Cryptography - Course Project

Linear Cryptanalysis of FF3-1 and FEA

Final Report

Team Members

- Nirbhay Sharma (B19CSE114)
- Mayank Raj (B19CSE053)

Introduction

Format preserving encryption is an encryption scheme in which the ciphertext is in the same format as the plaintext. It's applications are found in credit cards numbers and encrypting social security number. Techniques such as cycle walking can be used for transfroming blocks ciphers into format preserving ciphers. This method is inefficient when there is a significant size difference between domain size and target size. To tackle this problem self-domain tweakable Feistel ciphers were introduced. **FF1** and **FF3-1** (NIST) and **FEA-1** and **FEA-2** (South Korean standards) are examples of such schemes.

Linear cryptanalysis is a form a cryptanalysis based on finding affine approximations to the action of a cipher. Many attacks have been proposed on the format preserving encryptions. The paper "Linear Cryptanalysis of FF3-1 and FEA" develops linear cryptanlysis based new distingushing and message-recovery attacks on small domain Feistel ciphers.

Linear Cryptanalysis

As we are aware of the two attacks that can be applied on symmetric-key block ciphers, that are linear and differential cryptanalysis, in this part we are only talking about linear cryptanalysis with SPN (Substitution Permutation Network) as one usecase.

Before going into details of Linear cryptanalysis, a little glimpse of SPN is required, so in SPN we have various rounds and in each round we have,

1. **substitution step** - so A 16 bit data block is broken into 4 sub-blocks and then each sub-block is given as an input to the 4×4 s-box which basically maps 4 bits to another 4 bits. In this way substitution step is done.
2. **permutation step** - This step is just outputs a permutation of the output coming from previous substition layer.
3. **key mixing** - This step is just doing xor ($\oplus$) of the keys biths with the data input to this layer.

After some particular number of rounds we get some cipher text which is the encrypted version of input, and for decryption we use the reverse of the network just like we use in case of feistel structure and for s-boxes we have an inverse mapping to be used at the time of decryption.

So now let's have a look at linear cryptanalysis, Linear cryptanalysis was first introduced by Matsui at EUROCRYPT 93 as an attack on Data Encryption Scheme (DES) and later it became a successfull attack not only on DES but also on some other block ciphers as well, It is a known plaintext attack, It tries to take advantage of linear relationship expressions that may exists among bits of plain text and cipher text. the definition of linearity is to get a combination of bits whose xor's become zero. consider the input to be represented as $X = (X_1, X_2, \ldots, X_j)$ and output as $Y = (Y_1, Y_2, \ldots, Y_j)$ so our task is to find some linear relationship as follows

$$\alpha_1 X_1 \oplus \alpha_2 X_2 \oplus \ldots \oplus \alpha_j X_j \oplus \beta_1 Y_1 \oplus \beta_2 Y_2 \oplus \ldots \oplus \beta_j Y_j = 0$$

where $\alpha_i, \beta_i \in \{0, 1\}$

the aim of linear cryptanalysis is to find the expressions of such form with some high probability, consider the fact that if we take two values u and v randomly and put their xor to zero $u \oplus v = 0$, will have exactly $\frac{1}{2}$ probability so we want the probability to be $\frac{1}{2} + \epsilon$, where ϵ is the bias and this is what which is exploited in linear cryptanalysis for which ever expression we got higher bias, it means we have a high probability for that expression to occur and hence this can be exploited to get some information about keys etc.

To find the bias for an expression $Pr[X_1 \oplus X_2 \oplus \ldots \oplus X_j = 0]$, we have a Piling up lemma which say

$$Pr[X_1 \oplus X_2 \oplus \ldots \oplus X_j = 0] = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} \epsilon_i$$

where $\epsilon_i = Pr[X_i = 0] - \frac{1}{2}$ and

$\epsilon_{1,2,\ldots,n} = 2^{n-1} \prod_{i=1}^{n} \epsilon_i$, $\epsilon_{1,2,\ldots,n}$ is the overall bias for the expression $Pr[X_1 \oplus X_2 \oplus \ldots \oplus X_j = 0]$

so using the above expression, we can generate various probabilities and their probabilities for satisfying the criteria and we construct a linear approximation table which tells us which expressions are matching with which probability.

so for the SPN example we have s-box of 4×4 so the expression would be,

$$\alpha_1 X_1 \oplus \alpha_2 X_2 \oplus \alpha_3 X_3 \oplus \alpha_4 X_4 \oplus \beta_1 Y_1 \oplus \beta_2 Y_2 \oplus \beta_3 Y_3 \oplus \beta_4 Y_4 = 0 \qquad -eq-1$$

notice that we have two sequences here $\alpha_1\alpha_2\alpha_3\alpha_4$, $\beta_1\beta_2\beta_3\beta_4$ and linear approximation table is exactly utilizing this, it takes all possible combinations of these two sequences for input and output and accordingly generate the values by which expression in eq-1 can be satisfied. a sample linear approximation table is shown below

Input Sum	Output Sum 0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	−2	−2	0	0	−2	+6	+2	+2	0	0	+2	+2	0	0
2	0	0	−2	−2	0	0	−2	−2	0	0	+2	+2	0	0	−6	+2
3	0	0	0	0	0	0	0	0	+2	−6	−2	−2	+2	+2	−2	−2
4	0	+2	0	−2	−2	−4	−2	0	0	−2	0	+2	+2	−4	+2	0
5	0	−2	−2	0	−2	0	+4	+2	−2	0	−4	+2	0	−2	−2	0
6	0	+2	−2	+4	+2	0	0	+2	0	−2	+2	+4	−2	0	0	−2
7	0	−2	0	+2	+2	−4	+2	0	−2	0	+2	0	+4	+2	0	+2
8	0	0	0	0	0	0	0	0	−2	+2	+2	−2	+2	−2	−2	−6
9	0	0	−2	−2	0	0	−2	−2	−4	0	−2	+2	0	+4	+2	−2
A	0	+4	−2	+2	−4	0	+2	−2	+2	+2	0	0	+2	+2	0	0
B	0	+4	0	−4	+4	0	+4	0	0	0	0	0	0	0	0	0
C	0	−2	+4	−2	−2	0	+2	0	+2	0	+2	+4	0	+2	0	−2
D	0	+2	+2	0	−2	+4	0	+2	−4	−2	+2	0	+2	0	0	+2
E	0	+2	+2	0	−2	−4	0	+2	−2	0	0	−2	−4	+2	−2	0
F	0	−2	−4	−2	−2	0	+2	0	0	−2	+4	−2	−2	0	+2	0

In this manner linear cryptanalysis tries to find the linear relations among the bits of input and output and higher probability expressions are exploited to further crack some of the key bits etc.

Format Preserving encryption Schemes

Encryption schemes where input and output have the same format are known as Format preserving encryptions. General techniques of FPE are :

1. **Prefix Cipher**:

   For a domain d={0,1...t-1}

   We have a n-bit block cipher $E_k(.)$ with domain $N = 2^n \ \geq \ t$

   Permuted [0,1,...t-1] = Ordering [$E_k(0)$,$E_k(1)$,....$E_k(t-1)$]

   This method computably reasonable for t<2^{30}

2. **Cycle walking**

   Domain d={0,1.....t-1}

   Take a $E_k(.)$ with domain N such that $N \geq t$

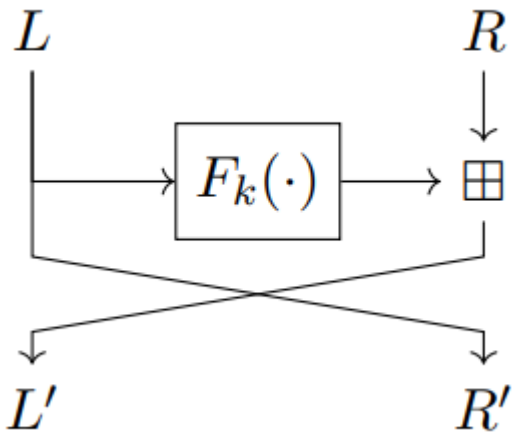
   The Map $x \in d$ to $E_k(\ldots E_k(..E_k(x)) = c$ such that $c \in d$

   Problem: Too many block cipher invocations if d is not dense in block cipher domain N

3. **Generalised Feistel**

If we have a message size of t

We choose two integers a and b such that $a, b \geq t$ with $a \geq L$ and $b \geq R$ and perform



