Spi-calcul et classes $\mathcal{H}_1$ et $\mathcal{H}_2$
Sujet de réflexion pour la première partie de l'examen

On considère des programmes écrit dans le *spi-calcul*. D'abord les expressions :

$$\begin{array}{rcll} e, \ldots & ::= & x & \text{variables} \\ & | & \{e_1\}_{e_2} & \text{chiffrements} \\ & | & \langle e_1, e_2 \rangle & \text{paires} \end{array}$$

Ces expressions dénotent des valeurs. Ensuite, les processus :

$$\begin{array}{rcll} P, Q, R, \ldots & ::= & 0 & \text{stop} \\ & | & !P & \text{réplication} \\ & | & P|Q & \text{parallèle} \\ & | & \text{case } e_1 \text{ of } \{x\}_{e_2} \Rightarrow Q & \text{déchiffrement} \\ & | & \text{case } e_1 \text{ of } \langle x, y \rangle \Rightarrow Q & \text{extraction de composantes} \\ & | & [e_1 = e_2]P & \text{test d'égalité} \\ & | & (\nu x)P & \text{création de nom frais} \\ & | & e_1\langle e_2 \rangle.P & \text{émission} \\ & | & e_1(x).P & \text{réception} \end{array}$$

Étant donné un processus P, l'ensemble $\text{sp}(P)$ des *sous-processus* de P et l'ensemble $\text{fv}(P)$ des variables libres de P sont définis comme suit :

$$\begin{array}{rclrcl} \text{sp}(0) & = & \emptyset & \text{fv}(0) & = & \emptyset \\ \text{sp}(!P) & = & \text{sp}(P) \cup \{!P\} & \text{fv}(!P) & = & \text{fv}(P) \\ \text{sp}(P|Q) & = & \text{sp}(P) \cup \text{sp}(Q) \cup \{P|Q\} & \text{fv}(P|Q) & = & \text{fv}(P) \cup \text{fv}(Q) \\ \text{sp}((\nu x)P) & = & \text{sp}(P) \cup \{(\nu x)P\} & \text{fv}((\nu x)P) & = & \text{fv}(P) \setminus \{x\} \\ \text{sp}(e_1\langle e_2\rangle.P) & = & \text{sp}(P) \cup \{e_1\langle e_2\rangle.P\} & \text{fv}(e_1\langle e_2\rangle.P) & = & \text{fv}(e_1) \cup \text{fv}(e_2) \cup \text{fv}(P) \\ \text{sp}(e_1(x).P) & = & \text{sp}(P) \cup \{e_1(x).P\} & \text{fv}(e_1(x).P) & = & \text{fv}(e_1) \cup (\text{fv}(P) \setminus \{x\}) \\ \text{sp}([e_1 = e_2]P) & = & \text{sp}(P) \cup \{[e_1 = e_2]P\} & \text{fv}([e_1 = e_2]P) & = & \text{fv}(e_1) \cup \text{fv}(e_2) \cup \text{fv}(P) \\ \multicolumn{6}{l}{\text{si } P = \text{case } e_1 \text{ of } \{x\}_{e_2} \Rightarrow Q :} \\ \text{sp}(P) & = & \text{sp}(Q) \cup \{P\} & \text{fv}(P) & = & \text{fv}(e_1) \cup \text{fv}(e_2) \cup (\text{fv}(Q) \setminus \{x\}) \\ \multicolumn{6}{l}{\text{si } P = \text{case } e_1 \text{ of } \langle x, y\rangle \Rightarrow Q :} \\ \text{sp}(P) & = & \text{sp}(Q) \cup \{P\} & \text{fv}(P) & = & \text{fv}(e_1) \cup (\text{fv}(Q) \setminus \{x, y\}) \end{array}$$

où l'ensemble $\text{fv}(e)$ des variables libres d'une expression e est défini par $\text{fv}(x) = \{x\}$, $\text{fv}(\{e_1\}_{e_2}) = \text{fv}(e_1) \cup \text{fv}(e_2)$, $\text{fv}(\langle e_1, e_2\rangle) = \text{fv}(e_1) \cup \text{fv}(e_2)$. On ne définira pas formellement l'ensemble des sous-expressions de P, qui est défini de façon analogue à l'ensemble des sous-processus.

