Habilitation Defense

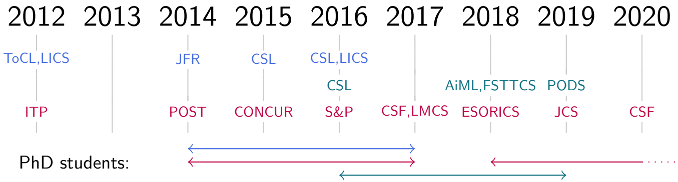
Contributions to the Verification of Cryptographic Protocols

David Baelde

Université Paris-Saclay

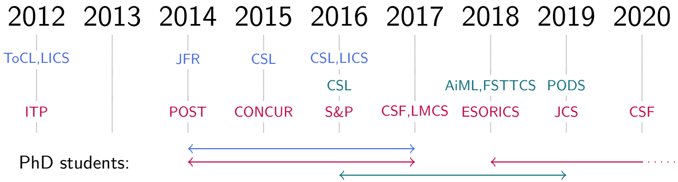
February 10, 2021

Research & supervision since 2012



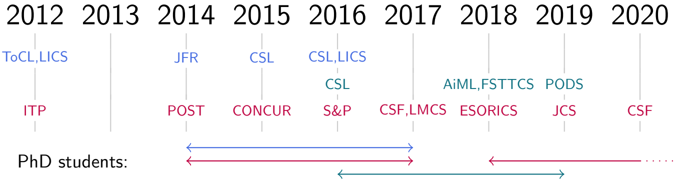
- Infinitary proof theory
- Proof systems for data logics
- Verification of cryptographic protocols

Research & supervision since 2012



- Infinitary proof theory — ANR RAPIDO
  🎓 Amina Doumane (Ackermann award) co-advised with Alexis Saurin
- Proof systems for data logics — ANR PRODAQ
  🎓 Anthony Lick co-advised with Sylvain Schmitz
- Verification of cryptographic protocols — ANR SEQUOIA & TECAP
  🎓 Lucca Hirschi (prix GdR sécurité) and
  Solène Moreau both co-advised with Stéphanie Delaune

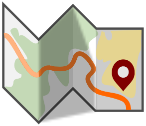
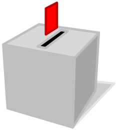
Research & supervision since 2012

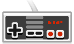


- Infinitary proof theory — ANR RAPIDO
  🎓 Amina Doumane (Ackermann award) co-advised with Alexis Saurin
  🔊 ESSLLI 2012 introductory course
- Proof systems for data logics — ANR PRODAQ
  🎓 Anthony Lick co-advised with Sylvain Schmitz
- Verification of cryptographic protocols — ANR SEQUOIA & TECAP
  🎓 Lucca Hirschi (prix GdR sécurité) and
  Solène Moreau both co-advised with Stéphanie Delaune
  🔊 MPRI M2 course in 2017–2019

Increasingly many activities are becoming digitalized.

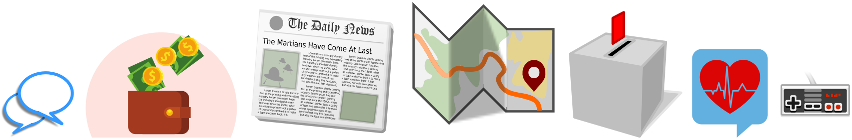




Security & Privacy

Increasingly many activities are becoming digitalized.

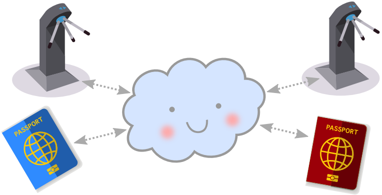


These systems must ensure important properties:

- security: secrecy, authenticity, no double-spending...
- privacy: anonymity, absence of tracking...

Frequent flaws at the hardware, software and specification levels.

Cryptographic protocols: a naive example



Each tag (T_i) owns a secret key k_i.

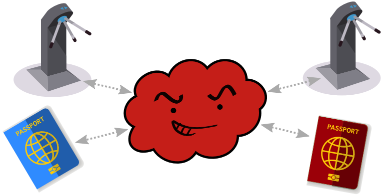
Reader (R) knows all legitimate keys.

$$\begin{array}{cccl} R & \rightarrow & T_i & : \; n_R \\ T_i & \rightarrow & R & : \; \mathsf{h}(n_R, k_i) \end{array}$$

Scenario under consideration:

- roles R, T_1, ..., T_n; arbitrary number of sessions for each role

Cryptographic protocols: a naive example



Each tag (T_i) owns a secret key k_i.

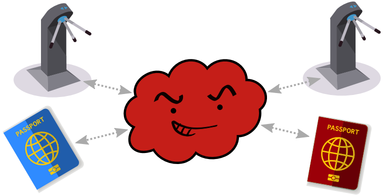
Reader (R) knows all legitimate keys.

$$\begin{array}{ccccl} R & \rightarrow & T_i & : & n_R \\ T_i & \rightarrow & R & : & \mathsf{h}(n_R, k_i) \end{array}$$

Scenario under consideration:

- roles R, T_1, ..., T_n; arbitrary number of sessions for each role
- attacker can intercept messages, inject new messages

Cryptographic protocols: a naive example



Each tag (T_i) owns a secret key k_i.

Reader (R) knows all legitimate keys.

