An Elliptic Curve Based Authenticated Key Agreement Protocol for Wireless Security

SeongHan Shin, Kazukuni Kobara, and Hideki Imai

Research Center for Information Security (RCIS)
National Institute of Advanced Industrial Science and Technology (AIST)
1-18-13 Sotokanda, Chiyoda-ku, Tokyo 101-0021 Japan
{seonghan.shin,kobara_conf,h-imai}@aist.go.jp
http://www.rcis.aist.go.jp/

Abstract. When we consider wireless security, it is strongly preferable to use password-based authentication and the elliptic curve based Diffie-Hellman protocol since the former provides a user-friendly authentication method and the latter is an efficient key agreement protocol. However, this combination does not necessarily guarantee security against off-line dictionary attacks (especially, "partition attacks"). In this paper, we propose an elliptic curve based authenticated key agreement (called EC-AKA) protocol that is secure against partition attacks as well as suitable for the following situation: (1) a client, who communicates with many different servers, remembers only one password and has insecure devices; (2) the counterpart servers are not perfectly secure against several attacks; (3) neither PKI (Public Key Infrastructures) nor TRM (Tamper-Resistance Modules) is available. The EC-AKA protocol is secure under the elliptic curve Diffie-Hellman problem in the random oracle model. We also show that the EC-AKA protocol achieves more strengthened security properties and efficiency compared with the existing protocols (employed in the IEEE 802.1x).

1 Introduction

The rapid advance of wireless technology has brought much attention from many researchers who, at the same time, have expressed concerns about security. As we know, the most fundamental security goals are authentication that is a means to verify who is communicating with whom or whether a party is a legitimate one, and confidentiality that is a means to protect messages exchanged over open networks (i.e., the Internet). One of the ways to achieve such security goals is to use an authenticated key agreement (AKA) protocol by which the involving parties authenticate each other and then share a common session key to be used for their subsequent secure channels. Up to now, many AKA protocols have been proposed where some take advantage of PKI (Public Key Infrastructure) and others are based on a secret shared between the parties (e.g., human-memorable password).

Compared to the wired networks, wireless ones typically place severe restrictions on designing such cryptographic protocols. Main obstacles include: client's

Y. Wang, Y. Cheung, and H. Liu (Eds.): CIS 2006, LNAI 4456, pp. 767–777, 2007.

mobile devices have constraints on available power consumption, followed by restriction of computing power; mobile devices are easy to be lost or stolen due to a holder's carelessness; wireless communications are more prone to interception than wired ones; communication bandwidth is already limited; and it is difficult to keep some information secure on mobile devices, and so on.

For efficiency, one can use elliptic curve groups whose use in public key cryptography was first proposed by Koblitz [1] and Miller [2] [1]. This is because public key schemes based on elliptic curve groups typically have lower processing requirements, and can achieve the same level of security with considerably shorter key sizes than counterparts based on the more traditional RSA and standard discrete logarithm settings. Such elliptic curve cryptographic systems and protocols are ideal for wireless environments where processing power, time and/or communication bandwidth are at a premium.

Therefore, when we consider wireless security it is strongly preferable to use password-based authentication and the elliptic curve based Diffie-Hellman protocol since the former provides a user-friendly authentication method and the latter is an efficient key agreement protocol. However, this combination sometimes results in insecurity against a special kind of off-line dictionary attacks known as "partition attacks". That is, the direct elliptic curve analogs of password-based AKA protocols are insecure against partition attacks (see [3]). Here is a simple example: given an affine point $(\mathcal{X}, \mathcal{Y})$ on an elliptic curve E, the $\mathcal{Y}$-coordinate may be used for an attacker to exclude invalid password candidates by executing a password-based AKA protocol once so that the attacker can sieve out the correct password at a logarithm rate.

1.1 Our Contributions

The first motivation of this work is to thwart partition attacks in an elliptic curve based AKA protocol. And the second motivation comes from the fact that the leakage of stored secrets is a more practical risk rather than breaking a well-studied cryptographic hard problem (e.g., the discrete logarithm problem). In order to deal with this problem, we consider the following situation: (1) a client, who communicates with a variety of servers, remembers only one password and has *insecure* devices (e.g., mobile phones or PDAs) with built-in memory capacity; (2) the counterpart servers are not perfectly secure against several attacks (e.g., virus or hacker); (3) neither PKI nor TRM is available.

In this paper, we propose an AKA (called EC-AKA) protocol based on the elliptic curve Diffie-Hellman protocol that is an analog of the original Diffie-Hellman protocol [4]. The EC-AKA protocol is suitable for the above situation in that it is secure against leakage of stored secrets from a client and servers, respectively, as well as secure against partition attacks. We prove that the EC-AKA protocol is provably secure in the random oracle model with the reduction to the elliptic curve Diffie-Hellman problem. Moreover, we show that the EC-AKA

[1] They observed that the discrete logarithm on elliptic curves over finite fields appeared to be intractable and hence ElGamal encryption and signature schemes have natural counterparts on these curves.

protocol achieves more strengthened security properties and efficiency compared with the existing password-based AKA protocols (e.g., [3,5]). Note that the authenticity of the EC-AKA protocol is based on password and an additional stored secret which might seem to be similar to that of EAP-FAST. However, the obvious distinction between the two protocols is that the EC-AKA protocol remains secure even if the stored secret on client's side is leaked out to an attacker while EAP-FAST does not.

2 An Elliptic Curve Based Authenticated Key Agreement (EC-AKA) Protocol

2.1 Preliminary

Here we consider an elliptic curve E defined over the field $GF(p^m)$, with either $p \geq 2^{160}$ and $m = 1$ or $p = 2$ and $m \geq 160$, where $q = p^m$ and p is a prime. For example, the curve in short Weierstrass form is

$$E : \mathcal{Y}^2 = \mathcal{X}^3 + a\mathcal{X} + b. \tag{1}$$

As shown in the literature [7], we can define an additive (abelian) group in the set of points on this curve (taken together with the point at infinity O). Let G_1 and G_2 be two generators of order q (i.e., $qG_1 \equiv qG_2 \equiv O \bmod p^m$) chosen from the points on E. This is the group where the elliptic curve discrete logarithm problem (EC-DLP) is defined: given two points G_1 and H on E it is hard to find an integer e such that $H \equiv e \cdot G_1$. On the other hand, the e multiple of G can be readily computed by using a method similar to the "square-and-multiply" for exponentiation in $GF(p)$.

Let k denote the security parameter for hash functions (say, 160 bits). Let N be a dictionary size of passwords (say, 36 bits for alphanumerical passwords with 6 characters). Let $\{0,1\}^\star$ denote the set of finite binary strings and $\{0,1\}^k$ the set of binary strings of length k. Let "$||$" denote the concatenation of bit strings in $\{0,1\}^\star$. Let us define secure one-way hash functions. While $\mathcal{H} : \{0,1\}^\star \rightarrow \mathbb{Z}_q^\star \backslash \{1\}$ denotes a full-domain hash (FDH) function, the other hash functions are denoted $\mathcal{H}_j : \{0,1\}^\star \rightarrow \{0,1\}^k$ for $j = 1, 2, 3$ and 4. Here $\mathcal{H}$ and $\mathcal{H}_j$ are distinct random functions one another. Let $\mathcal{C}$ and $\mathcal{S}$ be the identities of client and server, respectively, with representing each ID $\in \{0,1\}^\star$ as well.

2.2 The Protocol

In this subsection, we propose the EC-AKA protocol in detail (see Fig. 1 and 2).

During the initialization phase, server $\mathcal{S}$ sends its elliptic curve parameter param, which is generated in a form (E, q, G_1, G_2), to the client. The latter picks a secret value s_1 randomly chosen from $\mathbb{Z}_q^\star$ and registers securely a verification data v_1 to server $\mathcal{S}$ where pw is the client's password. Then client $\mathcal{C}$ remembers his password pw *and* additionally stores the secret value s_1 as well as the parameter param on insecure devices that may happen to leak s_1 and param in the end.

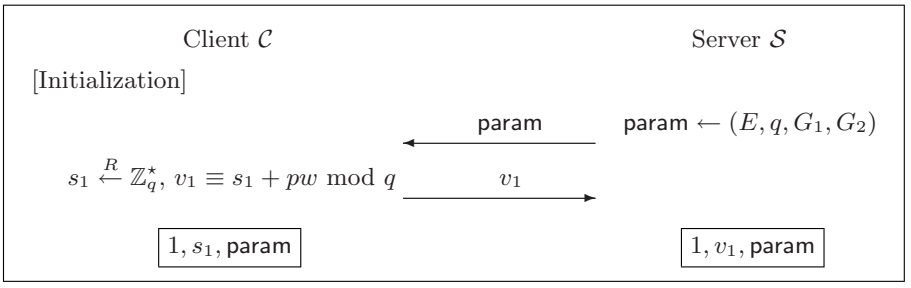

Fig. 1. The initialization of EC-AKA protocol where the enclosed values in rectangle represent stored secrets of client and server, respectively

The server $\mathcal{S}$ also stores the verification data v_1 and its parameter param on its databases both of which may be leaked out. Finally, they set a counter j as 1.

In the j-th ($j \geq 1$) execution of the EC-AKA protocol, client $\mathcal{C}$ should recover the verification data v_j by adding the secret value s_j with the password pw. With a randomly chosen value x from $\mathbb{Z}_q^\star$, the client computes the Diffie-Hellman public value X and calculates Z using a mask generation function as the addition of X with $W \cdot G_2$ where W is a full-domain hash of (j, v_j). Then client $\mathcal{C}$ sends $(\mathcal{C}, j, Z)$ to server $\mathcal{S}$. If the received counter j is incorrect, the server terminates the protocol. Otherwise, server $\mathcal{S}$ extracts X' from this masked Diffie-Hellman public value Z with $W \cdot G_2$. If the resultant value is a quadratic non-residue, the server terminates the protocol. The latter computes not only the Diffie-Hellman public vale $Y \equiv y \cdot G$ with a randomly chosen value y from $\mathbb{Z}_q^\star$ but also the keying material $K_{\mathcal{S}} \equiv y \cdot X'$ that is used to compute its authenticator $V_{\mathcal{S}}$ and a session key SK_j. Upon receiving $(\mathcal{S}, Y, V_{\mathcal{S}})$ from the server, client $\mathcal{C}$ computes the keying material $K_{\mathcal{C}}$ from Y and then generates his authenticator $V_{\mathcal{C}}$ and a session key SK_j, as long as the authenticator $V_{\mathcal{S}}$ is valid, before sending $V_{\mathcal{C}}$ to server $\mathcal{S}$. If the authenticator $V_{\mathcal{C}}$ is valid, server $\mathcal{S}$ actually computes a session key SK_j. At the end of the j-th protocol execution, client $\mathcal{C}$ (resp., server $\mathcal{S}$) refreshes s_j (resp., v_j) to a new one $s_{(j+1)}$ (resp., $v_{(j+1)}$) for the next session.

Remark 1. In order to prevent the invalid-curve attacks [6], both of client and server should check that a received point does indeed lie on the elliptic curve (e.g., by using formulas for the addition law that use both coefficients a and b of the equation of the elliptic curve).

3 Security

First, we give a clue about why the proposed EC-AKA protocol is secure against partition attacks.

