# CS3626 Homework 03 Spring 2024

# DES

**Total Points: 25**

Be as brief as possible and use your own words when describing concepts.   SHOW ALL WORK for Questions requiring calculations and algorithms.

Q-1: Calculate the following polynomial operations over a m(x), GF(2^x) where ai ∈ GF(2) given m(x)

Given these polynomials where:

A = x7 + x6 + 0 + x4 + 0 + x2 + 0 + x0

B = x6 + x5 + 0 + 0 + 0 + 0 + x0

m(x) = x8 + 0 + 0 + 0 + x4 + x3 + 0 + x1 + x0

ai operations are modulo-2 math.

Calculate the operations below; show the setup and operation steps and result, remember they are over m(x):

A + B =

A \* B =

A mod B =

2\*A + 3\*B + (A\* B) + (A+B) =

// recall the shortcut of 3\*A = A + 2\*A = A + (A << 1) over m(x)

// you already have A\*B and A+B from above

Use Extended Euclidean on the pair ( m(x), A) to find the modular inverse A-1 .

**5 points**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| a | b | q | | r | t1 | t2 | t3 |
|  |  |  |  | | 0 | 1 |  |

Q-2: Build the 4x4 State box from the following 128-bit plain text input:

Input = 4e 6f 20 6d 6f 72 65 20 73 65 63 72 65 74 73 2e

I forgot if you are supposed to build that state box row by row or column by column. So, I will just do it both ways.

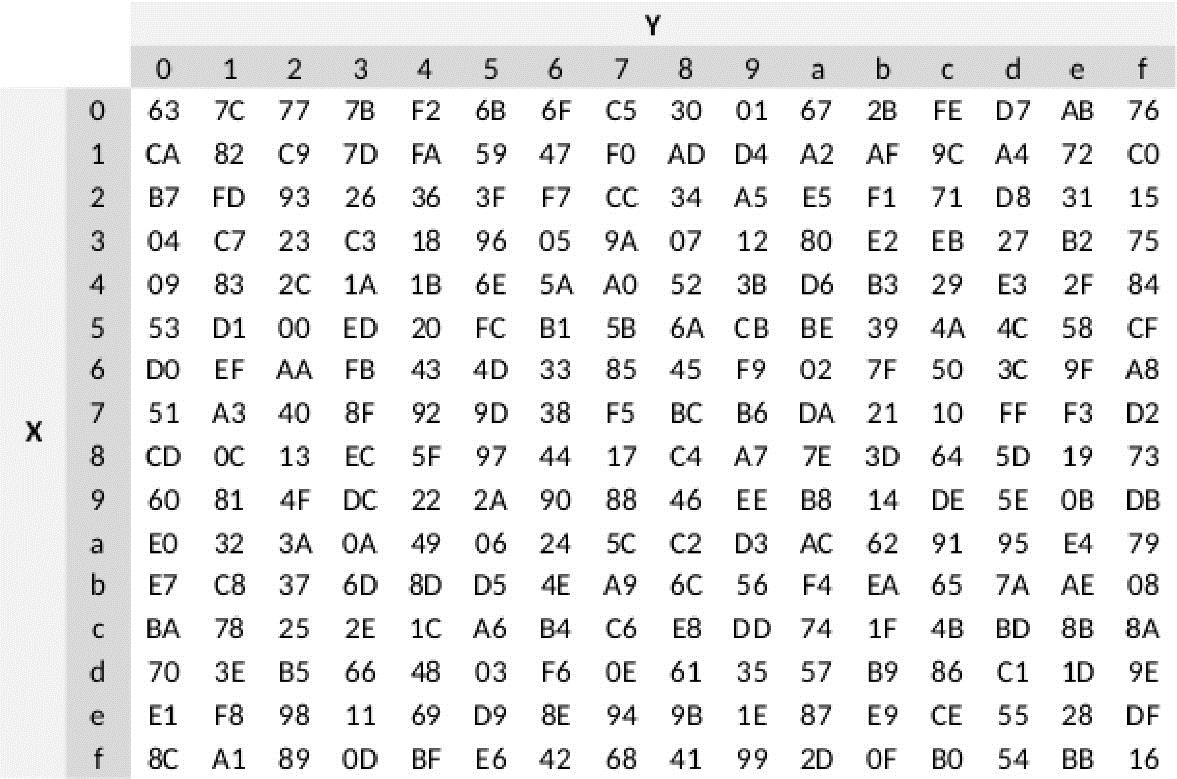
|  |  |  |  |
| --- | --- | --- | --- |
| 4e | 6f | 73 | 65 |
| 6f | 72 | 65 | 74 |
| 20 | 65 | 63 | 73 |
| 6d | 20 | 72 | 2e |

or

|  |  |  |  |
| --- | --- | --- | --- |
| 4e | 6f | 20 | 6d |
| 6f | 72 | 65 | 20 |
| 73 | 65 | 63 | 72 |
| 65 | 74 | 73 | 2e |

**1 point**

Q-3: Take the above state box convert the state box values by applying each byte to the AES-SBOX. What is the new state box result?



