Verification of cryptographic protocols: techniques, tools and link to cryptanalysis

Véronique Cortier

INRIA project Cassis, Loria

CNRS, Nancy, France

Context: cryptographic protocols

- Widely used: web (SSH, SSL, ...), pay-per-view, electronic purse, mobile phone, ...
- Should ensure: confidentiality authenticity integrity anonymity, ...

Context: cryptographic protocols

- Widely used: web (SSH, SSL, ...), pay-per-view, electronic purse, mobile phone, ...
- Should ensure: confidentiality authenticity integrity anonymity, ...
- Presence of an attacker
  - may read every message sent on the net,
  - may intercept and send new messages.

Credit Card Payment Protocol

- The waiter introduces the credit card.
- The waiter enters the amount m of the transaction on the terminal.
- The terminal authenticates the card.
- The customer enters his secret code.
  If the amount m is greater than 100 euros
  (and in only 20% of the cases)
  - The terminal asks the bank for the authentication of the card.
  - The bank provides the authentication.

More details

4 actors : the Bank, the Customer, the Card and Terminal.

Bank owns

- a signing key K_B^{-1}, secret,
- a verification key K_B, public,
- a secret symmetric key for each credit card K_{CB}, secret.

Card owns

- Data : last name, first name, card's number, expiration date,
- Signature's Value $VS = \{hash(\mathsf{Data})\}_{K_B^{-1}}$,
- secret key K_{CB}.

Terminal owns the verification key K_B for bank's signatures.

Credit card payment Protocol (in short)

The terminal reads the card:

$$1. \quad Ca \quad \rightarrow \quad T : \mathsf{Data}, \{hash(\mathsf{Data})\}_{K_B^{-1}}$$

Credit card payment Protocol (in short)

The terminal reads the card:

$$1. \quad Ca \quad \rightarrow \quad T : \mathsf{Data}, \{hash(\mathsf{Data})\}_{K_B^{-1}}$$

The terminal asks for the secret code:

$$\begin{array}{llll} 2. & T & \rightarrow & Cu : secret\ code? \\ 3. & Cu & \rightarrow & Ca : 1234 \\ 4. & Ca & \rightarrow & T : ok \end{array}$$

Credit card payment Protocol (in short)

The terminal reads the card:

$$1.\quad Ca \quad \rightarrow \quad T : \mathsf{Data}, \{hash(\mathsf{Data})\}_{K_B^{-1}}$$

The terminal asks for the secret code:

$$\begin{array}{llll} 2. & T & \rightarrow & Cu : secret\ code? \\ 3. & Cu & \rightarrow & Ca : 1234 \\ 4. & Ca & \rightarrow & T : ok \end{array}$$

The terminal calls the bank:

$$\begin{array}{llll} 5. & T & \rightarrow & B : auth? \\ 6. & B & \rightarrow & T : N_b \\ 7. & T & \rightarrow & Ca : N_b \\ 8. & Ca & \rightarrow & T : \{N_b\}_{K_{CB}} \\ 9. & T & \rightarrow & B : \{N_b\}_{K_{CB}} \\ 10. & B & \rightarrow & T : ok \end{array}$$

Some flaws

The security was initially ensured by:

- the cards were very difficult to reproduce,
- the protocol and the keys were secret.

But

- cryptographic flaw: 320 bits keys can be broken (1988),
- logical flaw: no link between the secret code and the authentication of the card,
- fake cards can be build.

Some flaws

The security was initially ensured by:

- the cards were very difficult to reproduce,
- the protocol and the keys were secret.

But

- cryptographic flaw: 320 bits keys can be broken (1988),
- logical flaw: no link between the secret code and the authentication of the card,
- fake cards can be build.

$\rightarrow$ "YesCard" build by Serge Humpich (1998).

How does the “YesCard” work?

Logical flaw

$$\begin{array}{llll} 1. & Ca & \rightarrow T & : \mathsf{Data}, \{hash(\mathsf{Data})\}_{K_B^{-1}} \\ 2. & T & \rightarrow Ca & : secret\ code? \\ 3. & Cu & \rightarrow Ca & : 1234 \\ 4. & Ca & \rightarrow T & : ok \end{array}$$

How does the “YesCard” work?

