Pseudorandom Permutations and Equivalence of Formal Expressions

Peeter Laud
peeter_l@ut.ee

Tartu Ülikool
Cybernetica AS

$\boxed{\textbf{Background}}$

- In cryptographic protocols, messages are exchanged between participants.
- When are two messages equivalent?
  - This question arises, when we analyse protocols.
  - Equivalence — the adversary cannot tell them apart.
- The messages come from the set of *formal expressions* — a certain term algebra.

Formal Expressions (1/2)

The set of formal expressions **Exp** is similar to a programming language without a looping construct.

For defining it, we need the following sets:

- The set of *formal keys* **Keys**.
- The set of *operators* **Op**.
  - The *arities* of operators: $ar : \mathbf{Op} \to \mathbb{N}$.
- The set *formal inputs* **Inp**.
  - Their meaning is different from nullary operators.

$\boxed{\textbf{Formal Expressions (2/2)}}$

The set **Exp** contains the following elements:

- All elements of **Keys** and **Inp** belong to **Exp**.
- If $o \in \mathbf{Op}$ and $E_1, \ldots, E_{ar(o)} \in \mathbf{Exp}$, then $o(E_1, \ldots, E_{ar(o)})$ also belongs to **Exp**.
- If $K \in \mathbf{Keys}$ and $E \in \mathbf{Exp}$, then $\{E\}_K$ also belongs to **Exp**.

Assume that **Op** also contains operators for pairing / tupling:

- If $E_1, \ldots, E_k \in \mathbf{Exp}$, then also $(E_1, \ldots, E_k) \in \mathbf{Exp}$.

Semantics of Expressions

Let $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ be a *secure* encryption scheme.

No efficient algorithm can *break* it.

- The semantics $[\![K]\!]$ of $K \in \mathbf{Keys}$ is a bit-string generated by $\mathcal{G}$.
  - Different invocations of $\mathcal{G}$ for different keys.
- $[\![I]\!]$ for $I \in \mathbf{Inp}$ is just a bit-string that is somehow distributed.
  - The values of $[\![I_1]\!]$ and $[\![I_2]\!]$ may be dependent.
- $[\![\{E\}_K]\!]$ is the bit-string returned by $\mathcal{E}_{[\![K]\!]}([\![E]\!])$.

For each $o \in \mathbf{Op}$ let $[\![o]\!]$ be an algorithm that takes $ar(o)$ bit-strings and returns a bit-string.

- $[\![o(E_1, \ldots, E_{ar(o)})]\!]$ is the bit-string returned by $[\![o]\!]([\![E_1]\!], \ldots, [\![E_{ar(o)}]\!])$.

$\boxed{\textbf{Motivation}}$

How to find whether the distributions of interpretations of two expressions are so close that no efficient algorithm can tell them apart?

- Depends on the meaning of security for encryption schemes.
- Known, if the scheme hides the identities of both plaintexts and keys.
  - Based on rewriting / “simplifying” the expressions.
- Such schemes are not the most primitive cryptographic primitives.
  - They are constructed from more primitive objects — *pseudorandom permutations (PRPs)*.
  - DES, AES, BlowFish, Idea, etc. are designed to be pseudorandom permutations.

Pseudorandom permutations

A pseudorandom permutation is an encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$, such that:

no efficient algorithm can tell apart the distributions of:

- $\mathcal{E}_k(\cdot)$, where k is distributed according to $\mathcal{G}()$.
- $\pi(\cdot)$, where π is a uniformly chosen random permutation of bit-strings (of the same length as the inputs to $\mathcal{E}_k$).

Pseudorandom permutations do not hide the identities of plaintexts and keys.

- A related notion is *pseudorandom function (PRF)*.
- PRPs are PRFs.

$\boxed{\textbf{Structure of the analysis}}$

- Introduce a set **Clms** of *claims*.
- Each claim states something about the distribution of some $[\![E]\!]$ or about the distribution of $[\![E_1]\!], \ldots, [\![E_k]\!]$.
- For the given distribution of (interpretations of) inputs, a claim may be either true or false.
- There are rules to derive (the trueness of) claims about $[\![E_1]\!], \ldots, [\![E_k]\!]$ from the claims about the subexpressions of $E_1, \ldots, E_k$.
- The analysis is seeded with claims about the distribution of inputs.
- The rules are used to derive claims about more complex expressions.

