Homework 4

1. *Consider creating a DES-like block cipher with a block size of 16. Assume that the cipher has a fixed matrix , just like DES, that operates exactly as DES's matrix. Given the IP matrix shown below, calculate the corresponding matrix, .*

The matrix is represented by “swapping” the position a number is at, at the number itself for the matrix . In this case for example, since 7 is in position 1, 1 would go in position 7 for . Another example is since 1 is in position 15, 15 would go in position 1 for

Applying this “swapping” gives us for matrix :

1. *The input to the expansion matrix E in DES, expressed in hexadecimal is B4 F3 92 C6. What is the output? Please express your answer in hexadecimal and put a little space between each group of 2 hex characters.*

First step, turn the Hexadecimal into binary.

The binary representation is:

Now, we use the expansion matrix:

Now we use the binary number , and put the bits in the corresponding position in the expansion matrix (e.g. if the number is 32, we put the 32nd bit in in this matrix).

Reading this to get the output (from left to right)

Converting this into Hex:

1. *Consider a portion of a single DES round where the input (expressed in HEX) to S boxes S1, S2, S3, S4, S5, S6, S7, and S8 is 24 13 AC BD 5F 6E. What are the 32 bits of output from the S-boxes? Express your result in binary.*

We first have to put the input into binary, writing this out we have

Since we are mapping 48 bits into 32 bits, we need to grab every 6 bits in the input, then use the 8 S box tables to look them up.

Bit input 1:

* Getting the row, we have to use the 1st and last bit, this means the row is
* For the column, we use the inner four bits, this means the column is
* Using we see that

Bit input 2:

* For the row,
* For the column,
* Using we see that

Bit input 3:

* Using

Bit input 4:

* Using

Bit input 5:

* Using

Bit input 6:

* Using

Bit input 7:

* Using

Bit input 8:

* Using

Now that we have all the 8 different Hex results, we can put them together!

Writing this in binary notation, our final answer is:

1. *Consider calculating the round 16 key in DES. Given that the input key, with odd parity bits, when described in HEX is "C761 4592 BCC2 5D38", determine the first 10 bits of the round 1 key.*

First, we need to write our actual key! Converting the Hex to Binary, we have for the key:

Now we can apply and set the left and right side to values and , is the leftmost 28 bits of , whereas is the rightmost 28 bits:

Applying the matrix, which is:

We take the top half of it for and the bottom half for (e.g. The th bit in the key is the 1st bit in .

Using we get the following values:

Now, we enter the round phases. Since this question is only asking for the first 10 bits of the round 1 key, we only need to do computation for 1 round. The first step is finding from . To do this, we need to apply since the value of the subscript is , we only need to perform a left cyclic shift by 1 bit!

Therefore, the value for is:

Similarly for we need to also apply to . Doing this shift, we get:

Now for the final part, applying onto , where the value of is a binary number where the first 28 bits are from and the last 28 are from . This permutation is a similar process to the first one we did. Since we are only looking at the first bits of , I will only show the first 10 bits of , which are:

If we combine and into one binary number, it looks like:

Now all we do is grab the bits in the positions given! Doing this, we get a value of (first 10 bits) being:

1. *(I was also going to copy this question down, but it is so long, sorry)*

We can represent the two scenarios A and B as two equations.

For Student , he initially has a time of 1 hour, which is equivalent to seconds. Then, he also can check keys every second. The number of keys he would need to check would be , where is the number of keys revealed. The reason for this is because every bit can be a or a . Therefore, there are two options for each bit in the key. So, the equation for the time he takes is:

For Student he initially takes 2 days to write his code, which in seconds is . Since their code is faster, he can check keys per second, and the number of keys he needs to check is also . So, the equation for Student is:

Setting these equal to each other:

I am going to apply the substitution to make solving this simpler, also, I will express the seconds in some factored forms

Now that we know the value for , we can get the value for :

Now, we utilize some handy exponent and logarithm rules:

The closest integer to this value is . Therefore, the professor should reveal bits of the bit code.

