An enhanced password authenticated key agreement protocol for wireless mobile network*

Zhigang Gao[1,2] and Dengguo Feng[1]

[1] State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China

[2] National Engineering Research Center of Information Security, Beijing 100190, China

{Zhigang2005, Feng}@is.iscas.ac.cn

Abstract. Password-based Authenticated Key Agreement (PAKA) protocols are widely used in wireless mobile networks, however many existing PAKA protocols have security flaws. In the 3GPP2 network, there are several PAKA protocols proposed to enhance the security of the Authentication Key distribution mechanism which is subjected to the Man-In-The-Middle attack. We point out the security flaws of such protocols in [4,5] and give two practical attacks on them. Moreover we propose an enhanced PAKA protocol that can resist undetectable on-line and off-line password guessing attacks, and formally analyze its security in the Random Oracle model. In addition, we consider a special version of Diffie-Hellman problem called Degenerate Diffie-Hellman problem and propose two assumptions called Computational and Decision Degenerate Diffie-Hellman assumption which are as difficult as CDH assumption and DDH assumption respectively.

Keywords: Password; Authentication; Key Agreement; Wireless Mobile Network.

1 Introduction

With the rapid development of wireless technology and applications, wireless communications become more and more popular in people's life. At the same time, security problems become important issues to be considered. Unlike wired networks which can resist part of attacks by physical access restrictions, everyone in the valid areas where are covered by radio access points can access the network resources if there is no available entity authentication mechanism. In the 3GPP2 network, the OTASP [18,19] (Over the Air Service Provisioning) is designed to enable and expedite the authentication and authorization procedures, by which potential wireless service subscribers can activate (i.e., become authorized for) new wireless services or current subscribers can request changes in their existing services, without the intervention of a third party or parties. One of the primary objectives of OTASP is to provide the Mobile Stations with a secure authentication key to facilitate authentication.

* Supported by the National Natural Science Foundation of China under Grant No. 60803129; The National High-Tech Research and Development Plan of China under Grant Nos. 2007AA120404， 2007AA120405(863); The Next Generation Internet Business and Equipment Industrialization Program under Grant No. CNGI-09-03-03.

* This work is accepted by Inscrypt2009.

However, the OTASP is not completely secure since it is subject to a Man-In-The-Middle attack. Recently some protocols [3-5] are developed to enhance the security of the Authentication key distribution mechanism of the OTASP. But these schemes do not achieve their design goals. We point out the security flaws of such schemes and give efficient attacks on them. We also give an enhanced PAKA protocol for wireless mobile network using elliptic curve arithmetic which can be used in both the 3GPP2 network and the Wireless Local Area Network.

1.1 Our Contributions

1. We point out that the attack proposed by Chang et al. [5] on Lu et al.'s [4] scheme does not hold, and present an off-line password guessing attack on Chang et al.'s protocol which is based on the Lu et al.'s scheme. Moreover, we show that Lu et al.'s scheme is not secure enough too and give an undetectable on-line password guessing attack [1] on their scheme;
2. We propose an improved PAKA protocol, which is efficient and secure against the off-line and undetectable on-line password guessing attacks, based on Lu et al.'s [4] scheme. Our protocol can be applied to CDMA2000 networks to enhance the authentication key distribution procedure, and can be used in Wireless Local Area Network under the EAP [12] framework too;
3. We consider a special version of Diffie-Hellman problem called Degenerate Diffie-Hellman (DDH) problem, and prove that the Computational DDH assumption is equivalent to the Computational DH assumption.

2 Related Work

In this paper, we focus on the PAKA protocols for wireless mobile networks. Recently, there are several efficient and practical PAKA protocols, such as schemes in [3-5,14-17]. However the protocols in [14-17] require certificates and the Public Key Infrastructure (PKI) which is expensive to construct and maintain. In contrast with the RSA-based cryptology and PKI, elliptic curve cryptology is more efficient. Especially in China, the elliptic curve cryptology has been adopted in the standard for Wireless Local Area Network and would be used in encryption and access control. Next we give a review and analysis of these protocols.

Firstly, Sui et al. [3] found out that Seo and Sweeney's simple authenticated key agreement protocol [2] suffers a reflection attack and loses its authentication capability. Then they developed an improved authenticated key agreement protocol with perfect forward secrecy using elliptic curve cryptography. They claim that their protocol eliminates the disadvantages of SAKA [2] and provides identity authentication, key validation, and perfect forward secrecy. They also show how their proposed protocol can be included in the 3GPP2 specifications for OTASP to improve the Authentication Key, which is the master key in IS-95 and CDMA2000 mobile networks, distribution.

Then, Lu et al. [4] point out the security flaw of Sui et al.'s method. Their protocol can not resist the off-line password guessing attack. So the adversaries can guess a password and verify the supposition by the messages transformed in the first and second step. Finally the attacker can retrieve the real password. Based on Sui et al.'s protocol, Lu et al. propose a new password-based authenticated key agreement protocol, which solves the problems of protocol in [3] and also could be used in 3GPP2 networks.

Furthermore, Chang et al. [5] argue that Lu et al.'s protocol [4] can not resist the parallel guessing attack. According to their paper, they lunch this attack by guessing Q_{B2}^{*}, which is a point value transformed in the second step of the protocol in [4], The point value is shown as (a,b), where $a,b \in [0,n-1]$ and n is a secure large prime. Therefore, they can guess all cases of (a,b) in $O(n^2)$ time, which is polynomial time. But, we should note that the variable n is a secure large prime, and generally the variable n should have 160 bits lengths at least. So Chang's attack is a power exponent time algorithm and can not solve by polynomial time adversaries in fact. Beside this, the modified version protocol proposed by Chang et al. involved the off-line password guessing attack, and we will show the practical attack in the later sections.

Besides the mistake of the Chang et al.'s article which is shown above, We also show that all of the protocols mentioned in this section suffer the undetectable on-line password guessing attack [1] in Section 4 and Section 5. The PAKA protocols which suffer undetectable on-line password guessing attack can not discover and consequently can not forbid such password guessing attacks. The servers in that situation actually act as oracles which assist the attacker to verify the passwords.

