TD 2: Security properties in the symbolic model

Margot Catinaud margot.catinaud@lmf.cnrs.fr

Exercise 1 (Authentication properties)

Try to find a way to express these two authentication properties in protocol between two parties A and B:

1. **Authentication:** Here at some point B want to be sure that she has indeed talked to A;
2. **Mutual authentication:** Here at some point B (resp. A) want to be sure that she talked to A (resp. B).

Hint: you can assume there is some event in the process to help you.

Exercise 2 (An interpretation of terms)

We want to define an interpretation on terms, based on random variables. These random variables will be parameterized by some security parameter $\eta \in \mathbb{N}$ and a "random" tape of bits $\rho \in \{0,1\}^*$ to draw the randomness (the interpretation of the names). Intuitively, η is an natural number that represent some characteristic number of the protocol, the length of keys is a classical example. As for the randomness, one way of doing it is to consider, for each $\eta \in \mathbb{N}$, a finite subset $\mathbb{T}_\eta$ of $\{0,1\}^*$ with all elements of the same length. Formally, we can define the interpretation domain $\mathcal{D}$ that correspond to those random variables as:

$$\mathcal{D} \stackrel{\text{def}}{=} \left\{ X : \mathbb{N} \times \{0,1\}^* \to \{0,1\}^* \cup \{\bot\} \;\middle|\; \forall \eta \in \mathbb{N},\ \forall \rho \in \{0,1\}^*,\ \rho \notin \mathbb{T}_\eta \Rightarrow X(\eta,\rho) = \bot \right\}$$

Therefore the interpretation function $[\![\cdot]\!]$ go from terms to $\mathcal{D}$ (we will write $[\![\cdot]\!]^\sigma_{\eta,\rho}$ for $[\![\cdot]\!](\sigma,\eta,\rho)$).
We also require that:

- we can split any tape ρ in two, ρ_h, ρ_a such that the interpretation of the names in $\mathcal{N}$ are sampled independently and uniformly at random of size η in ρ_h and the interpretation of any other symbols does not depend on ρ_h.
- the interpretation is compositional, meaning that you can give an interpretation of a function symbol $(f/n) \in \Sigma$ of arity n as a function from $\mathcal{D}^n$ to $\mathcal{D}$, and then get the interpretation of the terms recursively as intended: for all terms $t_1, \dots, t_n$ we have $[\![\ f(t_1,\dots,t_n)\ ]\!]^\sigma = \mathbb{M}(f)\Big([\![\ t_1\ ]\!]^\sigma, \dots, [\![\ t_n\ ]\!]^\sigma\Big)$.

We can now take a look at the interpretation of function symbols. Considering the function symbols neq and eq that we want to use to represent the non-equality and equality between two terms.

1. Find a valid interpretation for neq and eq to make them match their respective description.
2. With your interpretation, what is the interpretation of those terms:

   (a) $\mathsf{neq}(n,n)$ (b) $\mathsf{neq}(n,n')$ (c) $\mathsf{eq}(n,n)$ (d) $\mathsf{eq}(n,n')$

   where in each case, n and n' designate names in $\mathcal{N}$.
3. Does the interpretation of those terms match your expectation? Why?
4. Can you find a way to interpret neq and eq to make it work?
   Hint: Do you think an universal or existential quantification over ρ will work?
   Advice: Do not spend too much time on this question during the exercise session (5-10 minutes)

Exercise 3 (Secrecy)

