

# Computationally Sound Symbolic Secrecy with Hash Functions.

Véronique Cortier, Steve Kremer, Ralf Küsters and Bogdan Warinschi

# 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.

# Previous result: Soundness of secrecy properties

---

**Setting:** pairing, asymmetric encryption and signatures

**Theorem [Esop'05]**

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

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

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

**Corollary [Esop'05]**

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

# This does not work in general

---

Example:

$$A \rightarrow B : h(s)$$

$s$  is **weakly secret** but not **indistinguishable** to an attacker.

# This does not work in general

---

Example:

$$A \rightarrow B : h(s)$$

$s$  is **weakly secret** but not **indistinguishable** to an attacker.

**Goal:** A **sound** (and faithful) formal criterion for indistinguishability.

# Contributions

---

1. Design of a **new formal secrecy property**
2. Proof of its **soundness and its faithfulness** w.r.t. indistinguishability
3. **Mapping** from concrete to formal trace in our new setting:
  - pairing
  - asymmetric encryption
  - hashes (random oracle model)

# Patterns

---

Definition of what is “visible” to an intruder.

Given  $S = \{M_1, M_2, \dots, M_k\}$  and  $T$ , we define

$$\text{Pat}_T(S) = \{\text{Pat}_T^S(M_1), \text{Pat}_T^S(M_2), \dots, \text{Pat}_T^S(M_k)\} \quad \text{with}$$

# Patterns

---

Definition of what is “visible” to an intruder.

Given  $S = \{M_1, M_2, \dots, M_k\}$  and  $T$ , we define

$$\text{Pat}_T(S) = \{\text{Pat}_T^S(M_1), \text{Pat}_T^S(M_2), \dots, \text{Pat}_T^S(M_k)\} \quad \text{with}$$

$$\text{Pat}_T^S(a) = \begin{cases} a & \text{if } S, T \vdash a \\ \square & \text{otherwise} \end{cases}$$

$$\text{Pat}_T^S(\langle M_1, M_2 \rangle) = \langle \text{Pat}_T^S(M_1), \text{Pat}_T^S(M_2) \rangle$$

# Patterns

---

Definition of what is “visible” to an intruder.

Given  $S = \{M_1, M_2, \dots, M_k\}$  and  $T$ , we define

$$\text{Pat}_T(S) = \{\text{Pat}_T^S(M_1), \text{Pat}_T^S(M_2), \dots, \text{Pat}_T^S(M_k)\} \quad \text{with}$$

$$\text{Pat}_T^S(a) = \begin{cases} a & \text{if } S, T \vdash a \\ \square & \text{otherwise} \end{cases}$$

$$\text{Pat}_T^S(\langle M_1, M_2 \rangle) = \langle \text{Pat}_T^S(M_1), \text{Pat}_T^S(M_2) \rangle$$

$$\text{Pat}_T^S(\{M\}_{\text{ek}(a)}^r) = \begin{cases} \{\text{Pat}_T^S(M)\}_{\text{ek}(a)}^r & \text{if } S, T \vdash \text{dk}(a) \\ \{\text{Pat}_T^S(M)\}_{\text{ek}(a)}^r & \text{or if } r \in \text{Rand}_{adv} \\ \square & \text{otherwise} \end{cases}$$

# Patterns

Definition of what is “visible” to an intruder.

Given  $S = \{M_1, M_2, \dots, M_k\}$  and  $T$ , we define

$$\text{Pat}_T(S) = \{\text{Pat}_T^S(M_1), \text{Pat}_T^S(M_2), \dots, \text{Pat}_T^S(M_k)\} \quad \text{with}$$

$$\begin{aligned} \text{Pat}_T^S(a) &= \begin{cases} a & \text{if } S, T \vdash a \\ \square & \text{otherwise} \end{cases} \\ \text{Pat}_T^S(\langle M_1, M_2 \rangle) &= \langle \text{Pat}_T^S(M_1), \text{Pat}_T^S(M_2) \rangle \\ \text{Pat}_T^S(\{M\}_{\text{ek}(a)}^r) &= \begin{cases} \{\text{Pat}_T^S(M)\}_{\text{ek}(a)}^r & \text{if } S, T \vdash \text{dk}(a) \\ \{\text{Pat}_T^S(M)\}_{\text{ek}(a)}^r & \text{or if } r \in \text{Rand}_{adv} \\ \square & \text{otherwise} \end{cases} \\ \text{Pat}_T^S(h(M)) &= \begin{cases} h(\text{Pat}_T^S(M)) & \text{if } S, T \vdash M \\ \square & \text{otherwise} \end{cases} \end{aligned}$$

# Pattern-based secrecy definition

---

$\Pi$  protocol

$X_{A_i}^j$  nonce variable occurring in some role  $A_i$ .

