Available online at www.sciencedirect.com
SCIENCE DIRECT
Theoretical Computer Science 352 (2006) 280–287
Theoretical Computer Science
www.elsevier.com/locate/tcs

Security analysis of a password-based authentication protocol proposed to IEEE 1363

Zhu Zhao[a], Zhongqi Dong[b], Yongge Wang[c,*]

[a] *Hexi University, ZhangYe City, GanSu Province, PR China*
[b] *Lanzhou University, Lanzhou City, GanSu Province, PR China*
[c] *SIS Department, UNC Charlotte, Charlotte, NC, USA*

Received 25 June 2004; received in revised form 23 September 2005; accepted 18 November 2005

Communicated by M. Ito

Abstract

In recent years, several protocols for password-based authenticated key exchange have been proposed. These protocols aim to be secure even though the sample space of passwords may be small enough to be enumerated by an off-line adversary. In Eurocrypt 2000, Bellare, Pointcheval and Rogaway (BPR) presented a model and security definition for authenticated key exchange. They claimed that in the ideal-cipher model (random oracles), the two-flow protocol at the core of Encrypted Key Exchange (EKE) is secure. Bellare and Rogaway suggested several instantiations of the ideal cipher in their proposal to the IEEE P1363.2 working group. Since then there has been an increased interest in proving the security of password-based protocols in the ideal-cipher model. For example, Bresson, Chevassut, and Pointcheval have recently showed that the One-Encryption-Key-Exchange (OEKE) protocol is secure in the ideal cipher model. In this paper, we present examples of *real* (NOT ideal) ciphers (including naive implementations of the instantiations proposed to IEEE P1363.2) that would result in broken instantiations of the idealised AuthA protocol and OEKE protocol. Our result shows that the AuthA protocol can be instantiated in an insecure way, and that there are no well defined (let alone rigorous) ways to distinguish between secure and insecure instantiations. Thus, without a rigorous metric for ideal-ciphers, the value of provable security in ideal cipher model is limited.

Keywords: Password-based key agreement; Dictionary attacks; AuthA; EKE

1. Introduction

Numerous cryptographic protocols rely on passwords selected by users (people) for strong authentication. Since the users find it inconvenient to remember long passwords, they typically select short easily rememberable passwords. In these cases, the sample space of passwords may be small enough to be enumerated by an adversary thereby making the protocols vulnerable to a *dictionary* attack. It is desirable then to design password-based protocols that resist off-line dictionary attacks.

* Corresponding author. Tel.: +1 704 687 5491; fax: +1 704 687 4893.
E-mail addresses: zhaozhu@hxu.edu.cn (Z. Zhao), dongzha03@st.lzu.edu.cn (Z. Dong), yonwang@uncc.edu (Y. Wang).

0304-3975/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2005.11.038

The password-based protocol problem was first studied by Gong et al. [10] who used public-key encryption to guard against off-line password-guessing attacks. In another very influential work [4], Bellovin and Merritt introduced Encrypted Key Exchange (EKE), which became the basis for many of the subsequent works in this area. These protocols include SPEKE [13] and SRP [25,26]. Other papers addressing the above protocol problem can be found in [7,9,11,16]. Bellare et al. [2] defined a model for the password-based protocol problem and claimed that their model is rich enough to deal with password guessing, forward secrecy, server compromise, and loss of session keys. Then they claimed that in the ideal-cipher model (random oracles), the two-flow protocol at the core of EKE is secure. In addition, Bellare and Rogaway [3] suggested several instantiations (AuthA) of the ideal-cipher in their proposal to the IEEE P1363.2 Working Group. Recently, Bresson et al. [8] proposed a simplified version of AuthA, which is called One-Encryption-Key-Exchange (OEKE), and showed that OEKE achieves provable security against dictionary attacks in both the random oracle and ideal-cipher models under the computational Diffie–Hellman intractability assumption.

The ideal-cipher model was introduced by Bellare et al. [2] as follows. Fix finite sets of strings $\mathcal{G}$ and $\mathcal{C}$ where $|\mathcal{G}| = |\mathcal{C}|$. In the ideal-cipher model, choosing a random function h from Ω amounts to giving the protocol (and the adversary) a perfect way to encipher strings in $\mathcal{G}$: namely, for $K \in \{0, 1\}^*$, we set $\mathcal{E}_K : \mathcal{G} \to \mathcal{C}$ to be a random bijective function, and we let $\mathcal{D}_K : \{0, 1\}^* \to \mathcal{G}$ defined by $\mathcal{D}_K(y)$ be the value x such that $\mathcal{E}_K(x) = y$, if $y \in \mathcal{C}$, and undefined otherwise.

