What is an attacker ?

Hubert Comon

D. Baelde, G. Bana, R. Chadha, S. Delaune, C. Jacomme, A. Koutsos, S. Moreau, G. Scerri, R. Stanley-Oakes

March 15, 2021

Roadmap

First half: introduction

Second half: list of results

Formal methods for security

- Prove formally security properties (break the circle attacks $\leftrightarrow$ security patches)
- Find attacks
- In this presentation: mostly applications to security protocols/security API. But the scope is larger.

What is specific to this area of research ?

Statement of the question

$$P \overset{?}{\models} \phi$$

P is the *model*: a formal concurrent process, a distributed program, an API, a circuit,...

ϕ is a *security property*: confidentiality, agreement, integrity, indistinguishability, ...

what is the satisfaction relation ?

The satisfaction relation

For "any" attacker $\mathcal{A}$,

$$P \parallel \mathcal{A} \textit{ never violates } \phi$$

$\mathcal{A}$ is the main difference with model checking

The satisfaction relation

For "any" attacker $\mathcal{A}$,

$$P \parallel \mathcal{A} \textit{ never violates } \phi$$

$\mathcal{A}$ is the main difference with model checking

Indistinguishability properties

$$P_1 \overset{?}{\sim} P_2$$

"An attacker $\mathcal{A}$ interacting with either P_1 or P_2 cannot guess with which of the two it interacted.

What is an attacker ?

- We cannot just consider any $\mathcal{A}$

What is an attacker ?

- We cannot just consider any $\mathcal{A}$
- We need to define formally the class of attackers that we consider:
  How $\mathcal{A}$ interacts with P, what are the computation capabilities of $\mathcal{A}$

What is an attacker ?

- We cannot just consider any $\mathcal{A}$
- We need to define formally the class of attackers that we consider:
  How $\mathcal{A}$ interacts with P, what are the computation capabilities of $\mathcal{A}$

Examples of attackers classes

- The "Dolev-Yao" model (for protocols), with many variations
- The interactive polynomial time Turing machines (with many variations)
- A quantum attacker
- Possible side channel information leaks

A security property may be satisfied for some attakers and not for others: this is a problem for the promotion of formal methods

Which model should we choose ?

- The Dolev Yao model is well suited for automation, but it is less precise (we may miss attacks)
- The computational model is more realistic, but it is difficult to complete formal proofs
- In even more realistic models, it is even more difficult to formalize proofs.

Is it irreconcilable ?

Our basic idea

Instead of specifying what are the attacker's capabilities, classes of attackers are defined by axioms stating what they cannot do.

Our basic idea

Instead of specifying what are the attacker's capabilities, classes of attackers are defined by axioms stating what they cannot do.

Examples

- $S \nvdash n$ if n is a random number not appearing in S
- $n \sim n'$ if n, n' are two random numbers
- $n \oplus m \sim n'$ if m is an arbitrary message, not containing n, whose length is the same as the length of n.

Our basic idea

Instead of specifying what are the attacker's capabilities, classes of attackers are defined by axioms stating what they cannot do.

Examples

- $S \nvdash n$ if n is a random number not appearing in S
- $n \sim n'$ if n, n' are two random numbers
- $n \oplus m \sim n'$ if m is an arbitrary message, not containing n, whose length is the same as the length of n.

Prove

$$\forall \mathcal{A}. \quad (\mathcal{A} \models \mathsf{Ax} \;\Rightarrow\; \mathcal{A} \| P \models \phi)$$

Why does it reconcile the various approaches ?

- We do not commit to a specific attacker model
- We stay within a classical framework of first-order logic

This is just a first-order unsatisfiability issue

The language of the logic

- function symbols for basic constructions: pairing, encryption, decryption, hash, database query, ... (it is an arbitrary choice)
  EQ, EQL,... : Boolean valued function symbols
- built-in: conditionals (if then else), true, false
- Attacker's symbols (in red)
- One predicate symbol: $\sim$.

A typical template

$$A \to B: \quad m(s)$$
$$B \to A: \quad \mathrm{comp}_B(m(s))$$

B receives $g_1(m(s))$

A typical template

$$A \to B: \quad m(s)$$
$$B \to A: \quad \mathrm{comp}_B(m(s))$$

B receives $g_1(m(s))$
B sends $\mathrm{comp}_B(g_1(m(s)))$

A typical template

$$A \to B: \quad m(s)$$
$$B \to A: \quad \mathrm{comp}_B(m(s))$$

B receives $g_1(m(s))$
B sends $\mathrm{comp}_B(g_1(m(s)))$
$\mathcal{A}$ computes $g_2(m(s), \mathrm{comp}_B(g_1(m(s))))$

A typical template

$$\begin{aligned} A \to B: &\quad m(s) \\ B \to A: &\quad \mathrm{comp}_B(m(s)) \end{aligned}$$

B receives $g_1(m(s))$
B sends $\mathrm{comp}_B(g_1(m(s)))$
$\mathcal{A}$ computes $g_2(m(s), \mathrm{comp}_B(g_1(m(s))))$

Attacker's computations are represented by free function symbols, that may be interpreted in any way, as long as the axioms are satisfied.

