

# Verification of cryptographic protocols: relating computational and formal models

Véronique Cortier  
CNRS-Loria, Nancy

# Summary

---

- I. Introduction to computational models
- II. Comparing computational and formal models
- III. Soundness of formal models w.r.t. computational models

# Summary

---

## I. Introduction to computational models

1. Encryption
2. Cryptographic assumptions
3. Security notions

*This part is inspired from David Pointcheval presentations.*

# Summary

---

## I. Introduction to computational models

1. Encryption
2. Cryptographic assumptions
3. Security notions

*This part is inspired from David Pointcheval presentations.*

# Encryption: ancient period

---

- Caesar encryption: moving letters forward
- Cipher disk (Léone Battista Alberti 1466)



# Encryption: ancient period

---

- Caesar encryption: moving letters forward
- Cipher disk (Léone Battista Alberti 1466)



→ subject to statistical analysis

# Encryption: technical period

---

Automatic substitutions and permutations



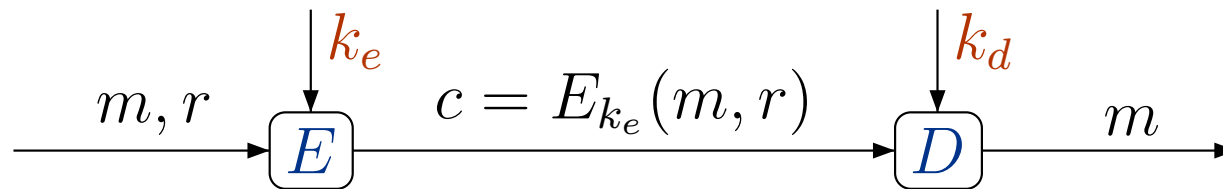
Enigma

# Encryption today

---

## Three algorithms

- $G$  - key generation
- $E$  - encryption
- $D$  - decryption

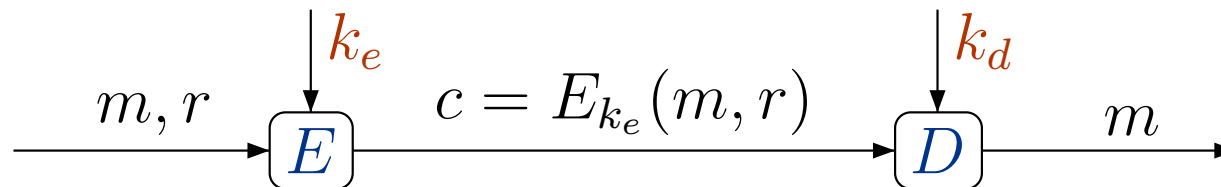


# Encryption today

---

## Three algorithms

- $G$  - key generation
- $E$  - encryption
- $D$  - decryption



$k_e = k_d$  **symmetric** encryption

$k_e \neq k_d$  **asymmetric** encryption

# Secrecy of asymmetric encryption

---

Known data:

- $c = E_{k_e}(m, r)$  ciphertext
- $k_e$  encryption key

A unique  $m$  satisfies the relation (with possibly several  $r$ )

→ At least an exhaustive search on  $m$  and  $r$  can lead to  $m$  !

# Secrecy of asymmetric encryption

---

Known data:

- $c = E_{k_e}(m, r)$  ciphertext
- $k_e$  encryption key

A unique  $m$  satisfies the relation (with possibly several  $r$ )

→ At least an exhaustive search on  $m$  and  $r$  can lead to  $m$  !

⇒ unconditional secrecy is impossible, we need algorithmic assumptions

# Summary

---

## I. Introduction to computational models

1. Encryption
2. Cryptographic assumptions
3. Security notions

# Integer Factoring and RSA

---

→ Use of algorithmically hard problems.

Factorization:

- $p, q \mapsto n = p \cdot q$     easy (quadratic)
- $n = p \cdot q \mapsto p, q$     difficult

# Integer Factoring and RSA

---

→ Use of algorithmically hard problems.

Factorization:

- $p, q \mapsto n = p \cdot q$  easy (quadratic)
- $n = p \cdot q \mapsto p, q$  difficult

RSA function  $n = pq$ ,  $p$  and  $q$  primes.

$e$ : public exponent

- $x \mapsto x^e \pmod n$  easy (cubic)
- $y = x^e \mapsto x \pmod n$  difficult  
 $x = y^d$  where  $d = e^{-1} \pmod{\phi(n)}$

# Complexity Estimates

---

Estimates for integer factoring **Lenstra-Verheul 2000**

Modulus (bits)	Operations ( $\log_2$ )
512	58
1024	80
2048	111
4096	149
8192	156

$\approx 2^{60}$  years

→ Can be used for RSA too.

# Algorithmically hard problems

---

- **RSA**: given  $g^a$  and  $a$ , figure out  $g$ .
- **Discrete logarithm (DL)**: given  $g^a$  and  $g$ , figure out  $a$ .
- **Computational Diffie-Hellman (CDH)**:  
given  $g$ ,  $g^a$ , and  $g^b$ , figure out  $g^{ab}$ .
- **Decisional Diffie-Hellman (DDH)**:  
given  $g$ ,  $g^a$ ,  $g^b$ , and  $g^c$ , is  $c = ab \pmod{|G|}$ ?

$$DDH < CDH < DL$$

# Security proof

---

## Proof by reduction

1. **Assumption:** the algorithmic problem  $P$  is hard = no polynomial algorithm ( $P = RSA, DL, DDH, CDH...$ )
2. **Reduction:**
  - Let  $A$  be a (polynomial) adversary that breaks the encryption scheme
  - Then  $A$  can be used to solve  $P$  in polynomial time
3. **Security result:** no polynomial adversary

# Summary

---

## I. Introduction to computational models

1. Encryption
2. Cryptographic assumptions
3. Security notions

# One-Wayness (OW)

---

**Adversary  $\mathcal{A}$ :** any probabilistic polynomial time Turing Machine (PPTM)

**Basic security notion: One-Wayness (OW)**

Without the private key, it is computationally impossible to recover the plaintext:

$$\Pr_{m,r}[\mathcal{A}(c) = m \mid c = E(m; r)]$$

is **negligible**.

**Negligibility:**  $f$  is **negligible** if for all polynomial  $p$ , there exists  $\eta_0$  s.t. for all  $\eta \geq \eta_0$

$$f(\eta) \leq 1/p(\eta)$$

# Not enough

---

- Does not exclude to recover half of the plaintext
- Even worse if one has already partial information of the message:
  - Subject: XXXX
  - My answer is: XXXX

# Not enough

---

- Does not exclude to recover half of the plaintext
- Even worse if one has already partial information of the message:
  - Subject: XXXX
  - My answer is: XXXX

→ Introduction of a notion of indistinguishability:

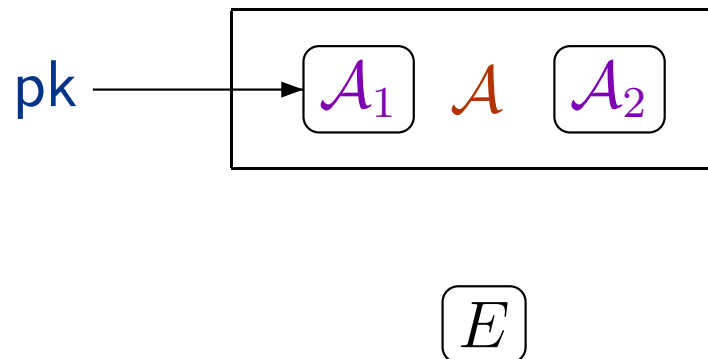
The adversary is not able to guess even a bit of the plaintext

# Indistinguishability (IND)

---

Game Adversary:  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

1. The adversary  $\mathcal{A}_1$  is given the public key  $pk$ .

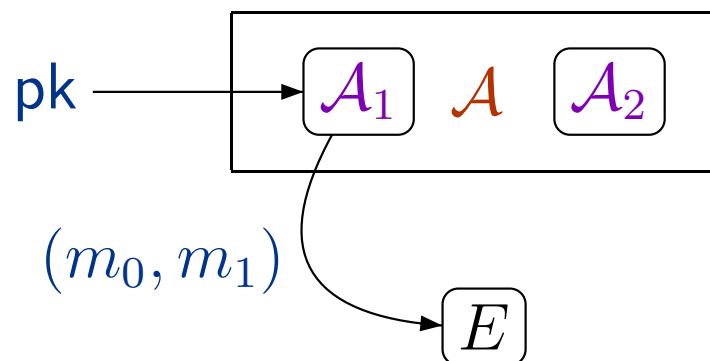


# Indistinguishability (IND)

---

Game Adversary:  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

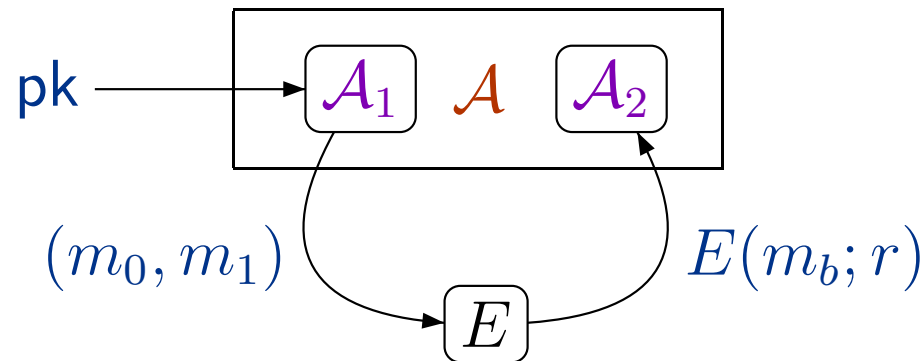
1. The adversary  $\mathcal{A}_1$  is given the public key  $pk$ .
2. The adversary  $\mathcal{A}_1$  chooses two messages  $m_0, m_1$ .



