Squirrel

Computer-Assisted Proofs of Protocols in the Computational Model

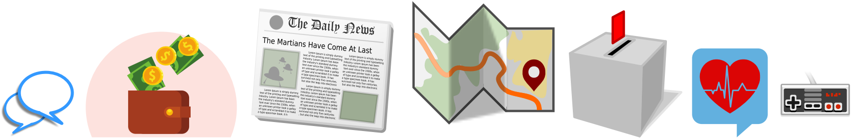
David Baelde

LMF, ENS Paris-Saclay & CNRS, Université Paris-Saclay

April 16, 2021

Security & Privacy

Increasingly many activities are becoming digitalized.

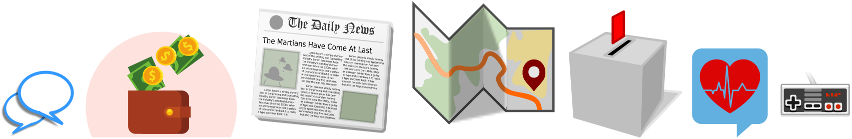


These systems must ensure important properties:

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

Security & Privacy

Increasingly many activities are becoming digitalized.



These systems must ensure important properties:

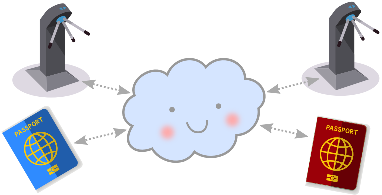
- security: secrecy, authenticity, no double-spending...
- privacy: anonymity, absence of tracking...

Frequent flaws at the hardware, software and specification levels.
Formal verification can help at all levels.

My focus: specifications of cryptographic protocols.

Modelling protocols using process algebra

Examples on authentication protocols



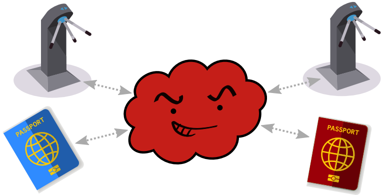
Processes:

- R for sessions of reader role;
- $T(k)$ for tag session with identity parameter k.

System $S := \,!R \mid !\, \textbf{new}\ k.\ !T(k)$.

Modelling protocols using process algebra

Examples on authentication protocols



Processes:

- R for sessions of reader role;
- $T(k)$ for tag session with identity parameter k.

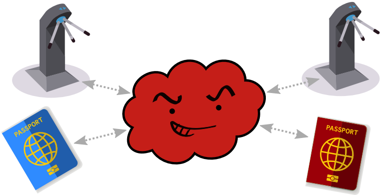
System $S \;:=\; !R \;|\; !\, \textbf{new}\; k.\; !T(k)$.

Reachability properties (trace properties)

- **Weak secrecy**: for any trace of S, attacker does not learn k.
- **Authentication**: for any trace of S, readers only issue *accept* events after the intended interaction with a tag.

Modelling protocols using process algebra

Examples on authentication protocols



Processes:

- R for sessions of reader role;
- $T(k)$ for tag session with identity parameter k.

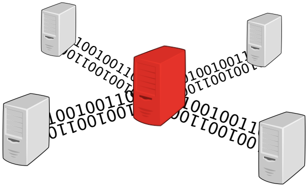
System $S := \ !R \mid !\ \textbf{new}\ k.\ !T(k)$.

Equivalence properties (hypertrace properties)

- **Anonymity**: $S \mid T(k_1) \approx S \mid T(k_2)$ — they are indistinguishable.
- **Strong unlinkability**: $S \approx\ !R \mid !\ \textbf{new}\ k.\ T(k)$.

Computational model

The mathematical setting for provable security in cryptography



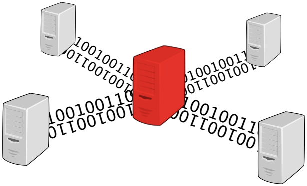
Messages = bitstrings

Secrets = random samplings

Computations = PPTIME Turing machines

Computational model

The mathematical setting for provable security in cryptography



Messages = bitstrings

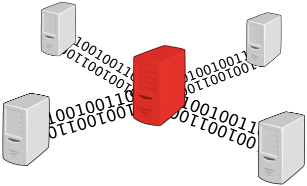
Secrets = random samplings

Computations = PPTIME Turing machines

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

Computational model

The mathematical setting for provable security in cryptography



Messages = bitstrings

Secrets = random samplings

Computations = PPTIME Turing machines

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

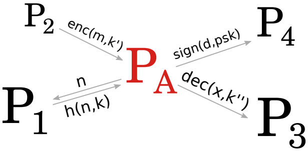
Example (Unforgeability, EUF-CMA)

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.