This paper studies the security issues with practical realization of the ideal cipher model by Bellare et al. [2]. We show that for several instantiations of the ideal-cipher (including naive implementations of instantiations suggested in [12]), the instantiated Bellare and Rogaway's protocol (AuthA) is not secure against off-line dictionary attacks. Our results show that realizing the ideal-cipher of Bellare, Pointcheval, and Rogaway (BPR) can be tricky. In particular, our results point out the weakness in the ideal-cipher methodology of BPR. That is, without a robust measuring method for deciding whether a given cipher is a "good realization" of the ideal-cipher, ideal-cipher model analysis [2,8] of a password-based protocol can be of limited value. Indeed, there is no well defined (let alone rigorous) way in [2] to distinguish between secure and insecure instantiations of an ideal-cipher. Note that Black and Rogaway [5] have done some initial research on the potential implementations of ideal-ciphers with arbitrary finite domains. However, it is still far from a complete solution.

One of the main applications of password-based protocols is in the environment of wireless and other more constrained devices (e.g., secure downloading of private credentials: SACRED [20]). Elliptic curve cryptography (ECC) has been extensively used in these constrained devices. However, most of the suggested password-based protocols are described in the group (or subgroup of) $\mathcal{G} = \mathbb{Z}_p^*$, and are either non-friendly or non-secure for ECC-based groups. For example, SRP [25,26] is based on a field and used both field operations of addition and multiplication, but ECC groups only have one group operation. Several ECC-based SRP protocols have been introduced in Lee and Lee [14]. We will show that one of these protocols is completely insecure. We will also discuss the security issues related ECC-based SRP protocols. As an example, we will also present a variant SRP5 of the original SRP protocol.

The organization of the paper is as follows: In Section 2, we informally address the security problems of password-based protocols. We mount attacks on several instantiations of BPR AuthA protocol and on instantiations of Bresson, Chevassut, and Pointcheval's OEKE protocol in Sections 3 and 4, respectively. In Section 5, we briefly discuss instantiations of OEKE and the SRP protocol. We draw our conclusions in Section 6.

2. Security of password authentication

Halevi and Krawczyk [11, Sections 2.2–2.3] introduced a notion of security for password authentication. They provide a list of basic attacks that a password-based protocol needs to guard against. In the following, we provide the list of attacks. An ideal password protocol should be secure against these attacks and we will follow these criteria when we discuss the security of password protocols.

- *Eavesdropping*. The attacker may observe the communications channel.
- *Replay*. The attacker records messages she has observed and re-sends them at a later time.
- *Man-in-the-middle*. The attacker intercepts the messages sent between the parties $\mathcal{C}$ and $\mathcal{S}$ and replaces these with her own messages. She plays the role of the client in the messages which it sends to the server, and at the same time plays the role of the server in the messages that she sends to the client. A special man-in-the-middle attack is the

small subgroup attack [15,17,23]. We illustrate this kind of attack by a small example. Let g be a generator of the group $\mathcal{G}$ of order $n = qt$ for some small $t > 1$. In a standard Diffie–Hellman key exchange protocol, the client $\mathcal{C}$ chooses a random x and sends g^x to the server $\mathcal{S}$, then $\mathcal{S}$ chooses a random y and sends g^y to $\mathcal{C}$. The shared key between $\mathcal{C}$ and $\mathcal{S}$ is g^{xy}. Now assume that the attacker $\mathcal{A}$ intercepts $\mathcal{C}$'s message g^x, replaces it with g^{xq}, and sends it to $\mathcal{S}$. $\mathcal{A}$ also intercepts $\mathcal{S}$'s message g^y, replaces it with g^{yq}, and sends it to $\mathcal{C}$. In the end, both $\mathcal{C}$ and $\mathcal{S}$ compute the shared key g^{qxy}. Since g^{qxy} lies in the subgroup of order t of the group generated by g^q, it takes on one of only t possible values. $\mathcal{A}$ can easily recover this g^{qxy} by an exhaustive search.

- *Impersonation*. The attacker impersonates the client or the server to get some useful information.
- *Password-guessing*. The attacker is assumed to have access to a relatively small dictionary of words that likely includes the secret password α. In an *off-line attack*, the attacker records past communications and searches for a word in the dictionary that is consistent with the recorded communications. In an *on-line attack*, the attacker repeatedly picks a password from the dictionary and attempts to impersonate $\mathcal{C}$ or $\mathcal{S}$. If the impersonation fails, the attacker removes this password from the dictionary and tries again, using a different password.
- *Partition attack*. The attacker records past communications, then goes over the dictionary and deletes those words that are not consistent with the recorded communications from the dictionary. After several tries, the attacker's dictionary could become very small.

We now informally sketch the definition of security in [11] for a password-based protocol. The attacker $\mathcal{A}$ is allowed to watch regular runs of the protocol between the client $\mathcal{C}$ and the server $\mathcal{S}$, and can also actively communicate with $\mathcal{C}$ and $\mathcal{S}$ in replay, impersonation, and man-in-the-middle attacks. A protocol is said to be *secure* in the presence of such an attacker if (i) whenever the server $\mathcal{S}$ accepts an authentication session with $\mathcal{C}$, it is the case that $\mathcal{C}$ did indeed participate in the authentication session; and (ii) $\mathcal{C}$ accepts an authentication session with $\mathcal{S}$, it is the case that $\mathcal{S}$ did indeed participate in the authentication session.

