ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

S. Mödersheim
Ch. Sprenger
AS 2009

Formal Methods for Information Security

Exercise Sheet 7

Hand-in date: Nov 16, 2009

Assignment 7.1: Stepwise refinement

Let us consider the "simulation square":

$$\begin{array}{ccc} s & \xrightarrow{R_a \cup Id} & s' \\ \downarrow{\scriptstyle R} & & \downarrow{\scriptstyle R} \\ t & \xrightarrow{R_c} & t' \end{array}$$

abstract system $\mathcal{S}_a = (S_a, I_a, R_a)$

concrete system $\mathcal{S}_c = (S_c, I_c, R_c)$

Here, R is the *simulation relation* and Id is the identity relation corresponding to $skip$.

(a) Show that condition (2) on slide 8 of Module 8 that is graphically represented by the simulation square above is equivalent to

$$R; R_c \subseteq (R_a \cup Id); R$$

where the semicolon (;) denotes relational composition defined by

$$R; R' = \{(x, y) \mid \exists z.\ (x, z) \in R \wedge (z, y) \in R'\}.$$

(b) Show that the refinement relation $\sqsubseteq$ is reflexive and transitive.

Hint: This is easiest to see using the graphical representation for intuition together with the formulation in (a) for a very concise formal proof.

(c) Consider abstract and concrete systems specified by (unparametrized) guarded commands of the following form:

$$\begin{array}{ll} name \equiv & \\ \quad G(v) & \text{guard: enabling predicate} \\ \quad v := f(v) & \text{action: assignment} \end{array}$$

Such a guarded command defines a transition relation on the set of states S as follows:

$$R = \{(s, s') \in S \times S \mid s \in [\![G]\!] \wedge s' = [\![f]\!](s)\}$$

where $[\![G]\!] \subseteq S$ is the set of states (interpretations of the variables v) satisfying $G(v)$, and $[\![f]\!] : S \to S$ is the function on states induced by the expression $f(v)$.

Instantiate the simulation condition (2) from slide 8 of Module 8 with relations R_a and R_c of this particular form (i.e., with guards G_a and G_c and action functions f_a and f_c) and simplify the result in order to obtain (semantic versions of) guard strengthening and action refinement.

Solution

(a) The original condition (2) from the lecture slides reads as follows:

$$\forall s, t, t'.\ (s,t) \in R \wedge (t,t') \in R_c \Rightarrow \exists s'.(s,s') \in R_a \cup Id \wedge (s',t') \in R$$

This is logically equivalent to

$$\forall s, t'.\ (\exists t.\ (s,t) \in R \wedge (t,t') \in R_c) \Rightarrow \exists s'.\ (s,s') \in R_a \cup Id \wedge (s',t') \in R$$

since t does not appear in the conclusion of the implication. Using relational composition, this can be expressed as

$$\forall s, t'.\ (s,t') \in R; R_c \Rightarrow (s,t') \in (R_a \cup Id); R$$

which is equivalent to the required

$$R; R_c \subseteq (R_a \cup Id); R. \tag{1}$$

(b) **Reflexivity:** Consider a transition system $\mathcal{S} = (S, I, R)$. We show that Id is a simulation relation on S. Condition (1) concerning initial states is clearly satisfied, since $(s,s) \in Id$ for all $s \in I$. Concerning condition (2) on transitions, we have to show that

$$Id; R \subseteq (R \cup Id); Id$$

which holds trivially.

Transitivity: We consider three transition systems $\mathcal{S}_i = (S_i, I_i, R_i)$ (for $i \in \{1,2,3\}$) and we assume that $\mathcal{S}_3 \sqsubseteq \mathcal{S}_2$ and $\mathcal{S}_2 \sqsubseteq \mathcal{S}_1$. We have to show that $\mathcal{S}_3 \sqsubseteq \mathcal{S}_1$.

By the definition of simulation there are simulation relations R_{23} and R_{12} showing the refinements $\mathcal{S}_3 \sqsubseteq \mathcal{S}_2$ and $\mathcal{S}_2 \sqsubseteq \mathcal{S}_1$, respectively. We now show that their composition $R_{12}; R_{23}$ is a simulation relation establishing $\mathcal{S}_3 \sqsubseteq \mathcal{S}_1$.