3 Preliminaries

Our protocol is based on the Elliptic Curve Cryptology (ECC), however it can be used in other algebraic structure in which the DDH assumption holds. In this section we list the assumptions with style of elliptic curve cryptology which will be used in our security proof. Let F_m be a finite field and $\mathbb{E}$ be an elliptic curve over F_m. Let $P \in \mathbb{E}$ is a point in $\mathbb{E}$ with a large prime order q.

Assumption 1. Computational Diffie-Hellman (CDH) Assumption. For $a,b \in Z_q^*$, $P \in \mathbb{E}$, given $(P, aP, bP, \mathbb{E})$, computing abP is hard.

Assumption 2. Decision Diffie-Hellman (DDH) Assumption. For $a,b,z \in Z_q^*$, $P \in \mathbb{E}$, given $(P, aP, bP, zP, \mathbb{E})$, deciding whether $zP = abP$ or not is hard.

The assumptions mentioned above are based on the Elliptic Curve Discrete Logarithm Problem (ECDLP). The Discrete Logarithm Problem also holds in most elliptic curves (while the DDH assumption is easy in some elliptic curves). The best known methods to solve ECDLP are Pollard approach and Pohlig-Hellman method. They are both fully exponential, while the best known methods to solve the Integer Factorization Problem (IFP) and the Discrete Logarithm Problem (DLP), on which most of the non-ECC cryptosystems rely, are sub-exponential.

Based on the ECDLP, we propose a modified version of Diffie-Hellman problem called Degenerate Diffie-Hellman problem. In the new problem, the two variable aP and bP, which are defined in

Assumption 1, are equal, and we need to compute the element a^2P instead of abP or distinguish a^2P from a random element in $\mathbb{E}$. Like the situation of Diffie-Hellman problem, we propose two assumptions which both based on the Degenerate Diffie-Hellman problem.

Assumption 3. Computational Degenerate Diffie-Hellman (CDDH) Assumption. For $x \in Z_q^*$, $P \in \mathbb{E}$, given $(P, xP, \mathbb{E})$, computing x^2P is hard.

Assumption 4. Decision Degenerate Diffie-Hellman (DDDH) Assumption. For $x, z \in Z_q^*$, $P \in \mathbb{E}$, given $(P, xP, zP, \mathbb{E})$, deciding whether $zP = x^2P$ or not is hard.

We emphasize that the Degenerate Diffie-Hellman problem is also hard in other algebraic structures which the Discrete Logarithm problem still exists, although we have adopted the Elliptic Curve arithmetic to describe it. Next, we discuss the difficulty of the CDDH assumption and the DDDH assumption.

Theorem 1. The CDDH assumption is equivalent to the CDH assumption.

Proof: Firstly, we prove that if the CDH assumption holds then the CDDH assumption holds. To achieve this, we only need to prove if the CDDH problem is easy then the CDH assumption is no longer tenable. Assuming that we can solve the CDDH problem in polynomial time, given a CDH instance $(P, aP, bP, \mathbb{G})$, we compute abP by the following step.

$$
\begin{aligned}
&step\,1.\text{compute } (a+b)P = ap + bp \\
&step\,2.\text{compute } a^2P \text{ and } b^2P \text{ using CDDH} \\
&step\,3.\text{compute } (a+b)^2 P \text{ using CDDH} \\
&step\,4.\text{compute } 2abP = (a+b)^2 P - a^2P - b^2P \\
&step\,5.\text{compute } abP = 2^{-1}(2abP)
\end{aligned}
$$

Secondly, we prove that if the CDDH assumption holds then the CDH assumption holds. To prove this, we only need to prove that if the CDH problem is easy then the CDDH assumption is no longer tenable. Assuming that we can solve the CDH problem, we show that we can solve the CDDH problem. Given a CDDH instance $(P, xP, \mathbb{G})$, let the values of both variables aP and bP, which is described in assumption 1, be xP, then we can compute x^2P using CDH assumption. □

Through the equality between the CDDH assumption and the CDH assumption can be proved easily, the relationship between the DDDH assumption and the DDH assumption is not obvious and it is still an open issue.

4 Attack on Lu et al.'s Protocol

In this section, we review Lu et al.'s [4] protocol and give a practical undetectable on-line password guessing attack.

4.1 Review of Lu et al.'s Protocol

As proposed in [4] and cited in [5], the two participants in Sui et al.'s protocol are called as Alice(A) and Bob(B), and they share a low-entropy password S which is selected from a uniformly distributed dictionary D of size $|D|$. The symbol E denotes an elliptic curve defined over a finite field F_m with large group order, and a point P with large prime order n is selected randomly from E. The value t is derived from the password S in a predetermined way, which is uniformly distributed in Z_n^*. H is a secure one-way function. The set $\{E, n, P, D, H\}$ is sent to Alice and Bob as public parameters.

Step 1: A first selects a random number $d_A \in Z_n$, and computes $Q_{A1} = (d_A + t)P, Q_{A2} = d_A^2 P$. Then A sends the message (Q_{A1}, Q_{A2}) to B.

Step 2: B selected two random numbers $d_{B1}, d_{B2} \in Z_n$, and computes $Y = Q_{A1} - tP = d_A P$, $Q_{B1} = d_{B1}P + d_{B2}Y$, $Q_{B2} = d_{B1}Y + d_{B2}Q_{A2}$, and $H_B = H(A \| B \| Q_{A1} \| Q_{B1} \| Q_{B2})$. Then B sends (Q_{B1}, H_B) to A.

Step 3: After receiving (Q_{B1}, H_B), A computes $X = d_A Q_{B1} = d_{B1} d_A P + d_{B2} d_A^2 P$, and verifies whether the equation $H_B = H(A \| B \| Q_{A1} \| Q_{B1} \| X)$ holds or not. If it holds, A send $H_A = H(B \| A \| Q_{B1} \| Q_{A1} \| X)$ to B. Then A sets the session key as $K_A = X$.

Step 4: When B receives the message, B checks whether the equation $H_A = H(B \| A \| Q_{B1} \| Q_{A1} \| Q_{B2})$ holds or not. If it holds, B sets the session key as $K_B = Q_{B2}$.

