Exercise Set 6

Information Security Lab
October 25-26, 2022 (Week 6)

Prof. David Basin
Sofia Giampietro
Xenia Hofmeier

Note: This exercise sheet does not require or benefit from the use of Tamarin.

Preliminary Definitions

In the following exercises we make use of the equational theories defined below. These theories model pairs of terms, hashes, symmetric encryption and asymmetric encryption. Recall from the lectures that that an equational theory is compromised of a function signature and a set of equations. A function signature is a set of function symbols and their arities. The equations are (implicitly) universally quantified.

$$\left.\begin{array}{l} \mathsf{fst}/1,\ \mathsf{snd}/1,\ \mathsf{pair}/2 \\ \\ \mathsf{fst}(\mathsf{pair}(x,y)) = x \\ \mathsf{snd}(\mathsf{pair}(x,y)) = y \end{array}\right\} := E_p$$

$$\left.\begin{array}{l} \mathsf{h}/1 \\ \\ \emptyset \end{array}\right\} := E_h$$

$$\left.\begin{array}{l} \mathsf{senc}/2,\ \mathsf{sdec}/2 \\ \\ \mathsf{sdec}(\mathsf{senc}(m,k),k) = m \end{array}\right\} := E_{se}$$

$$\left.\begin{array}{l} \mathsf{aenc}/2,\ \mathsf{adec}/2,\ \mathsf{pk}/1 \\ \\ \mathsf{adec}(\mathsf{aenc}(m,\mathsf{pk}(sk)),sk) = m \end{array}\right\} := E_{ae}$$

Where convenient, we make use of common shorthand for these functions. We write $\langle x, y, z\rangle$ for $\mathsf{pair}(x, \mathsf{pair}(y, z))$ and so on. We write $\{m\}_k$ and $\{m\}_{\mathsf{pk}(a)}$ for $\mathsf{senc}(m, k)$ and $\mathsf{aenc}(m, \mathsf{pk}(a))$ respectively. We also overload notation and write $E_x \cup E_y$ for the combination of two equational theories (with disjoint signatures) formed by the union of their respective signatures and sets of equations.

Part 1: Term Rewriting Systems

1.1 Equalities

Under the equational theory $E = E_p \cup E_h \cup E_{se} \cup E_{ae}$, determine which of the following equations hold. The symbols $x, y, z, a, b, c, d, k, m$ are constants.

$$\begin{aligned}
\mathsf{pair}(\mathsf{fst}(\mathsf{pair}(x,y)), \mathsf{snd}(\mathsf{pair}(x,y))) &=_? \mathsf{pair}(x,y)\\
\mathsf{pair}(\mathsf{fst}(\mathsf{pair}(x,y)), z) &=_? \mathsf{pair}(x,z)\\
\mathsf{sdec}(\mathsf{senc}(\mathsf{pair}(a,b), \mathsf{pair}(c,d)), \mathsf{pair}(c,d)) &=_? \mathsf{pair}(a,b)\\
\mathsf{pair}(\mathsf{senc}(k,m), \mathsf{aenc}(k, \mathsf{pk}(a))) &=_? \mathsf{pair}(\mathsf{aenc}(k, \mathsf{pk}(a)), \mathsf{senc}(k,m)))\\
\mathsf{sdec}(\mathsf{aenc}(m, \mathsf{pk}(a)), \mathsf{pk}(a))) &=_? m
\end{aligned}$$