On notera que $\mathrm{sp}(P)$ est toujours un ensemble fini de processus. Fixons un processus donné P_0 clos, c'est-à-dire tel que $\mathrm{fv}(P_0) = \emptyset$. Pour chaque processus de la forme $(\nu x)Q$ dans $\mathrm{sp}(P_0)$, ayant k variables libres $x_1, \ldots, x_k$, soit $n_{(\nu x)Q}$ un symbole de fonction d'arité k. L'idée est que le terme $n_{(\nu x)Q}(x_1, \ldots, x_k)$ dénote la valeur du nonce frais x lorsqu'on exécute $(\nu x)Q$, connaissant les valeurs $x_1, \ldots, x_k$.

On définit une sémantique simplifiée du spi-calcul comme suit. Pour tout $P \in \mathrm{sp}(P_0)$, on définit un jugement $\vdash \mathrm{acc}_P$ ("l'exécution de P_0 peut mener à devoir exécuter P") ; pour toute sous-expression e de P_0, on définit un second jugement $\vdash e = t$ ("il se peut que e vaille le terme t" — on notera qu'il n'y a qu'un nombre fini de sous-expressions e) ; finalement, on définit un jugement $t \triangleright t'$ ("sur le canal t, on peut voir passer le message t'").

Les règles permettant de dériver ces jugements sont comme suit. D'abord, P_0 peut démarrer :

$$\frac{}{\vdash P_0}\,(Start)$$

On considère ensuite chaque construction possible de processus :

$$\frac{\vdash !P}{\vdash P}\,(!) \quad \frac{\vdash P|Q}{\vdash P}\,(|_1) \quad \frac{\vdash P|Q}{\vdash Q}\,(|_2) \quad \frac{\vdash (\nu x)P}{\vdash P}\,(\nu)$$

$$\frac{\vdash (\nu x)P \quad \vdash x_1 = t_1 \ldots \vdash x_k = t_k}{\vdash x = n_{(\nu x)P}(t_1, \ldots, t_k)}\,(x = \nu)$$
$$(\text{où } \mathrm{fv}((\nu x)P) = \{x_1, \ldots, x_k\})$$

$$\frac{\vdash \text{case } e_1 \text{ of } \{x\}_{e_2} \Rightarrow Q \quad \vdash e_1 = \{t\}_{e_2[x_1 := t_1, \ldots, x_k := t_k]} \quad \vdash x_1 = t_1 \ldots \vdash x_k = t_k}{\vdash Q}\,(dec)$$

$$\frac{\vdash \text{case } e_1 \text{ of } \{x\}_{e_2} \Rightarrow Q \quad \vdash e_1 = \{t\}_{e_2[x_1 := t_1, \ldots, x_k := t_k]} \quad \vdash x_1 = t_1 \ldots \vdash x_k = t_k}{\vdash x = t}\,(x = dec)$$

où $x_1, \ldots, x_k$ sont les variables libres de e_2.

$$\frac{\vdash \text{case } e_1 \text{ of } \langle x, y\rangle \Rightarrow Q \quad \vdash e_1 = \langle t, u\rangle}{\vdash Q}\,(proj)$$

$$\frac{\vdash \text{case } e_1 \text{ of } \langle x, y\rangle \Rightarrow Q \quad \vdash e_1 = \langle t, u\rangle}{\vdash x = t}\,(x = \pi_1) \qquad \frac{\vdash \text{case } e_1 \text{ of } \langle x, y\rangle \Rightarrow Q \quad \vdash e_1 = \langle t, u\rangle}{\vdash y = u}\,(x = \pi_2)$$

$$\frac{\vdash [e_1 = e_2]P \quad \vdash e_1 = t \quad \vdash e_2 = t}{\vdash P}\,(=)$$

On a besoin ici de jugements auxiliaires exprimant les valeurs possibles d'expressions comme e_1 ou e_2 en fonction des valeurs des variables qui y sont libres :

$$\frac{\vdash e_1 = t_1 \quad \vdash e_2 = t_2}{\vdash \{e_1\}_{e_2} = \{t_1\}_{t_2}}\,(crypt) \qquad \frac{\vdash e_1 = t_1 \quad \vdash e_2 = t_2}{\vdash \langle e_1, e_2\rangle = \langle t_1, t_2\rangle}\,(pair)$$

