Squirrel, an Interactive Prover for Protocol Verification in the Computational Model

D. Baelde, S. Delaune, C. Jacomme, **A. Koutsos**, S. Moreau

October 28, 2020

Context

Security Protocols

Distributed programs which aim at providing some **security** properties.

Security Attacks

Attacks against security protocols can be very **damageable**, e.g. theft or privacy breach.

⇒ We need to check that protocols are secure:
formal methods allow to do that, with strong guarantees.

Formal Methods

Verification of Security Protocols

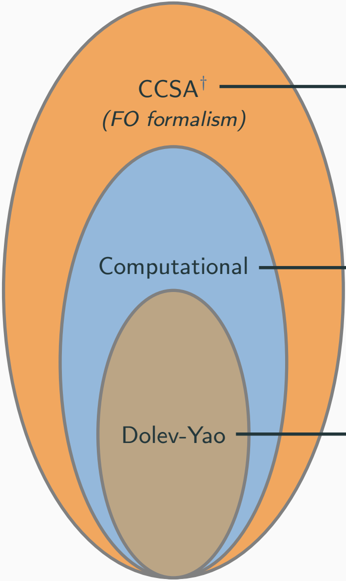
The goal is to completely rule-out classes of attacks.

$$\forall \mathcal{A} \in \mathcal{C}, \qquad (\, \mathcal{A} \mid\mid \mathsf{P} \,) \models \phi$$

Attacker Class

What is the class $\mathcal{C}$ of attackers?

Attacker Classes from the Literature



Adversary: any **arbitrary** function $\mathcal{A}$ that satisfies some **axioms**, i.e. $\mathcal{A} \models \mathsf{Ax}$
Defined by what he **cannot** do, e.g.:
$\{m_0\}_{\mathsf{pk}} \sim \{m_1\}_{\mathsf{pk}}$ (if $\mathsf{len}(m_0) = \mathsf{len}(m_1)$)
✓ Strong security ✓ Good automation

Adversary: any **PPT** function $\mathcal{A}$
models a real-world attacked.
✓ Strong security ✗ Limited automation

Adversary: **fixed** set of rules $\mathcal{E}$
Defined by what he **can** do.
E.g. $\mathsf{dec}(\{x\}_{\mathsf{pk}}, \mathsf{sk}) \rightarrow x$
✗ Reduced security ✓ Good automation

† *Computationally Complete Symbolic Attacker*

CCSA: Related Work

[BC12], **[CLCS14]**

- First framework, **only for reachability properties**.
- A tool implementing a decision procedure, tested on a few protocols for a small number of sessions.

[BC14], **[CK17]**

- New framework, both for **reachability and equivalence** properties.
- Framework allows to do **manual** proofs, only for a **bounded number** of sessions.

Our Contributions (under submission, S&P).

Contribution 2:
An interactive prover, SQUIRREL, to **mechanize proofs**.

Manual proofs, only for a bounded number of sessions.

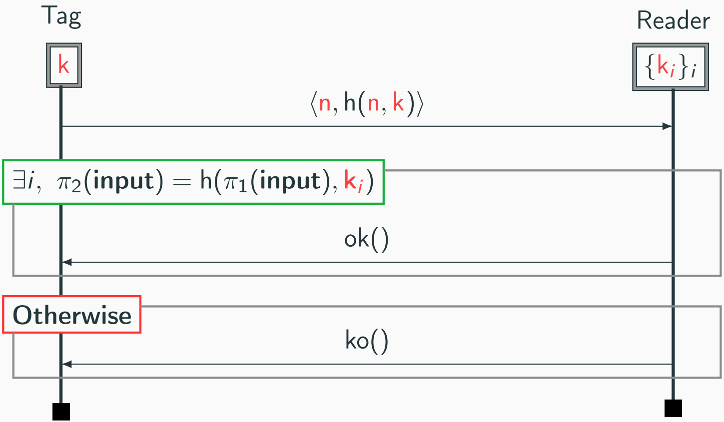
Contribution 1:
A **theoretical framework** (a meta-logic) to express and prove security properties (reachability and equivalence) for an **arbitrary number of sessions**.