# Indistinguishability (IND)

Game Adversary:  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

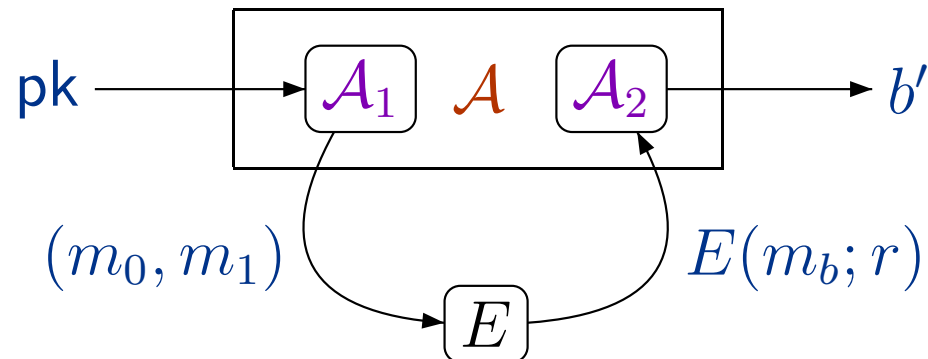
1. The adversary  $\mathcal{A}_1$  is given the public key  $pk$ .
2. The adversary  $\mathcal{A}_1$  chooses two messages  $m_0, m_1$ .
3.  $b = 0, 1$  is chosen at random and  $c = E(m_b; r)$  is given to the adversary.



# Indistinguishability (IND)

Game Adversary:  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

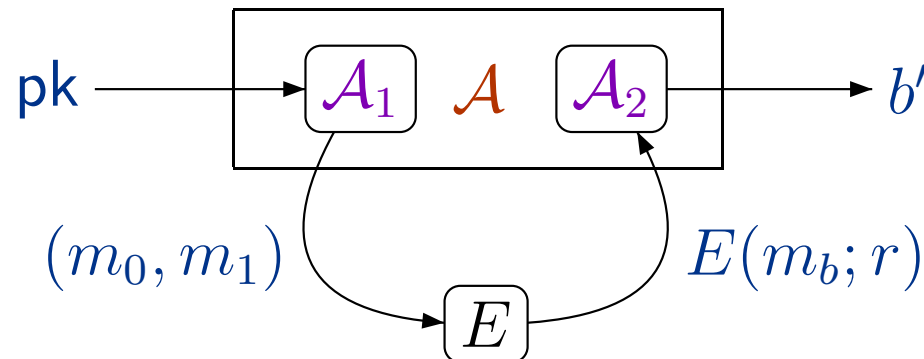
1. The adversary  $\mathcal{A}_1$  is given the public key  $pk$ .
2. The adversary  $\mathcal{A}_1$  chooses two messages  $m_0, m_1$ .
3.  $b = 0, 1$  is chosen at random and  $c = E(m_b; r)$  is given to the adversary.
4. The adversary  $\mathcal{A}_2$  answers  $b'$ .



# Indistinguishability (IND)

Game Adversary:  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

1. The adversary  $\mathcal{A}_1$  is given the public key  $pk$ .
2. The adversary  $\mathcal{A}_1$  chooses two messages  $m_0, m_1$ .
3.  $b = 0, 1$  is chosen at random and  $c = E(m_b; r)$  is given to the adversary.
4. The adversary  $\mathcal{A}_2$  answers  $b'$ .



The probability  $\Pr[b = b'] - \frac{1}{2}$  should be **negligible**.

# Non Malleability (NM)

---

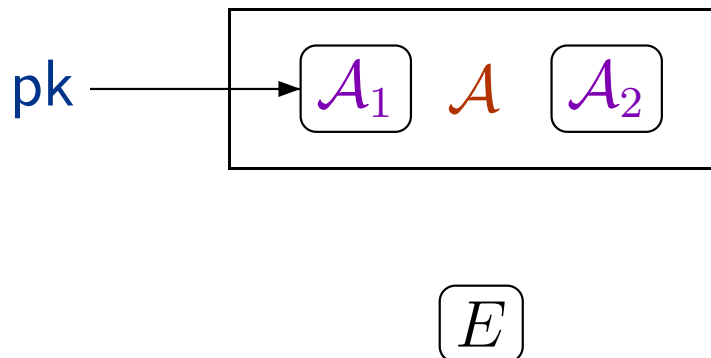
The adversary should not be able to produce a new ciphertext such that the plaintexts are meaningfully related.

# Non Malleability (NM)

---

Game Adversary:  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

1. The adversary  $\mathcal{A}_1$  is given the public key  $pk$ .

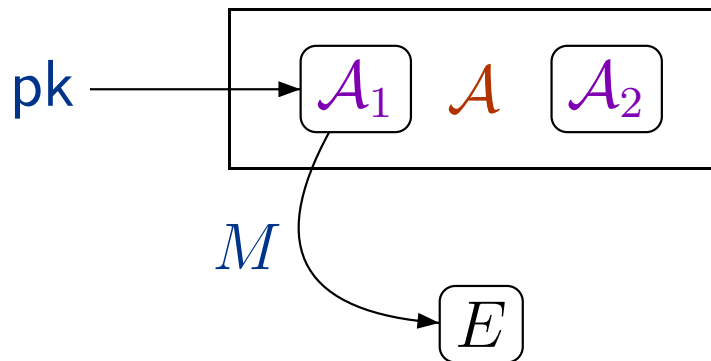


# Non Malleability (NM)

---

Game Adversary:  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

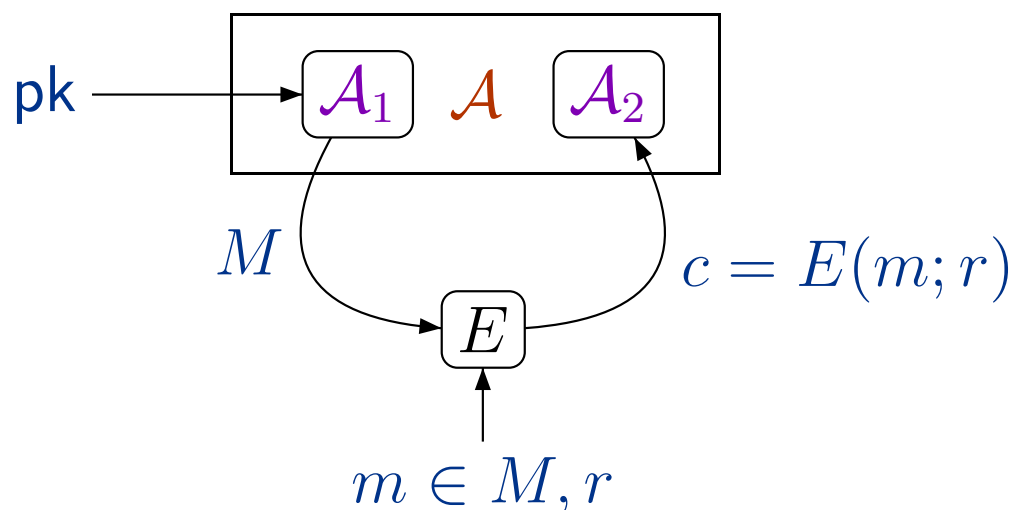
1. The adversary  $\mathcal{A}_1$  is given the public key  $pk$ .
2. The adversary  $\mathcal{A}_1$  chooses a message space  $M$ .



# Non Malleability (NM)

Game Adversary:  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

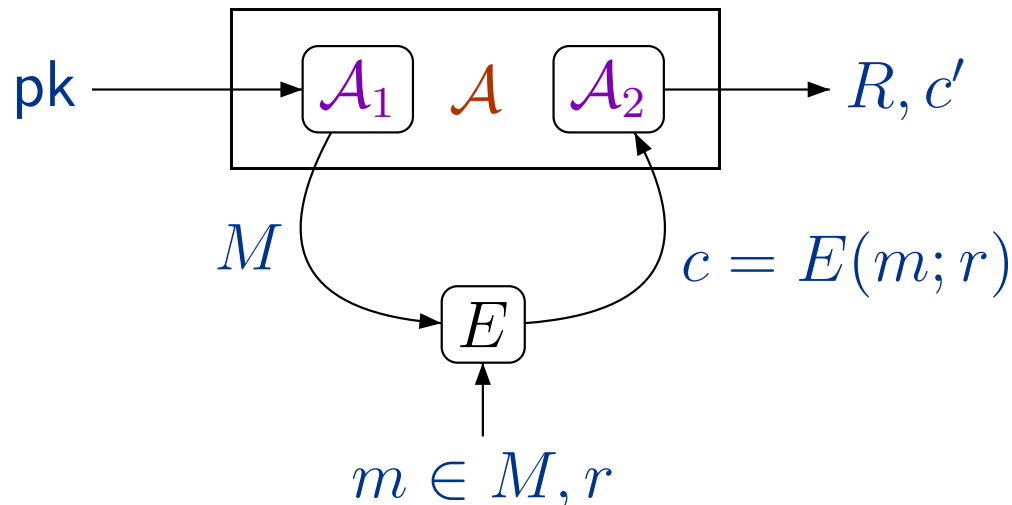
1. The adversary  $\mathcal{A}_1$  is given the public key  $pk$ .
2. The adversary  $\mathcal{A}_1$  chooses a message space  $M$ .
3. Two messages  $m$  and  $m^*$  are chosen at random in  $M$  and  $c = E(m; r)$  is given to the adversary.