1. *Let the state matrix to AES right before the SubBytes step be the matrix shown below. Show the state of the matrix right AFTER the SubBytes step:*

For this, we look at the AES S-Boxes chart and match the corresponding 2 length Hex strings with their row/column representation in the chart

Some pairs we can see for the first row is:

Applying the S-Box substitutions, the corresponding matrix state after the SubBytes step is:

One thing to note is that many of the given 2 length Hex strings are form a pattern on the chart, for example, to go from to , you go one row up and one column to the left, this is the same distance from to . Other rows share this pattern in the given input. This made table lookups super easy, so thank you （^\_^）

1. *Let the state matrix to AES right before the ShiftRows step be your answer from problem 7. Show the state of the matrix right AFTER the ShiftRows step.*

To perform the shift rows algorithm, we must shift row 1 by 0, row 2 by 1, row 3 by 2, and then row 4 by 3.

Performing this on our answer from Question 6:

Therefore, our resulting State Matrix is:

1. *Consider the process of AES Key Expansion. Imagine that we have:*

* *w[32] = A3 B4 C7 D9 (in hex)*
* *w[35] = 05 18 2E 6F (in hex).*

*Calculate w[36], showing each of the following intermediate results: RotWord(temp), SubWord(RotWord(temp)), Rcon[i/4], and the result of the XOR with Rcon[i/4]*

First, we need to observe the divisibility of 36 with respect to 4. We see that 36 is divisible by 4, so there are multiple steps involved to calculate .

The first step, we perform , this step involves doing a left cyclic shift by 1 bite. Performing this on , we get:

For the next step, we perform the step, which involves taking the result from , and using the AES S-box substitution chart on the variable .

Looking at the chart, we can see these substitutions for :

Applying these substitutions, now equals:

For step 3, we have to look at the array at the position , this equals a value of . The array is given as follows:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

Using the value at we see get the value of . Now we take the previous value from the step, and XOR these two values

Now for the final step! All we do is we take this value and XOR it with !

Now we have solved for . For a table representation of the steps:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |

1. *In class we discussed multiplication in the AES field GF(28 ) with the irreducible polynomial . Based on this discussion, derive the answer for the calculation of . Display your final result with two hexadecimal characters.*

Starting out, we can Express in binary as . Then, we can split into three steps, and . Doing each of these steps:

* , we keep the input the same (Left shift of 0)
* , we perfom a left shift by on the input
  + We need to correct this since we have an overflow. This overflow in polynomial notation is .
  + From the AES polynomial,
  + This means when we add (XOR) all the results, we also must include , and remove the overflow from the other result
* Now for , we do a left shift by on the input
  + ,
  + We have an overflow again, this overflow in polynomial notation is , we can express this as , performing the same substitution as above, we get
  + This means when we add all the results, we must also include , and remove the overflow from the other result

Now, we add up (XOR) all of the results. The results in bold are from the overflow additions:

This result in Hexadecimal comes out to be:

1. *Let the input to the MixCols (during AES encryption) be:*

*What’s the output in row 3 column 2? The matrix by which to “multiply” is:*

Starting off, we need to express the expression we would have for the new state matrix in , by simple matrix multiplication rules, would be

In this expression, addition is essentially equivalent to XOR operator on these 4 values, since only 1 and 0 can exist in the AES field (As long as we throw away overflow bits from adding). Binary numbers in **BOLD** represent values I will add at the end to achieve

Starting with , expressed in binary notation is .

Then, we must split into two parts, and :

* , we keep the input value the same, in binary, this is:
* 02, we shift left once:
* Now XOR these two values:

This means that

Now we perform , in binary is

* , we shift left once:
* As we can see, we have an overflow, this overflow bit is represented as in polynomial notation. From the AES polynomial, we know that
* This means when we add up the results, we must also include a factor of
* Since we accounted for this factor, the term becomes:

Next value we have to compute, , in binary is

* , we just keep the input the same:

Our final computation! , in binary is

* , we just keep the input the same:

Now that we have collected all of our bolded terms, we add them

This answer in hex notation is: