MPRI

SECURE: Proofs of Security Protocols

3. Correspondance Properties

Adrien Koutsos, Inria Paris

2026/2027

Outline

Example of a Security Proof

Authentication Protocols

- Execution Traces
- Macro Terms
- Local Proof System
- Cryptographic Rule: Collision Resistance
- Cryptographic Rule: Message Authentication Code
- Authentication of the Hash-Lock Protocol

Going Further

Example of a Security Proof

Protocol Branching

We consider a better version of PA in which S checks whether it is talking to I or not.

The PA Protocol, v2

1. $\mathsf{I} \to \mathsf{S}: \quad \nu\, \mathsf{n_I}. \qquad \mathbf{out}(\mathtt{I}, \{\langle \mathsf{pk_I}, \mathsf{n_I}\rangle\}_{\mathsf{pk_S}})$
2. $\mathsf{S} \to \mathsf{I}: \quad \nu\, \mathsf{n_S}.\, \mathbf{in}(\mathtt{S}, x).\, \mathbf{out}(\mathtt{S}, \text{if } \pi_1(d) = \mathsf{pk_I} \wedge \mathsf{len}(\pi_2(d)) = \mathsf{len}(\mathsf{n_S}))$
   $\text{then } \{\langle \pi_2(d), \mathsf{n_S}\rangle\}_{\mathsf{pk_I}}$
   $\text{else } \{\langle \mathsf{n_S}, \mathsf{n_S}\rangle\}_{\mathsf{pk_I}}$

where $d \equiv \mathsf{dec}(\mathsf{x}, \mathsf{sk_S})$.

*We hide which branck was taken using a **decoy** message in the else branch.*

Private Authentication: Anonymity

Lets now try to prove that this protocol provides **anonymity**:

- I_X is the initiator with identity X;
- S_X is the server, accepting messages from X.

The adversary must not be able to distinguish $\mathsf{I}_A \mid \mathsf{S}_A$ from $\mathsf{I}_C \mid \mathsf{S}_A$.

$$
\begin{aligned}
\mathsf{I}_X &: \nu\, r.\ \ \nu\, n_I. && \mathbf{out}(\mathtt{I}, \{\langle pk_X\,,\, n_I\rangle\}^{r}_{pk_S}) \\
\mathsf{S}_X &: \nu\, r_0.\, \nu\, n_S.\, \mathbf{in}(\mathtt{S}, x).\ && \mathbf{out}(\mathtt{S}, \text{if } \pi_1(d) = pk_X \wedge \mathsf{len}(\pi_2(d)) = \mathsf{len}(n_S)) \\
& && \qquad \text{then } \{\langle \pi_2(d)\,,\, n_S\rangle\}^{r_0}_{pk_X} \\
& && \qquad \text{else } \ \{\langle n_S\,,\, n_S\rangle\}^{r_0}_{pk_X}
\end{aligned}
$$

We assume the encryption is IND-CCA$_1$ and KP-CCA$_1$.

Private Authentication: Anonymity

$$\mathsf{I}_X \;: \nu\, \mathsf{r}.\;\; \nu\, \mathsf{n}_\mathsf{I}. \qquad\qquad \mathbf{out}(\mathtt{I}, \{\langle \mathsf{pk}_X\,,\, \mathsf{n}_\mathsf{I}\rangle\}^{\mathsf{r}}_{\mathsf{pk}_\mathbf{S}})$$

$$\mathsf{S}_X : \nu\, \mathsf{r}_0.\, \nu\, \mathsf{n}_\mathsf{S}.\, \mathbf{in}(\mathtt{S}, x).\, \mathbf{out}(\mathtt{S}, \text{if } \pi_1(d) = \mathsf{pk}_X \wedge \mathsf{len}(\pi_2(d)) = \mathsf{len}(\mathsf{n}_\mathsf{S}))$$
$$\text{then } \{\langle \pi_2(d)\,,\, \mathsf{n}_\mathsf{S}\rangle\}^{\mathsf{r}_0}_{\mathsf{pk}_\mathbf{X}}$$
$$\text{else } \{\langle \mathsf{n}_\mathsf{S}\,,\, \mathsf{n}_\mathsf{S}\rangle\}^{\mathsf{r}_0}_{\mathsf{pk}_\mathbf{X}}$$

To prove $\mathsf{I}_A \mid \mathsf{S}_A \approx \mathsf{I}_C \mid \mathsf{S}_A$, we have several **traces**:

$$\mathbf{in}(\mathtt{S}), \mathbf{out}(\mathtt{I}), \mathbf{out}(\mathtt{S}) \qquad \mathbf{in}(\mathtt{S}), \mathbf{out}(\mathtt{S}), \mathbf{out}(\mathtt{I}) \qquad \mathbf{out}(\mathtt{I}), \mathbf{in}(\mathtt{S}), \mathbf{out}(\mathtt{S})$$

Private Authentication: Anonymity

$$\mathsf{I}_X : \nu\, r.\ \nu\, n_I. \qquad \mathbf{out}(\mathtt{I}, \{\langle \mathsf{pk}_X\,,\, n_I\rangle\}^{r}_{\mathsf{pk}_S})$$

$$\mathsf{S}_X : \nu\, r_0.\nu\, n_S.\, \mathbf{in}(\mathtt{S}, x).\ \mathbf{out}(\mathtt{S}, \text{if } \pi_1(d) = \mathsf{pk}_X \wedge \mathsf{len}(\pi_2(d)) = \mathsf{len}(n_S))$$
$$\text{then } \{\langle \pi_2(d)\,,\, n_S\rangle\}^{r_0}_{\mathsf{pk}_X}$$
$$\text{else } \{\langle n_S\,,\, n_S\rangle\}^{r_0}_{\mathsf{pk}_X}$$

To prove $\mathsf{I}_A \mid \mathsf{S}_A \approx \mathsf{I}_C \mid \mathsf{S}_A$, we have several **traces**:

$$\mathbf{in}(\mathtt{S}), \mathbf{out}(\mathtt{I}), \mathbf{out}(\mathtt{S}) \qquad \mathbf{in}(\mathtt{S}), \mathbf{out}(\mathtt{S}), \mathbf{out}(\mathtt{I}) \qquad \mathbf{out}(\mathtt{I}), \mathbf{in}(\mathtt{S}), \mathbf{out}(\mathtt{S})$$

But there is a **more general trace**: its security implies the security of the other traces.
See **partial order reduction** (POR) techniques [1].

Private Authentication: Anonymity

$$\mathsf{I}_X \;:\; \nu\, r.\;\; \nu\, n_I. \qquad\qquad \mathbf{out}(\mathtt{I}, \{\langle \mathsf{pk}_X\,,\; n_I\rangle\}^{r}_{\mathsf{pk_S}})$$

$$\mathsf{S}_X : \nu\, r_0.\, \nu\, n_S.\, \mathbf{in}(\mathtt{S}, x).\; \mathbf{out}(\mathtt{S}, \text{if } \pi_1(d) = \mathsf{pk}_X \wedge \mathsf{len}(\pi_2(d)) = \mathsf{len}(n_S))$$
$$\text{then } \{\langle \pi_2(d)\,,\; n_S\rangle\}^{r_0}_{\mathsf{pk_X}}$$
$$\text{else } \{\langle n_S\,,\; n_S\rangle\}^{r_0}_{\mathsf{pk_X}}$$

To prove $\mathsf{I}_A \mid \mathsf{S}_A \approx \mathsf{I}_C \mid \mathsf{S}_A$, we have several **traces**:

$$\mathbf{in}(\mathtt{S}), \mathbf{out}(\mathtt{I}), \mathbf{out}(\mathtt{S}) \qquad \mathbf{in}(\mathtt{S}), \mathbf{out}(\mathtt{S}), \mathbf{out}(\mathtt{I}) \qquad \mathbf{out}(\mathtt{I}), \mathbf{in}(\mathtt{S}), \mathbf{out}(\mathtt{S})$$

But there is a **more general trace**: its security implies the security of the other traces.
See **partial order reduction** (POR) techniques [1].