$S$  set of sent messages

$s$  session number

$$\bar{S} = S \cup \{M \mid \{M\}_{ek(a)}^r \text{ subterm of } S, r \in \text{Rand}_{adv}\}$$

# Pattern-based secrecy definition

---

$\Pi$  protocol

$X_{A_i}^j$  nonce variable occurring in some role  $A_i$ .

$S$  set of sent messages

$s$  session number

$$\bar{S} = S \cup \{M \mid \{M\}_{ek(a)} \text{ subterm of } S, r \in \text{Rand}_{adv}\}$$

**Definition:**  $X_{A_i}^j$  is **secret** in  $\Pi$ , written  $\Pi \models^f \text{SecNonce}(i, j)$ , if:

$$n^{a_i, j, s} \text{ does not occur in } \text{Pat}_{n^{a_i, j, s}}(\bar{S}) \quad \forall S \forall s$$

# Examples

---

- $\phi_1 = \{h(\langle n_b, n' \rangle)\} = \overline{\phi_1}$ . Then  $\text{Pat}_{n_b}(\overline{\phi_1}) = \{\square\}$   
→  $n_b$  is intuitively hidden by  $n'$ .

# Examples

---

- $\phi_1 = \{h(\langle n_b, n' \rangle)\} = \overline{\phi_1}$ . Then  $\text{Pat}_{n_b}(\overline{\phi_1}) = \{\square\}$   
→  $n_b$  is intuitively hidden by  $n'$ .
- $\phi_2 = \{h(\langle n_b, \{n'\}_{\text{ek}(a)}^r \rangle), n'\} = \overline{\phi_2}$  where  $r \notin \text{Rand}_{adv}$ .  
 $\text{Pat}_{n_b}(\overline{\phi_2}) = \{\square, n'\}$ .  
→ The encryption of  $n'$  does hide  $n_b$ .

# Examples

---

- $\phi_1 = \{h(\langle n_b, n' \rangle)\} = \overline{\phi_1}$ . Then  $\text{Pat}_{n_b}(\overline{\phi_1}) = \{\square\}$   
→  $n_b$  is intuitively hidden by  $n'$ .
- $\phi_2 = \{h(\langle n_b, \{n'\}_{\text{ek}(a)}^r \rangle), n'\} = \overline{\phi_2}$  where  $r \notin \text{Rand}_{adv}$ .  
 $\text{Pat}_{n_b}(\overline{\phi_2}) = \{\square, n'\}$ .  
→ The encryption of  $n'$  does hide  $n_b$ .
- $\phi_3 = \{h(\langle n_b, \{n'\}_{\text{ek}(a)}^r \rangle)\}$  where  $r \in \text{Rand}_{adv}$ .  
Then  $\overline{\phi_3} = \phi_3 \cup \{n'\}$  and  $\text{Pat}_{n_b}(\overline{\phi_3}) = \{h(\langle n_b, \{n'\}_{\text{ek}(a)}^r \rangle), n'\}$ .

