Analysing Password Protocol Security Against Off-line Dictionary Attacks

Ricardo Corin, Jeroen Doumen, and Sandro Etalle

Faculty of Computer Science, University of Twente,
P.O.Box 217, 7500AE Enschede,
The Netherlands. Fax - (31 53)-489-4590
{corin,doumen,etalle}@cs.utwente.nl

Abstract. We study the security of password protocols against off-line dictionary attacks. In addition to the standard adversary abilities, we also consider further cryptographic advantages given to the adversary when considering the password protocol being instantiated with particular encryption schemes. We work with the applied pi calculus of Abadi and Fournet, in which the (new) adversary abilities are modelled as equations between terms. As case studies, we analyse the *Encrypted Password Transmission* (EPT) protocol of Halevi and Krawczyk, and the well-known *Encrypted Key Exchange* (EKE) of Bellovin and Merritt. Finally, we propose a modification to EKE that prevents a particular attack that arises when ciphertexts are distinguishable from random noise.

1 Introduction

Due to the low entropy available in user-chosen passwords, password protocols are usually subject to *off-line dictionary attacks* [13]. The attack is mounted by a passive adversary who eavesdrops protocol messages and then goes off-line to perform the password search.

Recently, there have been some attempts to deal with the formal verification of password protocols that are subject to off-line dictionary attacks [17, 9]. These approaches are based on the usual Dolev-Yao adversary [10], and assume *ideal* (also called *perfect*) encryption: encrypted messages do not leak *any* information to an adversary that does not have the correct key. This assumption has a crucial advantage: this fairly simple model can be automated.

Unfortunately, assuming ideal encryption to analyse password protocols is not realistic, since most practical encryption schemes are far from this ideal. In practice, password protocols are designed to be secure when instantiated with a particular encryption scheme, which makes the security against dictionary attacks dependant on the chosen cryptosystem. Typically, such security of an encryption scheme is characterized by

certain properties that the ciphertexts satisfy. For instance, an encryption scheme is said to be *repetition concealing* [4] if an adversary can not detect two instances of the same message encrypted with the same key (to achieve this, probabilistic [12] or stateful encryption is needed.) Similarly, an encryption scheme is *which-key concealing* if an adversary can not deduce if two messages are encrypted under the same key [4, 5]. Besides these general properties, usually each particular cryptosystem has its own subtleties that can also provide useful information to an adversary. For example, a public key in RSA consisting of a pair (n, e) can be distinguished from a random string because e is odd and n contains no small prime factors. As discussed by Mellovin and Merritt [6], this simple fact allows a dictionary attack over EKE when instantiated with RSA.

Ideally, given a password protocol, we would like to find out which are the *minimum* properties that the underlying encryption must satisfy to regard the protocol secure. This question has received increasing attention lately, for the case of general cryptographic protocols [4, 16]. There, formal approaches are related to *computational* notions of security, based mainly on complexity and probability theories. However, we would like to address the issue by staying in a formal framework, thus keeping our security proofs manageable and general. To do this, we adopt the *applied pi calculus* [2], an extension of the pi calculus developed by Abadi and Fournet [2]. Briefly, the applied pi calculus extends the pi calculus by adding value passing, primitive function symbols, and equations between terms. The latter allows to model special abilities of the adversary that appear when encryption is weakened. Even though it is fairly new, the calculus has proven to be a practical tool for analysis; recently, Fournet and Abadi studied a protocol for achieving private authentication [11, 1], which explicitly use special assumptions about the underlying encryption (More precisely, they require encryption to be which-key concealing.)

In this paper, we study password protocols using the applied pi calculus. Our contribution is twofold:

- First, we analyse, in a precise formal framework, the security of password protocols when they are instantiated with particular encryption schemes, which may or may not satisfy specific properties. We model (most of) these properties by simply extending the equational theory of the applied pi calculus. In particular, we model encryption schemes which are repetition and which-key revealing, and also encryption schemes that allow an adversary to distinguish ciphertexts and public keys from random noise.

– Second, we study, as illustrating examples, two well-known protocols: the EPT protocol of Halevi and Krawczyk [14], and the already mentioned EKE protocol [6]. The EPT protocol is our first case study. Since it is a simple protocol, it allows us to illustrate our analysis technique. For EPT, we show that security against dictionary attacks is achieved when encryption is repetition concealing. Then, we move on to study the EKE protocol. We choose to study EKE since it is the seminal password protocol secure against dictionary attacks, with several proposals following it (see e.g. [18, 8, 15].) For the EKE protocol we show that security can be established if encryption is which-key concealing, and ciphertexts and public keys are indistinguishable from random noise. Interestingly, our analysis helped us to identify a vulnerability of EKE that arises when ciphertexts are identifiable. To solve this, we propose a simple modification to EKE that (to the best of our knowledge) is novel.

Our approach can be summarised as follows. In an off-line dictionary attack, the adversary guesses the (weak) shared password and then tries to verify the guess with the eavesdropped session. Thus, we can regard a protocol secure if it provides no such verification possibility to the adversary. In the applied pi calculus, the information gathered by the adversary can be characterized by using special processes, called *frames*. Let the frame φ represent the information of an eavesdropped session and let K be the shared, weak password (K is free in φ, representing the fact that K can be "guessed" by the adversary.) Then, we can represent the notion of security of a password protocol against dictionary attacks by seeing whether the adversary can distinguish φ from $\nu K.\varphi$, in which K is bound, representing unguessability. Intuitively, this distinction models the situation in which the adversary "hits" the correct guess, and he can verify that fact by using φ. On the other hand, if φ and $\nu K.\varphi$ are indistinguishable, then an adversary has no way of verifying from φ that a given word (from e.g. a dictionary) is actually the correct password.