Outline

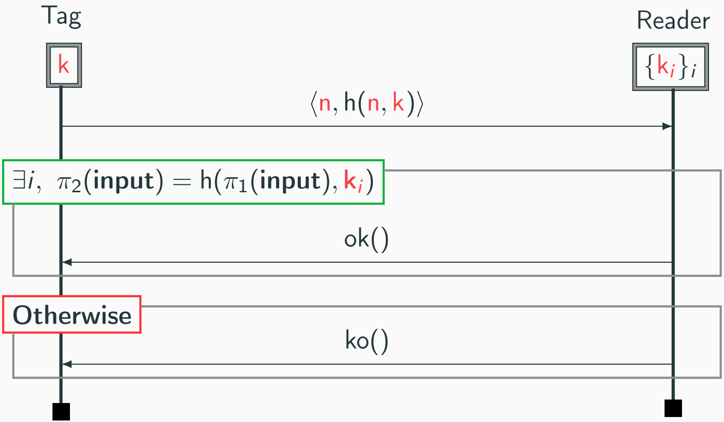
1. The Base logic: CCSA
2. The Meta-Logic
3. An Interactive Theorem Prover, SQUIRREL

The Base logic: CCSA

Basic Hash Protocol



Basic Hash Protocol



To formally model this protocol's security, we need to model:

- messages, i.e. distributions over bit-strings $\Rightarrow$ terms
- security properties (reachability or equivalence) $\Rightarrow$ formulas

Bana-Comon Approach: Messages

We model *protocol messages* using **terms** built upon:

- **names** $\mathcal{N}$, e.g. $\mathsf{n}_A, \mathsf{n}_B$, for random samplings (including keys).
- **function symbols** $\mathcal{F}$ for protocol functions, e.g.:

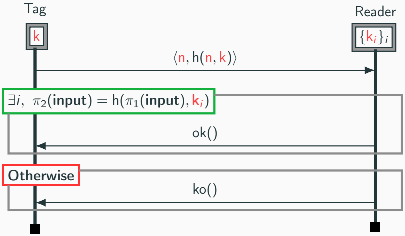
$$\mathsf{h}(\_,\_), \langle \_,\_ \rangle,\ \pi_i(\_),\ \mathsf{ok}(), \mathsf{if\_then\_else\_},\ \mathsf{eq}(\_,\_)$$

- **function symbols** $\mathcal{G}$ for adversarial computations, e.g.:

$$\mathsf{att}(\_)$$

- **variables** $x \in \mathcal{X}$.

Bana-Comon Approach: Messages, an Example



Terms for Basic Hash with two tags (with keys k_1, k_2):

$$\langle n, h(n, k_1) \rangle \qquad \begin{array}{lr} \text{if } \pi_2(\text{input}) = h(\pi_1(\text{input}), k_1) & \text{then ok()} \\ \text{else if } \pi_2(\text{input}) = h(\pi_1(\text{input}), k_2) & \text{then ok()} \\ \text{else} & \text{ko()} \end{array}$$

where input $=$ att(frame), and frame is the sequence of all messages sent over the network.

CCSA Approach: Formulas

We model security properties using **formulas**, which are built using a predicate $\sim$ of arbitrary arity.

$$\phi ::= \phi \vee \phi \mid \neg\phi \mid \exists x, \phi \mid \vec{u} \sim \vec{v}$$

Basic Hash Unlinkability (weak version)

Weak unlinkability for two tags and **two sessions**:

$$\langle \mathsf{n}, \mathsf{h}(\mathsf{n}, \mathsf{k}_0)\rangle, \langle \mathsf{n}', \mathsf{h}(\mathsf{n}', \mathsf{k}_0)\rangle \sim \langle \mathsf{n}, \mathsf{h}(\mathsf{n}, \mathsf{k}_0)\rangle, \langle \mathsf{n}', \mathsf{h}(\mathsf{n}', \mathsf{k}_1)\rangle$$

CCSA Approach: Semantics

Models of our logic are called computational models, where a computational model $\mathbb{M}$ interprets:

- **terms** as **PPT Turing machines**:
  - names as independent random samplings;
  - function symbols as deterministic machines.
- $\sim$ as **computational indistinguishability**.

Validity

We note $\mathbb{M} \models \phi$ when the base logic formula holds in the computational model $\mathbb{M}$.

A base logic formula ϕ is **valid** if $\mathbb{M} \models \phi$ for every $\mathbb{M}$.

CCSA Approach: Axioms

To prove that ϕ is **valid**, we **axiomatize** what the adversary **cannot do**. We restrict the models $\mathbb{M}$ using axioms Ax:

- **structural axioms**;
- **implementation axioms**, e.g. functional properties;
- **cryptographic axioms**, e.g. EUF-CMA, PRF.