First, we need to check that for each initial state $s_3 \in I_3$ there is an initial state $s_1 \in I_1$ such that $(s_1, s_3) \in R_{12}; R_{23}$. Since R_{12} is a simulation relation relating S_1 and S_2, there is a $s_2 \in I_2$ such that $(s_1, s_2) \in R_{12}$. By an analogous argument there is a $s_3 \in I_3$ such that $(s_2, s_3) \in R_{23}$. Hence, $(s_1, s_3) \in R_{12}; R_{23}$.

For the transition part, we have to show that

$$(R_{12}; R_{23}); R_3 \subseteq (R_1 \cup Id); (R_{12}; R_{23}) \tag{2}$$

We use the abbreviation $R^{=} = R \cup Id$ for the reflexive closure of a relation R. The simulation condition (1) above can now be rewritten as

$$R; R_c \subseteq {R_a}^{=}; R. \tag{3}$$

This together with the facts that $R; Id \subseteq {R_a}^{=}; R$ and that relational composition distributes over union implies that

$$R; {R_c}^{=} \subseteq {R_a}^{=}; R. \tag{4}$$

Now relation (2) can be seen as follows:

$$\begin{array}{lcll} (R_{12}; R_{23}); R_3 & \subseteq & R_{12}; (R_{23}; R_3) & \text{by associativity of ;} \\ & \subseteq & R_{12}; ({R_2}^{=}; R_{23}) & \text{by (3) and monotonicity of ;} \\ & \subseteq & (R_{12}; {R_2}^{=}); R_{23} & \text{by associativity of ;} \\ & \subseteq & ({R_1}^{=}; R_{12}); R_{23} & \text{by (4) and monotonicty of ;} \\ & \subseteq & {R_1}^{=}; (R_{12}; R_{23}) & \text{by associativity of ;} \end{array}$$

(c) Condition (2) for simulation from the lecture slides is implied by:

$$\forall s, t, t'.\ (s, t) \in R \wedge (t, t') \in R_c \Rightarrow \exists s'.\ (s, s') \in R_a \wedge (s', t') \in R$$

Here we have replaced $R_a \cup Id$ by R_a, since we want to simulate R_c by R_a and not skip (Id). We now instantiate R_a and R_c, taking into account that each of them is composed of a guard and an action function. At the same time we drop the skip

$$\begin{array}{l} \forall s, t, t'.\ (s, t) \in R \wedge t \in [\![G_c]\!] \wedge t' = [\![f_c]\!](t) \\ \Rightarrow \exists s'.\ s \in [\![G_a]\!] \wedge s' = [\![f_a]\!](s) \wedge (s', t') \in R \end{array}$$

Since there is just one possible s' and t' we can remove the respective quantifiers and simplify this to the following condition:

$$\forall s, t.\ (s, t) \in R \wedge t \in [\![G_c]\!] \Rightarrow s \in [\![G_a]\!] \wedge ([\![f_a]\!](s), [\![f_c]\!](t)) \in R$$

We can split this into two conditions:

$$\begin{array}{l} \forall s, t.\ (s, t) \in R \wedge t \in [\![G_c]\!] \Rightarrow s \in [\![G_a]\!] \\ \forall s, t.\ (s, t) \in R \wedge t \in [\![G_c]\!] \Rightarrow ([\![f_a]\!](s), [\![f_c]\!](t)) \in R \end{array}$$

These two conditions correspond to guard strengthening and action refinement for the case of parameter-less deterministic guarded actions considered here.

Note that in general we also need invariants to prove such proof obligations. We have ignored them here for simplicity.

Assignment 7.2: Security Protocol Refinement

Let us consider the third refinement step constructing the Otway-Rees/AN protocol from the abstract channel version. The simulation relation used in this refinement is:

$$kt2.cthds = kt3.cthds \wedge kt2.sthds = kt3.sthds \wedge$$
$$chan = absMsg(parts(IK)) \wedge ikk(chan) = keys(IK)$$

Recall that $parts(H)$ closes the set H of messages under all submessages (i.e., adds both projections of pairs and the cleartext of ciphertexts in H).

