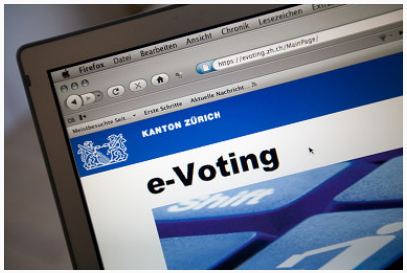
Reducing the number of sessions: a long-standing research question

Stéphanie DELAUNE, Univ Rennes, CNRS, IRISA

Workshop in honour of Véronique Cortier - January 31, 2023

Security protocols are everywhere !



Cryptographic protocols

Distributed programs which aims at providing some security properties relying on cryptography.

Security properties

Two main families of security properties:

- Reachability properties, *e.g.* weak secrecy, authentication, ...

  Is it possible to reach a bad state?

- Equivalence properties, *e.g.* anonymity, unlinkability, ...

  Is the attacker able to distinguish between two situations?

$\longrightarrow$ In this talk, we will consider both kinds of security properties.

Verification of security protocols

What did we know in the early 2000s?

- A problem well-known to be undecidable when considering an unbounded number of sessions. [Durgin *et al.*,99]
- Decision procedures dedicated to a bounded number of sessions exist. [Rusinowitch & Turuani, CSFW'01]

Observation:

known attacks only involve very few sessions (less than 6)

Verification of security protocols

What did we know in the early 2000s?

- A problem well-known to be undecidable when considering an unbounded number of sessions. [Durgin *et al.*,99]
- Decision procedures dedicated to a bounded number of sessions exist. [Rusinowitch & Turuani, CSFW'01]

Observation:

known attacks only involve very few sessions (less than 6)

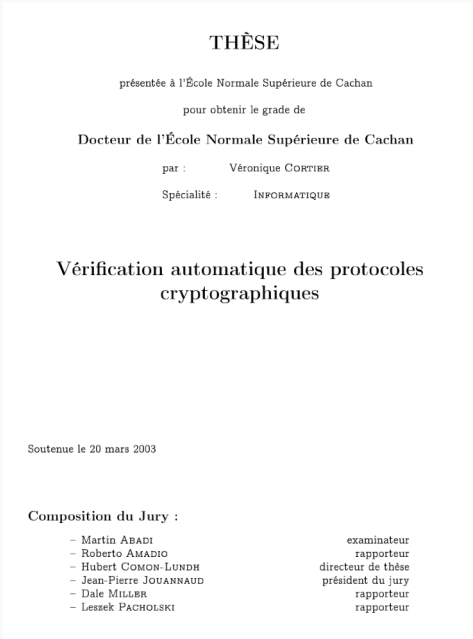
Long-standing research question

Identify criteria to bound a priori the number of sessions involved in an attack.

Vero's PhD thesis (2003)

In a nutshell, the problem remains undecidable even if:

- we bound the number of sessions in parallel
- we bound the depth of messages
- . . .



THÈSE

présentée à l'École Normale Supérieure de Cachan

pour obtenir le grade de

Docteur de l'École Normale Supérieure de Cachan

par : Véronique Cortier

Spécialité : Informatique

Vérification automatique des protocoles cryptographiques

Soutenue le 20 mars 2003

Composition du Jury :

- Martin Abadi — examinateur
- Roberto Amadio — rapporteur
- Hubert Comon-Lundh — directeur de thèse
- Jean-Pierre Jouannaud — président du jury
- Dale Miller — rapporteur
- Leszek Pacholski — rapporteur

Not so encouraging . . .

. . . despite Veronique's appetite and Hubert's enthusiasm.

. . . especially as they were looking for a very small bound!

> Un résultat de réduction sur les sessions permettrait d'obtenir des résultats de décidabilité dans le cadre d'un nombre *a priori* non borné de sessions. D'autre part, ce résultat serait immédiatement exploitable par les outils qui vérifient les protocoles pour un nombre fixé de sessions, comme ceux décrits au paragraphe 7.2.3. Il est bien sûr impossible de borner le nombre des sessions *en général* puisque, par exemple, le secret est indécidable.

At this time, existing tools were only able to consider very few sessions (typically 2 or 3).

. . . especially as they were looking for a very small bound!

> Un résultat de réduction sur les sessions permettrait d'obtenir des résultats de décidabilité dans le cadre d'un nombre *a priori* non borné de sessions. D'autre part, ce résultat serait immédiatement exploitable par les outils qui vérifient les protocoles pour un nombre fixé de sessions, comme ceux décrits au paragraphe 7.2.3. Il est bien sûr impossible de borner le nombre des sessions *en général* puisque, par exemple, le secret est indécidable.

At this time, existing tools were only able to consider very few sessions (typically 2 or 3).

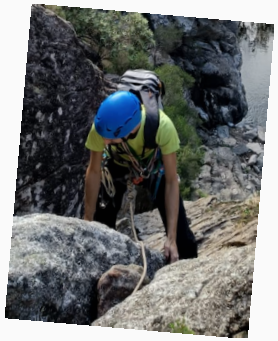
Some tools mentioned in Paragraphe 7.2.3:

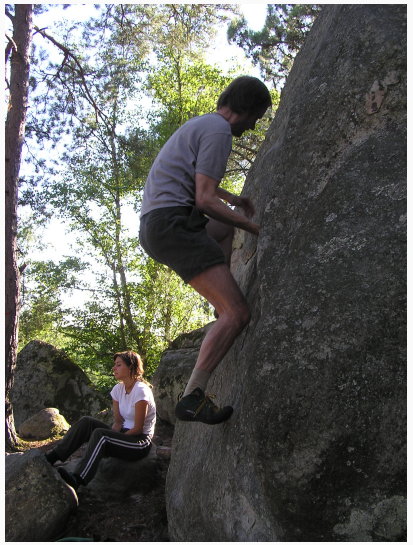
Casper/FDR, Mur$_\phi$, Athena, Casrul, Aviss, . . .

We like challenging problems








Our contributions

In Proceedings of the 35th IEEE Computer Security Foundations Symposium (CSF'22), IEEE Computer Society Press, Haifa, Israel, August 2022.