Logical flaw

$$
\begin{array}{llll}
1. & Ca & \rightarrow T & : \mathsf{Data}, \{hash(\mathsf{Data})\}_{K_B^{-1}} \\
2. & T & \rightarrow Ca & : secret\ code? \\
3. & Cu & \rightarrow Ca' & : 2345 \\
4. & Ca' & \rightarrow T & : ok
\end{array}
$$

How does the “YesCard” work?

Logical flaw

$$\begin{array}{llll} 1. & Ca & \rightarrow T & : \mathsf{Data}, \{hash(\mathsf{Data})\}_{K_B^{-1}} \\ 2. & T & \rightarrow Ca & : secret\ code? \\ 3. & Cu & \rightarrow Ca' & : 2345 \\ 4. & Ca' & \rightarrow T & : ok \end{array}$$

Remark: there is always somebody to debit.
$\longrightarrow$ creation of a fake card (Serge Humpich).

How does the “YesCard” work?

Logical flaw

$$\begin{array}{llll} 1. & Ca & \rightarrow T & : \mathsf{Data}, \{hash(\mathsf{Data})\}_{K_B^{-1}} \\ 2. & T & \rightarrow Ca & : secret\ code? \\ 3. & Cu & \rightarrow Ca' & : 2345 \\ 4. & Ca' & \rightarrow T & : ok \end{array}$$

Remark: there is always somebody to debit.
$\longrightarrow$ creation of a fake card (Serge Humpich).

$$\begin{array}{llll} 1. & Ca' & \rightarrow T & : \mathsf{XXX}, \{hash(\mathsf{XXX})\}_{K_B^{-1}} \\ 2. & T & \rightarrow Cu & : secret\ code? \\ 3. & Cu & \rightarrow Ca' & : 0000 \\ 4. & Ca' & \rightarrow T & : ok \end{array}$$

Map

1. Formal approaches
2. Tools and case study
3. Link between formal approaches and cryptanalysis

Formal approaches

- Messages are abstracted using terms.
  These terms are build over a fixed signature.
  E.g., $\Sigma = \{< >, \mathsf{enc}, \mathsf{dec}, \ldots\}$.

Formal approaches

- Messages are abstracted using terms.
  These terms are build over a fixed signature.
  E.g., $\Sigma = \{<>, \mathsf{enc}, \mathsf{dec}, \ldots\}$.
- The attacker can do symbolic manipulations on terms.

$$\frac{S \vdash \mathsf{enc}(M, k) \quad S \vdash k^{-1}}{S \vdash M} \qquad \frac{S \vdash \langle M_1, M_2 \rangle}{S \vdash M_i}\; i = 1, 2$$

Formal approaches

- Messages are abstracted using terms.
  These terms are build over a fixed signature.
  E.g., $\Sigma = \{<>, \mathsf{enc}, \mathsf{dec}, \ldots\}$.
- The attacker can do symbolic manipulations on terms.

$$\frac{S \vdash \mathsf{enc}(M, k) \quad S \vdash k^{-1}}{S \vdash M} \qquad\qquad \frac{S \vdash \langle M_1, M_2 \rangle}{S \vdash M_i}\ i = 1, 2$$

This approach allows to detect any logical attack that does not rely on weaknesses of the encryption algorithm.

Protocol description

Protocol:

$$\begin{array}{lll} T & \to Ca: & N_b \\ Ca & \to T: & \{N_b\}_{K_{CB}} \end{array}$$

$$\frac{S \vdash x}{S \vdash \{x\}_{K_{CB}}}$$

Secrecy properties:

$$S \vdash s?$$

Decidability and complexity results

- In general, secrecy preservation is undecidable.
- For a bounded number of sessions, secrecy is co-NP-complete [RusinowitchTuruani CSFW01]
  $\longrightarrow$ constraint solving
- For an unbounded number of sessions
  - for one-copy protocols, secrecy is DEXPTIME-complete [CortierComon RTA03] [SeildVerma LPAR04]
    $\longrightarrow$ tree automata, resolution theorem proving
  - for message-length bounded protocols, secrecy is DEXPTIME-complete [Durgin et al FMSP99] [Chevalier et al CSL03]

Adding algebraic operators

Some cryptographic primitives have algebraic properties.

- XOR

$$\begin{aligned} x \oplus (y \oplus z) &= (x \oplus y) \oplus z \\ x \oplus y &= y \oplus x \\ x \oplus x &= 0 \\ x \oplus 0 &= x \end{aligned}$$

Adding algebraic operators

Some cryptographic primitives have algebraic properties.