(a) Give the full definition of the message abstraction function $absMsg$ (one case appears on slide 36 of Module 8).

(b) Show that step 4 of the protocol refines its abstract counterpart. The guard strengthening and (incomplete) action refinement conditions are listed on slide 40 of Module 8. Which properties of $parts$ do you need to prove these conditions?

(c) Show that the concrete protocol (3rd refinement) is executable. Start from the initial state, where there are no threads and the intruder knowledge is $IK_0 = \mathit{Agents} \cup \{\mathsf{sk}(i, s)\}$. For each step in the sequence, check that its guard is true in the current state and write down the state resulting from the actions.

(d) Simulate this protocol execution on the level of the abstract protocol (2nd refinement). Check that the simulation relation holds after each step.

(e) We want to understand why the intruder of the second refinement needs to send a set of messages instead of just a single one. Consider the abstract state s_3 with intruder knowledge IK_3 and the concrete state t_3 with channel messages $chan_3$ obtained after the first three execution steps in (c) and (d) above.

Write down the additions to the intruder knowledge IK after each step of the following concrete execution sequence starting in state t_3:

$$\begin{array}{ll} DY\_fake(\langle\langle c, d, nb\rangle, \{\!|nb, a, b, kab|\!\}_{\mathsf{sk}(b,s)}\rangle) & IK_4 = IK_3 \cup \{\ldots\} \\ kt3\_step4(a, b, nb, kab, \langle c, d, nb\rangle) & IK_5 = IK_4 \cup \{\ldots\} \\ kt3\_step2(c, d, nb, nd) & IK_6 = IK_5 \cup \{\ldots\} \end{array}$$

We must be able to simulate this sequence by a corresponding sequence in the second refinement starting in state s_3:

$$\begin{array}{ll} fake(M) & chan_4 = chan_3 \cup M \\ kt2\_step4(a, b, nb, kab) & chan_5 = chan_4 \cup \{\ldots\} \\ kt2\_step2(c, d, nb, nd) & chan_6 = chan_5 \cup \{\ldots\} \end{array}$$

Write down the additions to the channel variable $chan$ after each step of the sequence. Discover the content of M during the simulation by inspecting the guards of the guarded actions following $fake(M)$: Which messages need to be in M in order to ensure the executability of this sequence?

Solution

(a) The three rules defining the function $absMsg$ for this protocol are:

$$\frac{\langle A, B, N_A\rangle \in H}{\mathsf{Insec}(\mathsf{msg1}(A, B, N_A)) \in absMsg(H)} \text{ (msg1)}$$

$$\frac{\langle A, B, N_A, N_B\rangle \in H}{\mathsf{Insec}(\mathsf{msg2}(A, B, N_A, N_B)) \in absMsg(H)} \text{ (msg2)}$$

$$\frac{\{\!|N, A, B, K|\!\}_{\mathsf{sk}(C)} \in H}{\mathsf{Secure}(C, \mathsf{msg34}(N, A, B, K)) \in absMsg(H)} \text{ (msg34)}$$

(b) **Guard strengthening:**

H1. $chan = absMsg(parts(IK))$
H2. $\langle X, \{\!|N_B, A, B, K_{AB}|\!\}_{\mathsf{sk}(B)}\rangle \in IK$
$\vdash$
$\mathsf{Secure}(B, \mathsf{msg34}(N_B, A, B, K_{AB})) \in chan$

We can rewrite H1 in the conclusion and then apply rule (msg34). This yields:

H1. $chan = absMsg(parts(IK))$
H2. $\langle X, \{\!|N_B, A, B, K_{AB}|\!\}_{\mathsf{sk}(B)}\rangle \in IK$
$\vdash$
$\{\!|N_B, A, B, K_{AB}|\!\}_{\mathsf{sk}(B)} \in parts(IK)$

This obviously holds by the definition of $parts$.