Answer:

$$\begin{aligned}
\mathsf{pair}(\mathsf{fst}(\mathsf{pair}(x,y)), \mathsf{snd}(\mathsf{pair}(x,y))) &= \mathsf{pair}(x,y)\\
\mathsf{pair}(\mathsf{fst}(\mathsf{pair}(x,y)), z) &= \mathsf{pair}(x,z)\\
\mathsf{sdec}(\mathsf{senc}(\mathsf{pair}(a,b), \mathsf{pair}(c,d)), \mathsf{pair}(c,d)) &= \mathsf{pair}(a,b)\\
\mathsf{pair}(\mathsf{senc}(k,m), \mathsf{aenc}(k, \mathsf{pk}(a))) &\neq \mathsf{pair}(\mathsf{aenc}(k, \mathsf{pk}(a)), \mathsf{senc}(k,m)))\\
\mathsf{sdec}(\mathsf{aenc}(m, \mathsf{pk}(a)), \mathsf{pk}(a))) &\neq m
\end{aligned}$$

1.2 Deductions

Using E as defined in the previous problem and letting a, b, i, N_z, K_z be constants. Consider an adversary who has learned the following set of terms I through observing network communication or compromising agents in a system:

$$I := \begin{cases} \mathsf{pk}(a), \quad \mathsf{pk}(b), \quad \mathsf{pk}(i), \quad \mathsf{sk}(i), \quad \mathsf{h}(K_2), \quad K_1, \quad \{N_3\}_{\mathsf{pk}(i)} \\ \{\langle \mathsf{h}(N_1), N_3\rangle\}_{\mathsf{pk}(b)}, \qquad \{\langle N_1, N_2\rangle\}_{\mathsf{h}(K_1)}, \qquad \{N_4\}_{\mathsf{h}(\langle N_1, K_2\rangle)}, \end{cases}$$

For each term t in the set T below, show how the adversary could deduce t by applying function symbols from E to the elements in I or explain why t cannot be deduced by the adversary.

$$\mathsf{h}(\langle \mathsf{h}(N_1), N_3\rangle) \qquad\qquad N_4 \qquad\qquad \{\langle N_2, \mathsf{h}(N_3)\rangle\}_{\mathsf{pk}(a)}$$

Answer:

- $\mathsf{h}(\langle \mathsf{h}(N_1), N_3\rangle)$ Yes, the adversary can deduce this first term: The adversary can deduce N_3 from $\{N_3\}_{\mathsf{pk}(i)}$ and $\mathsf{sk}(i)$ by applying the equation for asymmetric encryption. In a second step, he can hash K_1 to get $h(K_1)$ and then use it to decrypt $\{\langle N_1, N_2\rangle\}_{\mathsf{h}(K_1)}$ using symmetric decryption and get $\langle N_1, N_2\rangle$. The adversary can then apply fst and get N_1. He can then hash it to obtain $h(N_1)$. Finally he can apply pair to $h(N_1)$ and N_3 and then hash the result.
- N_4 No, the adversary can not deduce this term. N_4 only occurs in $\{N_4\}_{\mathsf{h}(\langle N_1, K_2\rangle)}$. To

decrypt this, we would need $h(\langle N_1, K_2\rangle)$. Since we don't have this hash value, we would need to construct it, i.e. we would need $\langle N_1, K_2\rangle$. We already know that the adversary can deduce N_1. Thus we only need K_2. The only place where it occurs is $\mathsf{h}(K_2)$. But we can not extract K_2 from the hash. Thus, N_4 cannot be decrypted.

- $\{\langle N_2, \mathsf{h}(N_3)\rangle\}_{\mathsf{pk}(a)}$ The term is deducible. The adversary can extract N_2 from K_1 and $\{\langle N_1, N_2\rangle\}_{\mathsf{h}(K_1)}$. This is done similarly as for N_1 in the first point. We also in the first point that the adversary can extract N_3 from $\{N_3\}_{\mathsf{pk}(i)}$ and $\mathsf{sk}(i)$. Thus he can deduce $\langle N_2, \mathsf{h}(N_3)\rangle$, by hashing N_3 and applying the pair function. Then the adversary can apply asymmetric encryption with $\mathsf{pk}(a)$ to get the desired term.