4.2 Undetectable On-line Password Guessing Attack on Lu et al.'s Protocol

Let the adversary called Eve who wants to retrieve the password shared by Alice and Bob, executes the following steps:

1. Eve first guesses (maybe Eve selects it from a prepared directory) a password S', and derives t' from S' through pre-defined methods. Then Eve selects a random number $d_A \in Z_n$, computes $Q_{A1} = (d_A + t')P$, $Q_{A2} = d_A^2 P$ and sends (Q_{A1}, Q_{A2}) to Bob. This stage is mainly the same as the first step in Lu et al.'s protocol.
2. When Eve receives the message sent by Bob, she computes $X = d_A Q_{B1}$, and checks whether the equation $H_B = H(A \| B \| Q_{A1} \| Q_{B1} \| X)$ holds. If it holds, Eve guesses the password

correctly, then she responses as Alice following the protocol described in section 3.1. If it is not true, Eve repeats the process until she gets the right password, then she could response the former unfinished session.

This is a practical attack especially in the client-and-server environment. The adversary Eve can personate a client to initiates many session requests to guess and verify her guess, while the server is difficult to discover that the requester is not the person he claimed.

5 Attack on Chang et al.'s Protocol

The protocol developed by Chang et al. [5] is a slim modified version of Lu et al.'s [4]. In this section, we will demonstrate that the modification which is being carried out by Chang et al. makes their protocol be involved in a more serious situation: it can not resist the off-line password guessing attack. In addition, we note that it can not resist the undetectable on-line password guessing attack too. We do not give the detail of the undetectable on-line password guessing attack because it is similar as the attack in section 4.2.

5.1 Review of Chang et al.'s Protocol

The public parameters of Chang et al.'s protocol are the same as Lu et al.'s protocol which is described in section 4.1.

Step 1: This step is the same as the first step in section 4.1.

Step2: This step is mainly the same as the second step in section 3.1, while Bob computes H_B using Y rather than Q_{B2}. So we have $H_B = H\left(A \| B \| Q_{A1} \| Q_{B1} \| Y\right)$. The following steps changes according to the modification of H_B.

Step3: After receiving $\left(Q_{B1}, H_B\right)$, A first verifies whether the equation $H_B = H\left(A \| B \| Q_{A1} \| Q_{B1} \| d_A P\right)$ holds or not. If it holds, A send $H_A = H\left(B \| A \| Q_{B1} \| Q_{A1} \| d_A P\right)$ to B. Then A sets the session key as $K_A = d_A Q_{B1}$.

Step 4: When B receives the message, B checks whether the equation $H_A = H\left(B \| A \| Q_{B1} \| Q_{A1} \| Y\right)$ holds or not. If it holds, B sets the session key as $K_B = Q_{B2}$.

5.2 The Off-line Password Guessing Attack on Chang et al.'s Protocol

Now, we suppose that Eve listened in the communication channels, and obtained all of the messages transformed between Alice and Bob. So, Eve have the messages $\left\{Q_{A1}, Q_{A2}, Q_{B1}, H_A, H_B\right\}$. Then Eve can get the shared password S by executing an off-line password guessing attack.

Eve first guesses a password S' from D in accordance with the way she wants (may be ordered), and computes the corresponding t'. Then she computes $Y' = Q_{A1} - t'P$ and $H_B{}' = H(A \| B \| Q_{A1} \| Q_{B1} \| Y')$. Eve compares $H_B{}'$ with H_B which were sent by Bob in Step 2 of Chang et al.'s protocol described in section 4.1. If they are equal, Eve has got the right password; else she guesses another password and repeats the procedure until her guess is correct.

The upper bound of the off-line password guessing attack is $O(|D|)$.

6 Security Model for Our PAKA Protocol

To prove the security of the new protocol proposed in this paper, we extend the formal security model which is introduced in [9]. In this model, there are three classes of participants: clients, servers and the adversary. The adversary may be a legal user; however, we do not distinguish between these situations because that the adversary can corrupt any client's password.

1. Notation. In this model, we fix two sets: Clients and Servers. Each principal is either a client or a server. Let U_i denote the ith user in the client set, while S_j denote the jth server in the server set. In the special kth execution of the protocol, we use $U_{i,j}^k$ to denote the client involved in this communication, while $S_{j,i}^k$ to denote the server. Let b be a bit chosen uniformly which is used in the $Test$ query.
2. Partnering. We use SID, a session ID, to indicate a execution of the protocol, where SID is the conjunction of all messages the participant sent and received in this interaction. For example, U_i sends M_0 and receives M_1 in the first step, and sends M_2 and receives M_3 in the second step, then the SID is $M_0 \| M_1 \| M_2 \| M_3$.

Definition 1. Partner. We say that a client $U_{i,j}^k$ and a server $S_{j,i}^{k'}$ are partnered if the following conditions are met: (1) They are both accepted (if exist); (2) $U_{i,j}^k$ and $S_{j,i}^{k'}$ share the same SID; (3) No oracle besides $U_{i,j}^k$ and $S_{j,i}^{k'}$ accepts with the same SID.

3. Oracle queries. The interactions between an adversary Eve and the participants of the protocol occur via oracle queries, which model the adversary's capabilities in the real attack. All oracle queries in our model are listed in the following:

– $Execute(U_i, S_j, k)$: This oracle is used to simulate the eavesdropping attack. The output of this query is made up of the messages that are exchanged during the honest execution of the protocol. The variable k denotes the sequence number of the protocol execution.

– $Send\left(U_{i,j}^{k} / S_{j,i}^{k}, m\right)$: This oracle query enables the adversary to perform an active attack on a client or a server. According to the input message m, the oracles $U_{i,j}^{k}$ and $S_{j,i}^{k}$ execute the operations defined in PAKA protocols. Finally the query is answered with the message produced in these operations.

– $Reveal\left(U_{i,j}^{k} / S_{j,i}^{k}\right)$: The adversary uses this query to gain the session key hold by $U_{i,j}^{k}$ or $S_{j,i}^{k}$. If the entity accepted, the session key is returned; else a symbol $\perp$ is returned.

– $CorruptClient\left(U_i\right)$: This query models exposure of a client U_i's password shared with a server. It is not necessary to give another corrupt query on servers, because U_i and S_j share the same password in this model.

– $CorruptServer\left(S_j\right)$: This query models exposure of a server S_j's secret key t.

– $Test\left(U_{i,j}^{k}\right)$: This query is used to measure the semantic security of the session key of client instance $U_{i,j}^{k}$. If $U_{i,j}^{k}$ is not accepted, it return $\perp$; else it return either the session key held by $U_{i,j}^{k}$ if $b=0$ or a random key with the same distribution as the real session key. This query can be launched only once.

