[ASSIGNMENT #1]

OCTOBER 4, 2021

[MHAMOUD SAMI]
[MOHAMED ABOHELAL]

Firstly, we generated a random S-box and its inverse.

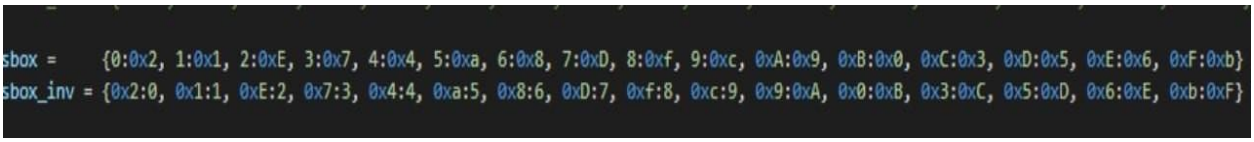


Figure 1 generating an s-box S

Secondly, we constructed its Linear Approximation Table.

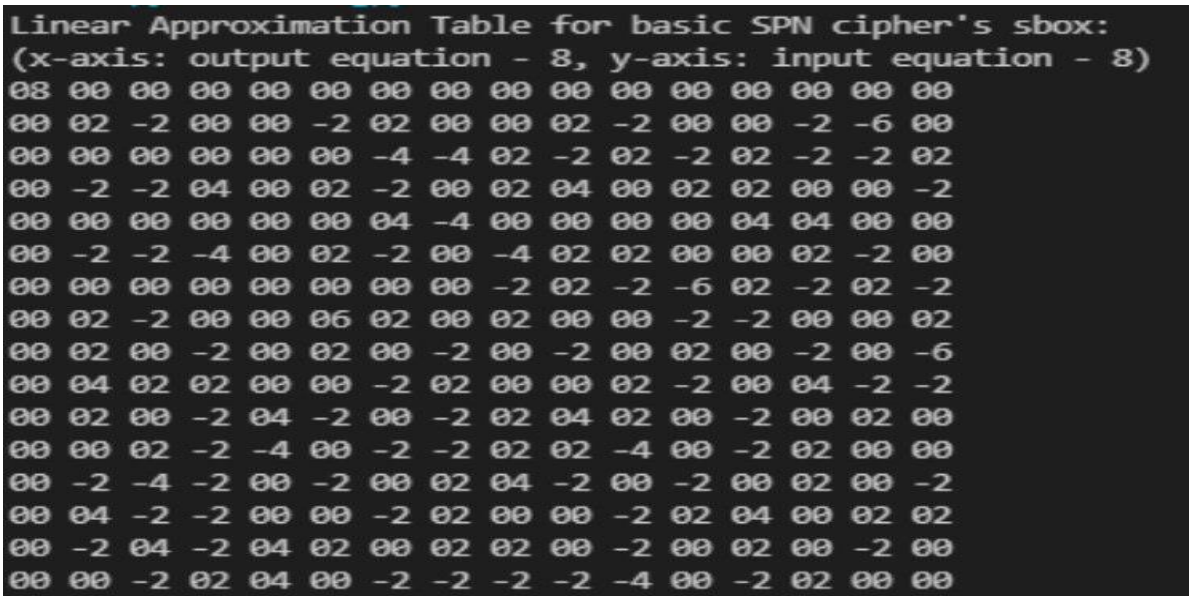


Figure 2 linear approximation table

Thirdly, we found the best Linear Approximation.