A typical template

$$\begin{aligned} A \to B: &\quad m(s) \\ B \to A: &\quad \mathrm{comp}_B(m(s)) \end{aligned}$$

B receives $g_1(m(s))$
B sends $\mathrm{comp}_B(g_1(m(s)))$
$\mathcal{A}$ computes $g_2(m(s), \mathrm{comp}_B(g_1(m(s))))$

Attacker's computations are represented by free function symbols, that may be interpreted in any way, as long as the axioms are satisfied.

Strong secrecy of s:

$$g_2(m(s), \mathrm{comp}_B(g_1(m(s)))) \sim g_2(m(s'), \mathrm{comp}_B(g_1(m(s'))))$$

A simple example

$$A \to B: \quad s \oplus k_1$$
$$B \to A: \quad s \oplus k_2$$

(both k_1, k_2 are shared secret keys)

A simple example

$$A \to B: \quad s \oplus k_1$$
$$B \to A: \quad s \oplus k_2$$

(both k_1, k_2 are shared secret keys)

does s remain confidential ?

$$s \oplus k_1,\ (g(s \oplus k_1) \oplus k_1) \oplus k_2 \quad \sim \quad n' \oplus k_1,\ (g(n' \oplus k_1) \oplus k_1) \oplus k_2$$

A simple example

$$A \to B: \quad s \oplus k_1$$
$$B \to A: \quad s \oplus k_2$$

(both k_1, k_2 are shared secret keys)

does s remain confidential ?

$$s \oplus k_1,\ (g(s \oplus k_1) \oplus k_1) \oplus k_2 \quad \sim \quad n' \oplus k_1,\ (g(n' \oplus k_1) \oplus k_1) \oplus k_2$$

Axiom 1: $v, u \oplus n_1 \sim v, n_2$
if n_1, n_2 are random numbers, u does not contain n_1, v does not contain n_1, n_2
Axiom 2: transitivity of $\sim$

A simple example

$$A \to B: \quad s \oplus k_1$$
$$B \to A: \quad s \oplus k_2$$

(both k_1, k_2 are shared secret keys)

does s remain confidential ?

$$s \oplus k_1,\ (g(s \oplus k_1) \oplus k_1) \oplus k_2 \quad \sim \quad n' \oplus k_1,\ (g(n' \oplus k_1) \oplus k_1) \oplus k_2$$

Axiom 1: $v, u \oplus n_1 \sim v, n_2$
if n_1, n_2 are random numbers, u does not contain n_1, v does not contain n_1, n_2
Axiom 2: transitivity of $\sim$
Assume k_1, k_2 do not occur in s:

$$\dfrac{\dfrac{}{s \oplus k_1, n_1 \sim n_2, n_1}\, A_1 \qquad \dfrac{}{s \oplus k_1, (g(s \oplus k_1) \oplus k_1) \oplus k_2 \sim s \oplus k_1, n_1}\, \mathbf{A1}}{s \oplus k_1, (g(s \oplus k_1) \oplus k_1) \oplus k_2 \sim n_2, n_1}\, \mathbf{A2}$$

What if the proof fails ?

If we use a complete first-order deduction system, then failure of the proof means that $\mathsf{Ax} \wedge \neg\phi$ is satisfiable: there is a model.
The model includes attacker's computations.

What if the proof fails ?

If we use a complete first-order deduction system, then failure of the proof means that $\mathsf{Ax} \wedge \neg\phi$ is satisfiable: there is a model. The model includes attacker's computations.

$$k_2 \oplus k_1, g(k_2 \oplus k_1) \oplus k_1 \oplus k_2 \quad \sim \quad n' \oplus k_1, g(n' \oplus k_1) \oplus k_1 \oplus k_2$$

is not provable.

What if the proof fails ?

If we use a complete first-order deduction system, then failure of the proof means that $\mathsf{Ax} \wedge \neg\phi$ is satisfiable: there is a model. The model includes attacker's computations.

$$k_2 \oplus k_1, g(k_2 \oplus k_1) \oplus k_1 \oplus k_2 \quad \sim \quad n' \oplus k_1, g(n' \oplus k_1) \oplus k_1 \oplus k_2$$

is not provable.