Private Authentication: Anonymity

Goal:

$$\begin{aligned} & \mathsf{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{A}\}}, \mathsf{out}_2^{\{\mathtt{r}:\mathsf{A};\mathtt{c}:\mathsf{A}\}}[\mathsf{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{A}\}}] \\ \sim\ & \mathsf{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{C}\}}, \mathsf{out}_2^{\{\mathtt{r}:\mathsf{A};\mathtt{c}:\mathsf{A}\}}[\mathsf{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{C}\}}] \end{aligned} \qquad (\mathsf{I_A} \mid \mathsf{S_A} \approx \mathsf{I_C} \mid \mathsf{S_A})$$

where: (notation: $\mathsf{out}_i^{\{\mathtt{r}:\mathtt{recipient};\mathtt{c}:\mathtt{content}\}}$)

$$\begin{aligned} \mathsf{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{X}\}} &\equiv \{\langle \mathsf{pk_X}\,,\, \mathsf{n_I}\rangle\}^{\mathsf{r}}_{\mathsf{pk_S}} \\ \mathsf{out}_2^{\{\mathtt{r}:\mathsf{Y};\mathtt{c}:\mathsf{X}\}}[\mathsf{M}] &\equiv \text{if } \pi_1(d[\mathsf{M}]) = \mathsf{pk_X} \wedge \mathsf{len}(\pi_2(d[\mathsf{M}])) = \mathsf{len}(\mathsf{n_S}) \\ &\quad \text{then } \{\langle \pi_2(d[\mathsf{M}])\,,\, \mathsf{n_S}\rangle\}^{\mathsf{ro}}_{\mathsf{pk_Y}} \\ &\quad \text{else } \{\langle \mathsf{n_S}\,,\, \mathsf{n_S}\rangle\}^{\mathsf{ro}}_{\mathsf{pk_Y}} \\ d[\mathsf{M}] &\equiv \mathsf{dec}(\mathbf{att}_0(\mathsf{M}), \mathsf{sk_S}) \end{aligned}$$

Private Authentication: Anonymity

Goal:

$$\mathsf{out}_1^{\{\mathtt{r:S;c:A}\}}, \mathsf{out}_2^{\{\mathtt{r:A;c:A}\}}[\mathsf{out}_1^{\{\mathtt{r:S;c:A}\}}]$$
$$\sim \mathsf{out}_1^{\{\mathtt{r:S;c:C}\}}, \mathsf{out}_2^{\{\mathtt{r:A;c:A}\}}[\mathsf{out}_1^{\{\mathtt{r:S;c:C}\}}] \qquad (\mathsf{I_A} \mid \mathsf{S_A} \approx \mathsf{I_C} \mid \mathsf{S_A})$$

where: (notation: $\mathsf{out}_i^{\{\mathtt{r:recipient;c:content}\}}$)

$$\mathsf{out}_1^{\{\mathtt{r:S;c:X}\}} \equiv \{\langle \mathsf{pk_X}, \mathsf{n_I}\rangle\}^{\mathsf{r}}_{\mathsf{pk_S}}$$
$$\mathsf{out}_2^{\{\mathtt{r:Y;c:X}\}}[\mathsf{M}] \equiv \text{if } \pi_1(d[\mathsf{M}]) = \mathsf{pk_X} \wedge \mathsf{len}(\pi_2(d[\mathsf{M}])) = \mathsf{len}(\mathsf{n_S})$$
$$\text{then } \{\langle \pi_2(d[\mathsf{M}]), \mathsf{n_S}\rangle\}^{\mathsf{ro}}_{\mathsf{pk_Y}}$$
$$\text{else } \{\langle \mathsf{n_S}, \mathsf{n_S}\rangle\}^{\mathsf{ro}}_{\mathsf{pk_Y}}$$
$$d[\mathsf{M}] \equiv \mathsf{dec}(\mathbf{att}_0(\mathsf{M}), \mathsf{sk_S})$$

Proof strategy: we only reason on the right terms:

1. Push encryption below branching
2. KP-CCA$_1$: $\mathsf{out}_2^{\{\mathtt{r:A;c:A}\}} \Rightarrow \mathsf{out}_2^{\{\mathtt{r:C;c:A}\}}$
3. IND-CCA$_1$: $\mathsf{out}_2^{\{\mathtt{r:C;c:A}\}} \Rightarrow \mathsf{out}_2^{\{\mathtt{r:C;c:C}\}}$
4. Conclude by α-renaming

Private Authentication: Anonymity

First, we push the branching under the encryption:

$$\dfrac{\begin{array}{ll} & \mathrm{out}_1^{\{\mathsf{r:S;c:A}\}}, \mathrm{out}_2^{\{\mathsf{r:A;c:A}\}}[\mathrm{out}_1^{\{\mathsf{r:S;c:A}\}}] \\ \sim & \mathrm{out}_1^{\{\mathsf{r:S;c:C}\}}, \underline{\mathrm{out}_2^{\{\mathsf{r:A;c:A}\}}}[\mathrm{out}_1^{\{\mathsf{r:S;c:C}\}}] \end{array}}{\begin{array}{ll} & \mathrm{out}_1^{\{\mathsf{r:S;c:A}\}}, \mathrm{out}_2^{\{\mathsf{r:A;c:A}\}}[\mathrm{out}_1^{\{\mathsf{r:S;c:A}\}}] \\ \sim & \mathrm{out}_1^{\{\mathsf{r:S;c:C}\}}, \underline{\mathrm{out}_2^{\{\mathsf{r:A;c:A}\}}}[\mathrm{out}_1^{\{\mathsf{r:S;c:C}\}}] \end{array}}\ \mathrm{R}$$

where:

$$\underline{\mathrm{out}_2^{\{\mathsf{r:Y;c:X}\}}}[\mathsf{M}] \equiv \left\{\begin{array}{l} \text{if } \pi_1(d[\mathsf{M}]) = \mathsf{pk_X} \wedge \mathrm{len}(\pi_2(d[\mathsf{M}])) = \mathrm{len}(\mathsf{n_S}) \\ \text{then } \langle \pi_2(d[\mathsf{M}])\,,\, \mathsf{n_S} \rangle \\ \text{else } \langle \mathsf{n_S}\,,\, \mathsf{n_S} \rangle \end{array}\right\}^{\mathsf{r_0}}_{\mathsf{pk_Y}}$$

We let $m_{\mathsf{X}}[\mathsf{M}]$ be the content of the encryption above.

Private Authentication: Anonymity

Then, we use KP-CCA$_1$ to change the encryption key:

$$\frac{\begin{array}{ll} & \mathsf{out}_1^{\{\mathsf{r:S;c:A}\}}, \mathsf{out}_2^{\{\mathsf{r:A;c:A}\}}[\mathsf{out}_1^{\{\mathsf{r:S;c:A}\}}] \\ \sim & \mathsf{out}_1^{\{\mathsf{r:S;c:C}\}}, \underline{\mathsf{out}}_2^{\{\mathsf{r:C;c:A}\}}[\mathsf{out}_1^{\{\mathsf{r:S;c:C}\}}] \end{array}}{\begin{array}{ll} & \mathsf{out}_1^{\{\mathsf{r:S;c:A}\}}, \mathsf{out}_2^{\{\mathsf{r:A;c:A}\}}[\mathsf{out}_1^{\{\mathsf{r:S;c:A}\}}] \\ \sim & \mathsf{out}_1^{\{\mathsf{r:S;c:C}\}}, \underline{\mathsf{out}}_2^{\{\mathsf{r:A;c:A}\}}[\mathsf{out}_1^{\{\mathsf{r:S;c:C}\}}] \end{array}} \text{ TRANS} + \text{KP-CCA}_1$$

since:

- the encryption randomness r_0 is correctly used;
- the key randomness n_A and n_C appear only in $\mathsf{pk}(\cdot)$ and $\mathsf{dec}(_, \mathsf{sk}(\cdot))$ positions.