# Non Malleability (NM)

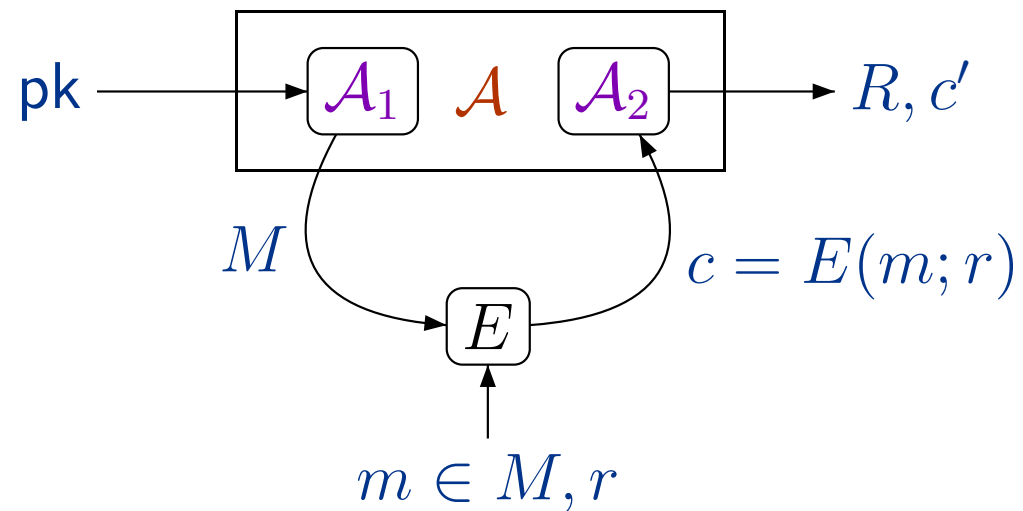
Game Adversary:  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

1. The adversary  $\mathcal{A}_1$  is given the public key  $pk$ .
2. The adversary  $\mathcal{A}_1$  chooses a message space  $M$ .
3. Two messages  $m$  and  $m^*$  are chosen at random in  $M$  and  $c = E(m; r)$  is given to the adversary.
4. The adversary  $\mathcal{A}_2$  outputs a binary relation  $R$  and a ciphertext  $c'$ .

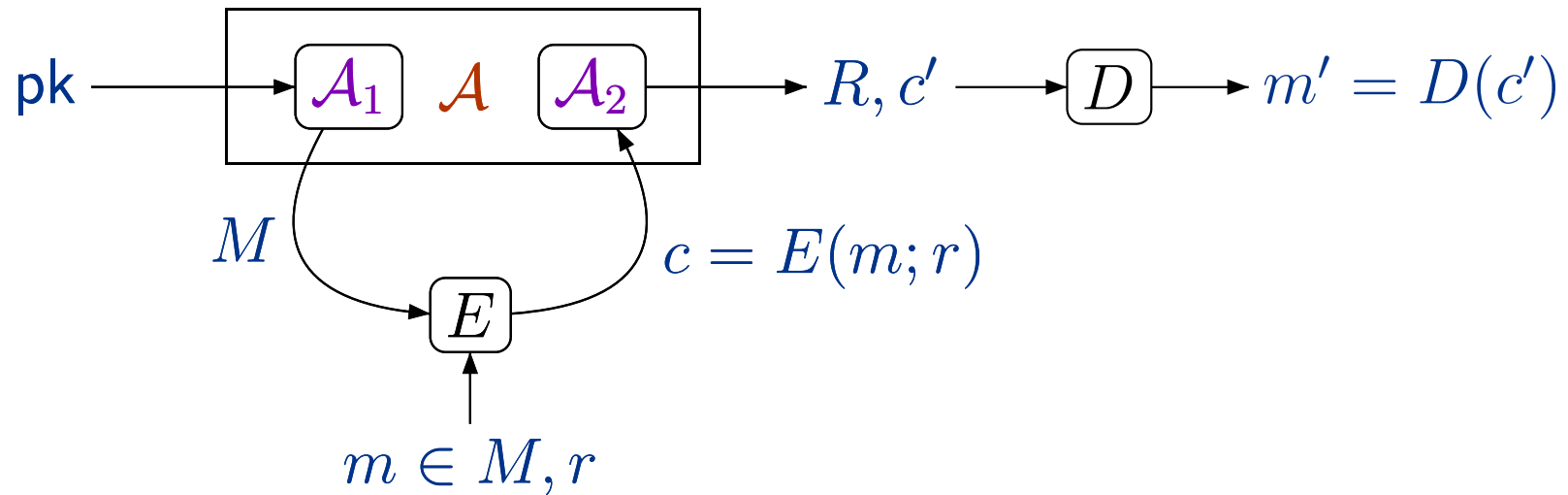


# Non Malleability (NM)

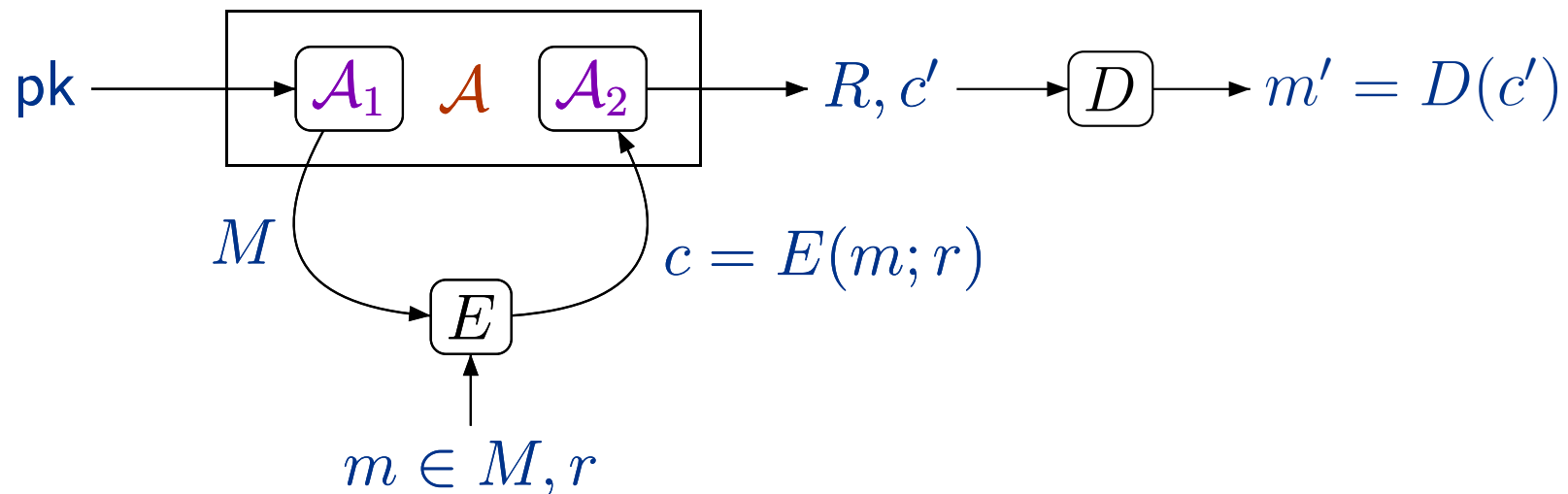
---



# Non Malleability (NM)



# Non Malleability (NM)



The probability  $\Pr[R(m, m')]$  –  $\Pr[R(m, m^*)]$  should be **negligible**.

# Relations

---

Non Malleability



Indistinguishability



One-Wayness

# Adding more security

---

The adversary is given access to **oracles** :

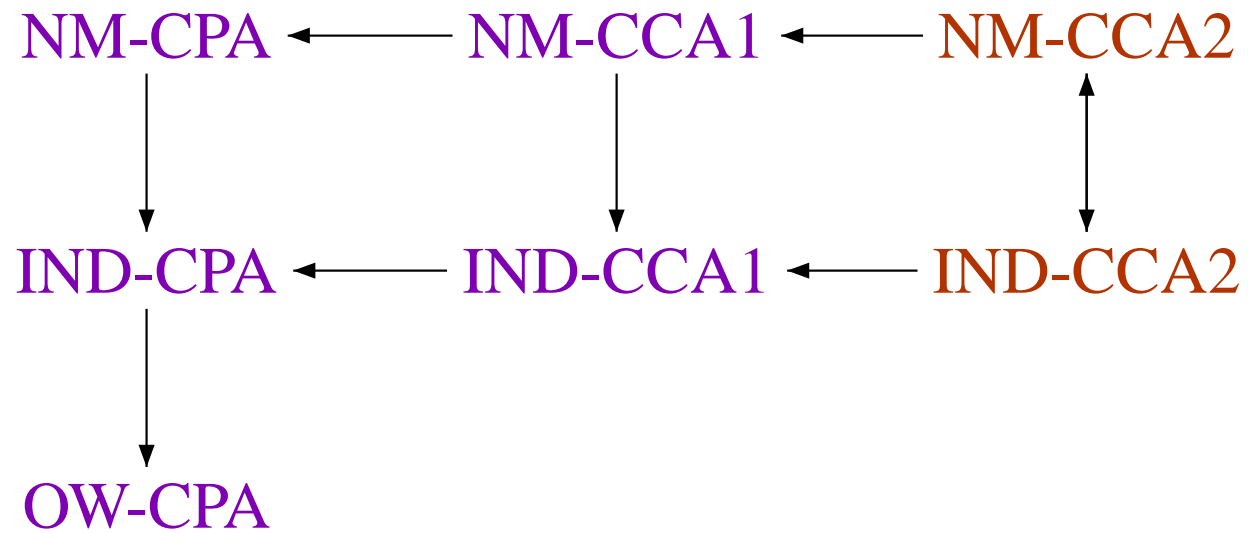
- **encryption** of all messages of his choice
- **decryption** of all messages of his choice

Three classical security levels:

- Chosen-Plaintext Attacks (**CPA**)
- Non adaptive Chosen-Ciphertext Attacks (**CCA1**)  
only before the challenge
- Adaptive Chosen-Ciphertext Attacks (**CCA2**)  
unlimited access to the oracle (except for the challenge)

# Relations

---



# Example: RSA

---

public	private
$n = pq$	$d = e^{-1} \pmod{\phi(n)}$
$e$ (public key)	(private key)

## RSA Encryption

- $E(m) = m^e \pmod n$
- $D(c) = c^d \pmod n$

OW-CPA = RSA problem      by definition!

# Summary

---

## II. Comparing computational and formal models

1. Messages
2. Adversary
3. Link?

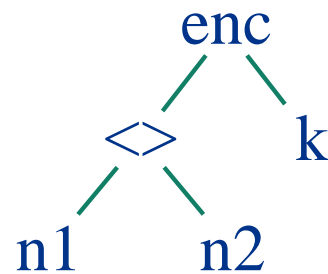
# Messages

---

Computational models Messages are bit-strings

0111000101010110100100

Formal models: Messages are modeled by terms.