Minimum 3 rounds should be done

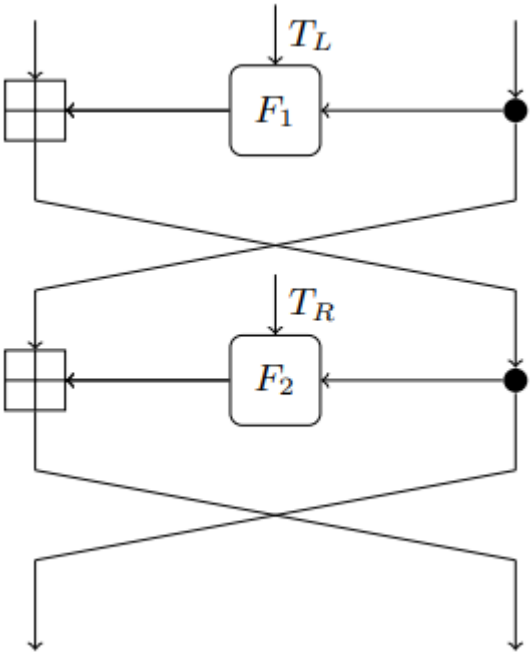
Construction FFX (FF1,FF2,FF3)

These FPEs were standardised by NIST. FF1 and FF2 uses atleast 10 round of Feistel and each round has one invocation to AES.

FF3 uses only 8 rounds and thus is faster than FF1/FF2. It consist of two components:

- Internal block cipher: It is used to encrypt data while preserving the format.
- Handling long messages: Due to this maximum input size get fixed.

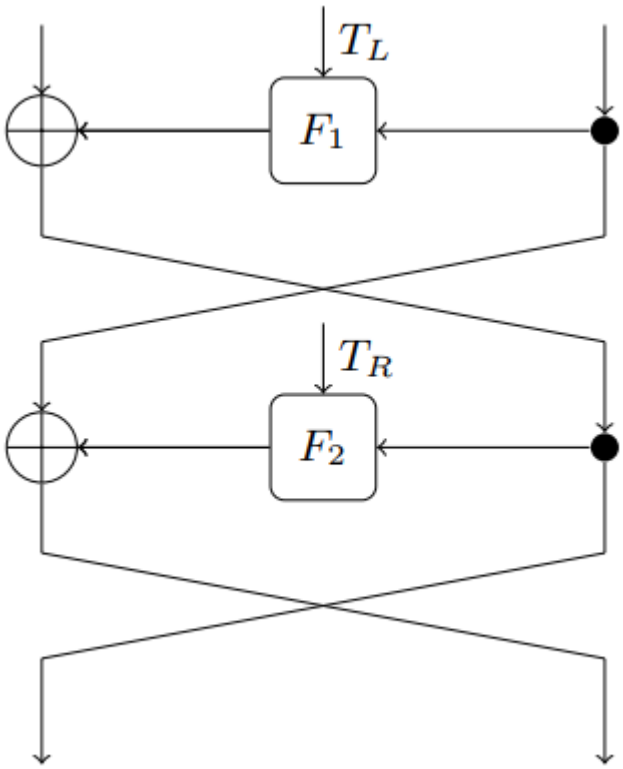
FF3-1 is an eight-round Feistel cipher over $Z/NZ \oplus Z/NZ$. The round functions F1, F2, . . . are defined as truncations of the AES with the round tweak and a unique round counter as the input; the details are not important for this work as these functions will be modelled as uniform random. The tweaks TL and TR are bitstrings of length 28.



Structure of two round FF3-1

Structural details of FEA-X

2 rounds of tweakable feistel structure are shown in the figure below:



the structural details of the format preserving ciphers (FEA-1, FEA-2) are as follows