A small bound on the number of sessions for security protocols

Véronique Cortier
Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

Antoine Dallon
DGA MI, Bruz, France

Stéphanie Delaune
Univ Rennes, CNRS, IRISA, France

***Abstract*—Bounding the number of sessions is a long-standing problem in the context of security protocols. It is well known that even simple properties like secrecy are undecidable when an unbounded number of sessions is considered. Yet, attacks on existing protocols only require a few sessions.**

In this paper, we propose a sound algorithm that computes a sufficient set of scenarios that need to be considered to detect an attack. Our approach can be applied for both reachability and equivalence properties, for protocols with standard primitives that are type-compliant (unifiable messages have the same type). Moreover, when equivalence properties are considered, else branches are disallowed, and protocols are supposed to be simple (an attacker knows from which role and session a message comes from). Since this class remains undecidable, our algorithm may return an infinite set. However, our experiments show that on most basic protocols of the literature, our algorithm computes a small number of sessions (a dozen). As a consequence, tools for a bounded number of sessions like DeepSec can then be used to conclude that a protocol is secure for an unbounded number of sessions.

an honest client C and an honest server S that each runs 3 sub-programs, yielding 6 sessions (or even less, depending on how programs are divided). The traceability attack on electronic passports [13] requires one honest run between the reader and the passport and then a replay against a passport, thus 3 sessions. Hence a tempting heuristic is to conclude that either an attack can be found within a few sessions, or the protocol is secure. Unfortunately, there is absolutely no formal guarantee that this is indeed the case. It is possible to construct protocols for which an arbitrary number of sessions can be needed for attacks. Hence, bounding the number of sessions is a long-standing problem. The goal is to identify criteria, achieved in practice, such that if there is an attack, then there is an attack within an *a priori* bounded number of sessions.

Related work. Several results have studied this question.

- Sybille Fröschle [27] proposes a decidability result for the "leakiness" property, that guarantees that all data are

Our contributions

Main result

A practical bound allowing us to rely on some existing tools to perform the security analysis.

1. We prove the correctness of our bound assuming some hypotheses (type-compliance and acyclic dependencies).
2. We implement our algorithm in HowMany to compute and generate the scenarios that need to be analysed.
3. We perform various case studies on protocols from the literature using HowMany + DeepSec/SAT-Equiv.

$\longrightarrow$ This result holds for both for **reachability** and **equivalence properties**.

A long journey . . .

Arapinis & Duflot, FST&TCS'07: typing result (weak secrecy) and still no bound on the number of sessions

A long journey . . .

Chrétien & Cortier & D., CONCUR'14: typing result for equivalence based properties

Chrétien & Cortier & D., CSF'15: a bound for both reachability and equivalence based properties (but not a practical one, e.g. 10^{19})

A long journey . . .

Chrétien & Cortier & Dallon & D., TOCL'20: extension of the typing result to deal with more cryptographic primitives . . .

Cortier & Delaune & Sundararajan, JAR'21: . . . and a bound on this setting (still not practical)

Typing messages for free in security protocols

RÉMY CHRÉTIEN, Independent researcher, France
VÉRONIQUE CORTIER, LORIA, CNRS, France
ANTOINE DALLON, LSV, CNRS & ENS Paris-Saclay, Université Paris-Saclay, France
STÉPHANIE DELAUNE, Univ Rennes, CNRS, IRISA, France

A Decidable Class of Security Protocols for Both Reachability and Equivalence Properties

Véronique Cortier ✉, Stéphanie Delaune & Vaishnavi Sundararajan

Journal of Automated Reasoning **65**, 479–520 (2021) | Cite this article

A long journey . . .

Cortier & Dallon & D., CSF'22: A practical bound allowing us to rely on some existing tools to perform the security analysis.

The tools we rely on are: DeepSec and Sat-Equiv.

Some related work

Of course, in the meantime, others were working on this topic ...

- Froschle, POST'15: leakiness is decidable for well-founded protocols
  $\longrightarrow$ typed model, excludes protocols with temporary secret
- D'Osualdo & Ong & Tiu, CSF'17: the case of depth-bounded processes relying on the theory of WSTS

$\longrightarrow$ reachability properties only (mainly weak secrecy), and the bound is far too be practical.

Outline

I. Typing result

II. Reachability properties

III. Equivalence properties

Part I. Typing result

1. Modelling messages and protocols
2. Our main assumption: type-compliance
3. Typing result

Modelling messages

Cryptographic operations are modelled using function symbols ...

Example: $\Sigma = \Sigma_c \cup \Sigma_d \cup \Sigma_{test}$

$$\Sigma_c = \{\mathsf{aenc},\ \mathsf{pk},\ \mathsf{sign},\ \mathsf{vk},\ \mathsf{ok},\ \langle\ \rangle\}$$

$$\Sigma_d = \{\mathsf{adec},\ \mathsf{getmsg},\ \mathsf{proj}_1,\ \mathsf{proj}_2\}, \text{ and } \Sigma_{test} = \{\mathsf{check}\}.$$

... and their properties using a rewriting system.

Example

$$\begin{array}{rcl@{\qquad}rcl} \mathsf{adec}(\mathsf{aenc}(x,\mathsf{pk}(y)),y) & \to & x & \mathsf{proj}_1(\langle x,y\rangle) & \to & x \\ \mathsf{getmsg}(\mathsf{sign}(x,y)) & \to & x & \mathsf{proj}_2(\langle x,y\rangle) & \to & y \end{array}$$

$$\mathsf{check}(\mathsf{sign}(x,y),\mathsf{vk}(y)) \to \mathsf{ok}$$

$\longrightarrow$ we consider shaped rewriting systems allowing one to model most classical primitives.

