Special topics in Logic and Security I

Master Year II, Sem. I, 2025-2026

Ioana Leuştean
FMI, UB

References

1 Cremers, C. J. F. (2006). Scyther : semantics and verification of security protocols Eindhoven: Technische Universiteit Eindhoven DOI: 10.6100/IR614943

2 Cremers C. and Mauw S. Operational Semantics and Verification of Security Protocols. Springer, 2012.

Operational semantics

The operational semantics of a security protocol P is a labelled transition system

$$(State, RunEvent, \rightarrow, st_0(P))$$

- $State = \mathcal{P}(RunTerm) \times \mathcal{P}(Run)$ where $Run = Inst \times RoleEvent^*$
- $st_0(P) = \langle\langle AKN_0(P), \emptyset \rangle\rangle$ where $AKN_0(P)$ is the initial adversary knowledge
- $RunEvent = Inst \times (RoleEvent \cup \{create(R) \mid R \in Role\}$
- $runsof(P, R) = \{(inst, s) \mid$ there exists kn such that $P(R) = (kn, s)$
  $inst = (\theta, \rho, \sigma)$ with $dom(\rho) = roles(s)\}$
- The *transition system* has four rules, one for each of the events:

  create, *send*, *recv*, *claim*

Rules for $(State, RunEvent, \to, st_0(P))$

$$[create_P]\ \frac{R \in dom(P) \quad ((\theta,\rho,\emptyset), s) \in runsof(P,R) \quad \theta \notin runsIDs(F)}{\langle\!\langle AKN, F \rangle\!\rangle \xrightarrow{((\theta,\rho,\emptyset),create(R))} \langle\!\langle AKN, F \cup \{((\theta,\rho,\emptyset),s)\} \rangle\!\rangle}$$

$$[send]\ \frac{e = send_l(R_1,R_2,m) \quad (inst,[e]\cdot s) \in F}{\langle\!\langle AKN, F \rangle\!\rangle \xrightarrow{(inst,e)} \langle\!\langle AKN \cup \{inst(m)\}, F \setminus \{(inst,[e]\cdot s)\} \cup \{(inst,s)\} \rangle\!\rangle}$$

$$[recv]\ \frac{e = recv_l(R_1,R_2,pt) \quad AKN \vdash m \quad (inst,[e]\cdot s) \in F \quad Match(inst,pt,m,inst')}{\langle\!\langle AKN, F \rangle\!\rangle \xrightarrow{(inst',e)} \langle\!\langle AKN, F \setminus \{(inst,[e]\cdot s)\} \cup \{(inst',s)\} \rangle\!\rangle}$$

$$[claim]\ \frac{e = claim_l(R,c,t) \quad (inst,[e]\cdot s) \in F}{\langle\!\langle AKN, F \rangle\!\rangle \xrightarrow{(inst,e)} \langle\!\langle AKN, F \setminus \{(inst,[e]\cdot s)\} \cup \{(inst,s)\} \rangle\!\rangle}$$

Operational semantics: traces

$$(State, RunEvent, \rightarrow, st_0(P))$$

- Execution:
  $[st_0, \alpha_1, st_1, \alpha_2, \ldots, \alpha_n, st_n]$ where $\alpha_i \in RunEvent$ și $st_i = \langle\langle AKN_i, F_i \rangle\rangle$

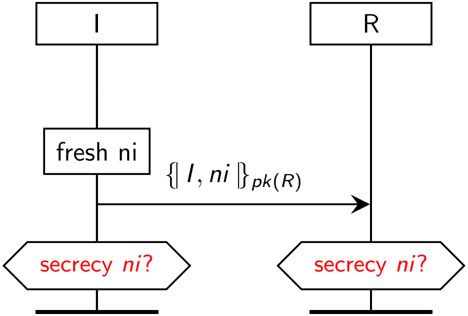
- Knowing the initial state we define the execution using traces $[\alpha_1, \alpha_2, \ldots, \alpha_n]$.

  Given a protocol P, we define $traces(P)$ as the set of the finite traces of the labelled transition system $(State, RunEvent, \rightarrow, st_0(P))$ associated to P.

Security properties

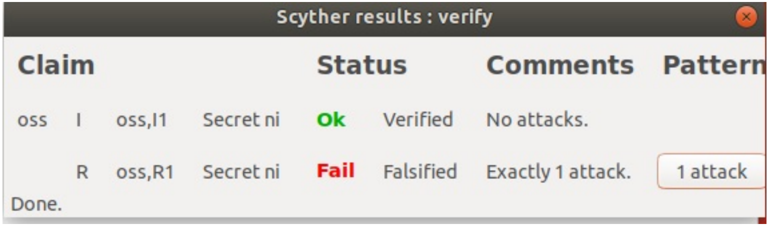
Scyther: OSS Protocol

```
protocol oss(I,R)
{role I
{fresh ni: Nonce;
send_1(I,R, {I,ni}pk(R) );
claim(I,Secret,ni);}

role R
{var ni: Nonce;
recv_1(I,R, {I,ni}pk(R) );
claim(R,Secret,ni);
}}
```



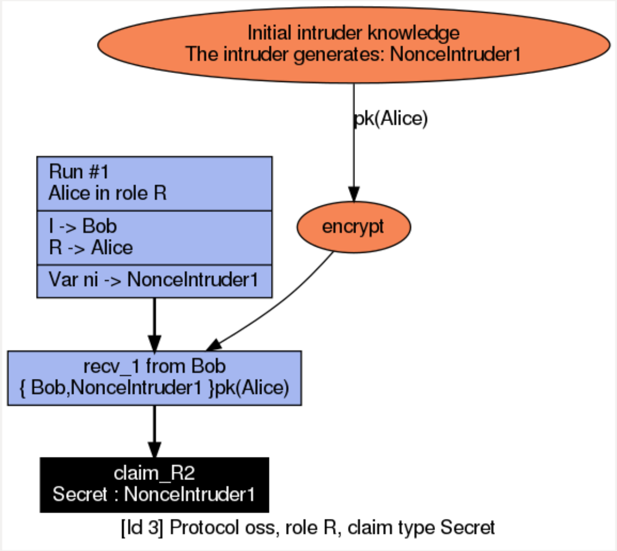
Scyther: OSS Protocol

```
protocol oss(I,R)
{role I
{fresh ni: Nonce;
send_1(I,R, {I,ni}pk(R) );
claim(I,Secret,ni);}

role R
{var ni: Nonce;
recv_1(I,R, {I,ni}pk(R));
claim(R,Secret,ni);
}}
```



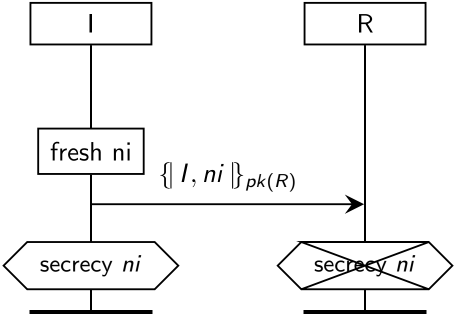
```
Claim                              Status               Comments          Pattern
oss   I   oss,I1   Secret ni       Ok     Verified      No attacks.
      R   oss,R1   Secret ni       Fail   Falsified     Exactly 1 attack.  1 attack
Done.
```

Scyther: OSS Protocol



[Id 3] Protocol oss, role R, claim type Secret

Scyther: OSS Protocol



The claims are locally analyzed!

Security properties

In our formalism, security properties are defined by (local) *claim* events. This means that an agent has a local view based on the messages he receives and the protocol should offer guarantee that the agent can be sure about certain properties.