- FEA-1 and FEA-2 has rounds depending upon the key size as shown in table below

Key Length	FEA-1	FEA-2
128	12	18
192	14	21
256	16	24

Linear approximations

Consider a function F:$F_2^n \rightarrow F_2^m$, and the conversion is dependent on some key K, and so linear discriminators are based on linear approximators with large coorrelation so mathematically a linear approximation F is is defined by a pair of mask (u1,u2) $\in F_2^n \oplus F_2^m$ space and is equal to

$$C^F_{u1,u2} = 2Pr[u_1^T F(x) = u_2^T x] - 1$$

where the expression is bias terms for F and is finally equal to

$$\frac{1}{2^n} \sum_{x \in F_2^n} (-1)^{u_1^T F(x) + u_2^T x}$$

the correlations for a uniform random function has mean 0 and standard deviation around $2^{-n/2}$ so if the standard deviation of the distribution is deviating from the above then the distinguisher can be constructed and the result is compared using some threshold t. Consider a function F: G $\rightarrow$ H, where G and H are finite Abelian groups, a linear approximator corresponds to pair of characters (ψ_1, ψ_2) of H and G is represented as

$$C^F_{\psi_1,\psi_2} = \frac{1}{|G|}\sum_{x\in G}\overline{\psi_1(F(x))}\psi_2(x)$$

where $\overline{\psi_1}$ denotes the complex conjugate of the ψ_1 and basically ψ_1 corresponds to a vector u $\in F_2^m$ and the important property of these group characters are that they are orthogonal in nature, mathematically speaking they have the following property

$$\sum_{x\in G}\overline{\chi(x)}\psi(x) = |G|$$

when $\chi = \psi$ and 0 otherwise

now a sequence of such function is constructed as $F_1, F_2, \ldots, F_l$ and the correlation between them can easily calculated using piling up principle of linear distinguisher (explained above under the heading Linear Distinguishers) and hence following from that the collective correlation can be found using below formulae

$$C^F_{\psi_1,\psi_{l+1}} = \prod_{i=1}^{l} C^{F_i}_{\psi_i,\psi_{i+1}}$$

and once the correlation is find output the breaking is simple and we can take the advantage of this bias term to break the cipher.

χ^2 distinguisher

The distinguishers on FEA-1, FEA-2 and FF3-1 are proposed on the basis of Pearson's χ^2-test for goodness-of-fit between distributions. The χ^2 distinguishers are use to distinguish non uniform distributions in cryptanalysis when exact knowledge about the distributions are missing.

A collection of linear approximations such that the set of pairs of input and output masks is a vector space can be called as multidimensional linear approximation.This can be generalised to arbitary groups given the set of input and output characters is a group under pointwise multiplication. The existence of multidimensional linear approximation implies that a particular probablity distribution related to the ciphertext is highly non-uniform and therefore Pearson's χ^2-test can used to verify this property which can termed as a distinguisher.

The relation between χ^2 2-distinguishers and multidimensional linear approximations is due to the link between correlations and the Fourier transformation of the probability distribution of the active parts of the input and output state. Pearson's χ^2-statistic is used for estimating goodnesds-of-fit between empirical probablity distribution $^p : S \rightarrow [0, 1]$ and uniform probablity distribution on S. The χ^2-statistic with q samples satisfies

$\chi^2/q = M||\hat{p} - 1/M||_2^2$

here, $||\ \ ||_2^2$ represents euclidean norm, $M = |S|$ and $1(x) = 1$ for all $x \epsilon S$.

As $q \to \infty$ the estimated distribution $\hat{p}$ tends to the true distribution p and χ^2/q tends to $M||p - 1/M||_2^2$. So, if the distribution comes to be uniform as $q \to \infty$, χ^2/q tends to 0.

Theorem: Let $F : G \to H$ be a function between finite Abelian groups G and H. Let Z be a subgroup of the group $H \oplus G$ and let Z^0 be the group of characters of $H \oplus G$ with kernel Z. If x is a uniform random variable on G, then

$$Pr[(F(x), x) \equiv z mod Z] = \frac{1}{|Z^0|} \sum_{\psi \epsilon Z^0} C^F_{\psi_H, \overline{\psi_G}} \psi(z)$$

ψ_H is the restriction of ψ to H and ψ_G is similarly for G.

This theorem, (proof can be found at [4]) can be applied to the multidimensional linear approximations. For FEA-1 and FEA-2, Z can be taken as the orthogonal complement of the F_2-vector space consisting of the masks in the multidimensional linear approximation. For FEA-1 and FF3-1, the right half of the plaintext is fixed and reduction modulo Z is equivalent to taking the difference of the left half of the ciphertext and the plaintext.