Modelling protocols

Protocols are modelled using a process algebra.

$$
\begin{array}{lll}
P, Q := & & \\
\quad | & \mathsf{in}^{\alpha}(c, u).P & \text{input} \\
\quad | & \mathsf{out}^{\alpha}(c, u).P & \text{output} \\
\quad | & \mathsf{new}\, n.P & \text{name generation} \\
\quad | & \ldots & \\
\quad | & !P & \text{replication} \\
\quad | & \mathsf{match}\, x\, \mathsf{with}\, (u_1 \to P_1 \mid \ldots \mid u_j \to P_j) & \text{filtering}
\end{array}
$$

Modelling protocols

Protocols are modelled using a process algebra.

$$
\begin{array}{lll}
P, Q := & & \\
\quad | & \mathsf{in}^{\alpha}(c, u).P & \text{input} \\
\quad | & \mathsf{out}^{\alpha}(c, u).P & \text{output} \\
\quad | & \mathsf{new}\, n.P & \text{name generation} \\
\quad | & \ldots & \\
\quad | & !P & \text{replication} \\
\quad | & \mathsf{match}\, x\, \mathsf{with}\, (u_1 \to P_1 \mid \ldots \mid u_j \to P_j) & \text{filtering}
\end{array}
$$

Semantics

$$(\mathsf{in}^{\alpha}(c, u).P \uplus \mathcal{P};\ \phi) \xrightarrow{\mathsf{in}^{\alpha}(c,R)} (P\sigma \uplus \mathcal{P}; \phi)$$

where R is a recipe such that $R\phi\downarrow$ is a message, and $R\phi\downarrow = u\sigma$

...

A variant of the Denning Sacco protocol (1981)

$$\mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb)) \longrightarrow$$

Is this protocol a good key exchange protocol?

A variant of the Denning Sacco protocol (1981)

$$\mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb)) \longrightarrow$$

Is this protocol a good key exchange protocol? No !

A variant of the Denning Sacco protocol (1981)

$\text{aenc}(\text{sign}(k, ska), \text{pk}(ekb))$ →

Is this protocol a good key exchange protocol? No !

Description of a possible attack:

$\text{aenc}(\text{sign}(k_{AC}, ska), pk(ekc))$ →

$\text{sign}(k_{AC}, ska)$

k_{AC}

A variant of the Denning Sacco protocol (1981)

$\mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))$

Is this protocol a good key exchange protocol? No !

Description of a possible attack:

$\mathsf{aenc}(\mathsf{sign}(k_{AC}, ska), pk(ekc))$

$\mathsf{sign}(k_{AC}, ska)$

k_{AC}

$\mathsf{aenc}(\mathsf{sign}(k_{AC}, ska), \mathsf{pk}(ekb))$

Example considering a very simple scenario

Denning Sacco protocol (variant)

$$A \to B : \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))$$

$$P_{\mathsf{DS}} = \left\{ \begin{array}{ll} & !\,\mathsf{new}\, k.\mathsf{out}^{\alpha_1}(c, \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))) \\ \mid & !\,\mathsf{in}^{\beta_1}(c, \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(ekb))).\\ \mid & \mathsf{out}^{\gamma_1}(c, \mathsf{pk}(ekb)).\mathsf{out}^{\gamma_2}(c, \mathsf{vk}(ska)). \end{array} \right.$$

Example considering a very simple scenario

Denning Sacco protocol (variant)

$$A \to B : \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))$$

$$P_{\mathsf{DS}} = \left\{ \begin{array}{ll} & !\,\mathsf{new}\, k.\mathsf{out}^{\alpha_1}(c, \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))) \\ \mid & !\,\mathsf{in}^{\beta_1}(c, \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(ekb))).\boxed{\mathsf{in}^{\beta_2}(c, x)} \\ \mid & \mathsf{out}^{\gamma_1}(c, \mathsf{pk}(ekb)).\mathsf{out}^{\gamma_2}(c, \mathsf{vk}(ska)). \end{array} \right.$$

Secrecy property

To analyse secrecy of the key as received by B, it suffices to add $\boxed{\mathsf{in}^{\beta_2}(c, x)}$, and to check whether β_2 is reachable.

Hypothesis: type-compliance

We give types to any atomic data, and types are then extended to arbitrary terms:

$$\delta(\mathsf{f}(t_1, \ldots, t_n)) = \mathsf{f}(\delta(t_1), \ldots, \delta(t_n)).$$

$\longrightarrow$ the typing system is thus structure preserving.

Hypothesis: type-compliance

We give types to any atomic data, and types are then extended to arbitrary terms:

$$\delta(\mathsf{f}(t_1, \ldots, t_n)) = \mathsf{f}(\delta(t_1), \ldots, \delta(t_n)).$$

$\longrightarrow$ the typing system is thus structure preserving.

Going back to Dening Sacco

We consider $\Delta = \{\tau_{ska}, \tau_{ekb}, \tau_{ekc}, \tau_k\}$.

- $\delta(k) = \delta(x) = \tau_k$,
- $\delta(ska) = \tau_{ska}$, $\delta(skb) = \tau_{skb}$, $\delta(ekc) = \tau_{ekc}$.

$\rightarrow$ $\delta(\mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(ekb))) = \mathsf{aenc}(\mathsf{sign}(\tau_k, \tau_{ska}), \mathsf{pk}(\tau_{ekb}))$

Hypothesis: type-compliance

We give types to any atomic data, and types are then extended to arbitrary terms:

$$\delta(\mathsf{f}(t_1, \ldots, t_n)) = \mathsf{f}(\delta(t_1), \ldots, \delta(t_n)).$$

$\longrightarrow$ the typing system is thus structure preserving.

Going back to Dening Sacco

We consider $\Delta = \{\tau_{ska}, \tau_{ekb}, \tau_{ekc}, \tau_k\}$.

- $\delta(k) = \delta(x) = \tau_k$,
- $\delta(ska) = \tau_{ska}$, $\delta(skb) = \tau_{skb}$, $\delta(ekc) = \tau_{ekc}$.

$\rightarrow$ $\delta(\mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(ekb))) = \mathsf{aenc}(\mathsf{sign}(\tau_k, \tau_{ska}), \mathsf{pk}(\tau_{ekb}))$

A protocol P is type-compliant w.r.t. a typing system (Δ, δ) if any pair of unifiable encrypted subterms have the same type.

Typing result

Typing result for reachability

Let P be a protocol type-compliant w.r.t. a typing system (Δ, δ). If $P \xrightarrow{\mathsf{tr}} K$ then there exists a well-typed execution $P \xrightarrow{\mathsf{tr}'} K'$ such that tr and tr$'$ are the same up to messages.

