\mathbb{G}}^{DDH}(A, g)$, in which we choose two random elements x, y in $Z_q - \{0\}$, choose a random bit b, and set $z = g^{xy}$ if $b = 0$ or choose z randomly from $\mathbb{G}$ if $b = 1$. Then we give g^x, g^y, z to the adversary A. Let r be the output of A. the experiment $Exp_{\mathbb{G}}^{DDH}(A, g)$ outputs 1 if $r = b$ or 0 otherwise.

Define the advantage of A in violating the DDH assumption with respect to the generator g as $Adv_{\mathbb{G},g}^{DDH}(A) = 2 \cdot Pr[Exp_{\mathbb{G}}^{DDH}(A, g) = 1] - 1$. The advantage function, $Adv_{\mathbb{G},g}^{DDH}(t)$, is defined as the maximum values of $Adv_{\mathbb{G},g}^{DDH}(A)$ over all A with time-complexity at most t, this means the maximal success probability over every adversary running within time t.

Often we assume that, independently of what generator g we choose, the CDH and DDH problem with respect to the generator g are hard. It means that for any generator g, $Adv_{\mathbb{G},g}^{CDH}(t)$ and $Adv_{\mathbb{G},g}^{DDH}(t)$ are very small for any reasonable t.

5.2 Security Proof

Semantic Security and Authentication

As the following theorems states, **PAKE-SPD** can provide the unilateral authentication and the agreed session key are semantically secure as long as the CDH assumption and the DDH assumption hold in $\mathbb{G}$.

Theorem 1 (Semantic Security/Unilateral Authentication) *Consider the protocol* ***PAKE-SPD*** *over* $\mathbb{G}$ *which is a cyclic group of prime order* q*, generated by* g*. Let Dict be an uniformly distributed dictionary of size* N*. For any adversary* A *within a time bound* t*, with less than* q_{active} *active interactions and* $q_{passive}$ *passive eavesdropping, and ask less than* q_{H_0}*,* q_{H_1}*,* q_{H_2} *hash queries to* H_0*,* H_1*,* H_2 *respectively, we have :*

$$Adv^{ss}_{PAKE-SPD}(A) \leq Adv^{DDH}_{\mathbb{G},g}(t)+2\cdot(\tfrac{1}{q}+4(q_{active}+q_{passive})(q_{H_1}+q_{H_2})^2 Adv^{CDH}_{G,g}(t+\epsilon)+4(q_{active}+q_{passive})(q_{H_1}+q_{H_2})Adv^{CDH}_{G,g}(t+\epsilon')+\tfrac{4q_{active}}{N}+\tfrac{4(q_{active}+q_{passive})^2}{2q}+\tfrac{2q'^2_{H_0}}{2q})$$

$$Adv^{auth}_{PAKE-SPD}(A) \leq \tfrac{1}{q}+2(q_{active}+q_{passive})(q_{H_1}+q_{H_2})^2 Adv^{CDH}_{G,g}(t+\epsilon)+2(q_{active}+q_{passive})(q_{H_1}+q_{H_2})Adv^{CDH}_{G,g}(t+\epsilon')+\tfrac{2q_{active}}{N}+(q_{active}+q_{passive})^2/(2q)+q'^2_{H_0}/2q.$$

where ϵ *is the time for* $O((q_{H_1}+q_{H_2})^2)$ *group operations,* ϵ' *is the time for* $O(q_{H_1}+q_{H_2})$ *group operations, and* $q'_{H_0}=q_{H_0}+q_{H_1}+q_{H_2}$*.*

We will prove Theorem 1 by a sequence of game reductions. The proof starts from G_0, which represents a real execution of the protocol in the random oracle model, to G_5.

For each game G_n, we define an event $SUCC^{ss-auth}_n$ corresponding to the case in which the adversary break the semantically secure but not violate the authentication, and the other event $SUCC^{auth}_n$ corresponding to the case in which the adversary breaks the authentication. Formally, event $SUCC^{ss-auth}_n$ happens if the adversary correctly guesses the bit involved in the $Test$ query,

but $SUCC_n^{auth}$ does not happen. Event $SUCC_n^{auth}$ occurs if in some time, the adversary successfully make a server instance S^i, terminates without a partner instance S^j.

Game G_0: This is the real protocol in the random oracle model, which starts by choosing a random password pw and then use pw to generate the verify key vk and the sign key sk. By the definition, we have

$$Adv^{ss}_{PAKE-SPD} = 2 \cdot Pr[SUCC_0^{ss}] - 1 \leq 2(Pr[SUCC_0^{ss-auth}] + Pr[SUCC_0^{auth}]) - 1,$$

$$ADV^{auth}_{PAKE-SPD} = Pr[SUCC_0^{auth}]$$

Game G_1: In this game, we modify G_0 by simulate all oracles as in the real attack. The detail of simulation is in Figure 5.1. Notice that we will use the suffix c or s to denote that a value is involved in a computer instance or a server instnce. One can easily find that this game is perfectly indistinguishable from the real game. Notice in the following games, the *Execute* oracle is answered by a series queries to *Send* oracle, Thus we have

$$Pr[SUCC_1^{ss-auth}] = PR[SUCC_0^{ss-auth}],$$

$$Pr[SUCC_1^{auth}] = PR[SUCC_0^{auth}]$$

Game G_2: We now modify the way on which queries to H_1 and H_2 are managed. In particular, whenever a query to H_1 or H_2 occurs, we query H_0

