Theory and Tool Support for the Formal Verification of Cryptographic Protocols

Sébastien Briais

École Polytechnique Fédérale de Lausanne

2007, December 17th
Cryptographic protocols are error-prone

Cryptographic protocols

To secure communication over insecure networks (e.g. Internet). A communication protocol that uses cryptography to achieve security goals.
The Yahalom protocol

1. $A \rightarrow B$: $(A, n_A)$
2. $B \rightarrow S$: $(B, Enc_{sk_B}(A, (n_A, n_B)))$
3. $S \rightarrow A$: $(Enc_{sk_A}(B, k_{AB}), (n_A, n_B), Enc_{sk_B}(A, k_{AB}))$
4. $A \rightarrow B$: $(Enc_{sk_B}(A, k_{AB}), Enc_{sk_{AB}}(n_B))$
The Yahalom protocol

1. $A \rightarrow B : (A \cdot n_A)$
The Yahalom protocol

1. $A \rightarrow B : (A \cdot n_A)$
2. $B \rightarrow S : (B \cdot \text{Enc}_{k_{BS}}^S (A \cdot (n_A \cdot n_B)))$
The Yahalom protocol

1. $A \rightarrow B : \ (A \cdot n_A)$
2. $B \rightarrow S : \ (B \cdot \text{Enc}_{k_{BS}}^S (A \cdot (n_A \cdot n_B)))$
3. $S \rightarrow A : \ (\text{Enc}_{k_{AS}}^S ((B \cdot k_{AB}) \cdot (n_A \cdot n_B)) \cdot \text{Enc}_{k_{BS}}^S (A \cdot k_{AB}))$
The Yahalom protocol

1. $A \rightarrow B : (A \cdot n_A)$
2. $B \rightarrow S : (B \cdot \text{Enc}_{k_{BS}}^s(A \cdot (n_A \cdot n_B)))$
3. $S \rightarrow A : (\text{Enc}_{k_{AS}}^s((B \cdot k_{AB}) \cdot (n_A \cdot n_B)) \cdot \text{Enc}_{k_{BS}}^s(A \cdot k_{AB}))$
4. $A \rightarrow B : (\text{Enc}_{k_{BS}}^s(A \cdot k_{AB}) \cdot \text{Enc}_{k_{AB}}^s n_B)$
Cryptographic protocols are error-prone

Cryptographic protocols
To secure communication over insecure networks (e.g. Internet).
A communication protocol that uses cryptography to achieve security goals.

... are error-prone

- Even when assuming perfect cryptographic primitives
- Canonical example: Needham-Schroeder with public key
Cryptographic protocols are error-prone

Cryptographic protocols
To secure communication over insecure networks (e.g. Internet).
A communication protocol that uses cryptography to achieve security goals.

... are error-prone
- Even when assuming perfect cryptographic primitives
- Canonical example: Needham-Schroeder with public key

Why is it difficult?
Distributed algorithms that have the obligation to behave robustly in the context of unknown hostile attackers.
The spi calculus approach
Abadi and Gordon

- Cryptographic protocols are described in a precise and concise way.

- Equations to formulate security objectives.
  - secrecy: $P^M/x \approx P^N/x$ for any $M$ and $N$
  - authenticity
The spi calculus approach
Abadi and Gordon

- Cryptographic protocols are described in a precise and concise way.

\[(\nu k_{AS}, k_{BS}) (\nu n_A) \overline{B}(A \cdot n_A) \cdot A(x_2) \cdot \phi_2 \overline{B} \langle E_2 \rangle \cdot 0 \]

\[| (\nu n_B) B(x_0) \cdot \phi_0 \overline{S}(B \cdot \text{Enc}_{k_{BS}}^s (A \cdot (\pi_2 (x_0) \cdot n_B))) \rangle \cdot B(x_3) \cdot \phi_3 0 \]

\[| (\nu k_{AB}) S(x_1) \cdot \phi_1 \overline{A} \langle E_1 \rangle \cdot 0 \]

- Equations to formulate security objectives.

  - secrecy: \( P\{M/x\} \approx P\{N/x\} \) for any \( M \) and \( N \)
  - authenticity
The spi calculus approach
Abadi and Gordon

○ Cryptographic protocols are described in a precise and concise way.

○ Equations to formulate security objectives.
  ▶ secrecy: $P^M_x \approx P^N_x$ for any $M$ and $N$
  ▶ authenticity
The spi calculus approach
Abadi and Gordon

- Cryptographic protocols are described in a precise and concise way.

- Equations to formulate security objectives.
  - secrecy: $P^{M/x} \approx P^{N/x}$ for any $M$ and $N$
  - authenticity
The spi calculus approach
Abadi and Gordon

- Cryptographic protocols are described in a precise and concise way.

- Equations to formulate security objectives.
  - secrecy: $P^{M/x} \approx P^{N/x}$ for any $M$ and $N$
  - authenticity
Testing equivalence

- Usually $\approx$ stands for *testing equivalence*.
- Intuitively, $P$ and $Q$ are testing equivalent *if and only if* they reveal the same information to observers (i.e. attackers).
Testing equivalence

- Usually \( \approx \) stands for testing equivalence.
- Intuitively, \( P \) and \( Q \) are testing equivalent if and only if they reveal the same information to observers (i.e. attackers).
- Formally, \( P \) passes the test \((R, \beta)\) iff \( P \mid R \downarrow \beta \), i.e. \( P \mid R \) may communicate on channel \( \beta \).
- \( P \simeq Q \) iff they pass the same tests, i.e. for any \((R, \beta)\),

\[
P \mid R \downarrow \beta \iff Q \mid R \downarrow \beta
\]
Testing equivalence

- Usually \( \approx \) stands for testing equivalence.
- Intuitively, \( P \) and \( Q \) are testing equivalent if and only if they reveal the same information to observers (i.e. attackers).
- Formally, \( P \) passes the test \((R, \beta)\) iff \( P \mid R \downarrow_\beta \), i.e. \( P \mid R \) may communicate on channel \( \beta \).
- \( P \simeq Q \) iff they pass the same tests, i.e. for any \((R, \beta)\),

\[
P \mid R \downarrow_\beta \iff Q \mid R \downarrow_\beta
\]

- **Problem**: infinite quantification over arbitrary observers \( R \).
- In practise, we define sound approximations that are easier to work with: bisimulations.
Bisimulations

- Behaviour of processes is described with a *Labelled Transitions System*: $P \xrightarrow{\mu} P'$
- Two processes are bisimilar if they can play the same transitions

\[
P \\
\downarrow \\
Q
\]
Bisimulations

- Behaviour of processes is described with a *Labelled Transitions System*: $P \xrightarrow{\mu} P'$
- Two processes are bisimilar if they can play the same transitions

\[
P \xrightarrow{\mu} P' \\
\downarrow{}^{Q} \\
Q
\]
Bisimulations

- Behaviour of processes is described with a *Labelled Transitions System*: $P \xrightarrow{\mu} P'$
- Two processes are bisimilar if they can play the same transitions

\[ P \xrightarrow{\mu} P' \]
\[ Q \xrightarrow{\mu} Q' \]
\[ Q \text{ replies to } P \]
Bisimulations

- Behaviour of processes is described with a *Labelled Transitions System*: $P \xrightarrow{\mu} P'$
- Two processes are bisimilar if they can play the same transitions

\[ \begin{align*}
P & \xrightarrow{\mu} P' \\
\downarrow & \\
Q & \xrightarrow{\mu} Q'
\end{align*} \quad \text{Q replies to } P \]
Bisimulations

- Behaviour of processes is described with a *Labelled Transitions System*: $P \xrightarrow{\mu} P'$
- Two processes are bisimilar if they can play the same transitions

\[ P \xrightarrow{\mu} P' \]
\[ Q \xrightarrow{\mu} Q' \]

\[ Q \text{ replies to } P \]

\[ P \text{ replies to } Q \]
Contributions

From protocol narrations to spi calculus

A formal semantics for protocol narrations.
A rigorous translation to spi calculus.
Deciding process equivalence

A new notion of bisimulation for the spi calculus.
A symbolic characterisation.
Contributions

Towards a certified tool
Formalization of large parts of the developed theory in Coq.
*Dream:* Have a correct-by-construction tool.
Contributions

1 subgoal

bisimilar P Q

Reasoning within Coq

Reason formally about cryptographic protocols in Coq.
Outline

1. From protocol narrations to spi calculus
2. An open variant of bisimulation for the spi calculus
3. A formalization in Coq
Outline

1. From protocol narrations to spi calculus
2. An open variant of bisimulation for the spi calculus
3. A formalization in Coq
The Yahalom protocol

1. $A \rightarrow B : (A \cdot n_A)$
2. $B \rightarrow S : (B \cdot \text{Enc}^{S}_{k_{BS}}(A \cdot (n_A \cdot n_B)))$
3. $S \rightarrow A : (\text{Enc}^{S}_{k_{AS}}((B \cdot k_{AB}) \cdot (n_A \cdot n_B)) \cdot \text{Enc}^{S}_{k_{BS}}(A \cdot k_{AB}))$
4. $A \rightarrow B : (\text{Enc}^{S}_{k_{BS}}(A \cdot k_{AB}) \cdot \text{Enc}^{S}_{k_{AB}} n_B)$
The Yahalom protocol in spi-calculus

\[(\nu k_{AS}, k_{BS})\]
\[(\nu n_A) \overline{B} ((A \cdot n_A) \cdot A(x_2) \cdot \phi_2 \overline{B} ((\pi_2(x_2) \cdot \text{Enc}^s_{\pi_2(\pi_1(\text{Dec}^s_{k_{AS}} \pi_1(x_2)))) \pi_2(\pi_2(\text{Dec}^s_{k_{AS}} \pi_1(x_2)))))) \cdot 0\]

\[| (\nu n_B) B(x_0) \cdot \phi_0 \overline{S} ((B \cdot \text{Enc}^s_{k_{BS}} (A \cdot (\pi_2(x_0) \cdot n_B))) \cdot B(x_3) \cdot \phi_3) 0\]

\[| (\nu k_{AB}) S(x_1) \cdot \phi_1\]
\[\overline{A} ((\text{Enc}^s_{k_{AS}} ((B \cdot k_{AB}) (\pi_1(\pi_2(\text{Dec}^s_{k_{BS}} \pi_2(x_1)))) \cdot \pi_2(\pi_2(\text{Dec}^s_{k_{BS}} \pi_2(x_1)))) \cdot \text{Enc}^s_{k_{BS}} (A \cdot k_{AB}))) \cdot 0\]
The Yahalom protocol in spi-calculus

\[(\nu k_{AS}, k_{BS})\]
\[(\nu n_A) B(\pi_2(x_2) \cdot \text{Enc}_{\pi_2(\text{Dec}_{k_{AS}}(\pi_1(x_2)))}^{\pi_2(\pi_1(\text{Dec}_{k_{AS}}(\pi_1(x_2)))))}) \cdot 0\]
\[
| (\nu n_B) B(x_0) \cdot \phi_0 S(B \cdot \text{Enc}_{k_{BS}}^{\pi_2(\pi_1(\text{Dec}_{k_{BS}}(\pi_1(x_2)))))}) \cdot B(x_3) \cdot \phi_3 0 \]
\[
| (\nu k_{AB})
S(x_1) \cdot \phi_1
\]
\[
\overline{A}(\text{Enc}_{k_{AS}}^{\pi_1(\text{Dec}_{k_{BS}}^{\pi_2(\pi_1(\text{Dec}_{k_{BS}}^{\pi_2(x_1))))})}) \cdot \text{Enc}_{k_{BS}}^{\pi_2(x_1)}) \cdot 0
\]
The Yahalom protocol in spi-calculus

\[
\begin{align*}
(\nu k_{AS}, k_{BS}) & \\
(\nu n_A) \overline{B}((A \cdot n_A)) \cdot A(x_2) \cdot \phi_2 \overline{B}((\pi_2(x_2) \cdot \text{Enc}_{\pi_2}(\pi_1(\text{Dec}_{k_{AS}} \pi_1(x_2)))) \cdot \pi_2(\pi_2(\text{Dec}_{k_{AS}} \pi_1(x_2)))) \cdot 0 \\
\mid (\nu n_B) B(x_0) \cdot \phi_0 \overline{S}((B \cdot \text{Enc}_{k_{BS}}(A \cdot (\pi_2(x_0) \cdot n_B)))) \cdot B(x_3) \cdot \phi_3 0 \\
\mid (\nu k_{AB}) & \\
S(x_1) \cdot \phi_1 & \\
\overline{A}((\text{Enc}_{k_{AS}}((B \cdot k_{AB}) \cdot (\pi_1(\pi_2(\text{Dec}_{k_{BS}} \pi_2(x_1)))) \cdot \pi_2(\pi_2(\text{Dec}_{k_{BS}} \pi_2(x_1)))) \cdot \text{Enc}_{k_{BS}}(A \cdot k_{AB}))) \cdot 0
\end{align*}
\]

\[
\begin{align*}
\phi_0 & = [A = \pi_1(x_0)] \\
\phi_1 & = [\pi_1(\pi_2(\text{Dec}_{k_{BS}}(\pi_2(x_1)))) : M] \land [B = \pi_1(x_1)] \land [A = \pi_1(\text{Dec}_{k_{BS}}(\pi_2(x_1)))] \\
\phi_2 & = [B = \pi_1(\pi_1(\text{Dec}_{k_{AS}}(\pi_1(x_2))))] \land [n_A = \pi_1(\pi_2(\text{Dec}_{k_{AS}}(\pi_1(x_2)))] \\
\phi_3 & = [A = \pi_1(\text{Dec}_{k_{BS}}(\pi_1(x_3)))] \land [n_B = \text{Dec}^s_{k_{BS}}(\pi_2(\text{Dec}_{k_{BS}}(\pi_1(x_3))) \cdot \pi_2(x_3)]
\end{align*}
\]
State explicitly the assumptions

A protocol narration does not explicitly state the initial knowledge and what is to be generated freshly during a protocol run.
State explicitly the assumptions

A protocol narration does not explicitly state the initial knowledge and what is to be generated freshly during a protocol run.

\[ A, S \text{ share } k_{AS} \]
\[ B, S \text{ share } k_{BS} \]
\[ A \text{ generates } n_A ; B \text{ generates } n_B ; S \text{ generates } k_{AB} ; \]
\[ A \xrightarrow{} B : (A \cdot n_A) ; \]
\[ B \xrightarrow{} S : (B \cdot \text{Enc}^s_{k_{BS}}(A \cdot (n_A \cdot n_B))) ; \]
\[ S \xrightarrow{} A : (\text{Enc}^s_{k_{AS}}((B \cdot k_{AB}) \cdot (n_A \cdot n_B))) \cdot \text{Enc}^s_{k_{BS}}(A \cdot k_{AB}) ; \]
\[ A \xrightarrow{} B : (\text{Enc}^s_{k_{BS}}(A \cdot k_{AB}) \cdot \text{Enc}^s_{k_{AB}} \cdot n_B) \]
State explicitly the assumptions

A protocol narration does not explicitly state the initial knowledge and what is to be generated freshly during a protocol run.

Principals act concurrently

A protocol narration describes an idealised sequential trace of execution whereas the principals act concurrently. $A \rightarrow B : M$ actually means

(i) $A$ asynchronously sends $M$ towards $B$,
(ii) $B$ receives some message
State explicitly the assumptions

A protocol narration does not explicitly state the initial knowledge and what is to be generated freshly during a protocol run.

Principals act concurrently

A protocol narration describes an idealised sequential trace of execution whereas the principals act concurrently.

\( A \rightarrow B : M \) actually means

(i) \( A \) asynchronously sends \( M \) towards \( B \),

(ii) \( B \) receives some message (intended to be \( M \))

Principals perform on-reception checks

(iii) \( B \) checks that the message it just received has the expected properties.
State explicitly the assumptions

A protocol narration does not explicitly state the initial knowledge and what is to be generated freshly during a protocol run.

\[ A, S \text{ share } k_{AS} \]
\[ B, S \text{ share } k_{BS} \]
\[ A \text{ generates } n_A \; ; \; B \text{ generates } n_B \; ; \; S \text{ generates } k_{AB} \; ; \]
\[ A \xrightarrow{} B : (A \cdot n_A) \; ; \]
\[ B \xrightarrow{} S : (B \cdot \text{Enc}_{k_{BS}}^S((A \cdot (n_A \cdot n_B)))) \; ; \]
\[ S \xrightarrow{} A : (\text{Enc}_{k_{AS}}^S((B \cdot k_{AB}) \cdot (n_A \cdot n_B)) \cdot \text{Enc}_{k_{BS}}^S(A \cdot k_{AB})) \; ; \]
\[ A \xrightarrow{} B : (\text{Enc}_{k_{BS}}^S(A \cdot k_{AB}) \cdot \text{Enc}_{k_{AB}}^S n_B) \]
Generating the checks

Current knowledge

\{A, B, S, k_{AS}, n_A\}

expected

\[
\text{Enc}_{k_{AS}}^S((B \cdot k_{AB}) \cdot (n_A \cdot n_B)) \cdot \text{Enc}_{k_{BS}}^S(A \cdot k_{AB})
\]
Generating the checks

Current knowledge

\[ \{ A, B, S, k_{AS}, n_A \} \]

\[ \text{expected} \]

\[ \frac{1}{(\text{Enc}_{k_{AS}}^S((B \cdot k_{AB}) \cdot (n_A \cdot n_B)) \cdot \text{Enc}_{k_{BS}}^S(A \cdot k_{AB}))} \]
Generating the checks

Current knowledge

\[ \{A, B, S, k_{AS}, n_A\} \]

<table>
<thead>
<tr>
<th>\text{expected}</th>
<th>\text{actual}</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{Enc}^s_{k_{AS}}((B \cdot k_{AB}) \cdot (n_A \cdot n_B)) \cdot \text{Enc}^s_{k_{BS}}(A \cdot k_{AB}) )</td>
<td>( x )</td>
</tr>
</tbody>
</table>
## Generating the checks

### Current knowledge

\[
\{A, B, S, k_{AS}, n_A\}
\]

<table>
<thead>
<tr>
<th>expected</th>
<th>actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>((\text{Enc}<em>{k</em>{AS}}^S((B \cdot k_{AB}) \cdot (n_A \cdot n_B)) \cdot \text{Enc}<em>{k</em>{BS}}^S(A \cdot k_{AB})))</td>
<td>(x)</td>
</tr>
<tr>
<td>(\text{Enc}<em>{k</em>{AS}}^S((B \cdot k_{AB}) \cdot (n_A \cdot n_B)))</td>
<td>(\pi_1(x))</td>
</tr>
<tr>
<td>(\text{Enc}<em>{k</em>{BS}}^S(A \cdot k_{AB}))</td>
<td>(\pi_2(x))</td>
</tr>
</tbody>
</table>
## Generating the checks

### Current knowledge

\[ \{A, B, S, k_{AS}, n_A\} \]

<table>
<thead>
<tr>
<th>expected</th>
<th>actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{Enc}^s_{k_{AS}}((B \cdot k_{AB})) \cdot (n_A \cdot n_B)) \cdot \text{Enc}^s_{k_{BS}}(A \cdot k_{AB}))</td>
<td>x</td>
</tr>
<tr>
<td>\text{Enc}^s_{k_{AS}}((B \cdot k_{AB})) \cdot (n_A \cdot n_B))</td>
<td>\pi_1(x)</td>
</tr>
<tr>
<td>\text{Enc}^s_{k_{BS}}(A \cdot k_{AB})</td>
<td>\pi_2(x)</td>
</tr>
<tr>
<td>((B \cdot k_{AB})) \cdot (n_A \cdot n_B))</td>
<td>\text{Dec}^s_{k_{AS}} \pi_1(x)</td>
</tr>
</tbody>
</table>

Sébastien Briais (EPFL)  
PhD Defense  
2007, December 17th  
14 / 29
## Generating the checks

### Current knowledge

\[ \{A, B, S, k_{AS}, n_A\} \]

<table>
<thead>
<tr>
<th>expected</th>
<th>actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>((\text{Enc}<em>{k</em>{AS}}^S((B \cdot k_{AB}) \cdot (n_A \cdot n_B)) \cdot \text{Enc}<em>{k</em>{BS}}^S(A \cdot k_{AB})))</td>
<td>(x)</td>
</tr>
<tr>
<td>(\text{Enc}<em>{k</em>{AS}}^S((B \cdot k_{AB}) \cdot (n_A \cdot n_B)))</td>
<td>(\pi_1(x))</td>
</tr>
<tr>
<td>(\text{Enc}<em>{k</em>{BS}}^S(A \cdot k_{AB}))</td>
<td>(\pi_2(x))</td>
</tr>
<tr>
<td>(((B \cdot k_{AB}) \cdot (n_A \cdot n_B)))</td>
<td>(\text{Dec}<em>{k</em>{AS}}^S \pi_1(x))</td>
</tr>
<tr>
<td>((B \cdot k_{AB}))</td>
<td>(\pi_1(\text{Dec}<em>{k</em>{AS}}^S \pi_1(x)))</td>
</tr>
<tr>
<td>((n_A \cdot n_B))</td>
<td>(\pi_2(\text{Dec}<em>{k</em>{AS}}^S \pi_1(x)))</td>
</tr>
<tr>
<td>(B)</td>
<td>(\pi_1(\pi_1(\text{Dec}<em>{k</em>{AS}}^S \pi_1(x))))</td>
</tr>
<tr>
<td>(k_{AB})</td>
<td>(\pi_1(\pi_2(\text{Dec}<em>{k</em>{AS}}^S \pi_1(x))))</td>
</tr>
<tr>
<td>(n_A)</td>
<td>(\pi_2(\pi_2(\text{Dec}<em>{k</em>{AS}}^S \pi_1(x))))</td>
</tr>
<tr>
<td>(n_B)</td>
<td></td>
</tr>
</tbody>
</table>
## Generating the checks

### Current knowledge

$$\{A, B, S, k_{AS}, n_A\}$$

<table>
<thead>
<tr>
<th>expected</th>
<th>actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\text{Enc}^s_{KAS}((B \cdot k_{AB}) \cdot (n_A \cdot n_B))) \cdot \text{Enc}^s_{KBS}(A \cdot k_{AB}))$</td>
<td>$x$</td>
</tr>
<tr>
<td>$\text{Enc}^s_{KAS}((B \cdot k_{AB}) \cdot (n_A \cdot n_B))$</td>
<td>$\pi_1(x)$</td>
</tr>
<tr>
<td>$\text{Enc}^s_{KBS}(A \cdot k_{AB})$</td>
<td>$\pi_2(x)$</td>
</tr>
<tr>
<td>$((B \cdot k_{AB}) \cdot (n_A \cdot n_B))$</td>
<td>$\text{Dec}^s_{KAS} \pi_1(x)$</td>
</tr>
<tr>
<td>$(B \cdot k_{AB})$</td>
<td>$\pi_1(\text{Dec}^s_{KAS} \pi_1(x))$</td>
</tr>
<tr>
<td>$(n_A \cdot n_B)$</td>
<td>$\pi_2(\text{Dec}^s_{KAS} \pi_1(x))$</td>
</tr>
<tr>
<td>$B$</td>
<td>$\pi_1(\pi_1(\text{Dec}^s_{KAS} \pi_1(x)))$</td>
</tr>
<tr>
<td>$k_{AB}$</td>
<td>$\pi_2(\pi_1(\text{Dec}^s_{KAS} \pi_1(x)))$</td>
</tr>
<tr>
<td>$n_A$</td>
<td>$\pi_1(\pi_2(\text{Dec}^s_{KAS} \pi_1(x)))$</td>
</tr>
<tr>
<td>$n_B$</td>
<td>$\pi_2(\pi_2(\text{Dec}^s_{KAS} \pi_1(x)))$</td>
</tr>
</tbody>
</table>
The Yahalom protocol in spi-calculus

\[(\nu k_{AS}, k_{BS}) \quad (\nu n_A) \overline{B}(A \cdot n_A) \cdot A(x_2) \cdot \phi_2 \overline{B}(\pi_2(x_2) \cdot Enc_s_{\pi_2(\pi_1(Dec_{k_{AS}} \pi_1(x_2)))}) \pi_2(\pi_2(Dec_{k_{AS}} \pi_1(x_2))) \cdot 0 \]

| (\nu n_B) B(x_0) \cdot \phi_0 \overline{S}(B \cdot Enc_{k_{BS}}(A \cdot (\pi_2(x_0) \cdot n_B))) \cdot B(x_3) \cdot \phi_3 \cdot 0 |

| (\nu k_{AB}) \quad S(x_1) \cdot \phi_1 |

\[\overline{A}(Enc_{k_{AS}}((B \cdot k_{AB}) \cdot (\pi_1(\pi_2(Dec_{k_{BS}} \pi_2(x_1)))) \cdot \pi_2(\pi_2(Dec_{k_{BS}} \pi_2(x_1)))) \cdot Enc_{k_{BS}}(A \cdot k_{AB})) \cdot 0 \]

\[\phi_0 = [A = \pi_1(x_0)] \]
\[\phi_1 = [\pi_1(\pi_2(Dec_{k_{BS}} \pi_2(x_1))) : M] \land [B = \pi_1(x_1)] \land [A = \pi_1(Dec_{k_{BS}} \pi_2(x_1))] \]
\[\phi_2 = [B = \pi_1(\pi_1(Dec_{k_{AS}} \pi_1(x_2)))) \land [n_A = \pi_1(\pi_2(Dec_{k_{AS}} \pi_1(x_2))))] \]
\[\phi_3 = [A = \pi_1(Dec_{k_{BS}} \pi_1(x_3))] \land [n_B = Dec_{\pi_2(Dec_{k_{BS}} \pi_1(x_3))}(x_3)] \]
Outline