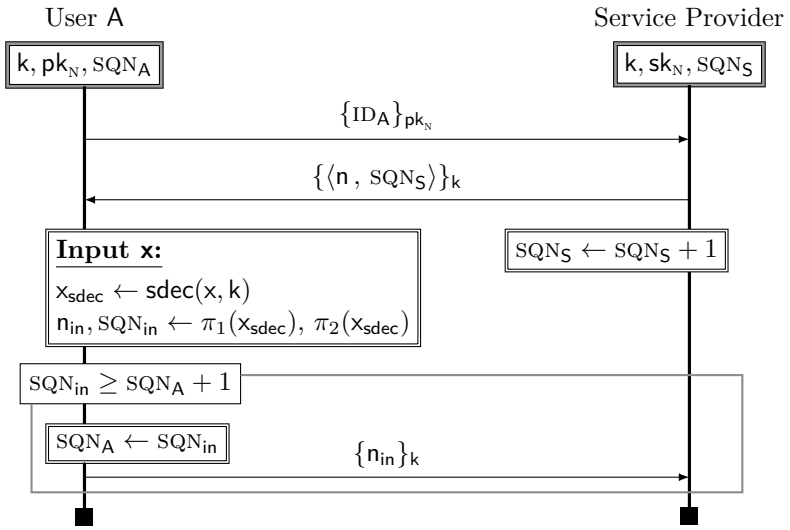
For each of the following processes, try to see if the secrecy of the secret n (or b for P_3) is ensured, otherwise exhibit a trace tr and a reasonable adversarial computation (under reasonable assumptions on your interpretation) of n.

1. $P_1^{(i)} = \mathbf{out}(c, \mathsf{senc}(n, k_i)).\mathbf{out}(c, k_i)$
2. $P_2 = \mathbf{in}(c, x).\mathbf{out}(c, \mathsf{senc}(n, x))$
3. $P_3 = \mathbf{out}(c, \mathsf{senc}(b, k)).\mathbf{in}(c, x).\mathbf{if}\ x = b\ \mathbf{then}\ \mathbf{out}(c, k)$
4. $P_4 = \mathbf{out}(c, \mathsf{senc}(n, k)).\mathbf{in}(c, x).\mathbf{if}\ x = k\ \mathbf{then}\ \mathbf{out}(c, k)$

5. $P_5 = \mathbf{in}(c, x).\mathbf{let}\ y = \mathsf{adec}(x, k)\ \mathbf{in}\ \mathbf{out}(c, k)\ \mathbf{else}\ \mathbf{out}(c, \mathsf{aenc}(n, \mathsf{pk}(k)))$
6. $P_6 = !P_5$

Exercise 4 (AKA-)

Consider the following protocol :



1. Give a set of function symbols, rewriting rules and equational theory that captures all primitives needed for this protocol.
2. Write it down formally all the agents in the process calculus in the case where there is only one user and one session by user.
3. Write it down formally all the agents in the process calculus in the case where there is multiple users and one session by user.
4. Add $\mathbf{store}(cell, t)$ and $\mathbf{get}(x, cell)$ constructs to your process calculus and propose a semantics capturing the intuitive meaning of this constructs for manipulating global states.
5. Write it down formally all the agents in the process calculus in the case where there is multiple users and multiple sessions by user.
6. Express the following security properties:
   (a) If the Service Provider S accepts then A and S agree on $\mathsf{n_{in}}$;
   (b) If the Service Provider S accepts then there is exactly one matching session of user A ;
   (c) We always have $\mathrm{SQN_S} \geqslant \mathrm{SQN_A}$;
   (d) The user A never accepts the same token $\{\langle \mathsf{n}, \mathrm{SQN_S}\rangle\}_\mathsf{k}$ twice.

Exercise 5 (Unification)

For each pair of terms, check if those are unifiable or not (find a most general unifier if possible).
Here, b and a are function symbols.

1. $\langle x, \mathsf{b}\rangle$ and $\langle \mathsf{a}, x\rangle$
2. $\langle \mathsf{b}, x\rangle$ and $\langle \mathsf{a}, y\rangle$
3. $\{x\}_\mathsf{a}$ and $\{\mathsf{b}\}_x$
4. $\langle x, y\rangle$ and $\langle\langle y, y\rangle, x\rangle$
5. $\langle \mathsf{a}, y\rangle$ and $\langle\langle y, y\rangle, \mathsf{a}\rangle$
6. z and $\langle x, y\rangle$

Exercise 6 (Bonus exercise – Strong secrecy)

Consider a protocol between two parties A and B exchanging some name s. In this exercise, we will propose a statement for the strong secrecy of s viewed as a trace property. Intuitively, the *strong secrecy of s* property means that no adversarial test, *i.e.* adversarial computation on the frame with a boolean output, can be true on s and false on a fresh new value.

Hint: You will need a helper process that will perform the test and your term algebra need to be able to represent all adversarial tests.