Axioms are given as **inference rules**.

$$\frac{\Delta_1 \vdash \phi_1 \ldots \Delta_1 \vdash \phi_1}{\Delta \vdash \phi}$$

CCSA Approach: Axioms

$$\frac{\dots}{\phi}\,\mathsf{Ax} \quad + \quad \text{Ax } \textbf{sound} \text{ (under crypto. assumptions)} \quad \rightarrow \quad \phi \text{ } \textbf{valid} \text{ (under crypto. assumptions)}$$

CCSA Approach: Examples of Axioms

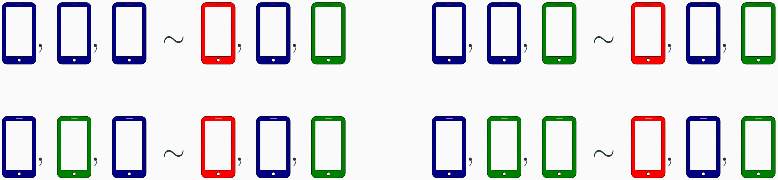
$$\text{DUP}\ \frac{\Delta \vdash \vec{u}, s \sim \vec{v}, t}{\Delta \vdash \vec{u}, s, s \sim \vec{v}, t, t}$$

$$\text{PRF}\ \frac{\Delta \vdash \vec{u}, \begin{array}{l}\text{if } \mathsf{HFresh}^{\mathsf{k}}(t; \vec{u}, t) \\ \text{then } \mathsf{n} \\ \text{else } \mathsf{h}(t, \mathsf{k})\end{array} \sim \vec{v}}{\Delta \vdash \vec{u}, \mathsf{h}(t, \mathsf{k}) \sim \vec{v}}$$

when $\mathsf{SC}^{\mathsf{n},\mathsf{k}}(t, \vec{u})$

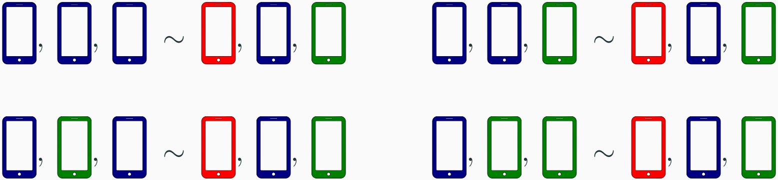
CCSA Approach: Limitations

Weak unlinkability for two tags A,B with **three sessions**:



CCSA Approach: Limitations

Weak unlinkability for two tags A,B with **three sessions**:



We have to **manually prove all these equivalences!**

- **Limited guarantees**: only proved for three sessions.
- Lots of **redundant** reasoning between cases.

Moreover, to prove security for a **fixed** n, e.g. 3, we often need to understand why security holds **for any** n.

The Meta-Logic

Solution: a Meta-Logic

Goal: prove security **for any interleaving**, in a single proof.

$$\begin{Bmatrix} \phi_1 & \phi_2 \\ \phi_3 & \phi_4 \\ \dots & \end{Bmatrix} + \begin{Bmatrix} \text{base logic} \\ \text{inference rules} \end{Bmatrix} \rightarrow \begin{Bmatrix} \text{proof}_1 & \text{proof}_2 \\ \text{proof}_3 & \text{proof}_4 \\ \dots & \end{Bmatrix}$$

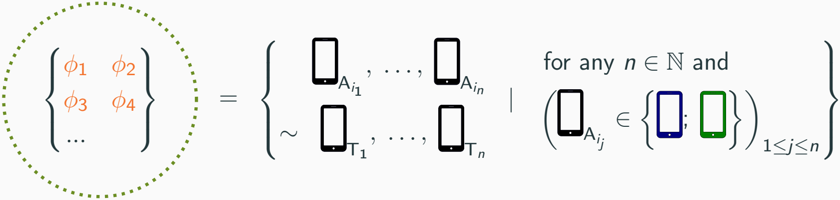
Solution: a Meta-Logic

Goal: prove security **for any interleaving**, in a single proof.

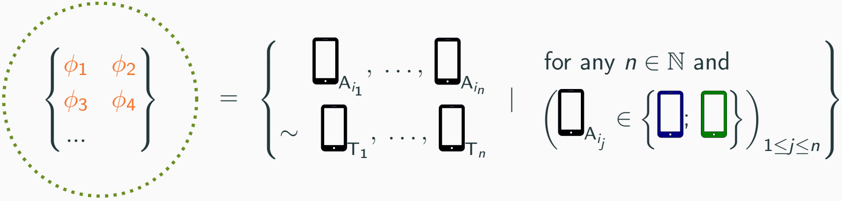
$$\left\{\begin{matrix}\phi_1 & \phi_2 \\ \phi_3 & \phi_4 \\ \dots & \end{matrix}\right\} + \left\{\begin{matrix}\text{base logic}\\ \text{inference rules}\end{matrix}\right\} \rightarrow \left\{\begin{matrix}\text{proof}_1 & \text{proof}_2\\ \text{proof}_3 & \text{proof}_4 \\ \dots & \end{matrix}\right\}$$

$$\psi + \left\{\begin{matrix}\text{meta-logic}\\ \text{inference rules}\end{matrix}\right\} \rightarrow \text{proof}$$