1. From protocol narrations to spi calculus
2. An open variant of bisimulation for the spi calculus
3. A formalization in Coq
Situation in the pi calculus

- Spi calculus is an extension of the pi calculus that incorporates cryptographic primitives.

\[
P, Q ::= 0 \mid a(x).P \mid \overline{a}(u).P \\
\mid [a = b]P \mid (\nu x)P \\
\mid P \mid Q \mid P + Q \mid !P
\]

- Open bisimulation (Sangiorgi) is at the basis of several tools that automatically checks equivalence of pi terms.
  e.g. the Mobility Workbench (Victor)
- Can we extend this notion to the spi calculus?
Situation in the pi calculus

- Spi calculus is an extension of the pi calculus that incorporates cryptographic primitives.

\[ P, Q ::= 0 | E(x).P | \overline{E}(F).P | \phi P | (\nu x) P | P \parallel Q | P + Q | ! P \]

\[ M, N ::= x | (M \cdot N) | \text{Enc}^s_N M \]

\[ E, F ::= ... | \pi_1(E) | \pi_2(E) | \text{Dec}^s_F E \]

\[ \phi ::= [E = F] | [E : \mathcal{N}] \]

Open bisimulation (Sangiorgi) is at the basis of several tools that automatically checks equivalence of pi terms, e.g. the Mobility Workbench (Victor).

Can we extend this notion to the spi calculus?
Situation in the pi calculus

- Spi calculus is an extension of the pi calculus that incorporates cryptographic primitives.

\[
P, Q ::= 0 \mid a(x).P \mid \overline{a}(u).P \\
\quad \mid [a = b]P \mid (\nu x)P \\
\quad \mid P | Q \mid P + Q \mid !P
\]

- Open bisimulation (Sangiorgi) is at the basis of several tools that automatically checks equivalence of pi terms, e.g., the Mobility Workbench (Victor).

- Can we extend this notion to the spi calculus?
Situation in the pi calculus

- Spi calculus is an extension of the pi calculus that incorporates cryptographic primitives.

\[ P, Q ::= 0 \mid a(x).P \mid \overline{a}(u).P \mid \lfloor a = b \rfloor P \mid (\nu x) P \mid P \mid Q \mid P + Q \mid ! P \]

- Open bisimulation (Sangiorgi) is at the basis of several tools that automatically checks equivalence of pi terms e.g. the Mobility Workbench (Victor)

- Can we extend this notion to the spi calculus?
Bisimulations in the pi calculus
The main differences is the way they handle substitutions
Bisimulations in the pi calculus

The main differences is the way they handle substitutions

\[
P \xrightarrow{a(x)} P' \\
\downarrow \hspace{2cm} \downarrow \\
Q \xrightarrow{a(x)} Q'
\]

Ground
Bisimulations in the pi calculus

The main differences is the way they handle substitutions

\[ P \xrightarrow{a(x)} P' \quad P'\{u/x\} \]

\[ Q \xrightarrow{a(x)} Q' \quad Q'\{u/x\} \]

for any name \( u \)

Early/Late
Bisimulations in the pi calculus

The main differences is the way they handle substitutions

\[ \begin{align*}
P & \xrightarrow{\mu} P' \\
Q & \xrightarrow{\mu} Q'
\end{align*} \]

for any \( \sigma \)

Open
Bisimulations in the pi calculus

The main differences is the way they handle substitutions

\[ P \xrightarrow{\mu} P' \]
\[ Q \xrightarrow{\mu} Q' \]

for any \( \sigma \) that respects \( D \)

Open

Distinctions \( D \) to prevent from fusing previously extruded names with free names.
Bisimulations in the pi calculus

The main differences is the way they handle substitutions

\[
P \xrightarrow{\mu} P'
\]

\[
D
\]

\[
Q \xrightarrow{\mu} Q'
\]

for any \(\sigma\) that respects \(D\)

Open

The quantification over all substitutions gives a call-by-need flavor to the bisimulation. This idea is exploited by the tools which needs to inspect only most general unifiers.
Bisimulations in spi calculus

- Consider $P(M) := (\nu k) \overline{c} \langle \text{Enc}_k^s M \rangle . 0$. We want $P(M) \approx P(N)$ since $k$ is private and never revealed.
Bisimulations in spi calculus

Consider \( P(M) := (\nu k) \overline{c}\langle \text{Enc}^s_k M \rangle \cdot 0 \).
We want \( P(M) \approx P(N) \) since \( k \) is private and never revealed.

\[
\begin{array}{ccc}
P(M) & \leftrightarrow & P(N) \\
\overline{c} (\nu k) \text{Enc}_k^s M & \downarrow & 0 \\
0 & \leftrightarrow & 0
\end{array}
\]
Bisimulations in spi calculus

Consider $P(M) := (\nu k) \overline{c} \langle \text{Enc}_k^s M \rangle . 0$.
We want $P(M) \approx P(N)$ since $k$ is private and never revealed.

\[
\begin{array}{c}
P(M) \quad P(N) \\
\overline{c} (\nu k) \text{Enc}_k^s M \quad \overline{c} (\nu k) \text{Enc}_k^s N \\
0 \quad 0
\end{array}
\]
Bisimulations in spi calculus

- Consider \( P(M) := (\nu k) \overline{c}\langle \text{Enc}^s_k M \rangle. 0 \).
  We want \( P(M) \approx P(N) \) since \( k \) is private and never revealed.

\[
P(M) \quad \xrightarrow{\overline{c}(\nu k)\text{Enc}^s_k M} \quad 0
\]

\[
P(N) \quad \xrightarrow{\overline{c}(\nu k)\text{Enc}^s_k N} \quad 0
\]

- Bisimulations of the pi calculus are too fine-grained.
Bisimulations in spi calculus

- Consider $P(M) := (\nu k) \overline{c} \langle \text{Enc}_k^s M \rangle . 0$.
  We want $P(M) \approx P(N)$ since $k$ is private and never revealed.

- Bisimulations of the pi calculus are too fine-grained.
- Some pair of messages should be indistinguishable.
- Bisimulations are extended with a data structure that represents the observer knowledge. This has led to various notions of environment-sensitive bisimulations (framed, alley, hedged, ...).
Hedged bisimulation

Borgström and Nestmann.

A hedge $h \in H$ is a finite set of pairs of messages. Intuitively $(M, N) \in h$ means that $M$ and $N$ are indistinguishable.