**HexVal = X0 X1  location of X0 = X, X1 = Y**

|  |  |  |  |
| --- | --- | --- | --- |
| 4e => 2F | 6f => A8 | 73 => 8F | 65 => 4D |
| 6f => A8 | 72 => 40 | 65 => 4D | 74 => 92 |
| 20 => B7 | 65 => 4D | 63 => FB | 73 => 8F |
| 6d => 3C | 20 => B7 | 72 => 40 | 2e => 31 |

or

|  |  |  |  |
| --- | --- | --- | --- |
| 4e => 2F | 6f => A8 | 20 => B7 | 6d => 3C |
| 6f => A8 | 72 => 40 | 65 => 4D | 20 => B7 |
| 73 => 8F | 65 => 4D | 63 => FB | 72 => 40 |
| 65 => 4D | 74 => 92 | 73 => 8F | 2e => 31 |

**2 points**

Q-4: What is the new state box result of applying the AES shift Columns function to the state box above:

Instead of shift columns, it should be row shift.

|  |  |  |  |
| --- | --- | --- | --- |
| 2F | A8 | 8F | 4D |
| 40 | 4D | 92 | A8 |
| FB | 8F | B7 | 4D |
| 31 | 3C | B7 | 40 |

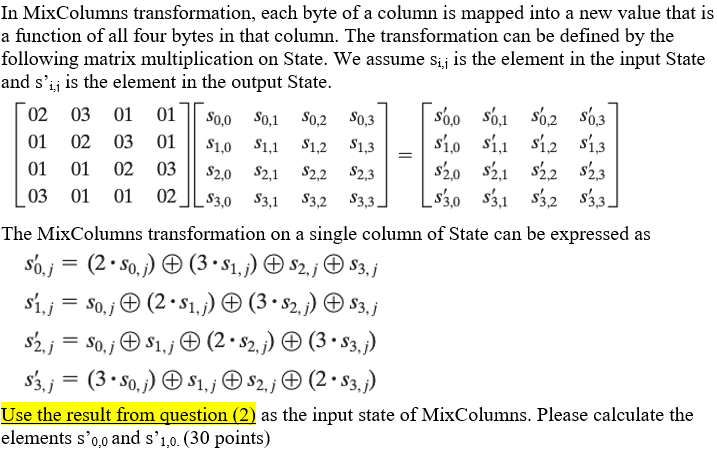
or

|  |  |  |  |
| --- | --- | --- | --- |
| 2F | A8 | B7 | 3C |
| 40 | 4B | B7 | A8 |
| FB | 40 | 8F | 4D |
| 31 | 40 | 92 | 8F |

**2 points**

**Q-5: Given the above state box, calculate the Mix Columns result of ONLY s’1,1**

**Recall:**



**Show all calculation steps (XOR operations and multiplication/shifts)**

**S0,1 = A8 = 1010 10002**

**S1, 1 = 4D = 0100 11012**

**S2,1 = 8F = 1000 11112**

**S3,1 = 3C = 0011 11002**

**S’1, 1 = S0,1 ⊕ (2\*S1, 1) ⊕ (3\*S2,1) ⊕ S3,1**

**S’1, 1 = 1010 1000 ⊕ (0010 \* 0100 1101) ⊕ (0011 \* 1000 1111) ⊕ 0011 1100**

**S’1, 1 = 1010 1000 ⊕ 1001 1010 ⊕ 0001 1010 1101 ⊕ 0011 1100**

**S’1, 1 = 0011 0010 ⊕ 0001 1001 0001**

**S’1, 1 = 0001 1010 0011 = 0x1A3**

**Or**

**S0,1 = A8 = 1010 10002**

**S1, 1 = 4D = 0100 11012**

**S2,1 = 40 = 0100 00002**

**S3,1 = 4D = 0100 11012**

**S’1, 1 = S0,1 ⊕ (2\*S1, 1) ⊕ (3\*S2,1) ⊕ S3,1**

**S’1, 1 = 1010 1000 ⊕ (0010 \* 0100 1101) ⊕ (0011 \* 0100 0000) ⊕ 0100 1101**

**S’1, 1 = 1010 1000 ⊕ 1001 1010 ⊕ 1100 0000 ⊕ 0100 1101**

**S’1, 1 = 0011 0010 ⊕ 1000 1101**

**S’1, 1 = 1011 1111 = 0xBF**

**S’1,1 = 0x1A3 or 0xBF**

**4 points**

**Q-6: Implement the following polynomial operation in a language (C++, C#, Python, Java):**

1. **Get Polynomial Degree**

**Write a function to find the highest degree of a given polynomial: e.g. if given**

**x13 + x7 + x4 + x1 + x0 the function get\_degree( a(x)) will return 13.**

**A pseudocode algorithm:**

// returns highest exponent value of shortcut representation of a polynomial

// returns -1 if a(x) == 0

get\_degree( unsigned int a(x))