Solution: a Meta-Logic



Solution: a Meta-Logic



$$\psi \quad = \quad \forall \tau,\ \mathsf{frame}@\tau \sim \mathsf{frame}^{\mathsf{u}}@\tau$$

Protocols as a Set of Actions

To do this, we need a **formal description of protocols**.

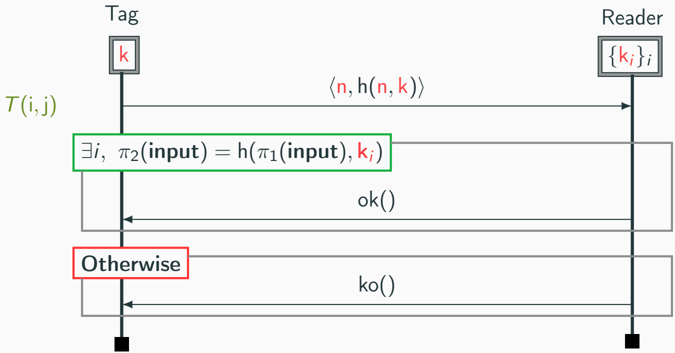
An Action is:

- a **condition**,
- and an **output** message.

A Protocol is:

- a finite set of **actions**,
- equipped with a **dependency relation** to constrain the execution order of actions.

Protocols as a Set of Actions

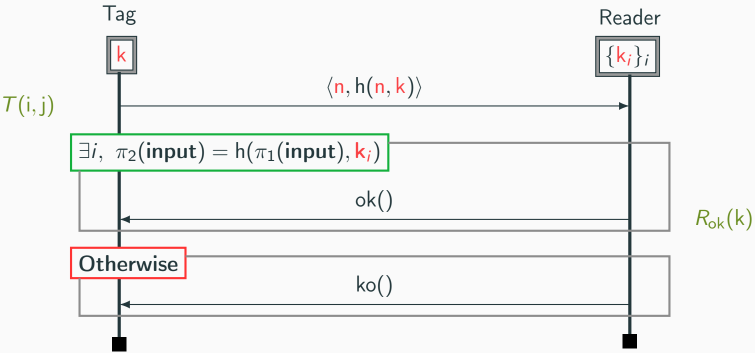


Action $T(i, j)$: session j of the tag i

Condition: true

Output: $\langle n[i, j], h(n[i, j], k[i]) \rangle$

Protocols as a Set of Actions

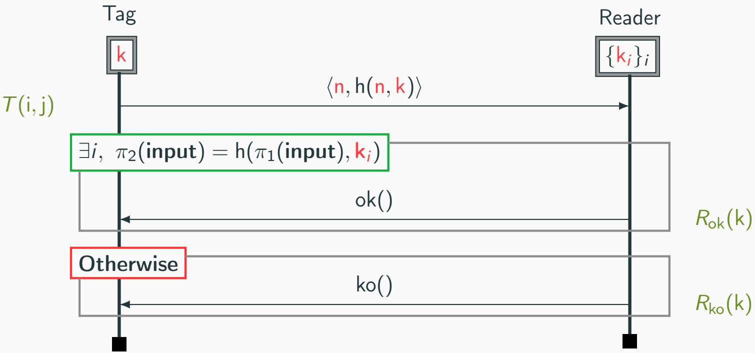


Action $R_{ok}[k]$: session k of the reader, accept

Condition: $\exists i, \pi_2(\text{input}@R_{ok}[k]) = h(\pi_1(\text{input}@R_{ok}[k]), k[i]))$

Output: ok()

Protocols as a Set of Actions

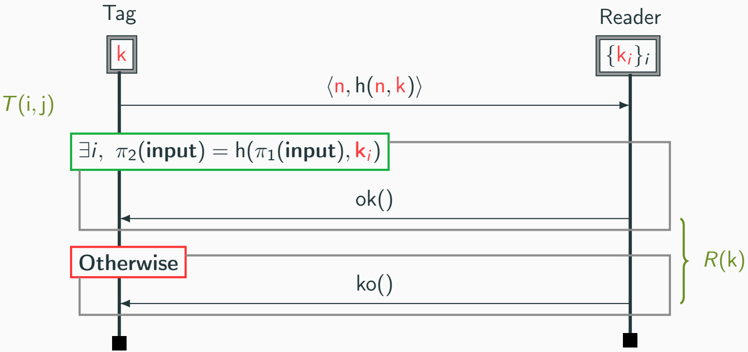


Action $R_{ko}[k]$: session k of the reader, reject

Condition: $\neg(\exists i, \pi_2(\text{input}@R_{ko}[k]) = h(\pi_1(\text{input}@R_{ko}[k]), k[i]))$