The language of claims (1/2)

Let $E, E_i, E'_i \in \mathbf{Exp}$. The language **Clms** contains:

- $\mathsf{C}(E)$. True, iff $[\![E]\!]$ is almost always the same bit-string.
- $\mathsf{R}(E)$. True, iff $[\![E]\!]$ has only a negligible chance of being a *particular* bit-string.
- $\mathsf{U}(E)$. True, iff the distribution of $[\![E]\!]$ is indistinguishable from the uniform distribution of bit-strings of length $|[\![E]\!]|$.
- $\not\mathsf{E}(E_1, E_2)$. True, iff $[\![E_1]\!]$ and $[\![E_2]\!]$ are almost always unequal.
- $\mathsf{SD}(\{(E_1, E'_1), \ldots, (E_k, E'_k)\})$. True, iff the distributions of $[\![(E_1, \ldots, E_k)]\!]$ and $[\![(E'_1, \ldots, E'_k)]\!]$ are indistinguishable.

$\boxed{\textbf{The language of claims (2/2)}}$

- $\mathsf{Ind}(\{E_1, \ldots, E_k\}, \{E'_1, \ldots, E'_l\})$. True, iff the distributions of $[\![(E_1, \ldots, E_k)]\!]$ and $[\![(E'_1, \ldots, E'_l)]\!]$ are *independent*.

This means the indistinguishability of the distributions of

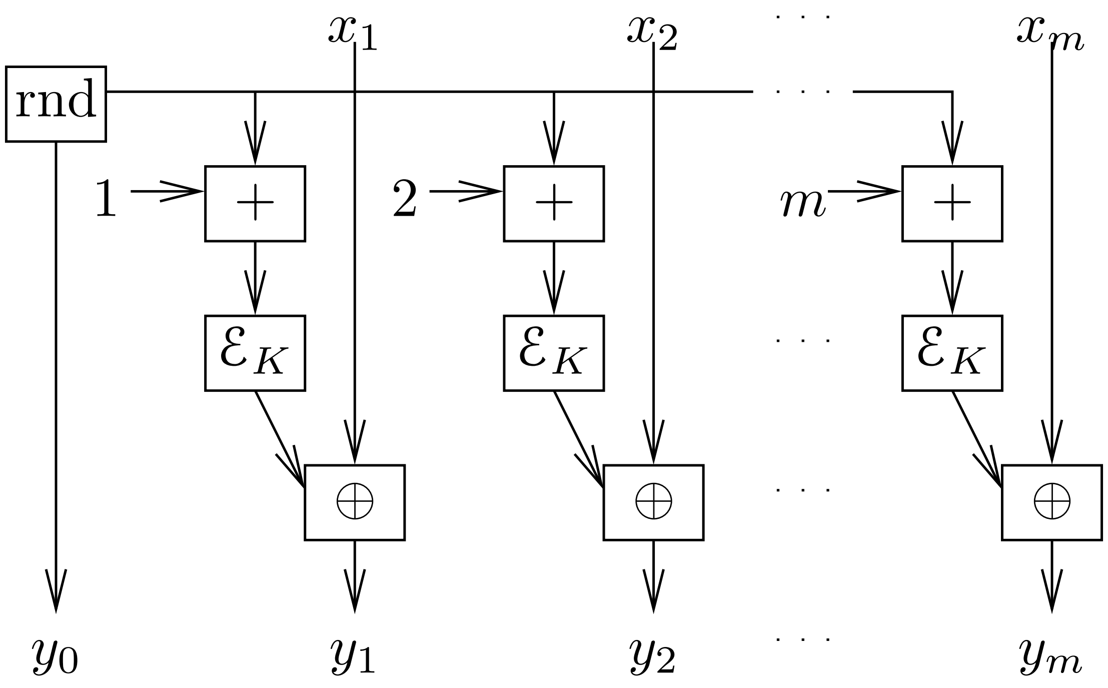
$$[\![((E_1, \ldots, E_k), (E'_1, \ldots, E'_l))]\!] \text{ and } ([\![(E_1, \ldots, E_k)]\!], [\![(E'_1, \ldots, E'_l)]\!]) \ .$$

Example: let I be the result of throwing a 6-sided die.

