Hypercube Feistel Scheme

Emmanuel Volte[(1)], Jacques Patarin[(2)], Valerie Nachef[(1)]

(1) Université de Cergy-Pontoise (2) Université de Versailles Saint-Quentin

Main idea

The objective of this research project is to build a new cipher scheme with a proved security and a good efficiency. For this, we will use a new kind of Feistel Scheme, that we will call Hypercube Feistel Scheme. This is a natural extention of our works in [3] and [2].

The main idea of this scheme is to use secure Feistel Scheme on small parts of a message, in order to limit the size of the memory for the round functions and in the same time have the same security, and repeat the process in all possible directions, so that the diffusion of the bits is really efficient. Since we can have more than three directions, we use the term of "Hypercube" for the scheme.

Hypercube Feistel Schemes, Notations

We know that we need $\Theta(k^2)$ memory when the message in divided in k parts with a fixed length of n bits, if we use a Feistel Scheme (it is a consequence of [2]). We suppose that $k = u^\tau$ with u and τ integers. And we consider that our message is in a hypercube $\underbrace{u \times u \times u \ldots u}_{\tau \text{ times}}$. If it is not the case, we have to do some padding on the message. We will mix the k parts in τ steps. At each step, we mix $u^{\tau-1}$ groups of u parts with a Feistel Scheme, by considering only one direction of the cube. Each Feistel Scheme cost $\Theta(u^2)$. Since we have τ steps and $u^{\tau-1}$ Feistel Schemes, it will cost us $\Theta(\tau \times u^{\tau+1}) = \Theta(\tau u k)$ instead of $\Theta(k^2)$. So we want to minimize τu when $k = u^\tau$ is given. Since $\tau = \frac{\ln k}{\ln u}$, we have to minimize $\frac{u}{\ln u}$. The minimum of this function is for $u = 3$.

Example

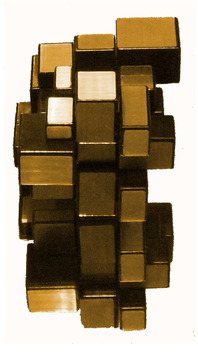
We take $\tau = u = 3$, so the message is divided in 27 parts of n bits. $m = I_0^0 I_0^1 \ldots I_0^{26}$.

I_0^0	I_0^1	I_0^2
I_0^3	I_0^4	I_0^5
I_0^6	I_0^7	I_0^8

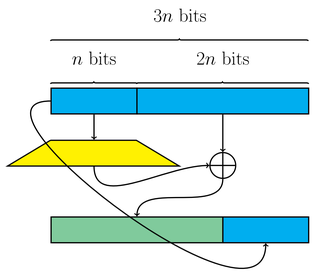
I_0^9	I_0^{10}	I_0^{11}
I_0^{12}	I_0^{13}	I_0^{14}
I_0^{15}	I_0^{16}	I_0^{17}

I_0^{18}	I_0^{19}	I_0^{20}
I_0^{21}	I_0^{22}	I_0^{23}
I_0^{24}	I_0^{25}	I_0^{26}

At the first step (step $s = 0$), we mix independently $I_0^0 I_0^1 I_0^2$, $I_0^3 I_0^4 I_0^5$, …, $I_0^{24} I_0^{25} I_0^{26}$ with a Feistel Scheme. At the end of this step we obtain a new message $m_1 = I_1^0 I_1^1 \ldots I_1^{26}$, then we mix in another direction, this is the second step ($s = 1$). We will mix $I_1^0 I_1^3 I_1^6$, $I_1^1 I_1^4 I_1^7$, $I_1^2 I_1^5 I_1^8$, $I_1^9 I_1^{12} I_1^{15}$, $I_1^{10} I_1^{13} I_1^{16}$,…$I_1^{20} I_1^{23} I_1^{26}$ with some Feistel schemes to obtain $m_2 = I_2^0 \ldots I_2^{26}$. Finally we mix toward the third direction (step $s = 2$) the independent parts $I_2^0 I_2^9 I_2^{18}$, $I_2^1 I_2^{10} I_2^{19}$,…$I_2^8 I_2^{17} I_2^{26}$. After this last step we get our cipher message.