Recall that the set of agents is partitioned in honest and corrupted agents $\mathit{Agent} = \mathit{Agent}_H \cup \mathit{Agent}_C$. For an instantiation $(\theta, \rho, \sigma) \in \mathit{Inst}$ we define the predicate $\mathit{honest}(\theta, \rho, \sigma)$ which is true if the roles are instantiated with honest agents, i.e. $\mathit{honest}(\theta, \rho, \sigma)$ iff $\mathit{range}(\rho) \subseteq \mathit{Agent}_H$

The security properties are defined as properties over traces.

Security properties

In the sequel we shall analyze *secrecy*, which means that certain information is not revealed to an adversary.

Formally, for a protocol P and a role R, the secrecy claim

$$\gamma = \mathit{claim}_l(R, \mathit{secret}, \mathit{rt})$$

is correct if

$$\begin{array}{c}\text{for any } t \in \mathit{traces}(P) \text{ and any } ((\theta,\rho,\sigma),\gamma) \in t \\ \mathit{honest}((\theta,\rho,\sigma)) \text{ implies } \mathit{AKN}(t) \nvdash (\theta,\rho,\sigma)(\mathit{rt}),\end{array}$$

where $\mathit{AKN}([\alpha_1,\ldots,\alpha_n]) = \mathit{AKN}_n$.

The OSS protocol: secrecy

$$OSS(i) = (\{i, r, ni, pk(r)\}, [send_1(i, r, \{\!| i, ni |\!\}_{pk(r)}), claim_2(i, secret, ni)])$$

$$OSS(r) = (\{i, r, sk(r)\}, [recv_1(i, r, \{\!| i, W |\!\}_{pk(r)}), claim_3(r, secret, W)])$$

Assume $\theta \in RID$ and $\rho = \{i \mapsto A, r \mapsto B\}$ such that $A, B \in Agent_H$.

Consider the following trace:

$$\begin{aligned} t = \ & [((\theta, \rho, \emptyset), create(r)), \\ & ((\theta, \rho, \{W \mapsto ne\}), recv_1(i, r, \{\!| i, W |\!\}_{pk(r)})) \\ & ((\theta, \rho, \{W \mapsto ne\}), claim_3(r, secret, W))] \end{aligned}$$

where $ne \in AdversaryFresh \subseteq AKN_0(P)$.

The OSS protocol: secrecy

$$OSS(i) = \begin{array}{l}(\{i, r, ni, pk(r)\},\\ [send_1(i, r, \{\!|\, i, ni \,|\!\}_{pk(r)}),\\ \ claim_2(i, secret, ni)])\end{array} \qquad OSS(r) = \begin{array}{l}(\{i, r, sk(r)\},\\ [recv_1(i, r, \{\!|\, i, W \,|\!\}_{pk(r)}),\\ \ claim_3(r, secret, W)])\end{array}$$

Assume $\theta \in RID$ and $\rho = \{i \mapsto A, r \mapsto B\}$ such that $A, B \in Agent_H$.

Consider the following trace:

$$t = \begin{array}{l}[((\theta, \rho, \emptyset), create(r)),\\ ((\theta, \rho, \{W \mapsto ne\}), recv_1(i, r, \{\!|\, i, W \,|\!\}_{pk(r)}))\\ ((\theta, \rho, \{W \mapsto ne\}), claim_3(r, secret, W))]\end{array}$$

where $ne \in AdversaryFresh \subseteq AKN_0(P) \subseteq AKN(t)$.

If $\gamma = claim_3(r, secret, W)$ then $((\theta, \rho, \{W \mapsto ne\}), \gamma) \in t$ and $honest(\theta, \rho, \{W \mapsto ne\})$ but $AKN(t) \vdash ne = (\theta, \rho, \{W \mapsto ne\})(W)$

Consequently, the *secrecy* claim of the **responder** r does not hold.

However, the *secrecy* claim of the **initiator** role holds!

The OSS protocol: secrecy

$$OSS(i) = \begin{array}{l}(\{i, r, ni, pk(r)\},\\ [send_1(i, r, \{\!| i, ni |\!\}_{pk(r)}),\\ \; claim_2(i, secret, ni)])\end{array} \qquad OSS(r) = \begin{array}{l}(\{i, r, sk(r)\},\\ [recv_1(i, r, \{\!| i, W |\!\}_{pk(r)}),\\ \; claim_3(r, secret, W)])\end{array}$$

We sketch a proof that $\zeta = claim_2(i, secret, ni)$ holds.

Assume that $t = [\alpha_1, \ldots, \alpha_n] \in traces(OSS)$ and $inst = (\theta, \rho, \sigma) \in Inst$ such that $(inst, \zeta) \in t$ and $honest(inst)$.

(*) We assume that $AKN(t) \vdash inst(ni)$.

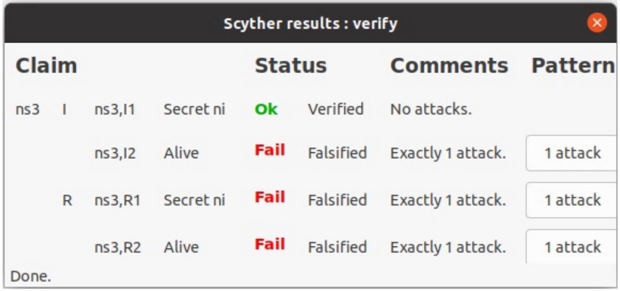
Hence there is the least $k < n$ such that $AKN_k \not\vdash inst(ni)$ and $AKN_{k+1} \vdash inst(ni)$. But [*send*] is the only deduction rule that enriches the adversay knowledge, so

$$AKN_{k+1} = AKN_k \cup \{inst(\{\!| i, ni |\!\}_{pk(r)})\} = AKN_k \cup \{\{\!| \rho(i), ni^{\#\theta} |\!\}_{pk(\rho(r))}\}.$$

It follows that, $AKN_k \cup \{\{\!| \rho(i), ni^{\#\theta} |\!\}_{pk(\rho(r))}\} \vdash ni^{\#\theta}$. According to the deduction system on terms this is possible only if $sk(\rho(r))$ belongs to the adversary knowledge, but this is impossible since all agents are honest. Consequently, the assumption (*) is false and we proved by contradiction that the *secrecy* claim of the **initiator** role holds.

Scyther: OSS Protocol

```
protocol oss(I,R)                 role R
{role I                           {var ni: Nonce;
{fresh ni: Nonce;                 recv_1(I,R, {I,ni}pk(R) );
send_1(I,R, {I,ni}pk(R) );        claim(R,Secret,ni);
claim(I,Secret,ni);               claim(R,Alive,I);
claim(I,Alive,R);}                }}
```


Scyther results : verify

Claim				Status		Comments	Pattern
ns3	I	ns3,I1	Secret ni	Ok	Verified	No attacks.	
		ns3,I2	Alive	Fail	Falsified	Exactly 1 attack.	1 attack
	R	ns3,R1	Secret ni	Fail	Falsified	Exactly 1 attack.	1 attack
		ns3,R2	Alive	Fail	Falsified	Exactly 1 attack.	1 attack

Done.

Security properties: Secrecy and Aliveness

- The *secrecy* claim $\gamma = \mathit{claim}_l(R, \mathit{secret}, \mathit{rt})$ holds if the information rt is not revealed to an adversary. Formally, for a protocol P and a role R, the secrecy claim $\gamma = \mathit{claim}_l(R, \mathit{secret}, \mathit{rt})$ is correct if

$$\text{for any } t \in \mathit{traces}(P) \text{ and any } ((\theta, \rho, \sigma), \gamma) \in t$$
$$\mathit{honest}((\theta, \rho, \sigma)) \text{ implies } \mathit{AKN}(t) \nvdash (\theta, \rho, \sigma)(\mathit{rt}),$$

  where $\mathit{AKN}([\alpha_1, \ldots, \alpha_n]) = \mathit{AKN}_n$.