Counter-model: choose $g(x) = 0$ (and $\oplus$ is commutative, with neutral lement 0). Many possible attacker models.

What if the proof fails ?

If we use a complete first-order deduction system, then failure of the proof means that $\mathsf{Ax} \wedge \neg\phi$ is satisfiable: there is a model. The model includes attacker's computations.

$$k_2 \oplus k_1, g(k_2 \oplus k_1) \oplus k_1 \oplus k_2 \quad \sim \quad n' \oplus k_1, g(n' \oplus k_1) \oplus k_1 \oplus k_2$$

is not provable.

Counter-model: choose $g(x) = 0$ (and $\oplus$ is commutative, with neutral lement 0). Many possible attacker models.

Exercises:
$g(n \oplus k_1) \oplus k_1 \sim n$ is not provable. Any counter-model ?
Does $n, g(n \oplus k_1) \oplus n \sim k_1, g(n \oplus k_1) \oplus n$ hold under Ax ?

Folding

The last ingredient to reduce the security question to a first-order entailment.

Folding

The last ingredient to reduce the security question to a first-order entailment.

Given two protocols P_1, P_2, we can compute sequences of terms t_{P_1}, t_{P_2} such that P_1 and P_2 are indistinguishable (w.r.t. a class of attackers defined by Ax) iff $\mathsf{Ax} \models t_{P_1} \sim t_{P_2}$

Folding

The last ingredient to reduce the security question to a first-order entailment.

Given two protocols P_1, P_2, we can compute sequences of terms t_{P_1}, t_{P_2} such that P_1 and P_2 are indistinguishable (w.r.t. a class of attackers defined by Ax) iff $\mathsf{Ax} \models t_{P_1} \sim t_{P_2}$

Example:

$$\begin{array}{lll} A \to B : & \nu n, \nu r. & \mathsf{aenc}(\langle n, \mathsf{pk}_A \rangle, \mathsf{pk}_B, r) \\ B \to A : & \nu r'. & \mathsf{aenc}(n, \mathsf{pk}_A, r') \end{array}$$

Folding

The last ingredient to reduce the security question to a first-order entailment.

Given two protocols P_1, P_2, we can compute sequences of terms t_{P_1}, t_{P_2} such that P_1 and P_2 are indistinguishable (w.r.t. a class of attackers defined by Ax) iff $\mathsf{Ax} \models t_{P_1} \sim t_{P_2}$

Example:

$$\begin{array}{lll} A \to B : & \nu n, \nu r. & \mathsf{aenc}(\langle n, \mathsf{pk}_A \rangle, \mathsf{pk}_B, r) \\ B \to A : & \nu r'. & \mathsf{aenc}(n, \mathsf{pk}_A, r') \end{array}$$

$t_P = m_1, m_2$ with:
$m_1 = \mathsf{aenc}(\langle n, \mathsf{pk}_A \rangle, \mathsf{pk}_B, r)$,
$m_2 = \mathsf{aenc}(\pi_1(\mathsf{dec}(g(m_1))), \mathsf{pk}_A, r')$

Folding

The last ingredient to reduce the security question to a first-order entailment.

Given two protocols P_1, P_2, we can compute sequences of terms t_{P_1}, t_{P_2} such that P_1 and P_2 are indistinguishable (w.r.t. a class of attackers defined by Ax) iff $\mathsf{Ax} \models t_{P_1} \sim t_{P_2}$

Example:

$$\begin{array}{lll} A \to B : & \nu n, \nu r. & \mathsf{aenc}(\langle n, \mathsf{pk}_A \rangle, \mathsf{pk}_B, r) \\ B \to A : & \nu r'. & \mathsf{aenc}(n, \mathsf{pk}_A, r') \end{array}$$

$t_P = m_1, m_2$ with:
$m_1 = \mathsf{aenc}(\langle n, \mathsf{pk}_A \rangle, \mathsf{pk}_B, r)$,
$m_2 = \mathsf{aenc}(\pi_1(\mathsf{dec}(g(m_1))), \mathsf{pk}_A, r')$

Warning: bounded behavior of P_1, P_2. Interleavings use attacker's symbols: the attacker schedules the messages.

