A Biased Survey of Models and Methods for Verifying Cryptographic Protocols

Jean Goubault-Larrecq

Laboratoire Spécification et Vérification, CNRS, ENS Cachan, INRIA Futurs, SECSI

Classical and Quantum Information Security — Dec 15, 2005

Outline

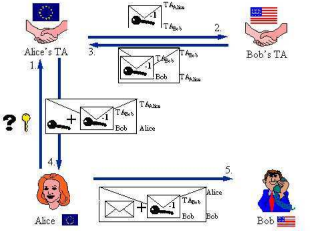
Cryptographic Protocols

Cryptography: 

Cryptographic Protocols

Cryptography:

Protocols:

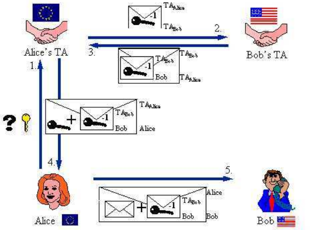


We may seek various properties:

(only a sample!

and pretty informal definitions for now, too)

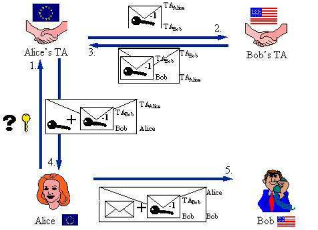
Cryptographic Protocols

Cryptography:  Protocols: 

We may seek various properties: (only a sample! and pretty informal definitions for now, too)

secrecy: M is secret if no adversay can emit M;

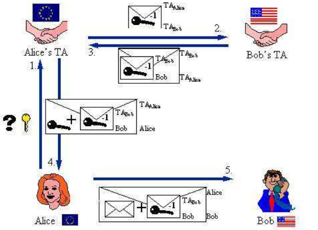
Cryptographic Protocols

Cryptography:  Protocols: 

We may seek various properties: (only a sample! and pretty informal definitions for now, too)

secrecy, **authenticity** (one form): the only process that can emit M is A;

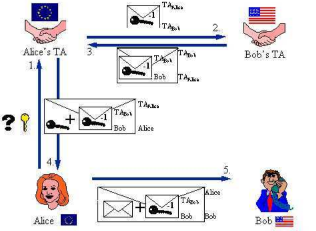
Cryptographic Protocols

Cryptography:  Protocols: 

We may seek various properties: (only a sample! and pretty informal definitions for now, too)

secrecy, **authenticity** , **anonymity**: you cannot link *A* with her message *M*, although *A* and *M* may be public;

Cryptographic Protocols

Cryptography:  Protocols: 

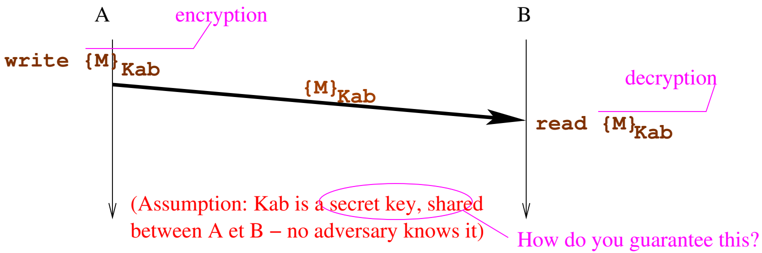
We may seek various properties: (only a sample! and pretty informal definitions for now, too)

secrecy, **authenticity** , **anonymity** , **fairness**: *A* cannot prove to *C* that she promised to sign with *B* before *A* and *B* indeed signed (together);
etc.

Outline

Fact: Cryptography is Not Enough

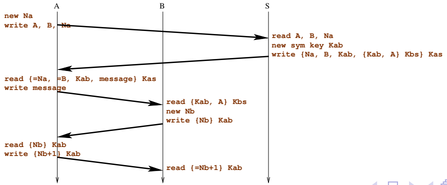
Even if you use perfect encryption algorithms (unbreakable), it is not easy to establish **secrecy** or **authentication**:



Fact: Cryptography is Not Enough

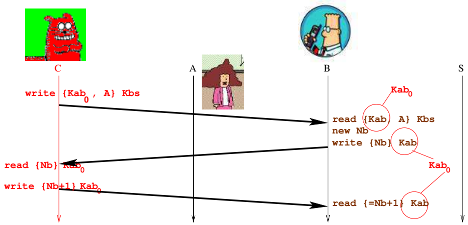
Even if you use perfect encryption algorithms (unbreakable), it is not easy to establish **secrecy** or **authentication.**
The **Needham-Schroeder** symmetric key protocol:

$$
\begin{array}{lll}
1. & A \longrightarrow S : & A, B, N_a \\
2. & S \longrightarrow A : & \{N_a, B, K_{ab}, \{K_{ab}, A\}_{K_{bs}}\}_{K_{as}} \\
3. & A \longrightarrow B : & \{K_{ab}, A\}_{K_{bs}} \\
4. & B \longrightarrow A : & \{N_b\}_{K_{ab}} \\
5. & A \longrightarrow B : & \{N_b + 1\}_{K_{ab}}
\end{array}
$$



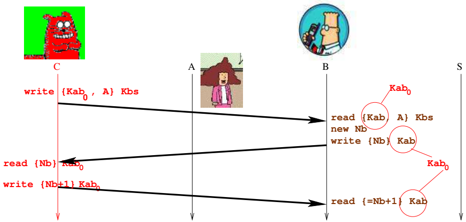
Fact: Cryptography is Not Enough

Even if you use perfect encryption algorithms (unbreakable), it is not easy to establish **secrecy** or **authentication.**
The **Needham-Schroeder** symmetric key protocol… and its attack:



Fact: Cryptography is Not Enough

Even if you use perfect encryption algorithms (unbreakable), it is not easy to establish **secrecy** or **authentication**



Purely logical attack!

A Word of Warning

This survey is *partial*. I could talk on models and methods for verifying protocols for *hours*.
Instead, this talk concentrates on:

- Logic-based models of security;

  no algebra, no probabilities, no Turing machines involved here.

A Word of Warning

This survey is *partial*. I could talk on models and methods for verifying protocols for *hours*.
Instead, this talk concentrates on:

- Logic-based models of security;

  no algebra, no probabilities, no Turing machines involved here.

- with a stress on automation of proof search;

A Word of Warning

This survey is *partial*. I could talk on models and methods for verifying protocols for *hours*.
Instead, this talk concentrates on:

- Logic-based models of security;

  no algebra, no probabilities, no Turing machines involved here.
- with a stress on automation of proof search;
- in the classical, non-quantum setting;

Outline

Fundamentals of Dolev-Yao, with a Modern View

- Handles reachability properties, e.g. secrecy:

$$\neg \mathbf{EF}\ \mathtt{knows}(M_{secret})$$

Fundamentals of Dolev-Yao, with a Modern View

- Handles reachability properties, e.g. secrecy:

$$\neg \mathbf{EF}\ \mathtt{knows}(M_{secret})$$

Handles safety properties, e.g., authentication,

$$\neg \mathbf{EF}\ (\exists M \cdot \mathtt{B\_received}(M) \wedge \neg \mathbf{G}^{-1} \mathtt{A\_sent}(M))$$

by the standard history variable trick.

Fundamentals of Dolev-Yao, with a Modern View

- Handles reachability properties, e.g. secrecy:

$$\neg \mathbf{EF}\ \mathtt{knows}(M_{secret})$$

- Assumes an active adversary, which can eavesdrop, forge messages, reroute communication, play with arbitrary many sessions (even in parallel) of several protocols.

Fundamentals of Dolev-Yao, with a Modern View

- Handles reachability properties, e.g. secrecy:

$$\neg \mathbf{EF}\ \mathtt{knows}(M_{secret})$$

- Assumes an active adversary, which can eavesdrop, forge messages, reroute communication, play with arbitrary many sessions (even in parallel) of several protocols.