- In the sequel we shall analyze *aliveness*, which is a form of *authentication*. Our goal is to establish that a certain agent is "alive": the *aliveness* claim $\gamma = \mathit{claim}_l(R, \mathit{alive}, R')$ is correct if, whenever the role R' is executed by an honest agent, this agent performed an event (action).

Authentication: aliveness

If R is a role, then the events belonging to R are

$$\begin{aligned} \mathit{RoleEvent}_R \quad ::= \quad & \mathit{send}_{\mathit{Label}}(R, \mathit{Role}, \mathit{RoleTerm}) \\ & \mid \mathit{recv}_{\mathit{Label}}(\mathit{Role}, R, \mathit{RoleTerm}) \\ & \mid \mathit{claim}_{\mathit{Label}}(R, \mathit{Claim}[, \mathit{RoleTerm}]) \end{aligned}$$

$\mathit{RoleEvent} = \bigcup\{\mathit{RoleEvent}_R \mid R \in \mathit{Role}\}$
$\mathit{RunEvent} = \mathit{Inst} \times (\mathit{RoleEvent} \cup \{\mathit{create}(R) \mid R \in \mathit{Role}\}$

We define
$\mathit{role} : \mathit{RoleEvent} \to \mathit{Role}$, $\mathit{role} =$ the role the event belongs to
$\mathit{actor} : \mathit{Inst} \times \mathit{RoleEvent} \to \mathit{Agent}$, $\mathit{actor}((\theta, \rho, \sigma), \mathit{re}) = \rho(\mathit{role}(\mathit{re}))$

The claim $\gamma = \mathit{claim}_l(R, \mathit{alive}, R')$ is correct if, whenever R' is executed by an honest agent, this agent performed an event (action). Formally, γ is correct if

$$\text{for any } t \in \mathit{traces}(P) \text{ and any } ((\theta, \rho, \sigma), \gamma) \in t$$
$$\mathit{honest}((\theta, \rho, \sigma)) \text{ implies there exists } \mathit{ev} \in t \text{ such that } \mathit{actor}(\mathit{ev}) = \rho(R').$$

The OSS protocol: aliveness

$$OSS(i) = \begin{array}{l}(\{i, r, ni, pk(r)\},\\ [send_1(i, r, \{\!|\, i, ni \,|\!\}_{pk(r)}),\\ \ claim_2(i, alive, r)])\end{array} \qquad OSS(r) = \begin{array}{l}(\{i, r, sk(r)\},\\ [recv_1(i, r, \{\!|\, i, W \,|\!\}_{pk(r)}),\\ \ claim_3(r, alive, i)])\end{array}$$

Assume $\theta \in RID$ and $\rho = \{i \mapsto A, r \mapsto B\}$ such that $A, B \in Agent_H$.

Consider the following trace:

$$t = \begin{array}{l}[((\theta, \rho, \emptyset), create(r)),\\ ((\theta, \rho, \{W \mapsto ne\}), recv_1(i, r, \{\!|\, i, W \,|\!\}_{pk(r)}))\\ ((\theta, \rho, \{W \mapsto ne\}), claim_3(r, alive, i))]\end{array}$$

where $ne \in AdversaryFresh \subseteq AKN_0(P) \subseteq AKN(t)$.

If $\gamma = claim_3(r, alive, i)$
then $((\theta, \rho, \{W \mapsto ne\}), \gamma) \in t$ and $honest(\theta, \rho, \{W \mapsto ne\})$
but there is no $ev \in t$ such that $actor(ev) = \rho(i) = A$ since
$actor(create(r)) = actor(recv_1(i, r, \{\!|\, i, W \,|\!\}_{pk(r)})) = \rho(r) = B$.

Consequently, the *aliveness* claim of the **responder** r does not hold.

The OSS protocol: aliveness

$$OSS(i) = \begin{array}{l}(\{i, r, ni, pk(r)\},\\ [send_1(i, r, \{\!|\, i, ni \,|\!\}_{pk(r)}),\\ claim_2(i, alive, r)])\end{array} \qquad OSS(r) = \begin{array}{l}(\{i, r, sk(r)\},\\ [recv_1(i, r, \{\!|\, i, W \,|\!\}_{pk(r)}),\\ claim_3(r, alive, i)])\end{array}$$

Assume $\theta \in RID$ and $\rho = \{i \mapsto A, r \mapsto B\}$ such that $A, B \in Agent_H$.

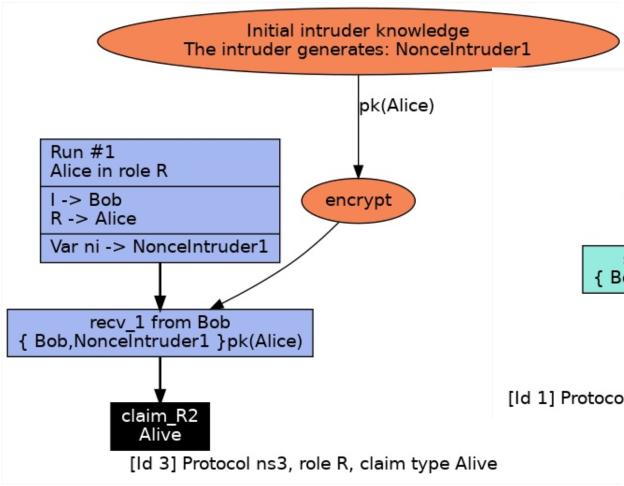
Consider the following trace:

$$t = \begin{array}{l}[((\theta, \rho, \emptyset), create(i)),\\ ((\theta, \rho, \emptyset), send_1(i, r, \{\!|\, i, ni \,|\!\}_{pk(r)}))\\ ((\theta, \rho, \emptyset), claim_2(i, alive, r))]\end{array}$$

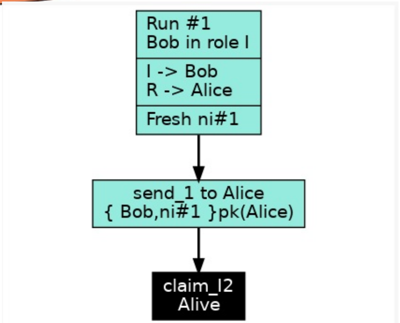
If $\gamma = claim_2(i, alive, r)$
then $(\theta, \rho, \emptyset), \gamma) \in t$ and $honest((\theta, \rho, \emptyset))$
but there is no $ev \in t$ such that $actor(ev) = \rho(r) = B$ since
$actor(create(i)) = actor(send_1(i, r, \{\!|\, i, ni \,|\!\}_{pk(r)})) = \rho(i) = A$.

Consequently, the *aliveness* claim of the **initiator** i does not hold.

Scyther: OSS Protocol



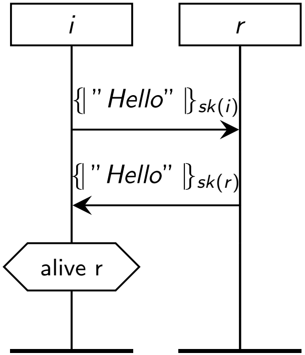
[Id 3] Protocol ns3, role R, claim type Alive



[Id 1] Protocol ns3, role I, claim type Alive

Authentication: aliveness

We consider the following protocol:



i
r
$\{\!|\text{"Hello"}|\!\}_{sk(i)}$
$\{\!|\text{"Hello"}|\!\}_{sk(r)}$
alive r

```
usertype Message;

protocol hello(I,R)
{role I
{const m: Message;
send_1(I,R, {m}sk(I) );
recv_2(R,I, {m}sk(R) );
claim(I,Alive,R);}

role R
{var M: Message;
recv_1(I,R, {M}sk(I));
send_2(R,I, {M}sk(R));
claim(R,Alive,I);
}}
```

Note the use of agents secret keys for encryption!