# No collisions

---

Messages are modeled by terms.

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

# No collisions

---

Messages are modeled by terms.

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

→ No collisions:

- $n \neq n'$
- $\{m\}_k \neq \{m'\}_{k'}$  except if  $m = m'$  and  $k = k'$ ;
- $\{\{m\}_k\}_k \neq m$ ;
- $\langle m, m' \rangle \neq \{m\}_k$ ;
- ...

# Adversary

---

## Computational models

Any probabilistic polynomial-time Turing machine

# Adversary

---

## Computational models

Any probabilistic polynomial-time Turing machine

**Formal models:** The intruder can perform only specific actions:

- encryption,
- pairing,
- projection,
- decryption if he has the inverse key,
- ...

# Adversary

---

## Computational models

Any probabilistic polynomial-time Turing machine

**Formal models:** The intruder can perform only specific actions:

- encryption,
- pairing,
- projection,
- decryption if he has the inverse key,
- ...

## Consequence: Perfect encryption assumption

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

# Formal and Concrete approaches: Summary

	Formal approach	Concrete approach
Messages	terms	bitstrings
Encryption	idealized	algorithm
Adversary	idealized	any polynomial algorithm
Proof	<ul style="list-style-type: none"><li>- constraint solving</li><li>- first-order-logic</li><li>- tree automata, ...</li></ul>	reduction to cryptographic assumptions

# Formal and Concrete approaches: Summary

	Formal approach	Concrete approach
Messages	terms	bitstrings
Encryption	idealized	algorithm
Adversary	idealized	any polynomial algorithm
Proof	<ul style="list-style-type: none"><li>- constraint solving</li><li>- first-order-logic</li><li>- tree automata, ...</li></ul> <p>automatic</p>	<p>reduction to cryptographic assumptions</p> <p>by hand, tedious and error-prone</p>

Link between the two approaches ?

# Interest of relating the two approaches

---

- Better understanding of

# Interest of relating the two approaches

---

- Better understanding of
  - abstraction of messages by terms

# Interest of relating the two approaches

---

- Better understanding of
  - abstraction of messages by terms
  - formal adversary

# Interest of relating the two approaches

---

- Better understanding of
  - abstraction of messages by terms
  - formal adversary
  - cryptographic assumptions

# Interest of relating the two approaches

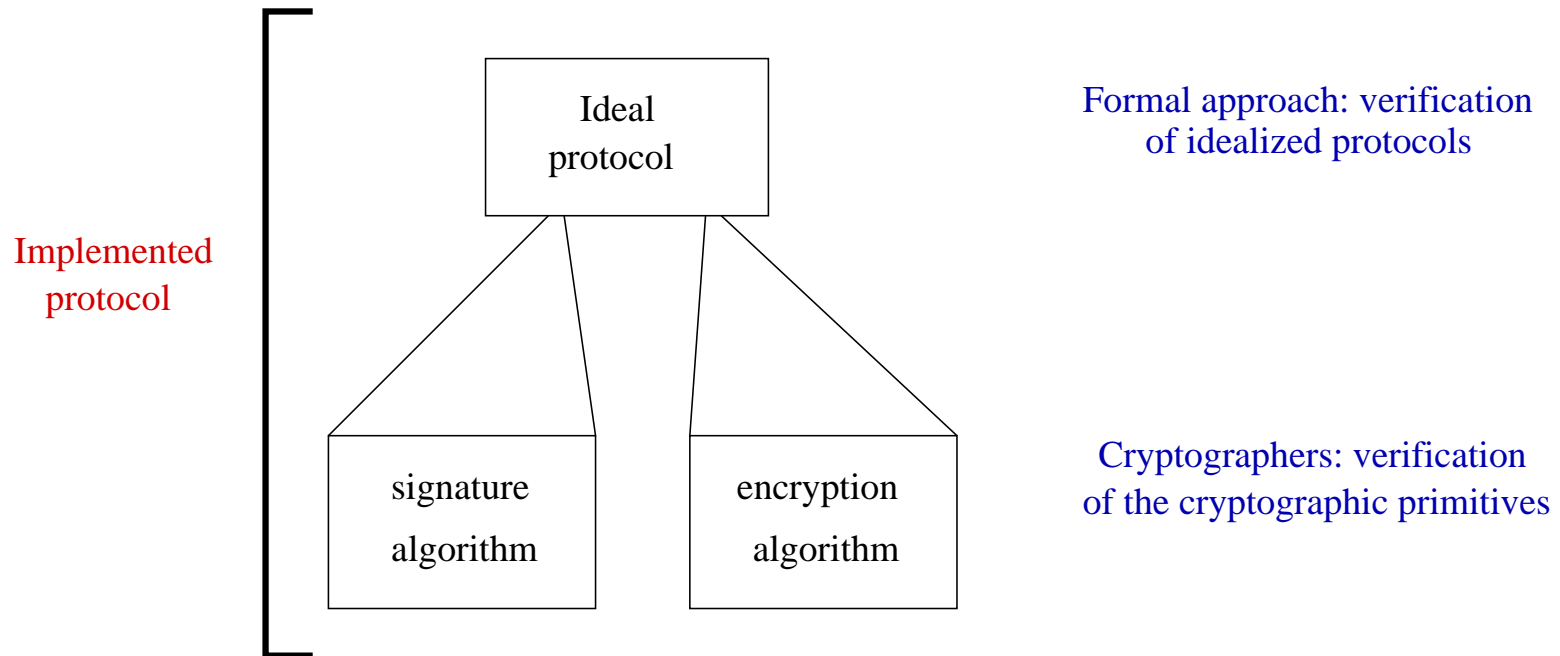
---

- Better understanding of
  - abstraction of messages by terms
  - formal adversary
  - cryptographic assumptions
- Automatic proof of computational security properties !

# Goal: soundness of the formal model

---

## Composition of two approaches



# Summary

---

## III. Soundness of formal models w.r.t. computational models

1. Introduction
2. Models
3. Main result
4. Extension
5. Related work

# A first soundness result

---

Theorem : [Micciancio-Warinschi TCC'04]

- for protocols where messages are formed from party identities, random nonces using asymmetric encryption and pairing,
- if the public key encryption algorithm is IND-CCA2,

# A first soundness result

---

Theorem : [Micciancio-Warinschi TCC'04]

- for protocols where messages are formed from party identities, random nonces using asymmetric encryption and pairing,
- if the public key encryption algorithm is IND-CCA2,

then, for trace properties, if the protocol is deemed secure using the formal approach, then the protocol is secure w.r.t. the computational approach.

# A first soundness result

---

Theorem : [Micciancio-Warinschi TCC'04]

- for protocols where messages are formed from party identities, random nonces using asymmetric encryption and pairing,
- if the public key encryption algorithm is IND-CCA2,

then, for trace properties, if the protocol is deemed secure using the formal approach, then the protocol is secure w.r.t. the computational approach.

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

# Extension to signatures and secrecy properties

---

For protocols with

- ciphertext forwarding,
- asymmetric encryption, provided the public key encryption algorithm is IND-CCA2,
- signatures, provided the signature algorithm is existentially unforgeable

We obtain:

- Soundness for the case of trace properties (like authentication)  
→ extension of [MW04]

# Extension to signatures and secrecy properties

---

For protocols with

- ciphertext forwarding,
- asymmetric encryption, provided the public key encryption algorithm is IND-CCA2,
- signatures, provided the signature algorithm is existentially unforgeable

We obtain:

- Soundness for the case of trace properties (like authentication)  
→ extension of [MW04]
- Soundness for the case of secrecy properties  
→ first result of soundness of secrecy

# Extension to signatures and secrecy properties

---

For protocols with

- ciphertext forwarding,
- asymmetric encryption, provided the public key encryption algorithm is IND-CCA2,
- signatures, provided the signature algorithm is existentially unforgeable

We obtain:

- Soundness for the case of trace properties (like authentication)  
→ extension of [MW04]
- Soundness for the case of secrecy properties  
→ first result of soundness of secrecy

⇒ Automatic proof of computational properties.

# Summary

---

## III. Soundness of formal models w.r.t. computational models

1. Introduction
2. Models
3. Main result
4. Extension
5. Related work

# Formal Model

---

- agents  $A_i, a_i$
- nonces (random numbers)  $X_{A_i}^j, n(a_i, j, s)$
- pairing  $\langle m_1, m_2 \rangle$
- asymmetric encryption  $\{m\}_{\text{ek}(a)}^l$

# Formal Model

---

- agents  $A_i, a_i$
- nonces (random numbers)  $X_{A_i}^j, n(a_i, j, s)$
- pairing  $\langle m_1, m_2 \rangle$
- asymmetric encryption  $\{m\}_{\text{ek}(a)}^l$

Rules of the form  $M_1 \rightarrow M_2$

A **protocol** is a finite set of **roles**.

$$\text{Roles} = (M_1, M_2)(M_3, M_4) \cdots (M_k, M_{k+1})$$

# Example 1

---

## Needham-Schroeder-Lowe protocol

$$A \rightarrow B : \quad \{N_a, A\}_{\text{ek}(B)}$$

$$B \rightarrow A : \quad \{N_a, N_b, B\}_{\text{ek}(A)}$$

$$A \rightarrow B : \quad \{N_b\}_{\text{ek}(B)}$$

# Example 1

## Needham-Schroeder-Lowe protocol

$$A \rightarrow B : \quad \{N_a, A\}_{\text{ek}(B)}$$

$$B \rightarrow A : \quad \{N_a, N_b, B\}_{\text{ek}(A)}$$

$$A \rightarrow B : \quad \{N_b\}_{\text{ek}(B)}$$

$$\Pi(1) = (\text{init} \rightarrow \{X_{A_1}^1, A_1\}_{\text{ek}(A_2)}^{\text{ag}(1)}),$$

$$(\{X_{A_1}^1, X_{A_2}^1, A_2\}_{\text{ek}(A_1)}^L \rightarrow \{X_{A_2}^1\}_{\text{ek}(A_2)}^{\text{ag}(1)})$$

$$\Pi(2) = (\{X_{A_1}^1, A_1\}_{\text{ek}(A_2)}^{L_1} \rightarrow \{X_{A_1}^1, X_{A_2}^1, A_2\}_{\text{ek}(A_1)}^{\text{ag}(1)}),$$

$$(\{X_{A_2}^1\}_{\text{ek}(A_2)}^{L_2} \rightarrow \text{stop})$$

Variables are local to each role and each session.