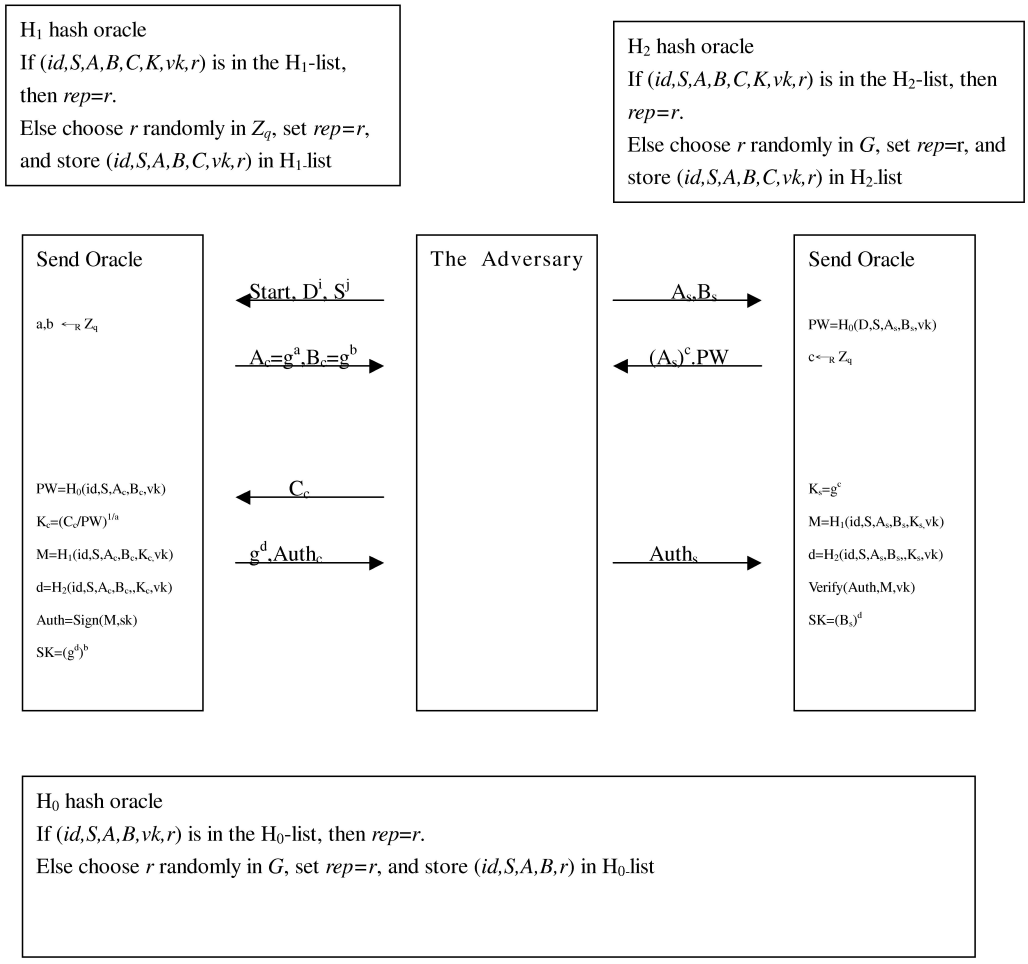


Figure 5.1: Game 1

as well. The output of H_1 and H_2 is still a element chosen randomly in G. The detail of the simulation is in Figure 5.2. The number of queries to H_0, becomes $q_{H_0} + q_{H_1} + q_{H_2}$, let $q'_{H_0} = q_{H_0} + q_{H_1} + q_{H_2}$.

Then, we halt all executions in which a collision occurs in the partial transcript (id, S, A, B, C), or a collisions occur on the output of H_0.

Since either A, B or C were simulated and thus chosen at random, the probability of collisions in the partial transcripts is at most $(q_{active}+q_{passive})^2/(2q)$, according to the birthday paradox. On the collisions of the output of H_0, by a similar reasoning, the probability of such collisions is bounded by $q'^2_{H_0}/2q$. Consequently,

$$|Pr[SUCC_2^{ss-auth}] - Pr[SUCC_1^{ss-auth}]| \leq (q_{active} + q_{passive})^2/(2q) + q'^2_{H_0}/2q,$$

$$|Pr[SUCC_2^{auth}] - Pr[SUCC_1^{auth}]| \leq (q_{active} + q_{passive})^2/(2q) + q'^2_{H_0}/2q$$

Game G_3: In this game, we add two additional secret random oracle H'_1 and H'_2 which are not accessible from the adversary. Recall that in previous games, the authenticator is the signature of of the output of H_1 and the partial Diffie-Hellman key exchange information d is the value of generator g raise to the output of H_2. In G_3, we will compute the authenticators directly from the output of H'_1 and compute the partial Diffie-Hellman key exchange information using H'_2.

After this modification, the authenticators and the partial Diffie-Hellman key exchange information become unpredictable to any adversary. Thus,

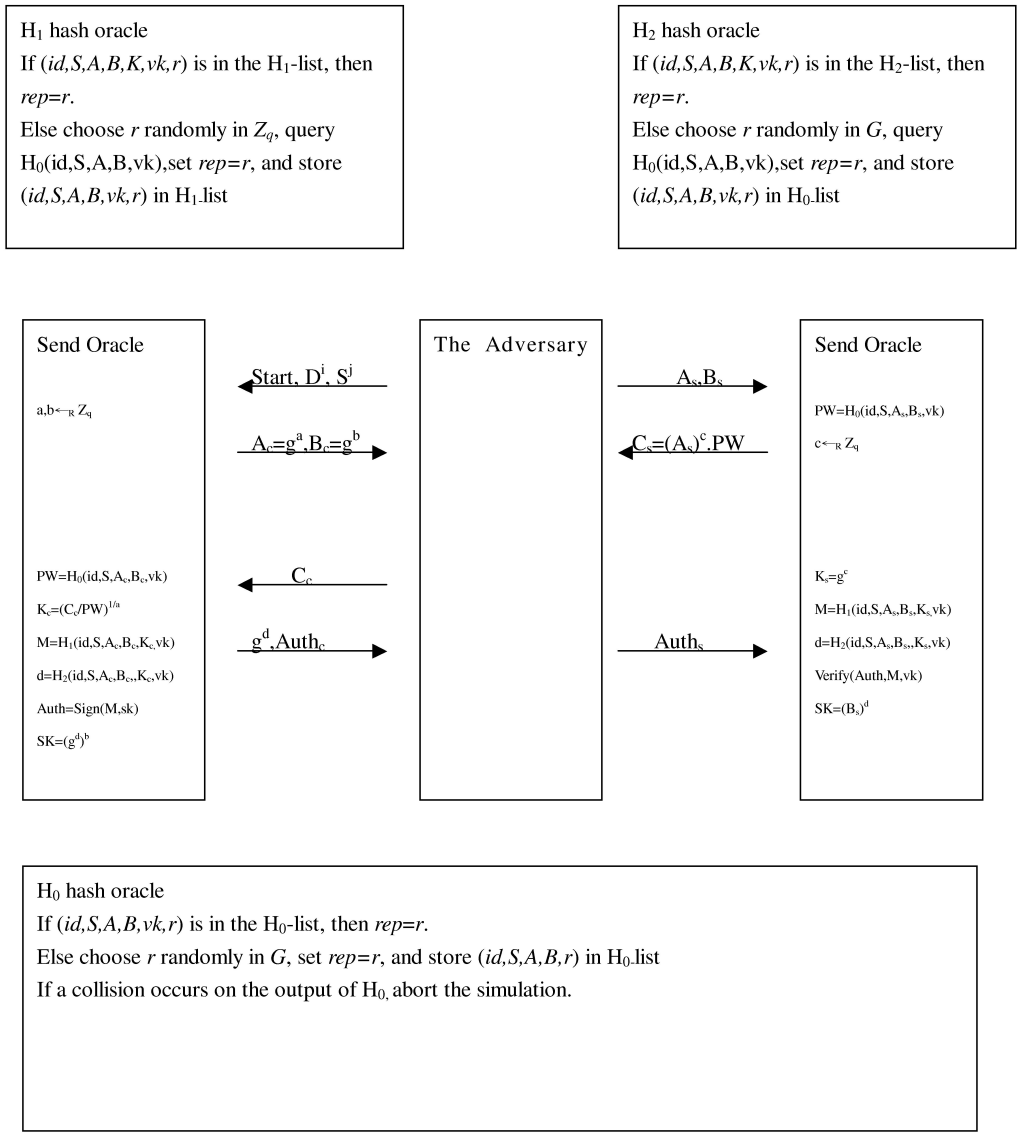


Figure 5.2: Game 2

under the DDH assumption, the agreed session keys are indistinguishable from random keys to the adversary.

