Chapter 12
Linear Crypt Analysis

February 15, 2010

12

Similar to Differential Crypt Analysis (DCA) Linear Crypt Analysis (LCA) is a technique for attacking encryption algorithms of many rounds with sub-keys entering linearly in each round. Like DCA it exploits non-uniformities in the structure of a round reducing them to linear expressions of unequal probabilities. A sufficiently large number of input/output pairs (known or chosen) can reveal the effect of these inequalities and enable conclusions about the sub-keys to be drawn.

LCA was first described by Mitsura Matsui in 1992, (Computer and Information Systems Laboratory Mitsubishi Electric Corporation, Japan). Matsui used LCA to describe an attack on DES. The discussion that follows here is directly taken from his original paper, and uses his notation.

12.1

The general idea of LCA is to find a linear approximation to a cryptographic algorithm, which holds with a certain probability $p \neq \frac{1}{2}$, of the form

$$\mathrm{P}[\mathrm{i}_1 i_2 ... i_r] \oplus K[l_1 l_2 \dots l_s] = C[j_1 j_2 \dots j_t] \tag{0.1}$$

Here P is the plain-text, C is the cypher-text, and K is the key (usually a subset of the sub-keys of a multi-round cypher).

The notation $P[i_1 i_2 \dots i_r]$ means the XOR sum of bits $i_1 i_2 ... i_r$ of P (that is $P[i_1] \oplus P[i_2] \dots \oplus P[i_r] = 0$ *or* 1).

If such an expression as equation **??** can be found then using N known plain-texts and cypher-text pairs we would expect equation **??** to be satisfied on Np occasions ($\pm 2.58\sqrt{Np(1-p)}$) for one percent confidence, assuming a binomial distribution).

This enables one to decide, given sufficiently large N, what is the value of $K[\,]$ given $P[\,]$ and $C[\,]$. For example, if $p > \frac{1}{2}$ so that equation **??** is on average true, but $P[\,]$ and $C[\,]$ are found to be on average different (example: $P[\,] = 1$ and $C[\,] = 0$) then we can plausibly conclude that $K[\,] = 1$. This would give us a relationship between key-bits effectively reducing the key space by one bit. We shall see how we can estimate many key bits later.

12.2

The method for producing approximations like equation **??** is to find such approximations for single rounds and then combine all the rounds together in such a way as to eliminate all intermediate data values, leaving only $P[\,]$, $C[\,]$ and key bits (from some or all of the rounds).

As an example consider the DES $f()$ function. Permuting bits does not affect the XOR sum of their values. The only non-linear operations which need to be linearly approximated are the S-boxes. Consider S_1, and the input bit corresponding to 10_{HEX}. (See table 11.1 of Chapter 11) The XOR of the four bits of the output of S_1 (corresponding to OF_{HEX}) is equal to the input bit only for fourteen out of the sixty-four possible input patterns for S_1. But the input to S_1 is the XOR of a data bit $P[i]$ and a key bit $K[i]$; so if $P[i] = C[j_1 j_2 j_3 j_4]$ (where $C[j_1 j_2 j_3 j_4]$ means the XOR of the four bit output of $S1$) we may assume $K[i] = 1$ with high probability. (It can be shown that for high confidence, if the probability of an expression is p we need Order$(\frac{1}{2} - p)^{-2}$ observations, approximately if p is very small.)

Of course in DES, and other algorithms, one must keep track of bits as they are moved around by permutations and expansions in functions such as DES $f()$. But the essential point is the fact that the S-boxes allow certain linear approximations with probabilities sufficiently different from $\frac{32}{64} = \frac{1}{2}$.

12.3

Putting together different rounds is done by assuming that the linear approximations of the individual rounds are simultaneously all true. If they are, then the linear equations for the relevant rounds may be XORed together to form a single linear approximation for the whole algorithm; which is true with probability

$$(1)/(2) + 2^{n-1} \prod_{i=1}^{n} (p_i - \tfrac{1}{2}) \tag{0.2}$$

if there are n stages and the probability of the linear approximation for the i^{th} stage being true is p_i. (This expression is the probability of the XOR sum of n random variables being zero, if their individual probabilities of being zero are p_i. It is easily proven by induction on n.)

We use as an example five round DES. Note that the identification of the bits is as in Matsui's paper: namely the least significant bit is bit 0, the most significant in a 32-bit word is then bit 31. (DES is other way round). Note also that the example takes into account the E(expansion) and P(permutation) of the DES $f - function$; but ignores the initial and final permutations of the complete algorithm. The linear approximations are:

Probability	Round		S-box
$\frac{22}{64}$	Round 1	$P_H[15] + X2[15] + P_L[27, 28, 30, 31] = K1[42, 43, 45, 46]$	S1
$\frac{12}{64}$	Round 2	$P_L[7, 18, 24, 29] + X3[7, 18, 24, 29] + X2[15] = K2[22]$	S5
$\frac{12}{64}$	Round 4	$X3[7, 18, 24, 29] + CL[7, 18, 24, 29] + X4[15] = K4[22]$	S5
$\frac{22}{64}$	Round 5	$C_H[15] + X4[15] + CL[27, 28, 30, 31] = K5[42, 43, 45, 46]$	S1

P_H, C_H are the high (left) half of P, C; P_L, C_L the low (right) halves. X_i is the input to $f()$ at round i.