# Example 2: On the use of labels

---

$$\begin{aligned} A \rightarrow B : & \quad \{N_a\}_{\text{ek}(B)}, \{\{N_a\}_{\text{ek}(B)}\}_{\text{ek}(B)} \\ B \rightarrow A : & \quad \{N_a\}_{\text{ek}(B)} \end{aligned}$$

# Example 2: On the use of labels

---

$$\begin{aligned} A \rightarrow B &: \quad \{N_a\}_{\text{ek}(B)}, \{\{N_a\}_{\text{ek}(B)}\}_{\text{ek}(B)} \\ B \rightarrow A &: \quad \{N_a\}_{\text{ek}(B)} \end{aligned}$$

$$\Pi(1) = (\text{init} \rightarrow \langle \{X_{A_1}^1\}_{\text{ek}(A_2)}^{\text{ag}(1)}, \{\{X_{A_1}^1\}_{\text{ek}(A_2)}^{\text{ag}(1)}\}_{\text{ek}(A_2)} \rangle)$$

or

$$\Pi(1) = (\text{init} \rightarrow \langle \{X_{A_1}^1\}_{\text{ek}(A_2)}^{\text{ag}(1)}, \{\{X_{A_1}^1\}_{\text{ek}(A_2)}^{\text{ag}(2)}\}_{\text{ek}(A_2)} \rangle)$$

# Example 2: On the use of labels

$$\begin{aligned} A \rightarrow B : & \quad \{N_a\}_{\text{ek}(B)}, \{\{N_a\}_{\text{ek}(B)}\}_{\text{ek}(B)} \\ B \rightarrow A : & \quad \{N_a\}_{\text{ek}(B)} \end{aligned}$$

$$\Pi(1) = (\text{init} \rightarrow \langle \{X_{A_1}^1\}_{\text{ek}(A_2)}^{\text{ag}(1)}, \{\{X_{A_1}^1\}_{\text{ek}(A_2)}^{\text{ag}(1)}\}_{\text{ek}(A_2)} \rangle)$$

or

$$\Pi(1) = (\text{init} \rightarrow \langle \{X_{A_1}^1\}_{\text{ek}(A_2)}^{\text{ag}(1)}, \{\{X_{A_1}^1\}_{\text{ek}(A_2)}^{\text{ag}(2)}\}_{\text{ek}(A_2)} \rangle)$$

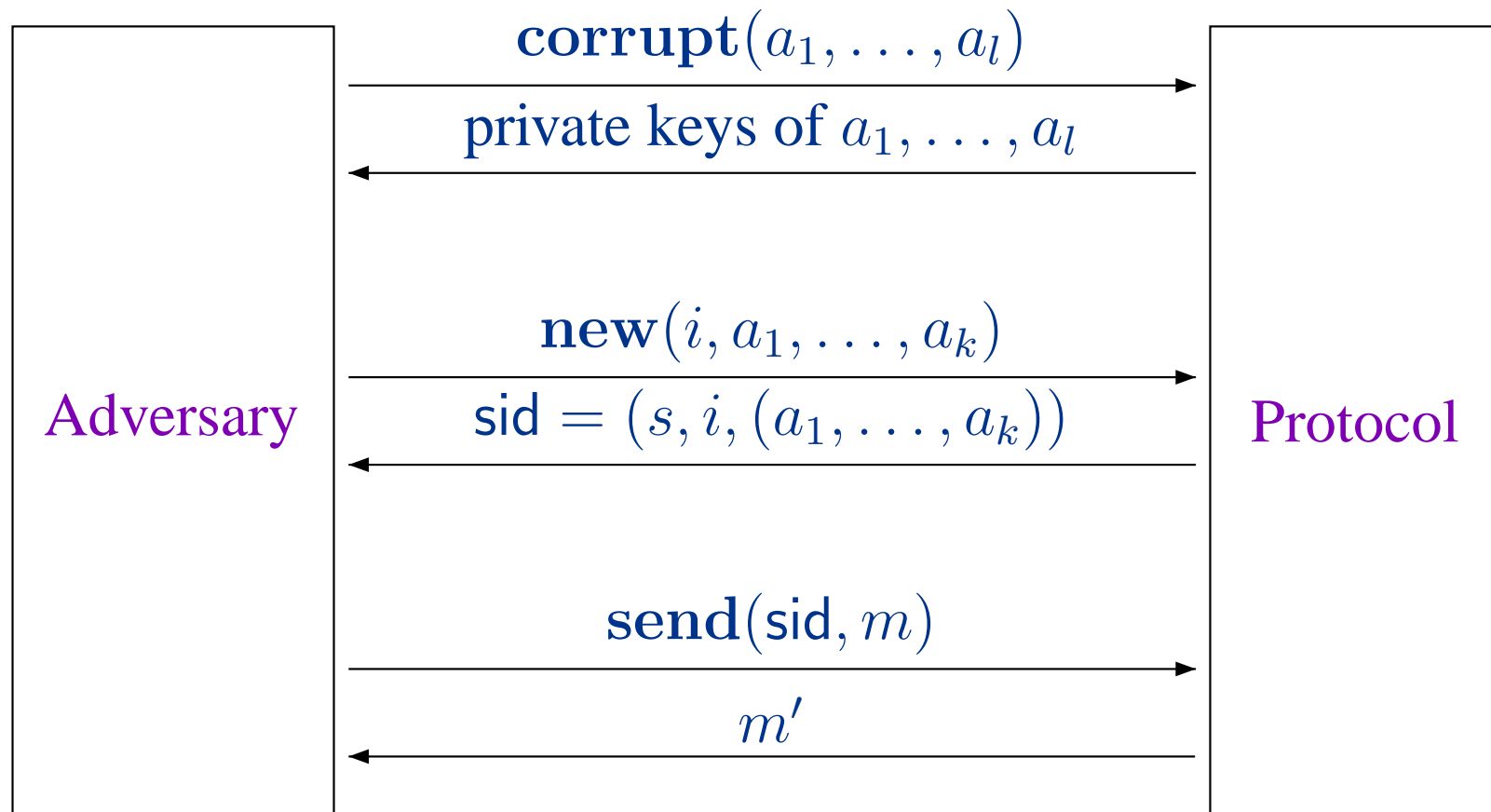
$$\Pi(2) = (\langle \{X_{A_1}^1\}_{\text{ek}(A_2)}^L, \{\{X_{A_1}^1\}_{\text{ek}(A_2)}^{L/L'}\}_{\text{ek}(A_2)} \rangle \rightarrow \{X_{A_1}^1\}_{\text{ek}(A_2)}^{L/L'})$$

or

$$\Pi(2) = (\langle \{X_{A_1}^1\}_{\text{ek}(A_2)}^L, \{\{X_{A_1}^1\}_{\text{ek}(A_2)}^{L/L'}\}_{\text{ek}(A_2)} \rangle \rightarrow \{X_{A_1}^1\}_{\text{ek}(A_2)}^{\text{ag}(1)})$$

# Network

---



# Formal Intruder Deduction Rules

---

$$\frac{S \vdash m_1 \quad S \vdash m_2}{S \vdash \langle m_1, m_2 \rangle}$$

$$\frac{S \vdash \langle m_1, m_2 \rangle}{S \vdash m_i} \quad i \in \{1, 2\}$$

# Formal Intruder Deduction Rules

---

$$\frac{S \vdash m_1 \quad S \vdash m_2}{S \vdash \langle m_1, m_2 \rangle}$$

$$\frac{S \vdash \langle m_1, m_2 \rangle}{S \vdash m_i} \quad i \in \{1, 2\}$$

$$\frac{S \vdash \text{ek}(b) \quad S \vdash m}{S \vdash \{m\}_{\text{ek}(b)}^{\text{adv}(i)}} \quad i \in \mathbb{N}$$

$$\frac{S \vdash \{m\}_{\text{ek}(b)}^l \quad S \vdash \text{dk}(b)}{S \vdash m}$$

# Hypotheses on the Implementation

---

- encryption : **IND-CCA2**  
→ 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

---

- encryption : **IND-CCA2**
  - the adversary cannot distinguish between  $\{n_0\}_k$  and  $\{n_1\}_k$  even if he has access to encryption and decryption oracles.
- parsing :
  - each bit-string has a label which indicates his type (identity, nonce, key, ...)
  - one can retrieve the (public) encryption key from an encrypted message.

# Formal semantics: global state

---

Global state  $(Sld, f, H)$

- Sld set of sessions ids of the form  
 $(n, j, (a_1, a_2, \dots, a_k)) \in (\mathbb{N} \times \mathbb{N} \times \text{ID}^k)$ ,
  - $n \in \mathbb{N}$  identifies the session,
  - $a_1, a_2, \dots, a_k$  identities involved in the protocol
  - $j$  index of the role

# Formal semantics: global state

---

Global state  $(Sld, f, H)$

- $Sld$  set of sessions ids of the form  $(n, j, (a_1, a_2, \dots, a_k)) \in (\mathbb{N} \times \mathbb{N} \times ID^k)$ ,
  - $n \in \mathbb{N}$  identifies the session,
  - $a_1, a_2, \dots, a_k$  identities involved in the protocol
  - $j$  index of the role
- $H$  set of terms, the messages sent on the network

# Formal semantics: global state

---

## Global state $(Sld, f, H)$

- $Sld$  set of sessions ids of the form  $(n, j, (a_1, a_2, \dots, a_k)) \in (\mathbb{N} \times \mathbb{N} \times ID^k)$ ,
  - $n \in \mathbb{N}$  identifies the session,
  - $a_1, a_2, \dots, a_k$  identities involved in the protocol
  - $j$  index of the role
- $H$  set of terms, the messages sent on the network
- $f(sid) = (\sigma, i, p)$  is the local state of  $sid$ .
  - $\sigma$  partial instantiation of the variables of the role  $\Pi(i)$
  - $p \in \mathbb{N}$  is the control point of the program.