  This actually *simplifies* the model: each message sent is sent to the adversary, each message received was built by the adversary.

Fundamentals of Dolev-Yao, with a Modern View

- Handles reachability properties, e.g. secrecy:

$$\neg \mathbf{EF}\ \mathtt{knows}(M_{secret})$$

- Assumes an active adversary, which can eavesdrop, forge messages, reroute communication, play with arbitrary many sessions (even in parallel) of several protocols.
- Assumes perfect cryptography primitives:

  "The only equations that hold between terms (with non-negligible probability) are $M = M$".

Fundamentals of Dolev-Yao, with a Modern View

- Handles reachability properties, e.g. secrecy:

$$\neg \mathbf{EF}\ \mathtt{knows}(M_{secret})$$

- Assumes an active adversary, which can eavesdrop, forge messages, reroute communication, play with arbitrary many sessions (even in parallel) of several protocols.
- Assumes perfect cryptography primitives:

  "The only equations that hold between terms (with non-negligible probability) are $M = M$".

A kind of idealization of [Dolev-Dwork-Naor, STOC'91]'s notion of non-malleable cryptography.

Fundamentals of Dolev-Yao, with a Modern View

- Handles reachability properties, e.g. secrecy:

$$\neg \mathbf{EF}\ \mathtt{knows}(M_{secret})$$

- Assumes an active adversary, which can eavesdrop, forge messages, reroute communication, play with arbitrary many sessions (even in parallel) of several protocols.
- Assumes perfect cryptography primitives:

  "The only equations that hold between terms (with non-negligible probability) are $M = M$".
- Models all capabilities of the adversary by a deduction system (see next slide).

Formalizing the Adversary's Knowledge

Given set E of messages eavesdropped by adversary, say that M is deducible from E, in notation $E \vdash M$, iff:

$$\frac{}{E, M \vdash M}\,(Ax)$$

$$\frac{E \vdash M \quad E \vdash K}{E \vdash \{M\}_K}\,(CryptI) \qquad \frac{E \vdash \{M\}_K \quad E \vdash K' \quad (K' \text{ inverse of } K)}{E \vdash M}\,(CryptE)$$

$$\frac{E \vdash M_1 \quad \dots \quad E \vdash M_n}{E \vdash (M_1, \dots, M_n)}\,(TupleI) \qquad \frac{E \vdash (M_1, \dots, M_n)}{E \vdash M_i}\,(TupleE_i), 1 \leq i \leq n$$

Formalizing the Adversary's Knowledge

Given set E of messages eavesdropped by adversary, say that M is deducible from E, in notation $E \vdash M$, iff:

$$\frac{}{E, M \vdash M}\,(Ax)$$

$$\frac{E \vdash M \quad E \vdash K}{E \vdash \{M\}_K}\,(CryptI) \qquad \frac{E \vdash \{M\}_K \quad E \vdash K' \quad (K' \text{ inverse of } K)}{E \vdash M}\,(CryptE)$$

$$\frac{E \vdash M_1 \quad \ldots \quad E \vdash M_n}{E \vdash (M_1, \ldots, M_n)}\,(TupleI) \qquad \frac{E \vdash (M_1, \ldots, M_n)}{E \vdash M_i}\,(TupleE_i), 1 \leq i \leq n$$

- Sending M means adding M to E;

Formalizing the Adversary's Knowledge

Given set E of messages eavesdropped by adversary, say that M is deducible from E, in notation $E \vdash M$, iff:

$$\frac{}{E, M \vdash M}\,(Ax)$$

$$\frac{E \vdash M \quad E \vdash K}{E \vdash \{M\}_K}\,(CryptI) \qquad \frac{\begin{array}{c}E \vdash \{M\}_K \quad E \vdash K' \\ (K' \text{ inverse of } K)\end{array}}{E \vdash M}\,(CryptE)$$

$$\frac{E \vdash M_1 \quad \ldots \quad E \vdash M_n}{E \vdash (M_1, \ldots, M_n)}\,(TupleI) \qquad \frac{E \vdash (M_1, \ldots, M_n)}{E \vdash M_i}\,(TupleE_i), 1 \leq i \leq n$$

- Sending M means adding M to E;
- Receiving a message M means that $E \vdash M$.

Deciding Protocol Insecurity

Thm [Durgin-Lincoln-Mitchell-Scedrov, FMSP'99] Reachability (=Insecurity) in Dolev-Yao models is undecidable.
Prf By reduction from 2-counter machines.

Deciding Protocol Insecurity

Thm [Rusinowitch-Turuani, CSFW'01] Reachability in Dolev-Yao models with a fixed number of sessions is NP-complete. (Note: only finitely many nonces.)

Deciding Protocol Insecurity

Thm [Rusinowitch-Turuani, CSFW'01] Reachability in Dolev-Yao models with a fixed number of sessions is NP-complete. (Note: only finitely many nonces.)

Prf Guess an interleaving of all sessions, create a fresh constant for each nonce. Then solve constraints

$$\begin{array}{rl} \vdash M_0 & \text{(first agent expects } M_0\text{, sends } N_1\text{)} \\ N_1 \vdash M_1 & \text{(second agent expects } M_1\text{, sends } N_2\text{)} \\ N_1, N_2 \vdash M_2 & \\ \vdots & \\ N_1, N_2, \ldots, N_k \vdash M_{secret} & \text{(adversary obtains secret)} \end{array}$$

Deciding Protocol Insecurity

Thm [Amadio-Charatonik, Concur'02] Under some (stringent, but necessary) conditions, with arbitrary many nonces, recursive agents but no forking, insecurity is decidable in EXPTIME.

Deciding Protocol Insecurity

Thm [Amadio-Charatonik, Concur'02] Under some (stringent, but necessary) conditions, with arbitrary many nonces, recursive agents but no forking, insecurity is decidable in EXPTIME.
Prf By encoding this into a (new) class of set constraints with renaming.

Deciding Protocol Insecurity

Thm [Amadio-Charatonik, Concur'02] Under some (stringent, but necessary) conditions, with arbitrary many nonces, recursive agents but no forking, insecurity is decidable in EXPTIME.
Prf By encoding this into a (new) class of set constraints with renaming.

Thm [Seidl-Verma, 05] With full parallel sessions, under the assumption of single blind copying and bounded number of nonces, insecurity is EXPTIME-complete.

Deciding Protocol Insecurity

Thm [Amadio-Charatonik, Concur'02] Under some (stringent, but necessary) conditions, with arbitrary many nonces, recursive agents but no forking, insecurity is decidable in EXPTIME.
Prf By encoding this into a (new) class of set constraints with renaming.

Thm [Seidl-Verma, 05] With full parallel sessions, under the assumption of single blind copying and bounded number of nonces, insecurity is EXPTIME-complete.
Prf By encoding into a decidable subclass of first-order logic (introduced in [Comon-Lundh-Cortier, RTA'03]).

Outline

Dolev-Yao in Full Parallel Multi-Session Mode

Pioneered by [Weidenbach, CADE'99] and [Blanchet, CSFW'01].

- Use a unary predicate `knows` instead of a binary relation $\vdash$:

  $\mathtt{knows}(M) \Leftrightarrow$ "the adversary is able to build M" (at whatever stage of the protocol).

Dolev-Yao in Full Parallel Multi-Session Mode

Pioneered by [Weidenbach, CADE'99] and [Blanchet, CSFW'01].

- Use a unary predicate `knows` instead of a binary relation $\vdash$:
  $\mathtt{knows}(M) \Leftrightarrow$ "the adversary is able to build M" (at whatever stage of the protocol).
- Abstract nonces N_a as Skolem functions $N_a(X, Y, Z)$ of the protocol-dependent variables X, Y, Z (identities, past nonces, etc.)