Typing result

Typing result for reachability

Let P be a protocol type-compliant w.r.t. a typing system (Δ, δ). If $P \xrightarrow{\mathsf{tr}} K$ then there exists a well-typed execution $P \xrightarrow{\mathsf{tr}'} K'$ such that tr and tr$'$ are the same up to messages.

- This is essentially the result established by Arapinis & Duflot FST&TCS'07
- We succeed to extend this typing result to equivalence based properties (with no else branch), Chrétien, Cortier & D., CONCUR'14

$\longrightarrow$ **If there is an attack, then there is a well-typed one.**

Part II - Reachability properties

1. Our main assumptions
2. Our algorithm to compute and generate the scenarios
3. Some case studies

Main result

Theorem

Let P be a protocol type-compliant, and α be a label. If $P \xrightarrow{\ell_1} \ldots \xrightarrow{\ell_n} \mathcal{K}$ with $\mathsf{Label}(\ell_n) = \alpha$ then there exists $A \in \mathsf{dep}(\alpha)$ such that:

- $P \xrightarrow{\ell'_1} \ldots \xrightarrow{\ell'_n} \mathcal{K}'$ with $\mathsf{Label}(\ell'_n) = \alpha$, and
- $\mathsf{Label}(\ell'_1 \ldots . \ell'_n) \subseteq A$.

$\longrightarrow$ $\mathsf{dep}(\alpha)$ computes an upper bound of the actions/labels that need to be considered to reach some action labelled α.

Main result

Theorem

Let P be a protocol type-compliant, and α be a label. If $P \xrightarrow{\ell_1} \ldots \xrightarrow{\ell_n} \mathcal{K}$ with $\mathsf{Label}(\ell_n) = \alpha$ then there exists $A \in \mathsf{dep}(\alpha)$ such that:

- $P \xrightarrow{\ell'_1} \ldots \xrightarrow{\ell'_n} \mathcal{K}'$ with $\mathsf{Label}(\ell'_n) = \alpha$, and
- $\mathsf{Label}(\ell'_1 . \ldots . \ell'_n) \subseteq A$.

$\longrightarrow$ $\mathsf{dep}(\alpha)$ computes an upper bound of the actions/labels that need to be considered to reach some action labelled α.

Going back to Denning Sacco

We have that $\mathsf{dep}(\beta_2) = \{\{\beta_2, \beta_1, \alpha_1\}\}$.

$\longrightarrow$ if there is an attack, there is one involving at most 3 actions.

Computation of dep

$$\{A_1, A_2\} \otimes \{B_1, B_2\} = \{A_1 \uplus B_1,\ A_1 \uplus B_2,\ A_2 \uplus B_1,\ A_2 \uplus B_2\}$$

Computation of dep

$$\{A_1, A_2\} \otimes \{B_1, B_2\} = \{A_1 \uplus B_1,\ A_1 \uplus B_2,\ A_2 \uplus B_1,\ A_2 \uplus B_2\}$$

Sequential dependencies

- output: $\mathsf{dep}(\alpha) = \{\{\alpha\}\} \otimes \mathsf{dep}(\mathsf{pred}(\alpha))$
- input of type τ: $\mathsf{dep}(\alpha) = \{\{\alpha\}\} \otimes \mathsf{dep}(\mathsf{pred}(\alpha)) \otimes \mathsf{dep}(\tau)$.

Computation of dep

$$\{A_1, A_2\} \otimes \{B_1, B_2\} = \{A_1 \uplus B_1,\ A_1 \uplus B_2,\ A_2 \uplus B_1,\ A_2 \uplus B_2\}$$

Sequential dependencies

- output: $\mathsf{dep}(\alpha) = \{\{\alpha\}\} \otimes \mathsf{dep}(\mathsf{pred}(\alpha))$
- input of type τ: $\mathsf{dep}(\alpha) = \{\{\alpha\}\} \otimes \mathsf{dep}(\mathsf{pred}(\alpha)) \otimes \mathsf{dep}(\tau)$.

Message dependencies

$S_{\mathsf{out}}(\tau)$ explores all the possibilities to extract a term of type τ from the output occurring in $\delta(P)$ (protocol considering only types).

- atomic type: $\mathsf{dep}(\tau) = S_{\mathsf{out}}(\tau)$
- otherwise $\tau = \mathsf{f}(\tau_1, \ldots, \tau_k)$, we have that:

$$\mathsf{dep}(\tau) = S_{\mathsf{out}}(\tau) \cup \big(\mathsf{dep}(\tau_1) \otimes \ldots \otimes \mathsf{dep}(\tau_k)\big)$$

Computation of dep

$$\{A_1, A_2\} \otimes \{B_1, B_2\} = \{A_1 \uplus B_1,\ A_1 \uplus B_2,\ A_2 \uplus B_1,\ A_2 \uplus B_2\}$$

Sequential dependencies

- output: $\mathsf{dep}(\alpha) = \{\{\alpha\}\} \otimes \mathsf{dep}(\mathsf{pred}(\alpha))$
- input of type τ: $\mathsf{dep}(\alpha) = \{\{\alpha\}\} \otimes \mathsf{dep}(\mathsf{pred}(\alpha)) \otimes \mathsf{dep}(\tau)$.

Message dependencies

$S_{\mathsf{out}}(\tau)$ explores all the possibilities to extract a term of type τ from the output occurring in $\delta(P)$ (protocol considering only types).

- atomic type: $\mathsf{dep}(\tau) = S_{\mathsf{out}}(\tau) \cup \mathsf{dep}_{\mathsf{init}}(\tau)$
- otherwise $\tau = \mathsf{f}(\tau_1, \ldots, \tau_k)$, we have that:

$$\mathsf{dep}(\tau) = S_{\mathsf{out}}(\tau) \cup \big(\mathsf{dep}(\tau_1) \otimes \ldots \otimes \mathsf{dep}(\tau_k)\big) \cup \mathsf{dep}_{\mathsf{init}}(\tau)$$

with $\mathsf{dep}_{\mathsf{init}}(\tau) = \emptyset$ when τ is a cv-alien type, and $\{\emptyset\}$ otherwise.