- XOR

$$\begin{aligned} x \oplus (y \oplus z) &= (x \oplus y) \oplus z \\ x \oplus y &= y \oplus x \\ x \oplus x &= 0 \\ x \oplus 0 &= x \end{aligned}$$

- Modular exponentiation

$$\begin{aligned} \mathsf{exp}(\mathsf{exp}(g, x), y) &= \mathsf{exp}(g, x \cdot y) \\ \mathsf{exp}(g, x \cdot y) &= \mathsf{exp}(g, y \cdot x) \end{aligned}$$

Adding algebraic operators

Some cryptographic primitives have algebraic properties.

- XOR

$$\begin{aligned} x \oplus (y \oplus z) &= (x \oplus y) \oplus z \\ x \oplus y &= y \oplus x \\ x \oplus x &= 0 \\ x \oplus 0 &= x \end{aligned}$$

- Modular exponentiation

$$\begin{aligned} \mathsf{exp}(\mathsf{exp}(g, x), y) &= \mathsf{exp}(g, x \cdot y) \\ \mathsf{exp}(g, x \cdot y) &= \mathsf{exp}(g, y \cdot x) \end{aligned}$$

- Homomorphism $\quad h(x \cdot y) = h(x) \cdot h(y)$

Adding algebraic operators

Some cryptographic primitives have algebraic properties.

- XOR

$$\begin{aligned} x \oplus (y \oplus z) &= (x \oplus y) \oplus z \\ x \oplus y &= y \oplus x \\ x \oplus x &= 0 \\ x \oplus 0 &= x \end{aligned}$$

- Modular exponentiation

$$\begin{aligned} \mathsf{exp}(\mathsf{exp}(g, x), y) &= \mathsf{exp}(g, x \cdot y) \\ \mathsf{exp}(g, x \cdot y) &= \mathsf{exp}(g, y \cdot x) \end{aligned}$$

- Homomorphism $h(x \cdot y) = h(x) \cdot h(y)$

$\longrightarrow$ These properties are modeled using equational theories or by extending the intruder power.

Some results with algebraic operators

Deducibility

- homomorphism NP-complete, homomorphism + XOR or Abelian groups EXPTIME [Lafourcade et al RTA05]
- convergent subterm theories, extension to AC properties [AbadiCortier Icalp04, CSFW05]

Bounded number of sessions

- Commutativity co-NP-complete [Chevalier et al ARSPA04]
- Exclusive Or co-NP-complete [Chevalier et al LICS03] [ComonShmatikov LICS03]
- Abelian groups + modular exponentiation (Diffie-Hellman) co-NP-complete [Chevalier et al FSTTCS03]

Unbounded number of sessions

- Exclusive Or decidable for one-copy protocols [ComonCortier RTA03]

Map

1. Formal approaches
2. Tools and case study
3. Link between formal approaches and cryptanalysis

The European project Avispa

Automated Validation of Internet Security Protocols and Applications

In collaboration with:

- Artificial Intelligence Laboratory, DIST, Univ. of Genova, Italy
- Eidgenoessische Technische Hochschule Zuerich (ETHZ), Zurich, Swiss
- Siemens Aktiengesellschaft, Munich, Germany