Private Authentication: Anonymity

Then, we use IND-CCA$_1$ to change the encryption content:

$$\frac{\begin{array}{rl} & \mathsf{out}_1^{\{\mathsf{r:S;c:A}\}}, \mathsf{out}_2^{\{\mathsf{r:A;c:A}\}}[\mathsf{out}_1^{\{\mathsf{r:S;c:A}\}}] \\ \sim & \mathsf{out}_1^{\{\mathsf{r:S;c:C}\}}, \underline{\mathsf{out}_2^{\{\mathsf{r:C;c:C}\}}}[\mathsf{out}_1^{\{\mathsf{r:S;c:C}\}}] \end{array} \; \Pi_1}{\begin{array}{rl} & \mathsf{out}_1^{\{\mathsf{r:S;c:A}\}}, \mathsf{out}_2^{\{\mathsf{r:A;c:A}\}}[\mathsf{out}_1^{\{\mathsf{r:S;c:A}\}}] \\ \sim & \mathsf{out}_1^{\{\mathsf{r:S;c:C}\}}, \underline{\mathsf{out}_2^{\{\mathsf{r:C;c:A}\}}}[\mathsf{out}_1^{\{\mathsf{r:S;c:C}\}}] \end{array}} \; \textsc{Trans} + \textsc{ind-cca}_1$$

since:

- the encryption randomness r_0 is correctly used;
- the key randomness n_C appear only in $\mathsf{pk}(\cdot)$ and $\mathsf{dec}(_, \mathsf{sk}(\cdot))$ positions.

And where Π_1 must be a proof of:

$$\left[\mathsf{len}(m_\mathsf{C}[\mathsf{out}_1^{\{\mathsf{r:S;c:C}\}}]) = \mathsf{len}(m_\mathsf{A}[\mathsf{out}_1^{\{\mathsf{r:S;c:C}\}}])\right].$$

Private Authentication: Anonymity

Recall that:

$$m_{\mathsf{X}}[\mathsf{M}] \equiv \text{if } \pi_1(d[\mathsf{M}]) = \mathsf{pk}_{\mathsf{X}} \wedge \mathsf{len}(\pi_2(d[\mathsf{M}])) = \mathsf{len}(\mathsf{n}_{\mathsf{S}})$$
$$\text{then } \langle \pi_2(d[\mathsf{M}]) \,,\, \mathsf{n}_{\mathsf{S}} \rangle$$
$$\text{else } \langle \mathsf{n}_{\mathsf{S}} \,,\, \mathsf{n}_{\mathsf{S}} \rangle$$

Then:

$$\frac{\mathcal{A}_{\mathsf{th}} \vdash_{\mathrm{GEN}} \mathsf{len}(m_{\mathsf{C}}[\mathrm{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{C}\}}]) = \mathsf{len}(m_{\mathsf{A}}[\mathrm{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{C}\}}])}{\left[\mathsf{len}(m_{\mathsf{C}}[\mathrm{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{C}\}}]) = \mathsf{len}(m_{\mathsf{A}}[\mathrm{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{C}\}}])\right]}\ \mathrm{GEN}$$

if $\mathcal{A}_{\mathsf{th}}$ contains the axiom[1]:

$$\forall x, y.\mathsf{len}(\langle x \,,\, y \rangle) = \mathsf{len}(x) + \mathsf{len}(y) + c$$

where c is some constant left unspecified.

[1] This axiom must be satisfied by the protocol implementation for the proof to apply.

Private Authentication: Anonymity

Then, we α-rename the key randomness $\mathsf{n_C}$, rewrite back the encryption, and conclude.

$$\frac{}{\begin{array}{l}\quad \mathsf{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{A}\}}, \mathsf{out}_2^{\{\mathtt{r}:\mathsf{A};\mathtt{c}:\mathsf{A}\}}[\mathsf{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{A}\}}] \\ \sim\ \mathsf{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{C}\}}, \underline{\mathsf{out}_2^{\{\mathtt{r}:\mathsf{C};\mathtt{c}:\mathsf{C}\}}}[\mathsf{out}_1^{\{\mathtt{r}:\mathsf{S};\mathtt{c}:\mathsf{C}\}}]\end{array}}\ \alpha\text{-EQU} + \mathrm{R} + \textsc{Refl}$$

Authentication Protocols

Authentication Protocol

We now focus on another class of security properties: **correspondance properties** (e.g. **authentication**).

These are properties on a **single** protocol, often expressed as a **temporal** property on **events** of the protocol. E.g.

> *If **Alice** accepts **Bob** at time τ then **Bob** must have initiated a session with **Alice** at time $\tau' < \tau$.*

To formalize the **cryptographic arguments** proving such properties, we will design a specialized **framework** and **proof system**.

The Hash-Lock Protocol

Let $\mathcal{I}$ be a finite set of identities.

Hash-Lock

$$
\begin{aligned}
\mathsf{T}(\mathtt{A},\mathtt{i}) &: \nu\, \mathtt{n}_{\mathtt{A},\mathtt{i}}.\ \mathbf{in}(\mathtt{A}_{\mathtt{i}},\mathtt{x}).\ \mathbf{out}(\mathtt{A}_{\mathtt{i}},\langle \mathtt{n}_{\mathtt{A},\mathtt{i}}\,,\,\mathsf{H}(\langle \mathtt{x}\,,\,\mathtt{n}_{\mathtt{A},\mathtt{i}}\rangle,\mathtt{k}_{\mathtt{A}})\rangle)\\
\mathsf{R}(\mathtt{j}) &: \nu\, \mathtt{n}_{\mathtt{R},\mathtt{j}}.\ \mathbf{in}(\mathtt{R}^1_{\mathtt{j}},_).\ \mathbf{out}(\mathtt{R}^1_{\mathtt{j}},\mathtt{n}_{\mathtt{R},\mathtt{j}}).\\
&\qquad\qquad \mathbf{in}(\mathtt{R}^2_{\mathtt{j}},\mathtt{y}).\\
&\qquad\qquad \mathbf{out}(\mathtt{R}^2_{\mathtt{j}},\text{if } \textstyle\bigvee_{\mathtt{A}\in\mathcal{I}} \pi_2(\mathtt{y}) = \mathsf{H}(\langle \mathtt{n}_{\mathtt{R},\mathtt{j}}\,,\,\pi_1(\mathtt{y})\rangle,\mathtt{k}_{\mathtt{A}}))\\
&\qquad\qquad\qquad \text{then ok}\\
&\qquad\qquad\qquad \text{else ko}
\end{aligned}
$$

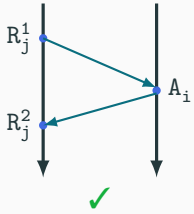
We consider N sessions of each tag, and M sessions of the reader:

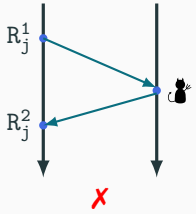
$$\nu\,(\mathtt{k}_{\mathtt{A}})_{\mathtt{A}\in\mathcal{I}}.\ \left(!_{\mathtt{A}\in\mathcal{I}}\ !_{\mathtt{i}<N}\ \mathsf{T}(\mathtt{A},\mathtt{i})\right)\ \mid\ \left(!_{\mathtt{j}<M}\ \mathsf{R}(\mathtt{j})\right)$$

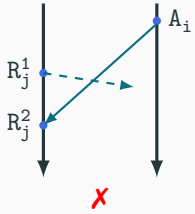
Notation: we write $\mathtt{R}^{\mathtt{i}}_{\mathtt{j}}$ to denote the i-th usage of channel $\mathtt{R}_{\mathtt{j}}$ in a process.

Authentication

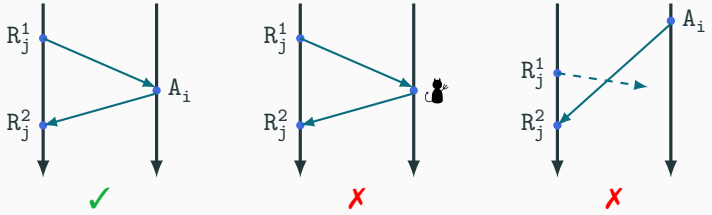
Examples of scenarios:







Examples of scenarios:



- **Middle scenario:** impossible thanks to **unforgeability** of the hash.
- **Right scenario:** impossible thanks to **freshness** of R's name n_R.

