Available online at www.sciencedirect.com
SCIENCE DIRECT
Electronic Notes in Theoretical Computer Science
ELSEVIER Electronic Notes in Theoretical Computer Science 121 (2005) 47–63
www.elsevier.com/locate/entcs

Analysing Password Protocol Security Against Off-line Dictionary Attacks

Ricardo Corin[1]

Faculty of Computer Science, University of Twente, The Netherlands

Jeroen Doumen[2]

Faculty of Computer Science, University of Twente, The Netherlands

Sandro Etalle[3]

Faculty of Computer Science, University of Twente, The Netherlands
CWI, Center for Mathematics and Computer Science Amsterdam

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 we present novel equational theories to model the (new) adversary abilities.These new abilities are crucial in the analysis of our case studies, the *Encrypted Password Transmission* (EPT) protocol of Halevi and Krawczyk, and the well-known *Encrypted Key Exchange* (EKE) of Bellovin and Merritt. In the latter, we find an attack that arises when considering the ability of distinguishing ciphertexts from random noise. We propose a modification to EKE that prevents this attack.

Keywords: Password protocols, dictionary attacks, verification, pi calculus.

[1] Email: corin@cs.utwente.nl
[2] Email: doumen@cs.utwente.nl
[3] Email: etalle@cs.utwente.nl

doi:10.1016/j.entcs.2004.10.007

1 Introduction

Due to the low entropy available in user-chosen passwords, password protocols are usually subject to *off-line dictionary attacks*. 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 [13,7,8]. These approaches are based on the usual Dolev-Yao adversary, and assume *ideal* or *perfect* encryption: ciphertexts do not leak *any* information to an adversary that does not have the correct key.

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 dependent on the chosen cryptosystem. Typically, the 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* [2] if an adversary cannot detect two instances of the same message encrypted with the same key (to achieve this, probabilistic [10] or stateful encryption is needed). Similarly, an encryption scheme is *which-key concealing* if an adversary cannot deduce if two messages are encrypted under the same key [2]. 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 [3], this simple fact allows a dictionary attack over EKE when instantiated with RSA.

In this paper, we study password protocols using the applied pi calculus [1]. Our contribution is twofold: First, we show how to 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 extending the equational theory of the applied pi calculus. In particular, we show how to 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 [11], and the already mentioned EKE protocol [3]. For EPT, we show that security against dictionary attacks is achieved when encryption is repetition concealing. 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 that (to the best of our knowledge) is novel.

2 Applied Pi Calculus

We first *very* briefly summarize the Applied Pi Calculus from Abadi and Fournet, borrowing from [1] (with definitions copied *verbatim*). The process grammar is similar to the one in the pi calculus, differing in the fact that messages can contain *terms* and names need not be just channel names:

$$P, Q, R ::= 0 \mid P|Q \mid !P \mid \nu n.P \mid if\ U = V\ then\ P\ else\ Q \mid u(x).P \mid \mid \overline{u}\langle V\rangle.P$$

We will describe our term grammar and its equational theory later, in Section 2.1.

In this calculus, plain processes as defined above are extended with *active substitutions* $\{x = V\}$, meaning the replacement of 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 x.

We use the standard notions of equivalences between processes, *reduction* ($\rightarrow$), structural equivalences ($\equiv$) and observational equivalence ($\approx$) as defined in [1] (we do not define them here, due to space constraints and the fact that we will mainly use *static equivalence*, described below).

Frames, ranging over $\phi, \varphi, ...$ are processes built up from active substitutions by parallel composition and restriction. The domain $dom(\phi)$ of a frame ϕ are all the variables that the frame exports, ie. the variables such that $\{x = V\}$ is in ϕ and x is not restricted. Given a term M, we denote $fn(M)$ as the set of free names appearing in M. Now we can define when two terms are equal in a given frame (this definition and the next one follow from [1]).

Definition 2.1 We say that two terms M and N are equal in the frame φ, and write $(M = N)\varphi$, if and only if $\varphi \equiv \nu\widetilde{n}.\sigma$, $M\sigma = N\sigma$ and $\{\widetilde{n}\} \cap (fn(M) \cup fn(N)) = \emptyset$ for some names $\widetilde{n}$ and substitution σ.

Example 2.2 We borrow an example from [1]. Let f and g be two functions with no equations (they can be thought as two independent one-way functions). Let $\phi_0 = \nu K, s.\{x = K, y = s\}$, $\phi_1 = \nu K.\{x = \mathsf{f}(K), y = \mathsf{g}(K)\}$, and $\phi_2 = \nu K.\{x = K, y = \mathsf{f}(K)\}$. In ϕ_0, x and y are unrelated, and hence they cannot be distinguished by any pair of terms M and N. The same happens in ϕ_1. However, in ϕ_2, x and y are related: y is obtained from x by applying f. More formally, let $M = \mathsf{f}(x)$ and $N = y$. Then $(M = N)\phi_2$ but *not* $(M = N)\phi_i$ for $i \in [0, 1]$.

The example motivates the definition of *static equivalence* $\approx_s$:

Definition 2.3 We say that two closed (i.e. with no free variables) 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 $(M = N)\psi$.

Given two closed extended processes A and B, we can "extract" their frames $\phi(A)$ and $\phi(B)$, by replacing every plain subprocess of A and B with $\mathbf{0}$. We then say that two closed extended processes are statically equivalent, and write $A \approx_s B$, when their frames are statically equivalent. We quote a useful lemma from [1].

Lemma 2.4 *[1].*

(i) *Observational equivalence and static equivalence coincide on frames. Observational equivalence is strictly finer than static equivalence on extended processes:* $\approx \subset \approx_s$.

(ii) *Static equivalence is closed under reduction and structural equivalence.*

Now we are ready to instantiate the calculus with our term grammar and equational theory.

2.1 Equational Theories for the Applied Pi Calculus

In this section we first instantiate the applied pi calculus with our term grammar. Then, we present the equational theory EQ_0 that represents the standard adversary abilities. Afterwards, we extend EQ_0 with EQ_1 and EQ_2, two equational theories that provide further adversary abilities. Our grammar for terms is shown in Table 1 (left).

Besides names 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 [4]. We also define the usual hash constructor. 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. revealing cryptosystems by simply restricting or not r (This modelling was already suggested in [1]). On the other hand, decryption is deterministic.

[4] 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 *simultaneously*. 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.

U, V	$::=$	terms
	$a, n, r, N_A, K...$	name
	$x, y, ...$	variable
	(U, V)	pair
	$\mathsf{fst}(U)$	field selector
	$\mathsf{snd}(U)$	field selector
	$\mathsf{pk}(U)$	public-key derivation
	$\mathsf{hash}(U)$	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

$$
\begin{aligned}
\mathsf{sdec}(\mathsf{senc}_r(x, y), y) &= x \\
\mathsf{pdec}(\mathsf{penc}_r(x, \mathsf{pk}(y)), y) &= x \\
\mathsf{fst}((x, y)) &= x \\
\mathsf{snd}((x, y)) &= y
\end{aligned}
$$

Table 1
Grammar for terms (top). Equational theory EQ_0 (bottom).

The standard equational theory EQ_0 is shown[5] in Table 1 (right). In EQ_0 we encode all the standard abilities of the adversary, given by decryption identities and pair projection. Here, $\mathsf{pk}(\cdot)$ and $\mathsf{hash}(\cdot)$ are modelled as non-reversible operations by implicitly not adding any ability in EQ_0.

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 $P(M) \doteq \nu r, K.\{x = \mathsf{penc}_r(M, \mathsf{pk}(K)), y = \mathsf{pk}(K)\}$. Process $P(M)$ 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 $\forall M : P(M) \approx_s \nu s, k.\{x = s, y = k\}$, which intuitively holds since r is an

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

unguessable seed. This expresses that the resulting ciphertext of encrypting a message M with a public key derived from K is indistinguishable from random "noise" (analogously for the public key derived from K). Notice that this holds even if M is possibly known (or guessable) by the environment.

However, one could argue that this approach 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 (this can happen, for example, when padding is weak or non-existent). In block ciphers, for instance, ciphertexts typically 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 [15] 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, even if the private key is kept secret. As an example (already mentioned in the 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*, 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.

2.2 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.

In EQ_1, we model the ability of an adversary to distinguish ciphertexts and public keys from regular messages: *sym_ciphertext* detects ciphertexts created with symmetric encryption, while *pk_ciphertext* recognizes public-key ciphertexts. Similarly, *valid_pk* detects public keys. Now, $\mathrm{EQ}_0 \cup \mathrm{EQ}_1$ 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 $\mathrm{EQ}_0 \cup \mathrm{EQ}_1 \cup \mathrm{EQ}_2$, the secure envelope property does not hold any more: x can be detected by *pk_ciphertext*, y by *valid_pk*

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

Table 2
Extended equational theories: EQ_1 (top). EQ_2 (bottom).

and finally x and y in conjunction can be recognized by *same_k*. However, a weaker form of a "secure envelope" still holds, namely $\forall M, M' : P(M) \approx_s P(M')$, which 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.

2.3 Modelling Dictionary Attacks

In an off-line dictionary attack, the adversary guesses the (weak) shared password and then tries to verify the guess with the eavesdropped session [13]. Thus, we can regard a protocol as secure if it provides no such verification possibility to the adversary. As we already mentioned, in the applied pi calculus the information gathered by the adversary can be characterized by using *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 checking whether the adversary can distinguish φ from $\nu K.\varphi$, in which K is bound, representing unguessability. More precisely, by "seeing" we use static equivalence ($\approx_s$), as defined in Definition 2.3.

Definition 2.5 Let ϕ be a frame in which K is free. Then we say that ϕ verifies a guess of K if $\phi \not\approx_s \nu K.\phi$. Conversely, we say that ϕ is secure w.r.t. K if $\phi \approx_s \nu K.\phi$.

Intuitively, a distinction of φ and $\nu K.\varphi$ 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.

For example, if $\phi = \nu Na.\{x = (Na, K)\}$, then $\nu K.\phi \not\approx_s \phi$. Intuitively, this

follows from the fact that K is free (guessable!) in ϕ, and thus can be used in a term to distinguish x from random noise. To see this formally, we let $M = \mathsf{snd}(x)$ and $N = K$. Then $(M = N)\phi$ but not $(M = N)\nu K.\phi$. On the other hand, if $\phi = \nu Na.\{x = \mathsf{senc}_r(Na, K)\}$ then $\nu K.\phi \approx_s \phi$. To see the $(\Rightarrow)$ of this claim is easy; we show the converse. Let M and N be such $(M = N)\phi$. Intuitively, we need to see that using K does not help in the equality of M and N in ϕ. We can assume that Na does not occur in M or N (because anyway we can rewrite Na to Na' in ϕ, by alpha conversion). The only case in which M and N can use K in ϕ is by decrypting x; for example, $M = \mathsf{sdec}(x, K)$. But in this case N cannot be Na, and so for $(M = N)\phi$ to hold we must set $N = M = \mathsf{sdec}(x, K)$. But this implies that $(M = N)\nu K.\phi$ also, by alpha converting K (to some suitable K') in ϕ.

We then obtain the following result:

Proposition 2.6 *Consider a process P, and suppose that K is free in $\phi(P)$. Suppose also that $P \rightarrow^* \equiv P'$. If $\phi(P)$ verifies K, then $\phi(P')$ verifies K. Also, if $\phi(P')$ is secure w.r.t. to K, then $\phi(P)$ is secure w.r.t. to K.*

The proposition follows from the fact that static equivalence is closed under reduction and structural equivalence, Lemma 2.4 (2).

Now we are ready to proceed to our case studies.