Dolev-Yao in Full Parallel Multi-Session Mode

Pioneered by [Weidenbach, CADE'99] and [Blanchet, CSFW'01].

- Use a unary predicate `knows` instead of a binary relation $\vdash$:
  $\texttt{knows}(M) \Leftrightarrow$"the adversary is able to build M" (at whatever stage of the protocol).
- Abstract nonces N_a as Skolem functions $N_a(X, Y, Z)$ of the protocol-dependent variables X, Y, Z (identities, past nonces, etc.)
- Encode everything into sets of Horn clauses.

Dolev-Yao in Full Parallel Multi-Session Mode

Pioneered by [Weidenbach, CADE'99] and [Blanchet, CSFW'01].

- Use a unary predicate `knows` instead of a binary relation $\vdash$:
  $\texttt{knows}(M) \Leftrightarrow$ "the adversary is able to build M" (at whatever stage of the protocol).
- Abstract nonces N_a as Skolem functions $N_a(X, Y, Z)$ of the protocol-dependent variables X, Y, Z (identities, past nonces, etc.)
- Encode everything into sets of Horn clauses.

Yes, Horn clauses are undecidable, so what?

Dolev-Yao in Full Parallel Multi-Session Mode

Pioneered by [Weidenbach, CADE'99] and [Blanchet, CSFW'01].

- Use a unary predicate `knows` instead of a binary relation $\vdash$:

  $\mathtt{knows}(M) \Leftrightarrow$ "the adversary is able to build M" (at whatever stage of the protocol).
- Abstract nonces N_a as Skolem functions $N_a(X, Y, Z)$ of the protocol-dependent variables X, Y, Z (identities, past nonces, etc.)
- Encode everything into sets of Horn clauses.

Yes, Horn clauses are undecidable, so what?

Let's examine our old friend, the symmetric key Needham-Schroeder protocol...

A Horn clause (pure Prolog) model
1. Abilities of the adversary.

$$
\begin{array}{rcll}
\mathtt{knows}(\{M\}_K) & \Leftarrow & \mathtt{knows}(M), \mathtt{knows}(K) & (C \text{ can encrypt}) \\
\mathtt{knows}(M) & \Leftarrow & \mathtt{knows}(\{M\}_{\mathtt{k}(\mathtt{sym},X)}), & \\
 & & \mathtt{knows}(\mathtt{k}(\mathtt{sym}, X)) & \ldots \text{and decrypt [symmetric keys])} \\
\mathtt{knows}([]) & & & (C \text{ can build} \\
\mathtt{knows}(M_1 :: M_2) & \Leftarrow & \mathtt{knows}(M_1), \mathtt{knows}(M_2) & \text{any list of known messages)} \\
\mathtt{knows}(M_1) & \Leftarrow & \mathtt{knows}(M_1 :: M_2) & (C \text{ can read heads)} \\
\mathtt{knows}(M_2) & \Leftarrow & \mathtt{knows}(M_1 :: M_2) & (C \text{ can read tails)} \\
\mathtt{knows}(\mathtt{suc}(M)) & \Leftarrow & \mathtt{knows}(M) & (C \text{ can add} \\
\mathtt{knows}(M) & \Leftarrow & \mathtt{knows}(\mathtt{suc}(M)) & \text{and subtract one)}
\end{array}
$$

2. Protocol clauses—current sessions (à la Blanchet)

$$1.\ A \longrightarrow S : A, B, N_a \quad \mathtt{knows}([\mathtt{a}, \mathtt{b}, \mathtt{na}([\mathtt{a}, \mathtt{b}])])$$

$1.\ A \rightarrow S:\ A, B, N_a$
$2.\ S \longrightarrow A : \{N_a, B, K_{ab}, \{K_{ab}, A\}_{K_{bs}} \}_{K_{as}}$

$$\mathtt{knows}\left(\begin{array}{l}\{[N_a, B, k_{ab}, \\ \quad \{[k_{ab}, A]\}_{\mathtt{k}(\mathtt{sym},[B,\mathtt{s}])} \\ \quad]\}_{\mathtt{k}(\mathtt{sym},[A,\mathtt{s}])})\end{array}\right) \Leftarrow \mathtt{knows}([A, B, N_a])$$

$$(k_{ab} \equiv \mathtt{k}(\mathtt{sym}, \mathtt{cur}(A, B, N_a)))$$

$2.\ S \longrightarrow A : \{N_a, B, K_{ab}, \{K_{ab}, A\}_{K_{bs}} \}_{K_{as}}$
$3.\ A \longrightarrow B : \{K_{ab}, A\}_{K_{bs}}$

$$\mathtt{knows}(M) \Leftarrow \mathtt{knows}(\{[\mathtt{na}([\mathtt{a}, \mathtt{b}]), \mathtt{b}, K_{ab}, M]\}_{\mathtt{k}(\mathtt{sym},[\mathtt{a},\mathtt{s}])})$$
$$\mathtt{a\_key}(K_{ab}) \Leftarrow \mathtt{knows}(\{[\mathtt{na}([\mathtt{a}, \mathtt{b}]), \mathtt{b}, K_{ab}, M]\}_{\mathtt{k}(\mathtt{sym},[\mathtt{a},\mathtt{s}])})$$

$3.\ A \longrightarrow B : \{K_{ab}, A\}_{K_{bs}}$
$4.\ B \longrightarrow A : \{N_b\}_{K_{ab}}$

$$\mathtt{knows}(\{\mathtt{nb}(K_{ab}, A, B)\}_{K_{ab}}) \Leftarrow \mathtt{knows}(\{[K_{ab}, A]\}_{\mathtt{k}(\mathtt{sym},[B,\mathtt{s}])})$$

$4.\ B \longrightarrow A : \{N_b\}_{K_{ab}}$
$5.\ A \longrightarrow B : \{N_b + 1\}_{K_{ab}}$

$$\mathtt{knows}(\{\mathtt{suc}(N_b)\}_{K_{ab}}) \Leftarrow \mathtt{knows}(\{N_b\}_{K_{ab}})$$

3. Protocol clauses—old sessions

$$
\begin{array}{l}
1.\ A \rightarrow S:\ A, B, N_a \\
2.\ S \longrightarrow A: \{N_a, B, K_{ab}, \\
\qquad \{K_{ab}, A\}_{K_{bs}} \\
\quad \}_{K_{as}}
\end{array}
\qquad
\mathtt{knows}\left(\begin{array}{l}\{[N_a, B, k_{ab}, \\ \quad \{[k_{ab}, A]\}_{\mathtt{k}(\mathtt{sym},[B,\mathtt{s}])} \\ \quad]\}_{\mathtt{k}(\mathtt{sym},[A,\mathtt{s}])})\end{array}\right) \Leftarrow \mathtt{knows}([A, B, N_a])
$$

$$(k_{ab} \equiv \mathtt{k}(\mathtt{sym}, \mathtt{prev}(A, B, N_a)))$$

4. Initial Knowledge of the Adversary

$$
\begin{array}{rcl}
\mathtt{agent}(\mathtt{a}) & & \mathtt{agent}(\mathtt{b}) \\
\mathtt{agent}(\mathtt{s}) & & \mathtt{agent}(\mathtt{i}) \\
\mathtt{knows}(X) & \Leftarrow & \mathtt{agent}(X) \\
\mathtt{knows}(\mathtt{k}(\mathtt{pub}, X)) & & \\
\mathtt{knows}(\mathtt{k}(\mathtt{prv}, \mathtt{i})) & & \\
\mathtt{knows}(\mathtt{k}(\mathtt{sym}, \mathtt{prev}(A, B, N_a))) & & \text{(old session keys} \\
& & \text{are compromised)}
\end{array}
$$

5. Security queries

$\bot \Leftarrow \mathtt{knows}(\mathtt{k}(\mathtt{sym}, \mathtt{cur}(\mathtt{a}, \mathtt{b}, N_a)))$

can C build K_{ab}

as created by S?