Output: ko()

Protocols as a Set of Actions



<u>Bonus</u> **Action** $R[k]$: session k of the reader, accept or reject

Condition: true

Output: find i s.t. $\pi_2(\text{input}@R[k]) = h(\pi_1(\text{input}@R[k]), k[i]))$ then ok()

else ko()

Meta-Logic: Syntax

Extension of the base logic with:

- **index variables** $\mathcal{I}$, to represent session numbers, agent numbers, etc: $\mathsf{i}_1, \dots, \mathsf{i}_n$
- **indexed names**, e.g. $\mathsf{n}[\mathsf{i}_1, \dots, \mathsf{i}_k]$
- **timestamps variables** $\mathcal{T}$ **and terms**, to quantify over all possible instants of a trace: τ or $T(\mathsf{i}, \mathsf{j})$.
- **macros** $\mathsf{cond}@\tau, \mathsf{input}@\tau, \mathsf{output}@\tau$ to talk about the condition, input and output of the action at instant τ
- **quantifications** over timestamps and indices.

Meta-Logic: Translation to the Base Logic

A meta-formula ψ represents a set of base-formulas. Roughly:

$$\psi \quad \text{represents} \quad \left\{ (\psi)^{\mathbb{T}} \mid \text{ for any "trace" } \mathbb{T} \right\}$$

Trace Model $\mathbb{T}$

A **trace model** $\mathbb{T}$ is a tuple $(\mathcal{D}_{\mathcal{I}}, \mathcal{D}_{\mathcal{T}}, <_{\mathcal{T}}, \sigma_{\mathcal{I}}, \sigma_{\mathcal{T}})$:

- $\mathcal{D}_{\mathcal{I}}, \mathcal{D}_{\mathcal{T}}$ are index and timestamp domains;
- $<_{\mathcal{T}}$ is a total ordering on $\mathcal{D}_{\mathcal{T}}$;
- $\sigma_{\mathcal{I}} : \mathcal{I} \rightarrow D_{\mathcal{I}}$ interprets index variables;
- $\sigma_{\mathcal{T}} : \mathcal{T} \rightarrow D_{\mathcal{T}}$ interprets timestamp variables.

Translation function $(\_)^{\mathbb{T}}$ from meta-formulas and terms to base-formulas and terms.

Meta-Logic: Translation to the Base Logic, an Example

Let's consider a trace of the Basic Hash protocol: $T[3, 1].R_{\mathsf{ok}}[2].T[3, 2]$.

- $\mathcal{D}_{\mathcal{I}} = \{1, 2, 3\}$.
- $\sigma_{\mathcal{I}} = \{\mathsf{i} \mapsto 3, \mathsf{j} \mapsto 1, \mathsf{j}' \mapsto 2, \mathsf{k} \mapsto 2\}$

Meta-Logic: Translation to the Base Logic, an Example

Let's consider a trace of the Basic Hash protocol: $T[3,1].R_{\mathsf{ok}}[2].T[3,2]$.

- $\mathcal{D}_{\mathcal{I}} = \{1, 2, 3\}$.
- $\sigma_{\mathcal{I}} = \{\mathsf{i} \mapsto 3, \mathsf{j} \mapsto 1, \mathsf{j}' \mapsto 2, \mathsf{k} \mapsto 2\}$

- $(\mathsf{n}[\mathsf{i}, \mathsf{j}])^{\mathbb{T}} := \mathsf{n}_{3,1}$
- $(\mathsf{n}[\mathsf{i}, \mathsf{j}'])^{\mathbb{T}} := \mathsf{n}_{3,2}$
- $(\mathsf{output}@T[\mathsf{i}, \mathsf{j}])^{\mathbb{T}} := \langle \mathsf{n}_{3,1}, \mathsf{h}(\mathsf{n}_{3,1}, \mathsf{k}_3)\rangle$

Meta-Logic: Translation to the Base Logic, an Example

Let's consider a trace of the Basic Hash protocol: $T[3,1].R_{\mathsf{ok}}[2].T[3,2]$.