L'émission de messages permet de définir une autre forme de jugement, $t \triangleright t'$, qui exprime que le canal de communication t peut voir passer le message t' :

$$\frac{\vdash e_1\langle e_2\rangle.P}{\vdash P}\,(send) \qquad \frac{\vdash e_1\langle e_2\rangle.P \quad \vdash e_1 = t_1 \quad \vdash e_2 = t_2}{t_1 \triangleright t_2}\,(\triangleright)$$

Ce nouveau jugement permet de définir les messages possiblement reçus par une instruction de réception :

$$\frac{\vdash e_1(x).P \quad \vdash e_1 = t_1 \quad \vdash t_1 \triangleright t}{\vdash P}\,(recv) \qquad \frac{\vdash e_1(x).P \quad \vdash e_1 = t_1 \quad \vdash t_1 \triangleright t}{\vdash x = t}\,(x = recv)$$

1. On demande de coder les jugements définis ci-dessus sous forme de clauses de Horn. On rappelle que P_0 est un processus supposé fixé, que $\mathrm{sp}(P_0)$ est fini et qu'il n'y a qu'un nombre fini de sous-expressions de P_0. On utilisera, pour chaque sous-processus $P \in \mathrm{sp}(P_0)$ de variables libres $x_1, \ldots, x_k$, un prédicat acc_P tel que $\mathrm{acc}_P(*)$ exprime $\vdash P$ (où $*$ est une constante) ; un prédicat $\mathrm{val}\langle t, t'\rangle$ exprimant que le canal t peut voir passer le message t', c'est-à-dire $t \triangleright t'$; et pour chaque sous-expression e de P_0 un prédicat $\mathrm{eq}_e(t)$ exprimant $\vdash e = t$.

   Formellement, on traduira les règles $(Start)$, $(!)$, $(|_1)$, $(|_2)$, (ν), $(x = \nu)$, (dec), $(x = dec)$, $(proj)$, $(x = \pi_1)$, $(x = \pi_2)$, $(=)$, $(crypt)$, $(pair)$, $(send)$, $(\triangleright)$, $(recv)$, $(x = recv)$, en des ensembles de clauses définies dont le plus petit modèle vérifie que :

   (i) $\mathrm{acc}_P(u)$ y est vrai si et seulement si $\vdash P$ est dérivable et $u = *$;

   (ii) $\mathrm{val}(u)$ est vrai si et seulement si u est de la forme $\langle t, t'\rangle$, et $t \triangleright t'$;

   (iii) $\mathrm{eq}_e(u)$ y est vrai si et seulement si $\vdash e = u$ est dérivable.

   On supposera pour simplifier, et ce sans perte de généralité, que toutes les variables liées dans chaque processus sont deux à deux disjointes.

   Les clauses obtenues sont elles dans la classe $\mathcal{H}_1$? On rappelle que le classe $\mathcal{H}_1$ est la classe des clauses de Horn de l'une des formes :

$$\begin{array}{rcl} \bot & \Leftarrow & P_1(t_1), \ldots, P_k(t_k) \\ P(X) & \Leftarrow & P_1(t_1), \ldots, P_k(t_k) \\ P(f(X_1, \ldots, X_n)) & \Leftarrow & P_1(t_1), \ldots, P_k(t_k) \end{array}$$

   où, dans le dernier cas, les variables $X_1, \ldots, X_n$ sont deux à deux disjointes.

2. Si on avait remplacé les règles $(crypt)$ et $(pair)$ par l'unique règle :

$$\frac{\vdash x_1 = t_1 \quad \ldots \quad \vdash x_k = t_k}{\vdash e = e[x_1 := t_1, \ldots, x_k := t_k]} \; (expr)$$

   montrer que les clauses obtenues ne seraient pas tombées dans $\mathcal{H}_1$ en général.

3. En conclure par les résultats du cours (à l'oral, ou bien d'après les résultats de l'examen 2003), que l'on peut vérifier si $\vdash P$, pour tout $P \in \mathrm{sp}(P_0)$, en temps exponentiel en la taille de P_0. De même pour vérifier toute propriété de la forme $\exists X_1, \ldots, X_k \cdot F$, où F est une conjonction de formules atomiques dépendant de X_1, ..., X_k, construites sur les prédicats acc_P, eq_e, val.