$\bot \Leftarrow \mathtt{knows}(K_{ab}), \mathtt{a\_key}(K_{ab})$

... as received by A?

$\bot \Leftarrow \mathtt{knows}(\{\mathtt{suc}(\mathtt{nb}(K_{ab}, A, B))\}_{K_{ab}}), \mathtt{knows}(K_{ab})$

... as received by B?

Important Remark: Security Proof = No Proof

A proof of $\perp$ (false) is an attack.

. . . i.e., a way of running clauses 1.–5.

which enables C to eventually know some sensitive data, here.

Important Remark: Security Proof = No Proof

A proof of $\perp$ (false) is an attack.

... i.e., a way of running clauses 1.–5.

which enables C to eventually know some sensitive data, here.

Selinger's Thesis: [Selinger, LACPV'01]

Security proof $\equiv$ no proof of $\perp$.

Important Remark: Security Proof = No Proof

A proof of $\perp$ (false) is an attack.

. . . i.e., a way of running clauses 1.–5.

which enables C to eventually know some sensitive data, here.

Selinger's Thesis: [Selinger, LACPV'01]

Security proof $\equiv$ no proof of $\perp$.

Constructively, the non-existence of a proof will be witnessed by a model.

by completeness of first-order logic [Gödel, 1930].

Deciding Sets of Horn Clauses

- Blanchet calls a dedicated prover, either SPASS [Weidenbach, CADE'96], or his own, built into his tool ProVerif.

Deciding Sets of Horn Clauses

- Blanchet calls a dedicated prover, either SPASS [Weidenbach, CADE'96], or his own, built into his tool ProVerif.
  This may fail to terminate… except that [Blanchet-Podelski, FoSSaCs'03] tagging enforces termination of Blanchet's selection strategy.

Deciding Sets of Horn Clauses

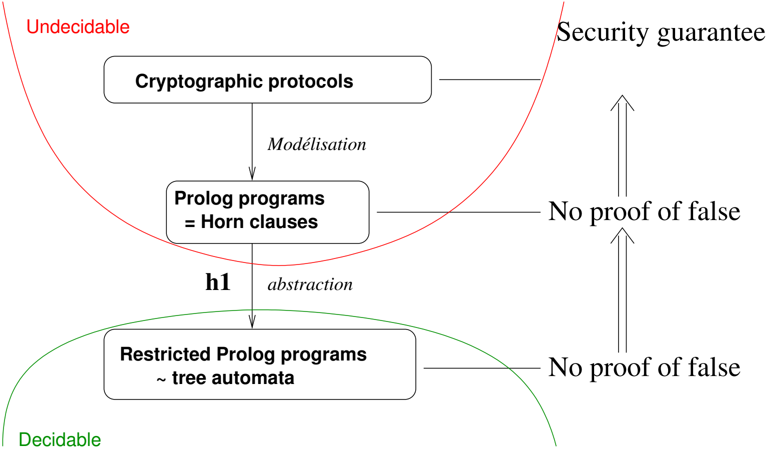
- Blanchet calls a dedicated prover, either SPASS [Weidenbach, CADE'96], or his own, built into his tool ProVerif.
- Or use upper approximations inside a decidable subclass of first-order logic:
  - Flat clauses [JGL, SECI'02] (DEXPTIME-complete);
  - $\mathcal{H}_1$ [Nielson-Nielson-Seidl, SAS'02; JGL, IPL'05] (DEXPTIME-complete, define regular tree languages).

Deciding Sets of Horn Clauses

- Blanchet calls a dedicated prover, either SPASS [Weidenbach, CADE'96], or his own, built into his tool ProVerif.
- Or use upper approximations inside a decidable subclass of first-order logic:
  - Flat clauses [JGL, SECI'02] (DEXPTIME-complete);
  - $\mathcal{H}_1$ [Nielson-Nielson-Seidl, SAS'02; JGL, IPL'05] (DEXPTIME-complete, define regular tree languages).

  Note: the latter allows us to generate formal proofs of security in Coq automatically [JGL, JFLA'04], but this is another story.

How Verification Works, Using Upper Approximations



Outline

An Interesting Story: The Needham-Schroeder Public Key Protocol. . .

B's public key;
B decrypts using his private key

A

B

```
new Na
write {Na, A} Kb
```

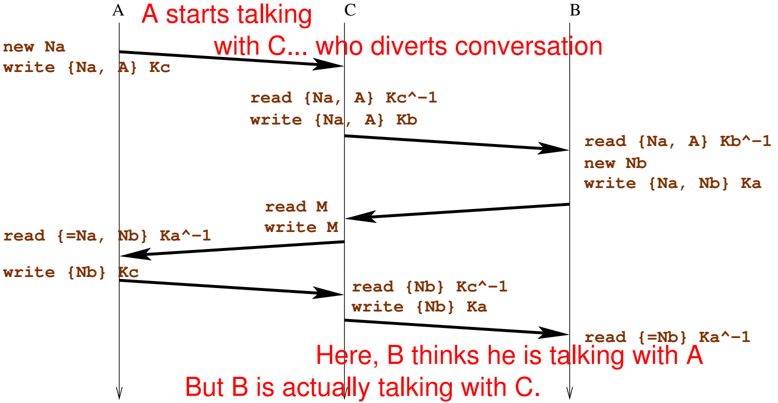
```
read {Na, A} Kb^-1
new Nb
write {Na, Nb} Ka
```

A's public key;
A decrypts with her private key

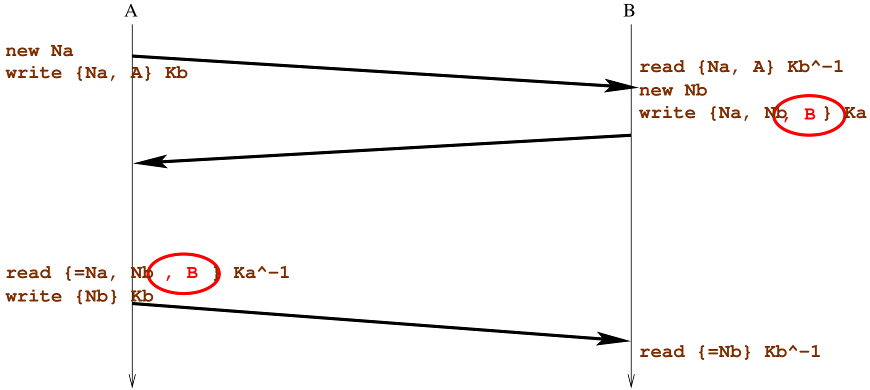
```
read {=Na, Nb} Ka^-1
write {Nb} Kb
```

```
read {=Nb} Kb^-1
```

. . . Is Vulnerable to a Man-In-The-Middle Attack [Lowe, IPL'95]. . .



. . . But Can Be Repaired Easily [Lowe, IPL'95]. . .



Security of Needham-Schroeder-Lowe

- You can then prove that Needham-Schroeder-Lowe is secure in the Dolev-Yao model.

Security of Needham-Schroeder-Lowe

- You can then prove that Needham-Schroeder-Lowe is secure in the Dolev-Yao model.
- Now, implement this using El Gamal encryption:
  - Let g be a generator of some cyclic group (say $\mathbb{Z}/p\mathbb{Z}^*$);
  - A generates x_A at random (nonce), publishes g^{x_A};
  - B generates r at random, publishes r;
  - Let $\texttt{Ka}^{-1} = x_A$ (A's private key), $\texttt{Ka} = g^{r x_A}$ (A's public key);

Security of Needham-Schroeder-Lowe

- You can then prove that Needham-Schroeder-Lowe is secure in the Dolev-Yao model.
- Now, implement this using El Gamal encryption:
  To encrypt M: $\{M\}_{\mathtt{Ka}} = M \oplus \mathtt{Ka}$ ($\approx$ one-time pad).
  About the most secure (classical) encryption scheme you can think of.