Authentication: aliveness

$i \longrightarrow r : \{\!|\, \text{"}Hello\text{"}\, |\!\}_{sk(i)}$
$r \longrightarrow i : \{\!|\, \text{"}Hello\text{"}\, |\!\}_{sk(r)}$

We sketch the proof that $\gamma = claim_l(i, alive, r)$ holds. We have to prove that for any $t \in traces(P)$ and any $((\theta, \rho, \sigma), \gamma) \in t$, $honest((\theta, \rho, \sigma))$ implies there exists $ev \in t$ such that $actor(ev) = \rho(r)$

Assume that t is a trace and $((\theta, \rho, \sigma), \gamma)$ is a label such that $A = \rho(i)$ and $B = \rho(r)$ are honest agents.

Since A and B played their roles correctly, there should be a label $((\theta, \rho, \sigma), recev_2(r, i, \{\!|\, \text{"}Hello\text{"}\, |\!\}_{sk(r)})$, which means that A received a message encrypted with the secret key of B.

Since A and B are honest, the adversary doesn't know the secret key of B, we infer that B actually sent the message $\{\!|\, \text{"}Hello\text{"}\, |\!\}_{sk(B)}$, so there should be a label $ev = ((\theta', \rho', \sigma'), sent_2(R_1, R_2, \{\!|\, \text{"}Hello\text{"}\, |\!\}_{sk(R_1)}) \in t$ such that $\rho'(R_1) = B$.

Consequently, $actor(ev) = \rho'(R_1) = B = \rho(r)$, and the claim is proved.

Authentication: aliveness

$i \longrightarrow r : \{\!|\,\text{"}\mathit{Hello}\text{"}\,|\!\}_{sk(i)}$
$r \longrightarrow i : \{\!|\,\text{"}\mathit{Hello}\text{"}\,|\!\}_{sk(r)}$

In the above proof for $\gamma = claim_l(i, alive, r)$ we found an event (action) $ev = ((\theta', \rho', \sigma'), sent_l(R_1, R_2, \{\!|\,\text{"}\mathit{Hello}\text{"}\,|\!\}_{sk(R_1)}) \in t$ such that $actor(ev) = \rho(r)$.

Note that:

- we don't know what roles played R_1 and R_2, so we don't know if R_1 played the correct role (the receiver),
- we don't know *when* the event took place, i.e., there is no relation between the runs θ and θ'.

The above property *alive* is a very weak form of authentication, so it will be called *weak-alive* and stronger versions of *aliveness* will be defined.

Authentication: Weak Aliveness

$role : RoleEvent \rightarrow Role$, $role$ = the role the event belongs to
$actor : Inst \times RoleEvent \rightarrow Agent$, $actor((\theta, \rho, \sigma), re) = \rho(role(re))$

- **Weak Aliveness**:
  the claim $\gamma = claim_l(R, \textit{weak-alive}, R')$ is correct if

  $$\text{for any } t \in traces(P) \text{ and any } ((\theta, \rho, \sigma), \gamma) \in t$$
  $$honest((\theta, \rho, \sigma)) \text{ implies there exists } ev \in t \text{ such that } actor(ev) = \rho(R').$$

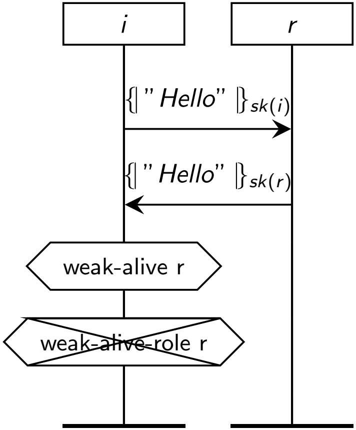
- **Weak Aliveness in the Correct Role**:
  the claim $\gamma = claim_l(R, \textit{weak-alive-role}, R')$ is correct if

  $$\text{for any } t \in traces(P) \text{ and any } ((\theta, \rho, \sigma), \gamma) \in t$$
  $$honest((\theta, \rho, \sigma)) \text{ implies there exists } ev \in t \text{ such that}$$
  $$actor(ev) = \rho(R') \text{ and } role(ev) = R'.$$

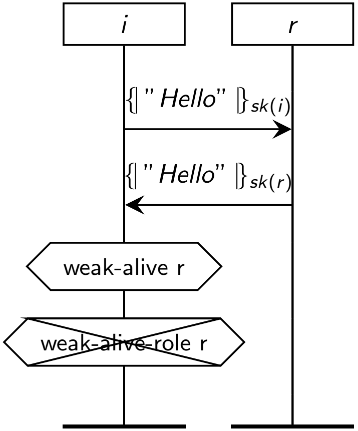
  In the above definition, if $ev = (inst, \epsilon)$, then $role(ev) := role(\epsilon)$.

Note that the **Alive** claim in Scyther verifies **Weak Aliveness**.

Authentication: Weak Aliveness


i
r
$\{\!|\text{"Hello"}|\!\}_{sk(i)}$
$\{\!|\text{"Hello"}|\!\}_{sk(r)}$
weak-alive r
~~weak-alive-role r~~

Authentication: Weak Aliveness



i
r
$\{\!|\, \text{"}Hello\text{"} \,|\!\}_{sk(i)}$
$\{\!|\, \text{"}Hello\text{"} \,|\!\}_{sk(r)}$
weak-alive r
~~weak-alive-role r~~

An attack on the *weak-alive-role r* claim:

$$A \longrightarrow B : \{\!|\, \text{"}Hello\text{"} \,|\!\}_{sk(A)}$$
$$B \longrightarrow E : \{\!|\, \text{"}Hello\text{"} \,|\!\}_{sk(B)}$$
$$E \longrightarrow A : \{\!|\, \text{"}Hello\text{"} \,|\!\}_{sk(B)}$$

A thinks that B (correctly) played the receiver role, but B played the initiator role (in a seession with E).

Authentication: Recent Aliveness

Let $<_t$ be the order of the run events (labels) in the trace t and assume that *runidof* (_) denotes the run identifier of its argument.

- **Recent Aliveness**:
  the claim $\gamma = \mathit{claim}_l(R, \mathit{recent\text{-}alive}, R')$ is correct if
  $$\text{for any } t \in \mathit{traces}(P) \text{ and any } (\mathit{inst}, \gamma) \in t$$
  $$\mathit{honest}(\mathit{inst}) \text{ implies there exists } \mathit{ev}, \mathit{ev}' \in t \text{ such that}$$
  $$\mathit{actor}(\mathit{ev}) = \rho(R'),\ \mathit{runidof}(\mathit{ev}') = \mathit{runidof}(\mathit{inst}) \text{ and } \mathit{ev}' <_t \mathit{ev} <_t (\mathit{inst}, \gamma).$$

  (the actor playing R' must have executed an event during the run of the claim event)

- **Recent Aliveness in the Correct Role**:
  the claim $\gamma = \mathit{claim}_l(R, \mathit{recent\text{-}alive\text{-}role}, R')$ is correct if
  $$\text{for any } t \in \mathit{traces}(P) \text{ and any } ((\theta, \rho, \sigma), \gamma) \in t$$
  $$\mathit{honest}((\theta, \rho, \sigma)) \text{ implies there exists } \mathit{ev}, \mathit{ev}' \in t \text{ such that}$$
  $$\mathit{actor}(\mathit{ev}) = \rho(R'),\ \mathit{role}(\mathit{ev}) = R',$$
  $$\mathit{runidof}(\mathit{ev}') = \mathit{runidof}(\mathit{inst}) \text{ and } \mathit{ev}' <_t \mathit{ev} <_t (\mathit{inst}, \gamma).$$

Exercises: seminar

Authentication: Synchronisation

Note that *aliveness* is a form of authentication that only requires that the communication partner executes some event, without any requirements on the messages he sends or receives.