# Formal Model

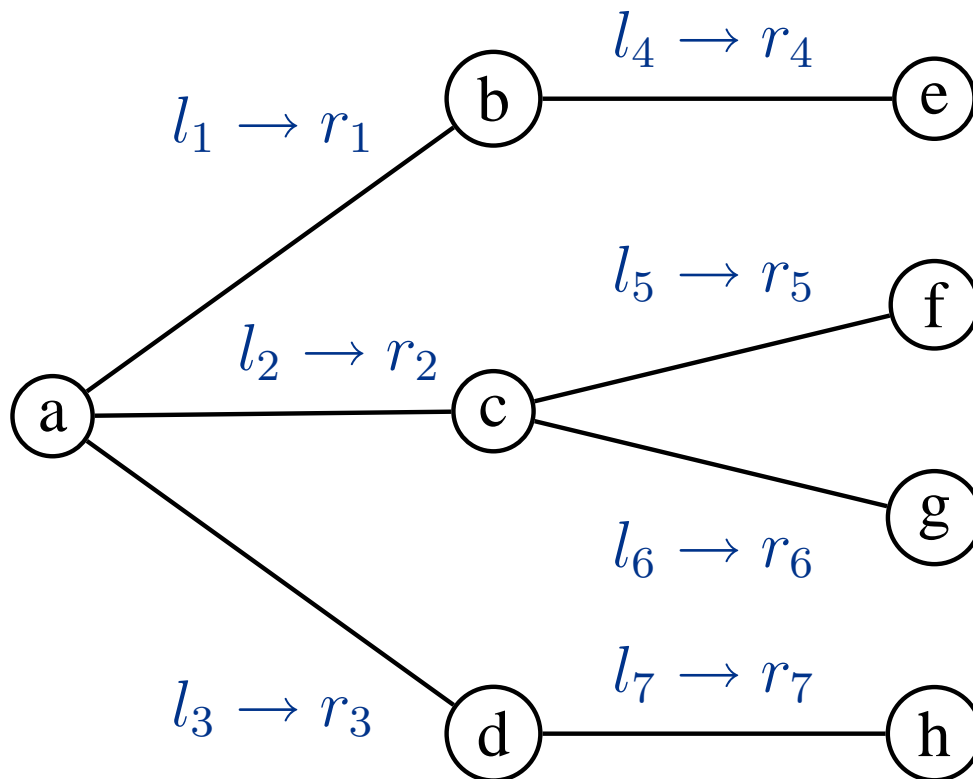
---

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

# Formal Model

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

A **protocol** is a finite set of **roles** of the form:



# Example

---

## 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

---

## Needham-Schroeder-Lowe protocol

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

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

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

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

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

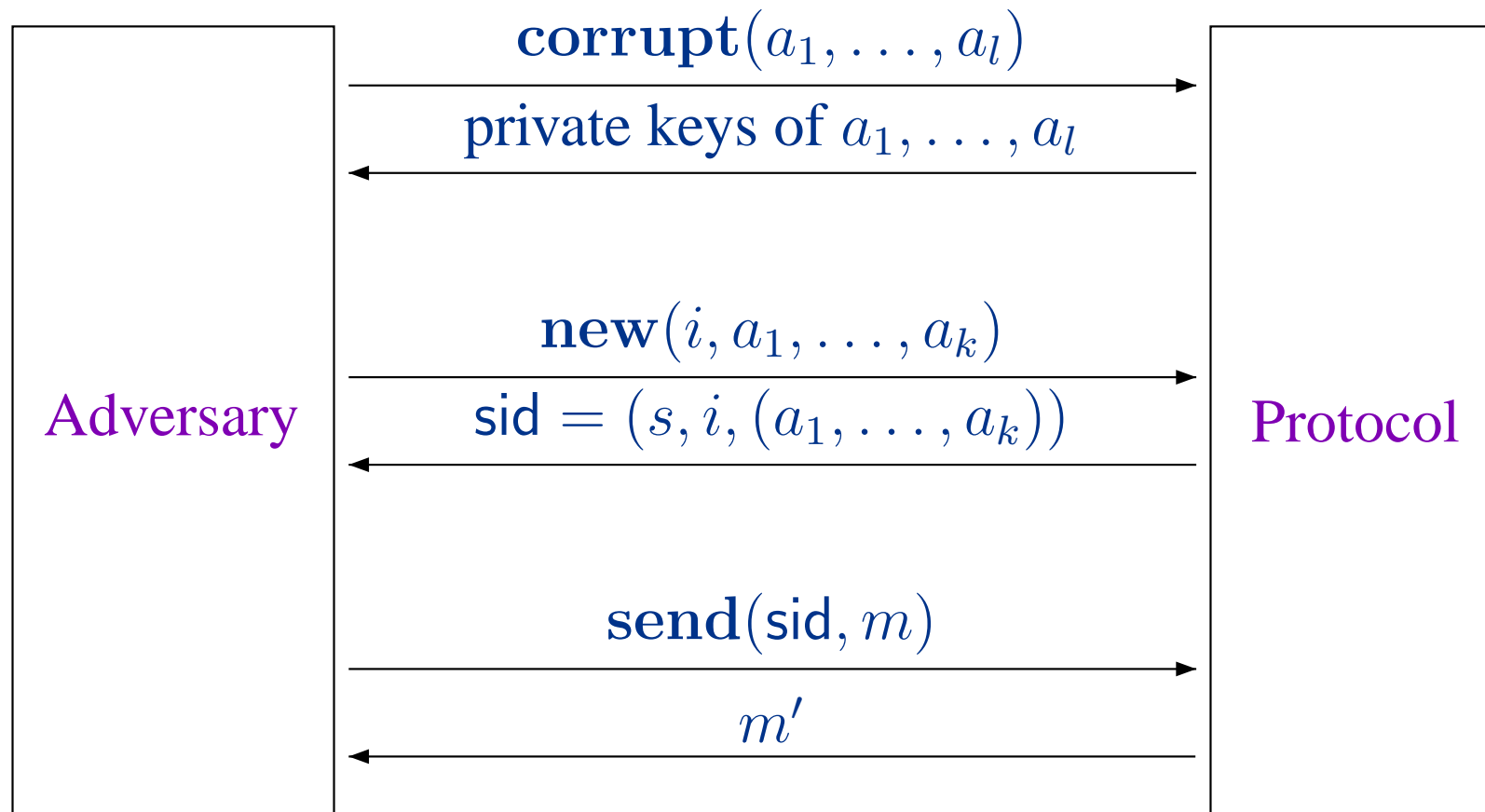
$$\Pi(2) = (\{X_{A_1}^1, A_1\}_{\text{ek}(A_2)}^{L_1} \rightarrow \{X_{A_1}^1, X_{A_2}^1, A_2\}_{\text{ek}(A_1)}^{\text{ag}(1)}),$$