2 Equational Theories for the Applied Pi Calculus

In this section we first instantiate the applied pi calculus (described in [2])[1] with our grammar for terms. We also present a standard equational theory EQ_0 that represents the standard adversary abilities. Afterwards,

[1] We also copied *verbatim* a brief overview from [2] in the Appendix for the readers's convenience.

we extend EQ_0 with EQ_1 and EQ_2, two equational theories that provide further adversary abilities.

From the applied pi calculus we use, in particular, the notions of *observational equivalence* ($\approx$) and *static equivalence* ($\approx_s$). Basically, $\approx$ relates processes in general, while $\approx_s$ relates frames, which are processes built up from active substitutions (denoted $\{x = M\}$) by parallel composition and restriction.

Our grammar for terms is shown in Table 1 (left). Besides names

U, V	::=	terms	
	$a, n, r, N_A, K...$	name	
	$x, y, ...$	variable	
	(U, V)	pair	$\mathsf{sdec}(\mathsf{senc}_r(x, y), y) = x$
	$\mathsf{fst}(U)$	field selector	$\mathsf{pdec}(\mathsf{penc}_r(x, \mathsf{pk}(y)), y) = x$
	$\mathsf{snd}(U)$	field selector	$\mathsf{fst}((x, y)) = x$
	$\mathsf{pk}(U)$	public-key derivation	$\mathsf{snd}((x, y)) = y$
	$\mathsf{hash}(U)$	cryptographic hash	
	$\mathsf{senc}_r(U, V)$	symmetric encryption	
	$\mathsf{penc}_r(U, V)$	public-key encryption	
	$\mathsf{sdec}(U, V)$	symmetric decryption	
	$\mathsf{pdec}(U, V)$	public-key decryption	

Table 1. Grammar for terms (left). Equational theory EQ_0 (right).

and variables, we have the usual pairing constructor, along with its projections. Given a name K representing a private key (and thus usually appearing restricted) we can derive a public-key $\mathsf{pk}(K)$ that can then be used for (public key) encryption. However, note that in general it may be not possible to obtain the public key from a previously created private key: Typically, the key pair is created *simultaneosly*. Thus, when a private key K is created using restriction ν, we assume that the corresponding public key is also created. Then, the constructor $\mathsf{pk}(K)$ is just a pointer to this public key. We also define the hash constructor, representing cryptographic hash.

Departing from Abadi and Fournet, our constructors for encryption (both symmetric and asymmetric) take a name r as *randomness* parameter. This allows us to model probabilistic encryption. By explicitly considering the randomness parameter as a name, say r, we can model easily repetition concealing vs. repetition revealing cryptosystems by simply restricting or not r. On the other hand, decryption is deterministic.

We also employ a standard initial equational theory, consisting of the equations shown in Table 1 (right) (EQ_0)[2]. In EQ_0 we encode all the standard abilities of the adversary, given by decryption identities and pair projection. The constructors $\mathsf{pk}(\cdot)$ and $\mathsf{hash}(\cdot)$ are modelled as non-reversible operations by implicitly not adding any ability in EQ_0. In the case of criptographic hash, not having equalities also models the property of being collision-free.

Under the standard equational theory EQ_0, non-deterministic encryption behaves as a "secure envelope", modelling a repetition concealing cryptosystem. Let us illustrate the practical implications of this with one example. Consider:

$$P(M,K) \doteq \nu r.\{x = \mathsf{penc}_r(M, \mathsf{pk}(K)), y = \mathsf{pk}(K)\}$$

Process $P(M,K)$ exports (in x) the encryption of M with $\mathsf{pk}(K)$, and also exports (in y) the used public key. Now, the property of "secure envelope" can be stated as:

$$\forall M, K : P(M,K) \approx_s \nu s, k.\{x = s, y = k\} \tag{1}$$

Here, (1) holds since r is an unguessable seed. Intuitively, (1) expresses that the resulting ciphertext of encrypting a message M with a public key derived from K is indistinguishable from random "noise".

However, one could argue that (1) models a too strong, often unrealistic encryption mechanism, in at least three ways:

(A1) First, in particular cryptosystems, it can be the case that ciphertexts are distinguishable from pure random noise, even though the plaintext or encrypting key is not leaked. For example, a usual indication of the presence of a ciphertext can be found in the length of the messages. In block ciphers, for instance, ciphertexts tipically consist of a certain number of blocks (e.g. a multiple of 128/256 bits). Similarly, numbers close to each other in RSA also give a good indication of a ciphertext. As another example, in the McEliece cryptosystem [19] every ciphertext is a *codeword*, with a small vector error added to it. This makes ciphertexts distinguishable from random noise.

(A2) Second, and similar to item 1 above, public keys can also be distinguishable from random noise in some cryptosystems, even if the private key is kept secret. As an example (already mentioned in the

[2] Plus all the equations obtained from reflexivity, symmetry, transitivity and substitution of terms for the variables x, y and r.

Introduction), a public key in RSA consists of two large naturals n and e, where e is always odd and n does not contain small prime factors.

(A3) Finally, encryption could be which-key *revealing* (i.e. not concealing), allowing an adversary not only to detect ciphertexts (as in item 1 above) but also to deduce if two ciphertexts were encrypted under the same key.

2.1 Cryptography and the Equational theory

To study the security of protocols under these more realistic scenarios, we first need to model (A1), (A2) and (A3) as adversary abilities. We achieve this with the equational theories of Table 2.