Notice that PW is computed from $PW = H_0(id, S, A = g^a, B, vk)$. We can find that the common secret $K = (C/PW)^{-a}$ depends only on A, B, C and on the verify key vk shared by the participants. This implies that G_3 and G_2 are indistinguishable as long as the adversary does not explicitly query H_1 or H_2 on input $(id, S, A = g^a, B, C, (C/PW)^{-a}, vk)$. Thus, games G_3 and G_2 are indistinguishable unless the following event $AskH$ occurs

- $AskH$ $(id, S, A = g^a, B, C, (C/PW)^{-a}, vk)$ has been queried by the adversary to H_1 or H_2 for some transcript $((id, A, B), (S, C), (Auth))$ with extra information g^d.

Then we have

$$|Pr[Succ_2^{ss-auth}] - Pr[Succ_3^{ss-auth}]| \leq Pr[AskH]$$

$$|Pr[Succ_2^{auth}] - Pr[Succ_3^{auth}]| \leq Pr[AskH]$$

Before to show that $Pr[AskH]$ is small enough, notice that by those modifications above, we no longer need to know the value K nor to compute the value K' either, because we don't use them to compute the authenticator and the partial Diffie-Hellman key exchange informations. Thus we can simplify our simulations on computing C_s . The detail of simulation can be found in Figure 5.3

Although the real password is not used, the original response of S, $PW \cdot A_s^c$ and the modified response of S, A_s^c are perfectly indistinguishable.

Since the authenticator is computed by a random oracle that is kept secret from the adversary. This mean that, for each execution of the protocol, the adversary cannot guess the authenticator better than at random, unless the same partial transcript (id, S, A, B, C) appeared in some another session with a real instance. But this case, the partial transcript collision, has been excluded in G_2.

Under the DDH assumption, because the adversary can not know anything about the partial Diffie-Hellman key exchange information which comes from a secret random oracle, he can distinguish the real session key and the random key only with negligible probability. Thus we can conclude that

$$Pr[Succ_3^{ss-auth}] \leq \frac{1}{2} + \frac{Adv_{\mathbb{G},g}^{DDH}(t)}{2}$$

The case for the event $Succ_3^{auth}$ is similar. Thus,

$$Pr[Succ_3^{auth}] \leq \frac{1}{q}$$

Game G_4: In this game, we want to introduce a random challenge (g^x, g^{xy}) into our simulation, where x and y is chosen randomly in Zq. In particular, we will try to compute g^y from (g^x, g^{xy}).

We insert the challenge (g^x, g^{xy}) into a random instance of the simulation of the device D. In particular, we choose a random device instance, D_α, uniformly and randomly in the set of all device instances involved in the

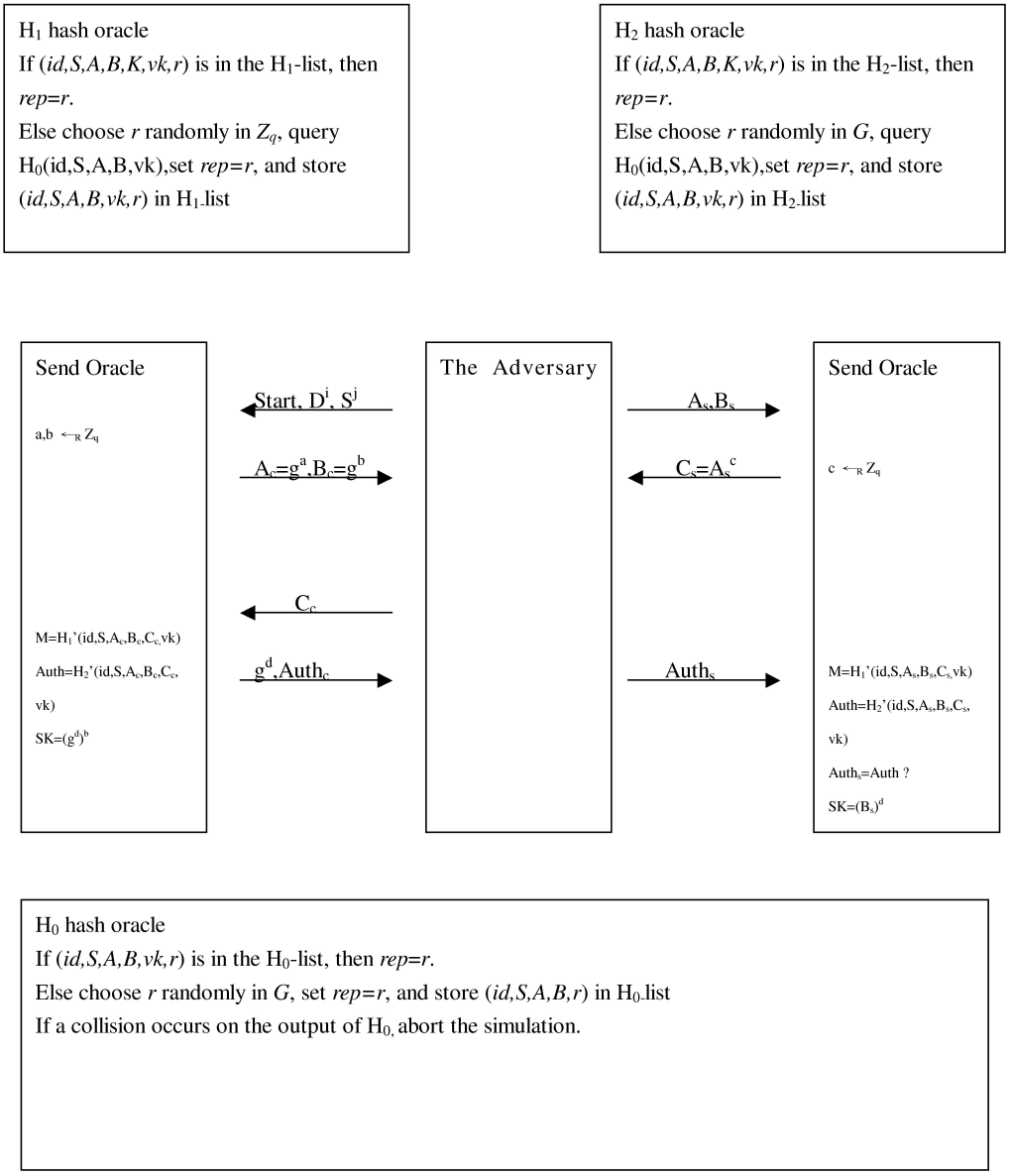


Figure 5.3: Game 3

simulation. Then we use g^x to replace g^a in the D_α instance. Since x is a random element chosen from Z_q, D_α is indistinguishable from the other device instances from the adversary.

We introduce the other part of challenge, g^{xy} into the simulation of the hash oracle H_0. When the input of H_0, (id, S, A, B, vk) meets $A = g^x$, we answer the hash oracle query as following : We first select a random value v in Zq, then set the response of H_0 to g^{xv} probability 1/2 or set the the response of H_0 to $g^{xv} \cdot g^{xy}$ with probability 1/2.

