Key-Data-Paired Differential Cryptanalysis

Luke Kenneth Casson Leighton

May 27, 2008

Abstract

Traditional Block cipher Differential Cryptanalysis finds relationships between one single bit of a key and a single bit of output, in one fixed location in the output block. The new technique described in this paper, which is being named Key-Data-Paired Differential Cryptanalysis, locates relationships between groups of bits in keys and groups of bits in output blocks (instead of just single bits in fixed locations). Using even as little as 512 blocks of known plaintext on a 128-bit block cipher (for example Rijndael-128), the correlations between groups are sufficient to derive statistical probabilities for individual bits in the key being 0 or 1. If there are no correlations, we anticipate these probabilities to indicate correct guesses for each bit to average 64 bits. Whilst only a few runs so far have been performed (taking 6 hours each), what is actually being consistently shown is about 76 bits being guessed, with one run going as high as 81 bits. Research is underway to ascertain whether those statistical probabilities can be used in a feedback loop to gain better accuracy, by generating weighted keys based on the previous probabilities.

1 Introduction

Block cipher design is based on the assumption that there will be no inter-relationships detectable between any bits of the key and any bits of the encrypted data, no matter what the input data is, and no matter how much input data is given. Cryptanalysis challenges this assumption. The analysis which is being called Key-Data-Paired Differential Analysis makes the assumption that there are inter-relationships between groups of bits in a key and groups of bits in the data.

Some rather unusual but perfectly reasonable statistical analysis is used to spot such inter-relationships, and, using a known plaintext attack, derive probabilities for certain bits of the key. As of yet, it is not possible to determine which bits have been correctly guessed and which have not: obviously, it is only by means of really knowing the key that the guesses can be confirmed as correct in the first place. Refinement of the rather crude process being used at present may yield better results in the future.

The analysis has startling implications for the design of block ciphers, not least is that the whole principle on which block ciphers are founded may be flawed. The key principle of block cipher design is that it is mathematically challenging to locate inter-relationships between groups of key bits and data bits, in the simple function $O = E(K, D)$. Some cryptographers have designed block ciphers with Tweaks, where there is additional feedback into the algorithm ($O = E(K, D, Tweak)$ or $O = Etweak(K, D)$) that would make the type of attack described in this paper ineffective. However, in its simplest form, this paper demonstrates that ECB mode ciphers, designed on the traditional $O = E(K, D)$ design, is likely to be fundamentally flawed.

2 Deriving the 3D Key-Data Histogram

The first step is as follows:

For any 128-bit key-data pair, first perform an encrypt $Ob = E(K, D)$. This will be called the baseline (for a given key-data pair). Then, perform 16384 (128 by 128) encrypts, by changing one bit of the key and one bit of the data in each encrypt $Onm = E(Kxor(1 << n), Dxor(1 << m))$. XOR the baseline Ob over each of the 16384 bit-tweaked encrypts. Counting the number of bits in each XORed result, we have a measure of how many bits have not changed in the output, for any given key-bit-tweak and input-data-bit-tweak. If the number of bits unchanged is greater than an arbitrarily chosen value (e.g. 80 bits) then the 3D histogram bucket for that key-data bit pair, n and m, is incremented.

The motivation behind deriving this histogram is to determine whether it could be shown that there were correlations between blocks with similar output, and the keys and data blocks that generated them. For perfect block ciphers, even if the number of similar bits was very large (e.g. 120 bits), there should still be absolutely no correlation in the 3D histogram. Any anomalies in the 3D histogram would be indicative of correlations in single bit-changes in the key or the data.

2.1 Searching for anomalies

The second stage involves analysing the histograms. Displaying the histogram as-is, with gnuplot, shows nothing that can be observed by eye: the 3D histogram looks entirely random. Utilising gnuplot's grid3d feature, however, starts to show strange anomalies, and these anomalies do not disappear even when plotting 256000 blocks (comprising 256000 * 128 * 128 encrypts, in total - some 4 billion blocks). However, it's still not enough to discern any useful information. As hinted at by the grid3d functionality of gnuplot, taking the average of 16 by 16 grids in the histogram, however, begins to shed some light. Performing a sort, by the height of the histograms on one row or by one column, on the averaged grid, immediately shows suspicious correlations. In several tests, sorting the histogram by row (e.g. a row representing key-bits 16-31) shows one of the columns are also in roughly height order. In a perfect cryptographic algorithm, this should simply not happen. The implications are simple: groups of bits of the key and the data have correlations between other groups. Although this is something that we'd expect of a block cipher, otherwise it would not be possible to perform a decrypt, the extent of the inter-relationship should not be sufficient to be able to make guesses about bits of the key, but this is exactly what is possible.

