

CIL: A Proof System for Computational Indistinguishability

Gilles Barthe³, Marion Daubignard², Bruce Kapron¹ and
Yassine Lakhnech²

¹University of Victoria

²VERIMAG, Université de Grenoble, CNRS

³IMDEA, Madrid

19th June, 2009

This work is partially supported by the ANR project SCALP

Aim

Provable security provides guarantees, but...

- Problems : nowadays, one scheme = one proof, proofs are intricate, and therefore somewhat unreliable...
- Our long-term goal is to prove cryptographic systems secure by enabling

Computer-Aided Cryptographic Proofs

at the level of abstract constructions and their implementations.

- Existing approaches: game-based techniques, Hoare logics, applied pi-calculus...
- Most security criteria rely on the concept of indistinguishability.

Hence our current subgoal: designing a versatile system of inference rules to prove indistinguishability.

Indistinguishability of Distributions

Advantage of an adversary in distinguishing \mathcal{D}_0 and \mathcal{D}_1

$$\text{Adv}(\eta, \mathcal{A}) = |\Pr[\mathcal{A}^{\mathcal{O}_i}(\mathcal{D}_1) = 1] - \Pr[\mathcal{A}^{\mathcal{O}_i}(\mathcal{D}_0) = 1]|$$

Indistinguishability of two distributions

\mathcal{D}_0 and \mathcal{D}_1 are indistinguishable iff $\sup_{\mathcal{A}}(\text{Adv}(\eta, \mathcal{A}))$ is a negligible function in η .

This is denoted $\mathcal{D}_0 \sim \mathcal{D}_1$.

def. : $f(\eta)$ negligible iff $\forall k \geq 0, \eta^k \times f(\eta) \xrightarrow{\eta \rightarrow \infty} 0$

Outline

- 1 Our Framework: Computational Frames
- 2 CIL: The Inference System
- 3 Reasoning With Oracles

Outline

- 1 Our Framework: Computational Frames
- 2 CIL: The Inference System
- 3 Reasoning With Oracles

What defines the distributions we are interested in?

The IND-CPA game for a scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D})$

- ① keys are drawn : $(pk, sk) \xleftarrow{r} \mathcal{K}(\eta)$,
 - ② $\mathcal{A}_1(pk)$ choses a pair of messages and outputs them plus some state information: (s, m_0, m_1) ,
 - ③ b is chosen at random, m_b is ciphered: $y = \mathcal{E}(m_b)$,
 - ④ $\mathcal{A}_2(s, pk, m_0, m_1, y)$ decides which message was encrypted.
- fresh random values are drawn – (key pairs)
 - adversary calls are made – (\mathcal{A}_1)
 - \mathcal{A}_2 gets as an input a tuple (s, m_0, m_1, y) , depending on the previous computations
 - \mathcal{A}_1 and \mathcal{A}_2 can query oracles.

Introduction To Computational Frame Syntax

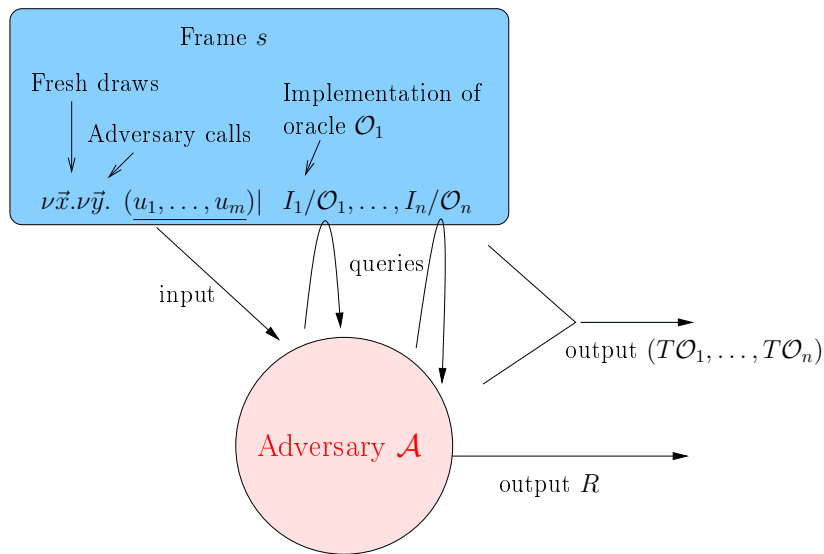
A frame is a distribution denoted

$$s = \nu \vec{x}. \nu \vec{a}. (u_1, \dots, u_m) | I_1 / \mathcal{O}_1, \dots, I_n / \mathcal{O}_n$$

where:

- 1 \vec{x} stands for $x_1 \stackrel{r}{\leftarrow} U_1, \dots, x_k \stackrel{r}{\leftarrow} U_k$. Those represent fresh drawings in independent distributions.
- 2 \vec{a} represents a list of adversary calls $a_i \stackrel{r}{\leftarrow} \mathcal{A}_i^{\vec{\mathcal{O}}}(in_i)$. The inputs (in_i 's) can depend on x_k 's and preceding a_j 's.
- 3 (u_1, \dots, u_m) are expressions depending on \vec{x} and \vec{a} .
- 4 I_j is the implementation of oracle \mathcal{O}_j .

Interaction with an adversary



ex. : event $R=1$.

Formal Computational Frame Semantics

Let $s = \nu \vec{x}. \nu \vec{a}. (u_1, \dots, u_m) | I_1 / \mathcal{O}_1, \dots, I_n / \mathcal{O}_n$ be a frame, and $\vec{\mathcal{A}} = (\mathcal{A}_1, \dots, \mathcal{A}_p, \mathcal{A})$.

They define $\vec{\mathcal{A}} || s$, the resulting distribution on $(\vec{x}, \vec{a}, \vec{I}, \vec{u}, R, T\hat{\mathcal{O}})$, as follows:

- 1 For each i , a value \hat{x}_i is drawn in U_i and assigned to x_i .
- 2 For each j , $a_j \stackrel{r}{\leftarrow} \mathcal{A}_j^{\vec{\mathcal{O}}}(in_j)$ means in_j is computed, \mathcal{A}_j gets it as an input, possibly calls oracles $\vec{\mathcal{O}}$ (implemented following \vec{I}), and outputs an answer \hat{a}_j , which is assigned to a_j .
- 3 Calls to oracles: anytime an adversary queries $\mathcal{O}_k(bs)$, it gets $I_k(bs)$ and as a **side effect**, $T\hat{\mathcal{O}} := T\hat{\mathcal{O}} :: [(k, bs, I_k(bs))]$.
- 4 Values $(\hat{u}_1, \dots, \hat{u}_m)$ are computed for expressions (u_1, \dots, u_m) , and given as an input to \mathcal{A} .
- 5 After some polytime computation including possible oracle calls, \mathcal{A} outputs a bitstring \hat{R} assigned to variable R , and $T\hat{\mathcal{O}}$, where $T\hat{\mathcal{O}}_k$ is the list of all queries to \mathcal{O}_k .

Conditional Indistinguishability of Frames $E \rightarrow s \sim t$

Conditional Advantage

Let $\vec{\mathcal{A}}$ be a list of adversaries. Its advantage in distinguishing s and t given E is:

$$\text{Adv}(\vec{\mathcal{A}}, s, t, \eta) = \\ |\Pr[(\vec{\hat{x}}, \vec{\hat{a}}, \vec{\hat{u}}, \vec{\hat{I}}, \vec{\hat{R}}, T\vec{\hat{\mathcal{O}}}) \stackrel{r}{\leftarrow} (\vec{\mathcal{A}} \parallel s) : \hat{R} = 1 | E] - \\ \Pr[(\vec{\hat{x}}, \vec{\hat{a}}, \vec{\hat{u}}, \vec{\hat{I}}, \vec{\hat{R}}, T\vec{\hat{\mathcal{O}}}) \stackrel{r}{\leftarrow} (\vec{\mathcal{A}} \parallel t) : \hat{R} = 1 | E]|$$

 $E \rightarrow s \sim t$

s is indistinguishable from t given E iff $\forall \vec{\mathcal{A}}, \text{Adv}(\vec{\mathcal{A}}, s, t, \eta)$ is negligible in η .

Conditional Negligibility of Events $E_2 \rightarrow s : E_1$

We can define negligibility for any event E_1 depending on variables in $\vec{x}, \vec{a}, \vec{u}, R, T\vec{O}, \vec{I}$ or \vec{O} .

$E_2 \rightarrow s : E_1$


Let $\vec{\mathcal{A}}$ be a list of adversaries. Event A is negligible in s iff $|\Pr[\alpha \xleftarrow{r} (\vec{\mathcal{A}} \parallel s) : E_1(\alpha) | E_2(\alpha)]|$ is negligible in η .

Outline

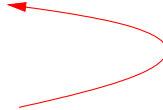
- 1 Our Framework: Computational Frames
- 2 CIL: The Inference System
- 3 Reasoning With Oracles

Two ways of reading a rule

- from top to bottom:

$$\frac{s \sim t}{v[s/y] \sim v[t/y]}$$


- the other way round (as for reductionist proofs):

$$\frac{s \not\sim t}{v[s/y] \not\sim v[t/y]}$$


suppose there is an adversary that breaks the conclusion,
then there is a way to modify it to break the premise!

The Substitution Rules

Let s be a frame, and let v be a poly-time term with a free variable y . Substitution of s to y is performed **avoiding name capture** (even for oracle implementations).

$$\frac{A \rightarrow s \sim t}{A \rightarrow v[s/y] \sim v[t/y]} \text{Sub}$$

$$\frac{A \rightarrow s : E_1}{A \rightarrow v(s) : E_1} \text{NegSub}$$

Idea of the reduction: the context is polytime simulatable.

Indistinguishability by Case Study

The use of this rule motivates the introduction of conditional reasoning.

one specific query was made to an oracle

... was not made to the oracle

the scheme is secure

(captured by an indistinguishability statement)

$$\frac{E \rightarrow s \sim t \quad s : \neg E \quad t : \neg E}{s \sim t} \text{CS}$$

Intuitively, either E holds and $s \sim t$, or $\neg E$ holds, but this happens with negligible probability.

Importing External Reasoning : 1.Equality

Def.: $s =_X t$ iff $[\alpha \stackrel{r}{\leftarrow} s : \Pi_X(\alpha)], = [\alpha \stackrel{r}{\leftarrow} t : \Pi_X(\alpha)]$
 where Π_X is the projection on X .

$$\frac{A \rightarrow s =_R t}{A \rightarrow s \sim t} \text{UNIV}$$

...because the advantage of any adversary is null.

$$\frac{A \rightarrow s : E(X) \quad A \rightarrow s =_X t}{A \rightarrow t : E(X)} \text{NegUNIV}$$

...because E depends exclusively on variables in X .

Importing External Reasoning : 2.Logical Disjunction

$$\begin{array}{c} \bigwedge_{i=1}^p (A_i[\vec{I}/\vec{O}] \rightarrow B_i[\vec{I}/\vec{O}]) \Rightarrow (A[\vec{I}/\vec{O}] \rightarrow B[\vec{I}/\vec{O}]) \\ A_1 \rightarrow s : B_1 \\ \vdots \\ A_p \rightarrow s : B_p \\ \hline A \rightarrow s : B \quad \text{UCR} \end{array}$$

- If $\forall i, \Pr[B_i|A_i]$ is negligible, then $\Pr[B|A]$ is negligible (with universal quantification of free variables, except for oracle names that we replace by implementations).

Importing External Reasoning : 2.Logical Disjunction

$$\begin{array}{c} \bigwedge_{i=1}^p (A_i[\vec{I}/\vec{O}] \rightarrow B_i[\vec{I}/\vec{O}]) \Rightarrow (A[\vec{I}/\vec{O}] \rightarrow B[\vec{I}/\vec{O}]) \\ A_1 \rightarrow s : B_1 \\ \vdots \\ A_p \rightarrow s : B_p \\ \hline A \rightarrow s : B \quad \text{UCR} \end{array}$$

- If $\forall i, \Pr[B_i|A_i]$ is negligible, then $\Pr[B|A]$ is negligible
- $\forall i, \Pr[B_i|A_i]$ actually is negligible

Importing External Reasoning : 2.Logical Disjunction

$$\frac{\begin{array}{c} \bigwedge_{i=1}^p (A_i[\vec{I}/\vec{O}] \rightarrow B_i[\vec{I}/\vec{O}]) \Rightarrow (A[\vec{I}/\vec{O}] \rightarrow B[\vec{I}/\vec{O}]) \\ A_1 \rightarrow s : B_1 \\ \vdots \\ A_p \rightarrow s : B_p \end{array}}{A \rightarrow s : B} \text{UCR}$$

Useful rules that we can derive from UCR:

$$\frac{A \rightarrow s : B}{s : A \wedge B} \text{UCR} \qquad \frac{\begin{array}{c} A \rightarrow s : B_1 \\ \vdots \\ A \rightarrow s : B_p \end{array}}{A \rightarrow s : \bigvee_{i=1}^p (B_i)} \text{UCR}$$

$$(A \rightarrow B) \Rightarrow (A \wedge B) \qquad (\bigwedge_{i=1}^p (A \rightarrow B_i)) \Rightarrow (A \rightarrow \bigvee_{i=1}^p (B_i))$$

A Reduction Rule (deducible from others)

v is a probabilistic poly-time term to be exhibited when applying the rule.