3 Encrypted Password Transmission (EPT) protocol

In this section we study the *Encrypted Password Transmission* (EPT) protocol [11]. 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:

$$S \rightarrow U : (N, \mathsf{pk}(\mathsf{K_S})) \qquad \text{(EPT.1)}$$
$$U \rightarrow S : \mathsf{penc}_r((N, P), \mathsf{pk}(K_S)) \qquad \text{(EPT.2)}$$

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

First, we translate the user and server into appropriate 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. By keeping c_{ij} free in the processes U and S, we allow adversaries (i.e. the environment) can eavesdrop on these channels. We define S as:

$$S \doteq \nu K_S, N. \quad (\overline{c_{SU}}\langle(N, \mathsf{pk}(K_S))\rangle \;.\; c_{US}(y). \\ \text{if } ((N, P) = \mathsf{pdec}(y, K_S)) \text{ then } P_S) \mid \{pks = \mathsf{pk}(K_S)\}$$

Process P_S models what happens after the session was established successfully. Note how the public key $\mathsf{pk}(K_S)$ is exported in variable pks. We assume that none of the values used during the protocol appear in P_S, and also that P_S never discloses K_S. If the decryption fails, then the process would abort (executing the implicit $\mathbf{0}$ of the *else* branch). The user process U is:

$$U \doteq \nu r. \quad (\; c_{SU}(x).\ \text{if } (\mathsf{hash}(\mathsf{snd}(x)) = \mathsf{hash}(pks)) \text{ then} \\ \overline{c_{US}}\langle \mathsf{penc}_r((\mathsf{fst}(x), P), \mathsf{snd}(x)\rangle.P_U)$$

Similarly, P_U is the process that the user executes after successful execution of the protocol (again, no value of the protocol appears in P_U). Now, a system of one user and 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 [1]:

$$\nu P.(U \mid S) \rightarrow\rightarrow\equiv (P_S \mid P_U \mid \nu P, K_S, N, r.\varphi)$$

$$\varphi \doteq \{x = (N, \mathsf{pk}(K_S)), y = \mathsf{penc}_r((\mathsf{fst}(x), P), \mathsf{snd}(x)), pks = \mathsf{pk}(K_S)\}$$

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 and obtain:

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

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.

Thanks to the second part of Proposition 2.6, if we can regard φ secure

w.r.t. to P, then we can be sure that every other earlier frame is also secure w.r.t. to K_S.

3.2 Security against dictionary attacks

The following lemma states that EPT is secure, in the sense that it does not provide a verification of a guess of a password to an adversary. To model this, we let P be guessable by removing it from the restriction, and then compare the result to the case in which it is still restricted, as in our formalization of guessing attacks in Definition 2.5:

Lemma 3.1 *Let P_S and P_U be processes where the names P, K_S, N and r do not appear free. Then:*

$$(P_S \mid P_U) \mid \nu P, K_S, N, r.\varphi \approx (P_S \mid P_U) \mid \nu K_S, N, r.\varphi \tag{3}$$

Proof. We first establish:

$$\nu P, K_S, N, r.\varphi \approx_s \nu K_S, N, r.\varphi \tag{4}$$

The case ($\Rightarrow$) is straightforward, so we show the converse. Let M and N be terms s.t. $(M = N)\nu K_S, N, r.\varphi$. We first note that K_S is not obtainable from ϕ, since constructor $\mathsf{pk}(\cdot)$ has no inverse in the equational theory. If P does not occur in M or N, then it is easy to obtain the claim. Suppose now P does occur in M or N. Since the private key K_S is not obtainable from ϕ, then y in ϕ can not be decrypted; then P can only be used in M or N to encrypt a term similar to y. But this is impossible since r is restricted in y, and we then obtain that $(M = N)\nu P, K_S, N, r.\varphi$, which establishes (4). Note that 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 2.4 (1), we lift the result to observational equivalence ($\approx$). □

Intuitively, the lemma follows from the fact that r is an unguessable, freshly generated seed, and then environment can never distinguish y in φ from noise.