Action refinement: We consider the part concerning $chan = absMsg(parts(IK))$:

H1. $chan = absMsg(parts(IK))$
H2. $\langle X, \{\!|N_B, A, B, K_{AB}|\!\}_{\mathsf{sk}(B)}\rangle \in IK$
$\vdash chan = absMsg(parts(\{X\} \cup IK))$

(Note that due to the different communication structure at the two abstraction levels, no message is added to the abstract variable $chan$, whereas X is sent to IK in the concrete case.)

In order to prove this sequent, we first note that from H2 it follows that $X \in parts(IK)$. But then we have $parts(\{X\} \cup IK) = parts(IK)$, since X is already a part of IK. Hence, we can rewrite this equation in the conclusion of the sequent, which now holds by assumption H1.

(c) Here is the specification of all five protocol steps of the third refinement:

$kt3\_step1(A, B, N_A) \equiv$
$N_A \notin dom(cthds)$
$IK := \{\, \langle A, B, N_A\rangle \,\} \cup IK$
$cthds := cthds[N_A \mapsto ((A, B), \mathsf{Init}(\bot))]$

$$
\begin{array}{l}
kt3\_step2(A, B, N_A, N_B) \equiv \\
\quad N_B \notin dom(cthds) \\
\quad \langle A, B, N_A \rangle \in IK \\
\quad IK := \{\, \langle A, B, N_A, N_B \rangle \,\} \cup IK \\
\quad cthds := cthds[N_B \mapsto ((A, B), \mathsf{Resp}(\bot))]
\end{array}
$$

$$
\begin{array}{l}
kt3\_step3(A, B, N_A, N_B, K_{AB}) \equiv \\
\quad K_{AB} \notin dom(sthds) \\
\quad \langle A, B, N_A, N_B \rangle \in IK \\
\quad IK := \{\, \langle \{\!|N_A, A, B, K_{AB}|\!\}_{\mathsf{sk}(A,s)}, \{\!|N_B, A, B, K_{AB}|\!\}_{\mathsf{sk}(B,s)} \rangle \,\} \cup IK \\
\quad sthds := sthds[K_{AB} \mapsto ((A, B), N_A, N_B)]
\end{array}
$$

$$
\begin{array}{l}
kt3\_step4(A, B, N_B, K_{AB}, X) \equiv \\
\quad N_B \mapsto ((A, B), \mathsf{Resp}(\bot)) \\
\quad \langle X, \{\!|N_B, A, B, K_{AB}|\!\}_{\mathsf{sk}(B,s)} \rangle \in IK \\
\quad IK := \{\, X \,\} \cup IK \\
\quad cthds := cthds[N_B \mapsto ((A, B), \mathsf{Resp}(K_{AB}))]
\end{array}
$$

$$
\begin{array}{l}
kt3\_step5(A, B, N_A, K_{AB}) \equiv \\
\quad N_A \mapsto ((A, B), \mathsf{Init}(\bot)) \\
\quad \{\!|N_A, A, B, K_{AB}|\!\}_{\mathsf{sk}(A,s)} \in IK \\
\quad cthds := cthds[N_A \mapsto ((A, B), \mathsf{Init}(K_{AB}))]
\end{array}
$$

We start the protocol execution from the initial state

$$t_0 = \langle cthds = \emptyset, sthds = \emptyset, IK = IK_0 \rangle$$

where $IK_0 = Agents \cup \{\mathsf{sk}(i, s)\}$. We say that a guarded action is *enabled* in a given state if its guard holds true in that state.

Executing the instance $kt3\_step1(a, b, na)$ creates a new initiator thread and leads us to the state

$$
\begin{array}{rl}
t_1 = & \langle cthds = [na \mapsto ((a, b), \mathsf{Init}(\bot))], \\
& \quad sthds = \emptyset, \\
& \quad IK = IK_0 \cup \{\langle a, b, na \rangle\} \rangle
\end{array}
$$

The only guard here requires that na is a fresh nonce.