3. Security issues with practical realizations of the ideal cipher model: on Bellare and Rogaway's AuthA

In the remainder of this paper, we will use the following notations: by $\mathcal{G} = \langle g \rangle$, we denote a cyclic group generated by g, and by $ord(g)$, we denote the order of g. For a symmetric encryption scheme $\mathcal{E}$ and a key π, $\mathcal{E}_\pi(x)$ denotes the ciphertext of x. We also assume that the client $\mathcal{C}$ holds a password α and the server $\mathcal{S}$ holds a key β which is a known function of α. In a protocol for a symmetric model, the client and the server share the same password, that is, $\beta = \alpha$. In this paper, we will abuse our notation by letting $\mathcal{C}$ and $\mathcal{S}$ also denote corresponding parties' identification strings. In a protocol for an asymmetric model, β will typically be chosen so that it is hard to compute α from $\mathcal{C}$, $\mathcal{S}$, and β. The password α might be a poor one. Probably the user selects some short easily-rememberable α and then installed β at the server. In the protocols, $\mathcal{H}$ is used to denote a secure hash function. We will also abuse our notation by using $\mathcal{C}$ (respectively, $\mathcal{S}$) to denote the identity number of the client (respectively, the server).

3.1. The AuthA protocol

Bellare et al. [2] defined a model for the password-based protocol problem and showed that their model is rich enough to deal with password guessing, forward secrecy, server compromise, and loss of session keys. Then they proved that in the ideal-cipher model (random oracles), the two-flow protocol at the core of EKE is secure. In addition, Bellare and Rogaway [3] suggested several instantiations of the ideal-cipher in their proposal to IEEE P1363.2 working group. In the protocol, the server $\mathcal{S}$ stores the value $\langle \mathcal{C}, \beta \rangle$ for each client $\mathcal{C}$ where $\beta = g^\alpha$. The protocol proceeds as follows:

(1) $\mathcal{C}$ chooses a random $x \in [1, ord(g) - 1]$, computes g^x, encrypts it with β, and sends the ciphertext $\mathcal{E}_\beta(g^x)$ to the Server $\mathcal{S}$.
(2) $\mathcal{S}$ chooses a random $y \in [1, ord(g) - 1]$, computes g^y, encrypts it with β, and sends the ciphertext $\mathcal{E}_\beta(g^y)$ to $\mathcal{C}$.
(3) AuthA authentication steps. Let $K = \mathcal{H}(\mathcal{C}||\mathcal{S}||g^x||g^y||g^{xy})$. Then there are three authentication methods for AuthA:
   (a) The server authenticates himself by sending $\mathcal{H}(K||2)$ to $\mathcal{C}$.
   (b) The client authenticates himself by sending $\mathcal{H}(K||g^{\alpha y})$ to $\mathcal{S}$.
   (c) Both server and client achieve mutual authentication by sending both of the messages in the above two steps.

The authors of [2] claimed that if the encryption function $\mathcal{E}$ is given by an ideal-cipher (random oracle), then the first-two-step sub-protocol (of AuthA) at the core of EKE is provably secure in their model. In the following sections, we present examples of *real* (NOT ideal) ciphers (including two naive implementations of the three instantiations proposed to IEEE P1363.2) that would result in broken instantiations of the idealised AuthA protocol. Indeed, in [2], the authors warn that "incorrect instantiation of the encryption primitive, including instances which are quite acceptable in other contexts, can easily destroy the protocol's security". Our examples confirm this argument.

3.2. Instantiation $\mathcal{E}_\beta(X) = X \cdot g^{\mathcal{H}(\beta)}$

Assume that $\mathcal{H}$ is a random oracle. Bellare and Rogaway [3] suggested the instantiation $\mathcal{E}_\beta(X) = X \cdot \mathcal{H}(\beta)$ of the ideal-cipher. Obviously, this is far from an ideal cipher. However, this misleading instantiation will give one impression that $\mathcal{E}_\beta(X) = X \cdot g^{\mathcal{H}(\beta)}$ could also be a "reasonable" instantiation of the ideal-cipher. Indeed, one may wonder, if $\mathcal{E}_\beta(X) = X \cdot \mathcal{H}(\beta)$ is an ideal cipher, why $\mathcal{E}_\beta(X) = X \cdot g^{\mathcal{H}(\beta)}$ is not? In the following, we will describe our attack on the two-step protocol with this instantiation $\mathcal{E}_\beta(X) = X \cdot g^{\mathcal{H}(\beta)}$.

