CSCI-4230-HW1

Isaac Llewellyn 661582602 llewei

September 24, 2018

Homework 1 Part 2

Q1. [25pnts]For the simplified DES, consider S-Box S0 and show how DiffCrypto attack would work. Show your work for partial credit.

S-Box0

$$\begin{pmatrix} 1 & 0 & 3 & 2 \\ 3 & 2 & 1 & 0 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 3 & 2 \end{pmatrix}$$

This attack focuses on the non-uniform distribution of values an S-Box may yield. Note, this attack depends on the attacker being able to run the encryption algorithm on their own. We do this by creating a table of XOR pairs and comparing the probability of their occurrences. This is called a differential distribution table; one for Sbox0 is show below and on slide 39 of Monday Week 2's slides. The attack on Sbox0 works quite nicely well because its configuration is static across SDES distributions. Differential Distribution Table

$$\begin{pmatrix} 16 & 0 & 0 & 0 \\ 0 & 1 & 5 & 2 \\ 0 & 5 & 3 & 0 \\ 1 & 2 & 0 & 5 \\ 1 & 2 & 4 & 1 \\ 5 & 0 & 2 & 1 \\ 0 & 1 & 1 & 6 \\ 2 & 5 & 1 & 0 \\ 1 & 2 & 4 & 1 \\ 4 & 1 & 1 & 2 \\ 2 & 1 & 1 & 4 \\ 1 & 4 & 2 & 1 \\ 4 & 1 & 1 & 2 \\ 1 & 2 & 4 & 1 \\ 1 & 4 & 2 & 1 \\ 2 & 1 & 1 & 4 \end{pmatrix}$$

An attack will choose known plain-texts and their respective xors and then feed them into the encryption algorithm, storing each result and the xor of each as well. We compare the results to our table to get more information about the keys. We note that S-Box0 has a disproportionate amount of 3's. From, we can take all inputs that yield an S-Box output of 3 and store them. Then we take one of the values and xor it against all the others. We can then learn the basic key-space to search from based off the xored results. Further repetition of this process reduces the key-space and will eventually yield a single key.

Q2. [25pnts] Consider the crypto system below and compute H(K—C)

$$H(X) = -\sum_{i=1}^{n} P(Xi) log2 * P(xi)$$

P = a,b,c with Pp(a) = 1/3 , Pp(b) = 1/6 , Pp(c) = 1/2
K = k1, k2, k3 with Pk(k1)=1/2 , Pk(k2)=1/3 , Pk(k3) = 1/4
C = 1,2,3,4

- ek1(a) = 1 ek1(b) = 2 ek1(c) = 2
- ek2(a) = 2 ek2(b) = 3 ek2(c) = 1
- ek3(a) = 3 ek3(b) = 4 ek3(c) = 1

H(K—C) = H(K) + H(P) - H(C)
H(K) = -(1/2log2(1/2) + 1/4log2(1/4) + 1/4log2(1/4) = -(-1/2 -1/2 -1/2) = 1.5
H(P) = -(1/3log2(1/3) + 1/6log2(1/6) + 1/2log2(1/2). = 1.46
H(C) =

Pc(1) = 1/6 + 1/8 = 7/24

Pc(2) = 1/12 + 1/12 + 1/4 = 5/12

Pc(3) = 1/12+ 1/24 = 1/8

Pc(4) = 1/24 + 1/8 = 1/6

= 7/24log2(7/24) + 8/12log2(5/12) + 1/8log2(1/8) + 1/6log2(1/6) = 1.85
H(K—C) = 1.5+1.46-1.85 = 1.11

Work in progress do not mistake for anything knowledgable. For this example we will look at the table and choose (D,0). In this case there are 8 key value pairs of the 16 inputs when xored with D and put into the S-Box that output 0. Note that these pairs are counted twice (D,k) (k,d) as either could be the key or value. We want possible inputs that xor to D Looking at ”$https://www.garykessler.net/library/byte_logic_table.html$”, we can see there are 16 pairs of values that xor to D These pairs are (0,D),(1,C),(2,F),(3,E),(4,9),(5,8),(6,B),(7,A),(8,5),(9,4),(A,7),(B,6),(C,1),(D,0),(E,3),(F,2). Note there are only 8 unique pairs as each pair has a flipped duplicate

1. f: 1111 -¿ 2
2. e: 1110 -¿ 3
3. d: 1101 -¿ 3
4. c: 1100 -¿ 1
5. b: 1011 -¿ 0
6. a: 1010 -¿ 2
7. 9: 1001 -¿ 1
8. 8: 1000 -¿ 0
9. 7: 0111 -¿ 1
10. 6: 0110 -¿ 0
11. 5: 0101 -¿ 2
12. 4: 0100 -¿ 3
13. 1: 0001 -¿ 0
14. 0: 0000 -¿ 1

The values are: 8,b,6,1 ....

As SBoxs are have a non uniform distribution, the attacker can note where the probabilities of a cipher-text occurring given a plaintext is high, and use that to find the sub-key value used. Once a correlation between input and cipher-text is found, it can be used to weaken the overall security.

””If the input to an S-box is a uniformly distributed random number, its output will be a uniformly distributed random number””