Going back to Denning Sacco: $\mathsf{dep}(\beta_2)$

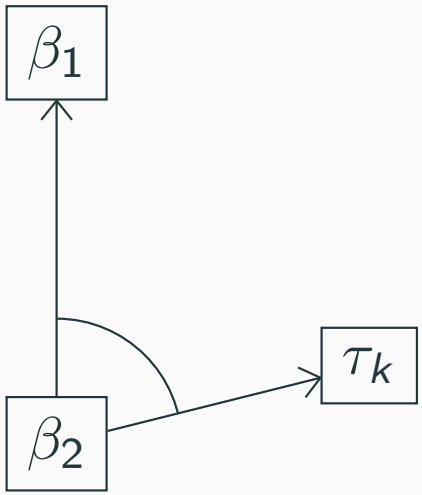
$$
\begin{aligned}
&P_{\mathsf{DS}} = \\
&!\,\mathsf{new}\ k. \\
&\mathsf{out}^{\alpha_1}(c, \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))) \\
&\mid !\,\mathsf{in}^{\beta_1}(c, \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(ekb))). \\
&\quad \mathsf{in}^{\beta_2}(c, x) \\
&\mid \mathsf{out}^{\gamma_1}(c, \mathsf{pk}(ekb)). \\
&\quad \mathsf{out}^{\gamma_2}(c, \mathsf{vk}(ska))
\end{aligned}
$$

τ $\quad \mathsf{dep}_{\mathsf{init}}(\tau) = \emptyset$

β_2

Going back to Denning Sacco: $\mathsf{dep}(\beta_2)$

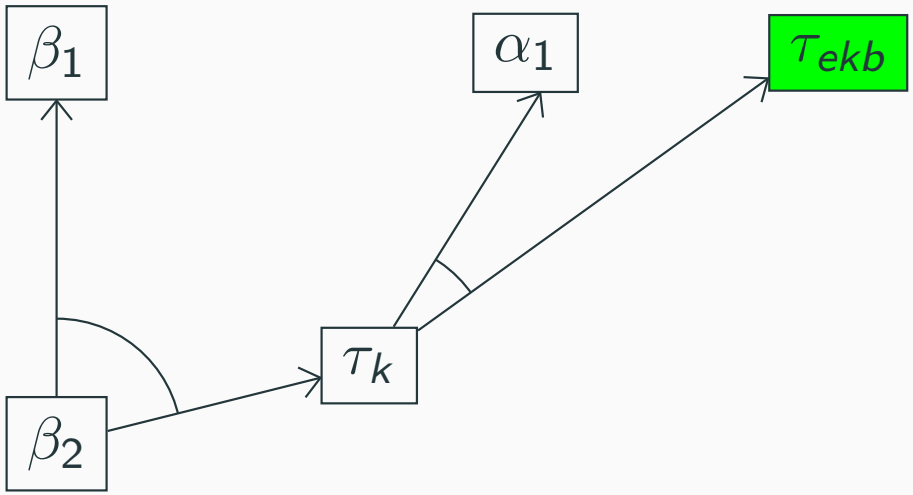
$$
\begin{aligned}
&P_{\mathsf{DS}} = \\
&!\,\mathsf{new}\ k. \\
&\mathsf{out}^{\alpha_1}(c, \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))) \\
&\mid !\,\mathsf{in}^{\beta_1}(c, \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(ekb))). \\
&\quad \mathsf{in}^{\beta_2}(c, x) \\
&\mid \mathsf{out}^{\gamma_1}(c, \mathsf{pk}(ekb)). \\
&\quad \mathsf{out}^{\gamma_2}(c, \mathsf{vk}(ska))
\end{aligned}
$$



τ $\quad \mathsf{dep}_{\mathsf{init}}(\tau) = \emptyset$

Going back to Denning Sacco: $\mathsf{dep}(\beta_2)$

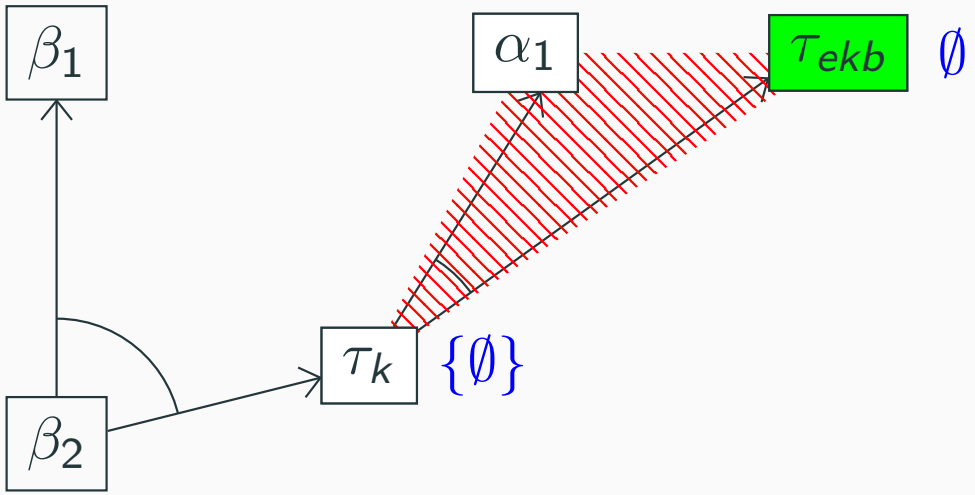
$$
\begin{aligned}
&P_{\mathsf{DS}} = \\
&!\,\mathsf{new}\ k. \\
&\mathsf{out}^{\alpha_1}(c, \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))) \\
&\mid !\,\mathsf{in}^{\beta_1}(c, \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(ekb))). \\
&\quad \mathsf{in}^{\beta_2}(c, x) \\
&\mid \mathsf{out}^{\gamma_1}(c, \mathsf{pk}(ekb)). \\
&\quad \mathsf{out}^{\gamma_2}(c, \mathsf{vk}(ska))
\end{aligned}
$$