No matter whether there is an authentication step (as in AuthA) or not, our attack works for the two-step protocol. If there is an authentication step, then the adversary $\mathcal{A}$ will launch impersonation attacks and use the authentication messages to verify whether the guessed password is a correct one. Without loss of generality, we assume that the server sends the first authentication message if any authentication message is ever sent between $\mathcal{C}$ and $\mathcal{S}$ (if the first authentication message is sent from client to server, then the following attack works when the adversary impersonate the server). If there is no authentication step, then the adversary could not check whether a guessed password is a correct one. However, in practice, the established session key will be used either to encrypt the actual data for the application protocol or to encrypt client's private credential (e.g., client's private key). In either case, the adversary $\mathcal{A}$ can verify whether the guessed password is a correct one by checking the redundancy in these encrypted data. Specifically, consider the following scenario. $\mathcal{A}$ impersonates the client, chooses a random z, and sends g^z to the server. The server $\mathcal{S}$ chooses a random y, sends $g^{y+\mathcal{H}(\beta)}$ to $\mathcal{A}$, and computes the shared key $K = \mathcal{H}(\mathcal{C}||\mathcal{S}||g^{z-\mathcal{H}(\beta)}||g^y||g^{(z-\mathcal{H}(\beta))y})$. $\mathcal{A}$ distinguishes the following three cases:

(1) This is an AuthA protocol and $\mathcal{S}$ sends $\mathcal{H}(K||2)$ to $\mathcal{A}$ for authentication. For each guessed β', $\mathcal{A}$ computes

$$K' = \mathcal{H}(\mathcal{C}||\mathcal{S}||g^{z-\mathcal{H}(\beta')}||g^{y+\mathcal{H}(\beta)-\mathcal{H}(\beta')}||g^{(y+\mathcal{H}(\beta)-\mathcal{H}(\beta'))(z-\mathcal{H}(\beta'))}).$$

Note that if $\beta' = \beta$, then $K' = K$ and $\mathcal{H}(K||2) = \mathcal{H}(K'||2)$. Thus $\mathcal{A}$ can decide whether β' is the correct password.

(2) $\mathcal{S}$ sends $\mathcal{E}_K(m)$ to $\mathcal{A}$, where m is some application data and has sufficient redundancy. For each guessed β', $\mathcal{A}$ computes K' as in the above item 1 and decrypts $\mathcal{E}_K(m)$ as $m' = \mathcal{E}_{K'}^{-1}(\mathcal{E}_K(m))$. If $\beta' = \beta$, then $K' = K$ and $m' = m$. Thus by checking the redundancy in m', $\mathcal{A}$ can decide whether she has guessed the password correctly.

(3) $\mathcal{S}$ sends $\mathcal{E}_K(\pi)$ to $\mathcal{A}$, where π is $\mathcal{C}$'s private key encrypted with $\mathcal{C}$'s password α. Similarly, for each guessed α', $\mathcal{A}$ first computes β', then computes K' as in the above item 1 and decrypts $\mathcal{E}_K(\pi)$ as $\pi' = \mathcal{E}_{K'}^{-1}(\mathcal{E}_K(\pi))$. If $\beta' = \beta$, then $K' = K$ and $\pi' = \pi$. $\mathcal{A}$ further decrypts π' with α' to see whether the decrypted value is the private key of $\mathcal{C}$. Since $\mathcal{A}$ knows $\mathcal{C}$'s public key, she can easily verify this fact. Thus, $\mathcal{A}$ can decide whether she has guessed the password correctly.

The above attack demonstrates the inherent weakness in the "ideal-cipher model methodology" by Bellare et al. [2]. That is, without a robust measuring method for deciding whether a given cipher is a "good realization", ideal-cipher model analysis of a password-based protocol can be of limited value. Indeed, there is no well defined (let alone rigorous) way in [2] to distinguish between secure and insecure instantiations of an ideal-cipher.

3.3. Instantiation $\mathcal{E}_\beta(X) = X \cdot \mathcal{H}(\beta)$

The first ideal-cipher instantiation for AuthA in [3] is: $\mathcal{E}_\beta(X) = X \cdot \mathcal{H}(\beta)$. The authors suggested that the group $\mathcal{G} = \langle g \rangle$ could be a group on which the Diffie–Hellman problem is hard:

> ...This group could be $\mathcal{G} = \mathbb{Z}_p^*$, or it could be a prime-order subgroup of this group, or it could be an elliptic curve group...(from [2])

After the introduction of the instantiation function, the authors [3] commented that "you apply the mask generation function $\mathcal{H}$ to β, interpret the result as a group element, and multiply by the plaintext". However, for most implementations, one may ignore this comment and just multiply the hash result with the plaintext. Naively, one can also interpret the hash result $\mathcal{H}(\beta)$ as a group element $g^{\mathcal{H}(\beta)}$. Then our attacks in Section 3.2 show that this instantiation is not secure. Indeed, from the ideal-cipher assumption, it is not clear that one needs to interpret the hash result as a group element other than $g^{\mathcal{H}(\beta)}$. One may feel that both $X \cdot \mathcal{H}(\beta)$ and $X \cdot g^{\mathcal{H}(\beta)}$ can be regarded as acceptable instantiations of the ideal cipher over $\mathbb{Z}_p^*$ (why not?). In the following, we mount an off-line dictionary attack on this instantiation without interpreting the result as a group element.

