CS406 : Project Report

Rahul Chunduru (160050072) & Utkarsh Gupta (160050032)

November 2018

1 Introduction

In this study, we present the linear and differential cryptanalysis of a reduced round **SPN** cipher. An **SPN**(Substitution-Permutation Network) cipher encrypts a given input text using an encryption key in several computational rounds. Each such round involves a *substitution* and *permutation* of the input text, followed by an intermixing (usually, *XOR*) with the *round key* bits.

We present this cryptanalysis for two reasons. First of all, many block ciphers that are actually employed today, resemble an SPN cipher *quite a bit*. Therefore, it's cryptanalysis can give insights into the functioning and vulnerabilities of real-world ciphers. Secondly, their cryptanalysis is relatively simple to understand than most other ciphers which are much more involved.

2 SPN

Following is the schema for one encryption round of an SPN cipher. M, K denotes plaintext,key, S denotes substitution, P denotes permutation. $||$ denotes the concatenation operation. We use 1-based indexing and $\oplus$ to denote xor operation. Our plaintext and keys are 16-bits.

$$M \to P(S(M'[1..4])||S(M'[5..8])||S(M'[9..12])||S(M'[13..16])) \quad M' = (M \oplus K)$$

In an n round SPN, the above operation is repeated n times, each time with a different and a random key. However, in the final round there is no permutation just a substitution, i.e. in the last round the schema is as follows

$$M \to S(M'[1..4])||S(M'[5..8])||S(M'[9..12])||S(M'[13..16]) \quad M' = (M \oplus K)$$

Substitution is done using *S-boxes*, where each S-box takes a 4bit input and outputs a 4bit output. The mapping between inputs and outputs is a perutation and all S-boxes use the same mapping.

3 Linear Cryptanalysis

Linear Cryptanalysis is a key-recovery attack which is based on exploiting the absence of perfect randomness in block ciphers by finding linear approximations in the ciphers that are expected to hold with non-negligible probability.

3.1 Basic Idea & Mathematics

For our purposes a Linear equation is an equation of the form

$$X_1 \oplus X_2 \oplus ... \oplus X_n = Y_1 \oplus Y_2 \oplus ... \oplus Y_m \tag{1}$$

where $X_i, Y_i \in \{0, 1\}$. Now note that if we are able to find such an equation to hold for a subset input and output bits for a cipher than the cipher is hopelessly weak, because such an equation allows us to recover extra information about the subset of plaintext bits from the ciphertext bits which allows us to build distinguishes which have advantage very close to 1. A *linear approximation* is a linear equation which is expected to hold with a certain probability. We perform linear cryptanalysis attack by finding linear approximations that hold with high probability in ciphers and use those approximations to perform key recovery.

We start by finding linear approximations of an S-box. Number of possible linear approximations involving input and input bits are $2^4 \times 2^4$ (subset of input bits, output bits). For each linear equation we find the probability with which it holds by iterating through all inputs and the corresponding outputs of the S-box. Then we find the bias(distance from half, as we expect for a perfectly random S-box that the linear approximation holds with probability $\frac{1}{2}$) of each linear approximation. This allows us to form linear approximations for input and output of one round substitution and permutation network.

The next step is to combine linear approximations across different rounds to get linear approximations for multiple round cipher. For this we use the *Piling principle*.

Piling Lemma For n independent, random binary variables, $X_1, X_2.., X_n$,

$$\Pr(X_1 \oplus X_2 \oplus ... \oplus X_n = 0) = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} \epsilon_i \tag{2}$$

where $\epsilon_i = \Pr(X_i = 0) - \frac{1}{2}$.

Proof by induction

The statement is trivially follows for $n = 1$ from the definition of ϵ, Assume that the statement holds for $n - 1$ i.e.

$$\Pr(X_1 \oplus X_2 \oplus ... \oplus X_{n-1} = 0) = \frac{1}{2} + 2^{n-2} \prod_{i=1}^{n-1} \epsilon_i$$

where ϵ_i have usual meaning, then we have for n,

$$\Pr(X_1 \oplus ... \oplus X_n = 0) = \Pr(X_1 \oplus ... \oplus X_{n-1}).\Pr(X_n = 0) + \Pr(X_1 \oplus ... \oplus X_{n-1} = 1).\Pr(X_n = 1)$$

using induction hypothesis above expression can be simplified to

$$= (\frac{1}{2} + 2^{n-2} \prod_{i=1}^{n-1} \epsilon_i)(\frac{1}{2} + \epsilon_n) + (\frac{1}{2} - 2^{n-2} \prod_{i=1}^{n-1} \epsilon_i)(\frac{1}{2} - \epsilon_n)$$

$$= \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} \epsilon_i$$

this concludes our proof. Thus piling principle allows us to combine linear approximation of biases ϵ_1, ϵ_2 to get a new linear approximation of bias $2\epsilon_2\epsilon_2$. To see this note that we can write a linear approximation $X_1 \oplus X_2 \oplus ... \oplus X_n = Y_1 \oplus Y_2 \oplus ... \oplus Y_m$ equivalently as $X_1 \oplus X_2 \oplus ... \oplus X_n \oplus Y_1 \oplus Y_2 \oplus ... \oplus Y_m = 0$ and treat Left hand side as a random variable. Note that while combining two linear approximations, they may have common X_i, Y_j, and thus the random variables are not truly independent but this works out well in practise.

So we proceed as follows, we first find linear approximations of S-boxes, then we trace the output bits, i.e. we see which output bits correspond to which input bit in the next round and then we use another linear approximation using those input bits, combining linear approximations allows us to form linear approximations of more than one round of the cipher. Having described this we now describe how to recover key bits from the key used in the last round. Its assumed that the attacker knows that all S-boxes used are same and also knows their mappings as well so that forming liner approximations for S-box is just an computational task. In next section we describe how to mount an attack.

3.2 Performing Attack

Suppose we have a 5 round SPN, with K_1, K_2, K_3, K_4, K_5 denoting the keys used in respective rounds. Using linear approximation suppose we find the following linear approximation to hold

$$P_5 \oplus P_7 \oplus P_8 \oplus P_8 \oplus U_{4,6} \oplus U_{4,8} = 0$$

with a bias of $\frac{1}{32}$. Then we perform attack as follows,

- For all 2^{16} plaintexts we get corresponding ciphertexts.
- Guess the 2^4 possibilities of 4 bits of the key that are involved with 4bit block having $U_{4,6}, U_{4,8}$.
- For all possibilities of the key invert the ciphertext and perform invert substitution to obtain $U_{4,6}, U_{4,8}$ with respect to this key, and check if the approximation holds or not.
- In this way for each possible key check how many of the plaintext and ciphertext pairs satusfy the linear approximation.
- choose the keys for which the resulting bias agrees closely with the bias of the approximation.

3.3 Results

We wrote a C++ program which carried out the above attack. We used the following linear approximation

$$P_5 \oplus P_7 \oplus P_8 \oplus P_8 \oplus U_{4,6} \oplus U_{4,8} = 0 \oplus U_{4,14} \oplus U_{4,16}$$

which holds with a bias of $\frac{1}{32}$. Note that the actual linear approximation to be used depends on the mapping of the S-boxes and that of permutations, also there are many choices available for the linear approximation. We use `rand()` function of C++ to generate random keys. And indeed we find that the key blocks corresponding to the bias of $\frac{1}{32}$ correspond to the blocks of actual key.

Also we observe that the success of the test depends on the pseudo randomness of the keys, as the keys need to be random enough so that the approximation that we made on the S-boxes(that they are independent) holds.

4 Differential Cryptanalysis

Briefly, Differential Cryptanalysis is a **key recovery** attack exploting the correlation between the *difference* of input plain texts and the *difference* in their corresponding cipher texts, that is observed in the cipher or it's components. Also, differential cryptanalysis is a **chosen-plaintext attack**. Usually, the studied difference is that of XOR of the texts and the input , output difference pair which occurs with high probability is termed as a *differential*.

4.1 Differential Cryptanalysis Insight

We describe briefly, the differential cryptanalysis of an SPN cipher to give an insight into the attack.

We look into the S-box component , for differentials, as it is the only *non-linear component* of the SPN cipher. We note that differentials for a single S-box hold with same probability for an keyed S-box (where input bits get XORed with key bits before S-box operation). This is because, the difference operation nullifies the affect of XORing with key bits upon the input texts.

After obtaining differentials for a single unkeyed S-box, we extend it to multiple round SPN cipher, assuming that differentials of S-boxes are independent. More formally, suppose if (a1, b1) and (b1,c1) are differentials of S-boxes S1, S2, respectively, with probabilities p1, p2. Then, we consider (a1, c1) to be a differential of S1-S2 network with probabiliy p1 * p2. This is justified in an SPN network as the output of one round is XORed with the *pseudorandom* round key before, it is outputed to the next S-box. In this manner, a set of differentials and their probabilities can be constructed for a multiple rounds of SPN cipher, while adjusting according to the permutation between the rounds appropriately.

Finally, after obtaining the differentials, we recover the key bits as follows. We start the attack by first recovering the last round key bits and then propagating backwards. For recovering the last round keybits, we first find differentials of the last - 1 rounds. We term all those keybits of the last round where output difference of the probable difference isn't zero as *target partial subkey* bits, as we target to retrieve them using the differential.