# Formal semantics: new query

---

$$(Sld, f, H) \xrightarrow{\text{new}(i, a_1, \dots, a_k)} (Sld', f', H')$$

- $s = |Sld| + 1$  new session id
- $H' = H \cup \{(s, i, (a_1, \dots, a_k))\}$
- $Sld' = Sld \cup \{(s, i, (a_1, \dots, a_k))\}$

# Formal semantics: new query

---

$$(Sld, f, H) \xrightarrow{\text{new}(i, a_1, \dots, a_k)} (Sld', f', H')$$

- $s = |Sld| + 1$  new session id
- $H' = H \cup \{(s, i, (a_1, \dots, a_k))\}$
- $Sld' = Sld \cup \{(s, i, (a_1, \dots, a_k))\}$
- $f'(s, i, (a_1, \dots, a_k)) = (\sigma, i, 1)$  where

$$\begin{cases} \sigma(A_j) & = a_j & 1 \leq j \leq k \\ \sigma(X_{A_i}^j) & = n(a_i, j, s) & j \in \mathbb{N} \end{cases}$$

# Formal semantics: send query

---

$$(Sld, f, H) \xrightarrow{\text{send}(\text{sid}, m)} (Sld, f', H') \quad \text{with} \quad f(\text{sid}) = (\sigma, j, p)$$

Let  $\Pi(j) = ((l_1^j, r_1^j), \dots, (l_{k_j}^j, r_{k_j}^j))$ .

- Either  $m = l_p^j \sigma \theta$ . Then  $f'(\text{sid}) = (\sigma \cup \theta, i, p + 1)$  and  $H' = H \cup \{r_p^j \sigma \theta\}$ .
- Or the state remains unchanged.

# Formal semantics: send query

---

$$(Sld, f, H) \xrightarrow{\text{send}(\text{sid}, m)} (Sld, f', H') \quad \text{with} \quad f(\text{sid}) = (\sigma, j, p)$$

Let  $\Pi(j) = ((l_1^j, r_1^j), \dots, (l_{k_j}^j, r_{k_j}^j))$ .

- Either  $m = l_p^j \sigma \theta$ . Then  $f'(\text{sid}) = (\sigma \cup \theta, i, p + 1)$  and  $H' = H \cup \{r_p^j \sigma \theta\}$ .
- Or the state remains unchanged.

The trace is **valid** if  $H \vdash m$

# Concrete semantics

---

## Global state $(Sld, f)$

- $Sld$  set of sessions ids of the form  $(n, j, (a_1, a_2, \dots, a_k)) \in (\mathbb{N} \times \mathbb{N} \times ID^k)$ ,
  - $n \in \mathbb{N}$  identifies the session,
  - $a_1, a_2, \dots, a_k$  identities involved in the protocol
  - $j$  index of the role
- $f(sid) = (\sigma, i, p)$  is the local state of  $sid$ .
  - $\sigma$  partial instantiation of the variables of the role  $\Pi(i)$
  - $p \in \mathbb{N}$  is the control point of the program.

# Concrete semantics

---

## Global state $(SId, f)$

- $SId$  set of sessions ids of the form  $(n, j, (a_1, a_2, \dots, a_k)) \in (\mathbb{N} \times \mathbb{N} \times ID^k)$ ,
  - $n \in \mathbb{N}$  identifies the session,
  - $a_1, a_2, \dots, a_k$  identities involved in the protocol
  - $j$  index of the role
- $f(sid) = (\sigma, i, p)$  is the local state of  $sid$ .
  - $\sigma$  partial instantiation of the variables of the role  $\Pi(i)$
  - $p \in \mathbb{N}$  is the control point of the program.

$\left. \begin{array}{l} \text{new}(i, a_1, \dots, a_k) \\ \text{send}(sid, m) \end{array} \right\}$  similar to the formal semantics

# Trace properties

---

**State:** assignments of the local variables

**Trace:** sequence of states

**SymbTr:** Set of symbolic traces

**ConcTr:** Set of concrete traces

# Trace properties

---

**State:** assignments of the local variables

**Trace:** sequence of states

**SymbTr:** Set of symbolic traces

**ConcTr:** Set of concrete traces

## Trace properties

- A **formal trace property** is any subset  $P^s \subseteq \text{SymbTr}$ .
- A **concrete trace property** is any subset  $P^c \subseteq \text{ConcTr}$ .

# Trace properties

---

**State:** assignments of the local variables

**Trace:** sequence of states

**SymbTr:** Set of symbolic traces

**ConcTr:** Set of concrete traces

## Trace properties

- A **formal trace property** is any subset  $P^s \subseteq \text{SymbTr}$ .
- A **concrete trace property** is any subset  $P^c \subseteq \text{ConcTr}$ .

## Satisfaction

- A protocol  $\Pi$  **satisfies symbolically**  $P^s$ , denoted by  $\Pi \models^s P^s$ , if any valid formal trace  $t^s \in P^s$ .
- A protocol  $\Pi$  **satisfies concretely**  $P^c$ , denoted by  $\Pi \models^c P^c$ , if any valid concrete trace  $t^c \in P^c$ , **with overwhelming probability**.

# Summary

---

## III. Soundness of formal models w.r.t. computational models

1. Introduction
2. Models
3. **Main result**
4. Extension
5. Related work

# Soundness of trace properties

---

Theorem 1 :

Every concrete trace is the image of a valid formal trace, except with negligible probability.

$$\Pr \left[ \exists t^s \in \text{Exec}^s(\Pi) \mid t^s \preceq \text{Exec}_\eta^c(\Pi) \right] \geq 1 - \nu_{\mathcal{A}}(\eta)$$

where  $\nu_{\mathcal{A}}(\cdot)$  is a negligible function.

# Soundness of trace properties

---

Theorem 1 :

Every concrete trace is the image of a valid formal trace, except with negligible probability.

$$\Pr \left[ \exists t^s \in \text{Exec}^s(\Pi) \mid t^s \preceq \text{Exec}_\eta^c(\Pi) \right] \geq 1 - \nu_{\mathcal{A}}(\eta)$$

where  $\nu_{\mathcal{A}}(\cdot)$  is a negligible function.

Corollary :

Let  $\Pi$  be protocol,  $P^s \subseteq \text{SymbTr}$  an arbitrary predicate on formal traces and  $P^c \subseteq \text{ConcTr}$  an arbitrary predicate on concrete traces such that  $c(P^s) \subseteq P^c$ .

Then  $\Pi \models^s P^s$  implies  $\Pi \models^c P^c$ .

# Soundness of trace properties

---

Theorem 1 :

Every concrete trace is the image of a valid formal trace, except with negligible probability.

$$\Pr \left[ \exists t^s \in \text{Exec}^s(\Pi) \mid t^s \preceq \text{Exec}_\eta^c(\Pi) \right] \geq 1 - \nu_{\mathcal{A}}(\eta)$$

where  $\nu_{\mathcal{A}}(\cdot)$  is a negligible function.

Corollary :

Let  $\Pi$  be protocol,  $P^s \subseteq \text{SymbTr}$  an arbitrary predicate on formal traces and  $P^c \subseteq \text{ConcTr}$  an arbitrary predicate on concrete traces such that  $c(P^s) \subseteq P^c$ .

Then  $\Pi \models^s P^s$  implies  $\Pi \models^c P^c$ .

Applications : authentication, secrecy, ...

# Proof idea

---

**Key result:** every concrete trace is the image of a valid formal trace, except with negligible probability.

$$\begin{array}{ccccc} \mathit{init}(1, a, b) & \longrightarrow & \{a, n_a\}_{K_b} & & \{n_a\}_{K_b} \text{ non valid!} \\ & & \uparrow & & \downarrow \\ \mathcal{A}: \mathit{init}(1, a, b) & & m_1 & \longrightarrow & \mathit{send}(m_2) \end{array}$$

# Proof idea

---

**Key result:** every concrete trace is the image of a valid formal trace, except with negligible probability.

$$\begin{array}{ccccc} \text{init}(1, a, b) & \rightarrow & \{a, n_a\}_{K_b} & & \{n_a\}_{K_b} \text{ non valid!} \\ & & \downarrow & & \uparrow \\ \mathcal{A} : \text{init}(1, a, b) & & m_1 & \rightarrow & \text{send}(m_2) \\ & & \uparrow & & \end{array}$$

Using the adversary  $\mathcal{A}$ , we build an adversary  $\mathcal{A}'$  that breaks encryption.

$$\mathcal{A}' : (\langle a, n_a^0 \rangle, \langle a, n_a^1 \rangle) \rightarrow \text{encryption oracle} \rightarrow \{a, n_a^\alpha\}_{K_b}$$

# Proof idea

---

**Key result:** every concrete trace is the image of a valid formal trace, except with negligible probability.

$$\begin{array}{ccccc} \text{init}(1, a, b) & \rightarrow & \{a, n_a\}_{K_b} & & \{n_a\}_{K_b} \text{ non valid!} \\ & & \downarrow & & \uparrow \\ \mathcal{A} : \text{init}(1, a, b) & & m_1 & \rightarrow & \text{send}(m_2) \\ & & \uparrow & & \downarrow \end{array}$$

Using the adversary  $\mathcal{A}$ , we build an adversary  $\mathcal{A}'$  that breaks encryption.

$$\begin{array}{l} \mathcal{A}' : (\langle a, n_a^0 \rangle, \langle a, n_a^1 \rangle) \rightarrow \text{encryption oracle} \rightarrow \{a, n_a^\alpha\}_{K_b} \\ \rightarrow \mathcal{A} \rightarrow \{n_a^\alpha\}_{K_b} \end{array}$$

# Proof idea

**Key result:** every concrete trace is the image of a valid formal trace, except with negligible probability.

$$\begin{array}{ccccc} \text{init}(1, a, b) & \rightarrow & \{a, n_a\}_{K_b} & & \{n_a\}_{K_b} \text{ non valid!} \\ & & \downarrow & & \uparrow \\ \mathcal{A} : \text{init}(1, a, b) & & m_1 & \rightarrow & \text{send}(m_2) \\ & \uparrow & & & \end{array}$$

Using the adversary  $\mathcal{A}$ , we build an adversary  $\mathcal{A}'$  that breaks encryption.

$$\begin{array}{ccccccc} \mathcal{A}' : & (\langle a, n_a^0 \rangle, \langle a, n_a^1 \rangle) & \rightarrow & \text{encryption oracle} & \rightarrow & \{a, n_a^\alpha\}_{K_b} \\ & & & & & \\ & & \rightarrow & \mathcal{A} & \rightarrow & \{n_a^\alpha\}_{K_b} & \rightarrow & \text{decryption oracle} & \rightarrow & n_a^\alpha & \rightarrow & \alpha \end{array}$$

# General proof

---

Assume there exists an adversary  $\mathcal{A}$  such that

$\Pr [ \exists t^s \in \text{Exec}^s(\Pi) \mid t^s \preceq \text{Exec}_\eta^c(\Pi) ]$  is not negligible.