Our attack in Section 3.2 does not work for AuthA with this instantiation. However, we can show that this instantiation will leak some information of the password α if the group is a subgroup of $\mathbb{Z}_p^*$ or an elliptic curve group. As an example, we illustrate the information leakage of AuthA with a subgroup of $\mathbb{Z}_p^*$. Assume that $p = tq + 1$ with $\gcd(t, q) = 1$. In practice, generally one chooses $p = 2q + 1$ for some large prime q (see, e.g., [18]), and $ord(g) = q$.

In the attack, the eavesdropper $\mathcal{A}$ intercepts the message $g^x \cdot \mathcal{H}(\beta)$, computes $(\mathcal{H}(\beta))^q = (g^x \cdot \mathcal{H}(\beta))^q$. For each guessed β', $\mathcal{A}$ checks whether $(\mathcal{H}(\beta'))^q = (\mathcal{H}(\beta))^q$. If the equation does not hold, then $\mathcal{A}$ deletes β' from her dictionary. Since $\mathcal{H}$ is a random oracle, the value $(\mathcal{H}(x))^q$ is uniformly distributed over the set $\{g_1^q, g_1^{2q}, \ldots, g_1^{tq}\}$ when x is chosen at random, where g_1 is a generator of $\mathbb{Z}_p^*$. That is, $\mathbb{Z}_p^* = \langle g_1 \rangle$. Thus, $\log t$ bits information of the password is leaked for each communication between the client and the server with different Diffie–Hellman parameters. Thus, after $\lceil |\alpha| / \log t \rceil$ observations of communications between the client and the server with different Diffie–Hellman parameters, the adversary will recover the password with high probability.

Despite the above attacks, we feel that AuthA could be securely instantiated by the cipher: $\mathcal{E}_\beta(X) = X \cdot \imath(\mathcal{H}(\beta))$, where $\mathcal{H}$ is a secure hash function and where $\imath$ maps a random string to a group element of order $ord(g)$ by "increasing" the random string one by one until reaching a group element with the above given property. This instantiation should work both for ECC based groups and for subgroups of $\mathbb{Z}_p^*$. But we would like to warn that we have not proved with reasonable assumptions that this is a secure instantiation of the ideal cipher. Of course, it has been proven [2] that if $\mathcal{E}_\beta(X) = X \cdot \imath(\mathcal{H}(\beta))$ is an ideal cipher then the above instantiation is provable secure against off-line dictionary attacks. But we have no mechanisms to measure whether the above cipher is an ideal cipher.

3.4. Instantiation $\mathcal{E}_\beta(X) = (r, X \cdot \mathcal{H}(r||\beta))$

The second ideal-cipher instantiation for AuthA in [3] is: $\mathcal{E}_\beta(X) = (r, X \cdot \mathcal{H}(r||\beta))$, where r is independently chosen at random for each session. After the introduction of this instantiation, the authors [3] did not mention that the hash result $\mathcal{H}(r||\beta)$ should be interpreted as a group element before applying the multiplication. However, we assume that the authors have this in mind when they introduce this instantiation. But this again shows that a naive implementation may multiply the hashing result with X directly without interpreting it as a group element since $\mathcal{E}_\beta(X) = (r, X \cdot \mathcal{H}(r||\beta))$ could be regarded as an acceptable ideal cipher. Indeed, the ideal cipher model does not address this tiny difference between the two implementations: interpreting the hashing result as a group element and not interpreting the hashing result as a group element.

Indeed this instantiation without interpreting the hashing result as a group element is completely insecure against partition attacks if the underlying group is a subgroup of $\mathbb{Z}_p^*$ or an elliptic curve group. The attack in Section 3.3 can be used to show that for each randomly chosen r, $\log t$ bits information of the password α is leaked. Thus after recording several communications with different r, the adversary can recover α.

3.5. Instantiation $\mathcal{E}_\beta(X)$ by a cipher

The third ideal-cipher instantiation for AuthA in [3] is simply a cipher, e.g., $\mathcal{E}_\beta(X) = \mathrm{AES}_\beta(X)$. AuthA with this instantiation is not secure against partition attacks if the underlying group is a subgroup of $\mathbb{Z}_p^*$ or an elliptic curve group. The insecurity of this instantiation has been observed by several authors, see, e.g., [19,6].