$\mathit{sym\_ciphertext}(\mathsf{senc}_r(x,y)) = \mathsf{true}$	$\mathit{same\_k}(\mathsf{senc}_r(x,k),\mathsf{senc}_s(y,k)) = \mathsf{true}$
$\mathit{pk\_ciphertext}(\mathsf{penc}_r(x,y)) = \mathsf{true}$	$\mathit{same\_k}(\mathsf{senc}_r(x,k),\mathsf{senc}_s(y,k)) = \mathsf{true}$
$\mathit{valid\_pk}(\mathsf{pk}(x)) = \mathsf{true}$	

Table 2. Extended equational theories: EQ_1 (left). EQ_2 (right).

In EQ_1, we model the ability of an adversary to distinguish ciphertexts and public keys from other, regular messages: *sym_ciphertext* detects ciphertexts created with symmetric encryption, while *pk_ciphertext* recognizes public-key ciphertexts. Similarly, *valid_pk* detects public keys. Now, the equational theory $\mathrm{EQ}_0 \cup \mathrm{EQ}_1$ models an adversary whose abilities include (A1) and (A2).

In EQ_2, we add the ability to deduce whether two messages are encrypted under the same key (with equality *same_k*.), modelling (A3).

When the equational theory is $\mathrm{EQ}_0 \cup \mathrm{EQ}_1 \cup \mathrm{EQ}_2$, (1) does not hold anymore: x can be detected by *pk_ciphertext*, y by *valid_pk* and finally x and y in conjunction can be recognized by *same_k*. However, a weaker form of the "secure envelope" still holds, namely:

$$\forall M, M', K : P(M,K) \approx_s P(M',K) \tag{2}$$

(2) states that even though an adversary can recognize a ciphertext $\mathsf{penc}_r(\cdot, \mathsf{pk}(K))$ and its encrypting public key $\mathsf{pk}(K)$, the adversary can still not glean any information about the plaintext M. Thus, this weaker notion of secure envelope is reduced to express secrecy of M (in a similar vein to the approach taken in the spi calculus [3].)

3 Encrypted Password Transmission (EPT) protocol

In this section we study the *Encrypted Password Transmission* (EPT) protocol [14]. We first present the protocol, then translate it into the calculus and finally analyse the security against dictionary attacks.

EPT is designed to be run between a server S and a user U. We assume that S and U share a weak password P, and that the server S has a strong public-private key pair. Also, U has stored a hash of S's public key, which has been previously securely communicated. The goal of the protocol is to authenticate U to S:

$$S \to U : (N, \mathsf{pk}(\mathsf{K_S})) \tag{EPT.1}$$

$$U \to S : \mathsf{penc}_r((N, P), \mathsf{pk}(K_S)) \tag{EPT.2}$$

The protocol proceeds as follows: First, the server sends in the clear to the user a message consisting of a random challenge ("nonce") N and his public key $\mathsf{pk}(K_S)$ (EPT.1). Then, the user checks that the received public key, when hashed, matches with his own (stored) hashed copy of the key. If it does not, then the user aborts. Otherwise, the user answers by encrypting a pair of N and P with the server's public key (EPT.2).

3.1 Translation in the calculus

The first thing to do is to translate the user and server into appropiate processes. Let c_{SU} be a channel name for communication from S to U and c_{US} a channel for communication from U to S. Since c_{ij} are free in the following processes U and S, an adversary (represented by the environment) can eavesdrop on these channels. We define S as:

$$\begin{aligned} S \doteq \nu K_S, N.\ (&\overline{c_{SU}}\langle(N, \mathsf{pk}(K_S))\rangle \\ &. c_{US}(y).\ \text{if } ((N, P) = \mathsf{pdec}(y, K_S)) \text{ then } P_S) \end{aligned}$$

Here, process P_S models what happens after the session was established succesfully. We assume that none of the values used during the protocol appear in P_S. If the decryption fails, then the process would abort (executing the implicit $\mathbf{0}$ of the *else* branch.)

The user process U is:

$$\begin{aligned} U \doteq \nu h, r.\ (&\{h = \mathsf{hash}(\mathsf{pk}(K_S))\} \mid c_{SU}(x).\ \text{if } (\mathsf{hash}(\mathsf{snd}(x)) = h) \text{ then} \\ &\overline{c_{US}}\langle\mathsf{penc}_r((\mathsf{fst}(x), P), \mathsf{snd}(x)\rangle.P_U) \end{aligned}$$

Similarly, P_U is the process that the user executes after succesful execution of the protocol (again, no value of the protocol appears in P_U.)

Now, a system of one user an one server can be setup by letting them share a password P: $\nu P.(U \mid S)$. A normal execution of this process can now be modelled by applying reductions and equivalences, as in [2]:

$$\nu P.(U \mid S) \rightarrow\rightarrow\equiv (P_S \mid P_U \mid \nu P, K_S, N, h, r.\varphi)$$
$$\varphi \doteq \{h = \mathsf{hash}(\mathsf{pk}(K_S)), x = (N, \mathsf{pk}(K_S)), y = \mathsf{penc}_r((\mathsf{fst}(x), P), \mathsf{snd}(x))\}$$

The first two reductions come from the message communications (EPT.1) and (EPT.2), and the last equivalence corresponds to scope tightenings. Moreover, by structural equivalence we can rewrite φ to:

$$\varphi \equiv \{h = \mathsf{hash}(\mathsf{pk}(K_S)), x = (N, \mathsf{pk}(K_S)), y = \mathsf{penc}_r((N, P), \mathsf{pk}(K_S))\}$$

Intuitively, φ (along with its restrictions) represents the information that the environment has learnt from eavesdropping a run of EPT between a user U and a server S. Next, we study whether the information recorded in φ can be exploited to mount an off-line dictionary attack.