Symbolic model

An idealized setting, also known as Dolev-Yao model



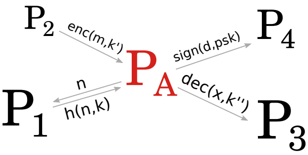
Messages = terms

Secrets = fresh constants (no probabilities)

Computations = equational theory

Symbolic model

An idealized setting, also known as Dolev-Yao model



Messages = terms

Secrets = fresh constants (no probabilities)

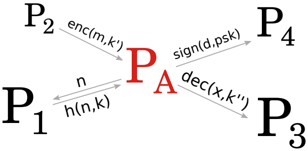
Computations = equational theory

Example (Equational theories)

- **Symmetric encryption**: $\mathsf{sdec}(\mathsf{senc}(x, y), y) =_{\mathsf{E}} x$.
- **Exclusive or**: assoc., commut., $x \oplus 0 =_{\mathsf{E}} x$ and $x \oplus x =_{\mathsf{E}} 0$.
- **Hash function**: no equation.

Symbolic model

An idealized setting, also known as Dolev-Yao model



Messages = terms

Secrets = fresh constants (no probabilities)

Computations = equational theory

Example (Equational theories)

- **Symmetric encryption**: $\mathsf{sdec}(\mathsf{senc}(x, y), y) =_{\mathsf{E}} x$.
- **Exclusive or**: assoc., commut., $x \oplus 0 =_{\mathsf{E}} x$ and $x \oplus x =_{\mathsf{E}} 0$.
- **Hash function**: no equation.
  Thus $\mathsf{h}(u, k) =_{\mathsf{E}} \mathsf{h}(v, k)$ implies $u =_{\mathsf{E}} v$,
  and $\mathsf{h}(u, k)$ indistinguishable from fresh name if k is private.

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

Unlinkability in the symbolic model

[B., Delaune & Hirschi, SP'16 and JCS'19] and [B., Delaune & Moreau, CSF'20]

Strong unlinkability for authentication protocols (e.g. RFID, e-passport) expressed as equivalence between multiple- and single-session systems.

- First time formal proofs (and some attack discoveries) for BAC, PACE, DAA, ABCDH, Feldhofer, OSK, LAK... using Proverif and Tamarin.

Unlinkability in the symbolic model

[B., Delaune & Hirschi, SP'16 and JCS'19] and [B., Delaune & Moreau, CSF'20]

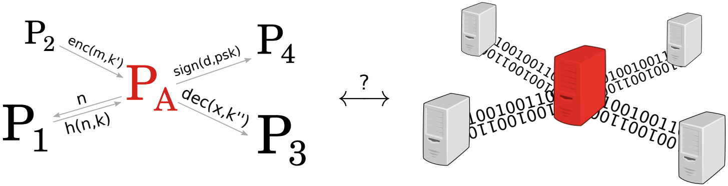
Strong unlinkability for authentication protocols (e.g. RFID, e-passport) expressed as equivalence between multiple- and single-session systems.

- First time formal proofs (and some attack discoveries) for BAC, PACE, DAA, ABCDH, Feldhofer, OSK, LAK... using Proverif and Tamarin.

Lessons

- Human guidance is required to reason about protocols with state.
- Limited support for Xor in Proverif and Tamarin: cannot handle simple RFID protocol with Xor (MW).
- Limited Diffie-Hellman support in Proverif: misses attack on PACE.

Computational soundness

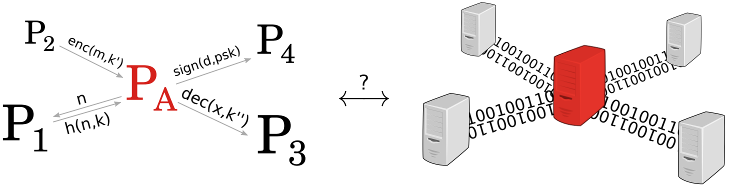


Some computational soundness theorems show that, in some cases, symbolic attackers account for all computational attacks.