Before the EC-AKA protocol execution, the client and the server can agree on the $\mathcal{Y}$-coordinate on curve E with a single bit $(+, -)$. Here we assume that the sign is $+$. Let us think of Z in the first flow of Fig. 2: $Z \equiv X + W$. In this

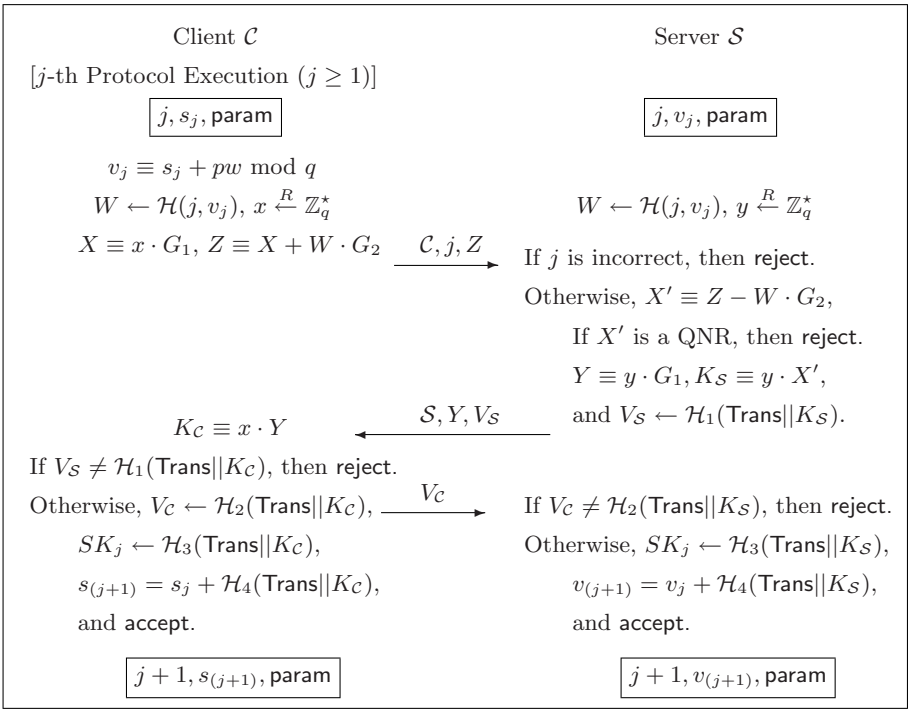

Fig. 2. The j-th execution of EC-AKA protocol where the enclosed values in rectangle represent stored secrets of client and server, respectively, and $\mathsf{Trans} = \mathcal{C}||\mathcal{S}||j||Z||Y$

case, an attacker can try the possible password candidates in order to get the demasked value X'. If X' is a quadratic non-residue, the attacker can exclude the password candidates used. From Hasse's theorem [7], the number of such values X' is in the range $[(q+1)/2-\sqrt{q}, (q+1)/2+\sqrt{q}]$. Hence the attacker can reduce the dictionary size by roughly half with such partition attack. That means the password can be sieved out to be correct one, given a number of protocol runs, at a logarithmic rate to the dictionary size.

However, the client in the EC-AKA protocol sends Z computed with an additional mask $W \cdot G_2$. Suppose an attacker who tries a guessed password on Z. The attacker cannot determine whether the guessed password is correct or not since all of the Legendre symbols $\left(\frac{X'}{q}\right)$ are quadratic residues. Thus the EC-AKA protocol is secure against partition attacks. This technique used in the EC-AKA protocol is quite different from [3] in that the latter ensures that any candidate $\mathcal{X}$-coordinate observed by an attacker is valid by utilizing an elliptic curve and its twisted one in order to obviate partition attacks.

3.1 Model and Security Notion

Here we introduce the model based on [8] and security notion.