3.2 Security against dictionary attacks

The following lemma states that the protocol EPT is secure, in the sense that it does not provide a useful verification of a guess of a password to an adversary. To model this, we make P be guessable by removing it from the restriction operator and then compare the result to the case in which it is still restricted:

Lemma 1. *Let P_S and P_U be processes where the names P, K_S, N, h and r do not appear free. Then,*

$$(P_S \mid P_U) \mid \nu P, K_S, N, h, r.\varphi \approx (P_S \mid P_U) \mid \nu K_S, N, h, r.\varphi \qquad (3)$$

The lemma follows from the fact that r is an unguessable, freshly generated seed. Because of this, the environment can never distinguish y in φ from noise, which allows us to conclude that $\nu P, K_S, h, r.\varphi \approx_s \nu K_S, h, r.\varphi$ (as in (1)). This holds even when we consider as equational theory our most powerful adversary, with $EQ_0 \cup EQ_1 \cup EQ_2$. Finally, by Lemma 2 from [2], we lift the result to observational equivalence and obtain the claim.

This uncovers the main cryptographic requirement of the protocol w.r.t. its underlying encryption: it must be repetition concealing. Discovering an attack in the case of repetition revealing (and thus deterministic) encryption is not difficult. To model deterministic encryption, we remove r from the restriction operator, thus letting the environment control it.

So, let ϕ_0 be $\nu P, K_S, N, h.\varphi$ and ϕ_1 be $\nu K_S, N, h.\varphi$. Then, $\phi_0 \not\approx_s \phi_1$. To see this, consider $\psi \doteq \nu h, t, u, s, v.\{h = \mathsf{hash}(\mathsf{pk}(u)), x = (t, \mathsf{pk}(u)), y = \mathsf{penc}_v(s, \mathsf{pk}(u))\}$. Here, $\phi_0 \approx_s \psi$. However, $\phi_1 \not\approx_s \psi$. Regarding Definition 3 of static equivalence [2], Definition 3, we can show the latter. Let $M_1 = y$ and $M_2 = \mathsf{penc}_r((\mathsf{fst}(x), P), \mathsf{snd}(x))$. Then, $(M_1 = M_2)\phi_1$ but not $(M_1 = M_2)\psi$. Thus, when encryption is not repetition concealing the protocol is not secure. This is in accordance with [14], where encryption is asked to be semantically secure [12][3].

The underlying encryption does not have to be which-key concealing to establish the security of EPT. In fact, all the abilities introduced by EQ_1 and EQ_2 do not affect (3). This makes EPT a robust protocol. Interestingly, in the next protocol we analyse (the Encrypted Key Exchange (EKE) protocol), the requirements are turned around: there, repetition concealing turns out to be irrelevant, while the other abilites (and in particular which-key concealing) are crucial to establish security against dictionary attacks.

4 Encrypted Key Exchange (EKE) protocol

In this section we analyse the Encrypted Key Exchange protocol, presented in [6]. The EKE protocol is designed to solve the problem of *authenticated key exchange* while being resistant against off-line and on-line dictionary attacks.

Differently from the EPT protocol studied in the previous section, which required U to have a stored hashed copy of the server's public key, EKE is a password-only protocol, which assumes *only* a weak shared password in common. The protocol can be described as follows:

$$A \to B : \mathsf{senc}_r(\mathsf{pk}(K), P) \tag{EKE.1}$$
$$B \to A : \mathsf{senc}_s(\mathsf{penc}_t(R, \mathsf{pk}(K)), P) \tag{EKE.2}$$
$$A \to B : \mathsf{senc}_u(N_A, R) \tag{EKE.3}$$
$$B \to A : \mathsf{senc}_v((N_A, N_B), R) \tag{EKE.4}$$
$$A \to B : \mathsf{senc}_w(N_B, R) \tag{EKE.5}$$

First, A generates a new private key K, and then derives the public key $\mathsf{pk}(K)$. Then, A encrypts $\mathsf{pk}(K)$ with the shared password P and sends it to B (EKE.1). Then, B extracts $\mathsf{pk}(K)$, generates a fresh session

[3] In fact, in [14] a stronger notion of security is required, necessary to resist active adversaries. We do not need that here, since we are dealing with passive adversaries only.

key R and encrypts it with $\mathsf{pk}(K)$. Then, B encrypts again the resulting message with P and sends it to A (EKE.2). The following three messages (EKE.i), $i = 3, 4, 5$, exchange nonces N_A and N_B to perform the "hand-shaking" necessary to defend against replay attacks.