- $\mathcal{D}_{\mathcal{I}} = \{1,2,3\}$.
- $\sigma_{\mathcal{I}} = \{\mathsf{i} \mapsto 3, \mathsf{j} \mapsto 1, \mathsf{j}' \mapsto 2, \mathsf{k} \mapsto 2\}$

- $(\mathsf{n}[\mathsf{i},\mathsf{j}])^{\mathbb{T}} := \mathsf{n}_{3,1}$
- $(\mathsf{n}[\mathsf{i},\mathsf{j}'])^{\mathbb{T}} := \mathsf{n}_{3,2}$
- $(\mathsf{output}@T[\mathsf{i},\mathsf{j}])^{\mathbb{T}} := \langle \mathsf{n}_{3,1}, \mathsf{h}(\mathsf{n}_{3,1}, \mathsf{k}_3)\rangle$
- $(\mathsf{cond}@R_{\mathsf{ok}}[\mathsf{k}])^{\mathbb{T}}$

$$
\begin{aligned}
&:= \ (\exists \mathsf{i}, \pi_2(\mathsf{input}@R[\mathsf{k}]) = \mathsf{h}(\pi_1(\mathsf{input}@R[\mathsf{k}]), \mathsf{k}[\mathsf{i}]))^{\mathbb{T}} \\
&:= \quad\ \pi_2(\mathsf{att}(\ldots)) = \mathsf{h}(\pi_1(\mathsf{att}(\ldots)), \mathsf{k}_1) \\
&\qquad \dot\vee \ \pi_2(\mathsf{att}(\ldots)) = \mathsf{h}(\pi_1(\mathsf{att}(\ldots)), \mathsf{k}_2) \\
&\qquad \dot\vee \ \pi_2(\mathsf{att}(\ldots)) = \mathsf{h}(\pi_1(\mathsf{att}(\ldots)), \mathsf{k}_3)
\end{aligned}
$$

Meta-Logic: Translation to the Base Logic, Some Details

Selected (simplified) rules:

$$(f(t_1, \ldots, t_n))^{\mathbb{T}} = f\left((t_1)^{\mathbb{T}}, \ldots, (t_n)^{\mathbb{T}}\right)$$

$$(\exists \mathsf{i}.\ \phi)^{\mathbb{T}} = \dot{\bigvee}_{k \in \mathcal{D}_{\mathcal{I}}} (\phi)^{\mathbb{T}\{\mathsf{i} \mapsto k\}}$$

$$(\forall \tau.\ \phi)^{\mathbb{T}} = \dot{\bigwedge}_{v \in \mathcal{D}_{\mathcal{T}}} (\phi)^{\mathbb{T}\{\tau \mapsto v\}}$$

Meta-Logic: Translation to the Base Logic, Some Details

Selected (simplified) rules:

$$(f(t_1,\dots,t_n))^{\mathbb{T}} = f\left((t_1)^{\mathbb{T}},\dots,(t_n)^{\mathbb{T}}\right)$$

$$(\exists \mathsf{i}.\ \phi)^{\mathbb{T}} = \dot{\bigvee}_{k\in\mathcal{D}_{\mathcal{I}}}(\phi)^{\mathbb{T}\{\mathsf{i}\mapsto k\}}$$

$$(\forall \tau.\ \phi)^{\mathbb{T}} = \dot{\bigwedge}_{v\in\mathcal{D}_{\mathcal{T}}}(\phi)^{\mathbb{T}\{\tau\mapsto v\}}$$

and for macros:

$$\begin{aligned}
(\mathsf{output}@\tau)^{\mathbb{T}} &= \textit{specified by the protocol}\\
(\mathsf{cond}@\tau)^{\mathbb{T}} &= \textit{specified by the protocol}\\
(\mathsf{input}@\tau)^{\mathbb{T}} &= \mathsf{att}\left((\mathsf{frame}@\mathsf{pred}(\tau))^{\mathbb{T}}\right)\\
(\mathsf{frame}@\tau)^{\mathbb{T}} &\approx \langle(\mathsf{frame}@\mathsf{pred}(\tau))^{\mathbb{T}},(\mathsf{output}@\tau)^{\mathbb{T}}\rangle
\end{aligned}$$

Meta-Logic: Semantics

Validity: Meta-Logic

We note $\mathbb{T}, \mathbb{M} \models \psi$ when the meta-logic formula ψ holds in trace model $\mathbb{T}$ and in the computational model $\mathbb{M}$:

$$\mathbb{T}, \mathbb{M} \models \psi \quad \text{iff.} \quad \mathbb{M} \models (\psi)^{\mathbb{T}}$$

A meta-logic formula ψ is **valid** if $(\psi)^{\mathbb{T}}$ is valid for every $\mathbb{T}$.

Meta-Logic: Lifting Axioms

Base logic rule

$$\text{DUP}\ \frac{\Delta \vdash \vec{u}, s \sim \vec{v}, t}{\Delta \vdash \vec{u}, s, s \sim \vec{v}, t, t}$$

Meta-logic rule

$$\text{DUP}\ \frac{\Delta \vdash \vec{u}, s \sim \vec{v}, t}{\Delta \vdash \vec{u}, s, s \sim \vec{v}, t, t}$$

Meta-Logic: Lifting Axioms

Base logic rule

$$\frac{\Delta \vdash \vec{u},\ \begin{array}{l}\text{if } \mathsf{HFresh}^{\mathsf{k}}(t; \vec{u}, t)\\ \text{then } \mathsf{n}\\ \text{else } \mathsf{h}(t, \mathsf{k})\end{array} \sim \vec{v}}{\Delta \vdash \vec{u}, \mathsf{h}(t, \mathsf{k}) \sim \vec{v}}\ \mathrm{PRF}$$

when $\mathsf{SC}^{\mathsf{n},\mathsf{k}}(t, \vec{u})$

Meta-logic rule

$$\frac{\Delta \vdash \vec{u},\ \begin{array}{l}\text{if } \mathsf{HFresh}^{\mathsf{k}[\vec{i}]}_{\mathcal{P}}(t; \vec{u}, t)\\ \text{then } \mathsf{n}\\ \text{else } \mathsf{h}(t, \mathsf{k}[\vec{i}])\end{array} \sim \vec{v}}{\Delta \vdash \vec{u}, \mathsf{h}(t, \mathsf{k}[\vec{i}]) \sim \vec{v}}\ \mathrm{PRF}$$

when $\mathsf{SC}^{\mathsf{n},\mathsf{k}[\vec{i}]}_{\mathcal{P}}(t, \vec{u})$

Meta-Logic: Lifting Axioms

Base logic rule

$$\text{PRF}\ \frac{\Delta \vdash \vec{u},\ \begin{array}{l}\text{if } \mathsf{HFresh}^{\mathsf{k}}(t; \vec{u}, t) \\ \text{then } \mathsf{n} \\ \text{else } \mathsf{h}(t, \mathsf{k})\end{array} \sim \vec{v}}{\Delta \vdash \vec{u}, \mathsf{h}(t, \mathsf{k}) \sim \vec{v}}$$

when $\mathsf{SC}^{\mathsf{n},\mathsf{k}}(t, \vec{u})$

$\mathsf{HFresh}^{\mathsf{k}}(t; \vec{u}, t)$ and $\mathsf{SC}^{\mathsf{n},\mathsf{k}}(t, \vec{u})$ can be checked/computed syntactically.

Meta-logic rule

$$\text{PRF}\ \frac{\Delta \vdash \vec{u},\ \begin{array}{l}\text{if } \mathsf{HFresh}_{\mathcal{P}}^{\mathsf{k}[\vec{i}]}(t; \vec{u}, t) \\ \text{then } \mathsf{n} \\ \text{else } \mathsf{h}(t, \mathsf{k}[\vec{i}])\end{array} \sim \vec{v}}{\Delta \vdash \vec{u}, \mathsf{h}(t, \mathsf{k}[\vec{i}]) \sim \vec{v}}$$

when $\mathsf{SC}_{\mathcal{P}}^{\mathsf{n},\mathsf{k}[\vec{i}]}(t, \vec{u})$

$\mathsf{HFresh}_{\mathcal{P}}^{\mathsf{k}[\vec{i}]}(t; \vec{u}, t)$ and $\mathsf{SC}_{\mathcal{P}}^{\mathsf{n},\mathsf{k}[\vec{i}]}(t, \vec{u})$ need to be checked/computed for:
- **direct** occurrences (syntactically),
- and **indirect** occurrences (any action of the protocol).

Meta-Logic: Basic Hash Protocol Security

Using the meta-logic inference rules, we are able to derive all at once a family of base logic formulas.

(Do you recall the long list of equivalences shown previously?)