τ $\quad \mathsf{dep}_{\mathsf{init}}(\tau) = \emptyset$

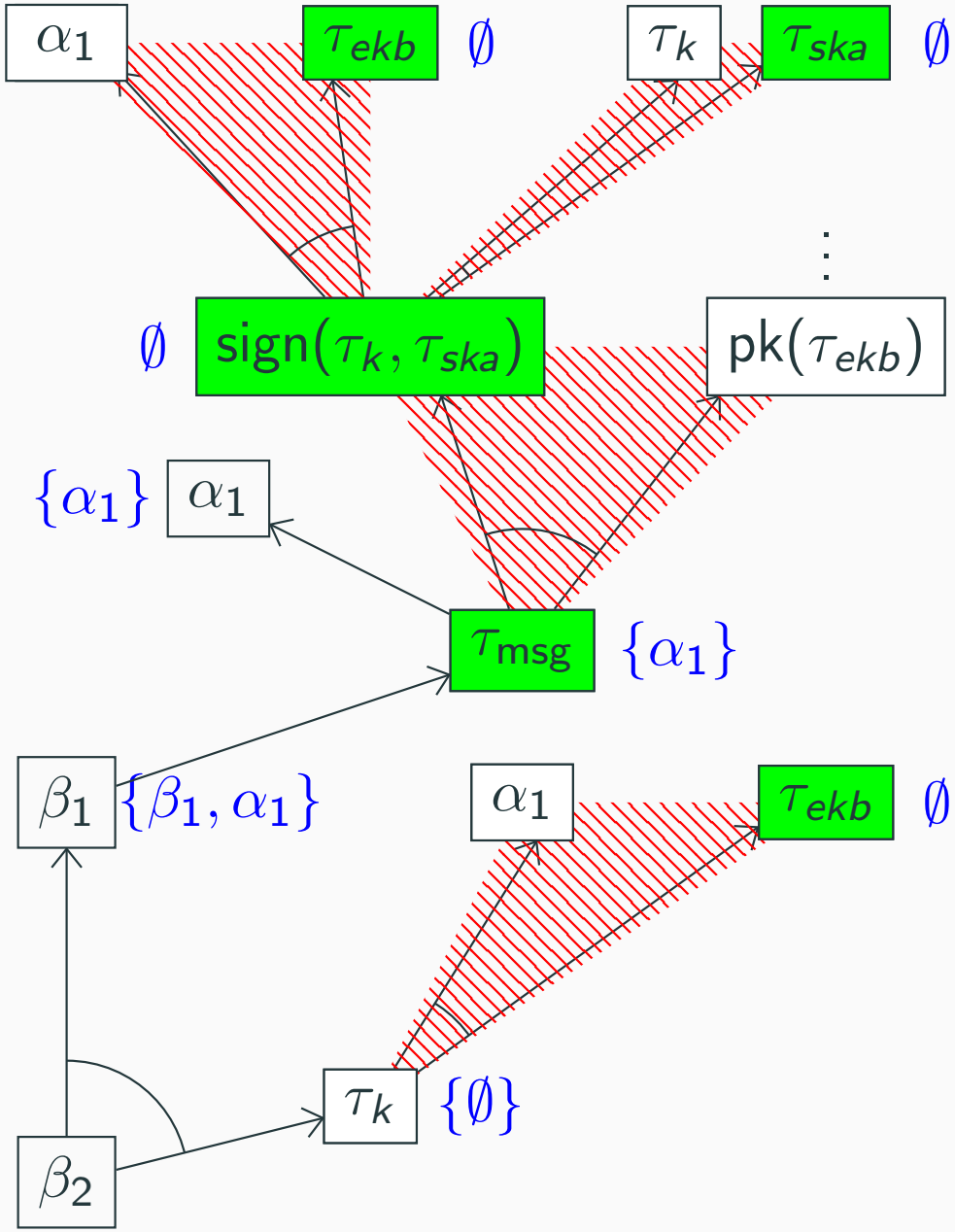
Going back to Denning Sacco: $\mathsf{dep}(\beta_2)$

$$
\begin{aligned}
&P_{\mathsf{DS}} = \\
&!\,\mathsf{new}\ k. \\
&\mathsf{out}^{\alpha_1}(c, \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))) \\
&\mid !\,\mathsf{in}^{\beta_1}(c, \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(ekb))). \\
&\quad \mathsf{in}^{\beta_2}(c, x) \\
&\mid \mathsf{out}^{\gamma_1}(c, \mathsf{pk}(ekb)). \\
&\quad \mathsf{out}^{\gamma_2}(c, \mathsf{vk}(ska))
\end{aligned}
$$