Authentication

Definition(informal)

If the j-th session of R accepts believing it talked to tag A, then:

- there exists a session i of tag A **properly interleaved** with the j-th session of R;
- **messages** have been **properly forwarded** between the i-th session of tag A and the j-th session of R.

The second condition can be relaxed to only require partial message correspondence.

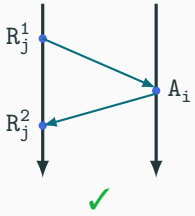
Next slides: a framework to express such temporal properties.

Authentication

Security Property

Anticipating, authentication will be captured by a formula that roughly looks like:

$$\left[\forall \mathtt{j}.\, \mathsf{accept}^{\mathtt{A}}@\mathtt{R}^2_{\mathtt{j}} \Rightarrow \exists \mathtt{i}.\, \begin{pmatrix} \mathtt{R}^1_{\mathtt{j}} < \mathtt{A}_{\mathtt{i}} < \mathtt{R}^2_{\mathtt{j}} \wedge \\ \mathsf{out}@\mathtt{R}^1_{\mathtt{j}} = \mathsf{in}@\mathtt{A}_{\mathtt{i}} \wedge \\ \mathsf{out}@\mathtt{A}_{\mathtt{i}} = \mathsf{in}@\mathtt{R}^2_{\mathtt{j}} \end{pmatrix}\right]$$



Outline

- Capturing **temporal properties as logical formulas** $[\cdot]$.
- **Dedicated proof-system** for $[\cdot]$:
  - **Generic** mathematical reasoning.
  - **Cryptographic** reasoning (CR, EUF).
- **Example**: authentication of Basic Hash.

Authentication Protocols

Execution Traces

Notations

- we let $\leq$ be the **prefix relation** over observable traces:

$$\mathtt{tr}_0 \leq \mathtt{tr}_1 \quad \text{iff.} \quad \exists \mathtt{tr}'.\ \mathtt{tr}_1 = \mathtt{tr}_0; \mathtt{tr}'$$

- $\mathtt{tr} : \mathtt{c}$ states that $\mathtt{tr}$ **ends with an output** on $\mathtt{c}$:

$$\mathtt{tr} : \mathtt{c} \quad \text{iff.} \quad \exists \mathtt{tr}'.\ \mathtt{tr} = \mathtt{tr}'; \mathbf{out}(\mathtt{c})$$

- $\mathtt{tr} : \mathtt{c}^n$ means that $\mathtt{tr} : \mathtt{c}$ and $\mathtt{tr}$ **contains exactly** n **outputs on** $\mathtt{c}$.

Notation: $\mathtt{tr} : \mathtt{c}^n \leq \mathtt{tr}'$ means $\mathtt{tr} : \mathtt{c}^n \wedge \mathtt{tr} \leq \mathtt{tr}'$.

POR Result (Assumed)

We let $\mathcal{T}_{\mathsf{io}}$ be the set of observable traces where all outputs are always **directly preceded** by an input on the same channel, i.e.:

$$\mathtt{tr} \in \mathcal{T}_{\mathsf{io}} \quad \text{iff.} \quad \forall \mathtt{tr}' : \mathtt{c} \leq \mathtt{tr}.\ \exists \mathtt{tr}''.\ \mathtt{tr}' = \mathtt{tr}''; \mathbf{in}(\mathtt{c}); \mathbf{out}(\mathtt{c})$$

Assumption: POR

We admit that to analyze the Hash-Lock protocol, it is sufficient to consider only observables traces in $\mathcal{T}_{\mathsf{io}}$.

Notation (predecessor)

For any $\mathtt{tr}$ and $\mathtt{c}$, we let:

$$\mathsf{pred}(\mathtt{tr}; \mathbf{in}(\mathtt{c}), \mathbf{out}(\mathtt{c})) \stackrel{\mathsf{def}}{=} \mathtt{tr}$$

Authentication of the Hash-Lock Protocol

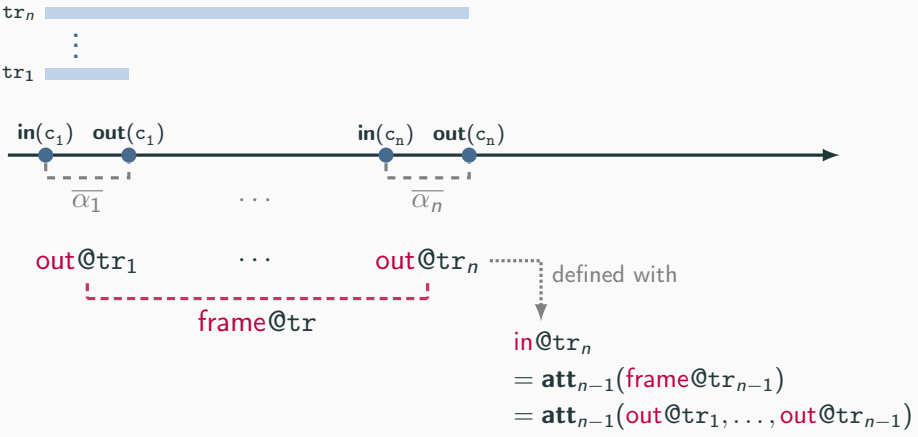
What do we lack to formalize and prove **authentication** for **Hash-Lock**?

- define the (generic) **terms representing** the output, input and acceptance, which we need to state the property;
- have a set of rules for $[\,\cdot\,]$ that can capture the security proof.

Authentication Protocols

Macro Terms

Macro Terms: Graphical Representation



Macro Terms

We define some **generic** terms by **induction** of the observable trace $\mathtt{tr} \in \mathcal{T}_{\mathsf{io}}$.

Let $\mathtt{tr} \in \mathcal{T}_{\mathsf{io}}$ with n inputs. If $\mathsf{frame}(\mathsf{P}, \mathtt{tr}) = t_1, \ldots, t_n$ then we let:

$$\mathsf{out}_{\mathsf{P}}@\mathtt{tr} \stackrel{\mathsf{def}}{=} \begin{cases} t_n & \text{if } \exists \mathsf{c}.\ \mathtt{tr} : \mathsf{c} \\ \mathsf{empty} & \text{otherwise} \end{cases}$$

$$\mathsf{frame}_{\mathsf{P}}@\mathtt{tr} \stackrel{\mathsf{def}}{=} \begin{cases} \mathsf{frame}_{\mathsf{P}}@\mathsf{pred}(\mathtt{tr}), \mathsf{out}_{\mathsf{P}}@\mathtt{tr} & \text{if } |\mathtt{tr}| > 1 \\ \epsilon & \text{otherwise} \end{cases}$$

$$\mathsf{in}_{\mathsf{P}}@\mathtt{tr} \stackrel{\mathsf{def}}{=} \begin{cases} \mathbf{att}_{n-1}(\mathsf{frame}_{\mathsf{P}}@\mathsf{pred}(\mathtt{tr})) & \text{if } |\mathtt{tr}| > 1 \\ \mathsf{empty} & \text{otherwise} \end{cases}$$

Remark: we omit P when it is clear from context.

