1 Spi-calculus: Syntax

Messages:

x, y, z	variables
m, n	names
$M, N, K, L ::=$	message
x	variable
n	name
$(M_1, \ldots, M_n)$	tuple
$\{M\}_K$	M encrypted with symmetric key K
$\{\!\lvert M \rvert\!\}_K$	M encrypted with asymmetric key K
$\#(M)$	hash of M
$\mathsf{Enc}(K)$	encryption key of asymmetric key K
$\mathsf{Dec}(K)$	decryption key of asymmetric key K

Processes:

$O, P, Q, R ::=$	process
$P \mid Q$	parallel composition
$!P$	replication
stop	inactivity
$\mathsf{out}\ L\ M$	output message M on channel L
$\mathsf{inp}\ L\ x;\ P$	input x from channel L (binding x in P)
$\mathsf{new}\ n;\ P$	generating name n (binding n in P)
$\mathsf{case}\ M\ \mathsf{is}\ (x_1, \ldots, x_n);\ P\ \mathsf{else}$	splitting tuple M (binding $x_1, \ldots, x_n$ in P)
$\mathsf{case}\ M\ \mathsf{is}\ \{x\}_K;\ P\ \mathsf{else}\ Q$	symmetrically decrypting M (binding x in P)
$\mathsf{case}\ M\ \mathsf{is}\ \{\!\lvert x \rvert\!\}_{K^{-1}};\ P\ \mathsf{else}\ Q$	asymmetrically decrypting M (binding x in P)
$\mathsf{if}\ M = N\ \mathsf{then}\ P\ \mathsf{else}\ Q$	conditional

Derived forms:

$\mathsf{if}\ M = N\ \mathsf{then}\ P$	$\triangleq$	$\mathsf{if}\ M = N\ \mathsf{then}\ P\ \mathsf{else}\ \mathsf{stop}$
$\mathsf{split}\ M\ \mathsf{is}\ (x_1, \ldots, x_n);\ P$	$\triangleq$	$\mathsf{case}\ M\ \mathsf{is}\ (x_1, \ldots, x_n);\ P\ \mathsf{else}\ \mathsf{stop}$
$\mathsf{decrypt}\ M\ \mathsf{is}\ \{x\}_K;\ P$	$\triangleq$	$\mathsf{case}\ M\ \mathsf{is}\ \{x\}_K;\ P\ \mathsf{else}\ \mathsf{stop}$
$\mathsf{decrypt}\ M\ \mathsf{is}\ \{\!\lvert x \rvert\!\}_{K^{-1}};\ P$	$\triangleq$	$\mathsf{case}\ M\ \mathsf{is}\ \{\!\lvert x \rvert\!\}_{K^{-1}};\ P\ \mathsf{else}\ \mathsf{stop}$

2 Spi-calculus: Operational Semantics

Structural Congruence, $P \equiv Q$:

$P \equiv P$	(Struct Refl)
$P \equiv Q \;\Rightarrow\; Q \equiv P$	(Struct Symm)
$P \equiv Q, Q \equiv R \;\Rightarrow\; P \equiv R$	(Struct Trans)
$Q \equiv R \;\Rightarrow\; P \mid Q \equiv P \mid R$	(Struct Par)
$P \mid \mathsf{stop} \equiv P$	(Struct Par Stop)
$P \mid Q \equiv Q \mid P$	(Struct Par Comm)
$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$	(Struct Par Assoc)
$!P \equiv P \mid !P$	(Struct Repl Par)
n not free in $P \;\Rightarrow\; P \mid \mathsf{new}\ n; Q \equiv \mathsf{new}\ n; (P \mid Q)$	(Struct New)

Process reductions, $P \to Q$:

$P' \equiv P \to Q \equiv Q' \;\Rightarrow\; P' \to Q'$	(Redn Equiv)
$P \to P' \;\Rightarrow\; P \mid Q \to P' \mid Q$	(Redn Par)
$P \to P' \;\Rightarrow\; \mathsf{new}\ n; P \to \mathsf{new}\ n; P'$	(Redn New)
$\mathsf{out}\ L\ M \mid \mathsf{inp}\ L\ x; P \to \{M/x\}P$	(Redn IO)
$\mathsf{if}\ M = M\ \mathsf{then}\ P\ \mathsf{else}\ Q \to P$	(Redn Cond 1)
$\mathsf{if}\ M = N\ \mathsf{then}\ P\ \mathsf{else}\ Q \to Q$ if $M \neq N$	(Redn Cond 2)
$\mathsf{case}\ (M_1, \ldots, M_n)\ \mathsf{is}\ (x_1, \ldots, x_n); P\ \mathsf{else}\ Q \to \{M_1, \ldots, M_n/x_1, \ldots, x_n\}P$	(Redn Split 1)
$\mathsf{case}\ M\ \mathsf{is}\ (x_1, \ldots, x_n); P\ \mathsf{else}\ Q \to Q$ if M is not an n-tuple	(Redn Split 2)
$\mathsf{case}\ \{M\}_K\ \mathsf{is}\ \{x\}_K; P\ \mathsf{else}\ Q \to \{M/x\}P$	(Redn SDecrypt 1)
$\mathsf{case}\ N\ \mathsf{is}\ \{x\}_K; P\ \mathsf{else}\ Q \to Q$ if N is not of the form $\{M\}_K$	(Redn SDecrypt 2)
$\mathsf{case}\ \{\!|M|\!\}_{\mathsf{Enc}(K)}\ \mathsf{is}\ \{\!|x|\!\}_{\mathsf{Dec}(K)^{-1}}; P\ \mathsf{else}\ Q \to \{M/x\}P$	(Redn ADecrypt 1)
$\mathsf{case}\ N\ \mathsf{is}\ \{\!|x|\!\}_{L^{-1}}; P\ \mathsf{else}\ Q \to Q$ if (N, L) is not of the form $(\{\!|M|\!\}_{\mathsf{Enc}(K)}, \mathsf{Dec}(K))$	(Redn ADecrypt 2)