When $A \neq g^x$, the behavior of H_0 is the same as the original one. When $A = g^x$, one can find thatthe output of H_0 is still a random element in $\mathbb{G}$. Its execution should be still indistinguishable from an execution of the original one.

Notice that once we introduce the random challenge (g^x, g^{xy}) into our simulation, for any partial transcript $(A = g^a, B, C)$ used by an device instance(and thus $A = g^a$ was simulated), we can bound the probability that there exist two distinct passwords pw_1, pw_2 (and correspondingly values vk_1, vk_2 and PW_1,PW_2) such that $(id, S, A, B, C, (C/PW_1)^{-a}, vk_1)$ and $(id, S, A, B, C, (C/PW_2)^{-a}, vk_2)$ have both been queried to H_1 or H_2 oracle. Formally, define the event

- $ColH$ - which occurs if for some partial transcript $(A = g^a, B, C)$ used in a communication by a device instance, there exist two valid element $PW_1 = H_0(id, S, A, B, vk_1)$ and $PW_2 = H_0(id, S, A, B, vk_2)$, such that

both $(id, S, g^a, B, C, (C/PW_1)^{-a}, vk_1)$ and $(id, S, g^a, B, C, (C/PW_2)^{-a}, vk_2)$ have been queried to H_1 or H_2.

Then we claim that $Pr[CollH] \leq 2(q_{active}+q_{passive})(q_{H_1}+q_{H_2})^2 Adv_{G,g}^{CDH}(t+\epsilon)$

Proof. Since the partial transcript is used by a device instance, we have that $A = g^a = g^x$ with probability $1/(q_{active} + q_{passive})$. Suppose to be in the execution where g^x is used, then we have $PW_1 = H_0(D, S, A, B, vk_1) = g^{xv_1}$ or $g^{xv_1} \cdot g^{xy}$,and $PW_2 = H_0(D, S, A, B, vk_2 = g^{xv_2}$ or $g^{xv_2} \cdot g^{xy}$. With probability $1/2$, PW_1 and PW_2 were generated from different forms. (one from g^{xv} and the other from $g^{xv} \cdot g^{xv}$. Wz.o.l.g, let $PW_1 = g^{xv_1}$ and $PW_2 = g^{xv_2} \cdot g^{xy}$). By the definition, both $(id, S, g^a, B, C, (C/PW_1)^{-x}, vk_1)$ and $(id, S, g^a, B, C, (C/PW_2)^{-x}, vk_2)$ have been queried. Now consider the value

$$\frac{(C/PW_1)^{-x}}{(C/PW_2)^{-x}} = (\frac{PW_2}{PW_1})^{-x} = (\frac{g^{xv_2} \cdot g^{xy}}{g^{xv_1}})^{-x} = \frac{g^{v_2} \cdot g^y}{g^{v_1}}$$

Then we can compute $g^y = (\frac{(C/PW_2)^{-x}}{(C/PW_1)^{-x}}) \cdot g^{v_1} \cdot g^{-v_2}$.

It tells that if the event $CollH$ happens, one can compute g^y by first computing all possible answers and choosing a random one for the output. The probability that g^y has been outputted correctly is $\frac{1}{2\cdot(q_{active}+q_{passive})\cdot(q_{H_1}+q_{H_2})^2}$.

We halt the simulation and claim the adversary is successful if $CollH$ occurs. Then we have

$$|Pr[Succ_4^{ss-auth}] - Pr[Succ_3^{ss-auth}]| \leq 2(q_{active}+q_{passive})(q_{H_1}+q_{H_2})^2 Adv_{G,g}^{CDH}(t+\epsilon)$$

$$|Pr[Succ_4^{auth}] - Pr[Succ_3^{auth}]| \leq 2(q_{active} + q_{passive})(q_{H_1} + q_{H_2})^2 Adv_{G,g}^{CDH}(t + \epsilon)$$

where ϵ is the time for $O((q_{H_1} + q_{H_2})^2)$ group operations.

The detail of the simulation is in Figure 5.4.

Game G_5: Since the password is actually never used in the simulation, we may choose it at the very end of the simulation. Thus, the entire simulation is basicly independent from the chosen password. In fact the security of the protocol only depends on the public parameters.

Before evaluating the probability of the event $AskH$, notice that we have excluded the collisions of partial transcript, the events $AskH$ can be split into three mutually-exclusive sub-events:

- $AskH-Passive$: it means that the transcript $((id, A = g^a, B), (S, C = g^{ac}), (Auth, g^d))$ comes from an execution between instances of servers and devices and the tuple $(id, S, A = g^a, B, C = g^{xc}, (C/PW)^{-a}, vk)$ have been queried to H_1 or H_2. With probability of $1/(q_{active}+q_{passive})$, this transcript is used by $D\alpha$ $(A = g^x)$ so that $(C/PW)^{-x}$ is in the hash record. Notice that PW can be g^{xv} or $g^{xv} \cdot g^{xy}$.

  With probability of $1/2$, $PW = g^{xv} \cdot g^{xy}$, consider the value

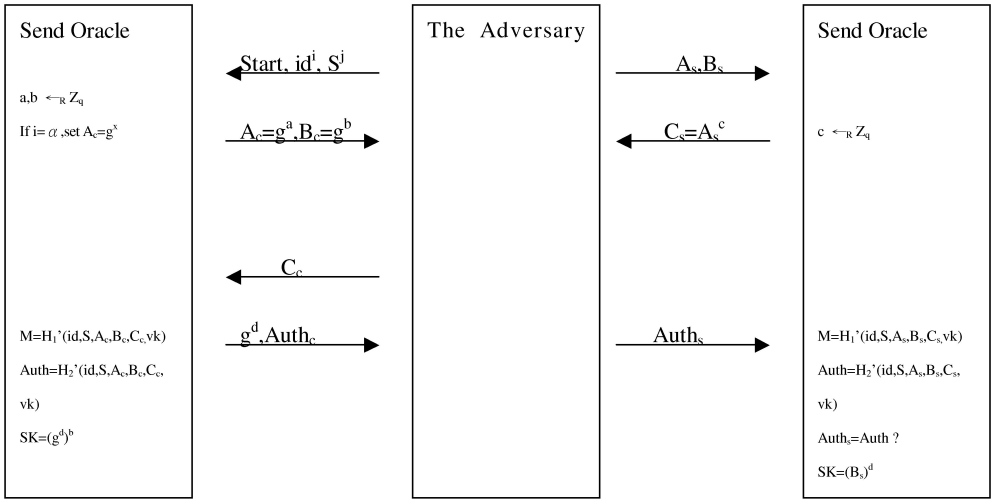
  $$(\frac{C}{PW})^{-x} = (\frac{g^{xc}}{g^{xv}g^{xy}})^{-x} = \frac{g^c}{g^v g^y})$$

  Then we can compute $g^y = \frac{g^c}{g^v \cdot (C/PW)^{-x}}$.