The Model. We denote by $\mathcal{C}$ and $\mathcal{S}$ two parties that participate in the key exchange protocol P. Each of them may have several instances called oracles involved in distinct, possibly concurrent, executions of P where we denote $\mathcal{C}$ (resp., $\mathcal{S}$) instances by $\mathcal{C}^i$ (resp., $\mathcal{S}^j$), or by $\mathcal{U}$ in case of any instance. During the execution of P, an adversary has the entire control of the network and additionally has access to the parties' stored secrets where the latter simulates *insecure* devices and databases. Let us show the capability of adversary $\mathcal{A}$ each query captures:

- $\mathsf{Execute}(\mathcal{C}^i, \mathcal{S}^j)$: This query models passive attacks, where the adversary gets access to honest executions of P between $\mathcal{C}^i$ and $\mathcal{S}^j$ by eavesdropping.
- $\mathsf{Send}(\mathcal{U}, m)$: This query models active attacks by having $\mathcal{A}$ send a message to instance $\mathcal{U}$. The adversary $\mathcal{A}$ gets back the response $\mathcal{U}$ generates in processing the message m according to the protocol P. A query $\mathsf{Send}(\mathcal{C}^i, \mathtt{Start})$ initializes the key exchange protocol.
- $\mathsf{Reveal}(\mathcal{U})$: This query handles the misuse of the session key by any instance $\mathcal{U}$. The query is only available to $\mathcal{A}$ if the instance actually holds a session key and the latter is released to $\mathcal{A}$.
- $\mathsf{Leak}(\mathcal{U})$: This query handles the leakage of the "stored" secrets by any instance $\mathcal{U}$. The adversary $\mathcal{A}$ gets back (s_j, param) and (v_j, param) where the former (resp., the latter) is released if the instance corresponds to $\mathcal{C}^i$ (resp., $\mathcal{S}^j$).
- $\mathsf{Test}(\mathcal{U})$: The Test-query can be asked at most once by the adversary $\mathcal{A}$ and is only available to $\mathcal{A}$ if the instance $\mathcal{U}$ is "fresh" in that the session key is not obviously known to the adversary. This query is answered as follows: one flips a (private) coin $b \in \{0, 1\}$ and forwards the corresponding session key SK ($\mathsf{Reveal}(\mathcal{U})$ would output) if $b = 1$, or a random value except the session key if $b = 0$.

Security Notion. The adversary $\mathcal{A}$ is provided with random coin tosses, some oracles and then is allowed to invoke any number of queries as described above, in any order. The aim of the adversary is to break the privacy of the session key in the context of executing P. The AKE security is defined by the game $\mathbf{Game}^{\mathsf{ake}}(\mathcal{A}, P)$, in which the ultimate goal of the adversary is to guess the bit b involved in the Test-query by outputting this guess b'. We denote the AKE advantage, by $\mathsf{Adv}_P^{\mathsf{ake}}(\mathcal{A}) = 2\Pr[b = b'] - 1$, as the probability that $\mathcal{A}$ can correctly guess the value of b. The protocol P is said to be (t, ε)-AKE-secure if $\mathcal{A}$'s advantage is smaller than ε for any adversary $\mathcal{A}$ running time t.

3.2 Elliptic Curve Diffie-Hellman Assumption

A (t, ε)-$\mathsf{ECDH}_{G,\mathbb{G}}$ attacker, in a finite cyclic group $\mathbb{G}$ of prime order q with G as a generator, is a probabilistic machine $\mathcal{B}$ running in time t such that its success probability $\mathsf{Succ}_{G,\mathbb{G}}^{\mathsf{ecdh}}(\mathcal{B})$, given random elements aG and bG to output abG, is greater than ε. We denote by $\mathsf{Succ}_{G,\mathbb{G}}^{\mathsf{ecdh}}(t)$ the maximal success probability over every adversaries running within time t. The ECDH-Assumption states that $\mathsf{Succ}_{G,\mathbb{G}}^{\mathsf{ecdh}}(t) \leq \varepsilon$ for any t/ε not too large.