Next, the execution of $kt3\_step2(a, b, na, nb)$ is clearly enabled for a fresh nb and its execution leads us to the state

$$
\begin{array}{rl}
t_2 = & \langle cthds = [na \mapsto ((a, b), \mathsf{Init}(\bot)), nb \mapsto ((a, b), \mathsf{Resp}(\bot))], \\
& \quad sthds = \emptyset, \\
& \quad IK = IK_0 \cup \{\langle a, b, na \rangle, \langle a, b, na, nb \rangle\} \rangle
\end{array}
$$

The execution of $kt3\_step3(a, b, na, nb, kab)$ for a fresh kab is clearly enabled and leads us to the state

$$\begin{aligned} t_3 = \; & \langle cthds = [na \mapsto ((a,b), \mathsf{Init}(\bot)), nb \mapsto ((a,b), \mathsf{Resp}(\bot))], \\ & sthds = [kab \mapsto ((a,b), na, nb)], \\ & IK = IK_0 \cup \\ & \quad \{\langle a, b, na\rangle, \langle a, b, na, nb\rangle, \\ & \quad\; \langle \{\!|na, a, b, kab|\!\}_{\mathsf{sk}(a,s)}, \{\!|nb, a, b, kab|\!\}_{\mathsf{sk}(b,s)}\rangle\} \; \rangle \end{aligned}$$

Next, $kt3\_step4\,(a, b, na, kab, \{\!|na, a, b, kab|\!\}_{\mathsf{sk}(a,s)})$ is clearly enabled and its execution leads to the state

$$\begin{aligned} t_4 = \; & \langle cthds = [na \mapsto ((a,b), \mathsf{Init}(\bot)), nb \mapsto ((a,b), \mathsf{Resp}(kab))], \\ & sthds = [kab \mapsto ((a,b), na, nb)], \\ & IK = IK_0 \cup \\ & \quad \{\langle a, b, na\rangle, \langle a, b, na, nb\rangle, \\ & \quad\; \langle \{\!|na, a, b, kab|\!\}_{\mathsf{sk}(a,s)}, \{\!|nb, a, b, kab|\!\}_{\mathsf{sk}(b,s)}\rangle, \\ & \quad\; \{\!|na, a, b, kab|\!\}_{\mathsf{sk}(a,s)}\} \; \rangle \end{aligned}$$

Finally, $kt3\_step5(a, b, na, kab)$ is enabled and its execution leads to the final state

$$\begin{aligned} t_5 = \; & \langle cthds = [na \mapsto ((a,b), \mathsf{Init}(kab)), nb \mapsto ((a,b), \mathsf{Resp}(kab))], \\ & sthds = [kab \mapsto ((a,b), na, nb)], \\ & IK = IK_0 \cup \\ & \quad \{\langle a, b, na\rangle, \langle a, b, na, nb\rangle, \\ & \quad\; \langle \{\!|na, a, b, kab|\!\}_{\mathsf{sk}(a,s)}, \{\!|nb, a, b, kab|\!\}_{\mathsf{sk}(b,s)}\rangle, \\ & \quad\; \{\!|na, a, b, kab|\!\}_{\mathsf{sk}(a,s)}\} \; \rangle \end{aligned}$$

(d) Here is the specification of all five protocol steps of the second refinement:

$$\begin{aligned} & kt2\_step1\,(A, B, N_A) \equiv \\ & \quad N_A \notin dom(cthds) \\ & \quad chan := \{\, \mathsf{Insec}(\mathsf{msg1}(A, B, N_A))\,\} \cup chan \\ & \quad cthds := cthds[N_A \mapsto ((A, B), \mathsf{Init}(\bot))] \end{aligned}$$

$$\begin{aligned} & kt2\_step2(A, B, N_A, N_B) \equiv \\ & \quad N_B \notin dom(cthds) \\ & \quad \mathsf{Insec}(\mathsf{msg1}(A, B, N_A)) \in chan \\ & \quad chan := \{\, \mathsf{Insec}(\mathsf{msg2}(A, B, N_A, N_B))\,\} \cup chan \\ & \quad cthds := cthds[N_B \mapsto ((A, B), \mathsf{Resp}(\bot))] \end{aligned}$$