Security of Needham-Schroeder-Lowe

- You can then prove that Needham-Schroeder-Lowe is secure in the Dolev-Yao model.
- Now, implement this using El Gamal encryption:
  To encrypt M: $\{M\}_{\mathtt{Ka}} = M \oplus \mathtt{Ka}$ ($\approx$ one-time pad).
  About the most secure (classical) encryption scheme you can think of.
- But you can replay Lowe's attack, then [Joux, priv. comm., sep. 2002; Warinschi, CSFW'03]!

$$\begin{array}{rcl}
B & \text{sends} & \{\mathtt{Na}, \mathtt{Nb}, \mathtt{B}\}_{\mathtt{Ka}} \\
 & = & (\mathtt{Na}, \mathtt{Nb}, \mathtt{B}) \oplus \mathtt{Ka} \\
\text{Adversary} & \text{xors with} & (0, 0, \mathtt{B} \oplus \mathtt{C}) \\
A & \text{expects} & (\mathtt{Na}, \mathtt{Nb}, \mathtt{C}) \oplus \mathtt{Ka} \\
 & = & \{\mathtt{Na}, \mathtt{Nb}, \mathtt{C}\}_{\mathtt{Ka}}
\end{array}$$

Security of Needham-Schroeder-Lowe

- You can then prove that Needham-Schroeder-Lowe is secure in the Dolev-Yao model.
- Now, implement this using El Gamal encryption:
  To encrypt M: $\{M\}_{\mathtt{Ka}} = M \oplus \mathtt{Ka}$ ($\approx$ one-time pad).
  About the most secure (classical) encryption scheme you can think of.
- But you can replay Lowe's attack .
- El Gamal encryption is malleable. Additional equations:

$$\begin{aligned} \{M\}_K &= M \oplus K & \qquad (M_1 \oplus M_2) \oplus M_3 &= M_1 \oplus (M_2 \oplus M_3) \\ M_1 \oplus M_2 &= M_2 \oplus M_1 & M \oplus 0 &= M \\ M \oplus M &= 0 & (M_1, M_2, M_3) \oplus (K_1, K_2, K_3) &= \\ & & (M_1 \oplus K_1, M_2 \oplus K_2, M_3 \oplus K_3) & \end{aligned}$$

The Need for Equational Theories

We need to enrich the Dolev-Yao model with equations:

- See the Joux example: $\oplus$ is ACUI + homomorphic wrt. pairing;

The Need for Equational Theories

We need to enrich the Dolev-Yao model with equations:

- See the Joux example: $\oplus$ is ACUI + homomorphic wrt. pairing;
- Modeling Diffie-Hellman, i.e., Abelian group laws:

$$(M_1 \times M_2) \times M_3 = M_1 \times (M_2 \times M_3) \qquad M_1 \times M_2 = M_2 \times M_1$$

$$M \times 1 = M \qquad M \times M^{-1} = 1$$

$$(XY)^{-1} = Y^{-1}X^{-1} \quad 1^{-1} = 1 \quad (X^{-1})^{-1} = X$$

plus new adversary rules:

$$\begin{array}{rcl} \texttt{knows}(g^{X \times Y}) & \Leftarrow & \texttt{knows}(g^X), \texttt{knows}(Y) \\ \texttt{knows}(g^1) & & \\ \texttt{knows}(Y^{-1}) & \Leftarrow & \texttt{knows}(Y) \end{array}$$

(This is essentially the generic gr

The Need for Equational Theories

We need to enrich the Dolev-Yao model with equations:

- See the Joux example: $\oplus$ is ACUI + homomorphic wrt. pairing;
- Modeling Diffie-Hellman, i.e., Abelian group laws.
- Modeling RSA (exercise; hard, see e.g. [Kapur-Narendran-Wang, RTA'03]);

The Need for Equational Theories

We need to enrich the Dolev-Yao model with equations:

- See the Joux example: $\oplus$ is ACUI + homomorphic wrt. pairing;
- Modeling Diffie-Hellman, i.e., Abelian group laws.
- Modeling RSA (exercise; hard, see e.g. [Kapur-Narendran-Wang, RTA'03]);
- Representing ciphers (i.e., decryption never fails):

$$dec(\{M\}_K, K) = M \qquad \{dec(M, K)\}_K = M$$

The Need for Equational Theories

We need to enrich the Dolev-Yao model with equations:

- See the Joux example: $\oplus$ is ACUI + homomorphic wrt. pairing;
- Modeling Diffie-Hellman, i.e., Abelian group laws.
- Modeling RSA (exercise; hard, see e.g. [Kapur-Narendran-Wang, RTA'03]);
- Representing ciphers (i.e., decryption never fails).
- Etc.

Formalizing the Adversary's Knowledge. . . Modulo T

$$\frac{}{E, M \vdash_T M}\ (Ax)$$

$$\frac{E \vdash_T M \quad E \vdash_T K}{E \vdash_T \{M\}_K}\ (CryptI) \qquad \frac{E \vdash_T \{M\}_K \quad E \vdash_T K' \quad (K' \text{ inverse of } K)}{E \vdash_T M}\ (CryptE)$$

$$\frac{E \vdash_T M_1 \quad \ldots \quad E \vdash_T M_n}{E \vdash_T (M_1, \ldots, M_n)}\ (TupleI) \qquad \frac{E \vdash_T (M_1, \ldots, M_n)}{E \vdash_T M_i}\ (TupleE_i), 1 \leq i \leq n$$

$$\frac{E \vdash_T M \qquad M \approx_T M'}{E \vdash_T M'}\ (T)$$

Finding Security Proofs Modulo T

- This is considerably harder, and a theme of active research. See [Kapur-Narendran-Wang, RTA'03], [Verma, RTA'03], [Cortier, RTA'03], [Verma, LPAR'03], [Chevalier-Küsters-Rusinowitch-Turuani, FST&TCS'03], [Verma, FST&TCS'04], etc.

Finding Security Proofs Modulo T

- This is considerably harder, and a theme of active research. See [Kapur-Narendran-Wang, RTA'03], [Verma, RTA'03], [Cortier, RTA'03], [Verma, LPAR'03], [Chevalier-Küsters-Rusinowitch-Turuani, FST&TCS'03], [Verma, FST&TCS'04], etc.
- When the number of sessions is bounded,
  - Insecurity with xor, or Abelian groups, is NP-complete [Chevalier-Rusinowitch-Turuani, LICS'03]; intruder deduction is in P.
  - With AC + homomorphism wrt. a hash, intruder deduction in P/NP (depending on coding) [Lafourcade-Treinen-et al.,'05].
  - With xor + homomorphism, or Abelian groups + homomorphism, intruder deduction in P [Delaune-Treinen-et al.,'06].

Outline

A Few Other Successes of Dolev-Yao Style Approaches

- [Pereira-Quisquater, CSFW'01] Group Diffie-Hellman is vulnerable for $n \geq 3$ participants, can be repaired for $n = 3$. (Uses modular exponentiation.)

A Few Other Successes of Dolev-Yao Style Approaches

- [Pereira, CSFW'04] There is no group key agreement protocol based solely on modular exponentiation for $n \geq 4$.

A Few Other Successes of Dolev-Yao Style Approaches

- [Pereira, CSFW'04] There is no group key agreement protocol based solely on modular exponentiation for $n \geq 4$.
- [Chadha-Kremer-Scedrov, CSFW'04] The Garay-MacKenzie multi-party contract signing protocol (1999) is correct with 3 signers, flawed with $n \geq 4$ signers.