  Since we can compute all possible value in $H_1 - list$ and $H_2 - list$ and randomly select one of them for the answer. There are at most $(q_{H_1}+q_{H_2})$ such possible answers. It follows that $Pr[AskH-Passive] \leq$

H_1 hash oracle
If (D,S,A,B,K,vk,r) is in the H_1-list, then $rep=r$.
Else choose r randomly in G, query H_0(D,S,A,B,vk),set $rep=r$, and store (D,S,A,B,vk,r) in H_1-list

H_2 hash oracle
If (D,S,A,B,K,vk,r) is in the H_2-list, then $rep=r$.
Else choose r randomly in G, query H_0(D,S,A,B,vk),set $rep=r$, and store (D,S,A,B,vk,r) in H_2-list



H_0 hash oracle
If (id,S,A,B,vk,r) is in the H_0-list, then $rep=r$.
Else If $A=g^x$, toss a random coin, set $rep=r=g^{xv}$ with probability 1/2, set $rep=r=g^{xv}.g^{xy}$ with probability 1/2
Else choose v randomly in G, set $rep=r$, and
Store (id,S,A,B,r) in H_0-list
If a collision occurs on the output of H_0, abort the simulation.

Figure 5.4: Game 4

$2(q_{active} + q_{passive})(q_{H_1} + q_{H_2})Adv_{G,g}^{CDH}(t + \epsilon')$ where ϵ' is the time to compute $(q_{H_1} + q_{H_2})$ possible answers, $O((q_{H_1} + q_{H_2})$ group operations.

- $AskH-WithDevice$: it means the transcript $((id, A, B), (S, C), (Auth, g^d))$ comes from an execution in which a device instance is attended, but (S, C) has not been sent by and instance of servers. This means that A, B has been simulated but C has been created by the adversary. Since we excluded the event $CollH$, for any (A, B, C) involved in a transcript with an device instance, there is at most one pw (and correspondingly vk) such that $PW = H_0(id, S, A, B, vk)$ and the corresponding hash query is made: the probability over a random password chosen at the very end only is pw less than q_{active}/N. So we have

$$Pr[AskH - WithDevice] \leq q_{active}/N$$

- $AskH-WithServer$: it means the transcript $((id, A, B), (S, C), (Auth, g^d))$ comes from an execution in which a server instance is attended, but (id, A, B) has not been sent by and instance of servers. This means that C has been simulated but A, B has been created by the adversary. A success query on this input may correspond to an attack where the adversary tries to impersonate the device. But each authenticator sent by the adversary can be related to at most one pw. But we choose and set vk, sk at the very end only, we have :

$$Pr[AskH - WithServer] \leq q_{active}/N$$

Now we can bound event $AskH$ by

$$Pr[AskH] \leq 2(q_{active}+q_{passive})(q_{H_1}+q_{H_2})Adv^{CDH}_{G,g}(t+\epsilon')+q_{active}/N+q_{active}/N$$

where ϵ' is the time for $O((q_{H_1}+q_{H_2})$ group operations.

Combine all the equations above, one gets

$Adv^{ss}_{PAKE-SPD}(A) \leq Adv^{DDH}_{\mathbb{G},g}(t) + 2 \cdot (+\frac{1}{q} + 4(q_{active} + q_{passive})(q_{H_1} + q_{H_2})^2 Adv^{CDH}_{G,g}(t+\epsilon) + 4(q_{active}+q_{passive})(q_{H_1}+q_{H_2})Adv^{CDH}_{G,g}(t+\epsilon') + \frac{4q_{active}}{N} + \frac{4(q_{active}+q_{passive})^2}{2q} + \frac{2q'^2_{H_0}}{2q})$

and

$Adv^{auth}_{PAKE-SPD}(A) \leq \frac{1}{q} + 2(q_{active} + q_{passive})(q_{H_1} + q_{H_2})^2 Adv^{CDH}_{G,g}(t+\epsilon) + 2(q_{active}+q_{passive})(q_{H_1}+q_{H_2})Adv^{CDH}_{G,g}(t+\epsilon') + \frac{2q_{active}}{N} + (q_{active}+q_{passive})^2/(2q) + q'^2_{H_0}/2q.$

where ϵ is the time for $O((q_{H_1} + q_{H_2})^2)$ group operations, ϵ' is the time for $O((q_{H_1} + q_{H_2})$ group operations, and $q'_{H_0} = q_{H_0} + q_{H_1} + q_{H_2}$.

Forward Security

As the following theorems states, **PAKE-SPD** can provide forward security, it means that even if the adversary has corrupted some user and got his password, the past communication, namely the session key exchanged before, is still semantic security to the adversary.

Theorem 2 (Forward Security) *Consider the protocol **PAKE-SPD** over $\mathbb{G}$ which is a cyclic group of prime order q, generated by g. Let Dict be an uniformly distributed dictionary of size N. For any adversary A within a*

time bound t, with less than q_p passive eavesdropping, and ask q_{H_0}, q_{H_1}, q_{H_2} hash queries to H_0, H_1, H_2 respectively, we have :

$$Adv^{fs}_{PAKE-SPD}(A(pw)) \leq Adv^{DDH}_{G,g}(t)+2\cdot(q_p(q_{H_0}+q_{H_1})Adv^{CDH}_{G,g}(t+O(1))+q^2_{H_0}/2q)$$

where $O(1)$ is used to choose a random answer from the H_1 or H_2 list.

Proof. Recall that to model the forward security, we provide the password for the adversary but disallow the adversary to query *Send* oracle, since the adversary can not communicate with the parties involved in the past transcript.

As in Theorem 1, the proof goes from a sequence of game reductions. For each game G_n, we define an event $Succ^{fs}_n$ corresponding to the case in which the adversary break the forward security (to distinguish a session key used by some fresh instance from a random key). Formally, $Succ^{fs}_n$ occurs if the adversary correctly guesses the bit involved in the $Test$ query.

Game G_0: The is the real protocol in the random oracle model, which starts by choosing a random pw and then give ps to the adversary. The key pair (vk, sk) is generated from pw. By the definition, we have

$$Adv^{fs}_{PAKE-SPD}(A(pw)) = 2 \cdot Pr[Succ^{fs}_0] - 1$$

Game G_1: In this game, we modify G_0 by simulating all oracles as in the real attack. This game is perfectly indistinguishable from the real game, Thus

$$Pr[Succ^{fs}_1] = Pr[Succ^{fs}_0]$$

Game G_2: Now, we halts the simulation if there is a collision on the output of H_0. According to the birthday paradox, the probability of this collision is bounded by $q_{H_0}^2/2q$. Consequently,

$$|Pr[Succ_2^{fs}] - Pr[Succ_1^{fs}]| \leq q_{H_0}^2/2q$$

Game G_3: We now modify the way to generate the partial Diffie-Hellman key exchange information and the authenticator. In particular, both them are now generated randomly, do not depend on the transcript or the password. Since the partial Diffie-Hellman key exchange information and the authenticator are both the output of the random oracles H_1 and H_2 and A can not query $Send$ oracle. This implies that G_3 and G_2 are indistinguishable as long as the adversary does not explicitly query H_1 or H_2 on input $(id, S, A = g^a, B = g^b, C, (C/PW)^{-a}, vk)$. Let $AskH$ denote the event that adversary do such query, Formally, we define

- $AskH$ - $(id, S, A = g^a, B = g^b, C, (C/PW)^{-a}, vk)$ has been queried by the adversary to H_1 or H_2 for some transcript $(id, A, B), (S, C), (g^d, Auth)$.