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

Variables are local to each role and each session.

# Network

---



# Formal Intruder Deduction Rules

---

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

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

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

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

# 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.
- hash : Random Oracle Model

# 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.
- hash : Random Oracle Model
- parsing :
  - each bit-string has a label which indicates his type (identity, nonce, key, hash, ...)
  - one can retrieve the (public) encryption key from an encrypted message.

# First result: correspondence between traces

---

## Theorem 1:

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

*Proof:* Adaptation of [Esop05] for hash functions in the Random Oracle Model

# First result: correspondence between traces

---

## Theorem 1:

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

*Proof:* Adaptation of [Esop05] for hash functions in the Random Oracle Model

## Corollary:

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

**Then**  $\Pi \models^s \phi^s$  **implies**  $\Pi \models^c \phi^c$ .

*Applications* : authentication,...

# Main result

---

## Theorem 2

$$\Pi \models^f \text{SecNonce}^f(i, j) \quad \text{iff} \quad \Pi \models^c \text{SecNonce}^c(i, j)$$

**Remark:** Our formal secrecy definition is both **sufficient** and **necessary** for indistinguishability in the computational world.

# Necessary condition

---

**Lemma 1:**  $n$  does not occur in  $\text{Pat}_n(\bar{S})$  if and only if

- $\bar{S} \not\vdash n$
- and  $\forall M$  subterm of  $S$  such that  $\bar{S} \vdash M$ ,  $\forall p$  such that  $M|_p = n$ ,
  - either there exists an encryption along  $p$
  - or  $\exists p' < p$  such that
    1.  $M|_{p'} = h(M')$  and
    2.  $\phi, n \not\vdash M'$ .

# Necessary condition

---

**Lemma 1:**  $n$  does not occur in  $\text{Pat}_n(\bar{S})$  if and only if

- $\bar{S} \not\vdash n$
- and  $\forall M$  subterm of  $S$  such that  $\bar{S} \vdash M$ ,  $\forall p$  such that  $M|_p = n$ ,
  - either there exists an encryption along  $p$
  - or  $\exists p' < p$  such that
    1.  $M|_{p'} = h(M')$  and
    2.  $\phi, n \not\vdash M'$ .