It starts like this:

$$\dfrac{\dfrac{\dfrac{\cdots}{\tau = T[\mathrm{i},\mathrm{j}], \ldots \vdash \ldots} \qquad \dfrac{\cdots}{\tau = R_{\mathrm{ok}}[\mathrm{k}], \ldots \vdash \ldots} \qquad \dfrac{\cdots}{\tau = R_{\mathrm{ko}}[\mathrm{k}], \ldots \vdash \ldots}}{\mathsf{frame}@\mathsf{pred}(\tau) \sim \mathsf{frame}^{\mathsf{u}}@\mathsf{pred}(\tau) \vdash \mathsf{frame}@\tau \sim \mathsf{frame}^{\mathsf{u}}@\tau}}{\vdash \mathsf{frame}@\tau \sim \mathsf{frame}^{\mathsf{u}}@\tau}\ \text{Ind}$$

An Interactive Theorem Prover, Squirrel

The Tool

- The **input language** is a variant of the applied-pi calculus.
- We have implemented:
  - the **translation** of the specification of the protocol from this input language to actions,
  - **proof tactics**, corresponding to inference rules,
  - **automated reasoning** to ease the proof effort.
- The **user** interacts with the prover by **calling tactics** to derive formulas step by step.

Case Studies

Protocol	Crypto. assumptions	Security properties
Basic Hash	PRF, EUF-CMA	Authentication & Unlinkability
Hash Lock	PRF, EUF-CMA	Authentication & Unlinkability
LAK (pairs)	PRF, EUF-CMA	Authentication & Unlinkability
MW	PRF, EUF-CMA, XOR	Authentication & Unlinkability
Feldhofer	CCA_1, PRF, EUF-CMA	Authentication & Unlinkability
Private Auth.	CCA_1, EUF-CMA, ENC-KP	Anonymity
Signed DDH	EUF-CMA, DDH	Authentication & Strong Secrecy
Additional case studies, using the composition framework from [CJS20]		
Signed DDH	EUF-CMA, DDH	Authentication & Strong Secrecy
SSH (fwd agent)	EUF-CMA, DDH	Authentication & Strong Secrecy

Basic Hash Protocol

[demo]

Conclusion

Our contribution

- **Meta-logic** built on the **CCSA model**.
- Set of **meta-logic inference rules** for proving reachability and equivalence properties.
- SQUIRREL, an interactive prover **implementing** these inference rules, used on various **case studies**.

Conclusion

Our contribution

- **Meta-logic** built on the **CCSA model**.
- Set of **meta-logic inference rules** for proving reachability and equivalence properties.
- SQUIRREL, an interactive prover **implementing** these inference rules, used on various **case studies**.

Current and future work

- Extend support to **stateful** and **more complex** protocols.
- More **proof automation**.

Conclusion

Our contribution

- **Meta-logic** built on the **CCSA model**.
- Set of **meta-logic inference rules** for proving reachability and equivalence properties.
- SQUIRREL, an interactive prover **implementing** these inference rules, used on various **case studies**.

Current and future work

- Extend support to **stateful** and **more complex** protocols.
- More **proof automation**.

Thank you for your attention!

References

Gergei Bana and Hubert Comon-Lundh. "Towards Unconditional Soundness: Computationally Complete Symbolic Attacker". In: *Principles of Security and Trust - First International Conference, POST 2012, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2012, Tallinn, Estonia, March 24 - April 1, 2012, Proceedings*. Ed. by Pierpaolo Degano and Joshua D. Guttman. Vol. 7215. Lecture Notes in Computer Science. Springer, 2012, pp. 189–208.

Gergei Bana and Hubert Comon-Lundh. "A Computationally Complete Symbolic Attacker for Equivalence Properties". In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*. Ed. by Gail-Joon Ahn, Moti Yung, and Ninghui Li. ACM, 2014, pp. 609–620.

Hubert Comon, Charlie Jacomme, and Guillaume Scerri. *Oracle simulation: a technique for protocol composition with long term shared secrets*. Research Report. INRIA ; LSV, ENS Paris Saclay, Université Paris-Saclay ; Université Versailles Saint-Quentin, Aug. 2020. url: `https://hal.inria.fr/hal-02913866`.

Hubert Comon and Adrien Koutsos. "Formal Computational Unlinkability Proofs of RFID Protocols". In: *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*. IEEE Computer Society, 2017, pp. 100–114.

Hubert Comon-Lundh, Véronique Cortier, and Guillaume Scerri. "A tool for automating the computationally complete symbolic attacker (Extended Abstract)". In: *Joint Workshop on Foundations of Computer Security and Formal and Computational Cryptography (FCS-FCC'14)*. Vienne, Austria, July 2014. url: `https://hal.inria.fr/hal-01080296`.

Adrien Koutsos. "Decidability of a Sound Set of Inference Rules for Computational Indistinguishability". In: *32nd IEEE Computer Security Foundations Symposium, CSF 2019, Hoboken, NJ, USA, June 25-28, 2019*. IEEE, 2019, pp. 48–61.