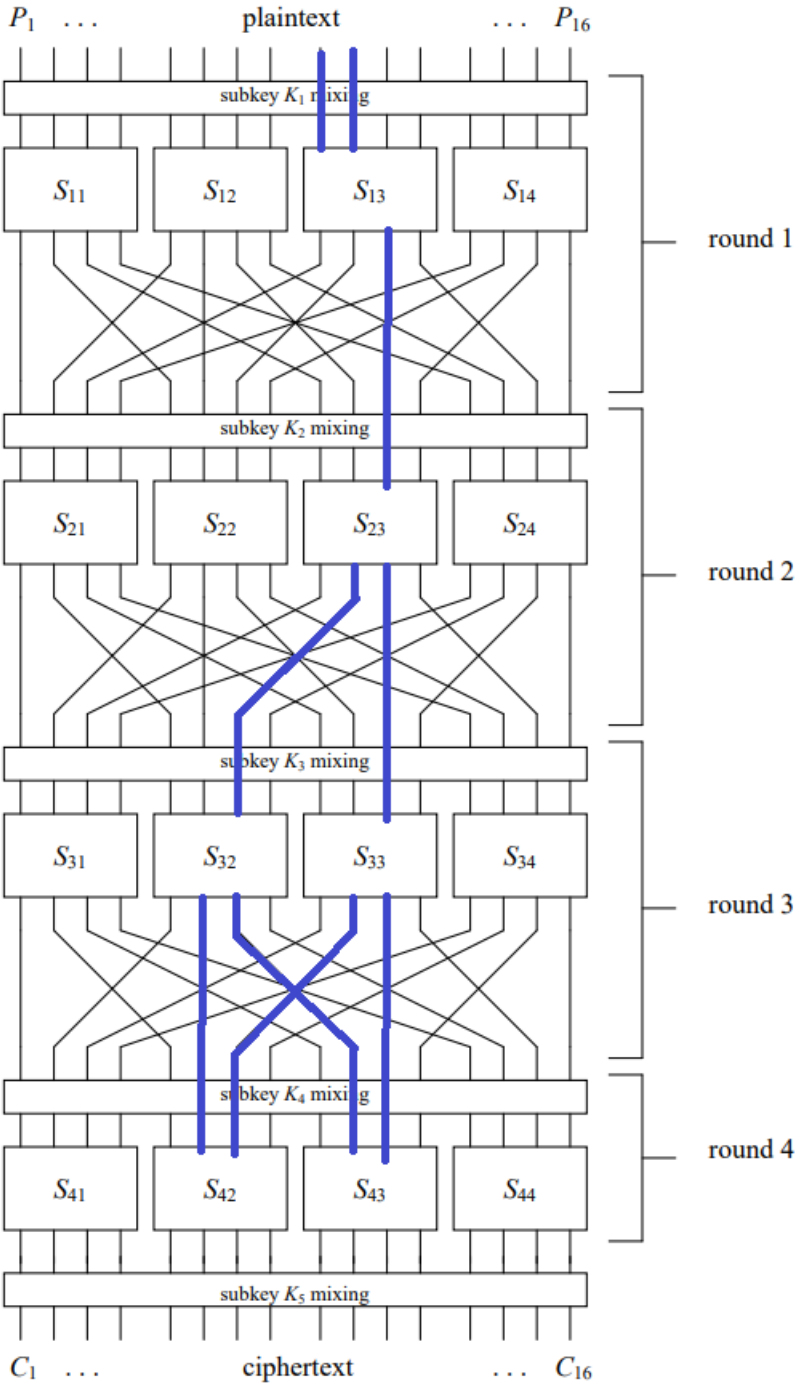


Figure 3 Linear approximation path

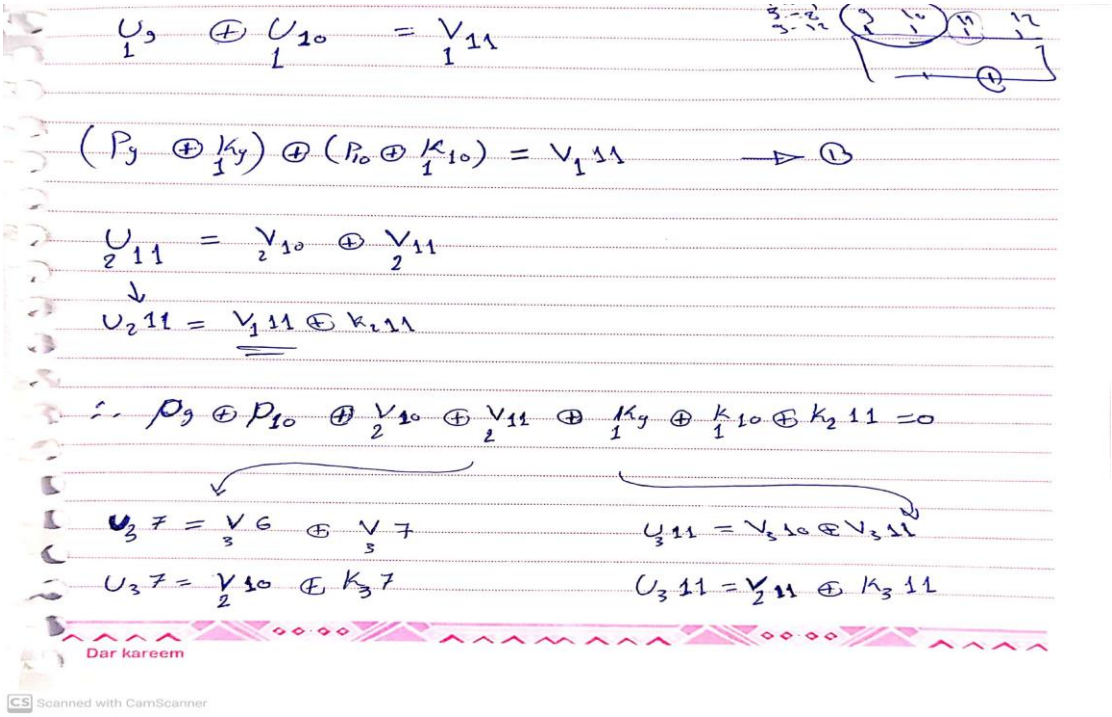
The next step, we calculated the bias of the path we have chosen.