Consequently, the Needham-Schroeder protocol (NSPK) satisfies the aliveness properties. However, we know that the protocol is not "secure", since the "man in the middle" attack is possible:

For analyzing this vulnerability a stronger form of *authentication is defined*: *synchornisation*. Informally, synchronisation states that the actual exchange of messages appears in the trace exactly as required in the description of the protocol.

In order to formally define synchronisation, we introduce the *event order* and the *cast function*.

RoleEvent Order

For a protocol P we define:

- the *RoleEvent Order* is the total order $\epsilon_1 <_R \cdots <_R \epsilon_n$ where R is a role with $P(R) = (kn, [\epsilon_1, \ldots, \epsilon_n])$.
- the *Communication Relation* $\dashrightarrow \subseteq \mathit{RoleEvent} \times \mathit{RoleEvent}$ is defined by

  $\epsilon_1 \dashrightarrow \epsilon_2$ if and only if
  there exist $\mathbf{l} \in \mathit{Label}$, $R, R' \in \mathit{Role}$, $rt_1, rt_2 \in \mathit{RoleTerm}$
  such that $\epsilon_1 = \mathit{send}_{\mathbf{l}}(R, R', rt_1)$ and $\epsilon_2 = \mathit{recev}_{\mathbf{l}}(R', R, rt_2)$
- the *Protocol Order* is

$$\prec_P = \left(\dashrightarrow \cup \bigcup_{R \in \mathit{Role}} <_R \right)^+$$

Protocol Order $\prec_P \subseteq \mathit{RoleEvent} \times \mathit{RoleEvent}$

For the NSPK protocol

$$
\begin{array}{ll@{\qquad}ll}
NS(i) = & (\{i, r, ni, sk(i), pk(i), pk(r)\}, & NS(r) = & (\{i, r, nr, sk(r), pk(r), pk(i)\}, \\
& [send_1(i, r, \{\!|\, ni, i \,|\!\}_{pk(r)}), & & [recv_1(i, r, \{\!|\, W, i \,|\!\}_{pk(r)}), \\
& recv_2(r, i, \{\!|\, ni, V \,|\!\}_{pk(i)}), & & send_2(r, i, \{\!|\, W, nr \,|\!\}_{pk(i)}), \\
& send_3(i, r, \{\!|\, V \,|\!\}_{pk(r)}), & & recv_3(i, r, \{\!|\, nr \,|\!\}_{pk(r)}), \\
& claim_4(i, synch)]) & & claim_5(r, synch)])
\end{array}
$$

the event order is:

$$
\begin{array}{ccc}
send_1(i, r, \{\!|\, ni, i \,|\!\}_{pk(r)}) & \prec_{NS} & recv_1(i, r, \{\!|\, W, i \,|\!\}_{pk(r)}) \\
\curlywedge_{NS} & & \curlywedge_{NS} \\
recv_2(r, i, \{\!|\, ni, V \,|\!\}_{pk(i)}) & \succ_{NS} & send_2(r, i, \{\!|\, W, nr \,|\!\}_{pk(i)}) \\
\curlywedge_{NS} & & \curlywedge_{NS} \\
send_3(i, r, \{\!|\, V \,|\!\}_{pk(r)}) & \prec_{NS} & recv_3(i, r, \{\!|\, nr \,|\!\}_{pk(r)}) \\
\curlywedge_{NS} & & \curlywedge_{NS} \\
claim_4(i, synch) & & claim_5(r, synch)
\end{array}
$$

Security properties: cast function

"As a protocol description can consist of a number of roles, which can be instantiated any number of times, we need some way to express which run is (supposedly) communicating with which other runs.

Therefore, we introduce the notion of a cast, borrowing intuition from a theatre play that is performed several times.

The cast for a particular performance of the play relates activity in the performance to particular roles. Likewise, the concrete activities in a protocol instance at the trace level, are assigned to the roles in the protocol. In the theater case, in different performances an actor can play different roles. Also, the same role can be taken up, for different performances of the play, by different actors. Likewise, run events associated with different roles may belong to the same agent and run events that are instances for the same role may belong to different agents."
[1, 3.3.2]

Security properties: cast function

For a protocol P a trace $t \in \mathit{traces}(P)$ a partial function

$$\Gamma : \mathit{RunEvent} \times \mathit{Role} \rightharpoonup \mathit{RID}$$

is a *cast* function if for all $\mathit{cev} = (\mathit{inst}, c) \in \mathit{RunEvent}$ the following property holds:

$$\Gamma(\mathit{cev}, R) = \theta \text{ if and only if}$$

- if $R = \mathit{role}(\mathit{cev})$ then $\theta = \mathit{runidof}(\mathit{cev})$
(the run identifier of the analyzed event is executing the corresponding role), **and**
- for any $\mathit{ev} \in t$, $\mathit{runidof}(\mathit{ev}) = \theta$ implies $\mathit{role}(\mathit{ev}) = R$
(the run identifier θ associated to a certain role is actually executing that role).

The cast function provides the correspondence between roles and run identifiers (in the context of a given claim). The cast function Γ depends on P and t.

Example: trace for the Needham-Schroeder protocol

Let t be the following trace:

$$\begin{array}{l}
[((1,\rho,\emptyset), create(i))\ , \\
((1,\rho,\emptyset), send_1(i,r,\{\!|\ ni, i\ |\!\}_{pk(r)})), \\
((2,\rho,\emptyset), create(r)), \\
((2,\rho,\{W \mapsto ni^{\#1}\}), recv_1(i,r,\{\!|\ W, i\ |\!\}_{pk(r)})), \\
((2,\rho,\{W \mapsto ni^{\#1}\}), send_2(r,i,\{\!|\ W, nr\ |\!\}_{pk(i)})), \\
((1,\rho,\{V \mapsto nr^{\#2}\}), recv_2(r,i,\{\!|\ ni, V\ |\!\}_{pk(i)})), \\
((1,\rho,\{V \mapsto nr^{\#2}\}), send_3(i,r,\{\!|\ V\ |\!\}_{pk(r)})), \\
((1,\rho,\{V \mapsto nr^{\#2}\}), claim_4(i, synch)), \\
((2,\rho,\{W \mapsto ni^{\#1}\}), recv_3(i,r,\{\!|\ nr\ |\!\}_{pk(r)})), \\
((2,\rho,\{W \mapsto ni^{\#1}\}), claim_5(r, synch))]
\end{array}$$

For $cev = ((1,\rho,\{V \mapsto nr^{\#2}\}), claim_4(i, synch))$, a cast function is

$$\Gamma(cev, i) = 1 \text{ and } \Gamma(cev, r) = 2$$

.

Authentication: Synchronization

For a protocol P the claim $\gamma = \mathit{claim}_I(R, \mathit{synch})$ is correct if

for any $t \in \mathit{traces}(P)$ there exists a cast function Γ such that

for any $(\mathit{inst}, \gamma) \in t$, if $\mathit{honest}(\mathit{inst})$ then

for any role events ϵ_1, $\epsilon_2 \in \mathit{RoleEvent}$, $\epsilon_1 \dashrightarrow \epsilon_2$ and $\epsilon_2 \prec_P \gamma$

the corresponding run events

- **occur in the right order in the trace**,
- **are executed in the runs defined by the the cast function** Γ,
- **have the same contents**.

Note that $\epsilon_1 \dashrightarrow \epsilon_2$ means that ϵ_1 and ϵ_2 are corresponding *send* and *receive* events, while $\epsilon_2 \prec_P \gamma$ means that all communications occur before the claim event in the protocol order.

