CPSC 418 / MATH 318 — Introduction to Cryptography
ASSIGNMENT 2

Name: Artem Golovin
Student ID: 30018900

Problem 1 — Binary polynomial arithmetic, 20 marks

a. i.

$$x^3$$
$$x^3 + 1$$
$$x^3 + x$$
$$x^3 + x + 1$$
$$x^3 + x^2$$
$$x^3 + x^2 + 1$$
$$x^3 + x^2 + x$$
$$x^3 + x^2 + x + 1$$

ii. 1.

$x^3 = x^2 \cdot x$
Solving for x:
$x^2 \cdot x = 0$
$x = 0$
Therefore x^3 is reducible

2.

$x^3 + 1 = (x + 1)(x^2 - x + 1)$
Solving for x:
$(x + 1)(x^2 - x + 1) = 0$
$x = -1$
Therefore $x^3 + 1$ is reducible

3.

$x^3 + x = x(x^2 + 1)$
Solving for x:
$x(x^2 + 1) = 0$
$x = 0$
Therefore $x^3 + x$ is reducible

4.

$$x^3 + x^2 = x^2(x+1)$$
Solving for x:
$$x(x^2+1) = 0$$
$$x = 0$$
Therefore $x^3 + x^2$ is reducible

5.

$$x^3 + x^2 + x = x(x^2+x+1)$$
Solving for x:
$$x(x^2+x+1) = 0$$
$$x = 0$$
Therefore $x^3 + x^2 + x$ is reducible

6.

$$x^3 + x^2 + x + 1 = (x+1)(x^2+1)$$
Solving for x:
$$(x+1)(x^2+1) = 0$$
$$x = -1$$
Therefore $x^3 + x^2 + x + 1$ is reducible

iii. A polynomial is irreducible if it's roots are not Integers, $x \notin \mathbb{Z}$. For polynomial of degree 3 to be reducible, it must be a product of degree 1 polynomial and degree 2 polynomial, as can be seen above. Possible factors will be x and $x+1$, with roots 0 and -1.

i. Assume $g(x) = x^3 + x^2 + 1$ is reducible, then $g(0) = 0$ or $g(-1) = 0$

$$g(0) = 0^3 + 0^2 + 1 = 1$$
$$g(-1) = (-1)^3 + (-1)^2 + 1 = 1$$

Therefore $x^3 + x^2 + 1$ is irreducible. □

ii. Assume $g(x) = x^3 + x + 1$ is reducible, then $g(0) = 0$ or $g(-1) = 0$

$$g(0) = 0^3 + 0 + 1 = 1$$
$$g(-1) = (-1)^3 + (-1) + 1 = -1$$

Therefore $x^3 + x + 1$ is irreducible. □

b. i. Let $f(x) = x^2 + 1$, $g(x) = x^3 + x^2 + 1$. Given the irreducible polynomial $p(x) = x^4 + x + 1$

$$\begin{aligned} f(x)g(x) &= (x^2+1)(x^3+x^2+1) \\ &= x^5 + x^4 + x^3 + 1 \end{aligned}$$

where x^5 is:

$$\begin{aligned} p(x) &= 0 \\ x^4 + x + 1 &= 0 \\ x^4 &= x + 1 \\ x^5 &= x^4 x = (x+1)x = x^2 + x \end{aligned}$$

Therefore:

$$\begin{aligned} f(x)g(x) &= x^2 + x + x^4 + x^3 + 1 \\ &= x^4 + x^3 + x^2 + x + 1 \\ &= x + 1 + x^3 + x^2 + x + 1 \\ &= 2x + x^3 + x^2 + 2 \\ &= x^3 + x^2 \end{aligned}$$

ii. Using the fact, that $p(x) = 0$ in $GF(2^4)$, the inverse of $f(x) = x$, where $f(x)g(x) = 1$ in $GF(2^4)$, we can find $g(x)$, as follows:

$$\begin{aligned} x^4 + x + 1 &= 0 \\ x^4 + x &= 1 \\ x(x^3 + 1) &= 1 \end{aligned}$$

Since $f(x) = x$, $g(x) = (x^3 + 1)$, that satisfies $f(x)g(x) = 1$.

c. i. *Proof.* Let $S = (S_3, S_2, S_1, S_0)$ be a 4-byte vector, where S_3, S_2, S_1, S_0 are bytes. Then we have:

$$S(y) = S_3 y^3 + S_2 y^2 + S_1 y + S_0$$

If $S(y)$ is multiplied by y, we get the following:

$$y \cdot S(y) = S_3 y^4 + S_2 y^3 + S_1 y^2 + S_0 y$$

Given $M(y) = y^4 + 1$, we have $y^4 = 1$, therefore $y \cdot S(y)$ results in:

$$\begin{aligned} y \cdot S(y) &= S_3 + S_2 y^3 + S_1 y^2 + S_0 y \\ &= S_2 y^3 + S_1 y^2 + S_0 y + S_3 \end{aligned}$$