- *The restriction to traces in $\mathcal{T}_{io}$ simplifies the definition of $\mathit{in}_P@\mathtt{tr}$.*
- *$\mathit{frame}_P@\mathtt{tr}$ is an alternative name for $\mathit{frame}(P, \mathtt{tr})$.*

Hash-Lock: Accept

Hash-Lock

$$
\begin{array}{lll}
\mathsf{T}(\mathtt{A},\mathtt{i}) & : \nu\, \mathsf{n}_{\mathtt{A},\mathtt{i}}. & \mathbf{in}(\mathtt{A}_{\mathtt{i}}, \mathtt{x}).\ \mathbf{out}(\mathtt{A}_{\mathtt{i}}, \langle \mathsf{n}_{\mathtt{A},\mathtt{i}}\ ,\ \mathsf{H}(\langle \mathtt{x},\ \mathsf{n}_{\mathtt{A},\mathtt{i}}\rangle, \mathsf{k}_{\mathtt{A}})\rangle) \\
\mathsf{R}(\mathtt{j}) & : \nu\, \mathsf{n}_{\mathtt{R},\mathtt{j}}. & \mathbf{in}(\mathtt{R}^1_{\mathtt{j}}, _).\ \mathbf{out}(\mathtt{R}^1_{\mathtt{j}}, \mathsf{n}_{\mathtt{R},\mathtt{j}}). \\
 & & \mathbf{in}(\mathtt{R}^2_{\mathtt{j}}, \mathtt{y}). \\
 & & \mathbf{out}(\mathtt{R}^2_{\mathtt{j}}, \text{if } \bigvee_{\mathtt{A}\in\mathcal{I}} \pi_2(\mathtt{y}) = \mathsf{H}(\langle \mathsf{n}_{\mathtt{R},\mathtt{j}}\ ,\ \pi_1(\mathtt{y})\rangle, \mathsf{k}_{\mathtt{A}})) \\
 & & \quad \text{then ok} \\
 & & \quad \text{else ko}
\end{array}
$$

To be able to state some **authentication** property of Hash-Lock, we need an additional macro. For all $\mathtt{tr} : \mathtt{R}^2_{\mathtt{j}} \in \mathcal{T}_{\mathsf{io}}$, we let:

$$\mathsf{accept}^{\mathtt{A}}@\mathtt{tr} \stackrel{\mathsf{def}}{=} \pi_2(\mathsf{in}@\mathtt{tr}) = \mathsf{H}(\langle \mathsf{n}_{\mathtt{R},\mathtt{j}}\ ,\ \pi_1(\mathsf{in}@\mathtt{tr})\rangle, \mathsf{k}_{\mathtt{A}})$$

We made sure that all names in the protocol are unique, so that they don't have to be renamed before the symbolic execution.

Authentication: Hash-Lock

The following formulas encode the fact that the **Hash-Lock** protocol provides **authentication**:

$$\forall \mathsf{A} \in \mathcal{I}.\ \forall \mathtt{tr} \in \mathcal{T}_{\mathsf{io}}.\ \forall \mathtt{tr}_1 : \mathtt{R}^1_{\mathsf{j}}, \mathtt{tr}_3 : \mathtt{R}^2_{\mathsf{j}} \text{ s.t. } \mathtt{tr}_1 < \mathtt{tr}_3 \leq \mathtt{tr},$$

$$\left[\mathsf{accept}^{\mathsf{A}}@\mathtt{tr}_3 \rightarrow \bigvee_{\substack{\mathtt{tr}_2 : \mathtt{A}_{\mathsf{i}} \\ \mathtt{tr}_1 \leq \mathtt{tr}_2 \leq \mathtt{tr}_3}} \begin{array}{l} \mathsf{out}@\mathtt{tr}_1 = \mathsf{in}@\mathtt{tr}_2 \wedge \\ \mathsf{out}@\mathtt{tr}_2 = \mathsf{in}@\mathtt{tr}_3 \end{array} \right]$$

This kind of one-sided properties are called **correspondance** properties. Proving their validity will require **additional rules**, to allow for **propositional reasoning**.

Authentication Protocols

Local Proof System

Local Judgements

We define a **judgment** dedicated to **correspondance properties**.

Definition

A **local judgement** $\Gamma \vdash \phi$ comprises a sequence of boolean terms $\Gamma = \phi_1, \ldots, \phi_n$ and a boolean term ϕ.

$\Gamma \vdash \phi$ is **valid** if and only if the following formula is valid:

$$\left[\phi_1 \to \cdots \to \phi_n \to \phi\right]$$

Boolean Connectives in Local Judgements

Careful not to confuse the boolean connectives at the **local** and **equivalence** levels!

Exercise

Determine which directions are correct.

$$[\phi \wedge \psi] \overset{?}{\Leftrightarrow} [\phi] \mathbin{\tilde{\wedge}} [\psi]$$

$$[\phi \vee \psi] \overset{?}{\Leftrightarrow} [\phi] \mathbin{\tilde{\vee}} [\psi]$$

$$[\phi \to \psi] \overset{?}{\Leftrightarrow} [\phi] \mathbin{\tilde{\to}} [\psi]$$

Boolean Connectives in Local Judgements

Careful not to confuse the boolean connectives at the **local** and **equivalence** levels!

Exercise

Determine which directions are correct.

$$[\phi \wedge \psi] \quad \Leftrightarrow \quad [\phi] \mathbin{\tilde{\wedge}} [\psi]$$

$$[\phi \vee \psi] \quad \Leftarrow \quad [\phi] \mathbin{\tilde{\vee}} [\psi] \qquad (\not\Rightarrow\text{: take } \phi = \mathsf{b}, \psi = \neg\mathsf{b})$$

$$[\phi \to \psi] \quad \Rightarrow \quad [\phi] \mathbin{\tilde{\to}} [\psi] \qquad (\not\Leftarrow\text{: idem})$$

The second relation works both ways when ϕ or ψ is a **constant** formula.

Local Proof System

Our **local judgement** can be trivially equipped with a **sequent calculus** that behaves as a standard FO sequent calculus.

$$\frac{}{\Gamma, \phi \vdash \phi} \qquad \frac{\Gamma \vdash \psi \qquad \Gamma, \psi \vdash \phi}{\Gamma \vdash \phi}$$

$$\frac{\Gamma \vdash \psi \qquad \Gamma \vdash \phi}{\Gamma \vdash \psi \wedge \phi} \qquad \frac{\Gamma, \psi, \phi \vdash \theta}{\Gamma, \psi \wedge \phi \vdash \theta}$$

$$\frac{\Gamma \vdash \phi}{\Gamma \vdash \psi \vee \phi} \qquad \frac{\Gamma \vdash \psi}{\Gamma \vdash \psi \vee \phi} \qquad \frac{\Gamma, \psi \vdash \theta \qquad \Gamma, \phi \vdash \theta}{\Gamma, \psi \vee \phi \vdash \theta}$$

$$\frac{\Gamma \vdash \psi \qquad \Gamma, \phi \vdash \theta}{\Gamma, \psi \to \phi \vdash \theta} \qquad \frac{\Gamma, \psi \vdash \phi}{\Gamma \vdash \psi \to \phi}$$

Local Proof System (cont.)

$$\frac{\Gamma, \phi \vdash \bot}{\Gamma \vdash \neg\phi} \qquad \frac{}{\Gamma, \bot \vdash \phi}$$

$$\frac{\Gamma_1, \phi, \psi, \Gamma_2 \vdash \theta}{\Gamma_1, \psi, \phi, \Gamma_2 \vdash \theta} \qquad \frac{\Gamma, \psi, \psi \vdash \phi}{\Gamma, \psi \vdash \phi} \qquad \frac{\Gamma \vdash \theta}{\Gamma, \phi \vdash \theta}$$

Local Proof System: Soundness

The local proof system is **sound**.

Proof

First, recall that for any Γ and θ:

$$\Gamma \vdash \theta \text{ is valid iff. } \Pr_\rho\left([\![(\wedge\Gamma) \wedge \neg\phi]\!]^{\eta,\rho}_{\mathbb{M}}\right) \text{ is negligible.} \qquad (\dagger)$$

Local Proof System: Soundness

We will only detail one rule, say:

$$\frac{\Gamma, \psi \vdash \theta \qquad \Gamma, \phi \vdash \theta}{\Gamma, \psi \vee \phi \vdash \theta.}$$

By the previous remark (†), since $(\Gamma, \psi \vdash \theta)$ and $(\Gamma, \phi \vdash \theta)$ are valid:

- $\mathsf{Pr}_\rho\left([\![(\wedge\Gamma) \wedge \psi \wedge \neg\theta]\!]_{\mathbb{M}}^{\eta,\rho}\right)$ is negligible.
- $\mathsf{Pr}_\rho\left([\![(\wedge\Gamma) \wedge \phi \wedge \neg\theta]\!]_{\mathbb{M}}^{\eta,\rho}\right)$ is negligible.