4. Freshness. There are two notions of freshness: with and without forward secrecy (fs) in [9]. We only consider the former, because of that we want to prove that our protocol providers perfect forward secrecy.

Definition 2. Freshness (with forward secrecy). We say that a client instance $U_{i,j}^{k}$ is fresh if the following conditions hold: (1) It has accepted; (2) No $Reveal$ queries have been made to $U_{i,j}^{k}$ or its partner; (3) If $U_{i,j}^{k}$ has been made a $CorruptClient$ query then he must not been made a $SendClient/Server$ query and vice versa.

Now, we can define the advantage of the adversary in attacking the PAKA protocol. We say that the adversary Eve wins the game define above, in semantic security scene, if she asks a single $Test$ query, $Test\left(U_{i,j}^{k}\right)$, where $U_{i,j}^{k}$ is fresh, and she outputs a single bit, b', and $b'=b$ (where *b* is the bit selected during the Test query). The PAKA advantage of the adversary is twice the probability that Eve wins, minus one.

Definition 3. Semantic Security (with forward secrecy). We say that a PAKA protocol is secure if the following conditions hold:

1. In the presence of a benign adversary, which faithfully conveys messages, both oracles always accept holding the same session key, and this key is distributed uniformly on $\{0,1\}^{k}$;

2. For any polynomial time adversary, The PAKA advantage of the adversary is negligible.

7 Our Proposed Protocol

In this section, we describe the improved password-based authentication key agreement protocol in detail. We are illumined by the method in Zhang et al.'s article [6] originally, then we apply a similar idea to our protocol to make it resist the Undetectable on-line password guessing attack. Our protocol also provides a key confirmation procedure which is necessary for analogous protocols. If there is a lack of key confirmation processes, the protocol will reach an illegality state in which both parties accepted with two different session keys.

Our protocol is based on Lu et al.'s protocol [4] and uses elliptic curve cryptology which could be computed efficient in both of mobile devices and consumer computer. We call the participants in our protocol Alice and Bob. Alice plays the role of clients and Bob of servers. Bob has an extra secret information than Alice besides a password shared with Alice. This is a little different from Lu et al.'s protocol. Our protocol is more suitable for an asymmetric authentication scene, such as the mobile network access authentication, than symmetric environments.

In the initial stage, Bob (the server) selects an elliptic curve $\mathbb{E}$ over a finite field F with large group order, and randomly selects a point P with large prime order q from $\mathbb{E}$. Bob selects a number $t \in Z_q^*$, and computes $Q = tP$. He selects three collision-resistant hash functions [7] $H_1 : D \to Z_q^*$, $H_2 : \{0,1\}^* \to \{0,1\}^l$ and $H_3 : \{0,1\}^* \to \{0,1\}^k$, where D is the set containing all the possible passwords, l is a secure parameter selected by Bob and k is the desirable length of session keys. Finally, Bob set $\{\mathbb{E}, q, P, Q, D, H_1, H_2, H_3\}$ as public parameters and keeps t secretly.

We assume that the Alice obtains the public parameters and a password S shared with the Bob in an extra register stage. Details of messages transfer are depicted as follows.

1. Alice first selects a random number $x \in Z_q^*$, and computes $T_{A1} = (x + H_1(S))P$, $T_{A2} = x^2 P$ and $T_{A3} = H_2(xQ)$. Then Alice sends $\{T_{A1}, T_{A2}, T_{A3}\}$ to Bob.

2. After receiving $\{T_{A1}, T_{A2}, T_{A3}\}$, Bob checks whether the equation $H_2(t(T_{A1} - H_1(S)P)) = T_{A3}$ holds or not. If it does not hold, Bob terminates the session and output $\perp$. Otherwise, Bob selects two random numbers $y_1, y_2 \in Z_q^*$, and computes $Y = T_{A1} - H_1(S)P = xP$, $T_{B1} = y_1 P + y_2 Y$, $T_{B2} = H_2(tY \,||\, S)$ and $H_B = H_2(A \,||\, B \,||\, T_{A1} \,||\, T_{A2} \,||\, T_{A3} \,||\, T_{B1} \,||\, T_{B2})$. Then Bob sends $\{T_{B1}, H_B\}$ to Alice.

3. When Alice receives the message, she first verifies whether the equation $H_B = H_2\left(A \| B \| T_{A1} \| T_{A2} \| T_{A3} \| T_{B1} \| H_2\left(xQ \| S\right)\right)$ holds or not. If it holds, A send $H_A = H_2\left(B \| A \| T_{B1} \| H_2\left(xQ \| S\right) \| T_{A1} \| T_{A2} \| T_{A3}\right)$ to B. Then A sets the session key as $K_A = H_3\left(A \| B \| T_{A1} \| T_{A2} \| T_{A3} \| T_{B1} \| xT_{B1}\right)$.

4. When Bob receives the third message, he checks whether the equation $H_A = H_2\left(B \| A \| T_{B1} \| T_{B2} \| T_{A1} \| T_{A2} \| T_{A3}\right)$ holds or not. If it holds, B sets the session key as $K_B = H_3\left(A \| B \| T_{A1} \| T_{A2} \| T_{A3} \| T_{B1} \| y_1 Y + y_2 T_{A2}\right)$.

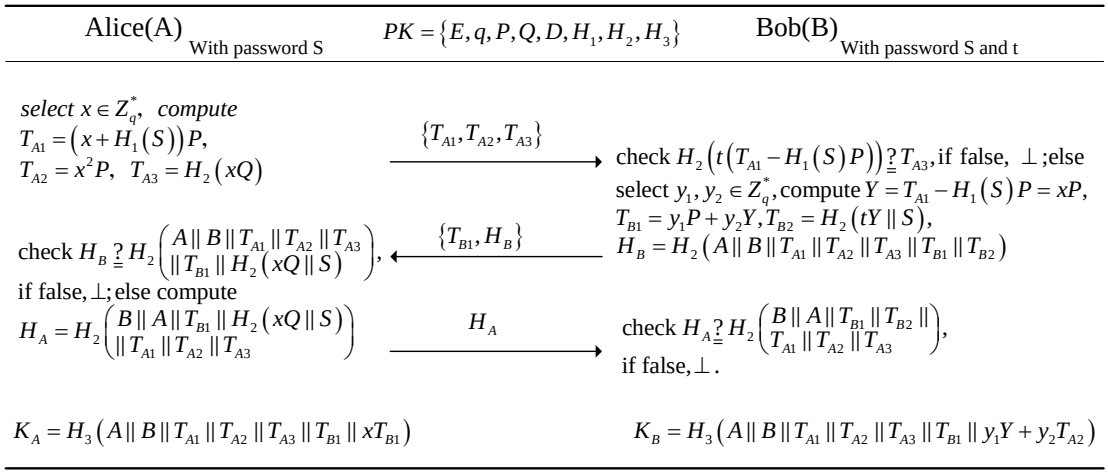

Fig. 1. Our Proposed PAKA Protocol

8 Security Analysis

In this section, we discuss the security of our protocol. First we prove that our protocol is secure (with forward secrecy) in the semantic security scene under the random oracle model [8], then we prove that our protocol can resist both undetectable on-line password guessing attack and the off-line password guessing attack.