4. On considère les processus suivants, représentant le protocole de Needham-Schroeder à clés symétriques :

1.	$A \longrightarrow S : A, B, N_a$
2.	$S \longrightarrow A : \{N_a, B, K_{ab}, \{K_{ab}, A\}_{K_{bs}}\}_{K_{as}}$
3.	$A \longrightarrow B : \{K_{ab}, A\}_{K_{bs}}$
4.	$B \longrightarrow A : \{N_b\}_{K_{ab}}$
5.	$A \longrightarrow B : \{N_b, B\}_{K_{ab}}$

$$
\begin{array}{rcl}
A(c_{\rightarrow s}, c_{\leftarrow s}, c_{\rightarrow b}, c_{\leftarrow b}, a, b, k_{as}) & =_{def} & (\nu N_a) c_{\rightarrow s}\langle\langle a, \langle b, N_a\rangle\rangle\rangle . c_{\leftarrow s}(x_1). \\
 & & \quad \text{case } x_1 \text{ of } \{x_2\}_{k_{as}} \Rightarrow \\
 & & \quad \text{case } x_2 \text{ of } \langle N'_a, x_3\rangle \Rightarrow \\
 & & \quad [N_a = N'_a]\text{case } x_3 \text{ of } \langle B, x_4\rangle \Rightarrow \\
 & & \quad [b = B]\text{case } x_4 \text{ of } \langle k^a_{ab}, x_5\rangle \Rightarrow \\
 & & \quad c_{\rightarrow b}\langle x_5\rangle . c_{\leftarrow b}(x_6). \\
 & & \quad \text{case } x_6 \text{ of } \{x_7\}_{k^a_{ab}} \Rightarrow \\
 & & \quad c_{\rightarrow b}\langle\langle x_7, b\rangle\rangle .0 \\
S(c_{\leftarrow a}, c_{\rightarrow a}, k_{as}, k_{bs}) & =_{def} & c_{\leftarrow a}(y_1). \\
 & & \quad \text{case } y_1 \text{ of } \langle y_2, y_3\rangle \Rightarrow \\
 & & \quad \text{case } y_3 \text{ of } \langle y_4, y_5\rangle \Rightarrow \\
 & & \quad (\nu k^s_{ab}) c_{\rightarrow a}\langle\{\langle y_5, \langle y_4, \langle k^s_{ab}, \{\langle k^s_{ab}, y_2\rangle\}_{k_{bs}}\rangle\rangle\rangle\}_{k_{as}}\rangle .0 \\
B(c_{\leftarrow a}, c_{\rightarrow a}, k_{bs}) & =_{def} & c_{\leftarrow a}(z_1). \\
 & & \quad \text{case } z_1 \text{ of } \{z_2\}_{k_{bs}} \Rightarrow \\
 & & \quad \text{case } z_2 \text{ of } \langle k^b_{ab}, z_3\rangle \Rightarrow \\
 & & \quad (\nu N_b) c_{\rightarrow a}\langle\{N_b\}_{k^b_{ab}}\rangle . \\
 & & \quad c_{\leftarrow a}(z_4).\text{case } z_4 \text{ of } \{z_5\}_{k^b_{ab}} \Rightarrow \\
 & & \quad \text{case } z_5 \text{ of } \langle z_6, z_7\rangle \Rightarrow \\
 & & \quad [z_6 = N_b]0 \\
I(c_{pub}, a, b) & =_{def} & (c_{pub}(m).c_{pub}(k).c_{pub}\langle\{m\}_k\rangle .0) \\
 & | & (c_{pub}(m).c_{pub}(k).\text{case } m \text{ of } \{x\}_k \Rightarrow c_{pub}\langle x\rangle .0) \\
 & | & (c_{pub}(m).c_{pub}(m').c_{pub}\langle\langle m, m'\rangle\rangle .0) \\
 & | & (c_{pub}(m).\text{case } m \text{ of } \langle x, y\rangle \Rightarrow c_{pub}\langle x\rangle .0) \\
 & | & (c_{pub}(m).\text{case } m \text{ of } \langle x, y\rangle \Rightarrow c_{pub}\langle y\rangle .0) \\
 & | & c_{pub}\langle a\rangle \mid c_{pub}\langle b\rangle \\
 & & \\
Sys & =_{def} & (\nu c_{pub})(\nu a)(\nu b)(\nu k_{as})(\nu k_{bs}) \\
 & & \quad\ !A(c_{pub}, c_{pub}, c_{pub}, c_{pub}, a, b, k_{as}) \\
 & & \quad \mid\ !B(c_{pub}, c_{pub}, k_{bs}) \\
 & & \quad \mid\ !S(c_{pub}, c_{pub}, k_{as}, k_{bs}) \\
 & & \quad \mid\ !I(c_{pub}, a, b)
\end{array}
$$