Authentication: Synchronization

For a protocol P the claim $\gamma = \mathit{claim}_l(R, \mathit{synch})$ is correct if

for any $t \in \mathit{traces}(P)$ there exists a cast function Γ such that
for any $(\mathit{inst}, \gamma) \in t$, if $\mathit{honest}(\mathit{inst})$ then
for any ϵ_1, $\epsilon_2 \in \mathit{RoleEvent}$, $\epsilon_1 \dashrightarrow \epsilon_2$ and $\epsilon_2 \prec_P \gamma$ imply
there exists $\mathit{inst}_1, \mathit{inst}_2$ such that

- occur in the right order in the trace
  $(\mathit{inst}_1, \epsilon_1) <_t (\mathit{inst}_2, \epsilon_2) <_t (\mathit{inst}, \gamma)$
- are executed in the runs defined by the the cast function Γ
  $\mathit{runidof}(\mathit{inst}_1) = \Gamma((\mathit{inst}, \gamma), \mathit{role}(\epsilon_1))$ and
  $\mathit{runidof}(\mathit{inst}_2) = \Gamma((\mathit{inst}, \gamma), \mathit{role}(\epsilon_2))$
- have the same contents
  $\mathit{cont}((\mathit{inst}_1, \epsilon_1)) = \mathit{cont}((\mathit{inst}_2, \epsilon_2))$,

  where *cont* is the content extraction function for *send* and *receive* events:
  $\mathit{cont}(\mathit{inst}, \mathit{send}_l(R, R', m)) = \mathit{inst}(R, R', m)$ and
  $\mathit{cont}(\mathit{inst}, \mathit{recv}_l(R', R, m)) = \mathit{inst}(R', R, m)$

Security properties: Synchronization

For a protocol P the claim $\gamma = \mathit{claim}_l(R, \mathit{synch})$ is correct if

$$\begin{gathered}
\text{for any } t \in \mathit{traces}(P) \text{ there exists a cast function } \Gamma \text{ such that} \\
\text{for any } (\mathit{inst}, \gamma) \in t, \text{ if } \mathit{honest}(\mathit{inst}) \text{ then} \\
\text{for any } \epsilon_1,\ \epsilon_2 \in \mathit{RoleEvent},\ \epsilon_1 \dashrightarrow \epsilon_2 \text{ and } \epsilon_2 \prec_P \gamma \text{ imply} \\
\text{there exists } \mathit{inst}_1, \mathit{inst}_2 \text{ such that} \\
(\mathit{inst}_1, \epsilon_1) <_t (\mathit{inst}_2, \epsilon_2) <_t (\mathit{inst}, \gamma) \text{ and} \\
\mathit{runidof}(\mathit{inst}_1) = \Gamma((\mathit{inst}, \gamma), \mathit{role}(\epsilon_1)) \text{ and} \\
\mathit{runidof}(\mathit{inst}_2) = \Gamma((\mathit{inst}, \gamma), \mathit{role}(\epsilon_2)) \text{ and} \\
\mathit{cont}((\mathit{inst}_1, \epsilon_1)) = \mathit{cont}((\mathit{inst}_2, \epsilon_2))
\end{gathered}$$

Example:
We prove that the claim $\gamma = \mathit{claim}_5(R, \mathit{synch})$ does not hold in the Needham-Schroeder protocol.

$$\begin{aligned}
NS(i) = \ & (\{i, r, ni, sk(i), pk(i), pk(r)\}, \\
& [\mathit{send}_1(i, r, \{\!|\, ni, i \,|\!\}_{pk(r)}), \\
& \mathit{recv}_2(r, i, \{\!|\, ni, V \,|\!\}_{pk(i)}), \\
& \mathit{send}_3(i, r, \{\!|\, V \,|\!\}_{pk(r)}), \\
& \mathit{claim}_4(i, \mathit{synch})])
\end{aligned}
\qquad
\begin{aligned}
NS(r) = \ & (\{i, r, nr, sk(r), pk(r), pk(i)\}, \\
& [\mathit{recv}_1(i, r, \{\!|\, W, i \,|\!\}_{pk(r)}), \\
& \mathit{send}_2(r, i, \{\!|\, W, nr \,|\!\}_{pk(i)}), \\
& \mathit{recv}_3(i, r, \{\!|\, nr \,|\!\}_{pk(r)}), \\
& \mathit{claim}_5(r, \mathit{synch})])
\end{aligned}$$

The "man in the middle" attack on NSPK

$$\begin{tikzcd}
A \arrow[r, "\{\!|\, na,A \,|\!\}_{pk(E)}"] & E & \\
 & E(A) \arrow[r, "\{\!|\, na,A \,|\!\}_{pk(B)}"] & B \\
 & E(A) & B \arrow[l, "\{\!|\, na,nb \,|\!\}_{pk(A)}"'] \\
A & E \arrow[l, "\{\!|\, na,nb \,|\!\}_{pk(A)}"'] & \\
A \arrow[r, "\{\!|\, z \,|\!\}_{pk(E)}"] & E & \\
 & E(A) \arrow[r, "\{\!|\, nb \,|\!\}_{pk(B)}"] & B
\end{tikzcd}$$

Obviously, $\{A, B\}$ are honest agents, E is a corrupted agent. We have to define a trace representing this attack.

The "man in the middle" attack on NSPK

$$\begin{tikzcd}
A \arrow[r, "\{\!|\, na, A \,|\!\}_{pk(E)}"] & E & \\
 & {\color{red}E(A)} \arrow[r, "\{\!|\, na, A \,|\!\}_{pk(B)}"] & B \\
 & {\color{red}E(A)} & B \arrow[l, "\{\!|\, na, nb \,|\!\}_{pk(A)}"'] \\
A & E \arrow[l, "\{\!|\, na, nb \,|\!\}_{pk(A)}"'] & \\
A \arrow[r, "\{\!|\, z \,|\!\}_{pk(E)}"] & E & \\
 & {\color{red}E(A)} \arrow[r, "\{\!|\, nb \,|\!\}_{pk(B)}"] & B
\end{tikzcd}$$

Define $\rho_1 = \{i \mapsto A, r \mapsto E\}$ and $\rho_2 = \{i \mapsto A, r \mapsto B\}$, where $\{A, B\}$ are honest agents, E is a corrupted agent. A trace for the "man in the middle" attack is:

$$
\begin{aligned}
&[((1, \rho_1, \emptyset), create(i))\ ,\\
&((1, \rho_1, \emptyset), send_1(i, r, \{\!|\, ni, i \,|\!\}_{pk(r)})),\\
&((2, \rho_2, \emptyset), create(r)),\\
&((2, \rho_2, \{W \mapsto ni^{\#1}\}), recv_1(i, r, \{\!|\, W, i \,|\!\}_{pk(r)})),\\
&((2, \rho_2, \{W \mapsto ni^{\#1}\}), send_2(r, i, \{\!|\, W, nr \,|\!\}_{pk(i)})),\\
&((1, \rho_1, \{V \mapsto nr^{\#2}\}), recv_2(r, i, \{\!|\, ni, V \,|\!\}_{pk(i)})),\\
&((1, \rho_1, \{V \mapsto nr^{\#2}\}), send_3(i, r, \{\!|\, V \,|\!\}_{pk(r)})),\\
&((2, \rho_2, \{W \mapsto ni^{\#1}\}), recv_3(i, r, \{\!|\, nr \,|\!\}_{pk(r)})),\\
&((2, \rho_2, \{W \mapsto ni^{\#1}\}), {\color{red}claim_5(r, synch)})]
\end{aligned}
$$