They remain limited by strong assumptions.

- No sound symbolic abstraction of Xor.
- It seems hard to account for the nuanced properties of hash functions using symbolic models.

Computational soundness



Some computational soundness theorems show that, in some cases, symbolic attackers account for all computational attacks.

They remain limited by strong assumptions.

- No sound symbolic abstraction of Xor.
- It seems hard to account for the nuanced properties of hash functions using symbolic models.

Alternative: direct verification in the computational model.

- Cryptoverif, Easycrypt ... and Squirrel.

The CCSA approach:

Computationally Complete Symbolic Attacker

[Bana & Comon, CCS'14]

First-order logic over computational models

Terms interpreted as PPTIME machines

- Names = constants n, k interpreted as uniform samplings
- Primitives = function symbols interpreted as deterministic machines

First-order logic over computational models

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.

First-order logic over computational models

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 $\vec{u} \sim \vec{v}$ interpreted as computational indistinguishability.

Example

- We have $\mathsf{EQ}(\mathsf{n}, \mathsf{m}) \sim \mathsf{false}$

First-order logic over computational models

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 $\vec{u} \sim \vec{v}$ interpreted as computational indistinguishability.

Example

- We have $\mathsf{EQ}(\mathsf{n}, \mathsf{m}) \sim \mathsf{false}$ and even $\mathsf{EQ}(\mathsf{n}, \mathbf{att}_1(\mathsf{m})) \sim \mathsf{false}$.

First-order logic over computational models

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 $\vec{u} \sim \vec{v}$ interpreted as computational indistinguishability.

Example

- We have $\mathsf{EQ}(\mathsf{n}, \mathsf{m}) \sim \mathsf{false}$ and even $\mathsf{EQ}(\mathsf{n}, \mathbf{att}_1(\mathsf{m})) \sim \mathsf{false}$.
- We also have $(\vec{u} \sim \vec{v}) \Rightarrow (\vec{u}, \mathsf{n} \sim \vec{v}, \mathsf{m})$ when the names n, m do not occur in the ground terms $\vec{u}, \vec{v}$.

Example: the MW protocol

[Molnar & Wagner, CCS'04]

Assume a PRF $\mathsf{h}(\_,\_)$.

Each tag T_i is associated to an identity id_i and key key_i.

Reader R has access to database of all credentials.

$$\begin{array}{lcl} R \to T_i & : & \mathsf{n}_R \\ T_i \to R & : & \langle \mathsf{n}_T, \mathsf{id}_i \oplus \mathsf{h}(\langle 0, \mathsf{n}_R, \mathsf{n}_T \rangle, \mathsf{key}_i) \rangle \\ R \to T_i & : & \mathsf{id}_i \oplus \mathsf{h}(\langle 1, \mathsf{n}_R, \mathsf{n}_T \rangle, \mathsf{key}_i) \end{array}$$

Example: the MW protocol

[Molnar & Wagner, CCS'04]

Assume a PRF $\mathsf{h}(\_,\_)$.
Each tag T_i is associated to an identity id_i and key key_i.
Reader R has access to database of all credentials.

$$
\begin{array}{lcl}
R \to T_i & : & \mathsf{n}_R \\
T_i \to R & : & \langle \mathsf{n}_T, \mathsf{id}_i \oplus \mathsf{h}(\langle 0, \mathsf{n}_R, \mathsf{n}_T \rangle, \mathsf{key}_i) \rangle \\
R \to T_i & : & \mathsf{id}_i \oplus \mathsf{h}(\langle 1, \mathsf{n}_R, \mathsf{n}_T \rangle, \mathsf{key}_i)
\end{array}
$$

Example (Interaction with a reader)

$$
\begin{array}{rcl}
t_{\text{input}} & \stackrel{def}{=} & \mathbf{att}_1(\mathsf{n}_R) \\
b^i_{\text{accept}} & \stackrel{def}{=} & \mathsf{EQ}\big(\mathsf{snd}(t_{\text{input}}) \oplus \mathsf{id}_i, \mathsf{h}(\langle 0, \mathsf{n}_R, \mathsf{fst}(t_{\text{input}}) \rangle, \mathsf{key}_i)\big)
\end{array}
$$

Authentication: $\mathsf{false} \sim b^i_{\text{accept}}$?