The European project Avispa

Automated Validation of Internet Security Protocols and Applications

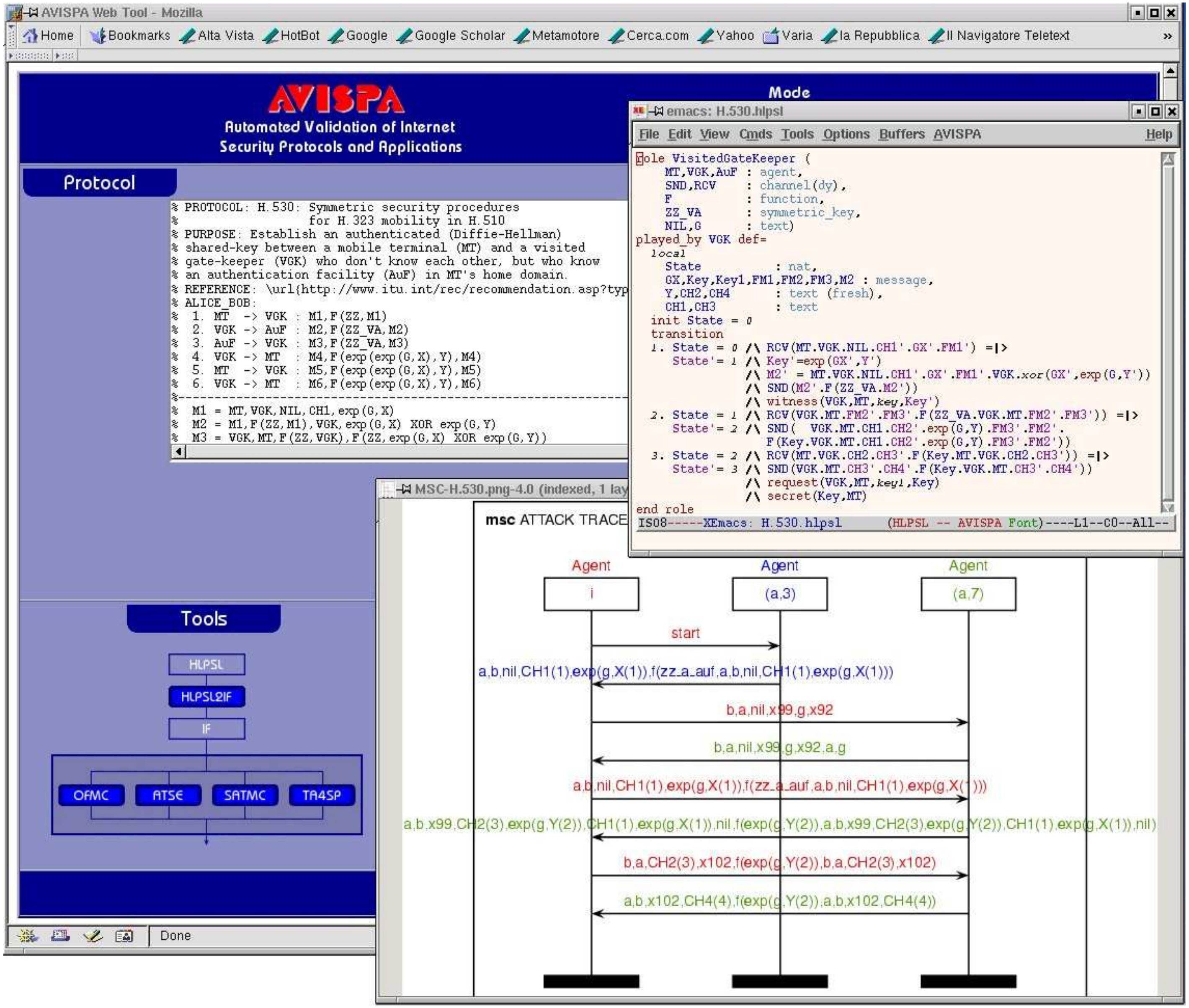
In collaboration with:

- Artificial Intelligence Laboratory, DIST, Univ. of Genova, Italy
- Eidgenoessische Technische Hochschule Zuerich (ETHZ), Zurich, Swiss
- Siemens Aktiengesellschaft, Munich, Germany

Four verification tools are proposed:

- On-the-fly Model-Checker (OFMC)
- Constraint-Logic-based Attack Searcher (CL-AtSe)
- SAT-based Model-Checker (SATMC)
- Tree Automata based on Automatic Approximations for the Analysis of Security Protocols (TA4SP)

The Avispa Platform: www.avispa-project.org



Results

- over 80 protocols analyzed (selected by Siemens and discussed by the IETF) in few minutes or few seconds for most of them
- tools for both a bounded number of sessions (search for attacks) and an unbounded number of sessions (security proof)
- first tool that allows algebraic properties (XOR)
- new attacks have been discovered
- publicly available: web interface, download, protocol library, ...
- already used by 45 sites including several companies (France Telecom, Siemens, SAP,...)

Other case study: Validation of a contactless electronic purse of France Telecom (RNTL project PROUVE)

Map

1. Formal approaches
2. Tools and case study
3. Link between formal approaches and cryptanalysis:
   A new branch of research in the Cassis team

Formal and Cryptographic approaches

	Formal approach	Cryptographic approach
Messages	terms	bitstrings
Encryption	idealized	algorithm
Adversary	idealized	any polynomial algorithm
Proof	automatic	by hand, tedious and error-prone

Link between the two approaches ?