While proving this lemma, the main cryptographic requirement of the protocol w.r.t. its underlying encryption is uncovered: 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 $\nu P, K_S, N.\varphi$ and ϕ_1 be $\nu K_S, N.\varphi$. Then, $\phi_0 \not\approx_s \phi_1$. 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

$(M_1 = M_2)\phi_0$. Thus, when encryption is not repetition concealing the protocol is not secure. This is in accordance with [11], where encryption is asked to be semantically secure [10] [6] .

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 EKE protocol), the requirements are the other way around: repetition concealing turns out to be irrelevant, while the feature of being which-key concealing is 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 [3]. The EKE protocol is designed to solve the problem of *authenticated key exchange* while being resistant against 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 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

The user process A can be defined as:

[6] In fact, in [11] 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.

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

Process P_A is executed after the successful 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 $\nu P.(A \mid B) \rightarrow^5 \equiv\equiv \nu k.(P_A \mid P_B \mid \varphi)$, where $\varphi = \nu P, K, N_A, N_B, R, r, s, t, u, v, w.\varphi_0$, with:

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

The five above reductions 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 to:

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

4.2 Analysis of EKE

Consider the frame $\varphi_P = \nu P, K, N_A, N_B, R, r, s, t, u, v, w.\varphi_0$. Here, φ_{P^-} is the same as φ_P but without P being restricted. Our analysis against dictionary attacks can be carried out by relating φ_P to φ_{P^-} by static equivalence. By lifting the restriction on same names and by adding equational theories (EQ_1 and EQ_2) to the framework, we can analyze a range of different scenarios. We first start by weakening the underlying encryption and consider EKE be-

ing 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 $\phi_P = \nu P, K, N_A, N_B, R, k.\varphi_0$ and similarly $\phi_{P^-} = \nu K, N_A, N_B, R, k.\varphi_0$. Frames ϕ_P and ϕ_{P^-} are the same frames as φ_P 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_P and ϕ_{P^-} where the seeds are known, then she could also distinguish φ_P and φ_{P^-} where the seeds are unguessable.

Lemma 4.1 *$\phi_P \approx_s \phi_{P^-}$ iff $\varphi_P \approx_s \varphi_{P^-}$.*

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 $\varphi_P \approx_s \phi_P$ and $\varphi_{P^-} \approx_s \phi_{P^-}$, which then allow us to conclude that $\phi_P \approx_s \phi_{P^-}$ iff $\varphi_P \approx_s \varphi_{P^-}$.

While establishing the lemma, we see two cryptographic requirements of EKE:

(i) 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 $\varphi_P \not\approx_s \varphi_{P^-}$.)

(ii) 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 [14] improves on EKE. To model a key K used twice, we represent two sessions sharing the same K: Let $\varphi_{K,P}$ 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_0. Similarly, let φ_{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, $\varphi_{K,P} \not\approx_s \varphi_{K,P^-}$ since the environment can distinguish them by comparing $\mathsf{sdec}(y_1, P)$ and $\mathsf{sdec}(y'_1, P)$.

Which-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, $\varphi_P \approx_s \varphi_{P^-}$. By Lemma 4.1, it suffices to show that $\phi_P \approx_s \phi_{P^-}$. However, this is not the case, and actually we can see that $\phi_P \not\approx_s \phi_{P^-}$. Let $M = \mathsf{sdec}(x_1, P)$ and $N = \mathsf{penc}_w(x, \mathsf{sdec}(y_1, P))$. Then, $(same\_k(M, N) = \mathsf{true})\phi_{P^-}$ but not $(same\_k(M, N) = \mathsf{true})\phi_P$.

Identifying public keys. While showing Lemma 4.1, 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 successful attack over P. The attack is exposed in our setting since when we add EQ_1, we see immediately that $\varphi_P \not\approx_s \varphi_{P^-}$, simply by noticing that decrypting message (EKE.1) returns a valid public key: Let $M = valid\_pk(\mathsf{sdec}(y_1, P))$ and $N = \mathsf{true}$. Then $(M = N)\phi_P$ but not $(M = N)\phi_{P^-}$.