4.1 Translation in the calculus

Let c_{AB} be a channel name for communications from A to B and c_{BA} a channel for communication from B to A. Since c_{ij} are free in the following processes A and B, a passive adversary (represented by the environment) can eavesdrop on these channels. The user process A is defined as:

$$\begin{aligned} A \doteq \nu K, r.(&\overline{c_{AB}}\langle \mathsf{senc}_r(\mathsf{pk}(K), P)\rangle \ .\ c_{BA}(x_1).(\nu k.(\{k = \mathsf{sdec}(\mathsf{pdec}(x_1, K), P)\} \\ &\mid (\nu u, N_A.\overline{c_{AB}}\langle \mathsf{senc}_u(N_A, k)\rangle \ .\ c_{BA}(x_2) \ . \\ &\text{if } (N_A = \mathsf{fst}(\mathsf{sdec}(x_2, k))) \text{ then } \nu w.\overline{c_{AB}}\langle \mathsf{senc}_w(\mathsf{snd}(\mathsf{sdec}(x_2, k)))\rangle.P_A) \end{aligned}$$

Process P_A is executed after the succesful execution of the protocol. We assume that none of the values introduced by the protocol appear in P_A, except for the exchanged session key (represented as free variable k in P_A.) Similarly, the user process B is defined as follows:

$$\begin{aligned} B \doteq (&c_{AB}(y_1) \ .\ \nu s, t, R.\overline{c_{BA}}\langle \mathsf{senc}_s(\mathsf{penc}_t(R, \mathsf{sdec}(y_1, P)), P)\rangle \\ &.\ c_{AB}(y_2) \mid \nu v, N_B.\overline{c_{BA}}\langle \mathsf{senc}_v((\mathsf{sdec}(y_2, R), N_B), R)\rangle \\ &.\ c_{AB}(y_3).\ \text{if } (N_B = \mathsf{sdec}(y_3, R)) \text{ then } P_B) \end{aligned}$$

Here, we ask the same restrictions for P_B that we asked for P_A. Similarly to the EPT protocol, we set up a session $\nu P.(A \mid B)$. Now, this protocol reduces to:

$$\nu P.(A \mid B) \rightarrow^5 \equiv\equiv \nu k.(P_A \mid P_B \mid \varphi) \tag{4}$$

Where $\rightarrow^5$ denotes five consecutive reductions, and:

$$\varphi = \nu P, K, N_A, N_B, R, r, s, t, u, v, w.\varphi_0$$

$$\begin{aligned} \text{With } \varphi_0 \doteq \{\ &y_1 = \mathsf{senc}_r(\mathsf{pk}(K), P), x_1 = \mathsf{senc}_s(\mathsf{penc}_t(R, \mathsf{sdec}(y_1, P)), P), \\ &y_2 = \mathsf{senc}_u(N_A, k), x_2 = \mathsf{senc}_v((\mathsf{sdec}(y_2, R), N_B), R) \\ &y_3 = \mathsf{senc}_w(\mathsf{snd}(\mathsf{sdec}(x_2, k)))\ \} \end{aligned}$$

The five reductions of (4) correspond to the messages exchanges (EKE.i), $i = 1, 2, 3, 4, 5$. The last two equivalences correspond to scope tightenings plus scope extrusion of k.

Finally, φ_0 can be shown equivalent (by equational rewriting) to:

$$\varphi_0 \equiv \{\ k = R, y_1 = \mathsf{senc}_r(\mathsf{pk}(K), P),\\ x_1 = \mathsf{senc}_s(\mathsf{penc}_t(R, \mathsf{pk}(K)), P), y_2 = \mathsf{senc}_u(N_A, R),\\ x_2 = \mathsf{senc}_v((N_A, N_B), R), y_3 = \mathsf{senc}_w(N_B, R)\}$$

4.2 Analysis of EKE

Consider the frame $\varphi_P = \nu K, N_A, N_B, R, r, s, t, u, v, w.\varphi_0$. Here, φ_P is the same as φ but without P being restricted. Our analysis against dictionary attacks can be carried out by relating φ to φ_P by static equivalence. By lifting the restriction on same names and by adding equational theories (EQ_0 and EQ_1) to the framework, we can analyze a range of different scenarios. We first start by weakening the underlying encryption and consider EKE being instantiated with a repetition revealing cryptosystem (although we still consider only standard abilities EQ_0.)

Repetition Concealing. An interesting fact to note in EKE is that security does not really depend on whether encryption is repetition concealing or not. To see this, consider the frame $\phi = \nu P, K, N_A, N_B, R, k.\varphi_0$ and similarly $\phi_P = \nu K, N_A, N_B, R, k.\varphi_0$. Frames ϕ and ϕ_P are the same frames as φ and φ_P respectively but with the randomness values (r, s, t, u, v and w) being "guessable". This models an encryption scheme that is not repetition concealing, since now the adversary controls the seeds and thus can detect repetitions of same messages. However, the following lemma states that this extra information cannot be exploited by the adversary. If the adversary can distinguish ϕ and ϕ_P where the seeds are known, then she could also distinguish φ and φ_P where the seeds are unguessable.

Lemma 2. *$\phi \approx_s \phi_P$ iff $\varphi \approx_s \varphi_P$*

The lemma holds since letting the adversary have the seeds r, s, t, u, v and w never helps in the task of distinguishing x_i nor y_j, for $i = 1, 2$ and $j = 1, 2, 3$. Having r does not help to recognize y_1, since $\mathsf{pk}(K)$ is indistinguishable from random noise when K is private (However this is not true anymore when considering EQ_1, as explained below.) Having s and t does not help to recognize x_1, since now R is random. The remaining cases are similar (for x_2, y_2 and y_3.) Then, we obtain that $\varphi \approx_s \phi$ and $\varphi_P \approx_s \phi_P$, which then allow us to conclude that $\phi \approx_s \phi_P$ iff $\varphi \approx_s \varphi_P$.

By proving this lemma, two crucial cryptographic requirements of the protocol are exposed:

1. Encryption may be repetition revealing since R is strong. In other words, it must be difficult to compromise R by brute-force attacking

y_2, x_2 and y_3 when encryption is repetition revealing, since otherwise x_1 could be distinguished from noise (and thus $\varphi \not\approx_s \varphi_P$.)