The "man in the middle" attack on NSPK

$\rho_1 = \{i \mapsto A, r \mapsto E\}$
$\rho_2 = \{i \mapsto A, r \mapsto B\}$

Let *mt* be the trace

$[((1, \rho_1, \emptyset), \mathit{create}(i))$,
$((1, \rho_1, \emptyset), \mathit{send}_1(i, r, \{\!|\, ni, i \,|\!\}_{pk(r)})),$
$((2, \rho_2, \emptyset), \mathit{create}(r)),$
$((2, \rho_2, \{W \mapsto ni^{\#1}\}), \mathit{recv}_1(i, r, \{\!|\, W, i \,|\!\}_{pk(r)})),$
$((2, \rho_2, \{W \mapsto ni^{\#1}\}), \mathit{send}_2(r, i, \{\!|\, W, nr \,|\!\}_{pk(i)})),$
$((1, \rho_1, \{V \mapsto nr^{\#2}\}), \mathit{recv}_2(r, i, \{\!|\, ni, V \,|\!\}_{pk(i)})),$
$((1, \rho_1, \{V \mapsto nr^{\#2}\}), \mathit{send}_3(i, r, \{\!|\, V \,|\!\}_{pk(r)})),$
$((2, \rho_2, \{W \mapsto ni^{\#1}\}), \mathit{recv}_3(i, r, \{\!|\, nr \,|\!\}_{pk(r)})),$
$((2, \rho_2, \{W \mapsto ni^{\#1}\}), \mathit{claim}_5(r, \mathit{synch}))]$

We prove
there is NO cast function Γ s.t.
for any $(\mathit{inst}, \gamma) \in \mathit{mt}$, if
$\mathit{honest}((\theta, \rho, \sigma))$ then
for any ϵ_1, $\epsilon_2 \in \mathit{RoleEvent}$,
$\epsilon_1 \dashrightarrow \epsilon_2$ and $\epsilon_2 \prec_P \gamma$
imply
there exists $\mathit{inst}_1, \mathit{inst}_2$ s.t.
$(\mathit{inst}_1, \epsilon_1) <_{mt} (\mathit{inst}_2, \epsilon_2) <_{mt} (\mathit{inst}, \gamma)$
and
$\mathit{runidof}(\mathit{inst}_1) = \Gamma((\mathit{inst}, \gamma), \mathit{role}(\epsilon_1))$
and
$\mathit{runidof}(\mathit{inst}_2) = \Gamma((\mathit{inst}, \gamma), \mathit{role}(\epsilon_2))$
and
$\mathit{cont}((\mathit{inst}_1, \epsilon_1)) = \mathit{cont}((\mathit{inst}_2, \epsilon_2))$

The "man in the middle" attack on NSPK

$\rho_1 = \{i \mapsto A, r \mapsto E\}$
$\rho_2 = \{i \mapsto A, r \mapsto B\}$

Let mt be the trace

$[((1, \rho_1, \emptyset), create(i))$,
$((1, \rho_1, \emptyset), send_1(i, r, \{\!| ni, i |\!\}_{pk(r)})),$
$((2, \rho_2, \emptyset), create(r)),$
$((2, \rho_2, \{W \mapsto ni^{\#1}\}), recv_1(i, r, \{\!| W, i |\!\}_{pk(r)})),$
$((2, \rho_2, \{W \mapsto ni^{\#1}\}), send_2(r, i, \{\!| W, nr |\!\}_{pk(i)})),$
$((1, \rho_1, \{V \mapsto nr^{\#2}\}), recv_2(r, i, \{\!| ni, V |\!\}_{pk(i)})),$
$((1, \rho_1, \{V \mapsto nr^{\#2}\}), send_3(i, r, \{\!| V |\!\}_{pk(r)})),$
$((2, \rho_2, \{W \mapsto ni^{\#1}\}), recv_3(i, r, \{\!| nr |\!\}_{pk(r)})),$
$((2, \rho_2, \{W \mapsto ni^{\#1}\}), claim_5(r, synch))]$

Assume Γ **is a cast function** for mt.
If $inst = (2, \rho_2, \{W \mapsto ni^{\#1}\})$,
$\gamma = claim_5(r, synch)$ and
$cv = (inst, \gamma)$ then
$honest(inst)$,
$\Gamma(cv, r) = 2$ and $\Gamma(cv, i) = 1$.

Let $\epsilon_1 = send_1(i, r, \{\!| ni, i |\!\}_{pk(r)})$
$\epsilon_2 = recv_1(i, r, \{\!| W, i |\!\}_{pk(r)})$
and note that
$inst_1 = (1, \rho_1, \emptyset)$ and
$inst_2 = (2, \rho_2, \{W \mapsto ni^{\#1}\})$ are the only choices such that
$(inst_1, \epsilon_1) <_{mt} (inst_2, \epsilon_2) <_{mt} (inst, \gamma)$.

Since
$cont(inst_1, \epsilon_1) = (A, E, \{\!| ni^{\#1}, A |\!\}_{pk(E)})$
and
$cont(inst_2, \epsilon_2) = (A, B, \{\!| ni^{\#1}, A |\!\}_{pk(B)})$
are different, our assumption is false.

Secrecy in NSPK

In the sequel we prove that the same trace provides a counter example for the secrecy claim of the responder:

$$\begin{aligned} NS(r) = \quad & (\{i, r, nr, sk(r), pk(r), pk(i)\}, \\ & [recv_1(i, r, \{\!|\, W, i \,|\!\}_{pk(r)}), \\ & send_2(r, i, \{\!|\, W, nr \,|\!\}_{pk(i)}), \\ & recv_3(i, r, \{\!|\, nr \,|\!\}_{pk(r)}), \\ & claim_5(r, secret, nr)]) \end{aligned}$$

Recall that the adversary knowledge is enriched by the send rule:

$$[send]\ \frac{e = send_l(R_1, R_2, m) \quad (inst, [e] \cdot s) \in F}{\langle\!\langle AKN, F \rangle\!\rangle \xrightarrow{(inst, e)} \langle\!\langle AKN \cup \{inst(m)\}, F \setminus \{(inst, [e] \cdot s)\} \cup \{(inst, s)\} \rangle\!\rangle}$$

so we follow the evolution of AKN along the trace.

Secrecy in NSPK

Let $\rho_1 = \{i \mapsto A, r \mapsto E\}$ and $\rho_2 = \{i \mapsto A, r \mapsto B\}$, where $\{A, B\}$ are honest agents, E is a corrupted agent.

$[((1, \rho_1, \emptyset), create(i))$,
$AKN_0 = Agent \cup \{pk(A), pk(B), pk(E)\} \cup \{sk(E)\}$

$((1, \rho_1, \emptyset), send_1(i, r, \{\!|\, ni, i \,|\!\}_{pk(r)})),$
$AKN_1 = AKN_0 \cup \{\{\!|\, ni^{\#1}, A \,|\!\}_{pk(E)}\}$

$((2, \rho_2, \emptyset), create(r)),$
$AKN_2 = AKN_1$

$((2, \rho_2, \{W \mapsto ni^{\#1}\}), recv_1(i, r, \{\!|\, W, i \,|\!\}_{pk(r)})),$
$AKN_3 = AKN_2$

$((2, \rho_2, \{W \mapsto ni^{\#1}\}), send_2(r, i, \{\!|\, W, nr \,|\!\}_{pk(i)})),$
$AKN_4 = AKN_3 \cup \{\{\!|\, ni^{\#1}, nr^{\#2} \,|\!\}_{pk(A)}\}$

Secrecy in NSPK

$((2, \rho_2, \{W \mapsto ni^{\#1}\}), send_2(r, i, \{\!|\, W, nr \,|\!\}_{pk(i)})),$
$AKN_4 = AKN_3 \cup \{\{\!|\, ni^{\#1}, nr^{\#2} \,|\!\}_{pk(A)}\}$