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 (EKE.2) with a good guess returns a valid ciphertext that *pk_ciphertext* can recognize, thus allowing an attack. However, we propose to 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 if K is secret. 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. The next theorem shows that the established key session is a *secure* key, i.e. an adversary cannot 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 $\nu k.(P_A \mid P_B \mid \varphi_P)$, the message 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 4.2 *Let P_A and P_B be processes with free variable k where R does not appear. Then, $\nu k.(P_A \mid P_B \mid \varphi_P) \approx \nu R.((P_A \mid P_B)\{k = R\} \mid \varphi'_{P^-})$.*

Proof Sketch. Thanks to Lemma 4.1, it suffices to show that $\phi \approx_s \phi_{P^-}$, which can be established by case analysis over the messages of φ_P as in Lemma 4.1. Then, we can see that $\varphi_{P^-} \approx_s \varphi'_{P^-}$ to conclude that $\varphi_P \approx_s \varphi'_{P^-}$. To obtain the desired observational equivalence we apply Lemma 2.3.

Intuitively, Theorem 4.2 says that EKE gives no verification of a guess of P, and 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 the applied pi calculus. 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 [9], 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 security protocol analysis that considered such an adversary ability. We also considered the ability in which an adversary can detect repetition of same messages.

These non-standard abilities 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 rely on. 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 one 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 new vulnerability. To solve this, we have proposed a simple modification (i.e. to change the order of encryption in (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. Another possible venue is to apply our technique to study the rich field of (later) proposed password protocols (e.g., the protocols proposed in [14,6,12].)

Recently, Blanchet proposed an extension to the tool Proverif [4] to prove (automatically) *strong secrecy* for security protocols [5]. Indeed, their (independently developed) notion of strong secrecy, presented in Definition 2 of [5], generalizes our Definition 2.5. Thus, it would be interesting to validate the (manual) proofs presented in this paper using Blanchet's technique.

Acknowledgements. We would like to thank Pieter Hartel and the anonymous reviewers for helpful comments. We would also like to thank Jonathan Herzog for mentioning a helpful comment that lead to this work. This work was carried out in the context of the LicenseScript project, supported by the Telematica Instituut.

References

[1] 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.

[2] 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, 2000.

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

[4] B. Blanchet. An efficient cryptographic protocol verifier based on Prolog rules. In *14th IEEE Computer Security Foundations Workshop (CSFW-14)*, pages 82–96. IEEE Computer Society, 2001.

[5] Bruno Blanchet. Automatic Proof of Strong Secrecy for Security Protocols. In *IEEE Symposium on Security and Privacy*, pages 86–100, Oakland, California, May 2004.

[6] V. Boyko, P. MacKenzie, and S. Patel. Provably secure password-authenticated key exchange using Diffie-Hellman. *LNCS*, 1807:156–171, 2000.

[7] 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.

[8] S. Delaune. Intruder deduction problem in presence of guessing attacks. In *Proc. Workshop on Security Protocols Verification (SPV'2003), Marseille, France, Sep. 2003*, pages 26–30, 2003.

[9] C. Fournet and M. Abadi. Hiding names: Private authentication in the applied pi calculus. volume 2609 of *LNCS*, pages 317–338. Springer, January 2003.

[10] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.

[11] S. Halevi and H. Krawczyk. Public-key cryptography and password protocols. *ACM Transactions on Information and System Security*, 2(3):25–60, 1999.

[12] J. Katz, R. Ostrovsky, and M. Yung. Forward secrecy in password-only key exchange protocols. volume 2576, pages 29 – 44. Springer, January 2003.

[13] G. Lowe. Analyzing protocols subject to guessing attacks. *Workshop on Issues in the Theory of Security (WITS'02)*, January 2002.

[14] S. Lucks. Open key exchange: How to defeat dictionary attacks without encrypting public keys. In *Security Protocols, 5th International Workshop*, volume 1361 of *LNCS*, pages 79–90. Springer, April 1997.

[15] 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.