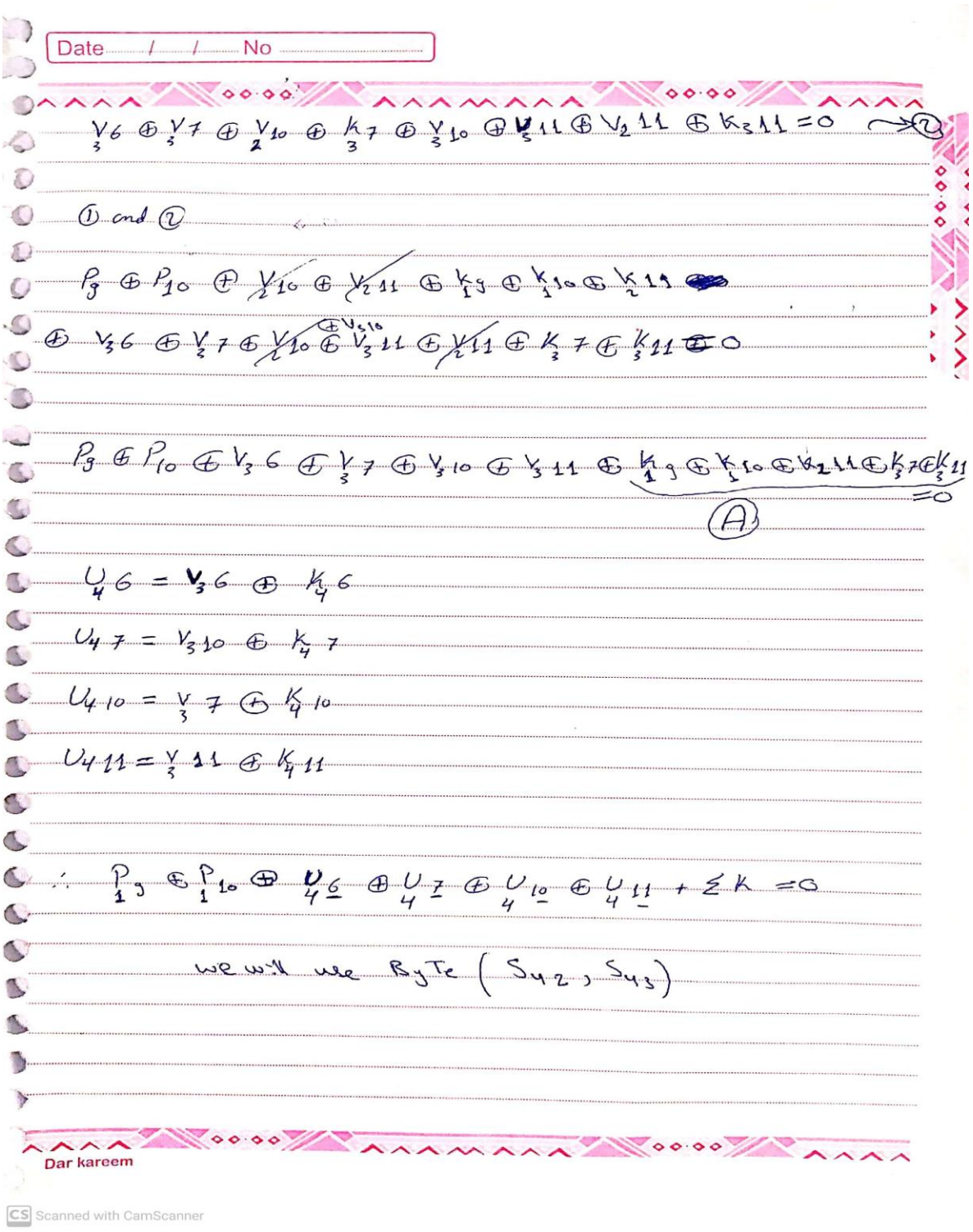
x1 ⊕ x2 = y2 ⟶ round 1

x3 = y2 ⊕ y3 ⟶ round 2

x3 = y2 ⊕ y3 ⟶ round 3

x3 = y2 ⊕ y3 ⟶ round 3





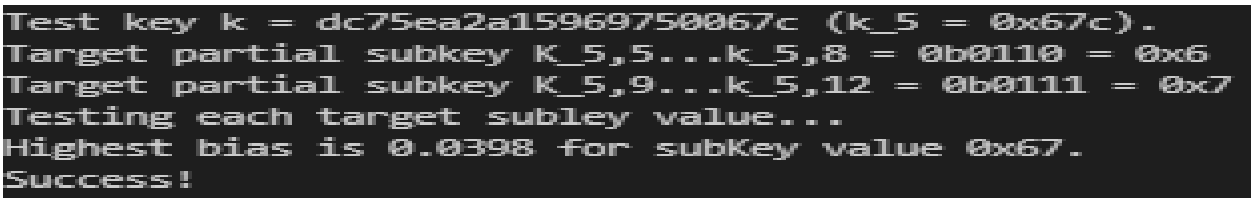
- Probability = $\frac{1}{2} - 2^3 \left(\frac{1}{4} - \frac{1}{2}\right)^4$
- Bias= -1/32

By application of the Piling-Up Lemma, the above expression holds with probability 15/32 (that is, with a bias of −1/32).

Now since ΣK is fixed, we note that must hold with a probability of either 15/32 or (1−15/32) = 17/32, depending on whether ΣK = 0 or 1, respectively. In other words, we now have a linear approximation of the first three rounds of the cipher with a bias of magnitude 1/32.

The last step, we extracted the key bits.

We would try all 256 values for the target partial subkey [K5,5...K5,12]



Discuss the results of the attacks.

After trying all 256 values for the target partial subkey we found that highest bias is 0.0398 for subkey value 0x67. And we can conclude that we can get the whole key of the encryption And of course we can't depend on the S-P-N or DES for encrypting our Data.

Our code to attack the SPN network

- GitHub - Mahmoudsami11095/Crypto

References

- https://github.com/hkscy/Basic-SPN-cryptanalysis
- https://ioactive.com/wp-content/uploads/2015/07/ldc_tutorial.pdf