Example: the MW protocol
[Molnar & Wagner, CCS'04]

Assume a PRF $\mathsf{h}(\_,\_)$.
Each tag T_i is associated to an identity id_i and key key_i.
Reader R has access to database of all credentials.

$$\begin{array}{lcl} R \rightarrow T_i & : & \mathsf{n}_R \\ T_i \rightarrow R & : & \langle \mathsf{n}_T, \mathsf{id}_i \oplus \mathsf{h}(\langle 0, \mathsf{n}_R, \mathsf{n}_T \rangle, \mathsf{key}_i)\rangle \\ R \rightarrow T_i & : & \mathsf{id}_i \oplus \mathsf{h}(\langle 1, \mathsf{n}_R, \mathsf{n}_T \rangle, \mathsf{key}_i) \end{array}$$

Example (Interaction with T_i and T_j)

$$\begin{array}{lcl} o_i & \stackrel{def}{=} & \langle \mathsf{n}_T, \mathsf{id}_i \oplus \mathsf{h}(\langle 0, \mathbf{att}_1(\ldots), \mathsf{n}_T \rangle, \mathsf{key}_i)\rangle \\ o'_j & \stackrel{def}{=} & \langle \mathsf{n}'_T, \mathsf{id}_j \oplus \mathsf{h}(\langle 0, \mathbf{att}_1(\ldots), \mathsf{n}'_T \rangle, \mathsf{key}_j)\rangle \end{array}$$

Anonymity: $o_i, o'_j \sim o_i, o'_i$?

Axiomatizing primitives

Example (EUF-CMA axiom)

Axiom scheme that holds in all models where h satisfies EUF-CMA:

$$\mathsf{true} \sim \left(\mathsf{EQ}(s, \mathsf{h}(t, \mathsf{k})) \dot{\Rightarrow} (\dot{\bigvee}_{u \in S} \mathsf{EQ}(u, t)) \right)$$

where $S = \{\ u \mid \mathsf{h}(u, \mathsf{k}) \text{ occurs in } s, t\}$ and k is only used in key position.

Axiomatizing primitives

Example (EUF-CMA axiom)

Axiom scheme that holds in all models where h satisfies EUF-CMA:

$$\mathsf{true} \sim \big(\mathsf{EQ}(s, \mathsf{h}(t, \mathsf{k})) \dot{\Rightarrow} (\dot{\bigvee}_{u \in S} \mathsf{EQ}(u, t)) \big)$$

where $S = \{\ u \mid \mathsf{h}(u, \mathsf{k})$ occurs in $s, t\}$ and k is only used in key position.

Example (PRF axiom)

$$\vec{v}, \mathsf{h}(t, \mathsf{k}) \sim \vec{v}, \mathsf{if}\ \dot{\bigvee}_{u \in S} \mathsf{EQ}(u, t)\ \mathsf{then}\ \mathsf{h}(t, \mathsf{k})\ \mathsf{else}\ \mathsf{n}$$

where n fresh, k used only as key and S is the set of hashes in $\vec{v}, t$.

Axiomatizing primitives

Example (EUF-CMA axiom)

Axiom scheme that holds in all models where h satisfies EUF-CMA:

$$\mathsf{true} \sim \left(\mathsf{EQ}(s, \mathsf{h}(t, \mathsf{k})) \mathrel{\dot{\Rightarrow}} (\dot{\bigvee}_{u \in S} \mathsf{EQ}(u, t)) \right)$$

where $S = \{\ u \mid \mathsf{h}(u, \mathsf{k})$ occurs in $s, t\}$ and k is only used in key position.

Example (PRF axiom)

$$\vec{v}, \mathsf{h}(t, \mathsf{k}) \sim \vec{v}, \mathsf{if}\ \dot{\bigvee}_{u \in S} \mathsf{EQ}(u, t)\ \mathsf{then}\ \mathsf{h}(t, \mathsf{k})\ \mathsf{else}\ \mathsf{n}$$

where n fresh, k used only as key and S is the set of hashes in $\vec{v}, t$.