8.1 Security with Forward Secrecy

Theorem 2. The protocol $\prod$ (denotes the protocol we proposed in this paper) is a secure (with forward secrecy) PAKA protocol if the CDH assumption, DDH assumption and DDDH assumption holds and the hash functions are modeled as random oracles.

The proof of Theorem 2 is given in Appendix A.

8.2 Resistance to The Off-line Password Guessing Attack

The security model defined in section 6 ensures that our protocol in section 7 is secure against active adversaries even he gets the passwords belongs to clients. However we should protect passwords of clients from the off-line password guessing attack in practical applications.

Theorem 3. The protocol $\prod$ is secure against the off-line password guessing attack if the DDH assumption and DDDH assumption holds and the hash functions are modeled as random oracles.

Proof: In this stage, we assume that the adversary Eve is passive, and then she do not modify the messages transformed in the channel but record them. At the end of the execution of a protocol instance, the view of Eve is $\{T_{A1}, T_{A2}, T_{A3}, T_{B1}, H_B, H_A\}$.

In the first step, Eve can guess a password S' and compute $x'P = T_{A1} - H_1(S)P$. In order to verify her guess at the password, she need to confirm whether $x'P$ and $\{T_{A2} = x^2P, T_{A3} = H_2(xQ)\}$ are match. But she can not verify her guess using T_{A2} and T_{A3} Because of that if she wants to use T_{A2} for verifying, she must solve the DDDH problem. And how to distinguish $x'P$ from T_{A3} is a DDH problem.

Eve also can not verify her guess using T_{B1}, H_B which are sent in the second step. In order to use T_{B1}, she needs to guess two random variable y_1 and y_2, however this is difficult for her. On the other hand if she wants to use H_B, she needs to compute the variable $T_{B2} = H_2(tY \| S)$. At this moment, she has $x'P = Y'$ and S', but she do not know the value of t, so she must guess the variable t which is an impossible mission for her. The analysis of H_A is similar as H_B.□

8.3 Resistance to The Undetectable On-line Password Guessing Attack

In the previous section, we prove that the adversary Eve can not get clients' password by off-line password guessing attack. Now we show that the Servers and Clients can detect the on-line password guessing attack and terminate the protocol's execution.

Theorem 4. The protocol $\prod$ is secure against the undetectable on-line password guessing attack if the CDH assumption holds and the hash functions are modeled as random oracles.

Proof: In our protocol, there are two different kinds of undetectable on-line password guessing attacks, on the Servers and on the Clients, we discuss them respectively. In this analysis, we assume that the adversary do not know the value of t.

In the former situation, the adversary Eve can perform her attacks in the first step only, because the server has nothing returned which Eve needs to use for verifying his guess in the third step. In the first step, Eve needs to construct the whole message $\{T_{A1}, T_{A2}, T_{A3}\}$. She guesses a password S' firstly,

then she selects $x' \in Z_q^*$ randomly and computes T_{A1} and T_{A2}, but she can not computes the correct $x'Q$ with $x'P$ and tP which is a computational Diffie-Hellman problem.

In the later case, Eve can and only can perform her attacks in the first step obviously. In this situation, she needs to constructs the message $\{T_{B1}, H_B\}$, and her views of the protocol execution is $\{T_{A1}, T_{A2}, T_{A3}\}$. We assume that Eve is very smart, so she will use T_{A1} to help guessing the password(if she do not, she can selects a point randomly, but the probability of that the point is equal with tY is neglectable). Firstly she guesses a password S' and computes $x'P = T_{A1} - H_1(S)P$, then she constructs $T_{B2} = H_2(tY \| S)$ with tP. However this is an impossible mission, because we assume that the CDH assumption holds. □

If the server and client receive a wrong message, he knows that there must be an adversary between them, so he can terminate the protocol execution. In a practical application, the authentication server can block the further attacks using identity and the adversary's IP address, if he detects an on-line password guessing attack.

8.4 Importance of keeping the variable t secret

In order to detect the undetectable on-line password guessing attack, we introduce an additional variable t to our scheme. We note that it is important to keep the variable t secret, because of that if the adversary knows the value of t, she can crack the password with the message $\{T_{A1}, T_{A3}\}$ sent in the first step by client. So if the value of t is exposed, the adversary can execute an off-line password guessing attack.

Actually, it is impossible to get the value of t by guessing for a polynomial time adversary. Besides this, the server can change the value of t from time to time which can be performed easily in practical systems.

9 Efficiency and Application

Our scheme designs for client-and-server systems in wireless network. The server manages the entire user's passwords and other profiles, while the client only needs to keep his password. Generally the server is more powerful than clients. In our scheme, the server needs to perform three point addition operations, six scalar multiplications and five hash operations, while the client needs to execute four scalar multiplications and six hash operations. Our scheme does not need any certificates and the support of The Public Key Infrastructure (PKI).

Comparing to the scheme in [3] which five scalar multiplication and one point addition operations are needed both for Alice and Bob, our protocol is more suitable for mobile device. Because of that, in our protocol, the client saves one scalar multiplication and one point addition operations. In contract with the scheme in [4] which needs five scalar multiplication, one point addition operations and three

hash operations for the server and three scalar multiplication and two hash operations for the client, our protocol has to execute one scalar multiplication and two hash operations more both for the server and the client. But that is indispensable to resist the undetectable on-line password guessing attack. We list the detail information in the Table 1.