- "$I \bmod 2$" and "$I \bmod 3$" are independent.
- "$I \bmod 3$" and "$I \bmod 4$" are not independent.

Example — definition

- A block-ciphers' mode of operation — the counter mode.
- $x_1 \cdots x_m$ is the plaintext, divided to blocks of the length of inputs to $\mathcal{E}_k$.
- $y_0 \cdots y_m$ is the ciphertext.
- $\mathcal{E}_k$ is a PRP.
- The result is an encryption scheme that hides the identities of plaintexts and keys.



- Let $r_0, \ldots, r_m$ be independent uniformly distributed bit-strings. We show

$$\mathsf{SD}(\{(x_1, x_1), \ldots, (x_m, x_m), (y_0, r_0), \ldots, (y_m, r_m)\}) \ .$$

$\boxed{\textbf{Example — inputs}}$

- $\mathbf{Inp} = \{x_1, \ldots, x_m, y_0, r_0, \ldots, r_m, 1, \ldots, m\}$.
- The following claims hold for the inputs:
  - $\mathsf{U}(y_0), \mathsf{U}(r_i)$;
  - $\mathsf{C}(i)$;
  - $\not\mathsf{E}(i, j)$, where $i \neq j$;
  - $\mathsf{Ind}(X, Y)$, where
    - $X \subseteq \{x_1, \ldots, x_m, y_0, r_0, \ldots, r_m\}$,
    - $Y \subseteq \{y_0, r_0, \ldots, r_m\}$,
    - $X \cap Y = \emptyset$.

Example — starting the derivation

Rule. $\mathsf{SD}(\{(E_1, E_1), \ldots, (E_k, E_k)\})$.
$\Rightarrow$ Gives us $\mathsf{SD}(\{(x_1, x_1), \ldots, (x_m, x_m)\})$.

Rule. If $|[\![E_1]\!]| = |[\![E_2]\!]|$, then $\mathsf{U}(E_1)$ and $\mathsf{U}(E_2)$ imply $\mathsf{SD}(\{(E_1, E_2)\})$.
$\Rightarrow$ Gives us $\mathsf{SD}(\{(y_0, r_0)\})$.

Rule. $\mathsf{SD}(\{(E_1, F_1), \ldots, (E_k, F_k)\})$, $\mathsf{SD}(\{(E'_1, F'_1), \ldots, (E'_l, F'_l)\})$, $\mathsf{Ind}(\{E_1, \ldots, E_k\}, \{E'_1, \ldots, E'_l\})$ and $\mathsf{Ind}(\{F_1, \ldots, F_k\}, \{F'_1, \ldots, F'_l\})$ imply $\mathsf{SD}(\{(E_1, F_1), \ldots, (E_k, F_k), (E'_1, F'_1), \ldots, (E'_l, F'_l)\})$.
$\Rightarrow$ Gives us $\mathsf{SD}(\{(x_1, x_1), \ldots, (x_m, x_m), (y_0, r_0)\})$.

Some "implicit rules":

- SD, Ind and $\not\mathsf{E}$ are symmetric.
- SD and Ind are monotone.

Assume now, that $\mathsf{SD}(\{(x_1, x_1), \ldots, (x_m, x_m), (y_0, r_0), \ldots, (y_{i-1}, r_{i-1})\})$ has been derived.

$\boxed{\textbf{Example — from } i-1 \textbf{ to } i \textbf{ (1/2)}}$

Rule. $\not\mathsf{E}(E_1, E_2)$ implies $\not\mathsf{E}(E \boxplus E_1, E \boxplus E_2)$, where $\boxplus$ is any *group operation*.

$\Rightarrow$ Gives us $\not\mathsf{E}(y_0 + i, y_0 + j)$ for all $j \in \{1, \ldots, i-1\}$.

Rule. $\mathsf{Ind}(X, \emptyset)$ for all $X \subset \mathbf{Exp}$.