1.3 Diffie-Hellman Groups

We now consider E_{dh}, the equational theory for Diffie-Hellman groups. It defines the function symbols $inv/1$, $1/0$, $exp/2$ and $*/2$. We use $exp(g, a)$, written g^a to denote exponentiation in the group, $*$ to denote multiplication in the exponent, 1 to denote the identity element and inv to represent a multiplicative inverse. These symbols are governed by the following equations:

$$\begin{aligned}(x^y)^z &= x^{(y*z)}\\ x^1 &= x\\ x*y &= y*x\\ (x*y)*z &= x*(y*z)\\ x*1 &= x\\ x*inv(x) &= 1\end{aligned}$$

Explain how the resulting equational theory encodes the Discrete Logarithm hardness assumption. How could this theory be extended to model an attacker who could break the discrete logarithm?

Answer: The discrete logarithm problem (DLP) can be stated as follows:

Given a finite group G, given elements $g \in G$ and $\beta \in \langle g\rangle$, find x such that $g^x = \beta$.

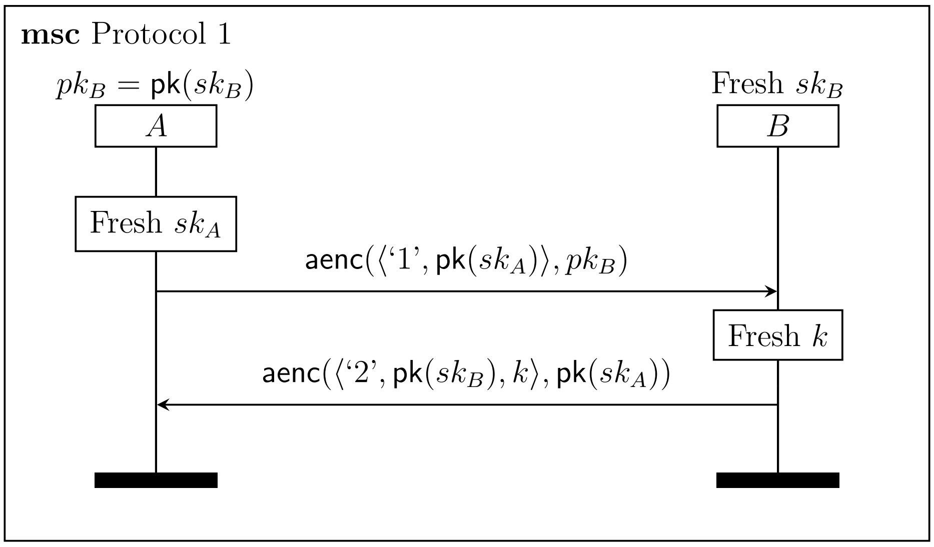
We call such x above the discrete logarithm of β. The DLP is generally considered a hard problem, i.e. there is no known *efficient* algorithm for this problem in the generic setting. The equational theory above models this hardness assumption by not containing any equation that allows an adversary to get x from g^x.

To model an attacker who can break the discrete logarithm, we could add a function $\log/2$ and the equation $\log(g^x, g) = x$ which describes calculating the discrete logarithm.

Part 2: Transition Systems and Traces

2.1 Encoding a Protocol as Multiset Rewriting Rules

Protocol 1 is described in the MSC diagram below. Translate Protocol 1 into multiset rewriting rules. Your translation should require **at most** five rewrite rules.


msc Protocol 1
$pk_B = \mathsf{pk}(sk_B)$
Fresh sk_B
A
B
Fresh sk_A
$\mathsf{aenc}(\langle\text{‘1’}, \mathsf{pk}(sk_A)\rangle, pk_B)$
Fresh k
$\mathsf{aenc}(\langle\text{‘2’}, \mathsf{pk}(sk_B), k\rangle, \mathsf{pk}(sk_A))$

Answer:

$$[\mathsf{Fr}(sk_x)] \mapsto [\mathsf{Agent}(\$X, sk_x), !\mathsf{PublicKey}(\$X, \mathsf{pk}(sk_x))]$$

$$[\mathsf{Agent}(\$A, sk_A), !\mathsf{PublicKey}(\$B, pk_B)] \mapsto [\mathsf{Out}(\mathsf{aenc}(\langle\text{‘1’}, \mathsf{pk}(sk_A)\rangle, pk_B)), \mathsf{ASent}(\$A, sk_A, B, pk_B)]$$

$$[\mathsf{Agent}(\$B, sk_B), \mathsf{In}(\mathsf{aenc}(\langle\text{‘1’}, pk_A\rangle, \mathsf{pk}(sk_B))), \mathsf{Fr}(k)] \mapsto [\mathsf{Out}(\mathsf{aenc}(\langle\text{‘2’}, \mathsf{pk}(sk_B), k\rangle, pk_A))]$$

$$[\mathsf{In}(\mathsf{aenc}(\langle\text{‘2’}, pk_B, k\rangle, \mathsf{pk}(sk_A))), \mathsf{ASent}(\$A, sk_A, \$B, pk_B)] \mapsto []$$

2.2 Extracting a Protocol from an LTS

The following multiset rewriting rules describes a two party key agreement protocol between an Initiator and a Responder. Each rule is written in the form:

$$[\mathit{Premise}] \mapsto [\mathit{Conclusion}]$$

Recall that persistent facts, prefixed with !, are **not** removed after being consumed in a rule. Fresh facts $\mathsf{Fr}(x)$ are automatically instantiated with a unique constant. Variables that are preceded by a \$ sign are publicly known (e.g. agent names: $\$A$, $\$B$, etc.).
By inspecting the multiset rewriting rules, draw the corresponding MSC diagram.

$$[\mathsf{Fr}(sk_x)] \mapsto [\mathsf{Agent}(\$X, sk_x), !\mathsf{PublicKey}(\$X, \mathsf{pk}(sk_x))]$$

$$[\mathsf{Agent}(\$A, sk_A), !\mathsf{PublicKey}(\$B, pk_B)] \mapsto [\mathsf{Out}(\langle \mathsf{pk}(sk_A), pk_B, \text{‘}\mathit{Params}\text{’}\rangle), \mathsf{Init1}(\$A, sk_A, pk_B)]$$

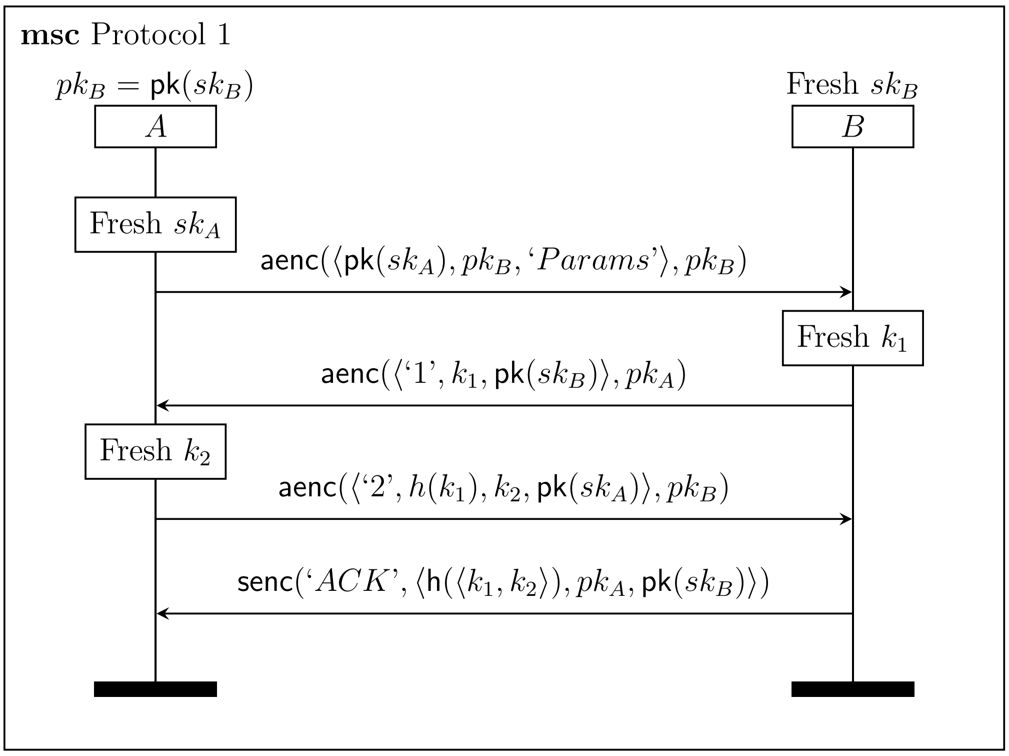
$$[\mathsf{Agent}(\$B, sk_B), \mathsf{Fr}(k_1), \mathsf{In}(\langle pk_A, \mathsf{pk}(sk_B), \text{‘}\mathit{Params}\text{’}\rangle)] \mapsto [\mathsf{Out}(\mathsf{aenc}(\langle \text{‘}1\text{’}, k_1, \mathsf{pk}(sk_B)\rangle, pk_A)), \mathsf{Resp1}(\$B, sk_B, pk_A, k_1)]$$

$$[\mathsf{Init1}(\$A, sk_A, pk_B), \mathsf{In}(\mathsf{aenc}(\langle \text{‘}1\text{’}, k_r, pk_B\rangle, \mathsf{pk}(sk_A))), \mathsf{Fr}(k_2)] \mapsto [\mathsf{Out}(\mathsf{aenc}(\langle \text{‘}2\text{’}, \mathsf{h}(k_r), k_2, \mathsf{pk}(sk_A)\rangle, pk_B)), \mathsf{Init2}(\$A, sk_A, pk_B, \langle \mathsf{h}(k_r, k_2), \mathsf{pk}(sk_A), pk_B\rangle)]$$

$$[\mathsf{Resp1}(\$B, sk_B, pk_A, k_1), \mathsf{In}(\mathsf{aenc}(\langle \text{‘}2\text{’}, \mathsf{h}(k_1), k_i, pk_A\rangle, \mathsf{pk}(sk_B)))] \mapsto [\mathsf{Out}(\mathsf{senc}(\text{‘}ACK\text{’}, \langle \mathsf{h}(\langle k_1, k_i\rangle), pk_A, \mathsf{pk}(sk_B)\rangle)), \mathsf{Resp2}(\$B, sk_B, pk_A, \mathsf{h}(\langle k_1, k_i\rangle)))]$$

$$[\mathsf{Init2}(\$A, sk_A, pk_B, k_f), \mathsf{In}(\mathsf{senc}(\text{‘}ACK\text{’}, k_f))] \mapsto [\mathsf{Init3}(\$A, pk_B, k_f)]$$

Answer:


msc Protocol 1
$pk_B = \mathsf{pk}(sk_B)$
A
Fresh sk_B
B
Fresh sk_A
$\mathsf{aenc}(\langle \mathsf{pk}(sk_A), pk_B, \text{‘}Params\text{’}\rangle, pk_B)$
Fresh k_1
$\mathsf{aenc}(\langle \text{‘}1\text{’}, k_1, \mathsf{pk}(sk_B)\rangle, pk_A)$
Fresh k_2
$\mathsf{aenc}(\langle \text{‘}2\text{’}, h(k_1), k_2, \mathsf{pk}(sk_A)\rangle, pk_B)$
$\mathsf{senc}(\text{‘}ACK\text{’}, \langle \mathsf{h}(\langle k_1, k_2\rangle), pk_A, \mathsf{pk}(sk_B)\rangle)$