**Thus** if  $n$  occurs in  $\text{Pat}_n(\bar{S})$ , there exists  $M$  such that

- $\bar{S} \vdash M \rightarrow$  there exists a (efficient) recipe to obtain  $M$  from  $S$
- There exists a recipe to obtain  $M$  from  $S, n$  that does not work with  $S, n'$ .

**Example:**  $S = \{\langle h(\langle n_b, n' \rangle), n' \rangle\}$

# Sufficient condition: difficulties

---

Suppose  $\mathcal{A}$  breaks the indistinguishability game. We have to construct  $\mathcal{B}$  that breaks encryption.

$\mathcal{B}$  generates two nonces  $n_0$  and  $n_1$  and simulates the protocol.

Suppose  $n_b$  appears in clear in  $M[n_b]$ .

- To send  $\{M[n_b]\}_{\text{ek}(a)}$ ,  $\mathcal{B}$  computes  $M[n_0]$  and  $M[n_1]$  and uses the left-right oracle
- To send  $h(M[n_b])$ ,  $\mathcal{B}$  generates a new random value.

# Sufficient condition: difficulties

---

Suppose  $\mathcal{A}$  breaks the indistinguishability game. We have to construct  $\mathcal{B}$  that breaks encryption.

$\mathcal{B}$  generates two nonces  $n_0$  and  $n_1$  and simulates the protocol.

Suppose  $n_b$  appears in clear in  $M[n_b]$ .

- To send  $\{M[n_b]\}_{ek(a)}$ ,  $\mathcal{B}$  computes  $M[n_0]$  and  $M[n_1]$  and uses the left-right oracle
- To send  $h(M[n_b])$ ,  $\mathcal{B}$  generates a new random value.

Then  $\mathcal{A}$  is given  $n_0, n_1$  and can still query the random oracle (controlled by  $\mathcal{B}$ ).

$\mathcal{B}$  must be able to **answer consistently** the hash queries.

**Example:**  $S = \{\{h(n_b)\}_{ek(a)}\}$  and the hash of  $n_0$  is queried.

# Sufficient condition: soundness result

---

- We define an **magical execution** where encrypted message  $\{m\}_{ek(a)}$  are replaced by  $\{r_m\}_{ek(a)}$  where  $r_m$  is a random value of equal length.

**Lemma 2:** If the encryption scheme is IND-CCA then

$$\Pr [\text{Exec}_{\mathcal{A},\Pi}(\eta) = 1] - \Pr [\text{Exec}_{\mathcal{A},\Pi}^o(\eta) = 1]$$

is negligible.

# Sufficient condition: soundness result

---

- We define an **magical execution** where encrypted message  $\{m\}_{ek(a)}$  are replaced by  $\{r_m\}_{ek(a)}$  where  $r_m$  is a random value of equal length.

**Lemma 2:** If the encryption scheme is IND-CCA then

$$\Pr [\text{Exec}_{\mathcal{A},\Pi}(\eta) = 1] - \Pr [\text{Exec}_{\mathcal{A},\Pi}^o(\eta) = 1]$$

is negligible.

- Using **Theorem 1**, we obtain that any concrete trace of  $\text{Exec}_{\mathcal{A},\Pi}^o$  is the image of a valid formal trace thus verifies the pattern-based secrecy property.

# Sufficient condition: soundness result

---

- We define an **magical execution** where encrypted message  $\{m\}_{\text{ek}(a)}$  are replaced by  $\{r_m\}_{\text{ek}(a)}$  where  $r_m$  is a random value of equal length.

**Lemma 2:** If the encryption scheme is IND-CCA then

$$\Pr [\text{Exec}_{\mathcal{A},\Pi}(\eta) = 1] - \Pr [\text{Exec}_{\mathcal{A},\Pi}^{\circ}(\eta) = 1]$$

is negligible.

- Using **Theorem 1**, we obtain that any concrete trace of  $\text{Exec}_{\mathcal{A},\Pi}^{\circ}$  is the image of a valid formal trace thus verifies the pattern-based secrecy property.
- We conclude using **Lemma 2** that hash queries can indeed be answered consistently.

# Further work

---

- **equivalent definition** of the pattern-based secrecy property: equivalence-based definition?
- Decidability result / **decision procedures** in the formal models
- Other **cryptographic primitives**: symmetric encryption, signatures, ...
- ...