  (The sheer complexity of the protocol makes any by-hand analysis unfeasible.)

A Few Other Successes of Dolev-Yao Style Approaches

- [Pereira, CSFW'04] There is no group key agreement protocol based solely on modular exponentiation for $n \geq 4$.
- [Chadha-Kremer-Scedrov, CSFW'04] The Garay-MacKenzie multi-party contract signing protocol (1999) is correct with 3 signers, flawed with $n \geq 4$ signers.
- [Kremer-Mukhamedov-Ritter, Fin.Crypto'05] The González-Markowitch fix of the Franklin-Tsudik multi-party fair exchange protocol is flawed; the new fix is proved using Mocha. Uses modular exponentiation.

A Few Other Successes of Dolev-Yao Style Approaches

- [Pereira, CSFW'04] There is no group key agreement protocol based solely on modular exponentiation for $n \geq 4$.
- [Chadha-Kremer-Scedrov, CSFW'04] The Garay-MacKenzie multi-party contract signing protocol (1999) is correct with 3 signers, flawed with $n \geq 4$ signers.
- [Kremer-Mukhamedov-Ritter, Fin.Crypto'05] The González-Markowitch fix of the Franklin-Tsudik multi-party fair exchange protocol is flawed; the new fix is proved using Mocha. Uses modular exponentiation.
- Early inter-protocol flaws fixed in the Microsoft security architecture [Fournet, Microsoft Research,'04], using Blanchet's tool ProVerif

Outline

The Spi-Calculus

This is another modeling style. Program your protocols in some simple, core language. E.g., the spi-calculus [Abadi-Gordon, CCS'97].

- Expressions denote messages, as in Dolev-Yao:

$$
\begin{array}{lcll}
e, \ldots & ::= & X & \text{variables} \\
 & \mid & f(e_1, \ldots, e_n) & \text{application of constructor } f \\
 & \mid & \{e_1\}_{e_2} & \text{symmetric encryption} \\
 & \mid & [e_1]_{e_2} & \text{asymmetric encryption}
\end{array}
$$

- Processes: next slide.

Spi-Calculus Processes

- Processes are programs, or whole systems, $P, Q, R, \ldots ::=$

	$\texttt{stop}$	stop
$\mid$	$![X]P$	replication
$\mid$	$P \mid Q$	parallel composition
$\mid$	$\texttt{new} X; P$	fresh name creation
$\mid$	$\texttt{out}(e_1, e_2); P$	writing to a channel
$\mid$	$\texttt{in}(e_1, X); P$	reading from a channel
$\mid$	$\texttt{let } X = e \texttt{ in } P$	local definition
$\mid$	$\texttt{case } e_1 \texttt{ of } f(X_1, \ldots, X_n) \Rightarrow P \texttt{ else } Q$	pattern-matching
$\mid$	$\texttt{case } e_1 \texttt{ of } \{X\}_{e_2} \Rightarrow P \texttt{ else } Q$	symmetric decryption
$\mid$	$\texttt{case } e_1 \texttt{ of } [X]_{e_2} \Rightarrow P \texttt{ else } Q$	asymmetric decryption
$\mid$	$\texttt{if } e_1 = e_2 \texttt{ then } P \texttt{ else } Q$	equality test
$\mid$	$f(e_1, \ldots, e_n)$	process call

Symmetric Key Needham-Schroeder in Spi

```
proc alice (to_s, from_s,
            to_b, from_b,
            A, B, Kas) =
   new Na;
   out (to_s, A, B, Na);
   in (from_s,
       {=Na, =B, Kab, M} Kas);
   out (to_b, M);
   in (from_b, {Nb} Kab);
   out (to_b, {s (Nb)} Kab);

proc bob (to_a, from_a, Kbs) =
   in (from_a, {Kab, A} Kbs);
   new Nb;
   out (to_a, {Nb} Kab);
   in (from_a, {=s (Nb)} Kab);
```

```
proc server (from_a, to_a) =
   in (from_a, A, B, Na);
   new Kab;
   out (to_a,
        {Na, B, Kab,
             {Kab, A} kxs(B)}
        kxs(A));

proc main = new c_pub; new a; new b;
            new Kas; new Kbs;
      ![A] ![B]
         ( alice (c_pub, c_pub,
                  c_pub, c_pub,
                  A, B, kxs (A))
         | bob (c_pub, c_pub,
                kxs (B)))
   | !server (c_pub, c_pub)
   );
```

Until Now, Nothing New Under the Sun

- You may translate spi-calculus processes to Horn clauses, losing some precision in passing. This is equivalent to a type system for security [Abadi-Blanchet, POPL'02]. (An undecidable one, by the way.)

Until Now, Nothing New Under the Sun

- You may translate spi-calculus processes to Horn clauses, losing some precision in passing. This is equivalent to a type system for security [Abadi-Blanchet, POPL'02]. (An undecidable one, by the way.)
- You may do the same by aiming at a decidable class. This is what [Nielson-Nielson-Seidl, SAS'02] do. The corresponding “type system” falls into $\mathcal{H}_1$, even in $\mathcal{H}_3$, a cubic-time class.

Until Now, Nothing New Under the Sun

- You may translate spi-calculus processes to Horn clauses, losing some precision in passing. This is equivalent to a type system for security [Abadi-Blanchet, POPL'02]. (An undecidable one, by the way.)
- You may do the same by aiming at a decidable class. This is what [Nielson-Nielson-Seidl, SAS'02] do. The corresponding "type system" falls into $\mathcal{H}_1$, even in $\mathcal{H}_3$, a cubic-time class.

This all decides reachability properties, e.g., secrecy or authentication. In other words, this is Dolev-Yao in disguise.

Until Now, Nothing New Under the Sun

- You may translate spi-calculus processes to Horn clauses, losing some precision in passing. This is equivalent to a type system for security [Abadi-Blanchet, POPL'02]. (An undecidable one, by the way.)
- You may do the same by aiming at a decidable class. This is what [Nielson-Nielson-Seidl, SAS'02] do. The corresponding "type system" falls into $\mathcal{H}_1$, even in $\mathcal{H}_3$, a cubic-time class.

This all decides reachability properties, e.g., secrecy or authentication. In other words, this is Dolev-Yao in disguise. But one can do more with process algebra. . .

Outline

How Many Secret Bits?

Look at the (*TupleI*) rule:

$$\frac{E \vdash M_1 \quad \dots \quad E \vdash M_n}{E \vdash (M_1, \dots, M_n)}$$

This indicates that, to "know" a pair, you need to "know" each component.

How Many Secret Bits?

Look at the (*TupleI*) rule:

$$\frac{E \vdash M_1 \quad \ldots \quad E \vdash M_n}{E \vdash (M_1, \ldots, M_n)}$$

This indicates that, to "know" a pair, you need to "know" each component.
In other words, a message is Dolev-Yao-secret if and only if the adversary cannot know all bits.

How Many Secret Bits?

Look at the (*TupleI*) rule:

$$\frac{E \vdash M_1 \quad \ldots \quad E \vdash M_n}{E \vdash (M_1, \ldots, M_n)}$$

This indicates that, to "know" a pair, you need to "know" each component.
In other words, a message is Dolev-Yao-secret if and only if the adversary cannot know all bits.
This looks ridiculous. In cryptography, M is secret if and only if the adversary cannot know more than a negligible proportion of the bits of M.

Outline

May-Testing Equivalence

Say that two processes P, Q are observationally equivalent ($P \simeq Q$) iff

For every Adv, for every barb β, $(P|Adv) \Downarrow \beta$ iff $(Q|Adv) \Downarrow \beta$

More precisely, this is may-testing equivalence.
A *barb* β is a channel + a direction (input/output).
$R \Downarrow \beta$ means that R may eventually emit/receive on β.