$$\begin{aligned} & kt2\_step3(A, B, N_A, N_B, K_{AB}) \equiv \\ & \quad K_{AB} \notin dom(sthds) \\ & \quad \mathsf{Insec}(\mathsf{msg2}(A, B, N_A, N_B)) \in IK \\ & \quad chan := \{\, \mathsf{Secure}(A, \mathsf{msg34}(N_A, A, B, K_{AB})), \\ & \qquad\quad \mathsf{Secure}(B, \mathsf{msg34}(N_B, A, B, K_{AB}))\,\} \cup chan \\ & \quad sthds := sthds[K_{AB} \mapsto ((A, B), N_A, N_B)] \end{aligned}$$

$$
\begin{array}{l}
kt2\_step4(A, B, N_B, K_{AB}) \equiv \\
\quad N_B \mapsto ((A, B), \mathsf{Resp}(\bot)) \\
\quad \mathsf{Secure}(B, \mathsf{msg34}(N_B, A, B, K_{AB})) \in chan \\
\quad cthds := cthds[N_B \mapsto ((A, B), \mathsf{Resp}(K_{AB}))]
\end{array}
$$

$$
\begin{array}{l}
kt2\_step5(A, B, N_A, K_{AB}) \equiv \\
\quad N_A \mapsto ((A, B), \mathsf{Init}(\bot)) \\
\quad \mathsf{Secure}(A, \mathsf{msg34}(N_A, A, B, K_{AB})) \in chan \\
\quad cthds := cthds[N_A \mapsto ((A, B), \mathsf{Init}(K_{AB}))]
\end{array}
$$

Now we simulate the sequence from (c) by the following sequence of guarded actions:

$$
\begin{array}{l}
kt2\_step1(a, b, na) \\
kt2\_step2(a, b, na, nb) \\
kt2\_step3(a, b, na, nb, kab) \\
kt2\_step4(a, b, nb, kab) \\
kt2\_step5(a, b, na, kab)
\end{array}
$$

Here is the the initial state and the resulting sequence of five states:

$$
\begin{array}{rl}
s_0 = & \langle cthds = \emptyset, sthds = \emptyset, chan = \emptyset \rangle \\
\\
s_1 = & \langle cthds = [na \mapsto ((a, b), \mathsf{Init}(\bot))], \\
& \quad sthds = \emptyset, \\
& \quad chan = \{\mathsf{Insec}(\mathsf{msg1}(a, n, na))\} \rangle \\
\\
s_2 = & \langle cthds = [na \mapsto ((a, b), \mathsf{Init}(\bot)), nb \mapsto ((a, b), \mathsf{Resp}(\bot))], \\
& \quad sthds = \emptyset, \\
& \quad chan = \{\mathsf{Insec}(\mathsf{msg1}(a, n, na)), \mathsf{Insec}(\mathsf{msg2}(a, b, na, nb))\} \rangle \\
\\
s_3 = & \langle cthds = [na \mapsto ((a, b), \mathsf{Init}(\bot)), nb \mapsto ((a, b), \mathsf{Resp}(\bot))], \\
& \quad sthds = [kab \mapsto ((a, b), na, nb)], \\
& \quad chan = \{\mathsf{Insec}(\mathsf{msg1}(a, n, na)), \mathsf{Insec}(\mathsf{msg2}(a, b, na, nb)), \\
& \qquad \mathsf{Secure}(a, \mathsf{msg34}(na, a, b, kab)), \\
& \qquad \mathsf{Secure}(b, \mathsf{msg34}(nb, a, b, kab))\} \rangle \\
\\
s_4 = & \langle cthds = [na \mapsto ((a, b), \mathsf{Init}(\bot)), nb \mapsto ((a, b), \mathsf{Resp}(kab))], \\
& \quad sthds = [kab \mapsto ((a, b), na, nb)], \\
& \quad chan = \{\mathsf{Insec}(\mathsf{msg1}(a, n, na)), \mathsf{Insec}(\mathsf{msg2}(a, b, na, nb)), \\
& \qquad \mathsf{Secure}(a, \mathsf{msg34}(na, a, b, kab)), \\
& \qquad \mathsf{Secure}(b, \mathsf{msg34}(nb, a, b, kab))\} \rangle \\
\\
s_5 = & \langle cthds = [na \mapsto ((a, b), \mathsf{Init}(kab)), nb \mapsto ((a, b), \mathsf{Resp}(kab))], \\
& \quad sthds = [kab \mapsto ((a, b), na, nb)], \\
& \quad chan = \{\mathsf{Insec}(\mathsf{msg1}(a, n, na)), \mathsf{Insec}(\mathsf{msg2}(a, b, na, nb)), \\
& \qquad \mathsf{Secure}(a, \mathsf{msg34}(na, a, b, kab)), \\
& \qquad \mathsf{Secure}(b, \mathsf{msg34}(nb, a, b, kab))\} \rangle
\end{array}
$$