Thus,

$$|Pr[Succ_3^{fs}] - Pr[Succ_2^{fs}]| \leq Pr[AskH]$$

After this modification, we knot that b is simulated and kept secret to the adversary and the partial Diffie-Hellman key exchange information, g^d, is randomly generated, the question for the adversary to distinguish the session

key g^{bd} from a random key is exactly a DDH problem, Thus we have

$$Pr[Succ_3^{fs}] \leq \frac{1}{2} + \frac{Adv_{G,g}^{DDH}(t)}{2}$$

Game G_3: To show that $Pr[AskH]$ is small enough, we need to introduce a random challenge (g^x, g^{xy}) into our simulation, where x and y is chosen randomly in Z_q. In particular, we will try to compute g^y from (g^x, g^{xy}).

We insert the challenge into some random partner instances. In particular, let (D_α, S_β) be the random partner instances we select. Then we use g^x to replace $A = g^a$ in (D_α and use $g^{xy} \cdot PW$ to replace $C = g^{ac} \cdot PW$ in S_β. One can find such such replacement give the same distribution of the transcript. Thus, the partner instances (D_α, S_β) are indistinguishable from the other partner instances.

If $AskH$ occurs, it means that the adversary has queried $(id, S, A = g^a, B = g^b, C, (C/PW)^{-a}, vk)$ to either H_1 or H_2 for some transcript $((id, A, B), (S, C), (g^d, Auth))$ With the probability of $1/q_p$, this transcript is generated by (D_α, S_β). Thus we have $A = g^x$ and $C = g^{xy} \cdot PW$. Further, $(C/PW)^{-x} = g^y$ and we get the answer for the random challenge.

Thus, if $AskH$ occurs, one can answer the random challenge by randomly choosing a answer from H_1 or H_2 list. Under the CDH assumption, we have

$$Pr[AskH] \leq q_p(q_{H_0} + q_{H_1}) Adv_{G,g}^{CDH}(t + O(1))$$

where $O(1)$ is used to choose a random answer from the H_1 or H_2 list.

Combine all the equations above, one gets

$$Pr[Succ_0^{fx}] \leq Adv_{G,g}^{DDH}(t) + 2 \cdot (q_p(q_{H_0} + q_{H_1})Adv_{G,g}^{CDH}(t + O(1)) + q_{H_0}^2/2q)$$

where $O(1)$ is used to choose a random answer from the H_1 or H_2 list.

Password Protection

As the following theorems states, the public computer can not learn anything about user's password after some executions of the protocol.

Theorem 3 (Password Protection) *Consider the protocol **PAKE-SPD** over $\mathbb{G}$ which is a cyclic group of prime order q, generated by g. Let Dict be an uniformly distributed dictionary of size N. For any adversary A within a time bound t, with less than q_p executions of the protocol, and ask q_{H_0}, q_{H_1}, q_{H_2} hash queries to H_0, H_1, H_2 respectively, we have :*

$$Pr[Succ_0^{pp}] \leq \frac{1}{N} + 2(q_p)(q_{H_1} + q_{H_2})Adv_{G,g}^{CDH}(t + \epsilon) + \frac{q_p^2}{2q} + \frac{q_{H_0}'^2}{2q}$$

where ϵ is the time cost for $O(q_{H_1} + q_{H_2})$ group operations and $q'_{H_0} = q_{H_0} + q_{H_1} + q_{H_2}$.

Proof. First notice that the only randomness which the public computer may decide is the choice of $B = g^b$. We will use similar game reductions as we used to prove Theorem 1. In particular, we define games G_0, G_1, G_2 in the same manner as in the proof of Theorem 1. By a similar reasoning, one gets

$$Pr[SUCC_1^{pp}] = PR[SUCC^{pp}]$$

$$|Pr[SUCC_2^{pp}] - Pr[SUCC_1^{pp}]| \leq \frac{q_p^2}{2q} + \frac{q_{H_0}'^2}{2q}$$

where $q'_{H_0} = q_{H_0} + q_{H_1} + q_{H_2}$ and the event $SUCC_n^{pp}$ occurs if the adversary correctly guess the password in the game G_n.

In the game G_3, since we control the behaviors of the device and the server, we can find that the common secret $K = K' = g^c$ do not depend on B. By a same reasoning, games G_3 and G_2 are indistinguishable unless event $AskH$ happens. Notice that in G_3, the adversary can compute the session key by $(g^d)^b$, but what we care is the protection of user's password pw. We define the same event $AskH$, thus,

$$|Pr[SUCC_3^{pp}] - Pr[SUCC_2^{pp}]| \leq Pr[AskH]$$

Game G_4 is the same as in the proof of Theorem 1. However, we don not need to exclude the event $ColH$ anymore because A can not control the device or the server. Thus,

$$Pr[SUCC_4^{pp}] = Pr[SUCC_3^{pp}]$$

Game G_5 is also the same as in the proof of Theorm 1. Since in each transcript $((id, A, B), (C, S), (Auth, g^d)$, A and C are simulated, the only element that the adversary can control is B. Thus, we can make sure that $AskH-WithDevice$ and $AskH-WithServer$ will not occurs. Consequently,

$AskH = AskH - Passive$ and one get $Pr[AskH - Passive] = 2(q_{active} + q_{passive})(q_{H_1} + q_{H_2})Adv_{G,g}^{CDH}(t+\epsilon)$ from a same reasoning in Theorem 1, where ϵ is the time cost for $O((q_{H_1} + q_{H_2}))$ group operations.

Recall that in G_5, the choice of the password is at the very end of the simulation. Thus the adversary will outputs his guess before we choose the real password. It means that $Pr[Succ_5^{pp}] = 1/N$, where N is the size of the dictionary .

Combine all equations, we have

$$Pr[Succ_0^{pp}] \leq \frac{1}{N} + 2(q_p)(q_{H_1} + q_{H_2})Adv_{G,g}^{CDH}(t+\epsilon) + \frac{q_p^2}{2q} + \frac{q_{H_0}'^2}{2q},$$

where ϵ is the time cost for $O(q_{H_1} + q_{H_2})$ group operations and $q'_{H_0} = q_{H_0} + q_{H_1} + q_{H_2}$.

On the Lost of the Secure Device Notice that we store only $sk' = x + t$ and $vk' = g^{x+t}$ in the secure device, where t is a random value chosen from Z_q. In particular, the original password-corresponding key pair (vk, sk) is masked by t. t should be chosen each time a secure device is registered for a user.

If an adversary has stolen user's secure device, he can obtain (vk', sk') and use this device to perform password authentication. However, he can not get user's password or (vk, sk) since he don't know the value t. once the user find his device lost, he can perform the password authentication and login the server without the secure device, then he can tell the server to

revoke the lost secure device by deleting $(sk = x + t, vk = g^{x+t}$. Thus the adversary can not login the system anymore. Finally, the user can register a new secure device by securely share a new random $t' \in Z_q$ with the server. $(sk = x + t', vk = g^{x+t'}$ will be the new masked key pair.