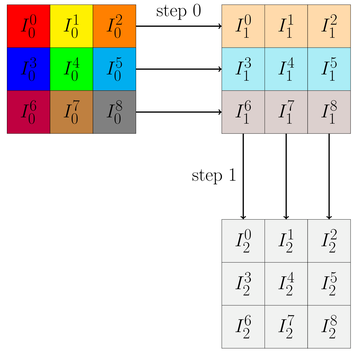
Round one of an Expanded Feistel Scheme with $k = 3$



General case

The message is divided into k parts of n bits : $I_0^0 I_0^1 \ldots I_0^{k-1}$. We suppose that k is a power of 3 : $k = 3^\tau$. We can mix the input I_0^0, I_0^1, …I_0^{k-1} with τ steps.
At the step s, where $0 \leq s \leq \tau - 1$, we will use $3^{\tau-1}$ Feistel schemes with only 3 parts For the i-th Feistel scheme ($0 \leq i < 3^{\tau-1}$) of the step s, we mix the following parts : $I_s^{3^{s+1}\beta+3^s\times 0+\alpha}$, $I_s^{3^{s+1}\beta+3^s+\alpha}$, $I_s^{3^{s+1}\beta+3^s\times 2+\alpha}$ where $\beta = \lfloor \frac{i}{3^s} \rfloor$ and $\alpha = i \bmod 3^s$. See figure bellow for an illustration with $\tau = 2$.

2D Example, $\tau = 2$



Example of a differential attack on a F_3^6 scheme

Here we give the differential path for an attack against a unbalanced Feistel Scheme whith 6 rounds and a message cut into 3 parts. This is the best CPA-1 attack found in [3]. There is a way to automatically generate all the generic attacks for different geometries : two-points attacks, rectangle attacks, square attacks. In order to prove that no better attack can be found, a thorough study of the variance is necessary.

0	0	0	Δ_1
1	0	Δ_1	0
2	•Δ_1	•0	0
3	•Δ_2	Δ_3	•Δ_1
4	0	•0	•Δ_2
5	0	Δ_2	0
6	Δ_2	0	0

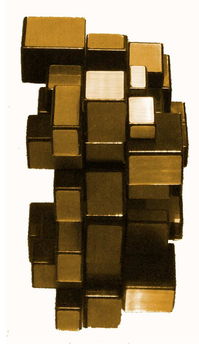
In red all the differential conditions needed to have automatically the others.

Parameters

The AES standard encrypt at most 256 bits. But if we use cipher function as in the CRUNCH hash function [1] we need to encrypt a 1024 bits message.
For n, it is recommended not to have a too small value. I think $n = 3$ is a minimum. $n = 8$ may be a good idea since small cards use a 8-bit processor. But we can also consider $n = 5$ or $n = 10$ because $2 \times 3 \times 5 = 30$ and $3 \times 10 = 30$, so it fits well in a 32 bit processor.
We can also use different kind of Feistel Scheme. And for a Feistel Scheme we can choose different number of turns.
Finally, for a better security, we can do a double mix (so 2τ steps instead of τ).

To do

- Choose parameters with the help of simulations (a software is already written).
- Find differential attacks on the scheme.
- Find security arguments.
- Think about hardware possibilities.



Références

[1] L. Goubin, M. Ivasco, W. Jalby, O. Ly, V. Nachef, J. Patarin, J. Treger, and E. Volte. CRUNCH. Technical report, Submission to NIST, October 2008.

[2] Valérie Nachef, Emmanuel Volte, and Jacques Patarin. Differential Attacks on Generalized Feistel Schemes. *Cryptology ePrint archive : 2011/750 : Listing for 2011*, 2011.

[3] Emmanuel Volte, Valérie Nachef, and Jacques Patarin. Improved Generic Attacks on Unbalanced Feistel Schemes with Expanding Functions. In Kaoru Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 94–111. Springer-Verlag, 2007.