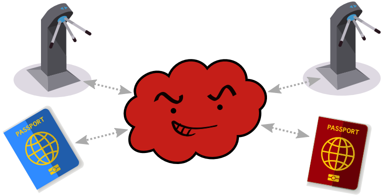
$$\begin{array}{lcccl} R & \rightarrow & T_i & : & n_R \\ T_i & \rightarrow & R & : & \mathsf{h}(n_R, k_i) \end{array}$$

Scenario under consideration:

- roles R, T_1, ..., T_n; arbitrary number of sessions for each role
- attacker can intercept messages, inject new messages

Readers correctly authenticate tags.

Cryptographic protocols: a naive example



Each tag (T_i) owns a secret key k_i.

Reader (R) knows all legitimate keys.

$$
\begin{array}{lllll}
R & \rightarrow & T_i & : & n_R \\
T_i & \rightarrow & R & : & \mathsf{h}(n_R, k_i)
\end{array}
$$

Scenario under consideration:

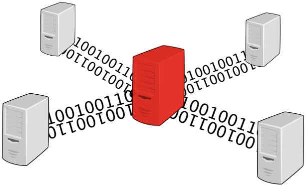
- roles R, T_1, ..., T_n; arbitrary number of sessions for each role
- attacker can intercept messages, inject new messages

Readers correctly authenticate tags.

Tags can be tracked: the protocol is not unlinkable.

- The attacker can obtain the pseudonym $\mathsf{h}(0, k_i)$ from a tag.

The computational model

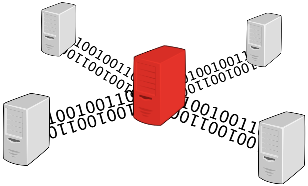


Messages = bitstrings

Secrets = random samplings

Participants = PPTIME Turing machines

The computational model



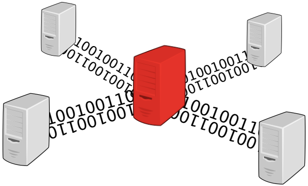
Messages = bitstrings

Secrets = random samplings

Participants = PPTIME Turing machines

In general, properties only hold with overwhelming probability, under some assumptions on cryptographic primitives.

The computational model



Messages = bitstrings

Secrets = random samplings

Participants = PPTIME Turing machines

In general, properties only hold with overwhelming probability, under some assumptions on cryptographic primitives.

Example (Unforgeability, EUF-CMA)

There is a negligible probability of success for the following game, for any attacker $\mathcal{A}$:

- Draw k uniformly at random.
- $\langle u, v \rangle := \mathcal{A}^{\mathcal{O}}$ where $\mathcal{O}$ is the oracle $x \mapsto \mathsf{h}(x, k)$.
- Succeed if $u = \mathsf{h}(v, k)$ and $\mathcal{O}$ has not been called on v.

Naive protocol in the computational model

Authentication

Attacker can interact with tags and readers,
wins if some reader accepts a message that has not been emitted by a tag.

Naive protocol in the computational model

Authentication

Attacker can interact with tags and readers,
wins if some reader accepts a message that has not been emitted by a tag.

- Attacker has to obtain some $\mathsf{h}(n_R, k_i)$ without querying T_i on n_R.
- Impossible if h is unforgeable.

Naive protocol in the computational model

Authentication

Attacker can interact with tags and readers,
wins if some reader accepts a message that has not been emitted by a tag.

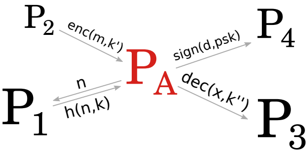
- Attacker has to obtain some $\mathsf{h}(n_R, k_i)$ without querying T_i on n_R.
- Impossible if h is unforgeable.

Privacy

Attacker interacts with either T_A, T_B or T_A, T_A
wins if he guesses in which situation he is.

- Success with probability almost 1 thanks to pseudonyms.

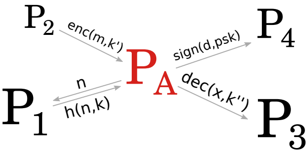
A symbolic model



Messages = terms modulo equations

Secrets = fresh constants (no probabilities)

A symbolic model



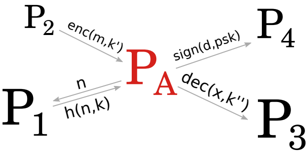
Messages = terms modulo equations

Secrets = fresh constants (no probabilities)

Example (Equational theory for symmetric encryption)

$$\mathsf{sdec}(\mathsf{senc}(x, y), y) =_{\mathsf{E}} x$$

A symbolic model



Messages = terms modulo equations

Secrets = fresh constants (no probabilities)

Participants defined using process algebra

Example (Equational theory for symmetric encryption)

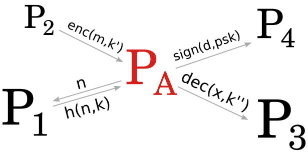
$$\mathsf{sdec}(\mathsf{senc}(x, y), y) =_{\mathsf{E}} x$$

Example (Processes)

$$T_i \stackrel{def}{=} \mathbf{in}(c, x).\mathbf{out}(c, \mathsf{h}(x, k_i)) \qquad S \stackrel{def}{=} T_i \mid T_j \mid P_A$$

$$P_A \stackrel{def}{=} \mathbf{out}(c, 0).\mathbf{in}(c, x).\mathbf{out}(c, 0).\mathbf{in}(c, y).\mathsf{if}\ x = y\ \mathsf{then}\ \mathbf{success}$$

A symbolic model



Messages = terms modulo equations

Secrets = fresh constants (no probabilities)

Participants defined using process algebra

Analyze system through LTS rather than explicit adversarial environment:

- Security properties modelled as reachability problems in the LTS.
- Privacy properties modelled as equivalence problems, e.g. may testing, trace equivalence.