Chapter 6

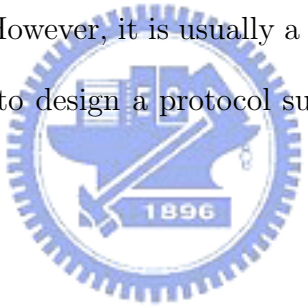
Conclusions

For the combination of the traditional password authentication and the mobile device, we give a solution by presenting a three-parties password-based authenticated key exchange protocol named **PAKE-SPD**. We modified the security model in [CPP04] to fit our scenario and discussed some security notions, including the semantic security and the forward security of session keys, the protection of user passwords and the authentication. The main idea of these security notions is to prevent anyone to gain any useful information about user's passwords or session keys. We prove the security of our protocol **PAKE-SPD** formally in the random oracle, assuming DDH and CDH holds. Consequently, in the execution of **PAKE-SPD**, the session keys, the authentication and the passwords are protected well.

When the authentication is performed by the secure device, one can find that the authentication data used in the protocol is $(sk' = x + t, g^{x+t})$. $sk = x = f(pw)$ and password pw is chosen from a small dictionary of size N, but t is chosen randomly from Z_q. Then $sk' = x + t$ is actually a random

element in Z_q. In the other word, we extend the size of the dictionary with the randomness t, from a relative small size of N, to the size of Z_q. Thus, **PAKE-SPD** can against the online dictionary attack if we force the authentication must be performed by the secure portable device.

A main drawback of PAKE-SPD is the computational cost of the devices. Since a mobile device is made for a user to carry with him. it should be lightweight and then have a low computational power. In **PAKE-SPD**, it requires about three power operations in a modular group, three hash operation and one sign operation. We can reduce the power operations by pre-computing [BPV98]. However, it is usually a memory-speed trade-off. A better solution should be to design a protocol such that less operations are required for the device.



Bibliography

[ACFP05] Michel Abdalla, Olivier Chevassut, Pierre-Alain Fouque, and David Pointcheval. A simple threshold authenticated key exchange from short secrets. In *Advances in Cryptology - ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 566–584. Springer, 2005.

[BM92] Steven M. Bellovin and Michael Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *IEEE Symposium on Security and Privacy*, pages 72–84, 1992.

[BM93] Steven M. Bellovin and Michael Merritt. Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise. In *ACM Conference on Computer and Communications Security*, pages 244–250, 1993.

[BMP00] Victor Boyko, Philip D. MacKenzie, and Sarvar Patel. Provably secure password-authenticated key exchange using diffie-hellman. In *Advances in Cryptology - EUROCRYPT 2000*, volume 1807

of *Lecture Notes in Computer Science*, pages 156–171. Springer, 2000.

[BPR00] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In *Advances in Cryptology - EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 139–155. Springer, 2000.

[BPV98] Victor Boyko, Marcus Peinado, and Ramarathnam Venkatesan. Speeding up discrete log and factoring based schemes via precomputations. In *Advances in Cryptology - EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 221–235, 1998.

[CBH05] Kim-Kwang Raymond Choo, Colin Boyd, and Yvonne Hitchcock. Examining indistinguishability-based proof models for key establishment protocols. In *Advances in Cryptology - ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 585–604. Springer, 2005.

[CPP04] Dario Catalano, David Pointcheval, and Thomas Pornin. Ipake: Isomorphisms for password-based authenticated key exchange. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 477–493. Springer, 2004.

[CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key en-

cryption. In *Advances in Cryptology - EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64. Springer, 2002.

[GL01] Oded Goldreich and Yehuda Lindell. Session-key generation using human passwords only. In *Advances in Cryptology - CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 408–432. Springer, 2001.

[GL03] Rosario Gennaro and Yehuda Lindell. A framework for password-based authenticated key exchange. In *Advances in Cryptology - EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 524–543. Springer, 2003.

[KOY01] Jonathan Katz, Rafail Ostrovsky, and Moti Yung. Efficient password-authenticated key exchange using human-memorable passwords. In *Advances in Cryptology - EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 475–494. Springer, 2001.

[KR06] Vladimir Kolesnikov and Charles Rackoff. Key exchange using passwords and long keys. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006*, volume 3876 of *Lecture Notes in Computer Science*, pages 100–119. Springer, 2006.

[MPS00] Philip D. MacKenzie, Sarvar Patel, and Ram Swaminathan. Password-authenticated key exchange based on rsa. In *Advances in Cryptology - ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 599–613. Springer, 2000.

[MSJ02] Philip D. MacKenzie, Thomas Shrimpton, and Markus Jakobsson. Threshold password-authenticated key exchange. In *Advances in Cryptology - CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 385–400. Springer, 2002.

[RG03] Mario Di Raimondo and Rosario Gennaro. Provably secure threshold password-authenticated key exchange. In *Advances in Cryptology - EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 507–523. Springer, 2003.

Appendix A

PAKE-SPD in the Symmetric Password Model

One can modify the protocol described in chapter 4 to get a simplified protocol which is executed in the symmetric password Model. The basic idea is to use the message authentication code to replace the signature.

Password Setting in the Symmetric Password Model

Let p be a large prime number such that the discrete logarithm problem defined in Z_p^* is hard and let $\mathbb{G} \in Z_p^*$ be a cyclic subgroup of prime order q, where g is a random generator of $\mathbb{G}$. We assume that for each server, a user has a identity id and corresponding password pw chosen from the dictionary. We then use a random string t to enhance the original pw and get $pw' = f(pw, t)$ as the enhanced password, where f is some hash function.

In this protocol, each user only remember his identity id and password pw, and his secure portable device carries the enhanced password pw'. The server has both user's password pw and the enhanced password pw'.

Password-base Authenticated Key Exchange with the Secure Portable Device in the Symmetric Password Model

The description of our protocol is given in Figure A.1, where H_0, H_1, H_2 are random oracles. One can notice that in the environment with the secure portable device, the execution of the protocol actually uses only the enchanced password pw'. The original password pw is not used.

Password-base Authenticated Key Exchange without the Secure Portable Device in the Symmetric Password Model

In a similar manner, the user should key-in his identity and password on the public computer if he did not bring the secure portable device. Thus, the public computer can use this password to execute the protocol.The description of the protocol is given in Figure A.2

The Security of Password-base Authenticated Key Exchange with the Secure Portable Device in the Symmetric Password Model

We can use a similar proof for **PAKE-SPD** in the asymmetric password model to prove the security of **PAKE-SPD** in the asymmetric password model. In particular, one can use another secure random oracle H_1'' to replace H_1' in the proof. H_1'' takes the same input as H_1' but the output of H_1'' is not a random signature but a random message authenticated code. The message authenticated code which generated by a secret random oracle is still random to the adversary. Thus, the other parts of the security analysis are the same.