If D is the half-domain of the cipher and T be the space of half-tweaks T_R then we can deduce that

$$H = D \oplus D, G = D \oplus T \text{ and } Z = (y_L, y_R, x_L, T_R) \epsilon D \oplus D \oplus D \oplus T | y_L - x_L = 0$$

For FEA-2 the full plaintext will be fixed, so $G = T$. In this case reduction modulo Z will cooresponding to truncating the ciphertext to its left half.

Lets $c(\psi)$ denotes the correlation of the approximation coressponding to $\psi \epsilon Z^0$. Using the previous theorem we can say that for all the 3 ciphers:

$$Pr[F(x), x) \equiv z \ mod \ Z] = \frac{1}{|Z^0|} \sum_{\psi \epsilon Z^0} c(\psi) \psi(z)$$

x is uniform random on input domain and F is the mapping to the ciphertext.

As the queries q increases in χ^2 statistic, the empirical distribution reaches p and χ^2/q statistic approaches the value:

$$\begin{aligned} M||p - 1/M||_2^2 &= ||c - \delta_1||_2^2 \\ &= \sum_{\psi \neq 1} |c(\psi)|^2 \end{aligned}$$

The χ^2 statistic can be interpreted as an alternative method to estimate the sum of the squared correlations $|c(\psi)|^2$ for $\psi \epsilon Z^0$ with $\psi \neq 1$. Using this the data complexity of χ^2 distinguisher for r rounds of FEA-1 and FF3-1 is of the order $N^{r/2-1.5}$. For FEA-2 the data complexity is of the order $N^{r/3-1.5}$. If we consider smallers choices for the group Z, we can set up χ^2-distinguishers even if the part of ciphertext is available.

Message recovery attacks

The message recovery attack using χ^2 distinguisher has the following setup, as shown in the previous section of χ^2 distinguisher, since the right half is fixed for input so the half domain is denoted by D and tweaks space is denoted by T and so H which is a ciphertest space is denoted by $D \oplus D$ since both the halfs are not constant and the G which is input space represented by $D \oplus T$ for FEA-1 and FF3-1 for which right half is fixed and G is T in case of FEA-2 where both the halfs of plaintext are kept fixed

attack for FEA-1 AND FF3-1

left half recovery

In this scenario the right half is fixed for the input messages and hence the χ^2 distinguisher is represented as

$$Z = \{(y_l, y_r, T_r) \in D \oplus D \oplus T | y_l = 0\}$$

so the core idea behind the attack is that since the right part is fixed and so the change in plain text affect only the left half of the ciphertext for tweaks T_r so consider $c_1(x)$ as the correlation corresponding to character ψ when plain text is (x_l, x_r) and the correlation is $c_2(x)$ for the input (x'_l, x_r) (right half is fixed) so as explained in the linear approximator section the correlation can be computed in combine form using piling up principle (from linear distinguisher) and hence the two function are same upto the subtraction of constant $\triangle = x_l - x'_l$ in the first round and hence

$$c_2(\psi) = \psi_D(\triangle) c_1(\psi)$$

where ψ_D is computed over the half domain space D, so the distributions of probabilities are shifted by a distance $\triangle$ so the attacks begins by estimating the distribution of the left half of the ciphertext twice, one for the message (x_l, x_r) with fixed tweak T_l (alternate tweaks) and another one for the arbitrary message (x'_l, x_r) (notice that the right half is still same) with the same alternate fixed tweaks T_l and then we need to compute the statistic

$$r(\triangle_g) = qN/4 ||p_1 - p_g||_2^2$$

where p_1 and p_g are the distributions that differ by $\triangle$ now all these values are ranked in ascending order for all $\triangle_g \in D$ and the value at the top of the list is a good candidate for $\triangle$

right half recovery

The idea behind the right half recovery is the exploitation of the following property, if we apply the left-half recovery method to any arbitrary (x_l, x_r) and (x'_l, x'_r)

References

1. https://www.isical.ac.in/~mridul/workshop/Slides/Somitra.pdf
2. https://ioactive.com/wp-content/uploads/2015/07/ldc_tutorial.pdf
3. https://pypi.org/project/ff3/
4. https://eprint.iacr.org/2021/815.pdf