Then we construct an adversary  $\mathcal{B}$  that breaks the encryption scheme, using  $\mathcal{A}$  as a sub-routine.

# General proof

---

Assume there exists an adversary  $\mathcal{A}$  such that

$\Pr [ \exists t^s \in \text{Exec}^s(\Pi) \mid t^s \preceq \text{Exec}_\eta^c(\Pi) ]$  is not negligible.

Then we construct an adversary  $\mathcal{B}$  that breaks the encryption scheme, using  $\mathcal{A}$  as a sub-routine.

## Steps

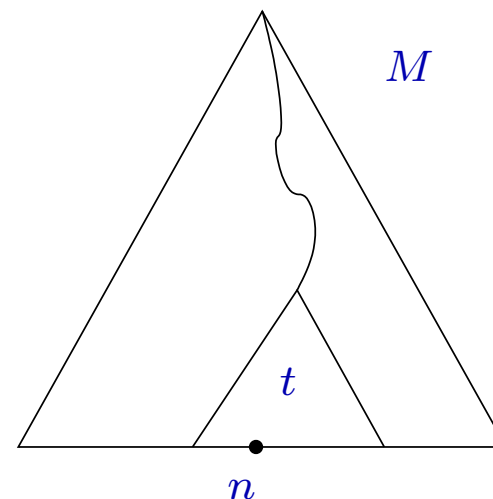
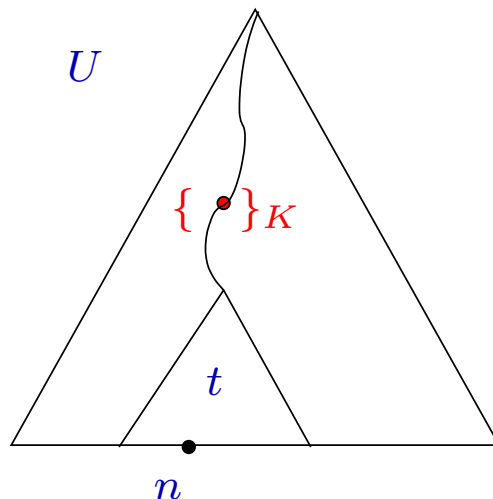
1. Characterization of non deducible terms
2. Simulation of the protocol by  $\mathcal{B}$
3. Attack of the encryption scheme

# Characterization of non deducible terms

**Lemma :** Let  $S$  be a set of messages,  $M$  be a message such that  $names(M) \subseteq S$ .

If  $S \not\vdash M$  then there exists  $t$  subterm of  $S$  and  $M$  such that:

- $S \not\vdash t$
- there exists  $U \in S$  s.t.  $U = C[\{C'[t]\}_{ek(a)}]$ ,  $S \not\vdash ek(a)$  and  $\{C'[t]\}_k$  is not a subterm of  $M$ .



# Simulation of the protocol: some initial guesses

---

- $\mathcal{B}$  guesses:
  - which nonce  $X_{A_i}^j$  will be attacked,
  - which session  $s$  will be attacked.

# Simulation of the protocol: some initial guesses

---

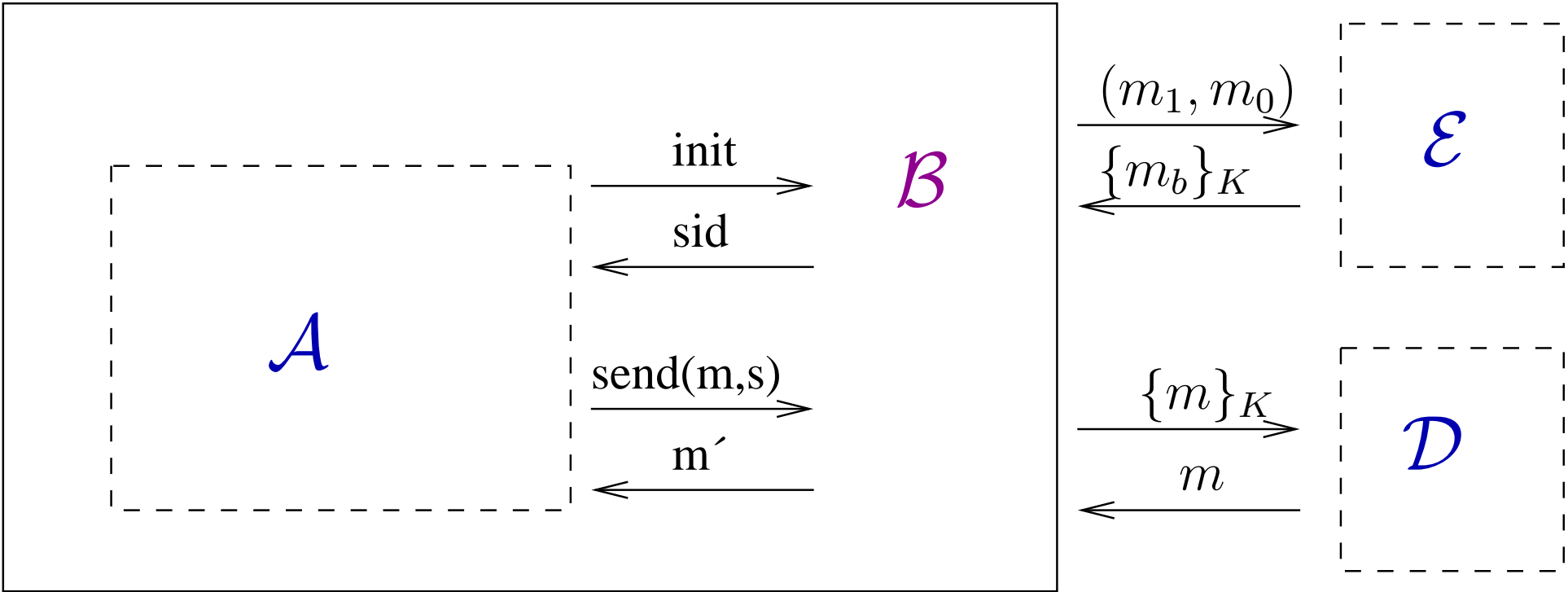
- $\mathcal{B}$  guesses:
  - which nonce  $X_{A_i}^j$  will be attacked,
  - which session  $s$  will be attacked.
- $\mathcal{B}$  generates two nonces  $n_0$  and  $n_1$  instead of  $n(a_i, j, s)$ ,

# Simulation of the protocol: some initial guesses

---

- $\mathcal{B}$  guesses:
  - which nonce  $X_{A_i}^j$  will be attacked,
  - which session  $s$  will be attacked.
- $\mathcal{B}$  generates two nonces  $n_0$  and  $n_1$  instead of  $n(a_i, j, s)$ ,
- $\mathcal{B}$  answers to  $\mathcal{A}$ 's queries using the oracles.

# Simulation of the protocol



# How to answer a valid $\text{send}(m, s)$ queries?

---

$\mathcal{A}$  sends  $m = c(M)$  valid ( $S \vdash M$ )     $M = l_i\theta$

- $M = \langle M_1, M_2 \rangle \rightarrow$  projection

# How to answer a valid $\text{send}(m, s)$ queries?

---

$\mathcal{A}$  sends  $m = c(M)$  valid ( $S \vdash M$ )     $M = l_i\theta$

- $M = \langle M_1, M_2 \rangle \rightarrow$  projection
- $M = \{M'\}_{\text{ek}(a)}^l$  with  $a$  corrupted  
 $\rightarrow \mathcal{B}$  decrypts  $M$  using  $\text{sk}(a)$

# How to answer a valid $\text{send}(m, s)$ queries?

---

$\mathcal{A}$  sends  $m = c(M)$  valid ( $S \vdash M$ )     $M = l_i\theta$

- $M = \langle M_1, M_2 \rangle \rightarrow$  projection
- $M = \{M'\}_{\text{ek}(a)}^l$  with  $a$  corrupted  
 $\rightarrow \mathcal{B}$  decrypts  $M$  using  $\text{sk}(a)$
- $M = \{M'\}_{\text{ek}(a)}^l$  with  $a$  honest

# How to answer a valid $\text{send}(m, s)$ queries?

---

$\mathcal{A}$  sends  $m = c(M)$  valid ( $S \vdash M$ )     $M = l_i\theta$

- $M = \langle M_1, M_2 \rangle \rightarrow$  projection
- $M = \{M'\}_{\text{ek}(a)}^l$  with  $a$  corrupted  
 $\rightarrow \mathcal{B}$  decrypts  $M$  using  $\text{sk}(a)$
- $M = \{M'\}_{\text{ek}(a)}^l$  with  $a$  honest
  - either  $\{M'\}_{\text{ek}(a)}^l$  has not been obtained using the encryption oracle, in which case  $\mathcal{B}$  retrieves  $M'$  using the decryption oracle.