3.3 Security Proofs

Suppose an active attacker $\mathcal{A}$, who gets the client's stored secret, is willing to break the semantic security of the EC-AKA protocol. The protocol is said to be secure if, when passwords are chosen from a dictionary of size N, $\mathsf{Adv}_P^{\mathsf{ake}}(\mathcal{A}) \leq O(q_s/N) + \varepsilon(\cdot)$ for some negligible function $\varepsilon(\cdot)$ in the security parameter. The first term represents the fact that the attacker can do no better than guess a password during each interaction to the parties where q_s is the number of queries to the Send-oracle.

Theorem 1. *The* EC-AKA *protocol is* provably secure *against an attacker, who asks the* $\mathsf{Leak}(\mathcal{C}^i)$*-query, in the random oracle model [9] if the elliptic curve Diffie-Hellman (*ECDH*) problem is hard.*

Proof. We prove this theorem by contradiction. Here we assume that the EC-AKA protocol is insecure in the sense that the attacker $\mathcal{A}$ can distinguish the key given by the Test-oracle. With the elliptic curve Diffie-Hellman instance as input, we show that an algorithm $\mathcal{B}$ can compute the Diffie-Hellman key by using the attacker $\mathcal{A}$ as a subroutine.

The algorithm $\mathcal{B}$ is given the ECDH instance $(G, P = aG, Q = bG)$ and should simulate all of the queries from attacker $\mathcal{A}$. When $\mathcal{A}$ asks a hash-query $\mathcal{H}_j(q)$, such that a record (j, q, r) appeared in the $\mathcal{H}_j$-oracle, the answer is r. Otherwise, answer r is chosen randomly from $\{0,1\}^k$ and the record (j, q, r) is added to the $\mathcal{H}_j$. Now, algorithm $\mathcal{B}$ sets $(G_1 = G, G_2 = Q)$, feeds it to attacker $\mathcal{A}$, and then simulates the protocol as usual. When $\mathcal{A}$ asks a $\mathsf{Send}(\mathcal{S}^j, *)$-query, $\mathcal{B}$ computes Y as follows: $Y \equiv yP$. We can easily see that the simulation is perfectly indistinguishable in the view of $\mathcal{A}$ since there exists a unique discrete logarithm for Y. After seeing a hash-query $\mathcal{H}_j(q)$ asked by $\mathcal{A}$, $\mathcal{B}$ can solve the ECDH problem with non-negligible probability. Let $W_i = \mathcal{H}(q_i)$ and $K_i = \mathsf{ECDH}_{G,\mathbb{G}}((Z - W_i G_2), Y) = \mathsf{ECDH}_{G,\mathbb{G}}(Z, Y) + \mathsf{ECDH}_{G,\mathbb{G}}(-W_i G_2, Y)$ such that the tuple (Z, Y, K_i) is in $\mathcal{H}_j$. With probability $1/q_h^2$, $\mathcal{B}$ can compute the Diffie-Hellman key $\mathsf{ECDH}_{G,\mathbb{G}}((W_0 - W_1)Q, yP) = K_1 - K_0$ since B already knows y, W_0 and W_1. The running time of $\mathcal{B}$ is the running time of $\mathcal{A}$ plus some constant time for modular multiplication. This concludes the proof.

Of course, the attacker can do on-line dictionary attacks with the success probability $O(q_s/N)$. But, notice that the EC-AKA protocol doesn't allow even on-line attacks without any leakage of stored secrets since the authentication depends on the strong secret v_j like [10,11].

Suppose an active attacker, who gets the server's stored secret, is willing to break security of the EC-AKA protocol by impersonating the compromised server. In that case, we cannot avoid this impersonation attack as all of the authentication protocols cannot. However, we can say the following theorem.