Since the union of two negligible (η-indexed families of) events is a negligible (η-indexed families of) events,

$$\begin{aligned} & \mathsf{Pr}_\rho\left([\![((\wedge\Gamma) \wedge \psi \wedge \neg\theta) \vee ((\wedge\Gamma) \wedge \phi \wedge \neg\theta)]\!]_{\mathbb{M}}^{\eta,\rho}\right) \text{ is negligible} \\ \Leftrightarrow \quad & \mathsf{Pr}_\rho\left([\![(\wedge\Gamma) \wedge (\psi \vee \phi) \wedge \neg\theta]\!]_{\mathbb{M}}^{\eta,\rho}\right) \text{ is negligible} \end{aligned}$$

Hence using (†) again, $\Gamma, \psi \vee \phi \vdash \theta$ is valid.

Authentication Protocols

Cryptographic Rule: Collision Resistance

Cryptographic Hash

A **keyed cryptographic hash** $\mathsf{H}(_,_)$ is **computationally collision resistant** if no PPTM adversary can built collisions, even when it has access to a hashing **oracle**.

More precisely, a hash is *collision resistant under hidden key attacks* (CR-HK) iff for every PPTM $\mathcal{A}$, the following quantity:

$$\Pr_{\mathsf{k}}\left(\mathcal{A}^{\mathcal{O}_{\mathsf{H}(\cdot,\mathsf{k})}}(1^{\eta}) = \langle m_1\,,\, m_2\rangle, m_1 \neq m_2 \text{ and } \mathsf{H}(m_1,\mathsf{k}) = \mathsf{H}(m_2,\mathsf{k})\right)$$

is negligible, where k is drawn uniformly in $\{0,1\}^{\eta}$.

CR Rule

Collision Resistance

If H is a CR-HK function, then the *ground* rule:

$$\frac{}{\mathsf{H}(m_1, \mathsf{k}) = \mathsf{H}(m_2, \mathsf{k}) \vdash m_1 = m_2}\ \mathrm{CR}$$

is sound, when k appears only in H key positions in m_1, m_2.

Authentication Protocols

Cryptographic Rule: Message Authentication Code

Message Authentication Code

A **message authentication code** is a symmetric cryptographic schema which:

- create **message authentication codes** using $\mathsf{mac}_{\cdot}(\cdot)$
- **verifies** mac using $\mathsf{verify}_{\cdot}(\cdot, \cdot)$

It must satisfies the functional equality:

$$\mathsf{verify}_k(\mathsf{mac}_k(m), m) = \mathsf{true}$$

MAC Security

A MAC must be **computationally unforgeable**, even when the adversary has access to a mac and verify **oracles**.

A MAC is *unforgeable against chosen-message attacks* (EUF-CMA) iff for every PPTM $\mathcal{A}$, the following quantity:

$$\Pr_{\mathsf{k}}\left(\begin{array}{r}\mathcal{A}^{\mathcal{O}_{\mathsf{mac}_{\mathsf{k}}}(\cdot),\mathcal{O}_{\mathsf{verify}_{\mathsf{k}}}(\cdot,\cdot)}(1^{\eta}) = \langle m\,,\,\sigma\rangle,\ m \text{ not queried to } \mathcal{O}_{\mathsf{mac}_{\mathsf{k}}}(\cdot) \\ \text{and } \mathsf{verify}_{\mathsf{k}}(\sigma, m) = 1\end{array}\right)$$

is negligible, where k is drawn uniformly in $\{0,1\}^{\eta}$.

EUF-MAC Rule

Take two messages s, m and a key $\mathsf{k} \in \mathcal{N}$ such that:

- s and m are ground;
- $\mathsf{k} \in \mathcal{N}$ appears only in mac or verify key positions in s, m.

Key Idea

To build a rule for EUF-CMA, we proceed as follow:

- compute $[\![s, m]\!]$ bottum-up, calling $\mathcal{O}_{\mathsf{mac}_\mathsf{k}(\cdot)}$ and $\mathcal{O}_{\mathsf{verify}_\mathsf{k}(\cdot,\cdot)}$ if necessary;
- log all sub-terms $\mathbb{S}_{\mathsf{mac}}(s, m)$ sent to $\mathcal{O}_{\mathsf{mac}_\mathsf{k}(\cdot)}$.

$\Rightarrow$ If $\mathsf{verify}_\mathsf{k}(s, m)$ then $m = u$ for some $u \in \mathbb{S}_{\mathsf{mac}}(s, m)$.

$\mathbb{S}_{mac}(s, m)$ *are the* ***calls*** *to* $\mathcal{O}_{mac_k(\cdot)}$ *needed to compute* s, m.

EUF-MAC Rule

$\mathbb{S}_{\mathsf{mac}}(\cdot)$ defined by induction on ground terms:

$$\mathbb{S}_{\mathsf{mac}}(\mathsf{n}) \stackrel{\mathsf{def}}{=} \emptyset$$

$$\mathbb{S}_{\mathsf{mac}}(\mathsf{verify}_{\mathsf{k}}(u_1, u_2)) \stackrel{\mathsf{def}}{=} \mathbb{S}_{\mathsf{mac}}(u_1) \cup \mathbb{S}_{\mathsf{mac}}(u_2)$$

$$\mathbb{S}_{\mathsf{mac}}(\mathsf{mac}_{\mathsf{k}}(u)) \stackrel{\mathsf{def}}{=} \{u\} \cup \mathbb{S}_{\mathsf{mac}}(u)$$

$$\mathbb{S}_{\mathsf{mac}}(f(u_1, \ldots, u_n)) \stackrel{\mathsf{def}}{=} \bigcup_{1 \leq i \leq n} \mathbb{S}_{\mathsf{mac}}(u_i) \qquad \text{(for other cases)}$$

EUF-MAC Rule

Message Authentication Code Unforgeability

If mac is an EUF-CMA function, then the *ground* rule:

$$\frac{}{\mathsf{verify}_{\mathsf{k}}(s, m) \vdash \bigvee_{u \in \mathcal{S}} m = u}\ \text{EUF-MAC}$$

is sound, when:

- $\mathcal{S} = \mathbb{S}_{\mathsf{mac}}(s, m)$;
- $\mathsf{k} \in \mathcal{N}$ appears only in mac or verify key positions in s, m.

Example

If t_1 t_2 and t_3 are terms which do not contain k, then:

$$\Phi \equiv \mathsf{mac}_{\mathsf{k}}(t_1), \mathsf{mac}_{\mathsf{k}}(t_2), \mathsf{mac}_{\mathsf{k}_0}(t_3)$$

$$\left[\mathsf{verify}_{\mathsf{k}}(g(\Phi), \mathsf{n}) \ \rightarrow\ \left(\mathsf{n} = t_1 \vee \mathsf{n} = t_2\right)\right]$$

EUF-MAC Rule: Exercise

Exercise

Assume mac is EUF-CMA. Show that the following rule is sound:

$$\frac{}{\mathsf{verify}_{\mathsf{k}}(\text{if } b \text{ then } s_0 \text{ else } s_1, m) \vdash \bigvee_{u \in \mathcal{S}_1 \cup \mathcal{S}_2} m = u}$$

when b, s_0, s_1, m are *ground* terms, and:

- $\mathcal{S}_i = \{u \mid \mathsf{mac}_{\mathsf{k}}(u) \in \mathbb{S}_{\mathsf{mac}}(s_i, m)\}$, for $i \in \{0, 1\}$;
- k appears only in mac or verify key positions in s_0, s_1, m.

Remark: we do not make *any* assumption on b, except that it is ground. E.g., we can have $b \equiv (\mathbf{att}(\mathsf{k}) = \mathsf{mac}_{\mathsf{k}}(0))$.