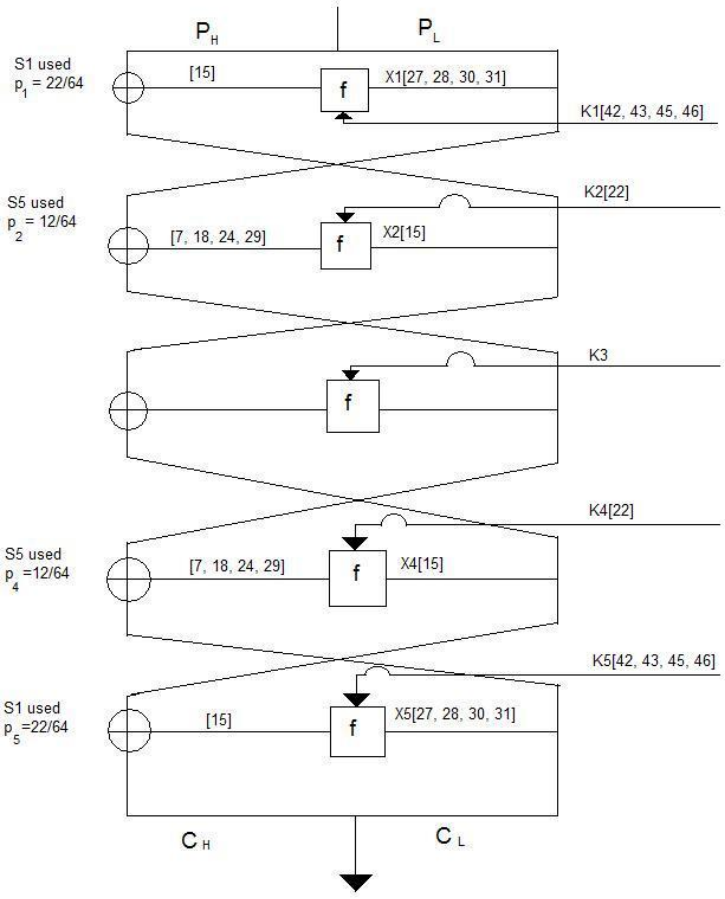
The input/output masks are for $S1$ (011011,0100) and for $S5$ (010000,1111) and the associated probabilities that the equations hold are $\frac{22}{64}$ and $\frac{12}{64}$ respectively as can be deduced from Table 6 of DES.

If all these expressions are true simultaneously we can XOR them together to get a linear approximation for five-round DES:

$$\mathrm{P}_H[15]+P_L[7, 18, 24, 27, 28, 29, 30, 31]+CH[15]+CL[7, 18, 24, 27, 28, 29, 30, 31] = K1[42, 43, 45, 46] + K2[22] + K4[22] + K5[42, 43, 45, 46] \tag{0.3}$$

This holds with probability of 0.519, using expansion in equation **??**.

12.3.1 5-Round DES - linear Approximation



12.4

Continuing with the example of five round DES we note that if we are successful (using $(\frac{1}{2}$ - $0.519)^{-2}) \sim 2770$ known plain-texts) we find out a linear relation between key bits; namely the right hand side of (3) = 0 *or* 1. In DES these are sub-key bits, easily related back to the user key; but still, effectively only one key-bit has been found.

However we may find more key-bits by reforming (3) as follows: do not approximate the last round but rather use the $f()$ function with the known C_L bits, and all possible values for the six relevant key-bits of $K5$ which affect $f(CL, K5)[15]$.

Equation **??** becomes:

$$\mathrm{P}_H[15] + P_L[7, 18, 24, 27, 28, 29, 30, 31] + CH[15] + CL[7, 18, 24, 29] + f(CL, K5)[15] = K1[42, 43, 45, 46] + K2[22] + K4[22] \quad (0.4)$$

If the correct key-bits at K5 are guessed, namely those that correspond to the observed CH[15], then the left hand side of equation **??** will have a more pronounced bias away from $\frac{N}{2}$ after N observations. Thus 64 tables are required, in which to count the number (T) of times the left hand side of **??** is zero, one table for each value of K5. The value of K5 which maximises $|T - \frac{N}{2}|$, the difference between T and $\frac{N}{2}$, is chosen. $K[\ ]$ is then estimated as before, using the value of p for four (in general $(n-1)$) rounds; p being either $< \frac{1}{2}$ or $> \frac{1}{2}$.

The theory shows that this method requires more tests, that is a larger N_i but it gives seven key bits instead of one.

Furthermore DES decrypts as well as encrypts. Given plain-text/cypher-text pairs it is possible to perform the LCA backwards and (in the case of five rounds) find seven further key-bits, six of which are from K1, which is the last decryption round key. In all we now have fourteen out of the fifty-six key-bits and it is easy to find the remainder (example: by exhaustive search).

LCA is a powerful tool for attacking cyphers which contain non-linear functions which have certain non-random characteristics. The linearisation of these functions (such as S-boxes) is avoided or discovered usually by computer search. A good cypher will ensure that the probability of such linear relationships is almost random (that is $\frac{1}{2}$) and any proposed design should be tested exhaustively for this. If this is so, and if the number of rounds is large; then N (the number of tests that an attacker needs to perform) will be larger than that required for an exhaustive search of the key-space.