May-Testing Equivalence

Say that two processes P, Q are observationally equivalent ($P \simeq Q$) iff

For every Adv, for every barb β, $(P|Adv) \Downarrow \beta$ iff $(Q|Adv) \Downarrow \beta$

More precisely, this is may-testing equivalence.
A *barb* β is a channel + a direction (input/output).
$R \Downarrow \beta$ means that R may eventually emit/receive on β.

In other words, $P \simeq Q$ iff no adversary Adv can make any difference between P and Q by just looking at, and interfering with their input-output activity.

Strong Secrecy, Strong Authentication

Let $P[M]$ be a process with a distinguished occurrence of M. M is strongly secret in P iff

$$\text{for every } M',\ P[M] \simeq P[M'].$$

In other words, no adversary can change behaviors as a result of observing a difference between M and M'.

Strong Secrecy, Strong Authentication

Let $P[M]$ be a process with a distinguished occurrence of M. M is strongly secret in P iff

$$\text{for every } M',\ P[M] \simeq P[M'].$$

In other words, no adversary can change behaviors as a result of observing a difference between M and M'.

Let $P[M, \mathtt{in}(e, X); Q]$ be a process with a distinguished occurrence of a message M and a distinguished occurrence of a sub-process $\mathtt{in}(e, X); Q$. X is strongly authentically M iff

$$P[M, \mathtt{in}(e, X); Q] \simeq P[M, \mathtt{in}(e, X); Q[X := M]]$$

(The second process throws X away and “magically” gets M.)

How Many Secret Bits (Again)?

One may argue that M is strongly secret iff no predicate of M can be used by Adv to make a decision.

How Many Secret Bits (Again)?

One may argue that M is strongly secret iff no predicate of M can be used by Adv to make a decision.
In other words, the adversary knows strictly less than 1 bit of M.
(Similarly for authentication.)

How Many Secret Bits (Again)?

One may argue that M is strongly secret iff no predicate of M can be used by Adv to make a decision.
In other words, the adversary knows strictly less than 1 bit of M.
(Similarly for authentication.)

Paradoxically, this is sometimes too much. E.g., in a password-based authentication system, the adversary will know whether the typed password is valid or not.

How Many Secret Bits (Again)?

One may argue that *M* is strongly secret iff no predicate of *M* can be used by *Adv* to make a decision.
In other words, the adversary knows strictly less than 1 bit of *M*.
(Similarly for authentication.)

Paradoxically, this is sometimes too much. E.g., in a password-based authentication system, the adversary will know whether the typed password is valid or not.

Funnily, we threw Dolev-Yao out the door, and it will come back through the window.

What Are Bisimulations?

- A classic way to establish observational equivalence.
- Usually takes the form of:

  A bisimulation is a relation $\approx$ between processes such that

$$
\begin{tikzcd}
P \arrow[d] & \approx & P' \arrow[d, dotted] \\
Q & \approx & Q'
\end{tikzcd}
\qquad
\begin{tikzcd}
P \arrow[d, dotted] & \approx & P' \arrow[d] \\
Q & \approx & Q'
\end{tikzcd}
$$

  Usually, $\approx$ implies $\simeq$; with some luck, the converse holds.

What Are Bisimulations?

- A classic way to establish observational equivalence.
- Usually takes the form of:

  A bisimulation is a relation $\approx$ between processes such that

$$\begin{tikzcd} P \arrow[d] & \approx & P' \arrow[d, dotted] \\ Q & \approx & Q' \end{tikzcd} \qquad \begin{tikzcd} P \arrow[d, dotted] & \approx & P' \arrow[d] \\ Q & \approx & Q' \end{tikzcd}$$

  Usually, $\approx$ implies $\simeq$; with some luck, the converse holds.
- Complex in the case of the spi-calculus: encryption, creation of fresh names complicate the picture.

What Are Bisimulations?

- A classic way to establish observational equivalence.
- Usually takes the form of:

  A bisimulation is a relation $\approx$ between processes such that

$$\begin{tikzcd} P \arrow[d] & \approx & P' \arrow[d, dotted] \\ Q & \approx & Q' \end{tikzcd} \qquad \begin{tikzcd} P \arrow[d, dotted] & \approx & P' \arrow[d] \\ Q & \approx & Q' \end{tikzcd}$$

  Usually, $\approx$ implies $\simeq$; with some luck, the converse holds.
- Any non-trivial notion of observational equivalence is undecidable [Hüttel, INFINITY'02], even for finite-control spi processes. In particular, framed bisimulation [Abadi-Gordon, NJC'98], also [Boreale-de-Nicola-Pugliese, LICS'99]

Hedged Bisimulation [Borgström-Nestmann,AMAST'04]

Refinement of [Boreale-de-Nicola-Pugliese, LICS'99], characterizes observational equivalence. On terms:

- Hedges *th* are finite sets of pairs (M_1, M_2) in each world that are thought to be indistinguishable.

$$\frac{(M_1, M_2) \in th}{th \vdash M_1 \approx M_2} \qquad \frac{th \vdash M_1 \approx M_2 \quad th \vdash K_1 \approx K_2}{th \vdash \{M_1\}_{K_1} \approx \{M_2\}_{K_2}}$$

Hedged Bisimulation [Borgström-Nestmann,AMAST'04]

Refinement of [Boreale-de-Nicola-Pugliese, LICS'99], characterizes observational equivalence. On terms:

- Hedges *th* are finite sets of pairs (M_1, M_2) in each world that are thought to be indistinguishable.

$$\frac{(M_1, M_2) \in th}{th \vdash M_1 \approx M_2} \qquad \frac{th \vdash M_1 \approx M_2 \quad th \vdash K_1 \approx K_2}{th \vdash \{M_1\}_{K_1} \approx \{M_2\}_{K_2}}$$

- A hedge is consistent iff relates names with names, encryptions with encryptions, is a partial bijection, and

$$\frac{th \vdash \{M_1\}_{K_1} \approx \{M_2\}_{K_2}}{K_1 \not\in \mathrm{dom}\ th \quad K_2 \not\in \mathrm{codom}\ th}$$

Hedged Bisimulation [Borgström-Nestmann,AMAST'04]

Refinement of [Boreale-de-Nicola-Pugliese, LICS'99], characterizes observational equivalence. On terms:

- It was observed in [JGL-Lasota-Nowak-Zhang, CSL'04] that it is equivalent to throw *th* away, and define $th \vdash \_ \approx \_$ directly, subject to the proviso that:

$$\frac{th \vdash M_1 \approx M_2 \quad th \vdash K_1 \approx K_2}{th \vdash \{M_1\}_{K_1} \approx \{M_2\}_{K_2}} \qquad \frac{th \vdash M_1 \approx M_2 \quad th \vdash K_1 \approx K_2}{th \vdash dec(M_1, K_1) \approx dec(M_2, K_2)}$$

where $dec(\{M\}_K, K) = M$, $dec(M, K) = \perp$ otherwise.

Hedged Bisimulation [Borgström-Nestmann,AMAST'04]

Refinement of [Boreale-de-Nicola-Pugliese, LICS'99], characterizes observational equivalence. On terms:

- It was observed in [JGL-Lasota-Nowak-Zhang, CSL'04] that it is equivalent to throw *th* away, and define $th \vdash \_ \approx \_$ directly, subject to the proviso that:

$$\frac{th \vdash M_1 \approx M_2 \quad th \vdash K_1 \approx K_2}{th \vdash \{M_1\}_{K_1} \approx \{M_2\}_{K_2}} \qquad \frac{th \vdash M_1 \approx M_2 \quad th \vdash K_1 \approx K_2}{th \vdash dec(M_1, K_1) \approx dec(M_2, K_2)}$$

where $dec(\{M\}_K, K) = M$, $dec(M, K) = \perp$ otherwise.
Dolev-Yao strikes again... in binary form.