Example (Information-hiding property of Xor)

$\vec{u}, t \oplus \mathsf{n} \sim \vec{u}, \mathsf{m}$ when n, m fresh and $\mathsf{len}(t) = \mathsf{len}(\mathsf{n})$

Axiomatizing primitives

Example (EUF-CMA axiom)

Axiom scheme that holds in all models where h satisfies EUF-CMA:

$$\mathsf{true} \sim \big(\, \mathsf{EQ}(s, \mathsf{h}(t, \mathsf{k})) \mathrel{\dot{\Rightarrow}} (\dot{\bigvee}_{u \in S}\, \mathsf{EQ}(u, t)) \,\big)$$

where $S = \{\ u \mid \mathsf{h}(u, \mathsf{k})$ occurs in $s, t\}$ and k is only used in key position.

Example (PRF axiom)

$$\vec{v}, \mathsf{h}(t, \mathsf{k}) \sim \vec{v}, \mathsf{if}\ \dot{\bigvee}_{u \in S}\, \mathsf{EQ}(u, t)\ \mathsf{then}\ \mathsf{h}(t, \mathsf{k})\ \mathsf{else}\ \mathsf{n}$$

where n fresh, k used only as key and S is the set of hashes in $\vec{v}, t$.

Example (Information-hiding property of Xor)

$$\vec{u}, t \oplus \mathsf{n} \sim \vec{u}, \mathsf{if}\ \mathsf{len}(t) = \mathsf{len}(\mathsf{n})\ \mathsf{then}\ \mathsf{m}\ \mathsf{else}\ (t \oplus \mathsf{n}) \qquad \text{when} \qquad \mathsf{n}, \mathsf{m}\ \text{fresh}$$

Verifying protocols using the CCSA logic

Assume some primitives and crypto assumptions.
Let Ax be the corresponding axiom schemes.

Computational indistinguishability

Consider protocols $\mathcal{P}$ and $\mathcal{Q}$ with bounded traces.

- Generate for each trace t_i the verification goal $\varphi_i := (\vec{u}_{t_i} \sim \vec{v}_{t_i})$ where $\vec{u}_{t_i}$ are the messages that $\mathcal{P}$ outputs for that trace, and similarly for $\vec{v}_{t_i}$ and $\mathcal{Q}$.
- Verify that $\mathsf{Ax} \models \varphi_i$ using any proof system for first-order logic.

Reachability properties

Consider a protocol with bounded traces and some reachability property.

- Generate for each trace t_i a goal $\varphi_{t_i} := (b_{t_i} \sim \mathsf{true})$.
- Verify that $\mathsf{Ax} \models \varphi_{t_i}$.

Limitations of the CCSA logic

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

Limitations of the CCSA logic

The CCSA 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

$$
\begin{array}{llcccc}
\text{meta-logic} & \Phi & & & & \\
 & \Downarrow & & & & \\
\text{base logic} & \varphi_{t_1},\ \varphi_{t_2},\ \ldots & + & \dfrac{\psi' \quad \psi''}{\psi} & = & \pi_{t_1},\ \pi_{t_2},\ \ldots
\end{array}
$$

Limitations of the CCSA logic

The CCSA 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.

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

The Squirrel Prover:

A Meta-Logic for Proving Protocols in the Computational Model

[B., Delaune, Jacomme, Koutsos & Moreau, SP'21]

Representing protocols and their executions

In our framework a protocol is given by:

- a partially ordered set of actions;
- for each action, a condition and an output term;
- some more things if mutable variables are considered.

We use indices to represent unbounded sets of actions and messages.

Representing protocols and their executions

In our framework a protocol is given by:

- a partially ordered set of actions;
- for each action, a condition and an output term;
- some more things if mutable variables are considered.

We use indices to represent unbounded sets of actions and messages.

Example (MW)

Actions:

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

Representing protocols and their executions

In our framework a protocol is given by:

- a partially ordered set of actions;
- for each action, a condition and an output term;
- some more things if mutable variables are considered.

We use indices to represent unbounded sets of actions and messages.