$((1, \rho_1, \{V \mapsto nr^{\#2}\}), recv_2(r, i, \{\!|\, ni, V \,|\!\}_{pk(i)})),$
$AKN_5 = AKN_4$

$((1, \rho_1, \{V \mapsto nr^{\#2}\}), send_3(i, r, \{\!|\, V \,|\!\}_{pk(r)})),$
$AKN_6 = AKN_5 \cup \{\{\!|\, nr^{\#2} \,|\!\}_{pk(E)}\}$

$((2, \rho_2, \{W \mapsto ni^{\#1}\}), recv_3(i, r, \{\!|\, nr \,|\!\}_{pk(r)})),$
$AKN_7 = AKN_6$

$((2, \rho_2, \{W \mapsto ni^{\#1}\}), claim_5(r, secret, nr))]$
$AKN_7 \vdash nr^{\#2}$ since $\{sk(E), \{\!|\, nr^{\#2} \,|\!\}_{pk(E)}\} \subseteq AKN_7$

Security properties

- For a protocol P the claim $\gamma = \mathit{claim}_l(R, \mathit{synch})$ is correct if

$$\begin{array}{c}
\text{for any } t \in \mathit{traces}(P) \text{ there exists a cast function } \Gamma \text{ such that} \\
\text{for any } (\mathit{inst}, \gamma) \in t, \text{ if } \mathit{honest}(\mathit{inst}) \text{ then} \\
\text{for any } \epsilon_1,\ \epsilon_2 \in \mathit{RoleEvent},\ \epsilon_1 \dashrightarrow \epsilon_2 \text{ and } \epsilon_2 \prec_P \gamma \text{ imply} \\
\text{there exists } \mathit{inst}_1, \mathit{inst}_2 \text{ such that} \\
(\mathit{inst}_1, \epsilon_1) <_t (\mathit{inst}_2, \epsilon_2) <_t (\mathit{inst}, \gamma) \text{ and} \\
\mathit{runidof}(\mathit{inst}_1) = \Gamma((\mathit{inst}, \gamma), \mathit{role}(\epsilon_1)) \text{ and} \\
\mathit{runidof}(\mathit{inst}_2) = \Gamma((\mathit{inst}, \gamma), \mathit{role}(\epsilon_2)) \text{ and} \\
\mathit{cont}((\mathit{inst}_1, \epsilon_1)) = \mathit{cont}((\mathit{inst}_2, \epsilon_2))
\end{array}$$

  This property is called *non-injective synchronization* ($\mathit{ni} - \mathit{synch}$) and it **does not prevent replay attacks**.
- *Injective synchronization* is a stronger property, that also prevents reply attacks. It is defined adding the requirement that Γ, the cast function, is injective. This means that, for the same role, two different instances of a claim correspond to different runs.

Security properties: sync vs agree

- For a protocol P the claim $\gamma = claim_l(R, synch)$ is correct if

$$\begin{array}{c}\text{for any } t \in traces(P) \text{ there exists a cast function } \Gamma \text{ such that}\\ \text{for any } (inst, \gamma) \in t, \text{ if } honest(inst) \text{ then}\\ \text{for any } \epsilon_1,\ \epsilon_2 \in RoleEvent,\ \epsilon_1 \dashrightarrow \epsilon_2 \text{ and } \epsilon_2 \prec_P \gamma \text{ imply}\\ \text{there exists } inst_1, inst_2 \text{ such that}\\ (inst_1, \epsilon_1) <_t (inst_2, \epsilon_2) <_t (inst, \gamma) \text{ and}\\ runidof(inst_1) = \Gamma((inst, \gamma), role(\epsilon_1)) \text{ and } runidof(inst_2) = \Gamma((inst, \gamma), role(\epsilon_2)) \text{ and}\\ cont((inst_1, \epsilon_1)) = cont((inst_2, \epsilon_2))\end{array}$$

- For a protocol P the claim $\gamma = claim_l(R, agree)$ is correct if

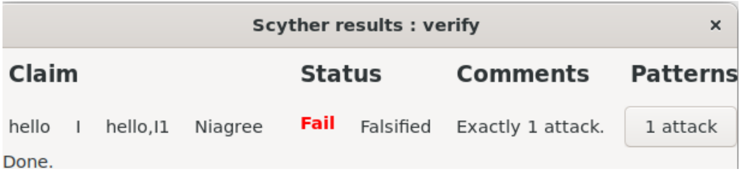
$$\begin{array}{c}\text{for any } t \in traces(P) \text{ there exists a cast function } \Gamma \text{ such that}\\ \text{for any } (inst, \gamma) \in t, \text{ if } honest(inst) \text{ then}\\ \text{for any } \epsilon_1,\ \epsilon_2 \in RoleEvent,\ \epsilon_1 \dashrightarrow \epsilon_2 \text{ and } \epsilon_2 \prec_P \gamma \text{ imply}\\ \text{there exists } inst_1, inst_2 \text{ such that}\\ (inst_1, \epsilon_1) <_t (inst, \gamma) \textbf{ and } (inst_2, \epsilon_2) <_t (inst, \gamma) \textbf{ and}\\ runidof(inst_1) = \Gamma((inst, \gamma), role(\epsilon_1)) \text{ and}\\ runidof(inst_2) = \Gamma((inst, \gamma), role(\epsilon_2)) \text{ and}\\ cont((inst_1, \epsilon_1)) = cont((inst_2, \epsilon_2))\end{array}$$

  The parties agree on the values of the variables (message content), but the communication order might not be respected.

Scyther: Niagree

```
protocol exagree(I,R)
   role I{
   fresh m: Nonce;
   fresh n: Nonce;
   send_1(I,R, (n,m));
   recv_2(R,I, {I,m}sk(R) );
   claim(I,Niagree);}
```

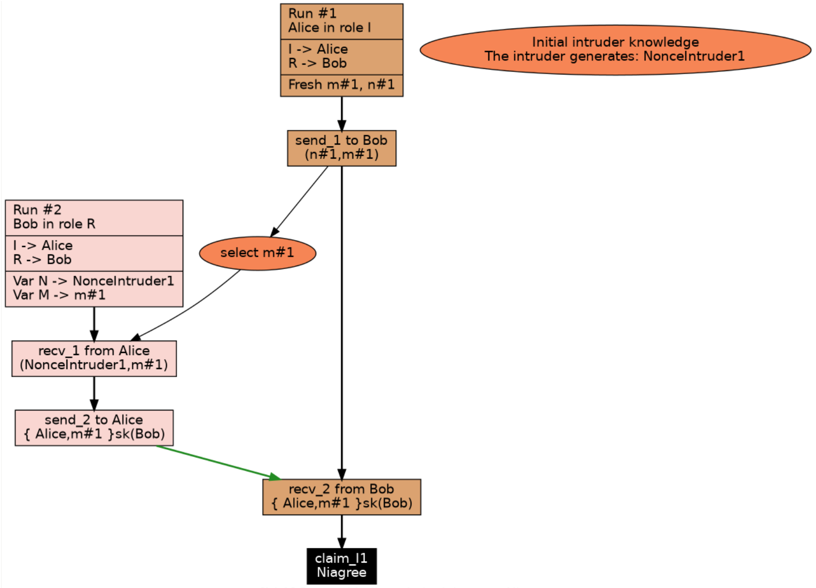
```
role R
 {var M,N: Nonce;
  recv_1(I,R,(N,M ));
  send_2(R,I, {I,M}sk(R) );}}
```



Claim				Status		Comments	Patterns
hello	I	hello,I1	Niagree	Fail	Falsified	Exactly 1 attack.	1 attack

Done.

Scyther: atack on Niagree



[Id 1] Protocol hello, role I, claim type Niagree