2. K must not be used more than once. This can be an important deficiency of EKE, since generation of new keys can sometimes be expensive. In fact, this is where the later protocol OKE [18] improves on EKE. To model a key K used twice, we represent two eavesdropped sessions sharing the same K: Let φ_K be $\nu K, P.(\nu R, N_A, N_B.\varphi_0 \mid \nu R', N'_A, N'_B.\varphi'_0)$ where φ'_0 is analogous to φ. Also let $\varphi_{K,P}$ be $\nu K.(\nu R, N_A, N_B.\varphi_0 \mid \nu R', N'_A, N'_B.\varphi'_0)$ where P is guessable. Now, $\varphi_K \not\approx_s \varphi_{K,P}$ since the environment can distinguish them by comparing $\mathsf{sdec}(y_1, P)$ and $\mathsf{sdec}(y'_1, P)$, which in $\varphi_{K,P}$ would match.

Which-key Concealing. Now we consider the possibility of an adversary to recognize under which key is a message encrypted. To model this, we consider as our equational theory $\mathrm{EQ}_0 \cup \mathrm{EQ}_2$.

We want to see if, under a cryptosystem that is not which-key concealing, $\varphi \approx_s \varphi_P$. By Lemma 2, it suffices to show that $\phi \approx_s \phi_P$. However, this is not the case, and actually we can see that $\phi \not\approx_s \phi_P$. Consider $\psi = \nu s, r, t, u, v, w.\{y_1 = r, x_1 = t, y_2 = u, x_2 = v, y_3 = w\}$. Here, $\phi \approx_s \psi$. However, $\phi_P \not\approx_s \psi$. Let $M = \mathsf{sdec}(x_1, P)$ and $N = \mathsf{penc}_w(x, \mathsf{sdec}(x_0, P)))$. Then, $(same\_k(M, N) = \mathsf{true})\phi_P$ but not $(same\_k(M, N) = \mathsf{true})\psi$.

We conclude that EKE is not secure against dictionary attacks when encryption is which-key revealing.

Identifying public keys. When we proved Lemma 2, we used the argument that $\mathsf{pk}(K)$ is indistinguishable from random noise when K is restricted and thus unguessable. If we consider $EQ_0 \cup EQ_1$ as our equational theory, this does not hold anymore. Intuitively, if an adversary is able to tell whether a public key is valid or not, an adversary could compare many eavesdropped sessions (with many messages (EKE.1)) and narrow the password space considerably, therefore mounting a succesful attack over P. The attack is exposed in our setting since when we add EQ_1, we see immediately that $\varphi \not\approx_s \varphi_P$, simply by noticing that decrypting message (EKE.1) returns a valid public key. A potential solution that is tempting here is to leave $\mathsf{pk}(K)$ unencrypted. Unfortunately, this opens another problem. In certain cryptosystems (as is the case of RSA, see [6]) a "fake" public key PK can be carefully chosen so that the space of encryptions collapses into one single value c. That is, every message M when encrypted with PK maps onto one single value c. This can be modelled

in our setting by adding the equation:

$$\mathsf{penc}_r(x, PK) = c$$

Then, if message (EKE.1) is left unencrypted, an adversary could send PK as message (EKE.1), impersonating A. Since B has no means to check that PK is indeed the "special" public key, B would continue the protocol and thus allow a dictionary attack over message (EKE.2). This vulnerability appears since B alone chooses the public key. Another version of EKE, also presented in [6] and further developed in [7], uses Diffie-Hellman to allow both parties A and B to create the key. Analysis of Diffie-Hellman exchanges is possible by modelling commutative and associative discrete exponentiation with the equational theory (see Section 5 in [2]), although we leave the analysis of EKE with Diffie-Hellman as future work.

Identifying ciphertexts. When we consider EQ_1, we can also distinguish ciphertexts. Then, a possible attack (similar to the one presented above on (EKE.1)) can be mounted on (EKE.2). Here, decrypting message (EKE.2) with a good guess returns a valid ciphertext that *pk_ciphertext* can recognize, thus allowing a potential attack. However, we can change (EKE.2) to:

$$B \to A : \mathsf{penc}_s(\mathsf{senc}_t(R, P), \mathsf{pk}(K)) \qquad \text{(EKE'.2)}$$

Now, message (EKE'.2) does not provide any verification of a guess of P, since the fact that *pk_ciphertext* can recognize (EKE'.2) is now irrelevant. Thus, the above attack is prevented. To the best of our knowledge, this modification to EKE has not been proposed before

Security against dictionary attacks. If we consider only EQ_0, that is, we assume that encryption is which-key concealing and public keys and ciphertexts are not recognizable, then we can state the security of EKE against dictionary attacks. Again, we proceed by comparing φ and φ_P. Furthermore, the next theorem shows that the established key session is in fact a *secure* key, in the sense that an adversary can not discover it by only overhearing the messages exchanged during the execution of the protocol. Let φ'_P be $\nu q, r, s, t, u, v, w.\varphi'_0$, where φ'_0 is:

$$\varphi'_0 \doteq \{y_1 = \mathsf{senc}_q(r, P), x_1 = \mathsf{senc}_s(t, P), y_2 = u, x_2 = v, y_3 = w\}$$

Then, we relate $\nu k.(P_A \mid P_B \mid \varphi)$, the result from the messages exchanges in EKE, with $\nu R.((P_A \mid P_B)\{k = R\} \mid \varphi'_P)$, in which P is made guessable but encrypting random values r and t.