Authentication Protocols

Authentication of the Hash-Lock Protocol

Authentication: Hash-Lock

Theorem

Assuming that the hash function is EUF-CMA[2], the Hash-Lock protocol provides **authentication**, i.e. for any identity $\mathsf{A} \in \mathcal{I}$, for any $\mathtt{tr} \in \mathcal{T}_{\mathsf{io}}$, $\mathtt{tr}_1 : \mathtt{R}_{\mathtt{j}}^1$ and $\mathtt{tr}_3 : \mathtt{R}_{\mathtt{j}}^2$ s.t.:

$$\mathtt{tr}_1 < \mathtt{tr}_3 \leq \mathtt{tr}$$

the following formula is valid:

$$\mathsf{accept}^{\mathsf{A}}@\mathtt{tr}_3 \vdash \bigvee_{\substack{\mathtt{tr}_2 : \mathtt{A}_{\mathtt{i}} \\ \mathtt{tr}_1 \leq \mathtt{tr}_2 \leq \mathtt{tr}_3}} \begin{array}{l} \mathsf{out}@\mathtt{tr}_1 = \mathsf{in}@\mathtt{tr}_2 \wedge \\ \mathsf{out}@\mathtt{tr}_2 = \mathsf{in}@\mathtt{tr}_3 \end{array}$$

[2] Taking $\mathsf{verify}_{\mathsf{k}}(s, m) \stackrel{\mathsf{def}}{=} s = \mathsf{H}(m, \mathsf{k})$.

Authentication: Hash-Lock

Proof. Let $a \in \mathcal{I}$, and let $\mathtt{tr} \in \mathcal{T}_{\mathsf{io}}$, $\mathtt{tr}_1 : \mathtt{R}^1_j$ and $\mathtt{tr}_3 : \mathtt{R}^2_j$ be s.t.:

$$\mathtt{tr}_1 < \mathtt{tr}_3 \leq \mathtt{tr}$$

We let:

$$\phi_{\mathsf{conc}} \overset{\mathsf{def}}{=} \bigvee_{\substack{\mathtt{tr}_2 : \mathtt{A}_i \\ \mathtt{tr}_1 \leq \mathtt{tr}_2 \leq \mathtt{tr}_3}} \mathsf{out}@\mathtt{tr}_1 = \mathsf{in}@\mathtt{tr}_2 \wedge \mathsf{out}@\mathtt{tr}_2 = \mathsf{in}@\mathtt{tr}_3$$

We must prove that the following local judgement is valid:

$$\mathsf{accept}^{\mathsf{A}}@\mathtt{tr}_3 \vdash \phi_{\mathsf{conc}}$$

i.e. that:

$$\pi_2(\mathsf{in}@\mathtt{tr}_3) = \mathsf{H}(\langle \mathsf{n}_{\mathsf{R},j}\,,\, \pi_1(\mathsf{in}@\mathtt{tr}_3)\rangle, \mathsf{k}_{\mathsf{A}}) \vdash \phi_{\mathsf{conc}}$$

Authentication: Hash-Lock

We use the EUF-MAC rule on the equality:

$$\pi_2(\mathsf{in}@\mathtt{tr}_3) = \mathsf{H}(\langle \mathsf{n}_{\mathsf{R},\mathsf{j}}\,,\, \pi_1(\mathsf{in}@\mathtt{tr}_3)\rangle, \mathsf{k}_\mathsf{A}) \qquad (\dagger)$$

The terms above are ground, and the key k_A is correctly used in them. Moreover, the set of *honest* hashes using key k_A appearing in (†), excluding the top-level hash, is:

$$\begin{aligned} & \mathbb{S}_{\mathsf{mac}}(\pi_2(\mathsf{in}@\mathtt{tr}_3), \langle \mathsf{n}_{\mathsf{R},\mathsf{j}}\,,\, \pi_1(\mathsf{in}@\mathtt{tr}_3)\rangle) \\ = \; & \mathbb{S}_{\mathsf{mac}}(\mathsf{in}@\mathtt{tr}_3) \\ = \; & \{\mathsf{H}(\langle \mathsf{in}@\mathtt{tr}_2\,,\, \mathsf{n}_{\mathtt{A},\mathtt{i}}\rangle, \mathsf{k}_\mathsf{A}) \mid \mathtt{tr}_2 : \mathtt{A}_\mathtt{i} < \mathtt{tr}_3\} \end{aligned}$$

The hashes in the reader's outputs can be seen as verify checks, and can therefore be ignored.

Authentication: Hash-Lock

Hence using EUF-MAC plus some basic reasoning, we have:

$$\dfrac{\dfrac{\mathsf{accept}^{\mathsf{A}}@\mathtt{tr}_3, \begin{array}{l}\langle \mathsf{in}@\mathtt{tr}_2\,,\, \mathsf{n}_{\mathsf{A},\mathtt{i}}\rangle = \\ \langle \mathsf{n}_{\mathsf{R},\mathtt{j}}\,,\, \pi_1(\mathsf{in}@\mathtt{tr}_3)\rangle\end{array} \vdash \phi_{\mathsf{conc}} \qquad \text{for every } \mathtt{tr}_2 : \mathtt{A}_\mathtt{i} < \mathtt{tr}_3}{\mathsf{accept}^{\mathsf{A}}@\mathtt{tr}_3, \bigvee_{\mathtt{tr}_2:\mathtt{A}_\mathtt{i}<\mathtt{tr}_3} \begin{array}{l}\langle \mathsf{in}@\mathtt{tr}_2\,,\, \mathsf{n}_{\mathsf{A},\mathtt{i}}\rangle = \\ \langle \mathsf{n}_{\mathsf{R},\mathtt{j}}\,,\, \pi_1(\mathsf{in}@\mathtt{tr}_3)\rangle\end{array} \vdash \phi_{\mathsf{conc}}}}{\mathsf{accept}^{\mathsf{A}}@\mathtt{tr}_3 \vdash \phi_{\mathsf{conc}}}$$

Authentication: Hash-Lock

We only have to show that for every $\mathtt{tr}_2 : \mathtt{A}_\mathtt{i} < \mathtt{tr}_3$:

$$\underbrace{\mathsf{accept}^{\mathsf{A}}@\mathtt{tr}_3,\ \mathsf{in}@\mathtt{tr}_2 = \mathtt{n}_{\mathtt{R},\mathtt{j}},\ \mathtt{n}_{\mathtt{A},\mathtt{i}} = \pi_1(\mathsf{in}@\mathtt{tr}_3)}_{\Gamma} \vdash \phi_{\mathsf{conc}}.$$

After some basic equality reasoning (+ minor assumptions), we have:

$$\Gamma \vdash \mathsf{out}@\mathtt{tr}_1 = \mathsf{in}@\mathtt{tr}_2 \wedge \mathsf{out}@\mathtt{tr}_2 = \mathsf{in}@\mathtt{tr}_3 \qquad (\ddagger)$$

Authentication: Hash-Lock

Recall that:

$$\phi_{\mathsf{conc}} \overset{\mathsf{def}}{=} \bigvee_{\substack{\mathtt{tr}_2:\mathtt{A}_i \\ \mathtt{tr}_1 \leq \mathtt{tr}_2 \leq \mathtt{tr}_3}} \mathsf{out}@\mathtt{tr}_1 = \mathsf{in}@\mathtt{tr}_2 \wedge \mathsf{out}@\mathtt{tr}_2 = \mathsf{in}@\mathtt{tr}_3$$

and we must show that $\Gamma \vdash \phi_{\mathsf{conc}}$. Hence, using $(\ddagger)$, it only remains to prove that whenever $\mathtt{tr}_2 < \mathtt{tr}_1$, we have:

$$\Gamma,\ \mathsf{out}@\mathtt{tr}_1 = \mathsf{in}@\mathtt{tr}_2,\ \mathsf{out}@\mathtt{tr}_2 = \mathsf{in}@\mathtt{tr}_3 \vdash \bot$$

This follows from the independence rule:

$$\frac{}{[t \neq \mathsf{n}]}\ =\text{-IND} \quad \text{when } t \text{ is ground and } \mathsf{n} \notin \mathsf{st}(t)$$

using the fact that:

$$\mathsf{out}@\mathtt{tr}_1 \overset{\mathsf{def}}{=} \mathsf{n}_{\mathsf{R},\mathsf{j}}$$

and that if $\mathtt{tr}_2 < \mathtt{tr}_1$ then $\mathsf{n}_{\mathsf{R},\mathsf{j}} \notin \mathsf{st}(\mathsf{in}@\mathtt{tr}_2)$.