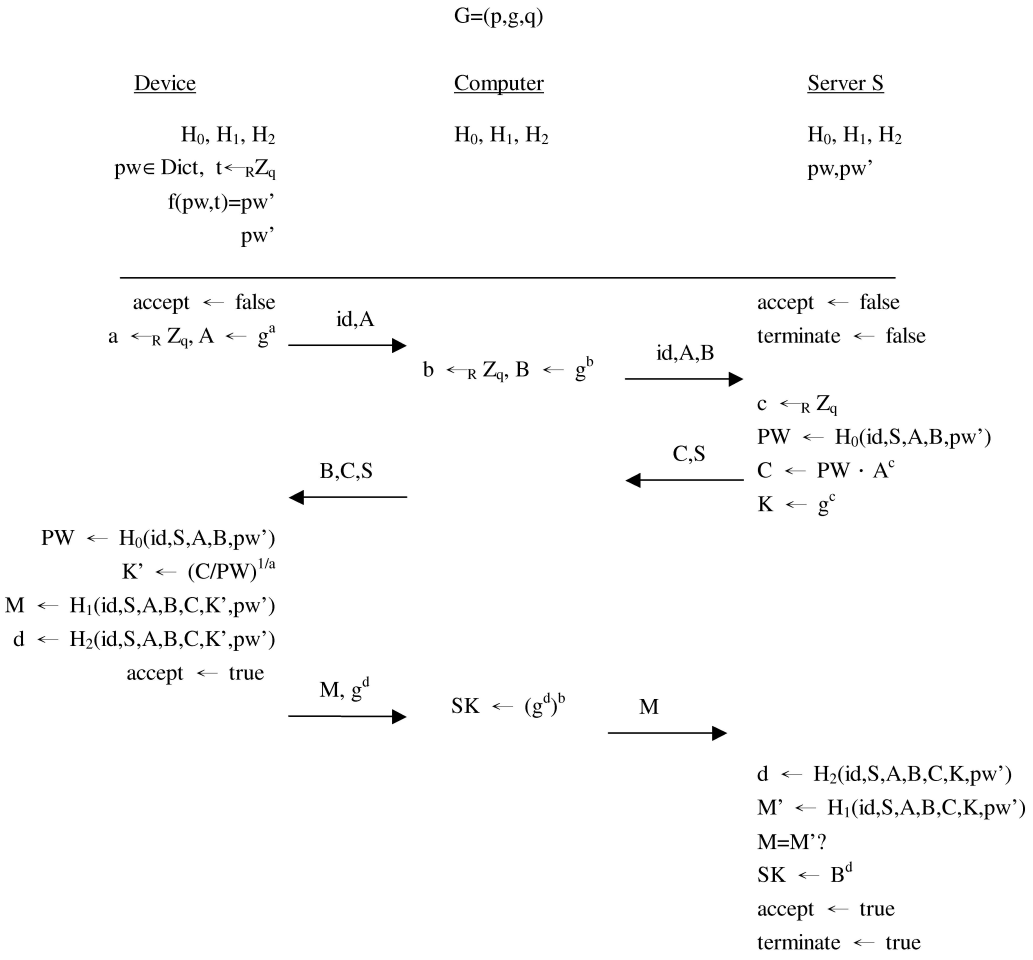


Figure A.1: Password-base Authenticated Key Exchange with the Secure Portable Device in the Symmetric Password Model

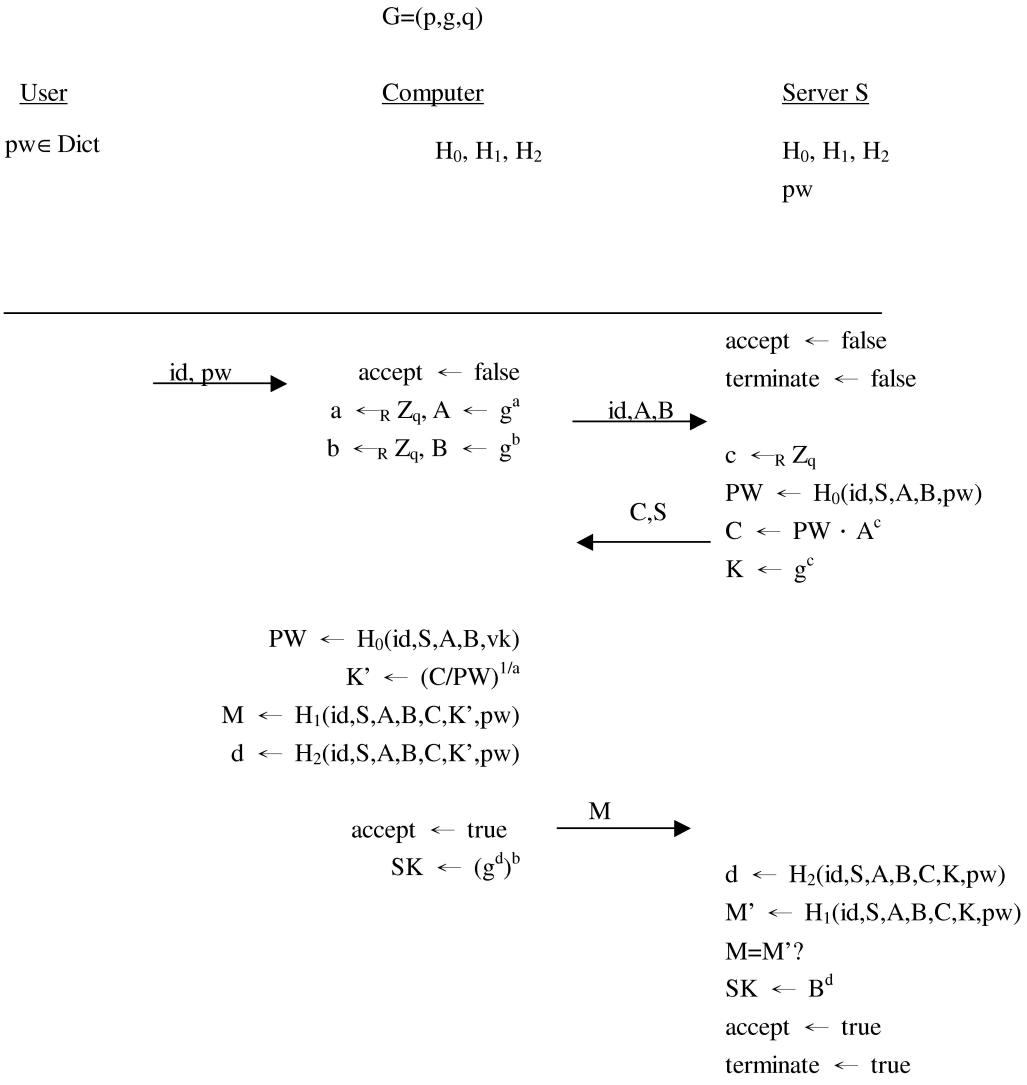


Figure A.2: Password-base Authenticated Key Exchange without the Secure Portable Device in the Symmetric Password Model

In the asymmetric password model, the server does not exactly know user's password but a transformed password. To get user's password, the server must perform offline dictionary attack on the transformed password. This kind of attack can be prevented if the user chose his password carefully. In fact, to choose a strong password is recommended for the password authentication today. There are many methods to test if a password is strong enough and one can make sure that this password does not appear in any existing dictionaries. However, in the symmetric password model, the server holds user's password. Thus not matter how strong the password is, we can not prevent that a server may use user's password to login some other server. The user should choose different and irrelative passwords between different servers.