3 Key-bit probability derivation from the 3D Histogram

The third stage is what is, at present, the most time-consuming and also the most crude of the steps in the analysis process. The averaged-3D-histogram analysis, above, alerts us to blocks of key-data bit-groups which may have inter-relationships, for example, key bits 0-15 and data bits 32-47 may be inter-related with key bits 112-127 and data bits 64-71. A crude simplistic approach to testing whether there is such an inter-relationship can be performed by taking each of the 512 initial key and encrypted data baseline pairs, we can simply go through XORing key bit 0, key bit 112, data bit 32 and data bit 64, for all 512 key-data pairs, and we call this total $Kp(n)$ where n is the key bit being analysed (in this case, $n = 0$). Next we XOR key bit 1, key bit 113, data bit 33 and data bit 65: this we call $Kp(1)$ etc. up to $Kp(15)$. Adding up these XOR results, and selecting yet more blocks which have been shown to have inter-relationships, we would anticipate the average of each $Kp(n)$ to be 256, if we have tested 512 key data pairs. No matter how many blocks we pick, assuming that the keys are picked entirely randomly, and assuming that the encrypted data has no correlations, then no matter what groups of key bits and data bits we choose to XOR together, the average of each $Kp(n)$ for all key bits 0 to 127 should be half the number of key-data pairs chosen.

For Rijndael-128, with only 512 key-data pairs, it turns out that this is not the case. Of the few runs performed so far, the number of $Kp(n)$ indicators above-average which turn out to correspond to the original key bits has varied between 76 and even once as high as 81. One would expect the number of $Kp(n)$ indicators that are above-average to be around 64 +/- a few - but not consistently above 76. From this startling - and crude - analysis - it is possible to conclude that there are indeed correlations between key-data groupings.

4 Key-bit probability feedback

A refinement of the approach is to use the previous values of $Kp(n)$ to make weighted keys. So, if the previous value of $Kp(0)$ was 384, and 512 key-data blocks were previously generated, then in the generation of the next set of 512 randomly-generated keys to be used, we ensure that bit 0 of all the keys has probability 0.75 that it will be 1 and probability 0.25 that it will be 0.

Unfortunately, when calculating the next set of $Kp(n)$ indicators, they turn out to not average out to 256 (half the number of key-data blocks) but to some other much higher value. Research needs to be undertaken to determine, mathematically, what this value should be.

It is anticipated that it may be possible to repeat the analysis, producing continued refinement of the probabilities in a positive feedback cycle. So, whilst the initial run could turn out to correctly guess 76 out of 128 bits of the key, the second run could possibly guess 80, the third 85 and so on. However, it may equally turn out to be the case that weighted keys skew the results in a negative feedback cycle rather than a positive one, and that the best weighting for each key bit is 0.5 - i.e. each key should be entirely random.

5 Set Theory on 3D histograms

There is an alternative approach to consider, which is the application of set theory to the histograms. Averaging the histogram values is presently subdividing the 128x128 grid into 64 cells, selecting key bits 0-15 and data bits 0-15 for the first cell, key bits 16-31 and data bits 0-15 for the second, and so on. Then, the calculation of $Kp(n)$ statistical probabilities proceeds.

There is an alternative approach to the calculation of the histograms which produces a different set, with its own correlations: calculating the averages by selecting 64 cells with different values from the histogram, for example by selecting key bits 0,8,16...112 and data bits 0,8,16...112 for the first cell, key bits 1,9,17...113 etc. The selection process needs research and confirmation, however, it should be clear that the number of 64-cell sets that can be picked is very extensive.

It is anticipated that combined application of several 64-cell-average sets, which overlap appropriately, will further refine the accuracy of the $Kp(n)$ statistical analysis.

6 DES

DES - the Data Encryption Standard - surprisingly came up with results close to random. i.e. the number of bits guessed in the key were consistently between 28 and 31, where the average number of bits to be expected is half of 56. Perhaps, given that IBM designed DES to be resistant to differential cryptanalysis, it should not come as a so much as a surprise. Further research is therefore needed, perhaps along the lines of 2nd order differential cryptanalysis, or just a more refined approach than the initial crude one taken. It is worth emphasising that the averaged 3D histograms are definitely skewed: there are obvious slopes indicating correlations between key and data bit ranges. Thus, it is anticipated that alternative approaches will give better results than at present.