Proof of (‡)

Since $\mathtt{tr}_1 : \mathtt{R}_\mathtt{j}^1 < \mathtt{tr}_3$ we know that:

$$\mathsf{out}@\mathtt{tr}_1 \overset{\mathsf{def}}{=} \mathsf{n}_{\mathsf{R},\mathsf{j}}$$

Moreover:

$$\mathsf{out}@\mathtt{tr}_2 \overset{\mathsf{def}}{=} \langle \mathsf{n}_{\mathsf{A},\mathsf{i}} \,,\, \mathsf{H}(\langle \mathsf{in}@\mathtt{tr}_2 \,,\, \mathsf{n}_{\mathsf{A},\mathsf{i}} \rangle, \mathsf{k}_\mathsf{A}) \rangle$$

Hence:

$$\Gamma \vdash \pi_1(\mathsf{out}@\mathtt{tr}_2) = \pi_1(\mathsf{in}@\mathtt{tr}_3) \qquad (\diamond)$$

Similarly:

$$\begin{aligned} \Gamma \vdash \pi_2(\mathsf{out}@\mathtt{tr}_2) &= \mathsf{H}(\langle \mathsf{in}@\mathtt{tr}_2 \,,\, \mathsf{n}_{\mathsf{A},\mathsf{i}} \rangle, \mathsf{k}_\mathsf{A}) \\ &= \mathsf{H}(\langle \mathsf{n}_{\mathsf{R},\mathsf{j}} \,,\, \pi_1(\mathsf{in}@\mathtt{tr}_3) \rangle, \mathsf{k}_\mathsf{A}) \\ &= \pi_2(\mathsf{in}@\mathtt{tr}_3) \end{aligned}$$

Consequently:

$$\Gamma \vdash \pi_2(\mathsf{out}@\mathtt{tr}_2) = \pi_2(\mathsf{in}@\mathtt{tr}_3) \qquad (\star)$$

Proof of (‡) (cont.)

Assuming that the pair and projections satisfy the property:

$$\overline{\left[(\pi_1\ x = \pi_1\ y) \to (\pi_2\ x = \pi_2\ y) \to x = y\right]}$$

We deduce from $(\star)$ and $(\diamond)$ that:

$$\Gamma \vdash \mathsf{out}(\mathtt{tr}_2) = \mathsf{in}(\mathtt{tr}_3)$$

Putting everything together, we get:

$$\Gamma \vdash \mathsf{out}(\mathtt{tr}_1) = \mathsf{in}(\mathtt{tr}_2) \wedge \mathsf{out}(\mathtt{tr}_2) = \mathsf{in}(\mathtt{tr}_3) \qquad (\ddagger)$$

Going Further

The Squirrel Prover

The Squirrel Prover is a **proof assistant** based on (an extensions of) the CCSA logic [5] we saw during the lectures:

- Users prove goals using sequences of `tactics`.
  - **Generic maths.** tactics, e.g. `apply`, `rewrite`.
  - **Crypto.** tactics, e.g. `cpa`.
  - **Probabilistic** tactics, e.g. `fresh`.
  - **Structural** tactics, e.g. `trans`.
- Development done using a proof-general mode.
  As in Rocq, EasyCrypt ...
- **Open-source tool**, project web-page:
  `https://squirrel-prover.github.io/`



Higher-Order Logic for Cryptography

The logic we saw works but misses several important features

- not internal support for **unbounded protocols**;
- little capacity for **generic mathematical reasoning**.

Problem: This is an obstacles for **mechanization**.

Solution: SQUIRREL uses a more complex **higher-order logic** [3].

Parametric vs Polynomial Security

A subtle discrepancy between security notions:

- We prove **parametric security**:
  for each fixed trace $\mathcal{T}$, there is no winning attacker

$$\forall \mathcal{T}. \forall \mathcal{A}.\ \Pr(\Phi \text{ holds in } \mathcal{T} \text{ against } \mathcal{A}) \text{ is overwhelming in } \eta$$

- We want **polynomial security**:
  there is no winning attacker **adaptively** choosing the trace $\mathcal{T}$ (of length η)

$$\forall \mathcal{A}.\ \Pr(\Phi \text{ holds in } \mathcal{T} \text{ chosen by } \mathcal{A}) \text{ is overwhelming in } \eta$$

Parametric vs Polynomial Security

How to obtain **polynomial security** using CCSA [2]:

- Change the **execution model**:
  frame@N where (N : `int`) instead of frame@τ.
- **Difficulty**: SQUIRREL induction rule is for a constant number of steps.
  Because $\sum_{i \leq P(\eta)} f_i(\eta)$ is not always negligible,
  even if $f_i(\eta)$ negligible $\forall i$ and $P(\eta)$ polynomial.
- Solution: move to a **concrete security** setting.
  - **concrete security predicates** $[\phi]_\epsilon$ and $\vec{u}_0 \sim_\epsilon \vec{u}_1$.
  - reasoning rules with **explicit bounds**.
  - support **general induction**:
    user must prove a uniform bound on all f_i's.
- Only the **correspondence logic** is implemented in SQUIRREL.

From Hardness Assumptions to Logical Rules

- We saw how to manually translate hardness assumptions to rules
  CCA1, EUF-CMA, CR, PRF, DDH, ...

hardness assumption (imperative, stateful programs)

$\Downarrow$

reasoning rules (pure, logic)

- Adding rules for new hardness assumptions is:
  tedious, **error-prone**, and **not in user-space** (Ocaml code).

From Hardness Assumptions to Logical Rules

It is possible to do better by **systematically synthesizing cryptographic reductions**, which allows to translates hardness assumptions into rules [4].

Inputs:

- an (imperative, stateful) **hardness assumption** $\mathcal{G}_0 \approx \mathcal{G}_1$.
- an **indistinguishability property** $u_0 \sim u_1$ to prove.

Goal: synthesize $\mathcal{S}$ poly-time such that $\begin{cases} \mathcal{S}^{\mathcal{G}_0}() = [\![u_0]\!]^{\eta,\rho} \\ \text{and } \mathcal{S}^{\mathcal{G}_1}() = [\![u_1]\!]^{\eta,\rho} \end{cases}$

From Hardness Assumptions to Logical Rules

- **General framework** to add new hardness assumptions.
- **Proof system** to establish the existence of $\mathcal{S}$.
- **Fully automated implementation** (heuristic based $\Rightarrow$ incomplete)

References i

[1] D. Baelde, S. Delaune, and L. Hirschi.
Partial order reduction for security protocols.
In *CONCUR*, volume 42 of *LIPIcs*, pages 497–510. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015.

[2] D. Baelde, C. Fontaine, A. Koutsos, G. Scerri, and T. Vignon.
A Probabilistic Logic for Concrete Security.
In *CSF*, pages 324–339. IEEE, 2024.

[3] D. Baelde, A. Koutsos, and J. Lallemand.
A Higher-Order Indistinguishability Logic for Cryptographic Reasoning.
In *LICS*, pages 1–13. IEEE, 2023.

[4] D. Baelde, A. Koutsos, and J. Sauvage.
Foundations for Cryptographic Reductions in CCSA Logics.
In *CCS*. ACM, 2024.

[5] G. Bana and H. Comon-Lundh.
A computationally complete symbolic attacker for equivalence properties.
In *CCS*, pages 609–620. ACM, 2014.