Table 1. Computation Complexity and Security Properties

Schemes	Security Properties						Model	Assumption	Computing Consumption	
	ksk	fs		pass guessing		uks			Clients	Servers
		pass	t	un-on	off					
Sui[3]	√	√	n/a	×	×	×	RO	DDH	5S+1P	5S+1P
Lu[4]	√	√	n/a	×	√	√	n/a	n/a	3S+2H	5S+1P+3H
Chen[5]	√	√	n/a	×	×	×	n/a	n/a	3S+2H	5S+1P+3H
Gao	√	√	√	√	√	√	RO	DDH & DDDH	4S+5H	6S+3P+6H

fs: forward secrecy
ksk: known key secrity
uks: unknown key share
un-on: undetectable on-line password guessing attack
off: off-line password guessing attack
n/a: there is not a valid proof or schemes are not related to

S: scalar multiplication
P: point addition operation
H: hash operation
RO: Random Oracle
Gao: the protocol in this paper
√/×: resist/not resist or support/not support

In [3], Sui et al. give a method to apply their protocol to improve the Authentication Key distribution in 3GPP2 networks. Our protocol can also be used in the 3GPP2 network with the same method, and the detail of the method is described in [3]. We note that our protocol also can be applied to the Wireless Local Area Network under the Extensible Authentication Protocol (EAP) framework [12]. In the IEEE 802.11i standard, the EAP framework is the normative way to realize the access control protocols. The most popular EAP methods which were used now are EAP-AKA, EAP-PSK and EAP-TLS. However the EAP-PSK method is simple and the EAP-TLS method requires the support of PKI. Our protocol can be used in this situation to strengthen the security of WLAN access control system. The detail method to apply our protocol with EAP is similar as the method in [13], so we do not give a detail description here.

10 Conclusion

Our research focuses on the password-based authenticated key agreement protocol for wireless network. In this paper, we review the mainly PAKA protocols based on the elliptic curve cryptology without certificates and PKI. We point out the security flaw of the protocols in [4, 5] and give two practical attacks on them. By considering the security strength, computation efficiency and security properties, we proposed an enhanced PAKA protocol which can resist undetectable on-line and off-line password guessing attacks. Moreover, we prove the security of our protocol in the Random-Oracle model.

References

1. Y. Ding and P. Hoster.: Undetectable on-line password guessing attacks. ACM Operating Systems Review, vol.29, no.4, pp.77-86, (Oct. 1995)

2. Seo, D. and Sweeney, P.: Simple Authenticated Key Agreement Algorithm, Electronics Letters, Vol. 35, 1999,pp. 1073–1074.
3. Sui, A. , Hui, L., Yiu, S., Chow, K., Tsang, W., Chong, C., Pun, K. and Chan, H. “An Improved Authenticated Key Agreement Protocol with Perfect Forward Secrecy for Wireless Mobile Communication,” IEEE Wireless Communications and Networking Conference (WCNC 2005), LA USA, 2005, pp. 2088–2093.
4. Lu, R., Cao, Z. and Zhu, H.: An Enhance Authentication Key Agreement Protocol for Wireless Mobile Communication, Computer Standards and Interfaces, Vol. 29, 2007, pp. 647-652.
5. Chin-Chen, Chang. and C. Shih-Chang (2008). An Improved Authentication Key Agreement Protocol Based on Elliptic Curve for Wireless Mobile Networks. Intelligent Information Hiding and Multimedia Signal Processing, 2008. IIHMSP '08 International Conference on.
6. Zhengfeng Zhang and Dengguo Feng.: On Password-based Key Exchange with Enhanced Security. In: 4rd SKLOIS Workshop on Security Protocols, pages 132-144, Beijing, 2009
7. M. Bellare and P. Rogaway.: Collision-Resistant Hashing: Towards Making UOWHFs Practical. LECTURE NOTES IN COMPUTER SCIENCE: 470-484. (1997)
8. M. Bellare and P. Rogaway.: Random oracles are practical: a paradigm for designing efficient protocols. ACM CCS’93, pp. 62-73, (1993)
9. M. Bellare, D. Pointcheval, et al.: Authenticated key exchange secure against dictionary attacks. LECTURE NOTES IN COMPUTER SCIENCE: 139-155, (2000)
10. V. S. Miller.: Use of elliptic curves in cryptography. Advances in Cryptology Crypto 85, Lecture Notes in Computer Science. Springer-Verlag, vol. 128, 1985, pp. 417-426.
11. N. Koblitz.: Elliptic Curve Cryptosystems. Mathematics of Computation, vol. 48, issue 177, 1987, pp. 203-209.
12. B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson.: Extensible Authentication Protocol (EAP), http://www.ietf.org/rfc/rfc3748.txt, 2004
13. F. Bersani, H. Tschofenig.: The EAP-PSK Protocol: a Pre-Shared Key EAP Method draft-bersani-eap-psk-11, http://tools.ietf.org/html/draft-bersani-eap-psk-11, 2006
14. Z. Wan, R. Deng, F. Bao and A. Ananda.: Anonymous DoS-Resistant Access Control Protocol Using Passwords for Wireless Networks, Proc. 30th IEEE Conference on Local Computer Networks (LCN), pages 328-335, 2005.
15. Z. Wan, B. Zhu, R. Deng, F. Bao and A. Ananda.: DoS-Resistant Access Control Protocol with Identity Confidentiality for Wireless Networks, Proc. IEEE Wireless Communications Networking Conference (WCNC), pages 1521-1526, 2005.
16. Singhal, A., V. Garg, et al.: Analysis and Enhancement of Two Key Agreement Protocols for Wireless Networks. Communication Systems Software and Middleware, 2007. COMSWARE 2007. 2nd International Conference on, IEEE 2007
17. Teo, J. C. M. and C. H. Tan.: Security analysis and improvement of an anonymous DoS-resistant access control protocol, Information, Communications & Signal Processing, 2007 6th International Conference on, IEEE 2007
18. 3GPP2 N.S0011 v1.0, OTASP and OTAPA, http://www.3gpp2.org
19. 3GPP2 C.S0016-B v1.0, Over-the-Air Service Provisioning of Mobile Stations in Spread Spectrum Standards, Oct. 2002, http://www.3gpp2.org

Appendix A The Proof of Theorem 2

Proof: According to definition 3, we should prove that our protocol satisfies the both conditions defined in definition 3. Firstly we prove that our protocol satisfies the first condition. At the end of protocol execution, Alice and Bob compute the session key K_A and K_B respectively.

$$\because xT_{B1} = x\left(y_1P + y_2Y\right) = x\left(y_1P + y_2xP\right) = y_1xP + y_2x^2P = y_1Y + y_2T_{A2}$$
$$\therefore K_A = K_B$$

Next, we prove that our protocol satisfies the second condition. Suppose that there exists an adversary Eve against our protocol. We will construct a PPT simulator $\mathbb{S}$ that makes use of Eve to solve the DDH problem. $\mathbb{S}$ Will take BDH challenge $\left(P, aP, bP, zP, \mathbb{E}\right)$, which is defined in section 3, and outputs a guess, β', as to whether the challenge is a Diffie-Hellman tuple.