Firstly, we assume that the underlying group $\mathcal{G}$ is a subgroup of $\mathbb{Z}_p^*$. The eavesdropper $\mathcal{A}$ tries to decrypt $\mathcal{E}_\beta(g^x)$ and $\mathcal{E}_\beta(g^y)$ with different guessed β' $(= g^{\alpha'})$. If either of the decrypted value $\mathcal{E}_{\beta'}^{-1}(\mathcal{E}_\beta(g^x))$ or $\mathcal{E}_{\beta'}^{-1}(\mathcal{E}_\beta(g^y))$ is not an element of $\mathcal{G}$, then $\mathcal{A}$ knows that α' is not the correct password. Since $\mathcal{E}_\beta(X)$ is an ideal cipher, only with probability $\left(||\mathcal{G}||/2^{|p|}\right)^2$ both $\mathcal{E}_{\beta'}^{-1}(\mathcal{E}_\beta(g^x))$ and $\mathcal{E}_{\beta'}^{-1}(\mathcal{E}_\beta(g^y))$ are elements of $\mathcal{G}$, where $|p|$ and $||\mathcal{G}||$ denote the length of p in

binary representation and the cardinality of $\mathcal{G}$, respectively. Thus for each execution of the protocol, $2\log(2^{|p|}/||\mathcal{G}||)$ bits information of the password α is leaked. After recording several executions of the protocol, $\mathcal{A}$ recovers the password.

Secondly, assume that the underlying group $\mathcal{G}$ is an elliptic curve group. For an elliptic curve group $E_{a,b}(F_p^*) = \langle g\rangle$ over the field F_p^*, the element $(x, y) \in \langle g\rangle$ is denoted by its x and y coordinates. For a random chosen $x \in F_p^*$, the probability that there exists a $y \in F_p^*$ such that (x, y) is a point on the curve is $1/2$. Thus, AuthA over elliptic curve groups with this instantiation is not secure against partition attacks.

4. Security issues with ideal ciphers in one encryption key exchange OEKE

Recently, Bresson et al. [8] formally modeled the AuthA protocol by the OEKE: only one flow is encrypted (using either a symmetric-encryption primitive or a multiplicative function as the product of a Diffie–Hellman value with a hash of the password). The authors pointed out that the advantage of OEKE over the classical EKE, wherein the two Diffie–Hellman values are encrypted, is its easiness of integration. For example, in Transport Layer Security (TLS) protocol with password-based key-exchange cipher suits [21,22].

OEKE is similar to AuthA except that the first message is not encrypted. In particular, the protocol proceeds as follows:

(1) $\mathcal{C}$ chooses a random $x \in [1, ord(g) - 1]$, computes g^x and sends g^x to the Server $\mathcal{S}$,
(2) $\mathcal{S}$ chooses a random $y \in [1, ord(g) - 1]$, computes g^y, encrypts it with β, and sends the ciphertext $\mathcal{E}_\beta(g^y)$ to $\mathcal{C}$,
(3) $\mathcal{C}$ computes Auth $= \mathcal{H}_1(\mathcal{C}||\mathcal{S}||g^x||g^y||g^{xy})$, and sends Auth to $\mathcal{S}$. $\mathcal{C}$ also computes session key $K = \mathcal{H}_0(\mathcal{C}||\mathcal{S}||g^x||g^y||g^{xy})$,
(4) $\mathcal{S}$ verifies that the value Auth is correct and computes the session key similarly,

where $\mathcal{H}_0$ and $\mathcal{H}_1$ are two independent random oracles. The authors [8] show that the protocol OEKE achieves provable security against dictionary attacks in both the random oracle and ideal-cipher models under the computational Diffie–Hellman intractability assumption. The authors [8] also observed that a simple block-cipher could not be used for the instantiation of the ideal-cipher due to the partition attacks.

The authors recommended two instantiations of the ideal cipher. In the first method which is essentially from [1], one encrypts the element, and re-encrypts the result, until one finally falls in the group $\mathcal{G}$. The second instantiation is the cipher $\mathcal{E}_\beta(X) = X \cdot \mathcal{H}(\beta)$ that we have discussed in Section 3.3. That is (see [8]), "to instantiate the encryption primitive as the product of a Diffie–Hellman value with a hash of the password, as suggested in [3]". Obviously, if one does not interpret the hashing output of password as a group element before applying the multiplication, then our attacks in Section 3.3 work for OEKE also. Thus, we have the same concern for OEKE: the ideal cipher model does not directly address the issues of interpreting the hashing output as group elements. From the ideal cipher model viewpoints, the two instantiations (one with interpretation of group elements and one without interpretation of group elements) have no essential difference. However, one instantiation results in broken protocol. This observation strengthens our viewpoint: without a rigorous way to distinguish between secure and insecure instantiations of an ideal-cipher, the value of the provable security in ideal-cipher model is limited.