We get encryptions of many plain text pairs with the same input difference as in the differential and for each encryption pair, we do a partial decryption of the encrypted texts (over the *target partial subkeys*) and run the S-box and P operations for the last round backwards. Now, if the difference of both the partially decrypted texts are same the output of the corresponding differential, then the selectivity of the key is increased. Finally, we choose the key bits as those whose selectivity is highest.

4.2 Differential Attack Experiment

A differential attack on the 4-round 16 bit SPN cipher has been conducted programmatically as follows.

4.2.1 Experiment Setup

The S-box and permutation operations of encryption are implemented programmatically. The secret keys for various rounds are picked at random. All the S-box, permutation operations across the rounds are set to be same, for programmatic ease.

4.2.2 Building the DDT of S-box

A **DDT** (**Difference Distribution Table**) of a S-box contains the input-output difference pairs of the S-box and their probabilities. This has be constructed for the S-box through exhaustive over the possible plain-text input pairs to the S-box.

4.2.3 Finding the most probable differentials

The differentials for 3 rounds of the cipher are found in the following way. We only considered differentials with *single S-box input difference* as this would fasten our search. At each level(round), for each of the S-box with non-zero input difference, all the output differences are considered, and these after permutation leads to different input differences to the next round.

This has been implemented in a *depth-first fashion* and the probability for the entire path is calulated as *product* of input-output choices taken at each S-box of the path, as per the *DDT*. At the end of the final (in this case 3rd) round, the input-output differences are stored. For faster search, intermediate paths whose probability is less a certain threshold are abandoned.

Finally, the differentials with the highest probability are selected.

4.2.4 Key bits Recovery

From the computed probable differentials, the *target partial subkey bits*(last round key bits in the non-zero bits of output difference) are recovered as follows.

Many plain-text pairs with the input difference of the differential are chosen and encrypted. For each such pair, over the entire subkey space, partial decryption is done on the ciphertexts(i.e, XORing) for each key, and the data is run through S-box *backwards*. If the resulting texts have the same output difference of the probable *3-round SPN circuit*, then, the selectivity of the key is increased.

Finally, the keybits with high selectivity are outputed.

4.2.5 Evaluation and Critique

The complexity of the attack on 4-round SPN cipher is $O(2^8)$ for DDT construction, $O(2^4 * (2^{16})^2)$ for finding differentials and $O(2^{16} * 2^8)$ for key recovery. Therefore, the overall attack is $O(2^{36})$ and the attack takes about 1-2mins on a personal computer, this is a great improvement over the naive algorithm which would take $O(2^{16*4})$ time.

The attack has been run over different instantiations of the keys. Here are our results.

For example, for the key instantiation of {0xeb4f , 0x1dc5, 0x290c, 0x706c, 0xca55}, differential (input = 0xB00 , output = 0x606 , probability = 0.026) and the target partial keys being the **1st** and **3rd** byte(the less significant), the retrieved probable keys along with their selectivity are as follows

subkey bits	**probability**
805	0.00585938
a00	0.00875854
a02	0.00598145
a05	**0.0200195**
a07	0.00595093
d05	0.00585938
f05	0.00881958

For the same differential, and key instantiation of {0xff70, 0xe0a6, 0x1bcc, 0x7592 , 0xe1ec}, the retrived key bits and their selectivities are

subkey bits	**probability**
109	0.00909424
10b	0.00604248
10c	**0.0213013**
10e	0.00601196
30c	0.006073
40c	0.00860596
60c	0.00604248

The results are satisfactory.

We note that the success of the attack seems to dependent on the instantiation of the keys, as the keys need to be pseudorandom enough for the independence of S-boxes to hold. There seems to be other factors which can result in random keybits, outputing the required output difference after a partial decryption, because of which some key estimates have about the selectivity of the true keybits.

5 Future Scope of work

We would like to increase the **search space** for the *linear and differential characteristics*, thereby, retrieving all the keybits of last round. Then, all the previous round key bits can be extracted in a similar fashion, by the linear and differential characteristics of lower level rounds.

We could use this model as benchmark for testing different forms of cryptanalysis such impossible differential , integral etc., Also, since *exhaustive search* over target subkey bits is not possible for real-life ciphers which have much larger input-output sizes, we would like to study the possible techniques and implement them.

6 Conclusion

The experiments demonstrates that linear and differential cryptanalysis are powerful and effective tools for SPN ciphers and also gives insight in breaking real-life ciphers using the same. While linear cryptanalysis is based on the linear characteristics of S-boxes, differential cryptanalysis is based on it's differential characteristics.

Thus, while building ciphers, the S-boxes must be chosen carefully, not to exhibit such properties. Also, if the S-boxes in the cipher are designed in a dependent fashion, i.e, one level of the cipher nullifing the characteristic of another, then the overall cipher would have no significant characteristics.