Initialization. At the initial phase, $\mathbb{S}$ construct two collections *U* and *S* where *U* contains all clients and *S* contains all servers. Without loss of generality, we assume that Eve launches q_{sess} protocol executions at most. $\mathbb{S}$ selects a number $1 \le I \le q_{sess}$. $\mathbb{S}$ guesses that the oracle $U_{i,j}^{I}$ is to be asked in the $Test$ query. Then $\mathbb{S}$ selects the public parameters $\left\{\mathbb{E}, q, P, D\right\}$ using the same way defined in section 7, and sends it to Eve.

Phase 1. Eve can issue send queries, reveal queries, hash queries, execute queries and corrupt queries. At the end of phase 1, Eve must launch a Test query on a fresh oracle which is selected by Eve randomly. On receiving a query launched by Eve, the simulator acts like this:

– $H_i\left(x\right)$. $\mathbb{S}$ maintains lists for every hash oracle. On a hash query $H_i\left(x\right)$, $\mathbb{S}$ check whether there is an entry $\left\langle x, r\right\rangle$ in $HList_i$. If it is true, returns r, else $\mathbb{S}$ selects randomly a element with the same distribution as the real word, then S saves $\left\langle x, r\right\rangle$ into $HList_i$ and returns r.

– $CorruptClient\left(U_i\right)$. $\mathbb{S}$ keeps a list $UList = \left\{\left\langle U_i, S_j, pass\right\rangle\right\}$. On a corrupt query for password, $\mathbb{S}$ check whether there is an entry $\left\langle U_i, S_j, pass\right\rangle$ in $UList$. If it is true, returns $pass$, else $\mathbb{S}$ selects randomly a password from D, then S saves $\left\langle U_i, S_j, pass\right\rangle$ into $UList$ and returns $pass$.

– $CorruptServer\left(S_j\right)$. $\mathbb{S}$ keeps a list $SList = \left\{\left\langle S_j, t\right\rangle\right\}$. On receiving this query, $\mathbb{S}$ check whether there is an entry $\left\langle S_j, t\right\rangle$ in $SList$. If it is true, returns t, else $\mathbb{S}$ selects randomly

$t \in Z_p^*$, then S saves $\langle S_j, t \rangle$ into $SList$ and returns t. We note that when $\mathbb{S}$ sets a new t for a server, he will send the correlative tP to Eve.

– $Send\left(U_{i,j}^k / S_{j,i}^k, m\right)$. According to the input message and destination oracle, there are four variants. We note that all of the hash function used in the simulation are replaced by the oracle query $H_i(x)$ defined above, and the results of $H_i(x)$ are controlled by $\mathbb{S}$.

- $Send\left(U_{i,j}^k, \lambda\right)$. This query initializes a new protocol execution. If the oracle U_i is not initialized properly, $\mathbb{S}$ selects a password for U_i and inserts the tuple $\langle U_i, S_j, pass \rangle$ into $UList$.
  – If $k = I$, $\mathbb{S}$ set $x_k = \perp$, $T_{A1} = aP + H_1(S)P$, $T_{A2} = P'$, $T_{A3} = H_2(taP)$ where $P' \in_R \mathbb{E}$;
  – Else $\mathbb{S}$ selects a random number $x_k \in Z_q^*$, and computer $T_{A1} = \left(x_k + H_1(S)\right)P, T_{A2} = x_k^2 P$ and $T_{A3} = H_2(x_k Q)$. Then $\mathbb{S}$ returns the tuple $\{T_{A1}, T_{A2}, T_{A3}\}$ and adds $\{x_k, T_{A1}, T_{A2}, T_{A3}\}$ to $U_{i,j}^k$'s transcript.
- $Send\left(S_{j,i}^k, (T_{A1}, T_{A2}, T_{A3})\right)$. If the oracle S_j is not initialized properly, $\mathbb{S}$selects a random number $t \in Z_p^*$, then $\mathbb{S}$ saves $\langle S_j, t \rangle$ into $SList$. $\mathbb{S}$ checks whether the equation $H_2\left(t\left(T_{A1} - H_1(S)P\right)\right) = T_{A3}$ holds or not. If it does not hold, S terminates the session and output $\perp$.
  – If $k = I$, $U_{i,j}^I$ and $S_{j,i}^I$ are partners, $\mathbb{S}$ set $y_{k1}, y_{k2} = \perp$, $Y = aP$, $T_{B1} = bP$, $T_{B2} = H_2(tY \| S)$ and $H_B = H_2(U_i \| S_j \| T_{A1} \| T_{A2} \| T_{A3} \| T_{B1} \| T_{B2})$;
  – Else $\mathbb{S}$ selects two random numbers $y_{k1}, y_{k2} \in Z_q^*$, and computes $Y = T_{A1} - H_1(S)P = x_k P$, $T_{B1} = y_{k1}P + y_{k2}Y$, $T_{B2} = H_2(tY \| S)$ and $H_B = H_2(U_i \| S_j \| T_{A1} \| T_{A2} \| T_{A3} \| T_{B1} \| T_{B2})$. Then $\mathbb{S}$ returns $\{T_{B1}, H_B\}$ and adds all of the message to $S_{j,i}^k$'s transcript.

- $Send\left(U_{i,j}^{k},\left(T_{B1},H_{B}\right)\right)$.
  - If $k=I$, $\mathbb{S}$ first verifies whether the equation $H_B = H_2\left(U_i \| S_j \| T_{A1} \| T_{A2} \| T_{A3} \| T_{B1} \| H_2\left(taP \| S\right)\right)$ holds or not. If it holds, $\mathbb{S}$ returns $H_A = H_2\left(S_j \| U_i \| T_{B1} \| H_2\left(taP \| S\right) \| T_{A1} \| T_{A2} \| T_{A3}\right)$ and set the state of $U_{i,j}^{k}$ as accepted;
  - Else $\mathbb{S}$ first verifies whether the equation $H_B = H_2\left(U_i \| S_j \| T_{A1} \| T_{A2} \| T_{A3} \| T_{B1} \| H_2\left(x_k Q \| S\right)\right)$ holds or not. If it does not hold, $\mathbb{S}$ returns $\perp$; else $\mathbb{S}$ returns $H_A = H_2\left(S_j \| U_i \| T_{B1} \| H_2\left(x_k Q \| S\right) \| T_{A1} \| T_{A2} \| T_{A3}\right)$. $\mathbb{S}$ computes the session key $sk = H_3\left(U_i \| S_j \| T_{A1} \| T_{A2} \| T_{A3} \| T_{B1} \| x_k T_{B1}\right)$ and set the state of $U_{i,j}^{k}$ as accepted.
- $Send\left(S_{j,i}^{k},H_{A}\right)$. $\mathbb{S}$ checks whether the equation $H_A = H_2\left(S_j \| U_i \| T_{B1} \| T_{B2} \| T_{A1} \| T_{A2} \| T_{A3} \|\right)$ holds or not. If it does not hold, $\mathbb{S}$ returns $\perp$; else $\mathbb{S}$ set the state of $S_{j,i}^{k}$ as accepted. If $k \neq I$, $\mathbb{S}$ sets the session key $sk = H_3\left(U_i \| S_j \| T_{A1} \| T_{A2} \| T_{A3} \| T_{B1} \| y_{k1} Y + y_{k2} T_{A2}\right)$.

– $Reveal\left(U_{i,j}^{k} / S_{j,i}^{k}\right)$. If $U_{i,j}^{k}$ or $S_{j,i}^{k}$ is accepted, the query is answered with the session key sk; else $\perp$ is returned.

– $Execute\left(U_i, S_j, k\right)$. On this query, $\mathbb{S}$ executes the protocol according to the send queries defined above without communicating with Eve. The condition that $k=I$ is also applied to the send queries. $\mathbb{S}$ sets the correlative lists and, Finally, the whole transcripts are returned to Eve.

– $Test\left(U_{i,j}^{k}\right)$. If $k \neq I$ or $U_{i,j}^{k}$ is not accepted, $\mathbb{S}$ aborts the simulation with failure (Event 1). If the element T_{B1} in the transcript of $U_{i,j}^{k}$ is not equal to bP, $\mathbb{S}$ aborts the simulation with failure(Event 2). $\mathbb{S}$ returns $H_3\left(U_i \| S_j \| \left(T_{A1} = aP + H_1\left(S\right)P\right) \| T_{A2} \| T_{A3} \| \left(T_{B1} = bP\right) \| zP\right)$ to Eve.

Phase 2. $\mathbb{S}$ repeats the same method it used in Phase 1, except that Eve is not allowed to reveal the target Test query or its partner oracle (if any), and A cannot corrupt party S_j.

Guess. Finally, the adversary Eve outputs a guess b' of b. $\mathbb{S}$ simply forwards the output of Eve to its challenger of the DDH problem.

According to the simulation, We claim that if $\mathbb{S}$ does not abort and Eve can not distinguish the simulation from the real attack, then the advantage of $\mathbb{S}$ against the DDH game is identical with the advantage of Eve against our protocol. Now, we evaluate the probability that $\mathbb{S}$ wins the DDH game.

Lemma 1. The probability of that Eve distinguishes the simulation from the real attack (Event 0) is neglectable.

Proof: According to the simulation, when the session sequence number k is not equal to the number I which is selected by $\mathbb{S}$, the simulation is simulated perfectly. So Eve can not distinguish the simulation from the real attack. If $k = I$, the only chance of Eve is to discover the difference between a^2P and a random element P' which is transformed in step 1 with aP. However this is a Decision Degenerate Diffie-Hellman (DDDH) problem which holds by assuming. □

Lemma 2. The probability of that Event 2 occurs is $\dfrac{Adv_E^{CDH}+1}{2|D|}$.

Proof: because the oracle $U_{i,j}^k$ which is be tested is fresh, so there is no corrupt query on $U_{i,j}^k$ and its partner S_j. Eve does not have the password of U_i. If Event 2 occurs, there must be a valid message (T_{B1}, H_B) forged by Eve. If the check executed in $Send\left(U_{i,j}^k, (T_{B1}, H_B)\right)$ is true with a different $T_{B1}{}'$, Eve must forge a right element which is equal to $H_2(taP \,||\, S)$ with $aP + H_1(S)P$ and tP. The advantage of that Eve guess the right password and compute $H_2(taP \,||\, S)$ is $\dfrac{Adv_E^{CDH}}{|D|}$, where $Adv_E^{CDH} = 2\Pr[E \text{ wins CDH}] - 1$.□

We are now ready to calculate the $\mathbb{S}$'s advantage ε' in solving the DBDH problem:

$$\begin{aligned}
&\Pr[\mathbb{S} \text{ not abort}] = \Pr\left[\overline{Event0} \wedge \overline{Event1} \wedge \overline{Event2}\right] \\
&= \Pr\left[\overline{Event2} \mid \overline{Event0} \wedge \overline{Event1}\right] \Pr\left[\overline{Event1} \mid \overline{Event0}\right] \Pr\left[\overline{Event0}\right] \\
&= \left(1 - \frac{Adv_E^{CDH}+1}{2|D|}\right) \frac{1}{q_{sess}} \left(1 - \frac{Adv_E^{DDDH}+1}{2}\right) \\
&= \frac{\left(2|D| - Adv_E^{CDH} - 1\right)\left(1 - Adv_E^{DDDH}\right)}{4q_{sess}|D|}
\end{aligned}$$

Suppose that Eve's advantage in winning the simulation game is ε, where is not a neglectable variable, at least, then we have

$$\varepsilon' \geq \frac{\left(2|D| - Adv_E^{CDH} - 1\right)\left(1 - Adv_E^{DDDH}\right)\varepsilon}{4q_{sess}|D|} .\square$$