Theorem 3. *Let P_A and P_B be processes with free variable k where the name R does not appear. Then,*

$$\nu k.(P_A \mid P_B \mid \varphi) \approx \nu R.((P_A \mid P_B)\{k = R\} \mid \varphi'_P)$$

Thanks to Lemma 2, it suffices to show that $\phi \approx_s \phi_P$, which can be established by case analysis over the messages of φ (using a similar argument as in Lemma 2.) Then, we can see that $\varphi_P \approx_s \varphi'_P$ to conclude that $\varphi \approx_s \varphi'_P$. To obtain the desired observational equivalence we apply Lemma 2 of [2].

Intuitively, theorem 3 says that EKE gives no verification of a guess of a password P, and that furthermore the session key k between P_A and P_B is indistinguishable from random noise to an adversary represented by the environment.

5 Conclusions

In this paper we have studied the security of password protocols against off-line dictionary attacks, using applied pi calculus of Abadi and Fournet. We argue that considering a standard adversary, with the usual restricted abilities (represented in our approach by the equational theory EQ_0), is not realistic enough to analyse thoroughly the security of password protocols. To this end, we have introduced two further equational theories EQ_1 and EQ_2, that model additional adversary abilities. The latter ability, namely which-key revealing, was already considered in [11], where the presence of such an ability would spoil immediately the privacy guarantees of the protocol. In this paper, we allow EQ_2 to enter the scene explicitly, and study whether its presence allows or not a dictionary attack. We also introduced the equational theory EQ_1, that models the ability of an adversary to distinguish ciphertexts and public keys from random noise. To the best of our knowledge, we do not know of other formal approach for verification of security protocols that considered such an adversary ability.

We also considered a further adversary ability, in which an adversary can detect repetition of same messages [4]. We model this ability by simply not restricting the used "randomness" parameter (but of course we could have achieved the same with a proper equation between terms, that

equates encryptions of same messages and same keys and any randomness.)

The ability to detect repetition of same messages, plus both EQ_1 and EQ_2 turned out to be crucial to decide the security of our case studies, EPT and EKE protocols. Moreover, we believe that our analysis helps to identify which are the precise cryptographic assumptions that a protocol needs to assume. For example, as we illustrated with the analysis of EKE, a protocol designer can decide whether to strengthen the underlying encryption or to require stronger key session and nonces, but asking for both may be unnecessary. Furthermore, our technique allows to spot possible sources of confusions and possible attacks. In particular, for EKE instantiated with an encryption scheme in which ciphertexts can be distinguished from random noise (i.e. EQ_1), we found a vulnerability. To solve this, we have proposed a simple modification (i.e. changing the orders of encryption in message (EKE.2) to (EKE'.2)) that prevents this attack.

As future work we plan to analyse the challenging problem of analysing security of password protocols subject to dictionary attacks in the presence of active adversaries.

Acknowledgments. We would like to thank Jonathan Herzog for communicating the idea that lead to this work. We would also like to thank Pieter Hartel for useful remarks.

References

1. M. Abadi. Private authentication. In *Proceedings of the 2002 Workshop on Privacy Enhancing Technologies*, LNCS, pages 27–40. Springer-Verlag, 2002.
2. M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *Proceedings of the 28th ACM Symposium on Principles of Programming Languages (POPL'01)*, pages 104–115. ACM, January 2001.
3. M. Abadi and A. Gordon. A calculus for cryptographic protocols: The spi calculus. In *Fourth ACM Conference on Computer and Communications Security*, pages 36–47. ACM Press, 1997.
4. M. Abadi and P. Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). In *Journal of Cryptology*, number 15, pages 103–127. Springer-Verlag, 2000.
5. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *Advances in Cryptology - ASIACRYPT*, volume 2482 of *Lecture Notes in Computer Science*, pages 566–582. Springer, 2001.
6. S. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. pages 72–84.

7. S. Bellovin and M. Merritt. Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise. In *ACM Conference on Computer and Communications Security*, pages 244–250, 1993.
8. V. Boyko, P. MacKenzie, and S. Patel. Provably secure password-authenticated key exchange using Diffie-Hellman. *Lecture Notes in Computer Science*, 1807:156–171, 2000.
9. R. Corin, S. Malladi, J. Alves-Foss, and S. Etalle. Guess what? here is a new tool that finds some new guessing attacks (extended abstract). In R. Gorrieri and R. Lucchi, editors, *IFIP WG 1.7 and ACM SIGPLAN Workshop on Issues in the Theory of Security (WITS)*, pages 62–71, Warsaw, Poland, Apr 2003. Dipartamento di Scienze dell'Informazione Universita di Bologna, Italy.
10. D. Dolev and A.C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.
11. C. Fournet and M. Abadi. Hiding names: Private authentication in the applied pi calculus. volume 2609 of *Lecture Notes in Computer Science*, pages 317–338. Springer-Verlag Heidelberg, January 2003.
12. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.
13. L. Gong, T. Mark A. Lomas, R. M.Needham, and J. Saltzer. Protecting poorly chosen secrets from guessing attacks. *IEEE Journal on Selected Areas in Communications*, 11(5):648–656, 1993.
14. S. Halevi and H. Krawczyk. Public-key cryptography and password protocols. *ACM Transactions on Information and System Security*, 2(3):25–60, 1999.
15. J. Katz, R. Ostrovsky, and M. Yung. Forward secrecy in password-only key exchange protocols. volume 2576, pages 29 – 44. Springer-Verlag Heidelberg, January 2003.
16. P. Lincoln, J. Mitchell, M. Mitchell, and A. Scedrov. A probabilistic poly-time framework for protocol analysis. In *ACM Conference on Computer and Communications Security*, pages 112–121, 1998.
17. Gavin Lowe. Analyzing protocols subject to guessing attacks. *Workshop on Issues in the Theory of Security (WITS'02)*, January 2002.
18. S. Lucks. Open key exchange: How to defeat dictionary attacks without encrypting public keys. In *Security Protocols, 5th International Workshop*, volume 1361 of *Lecture Notes in Computer Science*, pages 79–90. Springer, April 1997.
19. R.J. McEliece. A public–key cryptosystem based on algebraic coding theory. In *DSN Progress Report 42–44*, pages 114–116. Jet Propulsion Laboratory, Pasadena, 1978.