Theorem 2. *The* EC-AKA *protocol is* secure *against an attacker, who asks the* $\mathsf{Leak}(S^j)$*-query, unless the attacker do the server impersonation attack within a limited time period.*

Table 1. Classification and comparison of AKA protocols

Protocols	Client's possessions			Extension*1
	Password	Stored Secret	Public Info.	
EAP-MD5, LEAP	√			impossible
PAKE [3,5]	√			impossible
MA-DHKE*2 [13]	√		√	impossible
EAP-SIM [14]		√		possible*3
EAP-FAST	√	√		impossible
EC-AKA	√	√		possible*3
EAP-TLS		√	√	possible
EAP-TTLS, PEAP	√	(√)*4	√	impossible

*1: Whether or not each protocol can be extended to the multiple server scenario (with only one password).
*2: Mutual Authentication and Diffie-Hellman Key Exchange of Section 3.4.
*3: The number of stored secrets grows linearly to the number of servers.
*4: Optional.

We assume that the EC-AKA protocol runs at a fixed time period (e.g., a day) and an attacker obtains the secret (i.e., v_j) at that time. In this case, if the update of v_j between the client and the server is completed before the attacker does, the latter cannot do the impersonation attack any more because v_j is no longer valid.

4 Comparison

In this section, we compare the EC-AKA protocol with the existing AKA protocols (including EAP methods [12]). In Table 1, we classify each protocol in the viewpoint of which kind of information is needed for client authentication. Table 2 shows the comparative result of security properties when the leakage of stored secrets happen in each protocol. Though both the EAP-FAST and EC-AKA protocols are based on the password and additional secret stored on client's devices, the former is not adequate for multiple sever scenario and insecure against the leakage of stored secrets.

Efficiency, as well, is very important when considering practical applications for mobile devices with restricted computing power and wireless networks having limited bandwidth. As for communication overhead, we represent the points on the elliptic curve in a compressed form: given an affine point $(\mathcal{X}, \mathcal{Y})$ the $\mathcal{X}$-coordinate requires n bits where n is the bit length of the underlying field and the $\mathcal{Y}$-coordinate is represented by one bit in order to distinguish two solutions of a quadratic equation. In addition, the length of identities and counter is excluded. Table 3 indicates that the EC-AKA protocol is more efficient mainly in terms of computation costs of client and communication overheads compared to [3,5].

Table 2. Comparison of AKA protocols in a situation where no perfect TRM is available

Protocols	Security[*1] (of password)		
	on communications	against the leakage of stored secrets from client $\mathcal{C}$	against the leakage of stored secrets from server $\mathcal{S}$
EAP-MD5, LEAP	△[*2]	◯	X (△[*2])
PAKE [3,5]	◯	◯	X (△[*2])
MA-DHKE [13]	◯	◯	△[*2]
EAP-SIM [14]	◯	X	X
EAP-FAST	◯	X	X
EC-AKA	◯	◯[*3]	◯[*3]
EAP-TLS	◯	X	X
EAP-TTLS, PEAP	◯	◯	X (△[*2])

*1: ◯ guarantees the security of password against both on-line and off-line dictionary attacks. △ guarantees the security of password against on-line, but not off-line attacks. X guarantees the security of password against neither on-line nor off-line attacks.

*2: A client registers password verification data computed with a particular one-way function of the password, $f(pw)$, at the server instead of pw.

*3: Information-theoretically secure.

Table 3. Comparison of elliptic curve based AKA protocols, which do not rely on PKI, in terms of efficiency

Protocols	Computation costs of client[*1]	Communication overhead[*2]	The number of flows
EC-EKE [3]	$4Mul$, [$3Mul$] [*3]	100 Bytes	3
PAKE-EC [5]	$5Mul$, [$3Mul$] [*3]	160 Bytes	4
EC-AKA	$4Mul$, [$3Mul$]	60 Bytes	3

*1: Mul denotes the number of modular multiplications and the figures in the brackets are the remaining costs after pre-computation.

*2: For the minimum security parameters recommended for use in current practice: $|q| = |\mathcal{H}| = 160$ (for the elliptic curve Diffie-Hellman protocol and hash functions).

*3: $2Mul$ are needed for checking the group order.