{

int degree = -1;

if( a(x) == 0) {

return(degree);

}

while( a(x) != 0) {

++degree; // keep incrementing while there are 1’s in number

a(x) = a(x) >> 1; // divide by 2 which shifts out lowest bit value

}

return( degree);

}

1. **Polynomial Modulus over GF with m(x)**

**Write a function to long divide two polynomials, i.e. (A(x), m(x)). Use the binary shortcut to indicate the values of A(x) = an\*xn + an-1\*xn-1 +… a5\*x5 + a4x4 + a3x3 + a2 x2 + a1x1 + a0x0 becomes the binary number anan-1… a5a4a3a2a1a0 and perform the respective bitwise operations. This makes m(x) = 0x11B in hexadecimal for example. The function use modulo 2 math. A possible algorithm:**

Unsigned int divide\_galois( A(x), m(x))

{

result = A(x);

while (get\_degree(result) >= get\_degree(m(x)) {

result = result XOR ( get\_degree(result) - get\_degree(m(x)));

}

return( result);

}

unsigned int A(x) = multiplicand

unsigned int B(x) = multiplier

unsigned int m(x) = modulus = x8 + 0 + 0 + 0 + x4 + x3 + 0 + x1 + x0 =

1 0001 10112 = 0x11B

unsigned int multiply\_galois( A(x), B(x), m(x))

{

int index = 0;

int result = 0;

while( B(x) != 0) { // run until no bits left in multiplier

If ( b0 == 1) { // Test the lowest bit of b(x)

result = result XOR (A(x) << index)) // index is the current degree of the multiplicand

}

B(x) = B(x) >> 1 // SHIFT out the current b0 and bring new b0

++index; // raise the degree of the multiplicand for next loop in the line above

}

return( divide\_galois( result, m(x));

}

**Using the above calculate the following and provide the results here:**

**get\_degree( 0x3CF0) = \_\_\_\_\_13 = x^13\_\_\_\_\_\_\_\_\_**

**get\_degree( 0x10000) = \_\_\_\_\_16 = x^16\_\_\_\_\_\_\_\_\_**

**get\_degree( 0x00) = \_\_\_\_\_\_\_\_0\_\_\_\_\_\_\_\_\_\_\_\_**

**get\_degree( 0x01) = \_\_\_\_\_\_\_\_1 = x^0\_\_\_\_\_\_\_\_\_\_**

**divide\_galois( 0x1000, 0x11B) = \_\_\_1000 1110 = x^8-x^4-x^3-x\_\_\_\_\_\_\_\_\_\_\_**

**divide\_galois( 0xE1, 0x11B) = \_\_\_\_\_0011 = x-1\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**divide\_galois( 0x32CFE1, 0x11B) = \_\_0011 0001 0011 1111= x^13 + x^12 - 2x^8 - 2x^5 + x^4 + 3x^3 + x^2 + 5x + 4\_\_**

**divide\_galois( 0xE1, 0x11B) = \_\_\_\_\_\_\_\_\_\_0011 = x-1\_\_\_\_\_\_\_\_\_\_**

**multiply\_galois( 0xD5, 0x61, 0x11B) = 1111 111 = x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x^1 + 1\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**multiply\_galois( 0x1E3C, 0x1E3C, 0x11B) = \_0001 0100 1011 1111 1111\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. **3 points**
2. **4 points**
3. **4 points**

**Keep source code, it will be needed for future exercises.**

**AES KEYGEN and DECRYPTION will be in Next Assignment**

**Submission Guidelines:**

* No handwritten submission is accepted, always submit answers as text within this or similar document file with any support images embedded in the file.
* **EXCEPTION**: If asked for source code implementation you can submit those individually and as separate files in ASCII format in their original file format .cpp, .java, .py, .cs etc. or even as a .txt file will be acceptable. Do not insert code into the submission document file. It ruins spacing which makes .python and some languages (perl, awk etc.) difficult to test build.
* Do not submit ZIP files… ever… for anything in D2L. The system is extremely unhelpful with regards to those filetypes and grading.
* You may include your freehand drawing/image and handwritten scans in the submission. However, the writing and images must be clearly legible. Though, it is best to present non-handwritten submissions, generally, as is done in the professional setting.
* If asked, show all work/calculations/graphs etc. in the determination of the problem.
* **Please complete your entire work in a single Word Document and Save the file as: yournetid\_CS3502\_Assignment01.docx (e.g. ogarcia5\_CS3502\_Assignment01.docx.) and upload your file in D2L.**
* Please observe the submission due date and time. After the due date there is a 50% penalty for the next 24 hours. Any submission after 24 hours of the due date will be graded at 0%.
* If you include a reference or an image taken from other sources, please cite them appropriately. APA is preferred but cite them so they can be found. **NOTE: verbatim copying or even paraphrasing is plagiarism so if the source used constitutes your answer rather than simply *supporting* the answer, it will be considered invalid. This is especially true of source code implementation answers.**
* If you resubmit, please make sure to attach the file again. Your latest submission before the due date will be the one graded.