Formal model: several abstractions

Messages are modeled by terms.

- $\{m\}_k$: message m encrypted by k
- $\langle m_1, m_2\rangle$: pair of m_1 and m_2
- ...

$\longrightarrow$ no collisions:

$$\forall m, m', k, k' \quad \{m\}_k \neq \{m'\}_{k'}, \{\{m\}_k\}_k \neq m, \langle m, m'\rangle \neq \{m\}_k, \ldots$$

Formal model: several abstractions

Messages are modeled by terms.

- $\{m\}_k$: message m encrypted by k
- $\langle m_1, m_2 \rangle$: pair of m_1 and m_2
- ...

$\longrightarrow$ no collisions:

$$\forall m, m', k, k' \quad \{m\}_k \neq \{m'\}_{k'}, \{\{m\}_k\}_k \neq m, \langle m, m' \rangle \neq \{m\}_k, \ldots$$

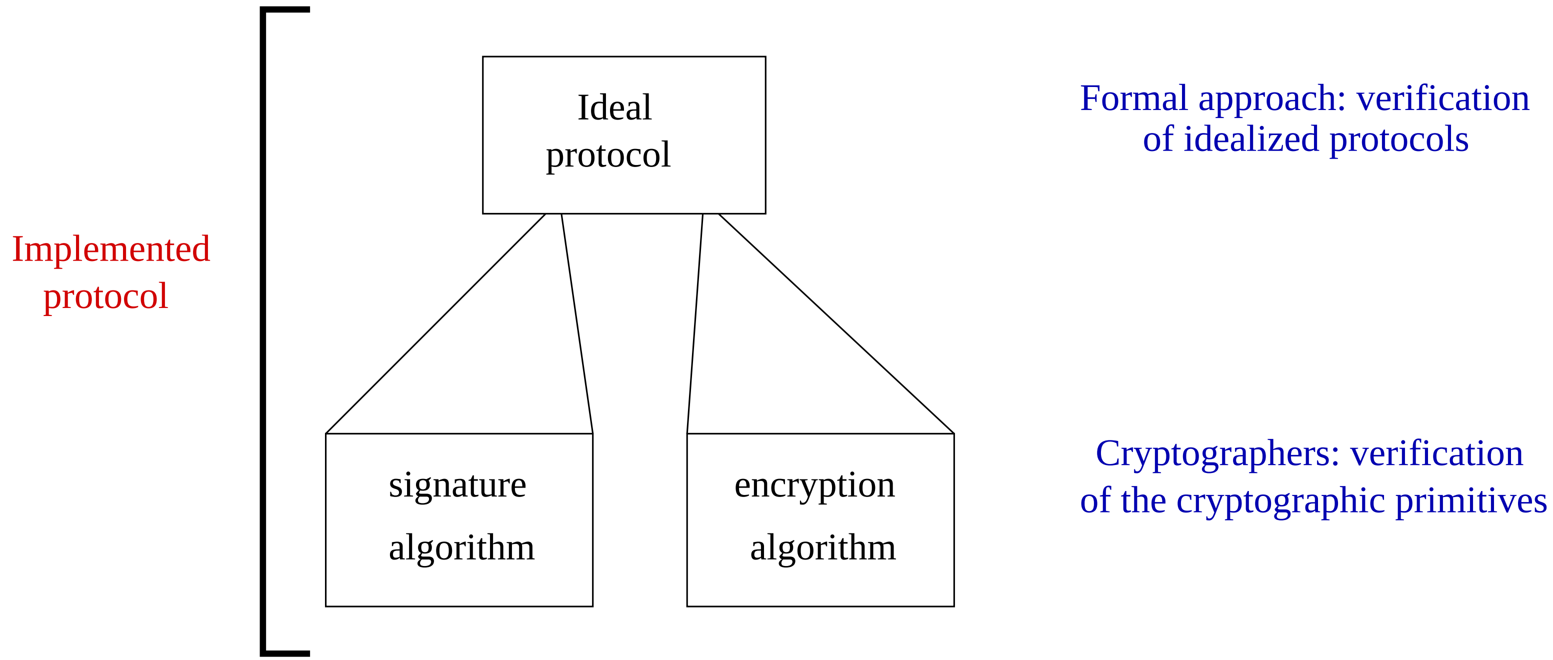
Perfect encryption assumption:

Nothing can be learned from $\{m\}_k$ except if k is known.