$$\frac{A \rightarrow s : E_1 \quad A(v(\alpha)) \Rightarrow A(\alpha) \quad A \rightarrow s : E_2 \wedge \neg E_1 \circ v}{A \rightarrow s : E_2} \text{ NegRED}$$

Idea: simple reduction by embedding an adversary against the conclusion and applying v to its output:

Output α satisfying $E_2|A$

$$\begin{aligned} & A(\alpha) \Rightarrow A(v(\alpha)) \\ & E_2(\alpha) \Rightarrow E_1(v(\alpha)) \end{aligned}$$

Output $v(\alpha)$ satisfying $E_1|A$

Outline

- 1 Our Framework: Computational Frames
- 2 CIL: The Inference System
- 3 Reasoning With Oracles**

Changing Answers to Queries

- One often needs to reduce breaking a scheme to solving a hard problem (e.g., inverting a one-way function).
- To trick the adversary against the scheme into solving the hard problem, replace the answer to some query by some bitstring related to the challenge to the hard problem.
- In our framework, it translates in changing the implementation of an oracle (say, \mathcal{O}_1) on one expression e .
- For negligibility, two cases: the event we are interested in has the form $e \notin T\mathcal{O}_1 \wedge \dots$ or the form $e \in T\mathcal{O}_1 \wedge \dots$

If the event contains 'e is not queried'...

e is an expression possibly depending on \vec{x}, \vec{a} or u_i 's or is R .

$$\text{NegOR}\forall \frac{A \rightarrow s | I_1 / \mathcal{O}_1 : e \notin T\mathcal{O}_1 \wedge E(\vec{x}, \vec{y}, R, T\mathcal{O}) \quad q \neq e \Rightarrow I_1(q) = I'_1(q) (*)}{A \rightarrow s | I'_1 / \mathcal{O}_1 : e \notin T\mathcal{O}_1 \wedge E(\vec{x}, \vec{y}, R, T\mathcal{O})}$$

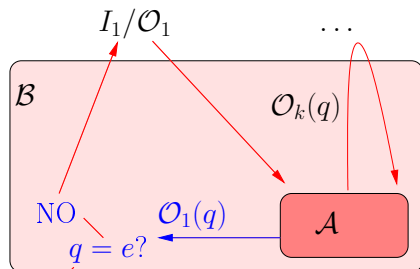
with:

(*) $q \neq e \Rightarrow I_1(q) = I'_1(q)$ meaning that the implementations yield the same result on any query but e . All variables occurring in the statement are quantified universally.

Intuitively, running an adversary in the first or second context leads to the same execution...

If the event contains 'e is queried'...

$$\text{NegOR}\exists \frac{s|I_1/\mathcal{O}_1 : e \in T\mathcal{O}_1 \wedge E(\vec{x}, \vec{y}, T\vec{\mathcal{O}}) \quad E \text{ is } T\mathcal{O}\text{-prefix closed} \quad q \neq e \Rightarrow I_1(q) = I'_1(q)}{s|I'_1/\mathcal{O}_1 : e \in T\mathcal{O}_1 \wedge E(\vec{x}, \vec{y}, T\vec{\mathcal{O}})}$$



YES
STOP, E holds (prefix-closed)

- $e \in T\mathcal{O}_1$ is ascertainable (i.e. the adversary can check whether it holds),
- if not, draw when to stop at random.

Two More Rules

$$\frac{A \rightarrow s|I_1/\mathcal{O}_1 : e \in T\mathcal{O}_1 \quad q \neq e \Rightarrow I_1(q) = I'_1(q)}{A \rightarrow s|I_1/\mathcal{O}_1 \sim s|I'_1/\mathcal{O}_1} \text{ OR}$$

Idea: querying e has same probability in both contexts.

$$\frac{s : E' \quad E(T\mathcal{O}) \Rightarrow \exists T\mathcal{O}' \preceq T\mathcal{O} \cdot E'(T\mathcal{O}')}{s : E} \text{ TEMP}$$

where $T\mathcal{O}' \preceq T\mathcal{O}$ denotes $T\mathcal{O}'$ prefix from $T\mathcal{O}$.

Idea: stop the execution of adversaries against $s : E$ once E' is fulfilled.

Conclusion

CIL is a system to prove indistinguishability of computational frames, in which we can prove:

- any asymmetric encryption scheme we could prove with the previous formalism (Hoare logic [CCS08]),
- ElGamal, Hashed El-Gamal in the ROM or standard model,
- OAEP,
- signature schemes (FDH is finished and PSS is nearly concluded)

Formalization of CIL in Coq is progressing (SCALP Project).