5. Secure remote password protocol (SRP)

If the underlying group $\mathcal{G}$ in OEKE is indeed a finite field, then one can instantiate the ideal-cipher with $\mathcal{E}_\beta(X) = X+\beta$ and obtain the secure remote password protocol (SRP6) [25,26]. But one needs to be careful that SRP protocol uses different values for the keying material computation which achieves stronger security. In the SRP6 protocol, the server $\mathcal{S}$ stores the value $\langle \mathcal{C}, \beta, s\rangle$ for each client $\mathcal{C}$, where $\beta = g^v$, $v = \mathcal{H}(s||\mathcal{C}||\alpha)$, s is a random seed for $\mathcal{C}$, and $\mathcal{H}$ is a predetermined hash function. Assume that the underlying group for the protocol is $\mathcal{G} = \mathbb{Z}_p^* = \langle g\rangle$. Then the protocol proceeds as follows:

(1) $\mathcal{C}$ sends his name $\mathcal{C}$ to the server $\mathcal{S}$.
(2) $\mathcal{S}$ sends s to $\mathcal{C}$.
(3) $\mathcal{C}$ chooses a random $x \in [1, ord(g) - 1]$ and sends g^x to $\mathcal{S}$.

(4) $\mathcal{S}$ chooses a random $y \in [1, ord(g) - 1]$ and sends $3\beta + g^y$ to $\mathcal{C}$.
(5) Let $u = \mathcal{H}(g^x||3\beta + g^y)$. $\mathcal{C}$ sends $M = \mathcal{H}(g^x||3\beta + g^y||S)$ to $\mathcal{S}$ where $S = g^{y(x+uv)}$.
(6) $\mathcal{S}$ verifies that M is correct and sends $\mathcal{H}(g^x||M||S)$ to $\mathcal{C}$.
(7) $\mathcal{C}$ verifies that $\mathcal{H}(g^x||M||S)$ is correct.
(8) Let $K = \mathcal{H}(S)$.

The role of u in SRP6 is to defeat an adversary $\mathcal{A}$ who may know β. If $\mathcal{A}$ knows β and u is fixed, she can impersonate $\mathcal{C}$ by sending $g^x \cdot g^{-vu} = g^{x-uv}$ instead of g^x in the third step. Then $g^{y(x-uv+uv)} = g^{xy}$, and $K = \mathcal{H}(g^{xy})$. Note that this additional value u in the SRP protocol achieves stronger security against stolen β while OEKE does not have this level of security.

If we instantiate the ideal cipher in OEKE with $\mathcal{E}_\beta(X) = X \cdot \imath(\mathcal{H}(\beta))$ and use the SRP6 shared secret computation method, then we get a natural generalization of the SRP protocol, where $\imath$ "appropriately" maps a random string to a group element of order $ord(g)$. For example, if we define $\imath(\mathcal{H}(\beta))$ by the following procedure, then we get the SRP5 protocol [24] which is currently under standardization in the IEEE 1363.2 standard working group.

(1) Let $x = \mathcal{H}(\beta)$.
(2) If x is a group element of order $ord(g)$, then let $\imath(\mathcal{H}(\beta)) = x$. Otherwise, increase x by one and go to step (2). Note that the sentence "increase x by one" can be any natural interpretation of "add one to a group element" in a group.

Since the original SRP protocol is based on a field and uses both field operations of addition and multiplication, there is no direct translation of SRP from the group $\mathbb{Z}_p^*$ to ECC-based groups. The above generalization SRP5 of SRP6 can be implemented over ECC groups.

Lee and Lee [14] have tried to design ECC-based SRP protocols and introduced four ECC-based SRP protocols EC-SRP1, EC-SRP2, EC-SRP3, and EC-SRP4. They used completely different key authentication steps (that is, the steps (5) to (7) are different). The key steps in their protocols are the different instantiations of the ideal cipher. That is, they recommended replacing the message $3\beta + g^y$ in the fourth step of the SRP protocol with the following messages:

(1) g^y for EC-SRP1.
(2) $g^\alpha \cdot g^{xy}$ for EC-SRP2.
(3) $(g^{x-\alpha})^y$ for EC-SRP3.
(4) $(g^{x-\alpha+1})^y$ for EC-SRP4.

The keying material K is the same as that in the original SRP protocol, i.e., $K = \mathrm{SHA}(g^{y(x+uv)})$. It is straightforward to check that the protocol EC-SRP1 is insecure against off-line dictionary attacks.

6. Conclusions

In this paper, we presented several examples of real ciphers that would result in broken instantiations of the idealised AuthA and OEKE protocols. Our results show that one should be extremely careful when designing or implementing password-based protocols with provable security in idea-cipher models: a provable security in ideal-cipher model does not necessarily say that the instantiation of the protocol is secure.

Acknowledgements

The third author would like to thank Prof. Alfred Menezes for many helpful discussions and comments over the research related to this paper. The authors would also like to thank the anonymous referees of this journal for detailed comments on this paper.