Hedged Bisimulation Per Se

- Definition omitted: rather horrible, see [Borgström-Nestmann, AMAST'04], Definition 15.
  - Quantifies universally over extensions of $h \vdash \_ \approx \_$ on receiving message;
  - Quantifies existentially over extensions relating sent messages M_1 and M_2 on message sends.

Hedged Bisimulation Per Se

- Definition omitted: rather horrible, see [Borgström-Nestmann, AMAST'04], Definition 15.
  - Quantifies universally over extensions of $h \vdash \_ \approx \_$ on receiving message;
  - Quantifies existentially over extensions relating sent messages M_1 and M_2 on message sends.
- Despite undecidability, you can extract sound reasoning principles, see [Boreale-Gorla, JTIT'02], and next slide. Implemented by Boreale and Gorla.

Sound Reasoning Principles

$$\frac{P \equiv Q}{(\sigma, \sigma) \vdash P \approx Q}$$

$$\frac{}{(\sigma, \sigma) \vdash C[P + \mathtt{if}\ M = M\ \mathtt{then}\ Q] \approx C[P + Q]}$$

$$\frac{M \text{ not a name bound in } \mathtt{new} n;\ C[\bullet]}{(\sigma, \sigma) \vdash \mathtt{new} n;\ C[P + \mathtt{if}\ n = M\ \mathtt{then}\ Q] \approx \mathtt{new} n;\ C[P]}$$

etc.

Bisimulations and Biprocesses

- An old idea, well-known to category-theorists [Joyal-Nielsen-Winskel, I&C'96]:

  A bisimulation between two transitions systems δ_1 on state set Q_1 and δ_2 on Q_2 is a transition system δ on $Q_1 \times Q_2$ such that $\pi_1(\delta) = \delta_1$ and $\pi_2(\delta) = \delta_2$.

Bisimulations and Biprocesses

- An old idea, well-known to category-theorists [Joyal-Nielsen-Winskel, I&C'96]:

  A bisimulation between two transitions systems δ_1 on state set Q_1 and δ_2 on Q_2 is a transition system δ on $Q_1 \times Q_2$ such that $\pi_1(\delta) = \delta_1$ and $\pi_2(\delta) = \delta_2$.
- Can give concrete representations of δ when δ_1 and δ_2 arise from processes with same control (only messages change), see [Abadi-Blanchet-Fournet, LICS'05], idea from [Simonet, POPL'03?]

Bisimulations and Biprocesses

- An old idea, well-known to category-theorists [Joyal-Nielsen-Winskel, I&C'96]:

  A bisimulation between two transitions systems δ_1 on state set Q_1 and δ_2 on Q_2 is a transition system δ on $Q_1 \times Q_2$ such that $\pi_1(\delta) = \delta_1$ and $\pi_2(\delta) = \delta_2$.
- Can give concrete representations of δ when δ_1 and δ_2 arise from processes with same control (only messages change), see [Abadi-Blanchet-Fournet, LICS'05], idea from [Simonet, POPL'03?]
  E.g., share $\mathtt{out}(e_1, e_2)$; and $\mathtt{out}(e_1', e_2')$; as

$$outC(diff(e_1, e_1'), diff(e_2, e_2'))$$

Security against Off-Line Guessing Attacks

Hmm, we probably won't have time for this. See [Baudet, SSP'05].
Idea [Lowe] is that some protocols are secure, although secret M is encrypted with a weak secret K (e.g., a password). You find K by enumeration, but cannot test whether K is the right one if you cannot recognize the right M.
Uses equational theories in an essential way.

Computational Notions of Security

- Familiar to cryptographers.
- Reduce security of protocol to security of basic cryptographic primitives, quantifying the probabilistic advantage that this gives to the adversary.

This seems very different from what we did above. But...

Relating Computational and Formal Security

We can in fact relate computational security with formal methods.

- A very trendy topic these days.

Relating Computational and Formal Security

We can in fact relate computational security with formal methods.

- A very trendy topic these days.
- Pioneering papers: [Abadi-Rogaway, IFIP-TCS'00] (using patterns—with a Dolev-Yao twist!), [Warinschi, CSFW'03] (comp. proof of Needham-Schroeder-Lowe), [Micciancio-Warinschi, JCS'04], etc.

Relating Computational and Formal Security

We can in fact relate computational security with formal methods.

- A very trendy topic these days.
- Pioneering papers: [Abadi-Rogaway, IFIP-TCS'00] (using patterns—with a Dolev-Yao twist!), [Warinschi, CSFW'03] (comp. proof of Needham-Schroeder-Lowe), [Micciancio-Warinschi, JCS'04], etc.
- with a Las Vegas model of computation, purely reduces to Dolev-Yao [Degano-Zunino, FoSSaCs'04; Baudet, JALC'05], under mild assumptions;
- FormaCrypt project: Blanchet, Pointcheval, Baudet, JGL, Cortier, Abadi, Kremer, Warinschi, Lubicz, just started.

Relating Computational and Formal Security

We can in fact relate computational security with formal methods.

- A very trendy topic these days.
- Pioneering papers: [Abadi-Rogaway, IFIP-TCS'00] (using patterns—with a Dolev-Yao twist!), [Warinschi, CSFW'03] (comp. proof of Needham-Schroeder-Lowe), [Micciancio-Warinschi, JCS'04], etc.
- with a Las Vegas model of computation, purely reduces to Dolev-Yao [Degano-Zunino, FoSSaCs'04; Baudet, JALC'05], under mild assumptions;
- FormaCrypt project: Blanchet, Pointcheval, Baudet, JGL, Cortier, Abadi, Kremer, Warinschi, Lubicz, just started.
- Ask me, or Mathieu Baudet, his 72-slide presentation on

Outline

What Can We Do Here?

Two ideas I have started but not really developed yet:

- Use linear logic. $\texttt{knows}(M_1)\&\texttt{knows}(M_2)$ means that adversary knows $\alpha|M_1\rangle + \beta|M_2\rangle$.

What Can We Do Here?

Two ideas I have started but not really developed yet:

- Use linear logic. $\texttt{knows}(M_1) \& \texttt{knows}(M_2)$ means that adversary knows $\alpha|M_1\rangle + \beta|M_2\rangle$.
  Good match with the rules of logic: adversary can then, say, use $\texttt{knows}(M_1)$ to pursue an attack, but will lose $\texttt{knows}(M_2)$ ($\approx$ measurement). (Ask me the draft. . .)

What Can We Do Here?

Two ideas I have started but not really developed yet:

- Use linear logic. $\mathtt{knows}(M_1)\&\mathtt{knows}(M_2)$ means that adversary knows $\alpha|M_1\rangle + \beta|M_2\rangle$.
  Good match with the rules of logic: adversary can then, say, use $\mathtt{knows}(M_1)$ to pursue an attack, but will lose $\mathtt{knows}(M_2)$ ($\approx$ measurement). (Ask me the draft...)
  Unfortunately, since probabilities are lost (as in methods above), no hope of proving [BB84] or [Ekert91], say.

What Can We Do Here?

Two ideas I have started but not really developed yet:

- Use linear logic. $\mathtt{knows}(M_1)\&\mathtt{knows}(M_2)$ means that adversary knows $\alpha|M_1\rangle + \beta|M_2\rangle$.
  Good match with the rules of logic: adversary can then, say, use $\mathtt{knows}(M_1)$ to pursue an attack, but will lose $\mathtt{knows}(M_2)$ ($\approx$ measurement). (Ask me the draft. . .)
  Unfortunately, since probabilities are lost (as in methods above), no hope of proving [BB84] or [Ekert91], say.
- Use classical logic but with terms modulo the theory of density matrices. I.e., pursue the equational theory theme. Less exciting, but might work better.