Appendix 1. Applied Pi Calculus Overview.

This section is copied verbatim from [2], and only added for the reviewer's convenience.

The grammar for processes is similar to the one in the pi calculus. There is only one difference: here, messages can contain terms (rather

than only names) and names need not be just channel names:

$P, Q, R ::=$	processes
0	null process
$P \mid Q$	parallel comp.
$!P$	replication
$\nu n.P$	name restriction
$\mathit{if}\ U = V\ \mathit{then}\ P\ \mathit{else}\ Q$	conditional
$u(x).P$	message input
$\overline{u}\langle V\rangle.P$	message output

Process 0 does nothing; $P \mid Q$ is the parallel composition of P and Q; $!P$ behaves as an infinite number of copies of P running in parallel. The process $\nu n.P$ makes a new name n and then behaves as P. The conditional construct $\mathit{if}\ U = V\ \mathit{then}\ P\ \mathit{else}\ Q$ is standard, only that $U = V$ represents equality. The input process $u(x).P$ is ready to input from channel u, then to run P with the actual message replaced for the formal parameter x. Finally, the output process $\overline{u}\langle V\rangle.P$ is ready to output message V on channel u, then to run P.

Also, plain processes are extended with active substitutions:

$A, B, C ::=$	extended processes
P	plain process
$A \mid B$	parallel composition
$\nu n.A$	name restriction
$\nu x.A$	variable restriction
$\{x = V\}$	active substitution

The substitution $\{x = V\}$ replaces the variable x with the term V. $\{x = V\}$ can represent the situation in which a term V has been sent to the environment, but the environment may not have the atomic names that appear in V. Still, the environment can refer to V by using the variable x. A *frame* is an extended process built up from active substitutions by parallel composition and restriction. Informally, frames represent the static knowledge gathered by the environment after communications with an extended process. Evaluation contexts, denoted $C[\_]$ are extended processes with a hole in the place of an extended process. As usual, names and variables have scopes, which are delimited by restrictions and by inputs. When E is any expression, $fv(E)$, $bv(E)$, $fn(E)$, and $bn(E)$ are the sets of free and bound variables and free and bound names of E, respectively.

5.1 Operational Semantics

Structural equivalences, written $A \equiv B$, relate extended processes that are equal by any capture-avoiding rearrangements of parallel compositions, restrictions, and active substitutions, and by equational rewriting of any terms in processes. Reductions, written $A \to B$, represent silent steps of computation (in particular, internal message transmissions and branching on conditionals). Labelled transitions, written $A \xrightarrow{\alpha} B$, represent interactions with the environment. They consist of message inputs and message outputs, respectively written $A \xrightarrow{a(U)} B$ and $A \xrightarrow{\nu \overline{a}\langle U \rangle} B$. Reductions and labelled transitions are closed by structural equivalence, hence by equational rewriting on terms.

5.2 Observational and Static Equivalences

In the analysis of protocols, we frequently argue that two given processes cannot any that is, that the processes are observationally equivalent. As in the spi calculus, the context represents an active adversary, and equivalences capture security properties in the presence of the adversary. The applied pi calculus has a useful, general theory of observational equivalence parameterized by and its equational theory [2].

5.3 Observational Equivalence

We write $A \Downarrow a$ when A can send a message on a, that is, when $A \to^* C[\overline{a}\langle M \rangle.P]$ for some evaluation context $C[\_]$ that does not bind a.

Definition 4. *Observational equivalence ($\approx$) is the largest symmetric relation $\mathcal{R}$ between closed extended processes with the same domain such that $A \mathcal{R} B$ implies:*

1. *if $A \Downarrow a$, then $B \Downarrow a$;*
2. *if $A \to^* A'$, then $B \to^* B'$ and $A' \mathcal{R} B'$ for some B';*
3. *$C[A] \mathcal{R} C[B]$ for all closing evaluation contexts $C[\_]$.*

5.4 Static equivalence

Static equivalence, written $\approx_s$, relates frames that cannot be distinguished by any term comparison.

Definition 5. *We say that two terms M and N are equal in the frame φ, and write $(M = N)\varphi$, if and only if $\varphi \equiv \nu n.\sigma$, $M\sigma = N\sigma$ and $\{n\} \cap (fn(M) \cup fn(N)) = \emptyset$ for some names n and substitution σ.*

Definition 6. *We say that two closed frames φ and ψ are* statically equivalent*, and write $\varphi \approx_s \psi$, when $dom(\varphi) = dom(\psi)$ and when, for all terms M and N, we have $(M = N)\varphi$ if and only if $(M = N)\psi$.*

We say that two closed extended processes are statically equivalent, and write $A \approx_s B$, when their frames are statically equivalent.

Lemma 7. *[2]. Observational equivalence and static equivalence coincide on frames. Observational equivalence is strictly finer than static equivalence on extended processes: $\approx \subset \approx_s$.*