$\longrightarrow$ The intruder can perform only specific actions like pairing and encrypting messages or decrypting whenever he has the inverse key.

Goal: soundness of the formal model

Composition of two approaches



Three approaches

1. A computationally sound logic for proving security properties for cryptographic protocols [Datta et al Icalp05]
   This enables a symbolic analysis of the protocol that has a computational interpretation

Three approaches

1. A computationally sound logic for proving security properties for cryptographic protocols [Datta et al Icalp05]
   This enables a symbolic analysis of the protocol that has a computational interpretation

2. Computational soundness of a Dolev-Yao like model [CortierWarinschi ESOP05]
   Existing formal models with asymmetric encryption and signatures are computationally sound, which allows the use of existing automatic tools

Three approaches

1. A computationally sound logic for proving security properties for cryptographic protocols [Datta et al Icalp05]
   This enables a symbolic analysis of the protocol that has a computational interpretation

2. Computational soundness of a Dolev-Yao like model [CortierWarinschi ESOP05]
   Existing formal models with asymmetric encryption and signatures are computationally sound, which allows the use of existing automatic tools

3. Computationally Sound Implementations of Equational Theories against Passive Adversaries [BaudetCortierKremer Icalp05]
   In particular, soundness of the Exclusive Or and soundness of deterministic symmetric encryption.

Three approaches

1. A computationally sound logic for proving security properties for cryptographic protocols [Datta et al Icalp05]
   This enables a symbolic analysis of the protocol that has a computational interpretation

2. Computational soundness of a Dolev-Yao like model
   [CortierWarinschi ESOP05]
   Existing formal models with asymmetric encryption and signatures are computationally sound, which allows the use of existing automatic tools

3. Computationally Sound Implementations of Equational Theories against Passive Adversaries [BaudetCortierKremer Icalp05]
   In particular, soundness of the Exclusive Or and soundness of deterministic symmetric encryption.

Secrecy Properties

Formal models : property on traces

A data s is secret if the adversary (which can only do symbolic manipulations on terms) can not produce s.

Concrete model : indistinguishability

The adversary (any polynomial time algorithm) should not be able to guess a bit of the secret.

Hypotheses on the Implementation

- asymmetric encryption : IND-CCA2
  $\longrightarrow$ the adversary cannot distinguish between $\{n_0\}_k$ and $\{n_1\}_k$
  even if he has access to encryption and decryption oracles.

Hypotheses on the Implementation

- asymmetric encryption : IND-CCA2
  $\longrightarrow$ the adversary cannot distinguish between $\{n_0\}_k$ and $\{n_1\}_k$ even if he has access to encryption and decryption oracles.
- signature : existentially unforgeable under chosen-message attack
  i.e. one can not produce a valid pair (m, σ)

Hypotheses on the Implementation

- asymmetric encryption : IND-CCA2
  $\longrightarrow$ the adversary cannot distinguish between $\{n_0\}_k$ and $\{n_1\}_k$ even if he has access to encryption and decryption oracles.
- signature : existentially unforgeable under chosen-message attack
  i.e. one can not produce a valid pair (m, σ)
- parsing :
  - each bit-string has a label which indicates his type (identity, nonce, key, signature, ...)
  - one can retrieve the (public) encryption key from an encrypted message.
  - one can retrieve the signed message from the signature

Combination result

The perfect public key encryption corresponds to the IND-CCA2 security notion

Theorem : [Cortier-Warinschi Esop'05] (work initiated by Micciancio-Warinschi TCC'04)

- for protocols with only public key encryption and signatures
- if a protocol is secure in the formal approach (proof given by a tool for example),
- if the public key encryption algorithm is IND-CCA2,
- if the signature is existentially unforgeable,

then the protocol is secure in the cryptographic approach.

Some future directions

- Group protocols - open-ended data structures (transaction list, message transducers, ...)
- Contract-signing protocol - complex properties such as fairness and abuse-freeness (no party can prove to a third party that it has the power to both enforce and cancel the contract)
- Link between the symbolic and computational models - further work: refinement of the symbolic models, new security properties, new cryptographic primitives, what are the limits?

French collaborations on that subject

- LIENS, ENS Ulm
- LIF, Marseille
- LSV, ENS de Cachan (RNTL project PROUVE)
- Verimag, Grenoble (RNTL project PROUVE)