A hedged bisimulation relates triples $(h, P, Q)$. 
Hedged bisimulation def. Borgström and Nestmann.

\[ P(M) := (\nu k) \overline{c}\langle \text{Enc}_k^s M \rangle . 0 \]

\[ P(M) \quad (c, c) \quad P(N) \]
Hedged bisimulation \textsuperscript{def.}
Borgström and Nestmann.

\[
P(M) := (\nu k) \overline{c} \langle \text{Enc}_k^s M \rangle . 0
\]

\[
\begin{array}{ccc}
P(M) & (c, c) & P(N) \\
\overline{c} (\nu k) \text{Enc}_k^s M & & \\
0 & & \\
\end{array}
\]
Hedged bisimulation \textsuperscript{def.}

Borgström and Nestmann.

\[
P(M) := (\nu k) \overline{c} \langle \text{Enc}_k^s M \rangle . \mathbf{0}
\]

\[
\begin{array}{ccc}
P(M) & (c, c) & P(N) \\
\overline{c} (\nu k) \text{Enc}_k^s M & \downarrow & \overline{c} (\nu k) \text{Enc}_k^s N \\
\mathbf{0} & \downarrow & \mathbf{0}
\end{array}
\]
Hedged bisimulation \textsuperscript{def.}
Borgström and Nestmann.

\[
P(M) := (\nu k) \overline{c} \langle \text{Enc}_k^s M \rangle . 0
\]

\[
\begin{align*}
P(M) & \quad (c, c) \quad P(N) \\
\overline{c} (\nu k) \text{Enc}_k^s M & \\
0 & \quad (\text{Enc}_k^s M, \text{Enc}_k^s N) & \\
\overline{c} (\nu k) \text{Enc}_k^s N & \\
0 & \quad 0
\end{align*}
\]
Hedged bisimulation $\text{def.}$

Borgström and Nestmann.

$$Q(M, N) := (\nu k) \overline{c}\langle \text{Enc}_k^s M \rangle . \overline{c}\langle \text{Enc}_k^s N \rangle . 0$$

$$Q(M, M) \quad (c, c) \quad Q(M, N)$$

$$\overline{c} (\nu k) \text{Enc}_k^s M \quad (\text{Enc}_k^s M, \text{Enc}_k^s M) \quad \overline{c} (\nu k) \text{Enc}_k^s M$$

$$\overline{c} \text{Enc}_k^s M \quad (\text{Enc}_k^s M, \text{Enc}_k^s N) \quad \overline{c} \text{Enc}_k^s N \quad 0$$
**Hedged bisimulation** def. Borgström and Nestmann.

\[ Q(M, N) := (\nu k) \overline{c} \langle \text{Enc}_k^s M \rangle . \overline{c} \langle \text{Enc}_k^s N \rangle . 0 \]

\[
\begin{array}{ccc}
Q(M, M) & (c, c) & Q(M, N) \\
\overline{c} (\nu k) \text{Enc}_k^s M & (\text{Enc}_k^s M, \text{Enc}_k^s M) & \overline{c} (\nu k) \text{Enc}_k^s M \\
\overline{c} \text{Enc}_k^s M & (\text{Enc}_k^s M, \text{Enc}_k^s N) & \overline{c} \text{Enc}_k^s N \\
0 & (\text{Enc}_k^s M, \text{Enc}_k^s N) & 0
\end{array}
\]
Hedged bisimulation \(\text{def.}\)
Borgström and Nestmann.

\[ Q(M, N) := (\nu k) \overline{c}\langle Enc_k^s M \rangle. \overline{c}\langle Enc_k^s N \rangle. 0 \]

\[ Q(M, M) \quad (c, c) \quad Q(M, N) \]

\[ \overline{c} (\nu k) Enc_k^s M \]
\[ \overline{c} Enc_k^s M \]
\[ 0 \]

\[ \overline{c} (\nu k) Enc_k^s M \]
\[ \overline{c} Enc_k^s N \]
\[ 0 \]

The hedge must be consistent \(\text{def.}\).

\[ O := c(x).c(y).[x = y]\overline{c}\langle \text{fail} \rangle. 0 \]
Hedged bisimulation \text{def.}
Borgström and Nestmann.

$$R(M) := (\nu k) \overline{c} \langle \text{Enc}^s_k M \rangle . \overline{c} \langle k \rangle . 0$$

\[
\begin{array}{c c c}
R(M) & (c, c) & R(N) \\
\overline{c} (\nu k) \text{Enc}^s_k M & (\text{Enc}^s_k M, \text{Enc}^s_k N) & \overline{c} (\nu k) \text{Enc}^s_k N \\
\overline{c} k & (k, k) & \overline{c} k \\
0 & 0 & 0
\end{array}
\]
**Hedged bisimulation** \[\text{def.}\]

Borgström and Nestmann.

\[ R(M) := (\nu k) \overline{c} \langle \text{Enc}^s_k M \rangle. \overline{c} \langle k \rangle. 0 \]

The hedge is **analysed** after outputs \[\text{def.}\].
Hedged bisimulation \( \text{def.} \)

Borgström and Nestmann.

\[
\begin{align*}
S_1(M) &:= (\nu k) \overline{c} \langle \text{Enc}^s_k M \rangle . c(x) . [x = k] \overline{c} \langle k \rangle . 0 \\
S_2(M) &:= (\nu k) \overline{c} \langle \text{Enc}^s_k M \rangle . c(x) . 0
\end{align*}
\]
Hedged bisimulation \[ \text{def.} \]
Borgström and Nestmann.

\[ S_1(M) := (\nu k) \overline{c}\langle \text{Enc}_k^s M \rangle. c(x). [x = k] \overline{c}\langle k \rangle. 0 \]
\[ S_2(M) := (\nu k) \overline{c}\langle \text{Enc}_k^s M \rangle. c(x). 0 \]

The possible pairs of input messages are constructed using the current knowledge and possibly some fresh names \[ \text{def.} \].
Open hedged bisimulation

Delaying instantiation of input variables

- Which names are subjects to substitutions?
  - Input variables.

- What are the possible objects of substitutions?
  - Messages constructed using the knowledge available at the moment of the input and possibly some fresh names.

- A variable dynamically typed as a name is not replaced by a compound message.
Open hedged bisimulation

Def. Delaying instantiation of input variables

- Which names are subjects to substitutions?
  - Input variables.

- What are the possible objects of substitutions?
  - Messages constructed using the knowledge available at the moment of the input and possibly some fresh names.

- A variable dynamically typed as a name is not replaced by a compound message.

Hence the form of S-environments $se = (h, v, \preceq, (\gamma_l, \gamma_r))$.

A S-environment is consistent if for any instantiation of input variables, the resulting hedge is consistent.
Open hedged bisimulation\footnote{\texttt{def.}}

Delaying instantiation of input variables

- Which names are subjects to substitutions?
  - Input variables.

- What are the possible objects of substitutions?
  - Messages constructed using the knowledge available at the moment of the input and possibly some fresh names.

- A variable \textit{dynamically typed} as a name is not replaced by a compound message\footnote{\texttt{LTS}}.
Open hedged bisimulation

Delaying instantiation of input variables

- Which names are subjects to substitutions?
  - Input variables.

- What are the possible objects of substitutions?
  - Messages constructed using the knowledge available at the moment of the input and possibly some fresh names.

- A variable dynamically typed as a name is not replaced by a compound message.

Hence the form of S-environments $se = (h, v, \prec, (\gamma_l, \gamma_r))$. 
Open hedged bisimulation \textsuperscript{def.}

Delaying instantiation of input variables

- Which names are subjects to substitutions?
  - Input variables.

- What are the possible objects of substitutions?
  - Messages constructed using the knowledge available at the moment of the input and possibly some fresh names.

- A variable dynamically typed as a name is not replaced by a compound message \textsuperscript{LTS}.

Hence the form of S-environments $se = (h, v, \preceq, (\gamma_l, \gamma_r))$.

### consistency of S-environments

A S-environment is consistent if for any instantiation of input variables, the resulting hedge is consistent.
Symbolic characterisation

- Relies on the definition of a *symbolic LTS* \(^\text{def.}\).
- The idea is to record —without checking— the conditions needed to enable a transition.
  \[
P \xrightarrow{\mu}^\Phi P'
\]
- The symbolic LTS helps to characterise precisely the set of substitutions \(\sigma\) such that \(P_\sigma \xrightarrow{\mu} P'\).
- Given a symbolic transition \(P \xrightarrow{\mu}^\Phi P'\), there is a finite complete set of solutions of \(\Phi\).
Outline

1. From protocol narrations to spi calculus
2. An open variant of bisimulation for the spi calculus
3. A formalization in Coq
Representation of binders

de Bruijn indices

\[
\text{Representation of } a(x).[\text{Dec}^s_k x : M](\nu l) \overline{b}\langle l \rangle . 0
\]

\[
\begin{array}{c}
z y x \cdots l k j \cdots c b a \\
0 \lambda. [\text{Dec}^s_1 l : M] \nu 3\langle 0 \rangle . 0
\end{array}
\]
Representation of binders

de Bruijn indices

Several operations have to be defined to handle de Bruijn indices.

**Example:** $\text{lift}_d(k, t)$ makes room for $k$ new binders in $t$

\[
\begin{align*}
\lambda. \overline{1}_0 \cdot 0 \\
\lambda. (\overline{1}_0 \cdot 0 | 1 \lambda. \overline{0}_{25} \cdot 0 )
\end{align*}
\]
Representation of binders

de Bruijn indices

In practise:

- 5 operations on indices, 6 types (names, messages, ...)
- about 60 useful facts relate these operations
- not scalable and tedious to define and prove several times the same operations/facts
Representation of binders

de Bruijn indices

Instead

1. define on names
2. lift to other types
Abstracting the labelled transition system

- There are several LTS to define.
- Some properties are shared (e.g. structural congruence preserves the transitions)
- These LTS all follow the same pattern.
- Instead of defining each LTS separately, we make a functor and thus defer the definition of the semantics to the definitions of the semantics of actions.
Abstracting the labelled transition system

We rely on a set of actions $\mathcal{A}$ and several functions to manipulate them:

- $\text{mkSil} : \mathcal{A}$ (silent)
- $\text{mkInp} : E \rightarrow \mathcal{A} \cup \{\bot\}$ (input)
- $\text{mkOutp} : E \times E \rightarrow (\mathcal{A} \times E) \cup \{\bot\}$ (output)
- $\text{mkRes} : \mathcal{A} \rightarrow \mathcal{A} \cup \{\bot\}$ (restriction)
- $\text{mkIf} : F \times \mathcal{A} \rightarrow \mathcal{A} \cup \{\bot\}$ (guard)
- $\text{mkInt} : \mathcal{A} \times \mathcal{A} \rightarrow \mathcal{A} \cup \{\bot\}$ (interact)
Abstracting the labelled transition system

We then define a parametrised LTS.

\[ \text{INPUT} \quad \frac{}{\text{mkInp}(E) = \alpha \in A} \quad E \lambda.P \xrightarrow{\alpha} \lambda.P \]

\[ \text{OUTPUT} \quad \frac{}{\text{mkOutp}(E, F) = (\alpha, M) \in A \times E} \quad \overline{E}\langle F\rangle.P \xrightarrow{\alpha} \langle M\rangle P \]

\[ \text{CLOSE-L} \quad \frac{}{P \xrightarrow{\alpha} F \quad Q \xrightarrow{\beta} C \quad \text{mkInt}(\alpha, \beta) = \gamma \in A} \quad P \parallel Q \xrightarrow{\gamma} F \parallel C \]
Overview of the formalization

- Monadic pi calculus
- Pi LTS
- Spi calculus
- Hedges and their properties
- Spi LTS: standard, with type constraints, symbolic and their properties
- Crash test: result about structural congruence
- Late hedged bisimulation, correctness of up-to techniques
- Small examples of bisimulations
Conclusion

- A formal semantics for protocol narrations.
  - A rigorous translation into spi calculus.
- An open style definition of bisimulation for the spi calculus.
  - It is a sound proof technique.
  - It is an extension of open bisimulation of the pi calculus.
  - Its projection down to the pi calculus has enabled us to better understand the original notion of open bisimulation.
  - A symbolic characterisation as a promising first step towards mechanisation.
- A formalization in a proof assistant.
  - Very useful while elaborating the theory.
  - Already a framework to reason formally about cryptographic protocols in Coq.
Future work

- Study furthermore open hedged bisimilarity.
  - Congruence properties.
  - Mechanisation.

- Complete the formalization in Coq.
  - Realise the dream of having a correct-by-construction equivalence checker for the spi calculus.
  - Define smart tactics for reasoning directly in Coq (e.g. interface with the tool that handles the decidable fragment)
Future work

- Study furthermore open hedged bisimilarity.
  - Congruence properties.
  - Mechanisation.

- Complete the formalization in Coq.
  - Realise the dream of having a correct-by-construction equivalence checker for the spi calculus.
  - Define smart tactics for reasoning directly in Coq (e.g. interface with the tool that handles the decidable fragment)

- Demos?
The end.
The spi calculus

Syntax

- Countably infinite set of *names*. Communication channels, nonces, atomic data, ...
- Messages

\[ M, N ::= x | (M \cdot N) | \text{Enc}^s_N M \]

- Expressions

\[ E, F ::= x | (E \cdot F) | \text{Enc}^s_F E \]
\[ \quad \quad | \quad \pi_1(E) | \pi_2(E) | \text{Dec}^s_F E \]

- Guards

\[ \phi ::= [E = F] | [E: \mathcal{N}] \]
Processes

\[ P, Q ::= 0 \mid E(x).P \mid \overline{E}\langle F\rangle.P \\]
\[ \quad \mid \phi P \mid (\nu x)P \\]
\[ \quad \mid P \parallel Q \mid P + Q \mid ! P \\]

Agents

\[ A ::= P \\]
\[ \quad \mid (x)P \\]
\[ \quad \mid (\nu \tilde{z})\langle M\rangle P \quad \text{where } \{\tilde{z}\} \subseteq n(M) \]
Labelled transitions system

Late semantics

\[ \text{INPUT} \quad e_c(E) = a \in \mathcal{N} \quad \frac{E(x).P \xrightarrow{a} (x)P}{P} \]

\[ \text{OUTPUT} \quad e_c(E) = a \in \mathcal{N} \quad e_c(F) = M \in \mathcal{M} \quad \frac{E\langle F \rangle.P \xrightarrow{\bar{a}} \langle M \rangle P}{P} \]

\[ \text{CLOSE-L} \quad \frac{P \xrightarrow{a} F \quad Q \xrightarrow{\bar{a}} C}{P \mid Q \xrightarrow{\tau} F \cdot C} \]

\[ \text{IF THEN} \quad \frac{P \xrightarrow{\mu} P' \quad e(\phi) = \text{true}}{\phi P \xrightarrow{\mu} P'} \]

\[ \text{RES} \quad \frac{P \xrightarrow{\mu} A}{(\nu z) P \xrightarrow{\mu} (\nu z) A} \quad z \notin n(\mu) \]

\[ \text{PAR-L} \quad \frac{P \xrightarrow{\mu} A}{P \mid Q \xrightarrow{\mu} A \mid Q} \]

+ SUM, REP- et ALPHA.
Evaluation of expressions and guards

Expressions:

\[ \begin{align*}
    e_c(a) & := a \\
    e_c(\text{Enc}_F^sE) & := \text{Enc}_N^sM \quad \text{if } e_c(E) = M \in M \\
    e_c((E_1 \cdot E_2)) & := (M_1 \cdot M_2) \quad \text{if } e_c(E_1) = M_1 \in M \\
    e_c(\text{Dec}_F^sE) & := M \quad \text{if } e_c(E) = \text{Enc}_N^sM \in M \\
    e_c(\pi_1(E)) & := M_1 \quad \text{if } e_c(E) = (M_1 \cdot M_2) \in M \\
    e_c(\pi_2(E)) & := M_2 \quad \text{if } e_c(E) = (M_1 \cdot M_2) \in M \\
    e_c(E) & := \bot \quad \text{otherwise}
\end{align*} \]

 Guards:

\[ \begin{align*}
    e([E = F]) & := \text{true} \quad \text{si } e_c(E) = e_c(F) = M \in M \\
    e([E : \mathcal{N}]) & := \text{true} \quad \text{si } e_c(E) = a \in \mathcal{N} \\
    e(\phi) & := \text{false} \quad \text{otherwise}
\end{align*} \]
Late hedged bisimulation

A symmetric consistent hedged relation $\mathcal{R}$ is a (strong) late hedged bisimulation if whenever $(h, P, Q) \in \mathcal{R}$, we have that

1. if $P \xrightarrow{\tau} P'$ then
   there exists $Q'$ such that $Q \xrightarrow{\tau} Q'$ and $(h, P', Q') \in \mathcal{R}$

2. if $P \xrightarrow{a} (x)P'$ (with $x \notin n(\pi_1(h))$)
   and $(a, b) \in h$ then
   there exist $y$ and $Q'$ such that $Q \xrightarrow{b} (y)Q'$ (with $y \notin n(\pi_2(h))$)
   and for all $B$ and $(M, N)$ such that $h \vdash_B (M, N)$
   we have $(h \cup B, P'\{M/x\}, Q'\{N/y\}) \in \mathcal{R}$.

3. if $P \xrightarrow{\bar{a}} (\nu\tilde{c})\langle M \rangle P'$ (with $\{\tilde{c}\} \cap n(\pi_1(h)) = \emptyset$)
   and $(a, b) \in h$ then
   there exist $\tilde{d}$, $Q'$ and $N$ such that $Q \xrightarrow{\bar{b}} (\nu\tilde{d})\langle N \rangle Q'$
   (with $\{\tilde{d}\} \cap n(\pi_2(h)) = \emptyset$)
   and $(I(h \cup \{(M, N)\}), P', Q') \in \mathcal{R}$. 
Synthesis of a hedge

The synthesis $S(h)$ is the smallest set that satisfies

\[
\begin{align*}
\text{SYN-INC} & \quad \frac{(M, N) \in h}{(M, N) \in S(h)} \\
\text{SYN-ENC-S} & \quad \frac{(M_1, N_1) \in S(h) \quad (M_2, N_2) \in S(h)}{(\text{Enc}^s_{M_2} M_1, \text{Enc}^s_{N_2} N_1) \in S(h)} \\
\text{SYN-PAIR} & \quad \frac{(M_1, N_1) \in S(h) \quad (M_2, N_2) \in S(h)}{((M_1 \cdot M_2), (N_1 \cdot N_2)) \in S(h)}
\end{align*}
\]
Possible inputs

Let $h \in H$, $(M, N) \in M \times M$

Let $B \subseteq \mathcal{N} \times \mathcal{N}$ a consistent hedge such that

- $\pi_1(B) \cap n(\pi_1(h)) = \emptyset$
- $\pi_2(B) \cap n(\pi_2(h)) = \emptyset$

i.e. the names of $B$ are fresh component-wise w.r.t. those of $h$.

We write $h \vdash_B (M, N)$ if

- $\forall (b_1, b_2) \in B : b_1 \in n(M) \lor b_2 \in n(N)$
- $(M, N) \in S(h \cup B)$
The analysis $\mathcal{A}(h)$ is the smallest hedge that is closed by $\text{analz}(\cdot)$.

\[
\begin{align*}
\text{ANA-INC:} & \quad \frac{(M, N) \in h}{(M, N) \in \text{analz}(h)} \\
\text{ANA-DEC-S:} & \quad \frac{(\text{Enc}_{M_2}^s M_1, \text{Enc}_{N_2}^s N_1) \in \text{analz}(h) \quad (M_2, N_2) \in S(h)}{(M_1, N_1) \in \text{analz}(h)} \\
\text{ANA-FST:} & \quad \frac{((M_1, M_2), (N_1, N_2)) \in \text{analz}(h)}{(M_1, N_1) \in \text{analz}(h)} \\
\text{ANA-SND:} & \quad \frac{((M_1, M_2), (N_1, N_2)) \in \text{analz}(h)}{(M_2, N_2) \in \text{analz}(h)}
\end{align*}
\]
## Irreducibles

$I(h)$ is the smallest hedge such that $S(I(h)) = S(A(h))$.

## Definition

A hedge $h$ is irreducible iff $I(h) = h$. 
Consistency of a hedge

Consistency

A hedge $h$ is consistent iff:
Whenever $(M, N) \in h$
- $M \in \mathcal{N} \iff N \in \mathcal{N}$
- whenever $(M', N') \in h : M = M' \iff N = N'$
- $M \neq (M_1 . M_2)$ and $N \neq (N_1 . N_2)$
- if $M = \text{Enc}_{M_2}^s M_1$ then $(M_2, N_2) \not\in S(h)$
- if $N = \text{Enc}_{N_2}^s N_1$ then $(M_2, N_2) \not\in S(h)$

Lemma

A consistent hedge is irreducible.
**Definition (S-environment)**

A S-environment is a quadruple \( se = (h, v, \triangleleft, (\gamma_l, \gamma_r)) \) where \( h \in H \), \( v \subseteq \mathcal{N} \times \mathcal{N} \) is a consistent hedge, \( \triangleleft \subseteq h \times v \), \( \gamma_l \subseteq \pi_1(v) \) and \( \gamma_r \subseteq \pi_2(v) \).

**Hedge available**

The *hedge available* to \((x, y) \in v\) according to \(\triangleleft\) is defined by

\[
se \mid_{(x,y)} := \{(M, N) \in h \mid (M, N) \triangleleft (x, y)\}.
\]

**Concrete hedge**

The *concrete hedge* of \( se \) is \( \mathcal{H}(se) := h \cup v \).
Respectful substitutions

Definition (Respectful substitutions)

Let \((\sigma, \rho)\) be a pair of substitutions, \(B \subseteq \mathcal{N} \times \mathcal{N}\) a consistent hedge and \(se = (h, v, \prec, (\gamma_l, \gamma_r))\) a S-environment. We say that \((\sigma, \rho)\) respects \(se\) with \(B\) — written \((\sigma, \rho) \triangleright_B se\) — if

1. \(\text{supp}(\sigma) \subseteq \pi_1(v)\)
2. \(\text{supp}(\rho) \subseteq \pi_2(v)\)
3. \(\forall (b_1, b_2) \in B : b_1 \in n(\sigma(\pi_1(v))) \lor b_2 \in n(\rho(\pi_2(v)))\)
4. \(\pi_1(B) \cap (n(\pi_1(h)) \setminus \pi_1(v)) = \emptyset\)
5. \(\pi_2(B) \cap (n(\pi_2(h)) \setminus \pi_2(v)) = \emptyset\)
6. \(\forall (x, y) \in v : (x\sigma, y\rho) \in S(\mathcal{I}(se|_{(x,y)}(\sigma, \rho) \cup B))\)
7. \(\forall x \in \gamma_l : x\sigma \in \mathcal{N}\)
8. \(\forall y \in \gamma_r : y\rho \in \mathcal{N}\)
Open hedged bisimulation

A symmetric consistent open hedged relation $\mathcal{R}$ is an *open hedged bisimulation* if for all $(se, P, Q) \in \mathcal{R}$, for all $\sigma, \rho$ and $B$ such that $(\sigma, \rho) \triangleright_B se$,

**internal communications**

If $P \sigma \xleftarrow{\tau} S_1 P'$ then there exist $Q'$ and $S_2$ such that $Q \rho \xrightarrow{\tau} S_2 Q'$ and $(se^{(\sigma, \rho)}_B + c(S_1, S_2), P', Q') \in \mathcal{R}$
Open hedged bisimulation

A symmetric consistent open hedged relation $\mathcal{R}$ is an open hedged bisimulation if for all $(se, P, Q) \in \mathcal{R}$, for all $\sigma, \rho$ and $B$ such that $(\sigma, \rho) \triangleright_B se$,

**inputs**

if $P_\sigma \xrightarrow{a} (x)P'$ (with $x \notin n(\pi_1(se_B^{(\sigma,\rho)})))$

and $(a, b) \in S(I(se_B^{(\sigma,\rho)})))$ then

there exist $y$, $Q'$ and $S_2$ such that $Q_\rho \xleftarrow{b} (y)Q'$ (with

$y \notin n(\pi_2(se_B^{(\sigma,\rho)})))$

and $(se_B^{(\sigma,\rho)} +_i (x, y) +_c (S_1, S_2), P', Q') \in \mathcal{R}$
A symmetric consistent open hedged relation $\mathcal{R}$ is an *open hedged bisimulation* if for all $(se, P, Q) \in \mathcal{R}$, for all $\sigma, \rho$ and $B$ such that $(\sigma, \rho) \triangleright_B se$,

**outputs**

if $P\sigma \xrightarrow{\overline{a}} (\nu \tilde{c}) \langle M \rangle P'$ (with $\{\tilde{c}\} \cap n(\pi_1(\mathcal{H}(se_B^{(\sigma, \rho)}))) = \emptyset$)

and $(a, b) \in S(\mathcal{I}(\mathcal{H}(se_B^{(\sigma, \rho)})))$ then

there exist $\tilde{d}, N, Q'$ and $S_2$ such that $Q\rho \xrightarrow{\overline{b}} (\nu \tilde{d}) \langle N \rangle Q'$

(with $\{\tilde{d}\} \cap n(\pi_2(\mathcal{H}(se_B^{(\sigma, \rho)}))) = \emptyset$)

and $(se_B^{(\sigma, \rho)})_o(M, N) +_c(S_1, S_2), P', Q') \in \mathcal{R}$
A LTS that collects type constraints

\[
\begin{align*}
\text{NC-SILENT} & \quad \tau. P \xrightarrow{\tau} P \\
\text{NC-INPUT} & \quad \text{ec}(E) = a \in \mathcal{N} \\
& \quad E(x). P \xrightarrow{a} (x)P \\
\text{NC-OUTPUT} & \quad \text{ec}(E) = a \in \mathcal{N} \\
& \quad E(F). P \xrightarrow{a} \langle M \rangle P \\
\text{NC-IFTHEN} & \quad \phi \xrightarrow{\mu} A \\
& \quad \phi P \xrightarrow{\mu} A \\
& \quad S = \text{nc}(\phi) \\
\end{align*}
\]

where \( \text{nc}([E:\mathcal{N}]) := \{\text{ec}(E)\} \) and \( \text{nc}([E=F]) := \emptyset \).
The two semantics are equivalent:

1. If $P \xrightarrow{\mu} A$ there exists $S \subseteq \mathcal{N}$ such that $P \xrightarrow{\mu} S \xrightarrow{} A$.

2. If $P \xrightarrow{\mu} S \xrightarrow{} A$ then $P \xrightarrow{\mu} A$.

Lemma

If $P \xrightarrow{\mu} S \xrightarrow{} A$ and $\sigma : \mathcal{N} \rightarrow \mathcal{M}$ is a substitution such that $S\sigma \subseteq \mathcal{N}$ then $P\sigma \xrightarrow{\mu\sigma} S\sigma \xrightarrow{} A\sigma$. 
A symbolic LTS

**S-GUARD**

\[ P \xrightarrow{\mu} A \]
\[ \phi P \xrightarrow{\mu} A \]

**S-INPUT**

\[ E(x).P \xrightarrow{e_a(E)} (x)P \]
\[ \{ E : N \} \]

**S-OUTPUT**

\[ \overline{E}\langle F\rangle.P \xrightarrow{e_a(E)} \langle e_a(F)\rangle P \]
\[ \{ E : N , [F : M] \} \]

**S-CLOSE-L**

\[ P \xrightarrow{c_1} F \]
\[ Q \xrightarrow{c_2} C \]
\[ P | Q \xrightarrow{\tau} F \cdot C \]
\[ \{ E = E' \} \& c_1 \& c_2 \]

**S-RES**

\[ (\nu z)P \xrightarrow{\mu} (\nu z)A \]
\[ \nu_+(z,c) \]
\[ z \notin n(\mu) \]
A transition constraint has the form $(\nu \tilde{z}) \Phi$ where $\Phi$ is a finite set of guards and $\tilde{z}$ is a finite set of names that occur in $\Phi$, i.e. $\{\tilde{z}\} \subseteq \text{n}(\Phi)$.

Composition of constraints:

- Conjunction of $c_1 = (\nu \tilde{z}_1) \Phi_1$ and $c_2 = (\nu \tilde{z}_2) \Phi_2$ where $\{\tilde{z}_1\} \cap \{\tilde{z}_2\} = \emptyset$, $\{\tilde{z}_1\} \cap \text{fn}(c_2) = \emptyset$ and $\{\tilde{z}_2\} \cap \text{fn}(c_1) = \emptyset$.

$$c_1 \& c_2 := (\nu \tilde{z}_1 \tilde{z}_2) (\Phi_1 \cup \Phi_2)$$

- Restriction of name $x$.
  If $c = (\nu \tilde{z}) \Phi$ and $x \notin \{\tilde{z}\}$:
  $$\nu_+ (x, c) := (\nu x \tilde{z}) \Phi \quad \text{if } x \in \text{fn}(c)$$
  $$:= c \quad \text{otherwise}$$
Abstract evaluation of expressions:

\[
\begin{align*}
\mathbf{e}_a(a) & := a & \text{if } a \in \mathcal{N} \\
\mathbf{e}_a(\text{Enc}_F^s E) & := \text{Enc}_{\mathbf{e}_a(F)}^s \mathbf{e}_a(E) \\
\mathbf{e}_a((E \cdot F)) & := (\mathbf{e}_a(E) \cdot \mathbf{e}_a(F)) \\
\mathbf{e}_a(\text{Dec}_F^s E) & := \begin{cases} E_1 & \text{if } \mathbf{e}_a(E) = \text{Enc}_{E_2}^s E_1 \\ \text{Dec}_{\mathbf{e}_a(F)}^s \mathbf{e}_a(E) & \text{otherwise} \end{cases} \\
\mathbf{e}_a(\pi_1(E)) & := \begin{cases} E_1 & \text{if } \mathbf{e}_a(E) = (E_1 \cdot E_2) \\ \pi_1(\mathbf{e}_a(E)) & \text{otherwise} \end{cases} \\
\mathbf{e}_a(\pi_2(E)) & := \begin{cases} E_2 & \text{if } \mathbf{e}_a(E) = (E_1 \cdot E_2) \\ \pi_2(\mathbf{e}_a(E)) & \text{otherwise} \end{cases}
\end{align*}
\]
Properties

Define $\succ_o$ as being the smallest precongruence on expressions that satisfies:

- $\pi_1 ((E_1 \cdot E_2)) \succ_o E_1$ if $e_c(E_2) \neq \bot$
- $\pi_2 ((E_1 \cdot E_2)) \succ_o E_2$ if $e_c(E_1) \neq \bot$
- $\text{Dec}_{E_2}^s \text{Enc}_{E_2}^s E_1 \succ_o E_1$ if $e_c(E_2) \neq \bot$

Extend this relation to agents in:

- $A \succ_o^= B$ ($A, B$ are concrete agents)
- $A \succ_o^e B$ ($A$ is symbolic, $B$ is concrete)

(two ways to handle concretions)
Appendix

Properties

continued

Theorem

Let $P, Q \in \mathcal{P}$ and assume that $P >_o Q$.

1. If $P \xrightarrow{\mu} S$ then $Q \xrightarrow{\mu} A$ and $A >_o B$

2. If $Q \xrightarrow{\mu} B$ then $P \xrightarrow{\mu} A$ and $A >_o B$

Theorem

Let $P, Q \in \mathcal{P}$ and $\sigma : \mathcal{N} \rightarrow \mathcal{M}$ a substitution.

1. If $P \xrightarrow{\mu_S} c$ then $A$ and $e(c\sigma) = \text{true}$ then $P_\sigma \xrightarrow{e_c(\mu_S\sigma)} B$ with $A_\sigma >_e B$

2. If $P_\sigma \xrightarrow{\mu} c$ then $P \xrightarrow{\mu_S} A$ with $e(c\sigma) = \text{true}$, $\text{nc}(c\sigma) = S$, $e_c(\mu_S\sigma) = \mu$ and $A_\sigma >_e B$
Operations on de Bruijn indices

- Parametrised by the binding depth \( d \)
- \( \text{mem}_d(i, t) \) returns \( \text{true} \) iff \( i \) is free in \( t \)
- \( \text{lift}_d(k, t) \) makes room for \( k \) new binders in \( t \)

Used in parallel composition of an agent and a process:

\[
(\lambda . P) \mid Q := \lambda . (P \mid \text{lift}_0(1, Q))
\]

\[
(\nu^k \langle F \rangle P) \mid Q := \nu^k \langle F \rangle (P \mid \text{lift}_0(k, Q))
\]

For instance:

\[
\lambda . 1 \langle 0 \rangle . 0 \quad 0 \lambda . \bar{0} \langle 24 \rangle . 0
\]

\[
\lambda . (1 \langle 0 \rangle . 0 \mid 1 \lambda . \bar{0} \langle 25 \rangle . 0)
\]
Operations on de Bruijn indices

- \( \text{swap}_d(k, t) \) makes a circular permutation of the \( k \) first indices in \( t \)
- \( \text{low}_d(t) \) removes the first index
- Used in restriction of an agent:

\[
\nu(\lambda.P) := \lambda.\nu \text{swap}_0(1, P) \\
\nu(\nu^k\langle F \rangle P) := \nu^{k+1}\langle F \rangle P \\
:= \nu^k\langle \text{low}_k(F) \rangle \nu \text{swap}_0(k, P)
\]

if \( \text{mem}_k(0, F) = \text{true} \) otherwise

- \( \text{Isubst}_d(k, \overline{E}, t) \) substitutes the \( |\overline{E}| \) first indices with the corresponding expression of \( \overline{E} \) in \( t \). The \( k \) first indices are bound in \( \overline{E} \).

\[
(\lambda.P) \bullet (\nu^k\langle F \rangle Q) := \nu^k(\text{Isubst}_0(k, F, P) \mid Q)
\]