5 Conclusions

When we consider wireless security, a combination of password-based authentication and the elliptic curve Diffie-Hellman protocol is strongly preferable mainly because it not only does not require any security infrastructure, but also provide computation and communication efficiency. However, such combination does not necessarily guarantee security against a special kind of off-line dictionary attacks, known as "partition attacks".

As one of the solutions for wireless security, we have proposed an elliptic curve based AKA (EC-AKA) protocol secure against partition attacks as well as

suitable for the following situation: (1) a client, who communicates with many different servers, remembers only one password and has insecure devices (e.g., mobile phones or PDAs); (2) the counterpart servers are not perfectly secure against several attacks (e.g., virus or hacker); (3) neither PKI nor TRM is available. The authenticity of the EC-AKA protocol is based on the client's relatively short password and an additional secret stored on insecure devices. We proved that the EC-AKA protocol is provably secure in the random oracle model with the reduction to the elliptic curve Diffie-Hellman problem. In addition, we analyzed its several security properties and efficiency while comparing with the existing AKA protocols (employed in the IEEE 802.1x [15]).

Acknowledgments. The authors appreciate anonymous reviewers for their helpful comments.

References

1. Koblitz, N.: Elliptic Curve Cryptosystems. Mathematics of Computation 48, 203–209 (1987)
2. Miller, V.: Use of Elliptic Curves in Cryptography. In: Williams, H.C. (ed.) Advances in Cryptology. LNCS, vol. 218, pp. 417–426. Springer, Heidelberg (1986)
3. Boyd, C., Montague, P., Nguyen, K.: Elliptic Curve based Password Authenticated Key Exchange Protocols. In: Varadharajan, V., Mu, Y. (eds.) Information Security and Privacy. LNCS, vol. 2119, pp. 487–501. Springer, Heidelberg (2001)
4. Diffie, W., Hellman, M.: New Directions in Cryptography. IEEE Transactions on Information Theory IT-22(6), 644–654 (1976)
5. Wong, D.S., Chan, A.H., Zhu, F.: Password Authenticated Key Exchange for Resource-Contrained Wireless Communications. In: Lorenz, P., Dini, P. (eds.) Networking - ICN 2005. LNCS, vol. 3421, pp. 827–834. Springer, Heidelberg (2005)
6. Antipa, A., Brown, D., Menezes, A., Struik, R., Vanstone, S.: Validation of Elliptic Curve Public Keys. PKC 2003. In: Desmedt, Y.G. (ed.) Public Key Cryptography - PKC 2003. LNCS, vol. 2567, pp. 211–223. Springer, Heidelberg (2002)
7. Blake, I.F., Seroussi, G., Smart, N.P.: Elliptic Curves in Cryptography. In: Jantke, K.P. (ed.) Analogical and Inductive Inference. LNCS, vol. 265, Springer, Heidelberg (1987)
8. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated Key Exchange Secure against Dictionary Attacks. In: Preneel, B. (ed.) Advances in Cryptology - EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000)
9. Bellare, M., Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In: ACM CCS'93, pp. 62–73. ACM Press, New York (1993)
10. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Stinson, D.R. (ed.) Advances in Cryptology - CRYPTO '93. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
11. Shoup, V.: On Formal Models for Secure Key Exchange. IBM Research Report RZ 3121 (1999), `http://eprint.iacr.org/1999/012`
12. IETF (Internet Engineering Task Force).: PPP Extensible Authentication Protocol (EAP). RFC 2284 (1998)

13. Halevi, S., Krawczyk, H.: Public-Key Cryptography and Password Protocols. In: ACM Transactions on Information and System Security, vol. 2(3), pp. 230–268. ACM Press, New York (1999)
14. Haverinen, H., Salowey, J.: Extensible Authentication Protocol Method for GSM Subscriber Identity Modules (EAP-SIM) (2004) draft-haverinen-pppext-eap-sim-16.txt
15. IEEE 802.1x.: Port Based Network Access Control. IEEE, http://www.ieee802/org/1/pages/802.1x.html