We can easily check that all pairs $(s_i, t_i) \in R$, where R is the simulation relation defined by

$$kt2.cthds = kt3.cthds \wedge kt2.sthds = kt3.sthds \wedge$$
$$chan = absMsg(parts(IK)) \wedge ikk(chan) = keys(IK)$$

In particular, note that

- $(s_0, t_0) \in R$, since $absMsg(parts(IK_0)) = \emptyset$ and $keys(IK_0) = \emptyset$, since the function $keys$ ignores the long-term keys.
- $(s_3, t_3) \in R$, since $absMsg(parts(IK))$ decomposes the pair of ciphertexts into two secure abstract messages.
- $(s_4, t_4) \in R$, since the addition of the first component of the pair of ciphertexts in IK at t_4 does not increase $absMsg(parts(IK))$ with respect to t_3.

(e) Let us first look at the concrete execution sequence. Let IK_3 denote the intruder knowledge in state t_3. The intruder can clearly fake the message in the parameter of DY_fake, resulting in the following evolution of the intruder knowledge during the execution of the three steps:

$DY\_fake(\langle\langle c, d, nb\rangle,$ $\{\!\lvert nb, a, b, kab\rvert\!\}_{\mathsf{sk}(b,s)}\rangle)$	$IK_4 = IK_3 \cup \{\langle\langle c, d, nb\rangle,$ $\{\!\lvert nb, a, b, kab\rvert\!\}_{\mathsf{sk}(b,s)}\rangle\}$
$kt3\_step4\,(a, b, nb, kab, \langle c, d, nb\rangle)$	$IK_5 = IK_4 \cup \{\langle c, d, nb\rangle\}$
$kt3\_step2(c, d, nb, nd)$	$IK_6 = IK_5 \cup \{\langle c, d, nb, nd\rangle\}$

Since $Agents \subseteq IK_3$, $nb \in analz(IK_3)$, and $\{\!\lvert nb, a, b, kab\rvert\!\}_{\mathsf{sk}(b,s)} \in analz(IK_3)$, the intruder can clearly fake the message $\langle\langle c, d, nb\rangle, \{\!\lvert nb, a, b, kab\rvert\!\}_{\mathsf{sk}(b,s)}\rangle$. In the second step, b receives this message and forwards $\langle c, d, nb\rangle$. The latter message is received in the last step, which creates a new thread with ID nd and sends the message $\langle c, d, nb, nd\rangle$.

Now we simulate this sequence in the abstract system. Let us call $chan_3$ the state of the channel variable in state s_3. By executing the three steps in the sequence, the content of this variable evolves as follows:

$fake(M)$	$chan_4 = chan_3 \cup M$
$kt2\_step4\,(a, b, nb, kab)$	$chan_5 = chan_4$
$kt2\_step2(c, d, nb, nd)$	$chan_6 = chan_5 \cup \{\mathsf{Insec}(\mathsf{msg2}(c, d, nb, nd))\}$

The guard of $kt2\_step4\,(a, b, nb, kab)$ requires that

$$\mathsf{Secure}(b, \mathsf{msg34}(nb, a, b, kab)) \in chan_4$$

but no new message is sent in this action. The guard of $kt2\_step2(c, d, nb, nd)$ requires

$$\mathsf{Insec}(\mathsf{msg1}(c, d, nb)) \in chan_4$$

Since neither of these two messages is in $chan_3$, the set M must consist of these two messages. Otherwise, this abstract sequence would not be executable. This example shows that the argument of $fake$ must be a set of messages; a single message is not enough.