L'intrus de Dolev-Yao y est représenté par le processus I, et le protocole y est représenté par le terme clos Sys.

Quelle propriété doit-on écrire pour décrire que la clé k^a_{ab} reçue par A reste secrète ? pour

k_{ab}^b ? pour k_{ab}^s ? Comment s'expriment-elles sous forme de clauses ?

5. La classe $\mathcal{H}_2^k$ ($k \geq 1$) est définie comme la classe des clauses de $\mathcal{H}_1$ vérifiant de plus les propriétés :

   (a) les variables libres dans la tête apparaissent au plus une fois dans le corps de la clause ;

   (b) les variables libres du corps ont au plus k occurrences.

   Par exemple, $P(f(X_1, X_2)) \Leftarrow Q(X_1), R(g(X_2, X_3, X_3))$ est une clause de $\mathcal{H}_2^2$. (NB : comme on va utiliser la règle de splitting sans splitting, nous étendons ici légèrement la définition des clauses de $\mathcal{H}_1$, qui seront toutes clauses de la forme $A \Leftarrow P_1(t_1), \ldots, P_m(t_m), q_1, \ldots, q_j$, où A vaut $\perp$, est de la forme $P(X)$, ou bien $P(f(X_1, \ldots, X_n))$, les variables X_i étant deux à deux distinctes, ou bien de la forme q ; q, q_1, …, q_j sont des *littéraux de splitting*, c'est-à-dire des prédicats 0-aires de la forme $\ulcorner \perp \Leftarrow B'(Y) \urcorner$ où B' est un bloc non vide $\{P'_1, \ldots, P'_m\}$, la sémantique intuitive d'un tel littéral de splitting étant d'exprimer que l'intersection des langages reconnus en P'_1, …, P'_m est non vide.)

   On rappelle que la stratégie de résolution ordonnée avec sélection, avec splitting sans splitting, pour la classe $\mathcal{H}_1$, revenait à utiliser les règles suivantes :

   – Résolution (1) :

   $$\frac{P(f(X_1, \ldots, X_n)) \Leftarrow B_1(X_1), \ldots, B_n(X_n) \quad A \Leftarrow P(f(t_1, \ldots, t_n)), P_1(t'_1), \ldots, P_m(t'_m)}{A \Leftarrow B_1(t_1), \ldots, B_n(t_n), P_1(t'_1), \ldots, P_m(t'_m)}$$

   où B_1, …, B_n, B désignent des blocs (possiblement vides), c'est-à-dire des ensembles de prédicats (par exemple, si $B_1 = \{Q, R\}$, $B_1(X_1)$ est la conjonction $Q(X_1), R(X_1)$) ; et A est $\perp$, ou un atome $Q(t)$, ou un littéral de splitting q ;

   – Résolution (2) :

   $$\frac{P(f(X_1, \ldots, X_n)) \Leftarrow B_1(X_1), \ldots, B_n(X_n) \quad A \Leftarrow P(X), B(X)}{A[X := f(X_1, \ldots, X_n)] \Leftarrow B_1(X_1), \ldots, B_n(X_n), B(f(X_1, \ldots, X_n))}$$

   où $B_1, \ldots, B_n$ désignent des blocs (possiblement vides) ; A est $\perp$, ou de la forme $Q(X)$, ou bien un littéral de splitting q