Example (MW)

Actions:

- $\mathsf{T}(i,j)$ and $\mathsf{T}'(i,j)$ for session j of T_i, with $\mathsf{T}(i,j) < \mathsf{T}'(i,j)$
- $\mathsf{R}(k)$ and $\mathsf{R}'(k)$ for session k of R, with $\mathsf{R}(k) < \mathsf{R}'(k)$

Representing protocols and their executions

In our framework a protocol is given by:

- a partially ordered set of actions;
- for each action, a condition and an output term;
- some more things if mutable variables are considered.

We use indices to represent unbounded sets of actions and messages.

Example (MW)

Actions:

- $\mathsf{T}(i,j)$ and $\mathsf{T}'(i,j)$ for session j of T_i, with $\mathsf{T}(i,j) < \mathsf{T}'(i,j)$
- $\mathsf{R}(k)$ and $\mathsf{R}'(k)$ for session k of R, with $\mathsf{R}(k) < \mathsf{R}'(k)$

Semantics:

- $\mathsf{output@T}(i,j) \stackrel{def}{=} \langle \mathsf{n}_T(i,j), \mathsf{h}(\langle 0, \mathsf{input@T}(i,j), \mathsf{n}_T(i,j)\rangle, \mathsf{key}(i))\rangle$
- $\mathsf{cond@R}'(k) \stackrel{def}{=} \exists i.\ \mathsf{snd}(t_{\mathsf{input}}) \oplus \mathsf{id}_i = \mathsf{h}(\langle 0, \mathsf{n}_R(k), \mathsf{fst}(t_{\mathsf{input}})\rangle, \mathsf{key}_i)$
- $\mathsf{frame@A} \stackrel{def}{=} \langle \mathsf{exec@A}, \mathsf{if\ exec@A\ then\ output@A}, \mathsf{frame@pred(A)}\rangle$

Meta-logic terms and formulas

Syntax

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

Example (Authentication for arbitrary traces of MW protocol)

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

Meta-logic terms and formulas

Syntax

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

Example (Authentication for arbitrary traces of MW protocol)

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

Semantics

Given protocol $\mathcal{P}$ and *trace model* $\mathbb{T}$, interpret Φ as base logic *term* $(\Phi)^{\mathbb{T}}_{\mathcal{P}}$.

- Indices and timestamps interpreted in finite domains.
- Interpretation of $<$ wrt. a fixed trace of executed actions.

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

Reachability properties

Sequents $\Gamma \vdash_{\mathcal{P}} \Phi$ where Γ is a multiset of meta-formulas, $\mathcal{P}$ a protocol.
Valid when, for all $\mathbb{T}$, the base logic formula $(\wedge\Gamma \Rightarrow \phi)^{\mathbb{T}}_{\mathcal{P}} \sim \mathsf{true}$ is valid.

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

Meta-logic sequents and proof systems

Reachability properties

Sequents $\Gamma \vdash_{\mathcal{P}} \Phi$ where Γ is a multiset of meta-formulas, $\mathcal{P}$ a protocol.
Valid when, for all $\mathbb{T}$, the base logic formula $(\wedge\Gamma \Rightarrow \phi)^{\mathbb{T}}_{\mathcal{P}} \sim \mathsf{true}$ is valid.

- Inference rules of standard classical first-order logic.
- Reasoning about ordering on timestamps, e.g. induction.
- Liftings of CCSA 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

Reachability properties

Sequents $\Gamma \vdash_{\mathcal{P}} \Phi$ where Γ is a multiset of meta-formulas, $\mathcal{P}$ a protocol.
Valid when, for all $\mathbb{T}$, the base logic formula $(\wedge\Gamma \Rightarrow \phi)^{\mathbb{T}}_{\mathcal{P}} \sim \mathsf{true}$ is valid.

- Inference rules of standard classical first-order logic.
- Reasoning about ordering on timestamps, e.g. induction.
- Liftings of CCSA 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

Reachability properties

Sequents $\Gamma \vdash_{\mathcal{P}} \Phi$ where Γ is a multiset of meta-formulas, $\mathcal{P}$ a protocol.
Valid when, for all $\mathbb{T}$, the base logic formula $(\wedge\Gamma \Rightarrow \phi)^{\mathbb{T}}_{\mathcal{P}} \sim \mathsf{true}$ is valid.