Computational proofs

Designing axioms for cryptographic libraries

IND-CCA1

$$w, \text{if } \mathrm{EQL}(u, u') \text{ then } \mathrm{aenc}(u, \mathrm{pk}_a, r) \text{ else } u'' \sim w, \text{if } \mathrm{EQL}(u, u') \text{ then } \mathrm{aenc}(u', \mathrm{pk}_a, r') \text{ else } u''$$

If sk_a only occurs in decryption position

Computational proofs

Designing axioms for cryptographic libraries

IND-CCA1

$$w, \text{if } \mathsf{EQL}(u, u') \text{ then } \mathsf{aenc}(u, \mathsf{pk}_a, r) \text{ else } u'' \\ \sim \quad w, \text{if } \mathsf{EQL}(u, u') \text{ then } \mathsf{aenc}(u', \mathsf{pk}_a, r') \text{ else } u''$$

If sk_a only occurs in decryption position

PRF

$$w, \text{if } c \text{ then } 0 \text{ else } H(t, k) \quad \sim \quad w, \text{if } c \text{ then } 0 \text{ else } n$$

where $c = \bigvee_{H(t_i,k) \sqsubseteq w,t} \mathsf{EQ}(t, t_i)$ and n is fresh

Computational proofs

Designing axioms for cryptographic libraries

IND-CCA1

$$w, \text{if } \mathrm{EQL}(u, u') \text{ then } \mathrm{aenc}(u, \mathrm{pk}_a, r) \text{ else } u'' \sim w, \text{if } \mathrm{EQL}(u, u') \text{ then } \mathrm{aenc}(u', \mathrm{pk}_a, r') \text{ else } u''$$

If sk_a only occurs in decryption position

PRF

$$w, \text{if } c \text{ then } 0 \text{ else } H(t, k) \sim w, \text{if } c \text{ then } 0 \text{ else } n$$

where $c = \bigvee_{H(t_i,k) \sqsubseteq w,t} \mathrm{EQ}(t, t_i)$ and n is fresh

Computational indistinguishability reduces to an entailment between two first-order formulas (no probabilities, no computation time).

Other approaches to computer assisted computational proofs

- CryptoVerif (game transformations),
- EasyCrypt (probabilistic relational Hoare logic),
- $\mathbf{F}^\star$ (proofs of programs).

Comparison

it is matter of

- taste
- applications
- time investment
- interactivity

Roadmap in 2014

At this stage (2014), it remained to:

- show that it is useful in practice (case studies)
- design axioms for many security primitives
- implement the logic and automate the proofs (as much as possible)
- drop the restriction(s)
- show the usefulness for other attacker models

Examples of Case studies

- Needham Schroeder protocols (new attacks, fixes), G. Bana
- Other classical protocols (new attacks, fixes), G. Scerri
- Key wrapping APIs, G. Scerri & R. Stanley-Oakes (CSF 2016)
- RFID protocols, H. Comon, A. Koutsos (CSF 2017)
- 5G AKA protocol, A. Koutsos (Euro S& P 2019)
- SSH with forwarding agent, C. Jacomme (CCS 2020)
- Several examples using SQUIRREL, Baelde et al (S& P 2021)

The prover SQUIRREL

Developed (under development) by D. Baelde, C. Jacomme, A. Koutsos (maybe others?)

- a meta-logic, allowing to combine reachability proofs and indistinguishability proofs
- The possibility to reason on unbounded traces
- An input as applied pi-calculus processes
- Avoids most of the time the expensive folding step
- Case studies include authentication and strong secrecy properties for SSH with forwarding agent

Decidability result

A result by A. Koutsos (2019)

For a given set of axioms, including the library independent computationaly sound axioms and the IND-CCA2 axiom, the logic is decidable.

Consequences

- If the proof fails, then there is an attack

Decidability result

A result by A. Koutsos (2019)

For a given set of axioms, including the library independent computationaly sound axioms and the IND-CCA2 axiom, the logic is decidable.

Consequences

- If the proof fails, then there is an attack
- There is a finite index equivalence relation on Probabilistic Polynomial Time Turing Machines: considering only one representative in each class is sufficient when looking for an attack.

Dropping the restriction

The main restiction is (was) the fixed number of sessions.

Two main recent advances

- In A. Koutsos work and in the Meta-Logic of SQUIRREL this restriction is (partly) droped: it is possible to construct proofs for an arbitrary number of sessions, provided it does not depend on the security parameter.
- As a sub-product of the composition result of Comon, Jacomme, Scerri 2020:
  Security of $!P$ against $\mathcal{A} \Leftarrow$ Security of P against $\mathcal{A}^{\mathcal{O}}$.

  Designing sound axioms w.r.t. $\mathcal{A}^{\mathcal{O}}$ reduces the unbounded sessions case to the bounded case.

Ongoing works

Other attacker models

Ongoing works

Other attacker models

Quantitative instrumentation

Ongoing works

Other attacker models

Quantitative instrumentation

More automated deduction...

Ongoing works

Other attacker models

Quantitative instrumentation

More automated deduction...

Beyond protocols

Ongoing works

Other attacker models

Quantitative instrumentation

More automated deduction...

Beyond protocols