   – Résolution (3) :

   $$\frac{q \qquad A \Leftarrow P_1(t_1), \ldots, P_m(t_m), q, q_1, \ldots, q_j}{A \Leftarrow P_1(t_1), \ldots, P_m(t_m), q_1, \ldots, q_j}$$

   où A est $\perp$, ou de la forme $Q(X)$, ou bien un littéral de splitting, et $q, q_1, \ldots, q_j$ sont des littéraux de splitting ; (note : nous n'avons pas évoqué cette règle en cours, mais c'est l'un des cas qui se présentent en présence de littéraux de splitting ;)

   – Splitting sans splitting :

   $$\frac{A \Leftarrow P_1(t_1), \ldots, P_m(t_m), B(X), q_1, \ldots, q_j}{\begin{array}{c} A \Leftarrow P_1(t_1), \ldots, P_m(t_m), q, q_1, \ldots, q_j \\ q \Leftarrow B(X) \end{array}}$$

où $q = \ulcorner \bot \Leftarrow B(X) \urcorner$; A vaut $\bot$, un littéral de splitting q, ou un atome $P(t)$; $q_1, \ldots, q_j$ sont des littéraux de splitting ; B est un bloc non vide ; et X n'est pas libre ni dans A ni dans aucun des t_i, $1 \leq i \leq m$.

On supposera toujours que la règle de splitting sans splitting est appliquée en priorité. Formellement, ceci signifie que l'on suppose que les trois règles de résolution ne sont appliquées que sur des prémisses qui ne sont pas des prémisses de la règle de splitting sans splitting.

Montrer que si l'on applique ces règles à des clauses de $\mathcal{H}_2^k$, on obtient encore des clauses de $\mathcal{H}_2^k$.

6. Intuitivement, lorsque B n'est pas vide, la seule chose que l'on peut faire avec la conclusion d'une règle de résolution (2), c'est d'unifier les littéraux restant de $B(f(X_1, \ldots, X_n))$ avec d'autres clauses automates de tête $P(f(X_1, \ldots, X_n))$. On admettra donc que la procédure reste complète si l'on remplace la règle de résolution (2) par la règle dite de résolution (2') :

$$\frac{\overbrace{P_j(f(X_1, \ldots, X_n)) \Leftarrow B_{j1}(X_1), \ldots, B_{jn}(X_n)}^{1 \leq j \leq m} \quad A \Leftarrow P_1(X), \ldots, P_m(X)}{A[X := f(X_1, \ldots, X_n)] \Leftarrow B_{11}(X_1), \ldots, B_{m1}(X_1), \ldots, B_{1n}(X_n), \ldots, B_{mn}(X_n)}$$

Par la question précédente, on notera que soit A est $\bot$ ou un littéral de splitting, et chaque X_i apparaît au plus k fois (et la conclusion se splitte aussitôt), soit A est de la forme $Q(X)$, et chaque X_i apparaît au plus une fois dans le corps de la conclusion.

On définit la *taille* des termes, des atomes, et des corps de clauses par : $|X| = 1$, $|f(t_1, \ldots, t_n)| = 1 + \sum_{i=1}^{n} |t_i|$, $|P(t)| = |t|$, $|q| = 1$, et la taille d'un corps de clause est la somme des tailles de ses atomes.

Montrer que si S est un ensemble de clauses de $\mathcal{H}_2^k$, dont la taille des corps est bornée par s, et n'utilisant que des symboles de fonction d'arité au plus a, alors toutes les clauses produites ont des corps de taille au plus $\max(k(a+1), s)$.

Montrer ensuite que les termes t arguments de prédicats dans les clauses C produites par ces règles à partir de S sont soit (a) des sous-termes de termes présents dans S, soit (b) des variables. On notera N le nombre de sous-termes de termes présents dans S.

En déduire que si S ne contient pas de symbole de splitting, alors l'application exhaustive de ces règles ne produit qu'un nombre *polynomial* de clauses de $\mathcal{H}_2^k$, à k, s, et a fixés. En déduire un algorithme de décision de S en temps polynomial, à k, s, et a fixés.

7. En déduire que les problèmes de secret, d'accessibilité ($\vdash P$) et généralement tout problème de la forme $\exists X_1, \ldots, X_k \cdot F$ (cf. question 3) dans le spi-calcul se décide, en fait, en temps polynomial.