τ $\quad \mathsf{dep}_{\mathsf{init}}(\tau) = \emptyset$

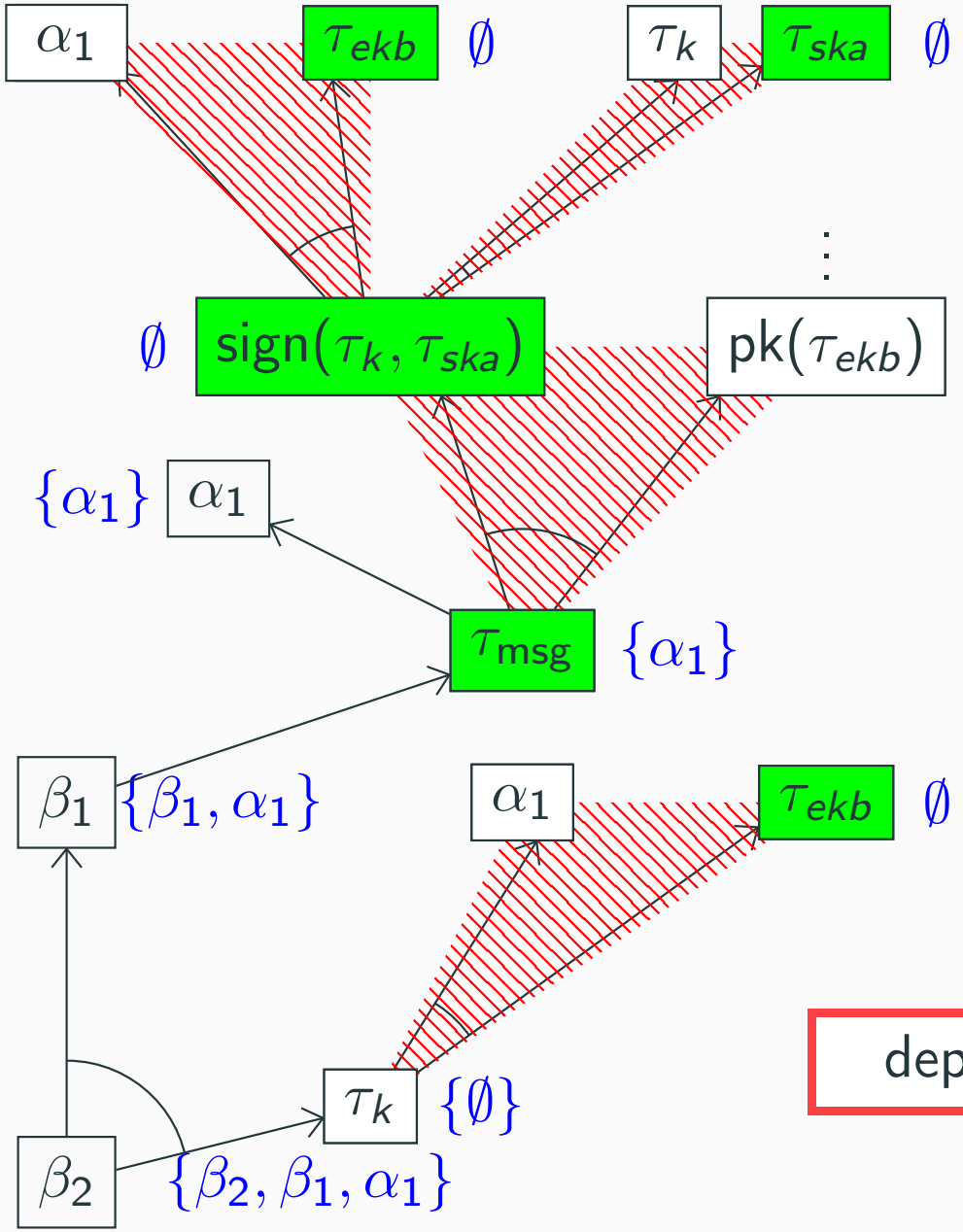
Going back to Denning Sacco: $\mathsf{dep}(\beta_2)$



$$
\begin{aligned}
&P_{\mathsf{DS}} = \\
&!\,\mathsf{new}\ k. \\
&\mathsf{out}^{\alpha_1}(c, \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))) \\
&\mid !\,\mathsf{in}^{\beta_1}(c, \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(ekb))). \\
&\quad \mathsf{in}^{\beta_2}(c, x) \\
&\mid \mathsf{out}^{\gamma_1}(c, \mathsf{pk}(ekb)). \\
&\quad \mathsf{out}^{\gamma_2}(c, \mathsf{vk}(ska))
\end{aligned}
$$

τ $\quad \mathsf{dep}_{\mathsf{init}}(\tau) = \emptyset$

Going back to Denning Sacco: $\mathsf{dep}(\beta_2)$



$$
\begin{aligned}
&P_{\mathsf{DS}} = \\
&!\,\mathsf{new}\ k. \\
&\mathsf{out}^{\alpha_1}(c, \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(ekb))) \\
&\mid !\,\mathsf{in}^{\beta_1}(c, \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(ekb))). \\
&\quad \mathsf{in}^{\beta_2}(c, x) \\
&\mid \mathsf{out}^{\gamma_1}(c, \mathsf{pk}(ekb)). \\
&\quad \mathsf{out}^{\gamma_2}(c, \mathsf{vk}(ska))
\end{aligned}
$$

τ $\quad \mathsf{dep}_{\mathsf{init}}(\tau) = \emptyset$

$$\mathsf{dep}(\beta_2) = \{\{\beta_2, \beta_1, \alpha_1\}\}$$

Our tool HowMany

The function dep returns a set of multisets of labels, and may also return infinite sets.

Example - Denning-Sacco protocol:

- simple scenario: $\mathsf{dep}(\beta_2) = \{\{\beta_2, \beta_1, \alpha_1\}\}$
- more involved scenario (involving a dishonest agent c):

$$\mathsf{dep}(\beta_2) = \left\{ \begin{array}{c} \{\beta_2,\, \beta_1,\, \alpha_1,\, \alpha_1'\}, \\ \{\beta_2,\, \beta_1,\, \alpha_1',\, \alpha_1',\, \gamma_1\} \end{array} \right\}$$

Our tool HowMany

The function dep returns a set of multisets of labels, and may also return infinite sets.

Example - Denning-Sacco protocol:

- simple scenario: $\mathsf{dep}(\beta_2) = \{\{\beta_2, \beta_1, \alpha_1\}\}$
- more involved scenario (involving a dishonest agent c):

$$\mathsf{dep}(\beta_2) = \left\{ \begin{array}{c} \{\beta_2,\, \beta_1,\, \alpha_1,\, \alpha'_1\}, \\ \{\beta_2,\, \beta_1,\, \alpha'_1,\, \alpha'_1,\, \gamma_1\} \end{array} \right\}$$

HowMany

Our tool implements a terminating algorithm that returns the same result than dep whenever it is finite, and $\perp$ otherwise.

Case studies

	Reachability (Wsec)						
	HowMany			SAT-Equiv		DeepSec	
	nb	size		mult.	unique	mult.	unique
Symmetric protocols							
Denning-Sacco	1	3 (8)		<1s		<1s	
Needham-Schroeder	16	20 (45)	28 (63)	12s	5s	32s	18m
Otway-Rees*	4	12 (20)	16 (28)	2s	1s	< 1s	1s
Wide-Mouth-Frog*	1	3 (6)		<1s		<1s	
Kao-Chow (variant)*	48	15 (27)	28 (47)	4m	1m	3s	2m
Yahalom-Paulson*	25	19 (35)	30 (56)	2m	44s	4h	TO
Yahalom-Lowe*	-	-	-	-	-	-	-
Asymmetric protocols							
Denning-Sacco	1	2 (4)		<1s		<1s	
Needham-Schroeder*	-	-	-	-	-	-	-
NS-Lowe*	2	7 (16)	8 (18)	<1s	<1s	< 1s	<1s

Trace inclusion

A protocol P is trace included in Q, written $P \sqsubseteq_t Q$, if for every $(\mathsf{tr}, \phi) \in \mathsf{trace}(P)$, there exists $(\mathsf{tr}', \phi') \in \mathsf{trace}(Q)$ such that $\mathsf{tr} =_{\mathcal{L}} \mathsf{tr}'$, and $\phi \sqsubseteq_s \phi'$.

