Simulating A Large Pseudo-Random Permutation Using Several Smaller Ones

Research Problem Proposal for Japanese-Indian Cooperative Programme:
Security Evaluations and Design of Components and Cryptographic Primitives for RFID and Sensor Networks

Presented at the Indo-Japan meeting, ISI, Kolkata, April 13, 2009 by

Subhamoy Maitra
Applied Statistics Unit
Indian Statistical Institute, Kolkata 700 108, India
Email: subho@isical.ac.in

1 Introduction and Motivation

Conventionally, Pseudo-Random Permutations (PRP) have been used to formally define the security model of a block cipher[3, 5, 6, 7, 2]. Over time, PRP has evolved into an important crypto-primitive. May software stream ciphers, such as RC4, rely on large arrays that store PRP's.

The idea of generating random permutation is presented in [4]. Given a permutation of N elements, $\prod = (\pi_0, \ldots, \pi_{N-1})$, a series of $N-1$ transpositions can produce random permutation as follows:

Algorithm RandPerm
From $i = N-1$ down to 1
$\pi_i \leftrightarrow \pi_{rand(0,i)}$;

where $rand(0, i)$ produces uniformly distributed random integers in the range 0 through i. Now let us briefly outline the proof [8, Page 171] that after $N-1$ many transpositions, the permutation is indeed random. For $N = 1$, the result is trivial. Consider that the algorithm produces random permutations for $N = p-1$ and let $\sum = (\sigma_0, \ldots, \sigma_{p-1})$ be any permutation of $(0, \ldots, p-1)$. Then $P(\prod = \sum) = P(\pi_{p-1} = \sigma_{p-1}) \cdot P[(\pi_0, \ldots, \pi_{p-2}) = (\sigma_0, \ldots, \sigma_{p-2})$, given that $\pi_{p-1} = \sigma_{p-1}]$. By the random transposition in the algorithm, the first probability is $\frac{1}{p}$ and by induction the second probability is $\frac{1}{(p-1)!}$. Thus, $P(\prod = \sum) = \frac{1}{p!}$.

In this work, we would like to develope an architecture to generate extremely large PRP's by combining smaller PRP's. Generating it is only the small PRP's that are stored thereby ensuring a minimalistic overhead on the memory requirement. The idea is to simulate operations on the virtual large permutation by means of actual operations on the smaller permutations.

There exist results in the literature [1] on constructing variable-length PRP's from fixed-length PRP's. But we are not aware of any work on simulating a big virtual permutation using several smaller ones.

2 Building a Large Permutation from Several Smaller Ones

Suppose we have m pseudo-random permutations $\Pi_q[0 \ldots 2^n - 1]$, $1 \leq q \leq m$, each being an array of size 2^n with distinct n-bit elements from $\{0, 1, \ldots, 2^n - 1\}$. Our goal is to combine these arrays with the help of a secret key K into a new array $\Pi[0 \ldots 2^{mn} - 1]$ of size 2^{mn}, containing distinct mn-bit elements from $\{0, 1, \ldots, 2^{mn} - 1\}$.

One possible solution might consist of a two-layer architecture. In the first layer, called *GenVirPerm*, we generate the big virtual permutation by combining the smaller permutations and the secret key. The second layer, called *QueryVirPerm*, produces an entry of the big permutation, given any arbitrary index.

Note that we need not store the entire virtual permutation, rather we can store a state array S that implicitly denotes how the virtual permutation can be combined out of the smaller permutations. When we query with an index i, algorithm *QueryVirPerm* responds with a unique entry v of the virtual permutation.

2.1 Desirable Theoretical Properties of the Big Virtual Permutation

We should design the algorithms *GenVirPerm* and *QueryVirPerm* in such a way that we can prove the following results.

Proposition 1 *Let $QVP(i)$ denote the output produced by algorithm* QueryVirPerm *for an input i. Then the set $\{QVP(0), QVP(1), \ldots, QVP(2^{mn} - 1)\}$ is a permutation of $\{0, 1, \ldots, 2^{mn} - 1\}$.*

Proposition 2 *Suppose for two distinct secret keys K_1 and K_2, the state arrays S_{K_1} and S_{K_2} produced by* GenVirPerm *are distinct with probability $1 - \epsilon$. Then the corresponding virtual permutations QVP_{K_1} and QVP_{K_2} are also distinct with probability $1 - f(\epsilon)$ for some function f.*

References

[1] D. L. Cook, M. Yung and A. Keromytis. Constructing Variable-Length PRPs and SPRPs from Fixed-Length PRPs. Cryptology ePrint Archive, Report 2008/496, year 2008.

[2] S. Even and Y. Mansour. A construction of a cipher from a single pseudorandom permutation. Journal of Cryptology, vol. 10, no. 3, 1997, pages 151-161.

[3] J. B. Kam and G. I. Davida. Structured Design of Substitution-Permutation Encryption Networks. IEEE Trans. Computers, volume 28, no 10, pages 747-753, 1979.

[4] D. E. Knuth. *The Art of Computer Programming*, vol. 3, Addison-Wesley Publishing Company, 1973.

[5] M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. SIAM Journal of Computing, 17:373386, 1988.

[6] M. Naor and O. Reingold. On the Construction of Pseudo-Random Permutations: Luby-Rackoff Revisited. Electronic Colloquium on Computational Complexity (ECCC), 1997.

[7] M. Naor and O. Reingold. Constructing Pseudo-Random Permutations with a Prescribed Structure. Journal of Cryptology, volume 15, year 2002, pages 97-102.

[8] E. M. Reingold, J. Nievergelt and N. Deo. Combinatorial Algorithms: Theory and Practice. Prentice Hall, 1977.