# How to answer a valid $\text{send}(m, s)$ queries?

---

$\mathcal{A}$  sends  $m = c(M)$  valid ( $S \vdash M$ )     $M = l_i\theta$

- $M = \langle M_1, M_2 \rangle \rightarrow$  projection
- $M = \{M'\}_{\text{ek}(a)}^l$  with  $a$  corrupted  
 $\rightarrow \mathcal{B}$  decrypts  $M$  using  $\text{sk}(a)$
- $M = \{M'\}_{\text{ek}(a)}^l$  with  $a$  honest
  - either  $\{M'\}_{\text{ek}(a)}^l$  has not been obtained using the encryption oracle, in which case  $\mathcal{B}$  retrieves  $M'$  using the decryption oracle.
  - or  $\{M'\}_{\text{ek}(a)}^l$  has been obtained using the encryption oracle,  
 $\Rightarrow M' = M_b$  and  $M_0$  and  $M_1$  are known to  $\mathcal{B}$ .

# How to answer a valid $\text{send}(m, s)$ queries?

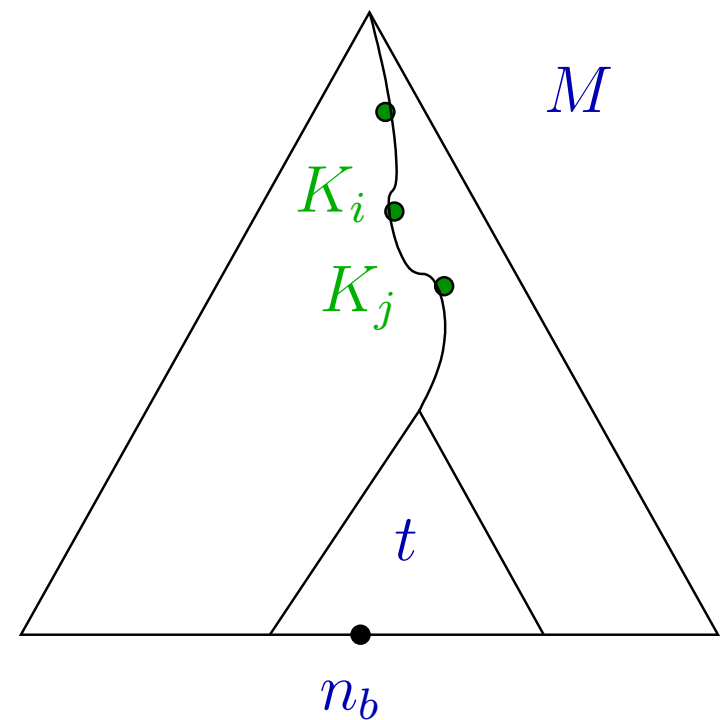
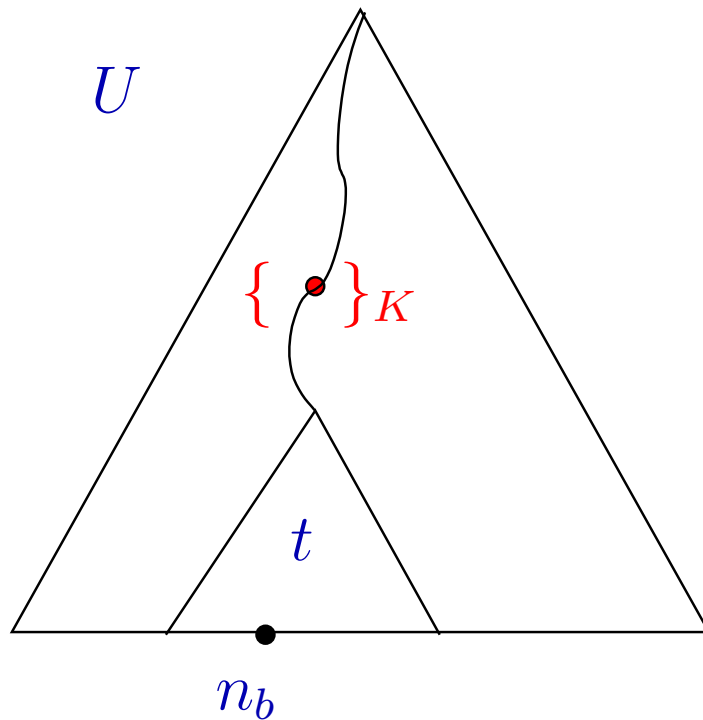
$\mathcal{A}$  sends  $m = c(M)$  valid ( $S \vdash M$ )  $M = l_i\theta$

- $M = \langle M_1, M_2 \rangle \rightarrow$  projection
- $M = \{M'\}_{\text{ek}(a)}^l$  with  $a$  corrupted  
 $\rightarrow \mathcal{B}$  decrypts  $M$  using  $\text{sk}(a)$
- $M = \{M'\}_{\text{ek}(a)}^l$  with  $a$  honest
  - either  $\{M'\}_{\text{ek}(a)}^l$  has not been obtained using the encryption oracle, in which case  $\mathcal{B}$  retrieves  $M'$  using the decryption oracle.
  - or  $\{M'\}_{\text{ek}(a)}^l$  has been obtained using the encryption oracle,  
 $\Rightarrow M' = M_b$  and  $M_0$  and  $M_1$  are known to  $\mathcal{B}$ .

Then  $\mathcal{B}$  answers  $c(r_i\theta)$  following the protocol rule, possibly using the left-right encryption oracles.

# How to break encryption?

$\mathcal{A}$  sends  $m = c(M)$  non valid ( $S \not\equiv M$ )



# Summary

---

## III. Soundness of formal models w.r.t. computational models

1. Introduction
2. Models
3. Main result
4. Extension
5. Related work

# Extension to signatures and secrecy properties

---

For protocols with

- ciphertext forwarding,
- public encryption, provided the public key encryption algorithm is IND-CCA2,
- signatures, provided the signature algorithm is existentially unforgeable

We obtain:

- Soundness for the case of trace properties (like authentication)  
→ extension of [MW04]
- Soundness for the case of secrecy properties  
→ first result of soundness of secrecy

⇒ Automatic proof of computational properties.

# Secrecy Properties

---

Formal models : property on traces

A data  $s$  is secret if the adversary can not produce  $s$ .

# Secrecy Properties

---

Formal models : property on traces

A data  $s$  is secret if the adversary can not produce  $s$ .

Concrete model : indistinguishability

The secrecy of  $s$  is defined through the following game:

- Two nonces  $n_0$  and  $n_1$  are randomly generated;
- The adversary interacts with the protocol where  $s$  is instantiated with  $n_b$ ,  $b \in \{0, 1\}$ ;
- We give the pair  $(n_0, n_1)$  to the adversary;
- The adversary gives  $b'$ ,

The data  $s$  is secret if  $\Pr[\text{Exp}^1 = 1] - \Pr[\text{Exp}^0 = 1]$  is a negligible function.

# Soundness of secrecy properties

---

We consider a particular predicate  $\text{SecNonce}(i, j)$  which says that the nonce  $n(a_i, j, s)$  must remain secret for any session  $s$  in which honest agents are involved.

## Theorem 2

If  $\Pi \models^c \text{SecNonce}^c(i, j)$  then  $n(a_i, j, s)$  remains secret in  $\Pi$  for the indistinguishability property.

*i.e. secrecy (expressed on traces) implies indistinguishability*

## Corollary

If  $\Pi \models^f \text{SecNonce}^f(i, j)$  then  $n(a_i, j, s)$  remains secret in  $\Pi$  for the indistinguishability property.

# Summary

---

## III. Soundness of formal models w.r.t. computational models

1. Introduction
2. Models
3. Main result
4. Extension
5. Related work

# Related Work 1/3

---

[BPW03] Michael Backes and Birgit Pfiztmann and Michael Waidner, *A Universally Composable Cryptographic Library*, Cryptology ePrint Archive, Report 2003/015, 2003, <http://eprint.iacr.org/>

[Can04] Ran Canetti, *Universally Composable Signature, Certification, and Authentication*, Proc. of 17th IEEE Computer Security Foundations Workshop (CSFW'04).

- very general results : symmetric and asymmetric encryption, pairing, signatures, MACs.
- less abstracted model than classical Dolev-Yao models,
- automatic verification have to be developed specifically for this model.

# Related Work 2/3

---

[BPW03] Peeter Laud, *Symmetric encryption in automatic analyses for confidentiality against active adversaries*, Proc. of 2004 IEEE Symposium on Security and Privacy, pages 71-85, 2004.

- symmetric encryption
- bounded number of sessions
- obtain a automatic procedure for computationally sound proof of confidentiality properties

# Related Work 3/3

---

[DDMP0503] A. Datta, A. Derek, J. C. Mitchell, D. Pavlovic, *A Derivation System and Compositional Logic for Security Protocols*, Journal of Computer Security (Special Issue of Selected Papers from CSFW-16)

- asymmetric encryption
- design of a sound derivation system
- direct automatization of the proof in the concrete model

# Some other references

---

- BDPR98** Mihir Bellare, Anand Desai, David Pointcheval and Phillip Rogaway. *Relations Among Notions of Security for Public-Key Encryption Schemes* In Advances in Cryptology - Proceedings of CRYPTO '98, pages 26-45, LNCS 1462, Springer-Verlag, 1998.
- CW'05** Veronique Cortier and Bogdan Warinschi *Computationally Sound, Automated Proofs for Security Protocols*. 14th European Symposium on Programming Languages (ESOP'05), pages 157-171, LNCS 3444, Springer-Verlag, 2005.
- FOPS05** Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval and Jacques Stern *RSA-OAEP is Secure under the RSA Assumption*. Journal of Cryptology. Volume 17, Number 2, pages 81-104, Springer-Verlag, 2004.
- MW04** Daniele Micciancio and Bogdan Warinschi. *Soundness of Formal Encryption in the presence of Active Adversaries*. Theory of Cryptography Conference (TCC 2004), pages 133-151, LNCS 2951, Springer-Verlag, 2004.