- Inference rules of standard classical first-order logic.
- Reasoning about ordering on timestamps, e.g. induction.
- Liftings of CCSA 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 $\vdash_{\mathcal{P},\mathcal{P}'} \mathsf{frame}@t \sim \mathsf{frame}@t$.

The two proof systems interact:
use reachability property to prove an equivalence, and conversely.

An authentication goal for MW

Let $\phi := \exists i, j.\ \mathsf{T}(i,j) < \mathsf{R}'(k) \quad \wedge\ \mathsf{input@T}(i,j) = \mathsf{output@R}(k)$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad \wedge\ \mathsf{input@R}'(k) = \mathsf{fst}(\mathsf{output@T}(i,j)).$

Prove $\mathsf{cond@R}'(k) \vdash \phi$ by EUF, which yields two cases:

- $\mathsf{T}(i,j) < \mathsf{R}'(k), \langle 0, \mathsf{n}_R(k), \mathsf{fst}(\mathsf{input@R}'(k))\rangle = \langle 0, \mathsf{input@T}(i,j), \mathsf{n}_T(i,j)\rangle \vdash \phi$ using obvious choices for existentials.
- $\mathsf{R}'(k') < \mathsf{R}'(k), \langle 0, \_, \_\rangle = \langle 1, \_, \_\rangle \vdash \phi$ absurd since $0 = 1$.

An authentication goal for MW

$$\text{Let } \phi := \exists i,j.\ \mathsf{T}(i,j) < \mathsf{R}'(k) \ \wedge\ \mathsf{input@T}(i,j) = \mathsf{output@R}(k)$$
$$\wedge\ \mathsf{input@R}'(k) = \mathsf{fst}(\mathsf{output@T}(i,j)).$$

Prove $\mathsf{cond@R}'(k) \vdash \phi$ by EUF, which yields two cases:

- $\mathsf{T}(i,j) < \mathsf{R}'(k), \langle 0, \mathsf{n}_R(k), \mathsf{fst}(\mathsf{input@R}'(k))\rangle = \langle 0, \mathsf{input@T}(i,j), \mathsf{n}_T(i,j)\rangle \vdash \phi$ using obvious choices for existentials.
- $\mathsf{R}'(k') < \mathsf{R}'(k), \langle 0, \_, \_\rangle = \langle 1, \_, \_\rangle \vdash \phi$ absurd since $0 = 1$.

Reasoning only relies on unforgeability of h, nothing to do with Xor!
It also seems close to what a cryptographer would say.

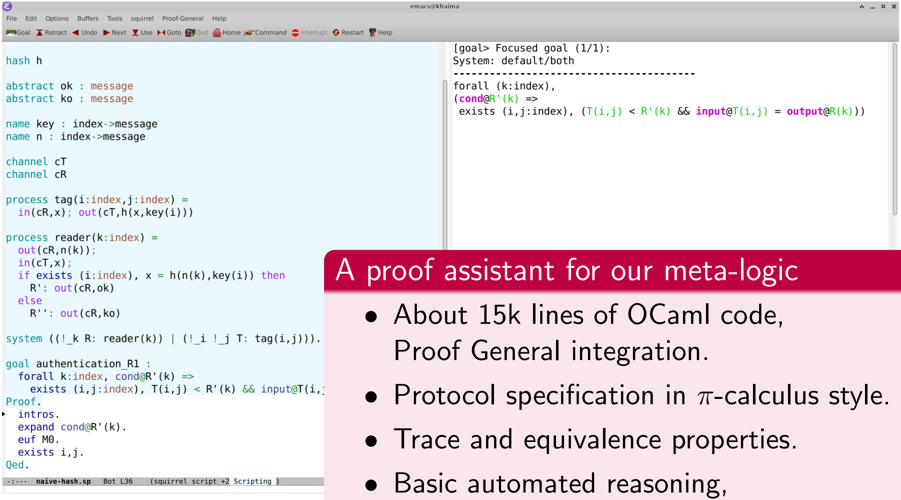
Unlinkability for MW

Let $E(T) := \mathsf{frame}@T \sim \mathsf{frame}@T$.
Prove $\vdash_{\mathcal{M},\mathcal{S}} \mathsf{frame}@\tau \sim \mathsf{frame}@\tau$ by induction:

- Obvious if $\tau = \mathsf{init}$.
- When $\tau = \mathsf{R}(k)$:
  $E(\mathsf{pred}(\mathsf{R}(k))) \vdash \mathsf{frame}@\mathsf{pred}(\mathsf{R}(k)), \mathsf{n}_R(k) \sim \mathsf{frame}@\mathsf{pred}(\mathsf{R}(k)), \mathsf{n}_R(k)$
  by freshness and $\mathsf{R}(k) < \mathsf{R}(k) \vee \mathsf{R}'(k) < \mathsf{R}(k) \vdash \bot$.
- When $\tau = \mathsf{T}(i,j)$:
  $$E(\mathsf{pred}(\mathsf{T}(i,j))) \vdash \begin{array}{l} \mathsf{frame}@\mathsf{pred}(\mathsf{T}(i,j)), \mathsf{n}_T(i,j), \mathsf{id}(i) \oplus \mathsf{h}(\ldots, \mathsf{key}(i)) \sim \\ \mathsf{frame}@\mathsf{pred}(\mathsf{T}(i,j)), \mathsf{n}_T(i,j), \mathsf{id'}(i,j) \oplus \mathsf{h}(\ldots, \mathsf{key'}(i,j)) \end{array}$$
  by PRF, Xor and freshness.
- When $\tau = \mathsf{R}'(k)$:
  $$E(\mathsf{pred}(\mathsf{R}'(k))) \vdash \begin{array}{l} \mathsf{frame}@\mathsf{pred}(\mathsf{R}'(k)), \mathsf{if}\ \mathsf{exec}@\mathsf{R}'(k)\ \mathsf{then}\ \mathsf{output}@\mathsf{R}'(k) \sim \\ \mathsf{frame}@\mathsf{pred}(\mathsf{R}'(k)), \mathsf{if}\ \mathsf{exec}@\mathsf{R}'(k)\ \mathsf{then}\ \mathsf{output}@\mathsf{R}'(k) \end{array}$$
  using authentication lemmas to replace $\mathsf{exec}@\mathsf{R}'(k)$ on both sides with formula that contains only public information,
  followed by PRF, Xor and freshness.

(I'm omitting some complexities wrt. the output.)

The Squirrel prover



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.

Summary of contributions

Squirrel

First-time mechanized proofs using CCSA approach:

- NSL, PA, Feldhofer, LAK, MW, SSH
- Hashes, signatures, encryptions, Xor & Diffie-Hellman
- Authentication, strong secrecy, unlinkability

Summary of contributions

Squirrel

First-time mechanized proofs using CCSA approach:

- NSL, PA, Feldhofer, LAK, MW, SSH
- Hashes, signatures, encryptions, Xor & Diffie-Hellman
- Authentication, strong secrecy, unlinkability

Comparison with other approaches

Computational verification: in between Cryptoverif (game-hopping) and Easycrypt (pRHL) in terms of automation; also expressivity trade-offs.

Summary of contributions

Squirrel

First-time mechanized proofs using CCSA approach:

- NSL, PA, Feldhofer, LAK, MW, SSH
- Hashes, signatures, encryptions, Xor & Diffie-Hellman
- Authentication, strong secrecy, unlinkability

Comparison with other approaches

Computational verification: in between Cryptoverif (game-hopping) and Easycrypt (pRHL) in terms of automation; also expressivity trade-offs.

Symbolic verification:

- Similarities with Tamarin: logic over traces, backward reasoning.
- Computational guarantees! also, no implicit assumptions.
- No automated attack finding.
- Less automated, but sometimes just as easy, even better for MW.

Future work on/with Squirrel

Foundations

- Truly unbounded guarantees:
  validity of meta-logic formulas only means security for each trace.
- Branching-time logic, e.g. for weak secrets or audits.
- Maintaining a coherent, usable implementation.
- Engineering trust: code generation, partial Coq certification.

Complex applications

- Protocols with state, oracles, compromises...
- Extensive models of deployed protocols e.g. Signal, TLS, Webauthn.
- Scalability issues: more automation (SMT), composition results.
- Bridging implementation and specification-level security:
  interoperable tools through standard computational semantics.