References

[1] M. Bellare, A. Boldyreva, A. Desai, D. Pointcheval, Key-privacy in public-key encryption, in: Asiancrypt '01, Lecture Notes in Computer Science, Vol. 2248, Springer, Berlin, 2001, pp. 566–582.
[2] M. Bellare, D. Pointcheval, P. Rogaway, Authenticated key exchange secure against dictionary attacks, Advances in Cryptology—Eurocrypt' 2000, Lecture Notes in Computer Science, Vol. 1807, Springer, Berlin, 2000, pp. 139–155.
[3] M. Bellare, P. Rogaway, The AuthA protocol for password-based authenticated key exchange, Contributions to IEEE P1363.2 working group. http://grouper.ieee.org/groups/1363/passwdPK/contributions.html
[4] S. Bellovin, M. Merritt, Encrypted key exchange: password-based protocols secure against dictionary attacks, Proc. 1992 IEEE Computer Society Conf. on Research in Security and Privacy, 1992, pp. 72–84.
[5] J. Black, P. Rogaway, Ciphers with arbitrary finite domains, Proc. CT-RSA 2002, pp. 114–130.
[6] C. Boyd, P. Montague, K. Nguyen, Elliptic curve based password authenticated key exchange protocols, in: ACISP '01, Lecture Notes in Computer Science, Vol. 2119, Springer, Berlin, 2001, pp. 487–501.
[7] V. Boyko, P. MacKenzie, S. Patel, Provable secure password authenticated key exchange using Diffie–Hellman, Advances in Cryptology—Eurocrypt' 2000, Lecture Notes in Computer Science, Vol. 1807, Springer, Berlin, 2000, pp. 156–171.
[8] E. Bresson, O. Chevassut, D. Pointcheval, Security proofs for an efficient password-based key exchange, Proc. 10th ACM Conf. on Computer and Communications Security, 2003, pp. 241–250.
[9] P. Buhler, T. Eirich, M. Steiner, M. Waidner, Secure password-based cipher suite for TLS, Proc. Network and Distributed Systems Security Symposium, February, 2000.
[10] L. Gong, M. Lomas, R. Needham, J. Saltzer, Protecting poorly chosen secrets from guessing attacks, IEEE J. Selected Areas Comm. 11 (5) (1993) 648–656.
[11] S. Halevi, H. Krawczyk, Public-key cryptography and password protocols, ACM Trans. Inform. System Security 2 (3) (1999) 230–268.
[12] IEEE P1363.2 ⟨ http://grouper.ieee.org/groups/1363/passwdPK/submissions.html⟩
[13] D. Jablon, Strong password-only authentication key exchange, ACM Comput. Communications Rev. 26 (5) (1996) 5–26 This is also a submission to IEEE P1363.2. Available from IEEE P1363.2 ⟨ http://grouper.ieee.org/groups/1363/passwdPK/submissions.html⟩.
[14] Y. Lee, J.Lee. EC-SRP protocol: elliptic curve secure remote password protocol, Korea Inst. Inform. Security Cryptol. 9 (1) (1999) 85–102.
[15] C. Lim, P. Lee, A key recovery attack on discrete log-based schemes using a prime order subgroup, Advances in Cryptology—Crypto' 97, Lecture Notes in Computer Science, Vol. 1294, Springer, Berlin, 1997, pp. 249–263.
[16] S. Lucks. Open key exchange: how to defeat dictionary attacks without encrypting public keys, Proc. Security Protocols Workshop, Lecture Notes in Computer Science, Vol. 1361, Springer, Berlin, 1997.
[17] A. Menezes, M. Qu, S. Vanstone, Key agreement and the need for authentication, Workshop records of PKS'95, Toronto, Canada.
[18] A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press, Boca Raton, FL, 1996.
[19] S. Patel, Number theoretic attacks on secure password schemes, in: Proc. IEEE Symp. on Security and Privacy, IEEE Press, Silver Spring, MD, 1997, pp. 236–247.
[20] SACRED, Securely Available Credentials (sacred), IETF Working Group. More information is available from: ⟨http://www.ietf.org/html.charters/sacred-charter.html⟩
[21] M. Steiner, P. Buhler, T. Eirich, M. Waidner, Secure password-based cipher suite for TLS, ACM Trans. Inform. System Security 4 (2) (2001) 134–157.
[22] D. Taylor, Using SRP for TLS authentication, November 2002, working in progress Internet Draft.
[23] P. van Oorschot, M. Wiener, On Diffie–Hellman key agreement with short exponents, Advances in Cryptology—Eurocrypt' 96, Lecture Notes in Computer Science, Vol. 1070, Springer, Berlin, 1996, pp. 332–343.
[24] Y. Wang. Elliptic curve based SRP protocol SRP5, IEEE P1363.2 standards. http://grouper.ieee.org/groups/1363/passwdPK/contributions.html
[25] T. Wu, The secure remote password protocol, in: Proc. 1998 Internet Society Symp. on Network and Distributed Systems Security, San Diego, CA, 1998, pp. 97–111.
[26] T. Wu, SRP6: Improvements and refinements to the secure remote password protocol. ⟨http://srp.stanford.edu/⟩