$\longrightarrow$ trace equivalence: $P \approx_t Q \Leftrightarrow P \sqsubseteq_t Q$ and $Q \sqsubseteq_t P$

This is the notion we use to model e.g. unlinkability, anonymity, or strong secrecy.

Trace inclusion

A protocol P is trace included in Q, written $P \sqsubseteq_t Q$, if for every $(\mathrm{tr}, \phi) \in \mathrm{trace}(P)$, there exists $(\mathrm{tr}', \phi') \in \mathrm{trace}(Q)$ such that $\mathrm{tr} =_{\mathcal{L}} \mathrm{tr}'$, and $\phi \sqsubseteq_s \phi'$.

$\longrightarrow$ trace equivalence: $P \approx_t Q \Leftrightarrow P \sqsubseteq_t Q$ and $Q \sqsubseteq_t P$

This is the notion we use to model e.g. unlinkability, anonymity, or strong secrecy.

1. Some additional assumptions
2. Our algorithm to compute and generate the scenarios
3. Some case studies

Additional assumptions

$\longrightarrow$ We consider protocols with a simple structure.

Simple processes: each process in parallel has its own dedicated channel.

Protocols without else branches: our processes do not use the match construct.

Main result

Theorem

Let P be a simple protocol type-compliant w.r.t. some typing system (Δ, δ). Let Q be another simple protocol such that $P \not\sqsubseteq_t Q$. There exists a trace $(\mathsf{tr}, \phi) \in \mathsf{trace}(P)$ witnessing this non-inclusion such that $\mathsf{Label}(\mathsf{tr}) \subseteq A$ for some $A \in \mathsf{dep}(P)$.

We have that:

$$\mathsf{dep}(P) = \{\emptyset\} \cup S_{\mathsf{reach}}(P) \cup S_{\mathsf{test}}(P)$$

where:

- $S_{\mathsf{reach}}(P) = \bigcup_{\alpha \in \mathsf{Label}(P)} \mathsf{dep}(\alpha)$
- $S_{\mathsf{test}}(P) = \bigcup_{\tau \in St(\delta(P))} \mathsf{dep}(\tau) \otimes S_{\mathsf{out}}(\tau)$

$\longrightarrow$ this requires a precise characterization of static inclusion

Case studies

	Equivalence (Kpriv)						
	HowMany			**SAT-Equiv**		**DeepSec**	
	nb	size		mult.	unique	mult.	unique
Symmetric protocols							
Denning-Sacco	5	5 (12)	14 (31)	<1s	<1s	<1s	<1s
Needham-Schroeder	83	33 (72)	47 (107)	6m	1m	TO	TO
Otway-Rees⋆	22	15 (23)	27 (48)	8s	7s	1s	48m
Wide-Mouth-Frog⋆	4	5 (8)	12 (20)	<1s	<1s	<1s	<1s
Kao-Chow (variant)⋆	385	29 (48)	55 (91)	10h	2h	TO	TO
Yahalom-Paulson⋆	147	29 (50)	45 (85)	45m	8m	TO	TO
Yahalom-Lowe⋆	-	-	-	-	-	-	-
Asymmetric protocols							
Denning-Sacco	5	3 (4)	8 (11)	<1s	<1s	<1s	<1s
Needham-Schroeder⋆	-	-	-	-	-	-	-
NS-Lowe⋆	20	9 (19)	14 (31)	<1s	<1s	<1s	<1s

Conclusion

Our contribution

A practical bound allowing us to rely on some existing tools to perform the security analysis both for **reachability** and **equivalence properties**.

1. We prove the correctness of our bound.
2. We implement our algorithm in HowMany.
3. We perform various case studies.

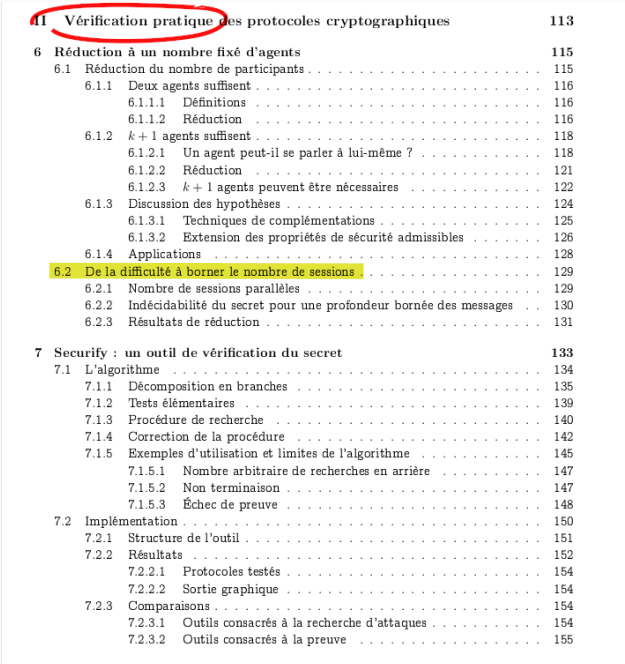
Future work

- authentication properties
- trade-off between unique and multiple scenarios

Conclusion

Going back to Vero's Phd thesis ...

- We now have a practical bound that is low enough so that we can use exisiting tool dedicated to bounded verification.
- This is also due to the progress done in the meantime by verification tools.



A mostly theoretical work which is nonetheless interesting !

Bravo Véro !

L'informatique, ça consiste à modifier les fichiers de configuration Windows . . .

mais pas que ;-)