Verification in the symbolic model

Trace properties

Undecidable in general, some restrictions decidable.
Mature automated tools borrowing, e.g., from rewriting and logic.

- Casper, Proverif, AVISPA, Scyther, Tamarin
  (Oxford, Inria Paris & Nancy, ETH Zürich, CISPA)
- Breaking/fixing/proving Google SSO, 3G/5G authentication, Neuchatel & Belenios e-voting, WPA2, Signal, TLS 1.3, etc.

Verification in the symbolic model

Trace properties

Undecidable in general, some restrictions decidable.
Mature automated tools borrowing, e.g., from rewriting and logic.

- Casper, Proverif, AVISPA, Scyther, Tamarin
  (Oxford, Inria Paris & Nancy, ETH Zürich, CISPA)
- Breaking/fixing/proving Google SSO, 3G/5G authentication, Neuchatel & Belenios e-voting, WPA2, Signal, TLS 1.3, etc.

Equivalence properties

- Bounded sessions: several tools and some decision procedures
  SPEC, Apte, Akiss, DeepSec, SAT-Equiv (ANU, LSV, Inria Nancy)
- Unbounded sessions: diff-equivalence in Proverif and Tamarin

Contributions to the verification of cryptographic protocols

Unlinkability [SP'16, JCS'19, CSF'20 distinguished paper]

- Modelling: importance of chosen equivalence and reader model
- Verification through sufficient conditions using Proverif and Tamarin

Contributions to the verification of cryptographic protocols

Unlinkability [SP'16, JCS'19, CSF'20 distinguished paper]

- Modelling: importance of chosen equivalence and reader model
- Verification through sufficient conditions using Proverif and Tamarin

Trace equivalence verification for bounded sessions

- Support for exclusive or in Akiss [CSF'17]
- Partial order reductions [POST'14, CONCUR'15, LMCS'17, ESORICS'18]

Contributions to the verification of cryptographic protocols

Unlinkability [SP'16, JCS'19, CSF'20 distinguished paper]

- Modelling: importance of chosen equivalence and reader model
- Verification through sufficient conditions using Proverif and Tamarin

Trace equivalence verification for bounded sessions

- Support for exclusive or in Akiss [CSF'17]
- Partial order reductions [POST'14, CONCUR'15, LMCS'17, ESORICS'18]

Proofs in the computational model [submitted]

- A meta-logic for proving trace and equivalence properties in the computational model

Contributions to the verification of cryptographic protocols

Unlinkability [SP'16, JCS'19, CSF'20 distinguished paper]

- Modelling: importance of chosen equivalence and reader model
- Verification through sufficient conditions using Proverif and Tamarin

Trace equivalence verification for bounded sessions

- Support for exclusive or in Akiss [CSF'17]
- Partial order reductions [POST'14, CONCUR'15, LMCS'17, ESORICS'18]

Proofs in the computational model [submitted]

- A meta-logic for proving trace and equivalence properties in the computational model

Focus 1/2

Partial Order Reductions for Protocol Equivalence Verification

joint work with Delaune & Hirschi

A classic problem

In 2012, bounded equivalence verification in SPEC, Apte & Akiss scales badly due to an exhaustive enumeration of traces.

A classic problem

In 2012, bounded equivalence verification in SPEC, Apte & Akiss scales badly due to an exhaustive enumeration of traces.

Example ($!^a_c\ \mathbf{in}(c,x).\mathbf{in}(c,y).\mathbf{out}(c,u)$)

$\mathbf{out}(a, c_1).$ $\mathbf{in}(c_1, R_1).$ $\mathbf{in}(c_1, w_2)$

$\mathbf{out}(a, c_2).$ $\mathbf{in}(c_2, R_2).\mathbf{in}(c_2, R_2').\mathbf{out}(c_2, w_2).$

A classic problem

In 2012, bounded equivalence verification in SPEC, Apte & Akiss scales badly due to an exhaustive enumeration of traces.

Example ($!^a_c$ $\mathbf{in}(c, x).\mathbf{in}(c, y).\mathbf{out}(c, u)$)

$\mathbf{out}(a, c_1).$ $\mathbf{in}(c_1, R_1).$ $\mathbf{in}(c_1, w_2)$

$\mathbf{out}(a, c_2).$ $\mathbf{in}(c_2, R_2).\mathbf{in}(c_2, R_2').\mathbf{out}(c_2, w_2).$

A classic problem

In 2012, bounded equivalence verification in SPEC, Apte & Akiss scales badly due to an exhaustive enumeration of traces.

Example ($!^a_c\ \mathbf{in}(c, x).\mathbf{in}(c, y).\mathbf{out}(c, u)$)

$\mathbf{out}(a, c_1). \qquad \mathbf{in}(c_1, R_1). \qquad \mathbf{in}(c_1, w_2)$

$\mathbf{out}(a, c_2). \qquad \mathbf{in}(c_2, R_2).\mathbf{in}(c_2, R_2').\mathbf{out}(c_2, w_2).$

A classic problem

In 2012, bounded equivalence verification in SPEC, Apte & Akiss scales badly due to an exhaustive enumeration of traces.

Example ($!^a_c\ \mathbf{in}(c,x).\mathbf{in}(c,y).\mathbf{out}(c,u)$)

$$\mathbf{out}(a,c_1).\qquad \mathbf{in}(c_1,R_1).\qquad \mathbf{in}(c_1,w_2)$$

$$\mathbf{out}(a,c_2).\qquad \mathbf{in}(c_2,R_2).\mathbf{in}(c_2,R_2').\mathbf{out}(c_2,w_2).$$

A classic problem

In 2012, bounded equivalence verification in SPEC, Apte & Akiss scales badly due to an exhaustive enumeration of traces.

Example ($!^a_c\ \mathbf{in}(c,x).\mathbf{in}(c,y).\mathbf{out}(c,u)$)

$$\mathbf{out}(a, c_1). \quad \mathbf{in}(c_1, R_1). \quad \mathbf{in}(c_1, w_2)$$

$$\mathbf{out}(a, c_2). \quad \mathbf{in}(c_2, R_2).\mathbf{in}(c_2, R'_2).\mathbf{out}(c_2, w_2).$$

- A classic problem in the verification of concurrent systems, which has been tackled using many partial order reduction (POR) techniques.
- Some POR used for verifying trace properties of crypto protocols.
- A new problem for equivalences of crypto protocols.

Compression

A first technique, directly inspired by focusing from proof theory.

Execution strategy in two alternating phases

- Output phase: eagerly execute outputs
- Focus on some process, initiating a session if possible
- Input phase: execute inputs of the process under focus

Parallel compositions processed only in output phase,
conditionals executed transparently in both phases.

Compression

A first technique, directly inspired by focusing from proof theory.

Execution strategy in two alternating phases

- Output phase: eagerly execute outputs
- Focus on some process, initiating a session if possible
- Input phase: execute inputs of the process under focus

Parallel compositions processed only in output phase, conditionals executed transparently in both phases.

Example $(!^a_c\ \mathbf{in}(c,x).\mathbf{in}(c,y).\mathbf{out}(c,u))$

$$\mathbf{out}(a,c_2).\mathbf{in}(c_2,R_2).\mathbf{in}(c_2,R_2').\mathbf{out}(c_2,w_2).\mathbf{out}(a,c_1).\mathbf{in}(c_1,R_1).\mathbf{in}(c_1,w_2)\ldots$$

Lifting POR to equivalence

Compression preserves enough reachable states e.g. to verify secrecy, but it is unsound to verify trace equivalence only along compressed traces.

Example

$\mathbf{out}(c, n).\mathbf{out}(c, \mathsf{h}(n, k))$ vs. $\mathbf{out}(c, n) \mid \mathbf{out}(c, \mathsf{h}(n, k))$

Lifting POR to equivalence

Compression preserves enough reachable states e.g. to verify secrecy, but it is unsound to verify trace equivalence only along compressed traces.

Definition (Action-determinism)

There is never two inputs (resp. outputs) in parallel on the same channel.

Example

$\mathbf{in}(c,x).\big(P_1 \mid \mathbf{out}(c,x).P_2\big)$ where $P_i = \mathbf{in}(c_i,x).Q$

Lifting POR to equivalence

Compression preserves enough reachable states e.g. to verify secrecy, but it is unsound to verify trace equivalence only along compressed traces.

Definition (Action-determinism)

There is never two inputs (resp. outputs) in parallel on the same channel.

Example

$\mathbf{in}(c, x).\big(P_1 \mid \mathbf{out}(c, x).P_2\big)$ where $P_i = \mathbf{in}(c_i, x).Q$

Solution: annotate processes & actions with info. about process structure.

Lemma

Two action-deterministic processes are trace equivalent iff they can execute the same annotated traces, resulting in statically equivalent configurations.

Lifting POR to equivalence

Compression preserves enough reachable states e.g. to verify secrecy, but it is unsound to verify trace equivalence only along compressed traces.

Definition (Action-determinism)

There is never two inputs (resp. outputs) in parallel on the same channel.

Lemma

Two action-deterministic processes are
trace equivalent iff they can execute the same annotated traces, resulting in statically equivalent configurations.

Corollary

Regular and compressed trace equiv. coincide for action-det. processes.

Full POR technique

Reduced strategy further constrains compressed executions, only allowing lexicographically minimal representatives of each partial order.

Data dependencies now taken into account.

Full POR technique

Reduced strategy further constrains compressed executions, only allowing lexicographically minimal representatives of each partial order.

Data dependencies now taken into account.

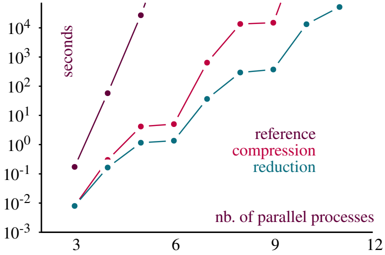
Integration with symbolic semantics

Verification algorithms rely on symbolic semantics and constraint solving.

- Compression immediately lifted to that setting.
- Reduction:
  new *dependency* constraints forbid (some) violations of strategy.

Full POR technique

Reduced strategy further constrains compressed executions, only allowing lexicographically minimal representatives of each partial order.



Data dependencies
now taken into account.

Integration with symbolic semantics

Verification algorithms rely on symbolic semantics and constraint solving.

- Compression immediately lifted to that setting.
- Reduction:
  new *dependency* constraints forbid (some) violations of strategy.

Conclusion

Success

Our techniques are used in all relevant tools:

- SPEC and Apte [B., Delaune & Hirschi, 2014, 2015 & 2017]
- Akiss [Kremer, 2016] and DeepSec [Cheval et al., 2018]

Further work on symmetry reductions [Cheval et al., 2019].

Conclusion

Success

Our techniques are used in all relevant tools:

- SPEC and Apte [B., Delaune & Hirschi, 2014, 2015 & 2017]
- Akiss [Kremer, 2016] and DeepSec [Cheval et al., 2018]

Further work on symmetry reductions [Cheval et al., 2019].

Discussion

- Soundness result applies to LTS without internal communication. Holds for a reasonable subclass of action-deterministic processes; status yet unknown for standard assumption.

Conclusion

Success

Our techniques are used in all relevant tools:

- SPEC and Apte [B., Delaune & Hirschi, 2014, 2015 & 2017]
- Akiss [Kremer, 2016] and DeepSec [Cheval et al., 2018]

Further work on symmetry reductions [Cheval et al., 2019].

Discussion

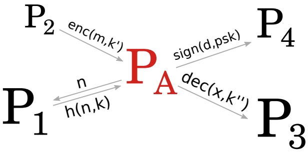
- Soundness result applies to LTS without internal communication.
  Holds for a reasonable subclass of action-deterministic processes; status yet unknown for standard assumption.
- More general technique [B., Delaune & Hirschi, 2018]
  builds on standard POR concepts but yields mixed results so far.

Focus 2/2

A Meta-Logic for Proving Protocols in the Computational Model

joint work with Delaune, Jacomme, Koutsos & Moreau

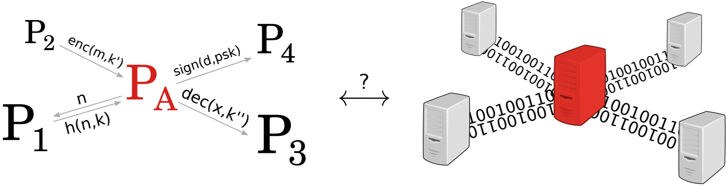
Limitations of the symbolic models



Symbolic models are elementary and enable high automation, but. . .

- Limited support for primitives with algebraic properties, e.g. xor.
- A proof holds because some capabilities have *not* been included: implicit assumptions.

Limitations of the symbolic models



Symbolic models are elementary and enable high automation, but. . .

- Limited support for primitives with algebraic properties, e.g. xor.
- A proof holds because some capabilities have *not* been included: implicit assumptions.
- Symbolic proofs are generally not computationally sound.

First-order logic over computational models

The computationally complete symbolic attacker [Bana & Comon, 2012 & 2014]

Terms interpreted as PPTIME machines

- Names = constants n, k interpreted as uniform samplings
- Primitives = function symbols interpreted as deterministic machines
- Attacker computations = *adversarial* function symbols $\mathbf{att}_i$ interpreted as PPTIME machines

First-order logic over computational models

The computationally complete symbolic attacker [Bana & Comon, 2012 & 2014]

Terms interpreted as PPTIME machines

- Names = constants n, k interpreted as uniform samplings
- Primitives = function symbols interpreted as deterministic machines
- Attacker computations =
  adversarial function symbols $\mathbf{att}_i$ interpreted as PPTIME machines

Example (Interaction with a reader in naive protocol)

$$t_{\text{input}} \stackrel{def}{=} \mathbf{att}_1(\mathsf{n}_R)$$

First-order logic over computational models

The computationally complete symbolic attacker [Bana & Comon, 2012 & 2014]

Terms interpreted as PPTIME machines

- Names = constants n, k interpreted as uniform samplings
- Primitives = function symbols interpreted as deterministic machines
- Attacker computations =
  adversarial function symbols $\mathbf{att}_i$ interpreted as PPTIME machines
- Some symbols with fixed interpretation: true, false, EQ, etc.

Example (Interaction with a reader in naive protocol)

$$t_{\text{input}} \stackrel{def}{=} \mathbf{att}_1(\mathsf{n}_R) \qquad \varphi_{\text{accept}} \stackrel{def}{=} \mathsf{EQ}\big(t_{\text{input}}, \mathsf{h}(\mathsf{n}_R, \mathsf{k}_i)\big)$$

First-order logic over computational models

The computationally complete symbolic attacker [Bana & Comon, 2012 & 2014]

Terms interpreted as PPTIME machines

- Names = constants n, k interpreted as uniform samplings
- Primitives = function symbols interpreted as deterministic machines
- Attacker computations = *adversarial* function symbols $\mathbf{att}_i$ interpreted as PPTIME machines
- Some symbols with fixed interpretation: true, false, EQ, etc.

Indistinguishability

Predicate $\sim$ interpreted as computational indistinguishability.

Example (Interaction with a reader in naive protocol)

$$t_{\mathrm{input}} \stackrel{def}{=} \mathbf{att}_1(\mathsf{n}_R) \qquad \varphi_{\mathrm{accept}} \stackrel{def}{=} \mathsf{EQ}(t_{\mathrm{input}}, \mathsf{h}(\mathsf{n}_R, \mathsf{k}_i))$$

$$\mathsf{n} \sim \mathsf{m} \qquad \mathsf{false} \sim^{?} \varphi_{\mathrm{accept}}$$

First-order logic over computational models

The computationally complete symbolic attacker [Bana & Comon, 2012 & 2014]

Cryptographic assumptions

Restrict interpretations of primitives to satisfy some crypto assumptions.

Example (Collision-resistance axiom)

$$\mathsf{true} \ \sim \ \mathsf{EQ}(\mathsf{h}(u, \mathsf{k}), \mathsf{h}(v, \mathsf{k})) \dot{\Rightarrow} \mathsf{EQ}(u, v)$$

where u and v are ground and k is only used as $\mathsf{h}(\_, \mathsf{k})$

First-order logic over computational models

The computationally complete symbolic attacker [Bana & Comon, 2012 & 2014]

Cryptographic assumptions

Restrict interpretations of primitives to satisfy some crypto assumptions.

Example (Collision-resistance axiom)

$$\mathtt{true} \sim \mathsf{EQ}(\mathsf{h}(u, \mathsf{k}), \mathsf{h}(v, \mathsf{k})) \dot{\Rightarrow} \mathsf{EQ}(u, v)$$

where u and v are ground and k is only used as $\mathsf{h}(\_, \mathsf{k})$

Full methodology

Given processes P and P' with bounded traces
and axiom schemes Ax including crypto assumptions,

First-order logic over computational models

The computationally complete symbolic attacker [Bana & Comon, 2012 & 2014]

Cryptographic assumptions

Restrict interpretations of primitives to satisfy some crypto assumptions.

Example (Collision-resistance axiom)

$$\mathsf{true} \;\sim\; \mathsf{EQ}(\mathsf{h}(u,\mathsf{k}),\mathsf{h}(v,\mathsf{k})) \dot{\Rightarrow} \mathsf{EQ}(u,v)$$

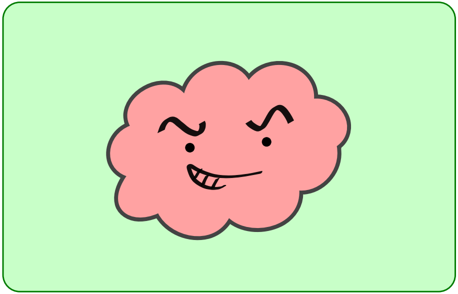
where u and v are ground and k is only used as $\mathsf{h}(\_,\mathsf{k})$

Full methodology

Given processes P and P' with bounded traces
and axiom schemes Ax including crypto assumptions,

- generate for each trace t_i a goal $\varphi_{t_i} := \vec{u}_{t_i} \sim \vec{u'}_{t_i}$;
- verify that $\mathrm{Ax} \models \varphi_{t_i}$ using some proof system for first-order logic.

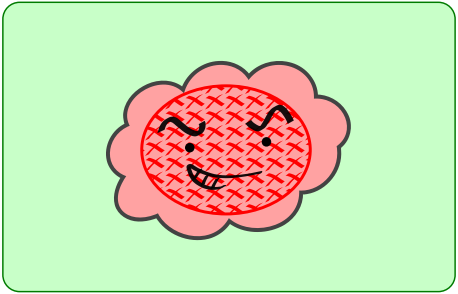
Contrasting the Dolev-Yao and Bana-Comon approaches



In summary:

- We are still dealing with symbolic expressions.

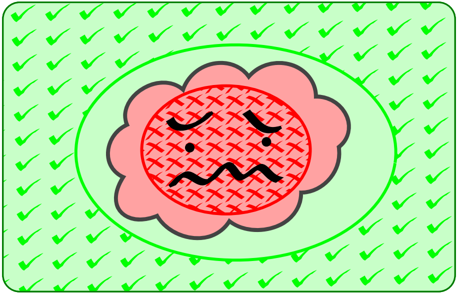
Contrasting the Dolev-Yao and Bana-Comon approaches



In summary:

- We are still dealing with symbolic expressions.

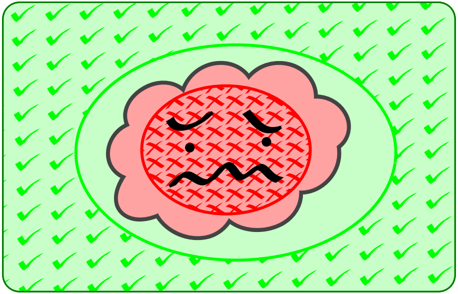
Contrasting the Dolev-Yao and Bana-Comon approaches



In summary:

- We are still dealing with symbolic expressions.
- Instead of specifying what the attacker can do, specify what is safe.

Contrasting the Dolev-Yao and Bana-Comon approaches



In summary:

- We are still dealing with symbolic expressions.
- Instead of specifying what the attacker can do, specify what is safe.

Example (Information-hiding axiom for xor)

$$\vec{u}, t \oplus \mathsf{n} \sim \vec{u}, \mathsf{m} \quad \text{when} \quad \mathsf{n}, \mathsf{m} \not\sqsubseteq \vec{u}, t \quad \text{and} \quad \mathsf{len}(t) = \mathsf{len}(\mathsf{n})$$

A meta-logic over the Bana-Comon logic

The Bana-Comon approach has some practical limitations:

- So far, automatically verifying $\mathsf{Ax} \models \varphi_t$ remains infeasible.
- The methodology assumes a fixed bound b on protocol traces.

base logic $\quad \varphi_{t_1},\ \varphi_{t_2},\ \ldots \quad + \quad \dfrac{\psi' \quad \psi''}{\psi} \quad = \quad \pi_{t_1},\ \pi_{t_2},\ \ldots$

A meta-logic over the Bana-Comon logic

The Bana-Comon approach has some practical limitations:

- So far, automatically verifying $\mathsf{Ax} \models \varphi_t$ remains infeasible.
- The methodology assumes a fixed bound b on protocol traces.

$\rightsquigarrow$ Develop a meta-logic

meta-logic $\qquad \Phi$

$$\Downarrow$$

base logic $\qquad \varphi_{t_1},\ \varphi_{t_2},\ \ldots \quad + \quad \dfrac{\psi' \quad \psi''}{\psi} \quad = \quad \pi_{t_1},\ \pi_{t_2},\ \ldots$

A meta-logic over the Bana-Comon logic

The Bana-Comon approach has some practical limitations:

- So far, automatically verifying $\mathsf{Ax} \models \varphi_t$ remains infeasible.
- The methodology assumes a fixed bound b on protocol traces.

$\rightsquigarrow$ Develop a meta-logic suitable for interactive proofs, independent of b.

$$
\begin{array}{lccccccc}
\text{meta-logic} & \Phi & + & \dfrac{\Psi' \quad \Psi''}{\Psi} & = & \Pi \\[2ex]
 & \Downarrow & & \Downarrow & & \Downarrow \\[2ex]
\text{base logic} & \varphi_{t_1},\ \varphi_{t_2},\ \ldots & + & \dfrac{\psi' \quad \psi''}{\psi} & = & \pi_{t_1},\ \pi_{t_2},\ \ldots
\end{array}
$$

Meta-logic terms and formulas

Example (Authentication for arbitrary traces of naive protocol)

$$\forall k.\ \ \mathsf{cond}@\mathsf{R}'(k) \Rightarrow\ \exists i,j.\ \ \mathsf{T}(i,j) < \mathsf{R}'(k) \wedge \mathsf{input}@\mathsf{T}(i,j) = \mathsf{output}@\mathsf{R}(k)$$

- $\mathsf{T}(i,j)$ = action of session j of T_i
- $\mathsf{R}(k)$ and $\mathsf{R}'(k)$ = actions of session k of R

Meta-logic terms and formulas

Example (Authentication for arbitrary traces of naive protocol)

$$\forall k.\ \ \mathsf{cond@R}'(k) \Rightarrow\ \exists i,j.\ \ \mathsf{T}(i,j) < \mathsf{R}'(k) \wedge \mathsf{input@T}(i,j) = \mathsf{output@R}(k)$$

- $\mathsf{T}(i,j)$ = action of session j of T_i
- $\mathsf{R}(k)$ and $\mathsf{R}'(k)$ = actions of session k of R

Syntax

Meta-formulas Φ feature indices, timestamps, macros, quantifications over timestamp and index variables.

Semantics

Given protocol $\mathcal{P}$ and trace model $\mathbb{T}$, interpret Φ as base logic *term* $(\Phi)^{\mathbb{T}}_{\mathcal{P}}$. Meta-formula Φ is valid wrt. $\mathcal{P}$ when $\mathcal{M} \models (\Phi)^{\mathbb{T}}_{\mathcal{P}} \sim \mathsf{true}$ for all $\mathbb{T}$ and $\mathcal{M}$.

Meta-logic sequents and proof systems

Trace properties

Sequents $\Gamma \vdash_{\mathcal{P}} \Phi$ where Γ is a multiset of meta-formulas, $\mathcal{P}$ a protocol.

- Inference rules of standard classical first-order logic.
- Reasoning about ordering on timestamps, e.g. induction.
- Liftings of Bana-Comon axioms, in particular crypto. assumptions.

Meta-logic sequents and proof systems

Trace properties

Sequents $\Gamma \vdash_{\mathcal{P}} \Phi$ where Γ is a multiset of meta-formulas, $\mathcal{P}$ a protocol.

- Inference rules of standard classical first-order logic.
- Reasoning about ordering on timestamps, e.g. induction.
- Liftings of Bana-Comon axioms, in particular crypto. assumptions.

Equivalence properties

Sequents $\ldots \vdash_{\mathcal{P},\mathcal{P}'} \vec{u} \sim \vec{v}$ for protocols $\mathcal{P}$ and $\mathcal{P}'$.

Valid when, for all $\mathbb{T}$, the base logic formula $(\vec{u})^{\mathbb{T}}_{\mathcal{P}} \sim (\vec{v})^{\mathbb{T}}_{\mathcal{P}'}$ is valid.

Meta-logic sequents and proof systems

Trace properties

Sequents $\Gamma \vdash_{\mathcal{P}} \Phi$ where Γ is a multiset of meta-formulas, $\mathcal{P}$ a protocol.

- Inference rules of standard classical first-order logic.
- Reasoning about ordering on timestamps, e.g. induction.
- Liftings of Bana-Comon axioms, in particular crypto. assumptions.

Equivalence properties

Sequents $\ldots \vdash_{\mathcal{P},\mathcal{P}'} \vec{u} \sim \vec{v}$ for protocols $\mathcal{P}$ and $\mathcal{P}'$.

Valid when, for all $\mathbb{T}$, the base logic formula $(\vec{u})^{\mathbb{T}}_{\mathcal{P}} \sim (\vec{v})^{\mathbb{T}}_{\mathcal{P}'}$ is valid.

Protocols $\mathcal{P}$ and $\mathcal{P}'$ are indistinguishable when $\vdash_{\mathcal{P},\mathcal{P}'} \mathsf{frame}@t \sim \mathsf{frame}@t$.

Meta-logic sequents and proof systems

Trace properties

Sequents $\Gamma \vdash_{\mathcal{P}} \Phi$ where Γ is a multiset of meta-formulas, $\mathcal{P}$ a protocol.

- Inference rules of standard classical first-order logic.
- Reasoning about ordering on timestamps, e.g. induction.
- Liftings of Bana-Comon axioms, in particular crypto. assumptions.

Equivalence properties

Sequents $\ldots \vdash_{\mathcal{P},\mathcal{P}'} \vec{u} \sim \vec{v}$ for protocols $\mathcal{P}$ and $\mathcal{P}'$.

Valid when, for all $\mathbb{T}$, the base logic formula $(\vec{u})^{\mathbb{T}}_{\mathcal{P}} \sim (\vec{v})^{\mathbb{T}}_{\mathcal{P}'}$ is valid.

Protocols $\mathcal{P}$ and $\mathcal{P}'$ are indistinguishable when $\vdash_{\mathcal{P},\mathcal{P}'} \mathsf{frame}@t \sim \mathsf{frame}@t$.

- Liftings of Bana-Comon rules + induction +
  ability to leverage trace properties.

The Squirrel prover 

emacs@khaima

File Edit Options Buffers Tools squirrel Proof-General Help

Goal Retract Undo Next Use Goto Qed Home Command Interrupt Restart Help

```
hash h

abstract ok : message
abstract ko : message

name key : index->message
name n : index->message

channel cT
channel cR

process tag(i:index,j:index) =
  in(cR,x); out(cT,h(x,key(i)))

process reader(k:index) =
  out(cR,n(k));
  in(cT,x);
  if exists (i:index), x = h(n(k),key(i)) then
    R': out(cR,ok)
  else
    R'': out(cR,ko)

system ((!_k R: reader(k)) | (!_i !_j T: tag(i,j))).

goal authentication_R1 :
  forall k:index, cond@R'(k) =>
    exists (i,j:index), T(i,j) < R'(k) && input@T(i,j
Proof.
  intros.
  expand cond@R'(k).
  euf M0.
  exists i,j.
Qed.
```

-:--- naive-hash.sp Bot L36 (squirrel script +2 Scripting)

```
[goal> Focused goal (1/1):
System: default/both
----------------------------------------
forall (k:index),
(cond@R'(k) =>
 exists (i,j:index), (T(i,j) < R'(k) && input@T(i,j) = output@R(k)))
```

A proof assistant for our meta-logic

- About 15k lines of OCaml code, Proof General integration.
- Protocol specification in π-calculus style.
- Trace and equivalence properties.
- Basic automated reasoning, tactics and proof-search combinators.

Conclusion

Success

Mechanized proofs for arbitrary traces using Bana-Comon approach:

- Authentication, strong secrecy, unlinkability for various protocols using hashes, signatures, encryptions, xor & Diffie-Hellman

Unlinkability of some RFID protocol with xor could not be proved using Tamarin [B., Delaune & Moreau, 2020].

Conclusion

Success

Mechanized proofs for arbitrary traces using Bana-Comon approach:

- Authentication, strong secrecy, unlinkability for various protocols using hashes, signatures, encryptions, xor & Diffie-Hellman

Unlinkability of some RFID protocol with xor could not be proved using Tamarin [B., Delaune & Moreau, 2020].

Challenges

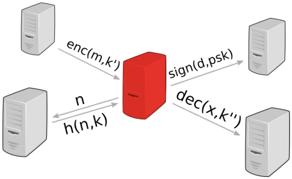
- Tackle more complex case studies, random oracle model, forward and post-compromise privacy, etc.
  Preliminary successes on protocols with mutable state.
- Improve automation: combining decision procedures à la SMT.
- Provide true unbounded guarantees:
  validity of meta-logic formulas only means security for each trace.

Perspectives

Perspectives

Computational model

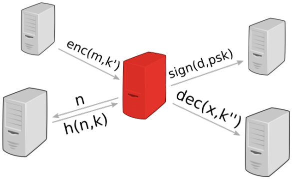
- Active and impactful field of research
- Cryptoverif & Easycrypt very successful; Bana-Comon in Coq closely related
- Finding interesting targets for Squirrel, e.g. protocols with mutable state
- Cooperation thanks to standard semantics



Perspectives

Computational model

- Active and impactful field of research
- Cryptoverif & Easycrypt very successful; Bana-Comon in Coq closely related
- Finding interesting targets for Squirrel, e.g. protocols with mutable state
- Cooperation thanks to standard semantics



Symbolic models

- Will remain king in automation and attack finding
- A future where partial orders are considered from the beginning: already SAT-Equiv, soon Akiss
- Need to better understand the range of equivalences between trace and diff-equivalences, and the corresponding threat models