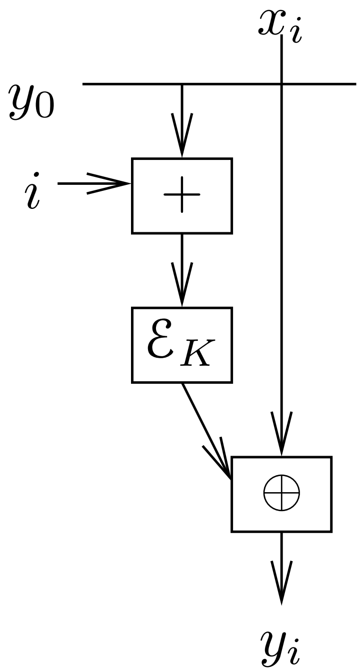
$\Rightarrow$ Gives us $\mathsf{Ind}(\{x_1, \ldots, x_m, y_0, \ldots, y_{i-1}\}, \emptyset)$.

Rule. Let $X, Y \subset \mathbf{Exp}$, $E \in \mathbf{Exp}$, $K \in \mathbf{Keys}$. If

- $\mathsf{Ind}(X, Y)$,
- for all $E' \in X \cup Y \cup \{E\}$:
  - K occurs in E' only as $\{\cdots\}_K$,
  - for all subexpressions $\{E''\}_K$ of E': $\not\mathsf{E}(E, E'')$,

then $\mathsf{Ind}(X \cup \{\{E\}_K\}, Y)$.

$\Rightarrow$ Gives us $\mathsf{Ind}(\{x_1, \ldots, x_m, y_0, \ldots, y_{i-1}\}, \{\{y_0 + i\}_K\})$.



$\boxed{\textbf{Example — from } i-1 \textbf{ to } i \textbf{ (2/2)}}$

Rule. If K occurs in E only as $\{\cdots\}_K$ and if for all subexpressions $\{E''\}_K$ of E we have $\not\mathsf{E}(E, E'')$, then $\mathsf{U}(\{E\}_K)$.

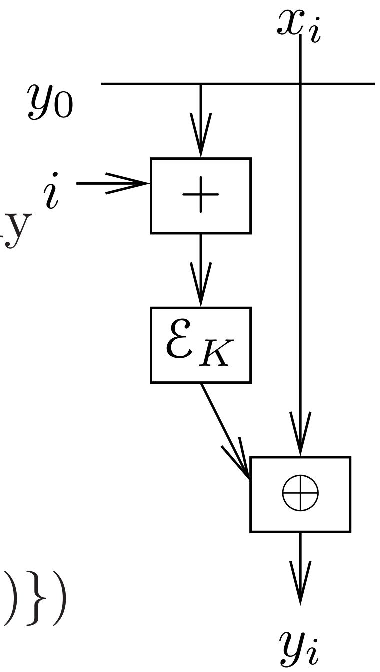
$\Rightarrow$ Gives us $\mathsf{U}(\{y_0 + i\}_K)$.

Rule. $\mathsf{U}(E)$ and $\mathsf{Ind}(\{E\}, \{E'\})$ imply $\mathsf{U}(E \boxplus E')$.

$\Rightarrow$ Gives us $\mathsf{U}(y_i)$.

Rule. $\mathsf{Ind}(X, Y)$, $\mathsf{U}(E)$ and $\mathsf{Ind}(\{E\}, X \cup Y \cup \{E'\})$ imply $\mathsf{Ind}(X \cup \{E \boxplus E'\}, Y)$.

$\Rightarrow$ Gives us $\mathsf{Ind}(\{x_1, \ldots, x_m, y_0, \ldots, y_{i-1}\}, \{y_i\})$.



$\mathsf{U}(y_i)$ and $\mathsf{U}(r_i)$ give $\mathsf{SD}(\{(y_i, r_i)\})$.

$\mathsf{SD}(\{(x_1, x_1), \ldots, (x_m, x_m), (y_0, r_0), \ldots, (y_{i-1}, r_{i-1}), (y_i, r_i)\})$ can now be derived by that long rule. . . .

Conclusions and pointers

- Properties of PRPs can be modeled with methods of static program analysis.
- Finding out, how to analyse cryptographic protocols, if the security requirement of the encryption primitive is being a PRP, may be an interesting research direction.

"Computationally Secure Information Flow", PhD thesis. Chapter 6 covers the material presented in this talk.

Available from `http://www.ut.ee/~peeter_l/research/csif`