We can see that all bytes have shifted left, resulting in new vector $S' = y \cdot S(y) = (S_2, S_1, S_0, S_3)$. □

ii. *Proof.* Given $i \in \mathbb{Z}$, $i \geq 0$, $j \in \mathbb{Z}$, $j \equiv i \pmod 4$, and the fact that we're working with 4 byte arithmetic, where $M(y) = y^4 + 1$, we can show, that $y^i = y^j$.
By definition of divisibility, $i = 4k + j$, $k \in \mathbb{Z}$. Therefore y^i can be defined as:

$$y^i \equiv y^{4k+j} \pmod{y^4 + 1}$$

By deriving j from $i = 4k + j$, we get:

$$\begin{aligned} i &= 4k + j \\ j &= i - 4k \\ j &= i \pmod 4 \end{aligned}$$

Therefore,

$$\begin{aligned} y^i &= y^{4k+j} \\ &= y^{4k} \cdot y^j \\ &= (y^4)^k \cdot y^j \\ &= y^j \end{aligned}$$

Note that, $y^4 = 1$ and $j \equiv i \pmod 4$ with $0 \leq j \leq 3$. □

iii. *Proof.* Let $S = (S_3, S_2, S_1, S_0)$ be a 4 byte vector and let $S(y) = S_3y^3 + S_2y^2 + S_1y + S_0$ be its polynomial form. Given $y^i (i \geq 0)$ and $j \equiv i \pmod 4$, where $0 \leq j \leq 3$, we can show that $y^i \cdot S(y)$ results in a circular left shift of $S(y)$ by j bytes.
As seen in previous proof, $y^i = y^j$, where $j \equiv i \pmod 4$ and $0 \leq j \leq 3$. Proof i. also shows that $y \cdot S(y)$ results in circular left shift by *one* byte. It can also be written as $y^1 \cdot S(y)$. Therefore, $y^i \cdot S(y)$ results in

$$\begin{aligned} y^i \cdot S(y) &= S_3y^{3+i} + S_2y^{2+i} + S_1y^{1+i} + S_0y^i \\ &= S_3y^{3+(i \pmod 4)} + S_2y^{2+(i \pmod 4)} + S_1y^{1+(i \pmod 4)} + S_0y^{i \pmod 4} \end{aligned}$$

Which will always result in circular left shift by j bytes. □

Problem 2 — Arithmetic with the constant polynomial of MixColumns in AES, 13 marks

a.

$$
\begin{aligned}
c_1(x) &= 1 \\
c_2(x) &= x \\
c_3(x) &= x + 1
\end{aligned}
$$

b. i. Let $b(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$, $b(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$, and $(02) = 00000010 = x$. $d(x) = x \cdot b(x)$

$$
\begin{aligned}
(02) \cdot b(x) = x \cdot b(x) &= (x)(b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0) \\
&= b_7x^8 + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x
\end{aligned}
$$

Where

$$
\begin{aligned}
x^8 + x^4 + x^3 + x + 1 &= 0 \\
x^8 &= x^4 + x^3 + x + 1
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
x \cdot b(x) &= b_7(x^4 + x^3 + x + 1) + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x \\
&= b_6x^7 + b_5x^6 + b_4x^5 + (b_7 + b_3)x^4 + (b_7 + b_2)x^3 + b_1x^2 + (b_7 + b_0)x + b_7
\end{aligned}
$$

The result of multiplication, bits $d_i, 0 \leq i \leq 7$:

$$
\begin{aligned}
d_7 &= b_6 \\
d_6 &= b_5 \\
d_5 &= b_4 \\
d_4 &= (b_7 + b_3) \\
d_3 &= (b_7 + b_2) \\
d_2 &= b_2 \\
d_1 &= (b_7 + b_0) \\
d_0 &= b_7
\end{aligned}
$$

ii. Let $b(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$, $b(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$, and $(03) = 00000011 = x + 1$. $e(x) = (x + 1) \cdot b(x)$

$$
\begin{aligned}
(03) \cdot b(x) = (x + 1) \cdot b(x) &= (x + 1)(b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0) \\
&= (b_7x^8 + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x) + \\
&\quad (b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0)
\end{aligned}
$$

Where

$$
\begin{aligned}
x^8 + x^4 + x^3 + x + 1 &= 0 \\
x^8 &= x^4 + x^3 + x + 1
\end{aligned}
$$

$$(x+1)b(x) = (b_7+b_6)x^7 + (b_6+b_5)x^6 + (b_5+b_4)x^5 + (b_7+b_4+b_3)x^4 + (b_7+b_3+b_2)x^3 + \\ +(b_2+b_1)x^2 + (b_7+b_1+b_0)x + (b_7+b_0)$$

The result of multiplication, bits $e_i, 0 \leq i \leq 7$:

$$\begin{aligned}
e_7 &= (b_7+b_6) \\
e_6 &= (b_6+b_5) \\
e_5 &= (b_5+b_4) \\
e_4 &= (b_7+b_4+b_3) \\
e_3 &= (b_7+b_3+b_2) \\
e_2 &= (b_2+b_1) \\
e_1 &= (b_7+b_1+b_0) \\
e_0 &= (b_7+b_0)
\end{aligned}$$

c. i. Computing $t(y) = s(y)c(y) \pmod{y^4+1}$:

$$\begin{aligned}
t(y) &= s(y)c(y) \pmod{y^4+1} \\
&= (s_3y^3+s_2y^2+s_1y+s_0)((03)y^3+(01)y^2+(01)y+(02)) \pmod{y^4+1} \\
&= (03)(s_3y^6+s_2y^5+s_1y^4+s_0y^3)+ \\
&(01)(s_3y^5+s_2y^4+s_1y^3+s_0y^2)+ \\
&(01)(s_3y^4+s_2y^3+s_1y^2+s_0y)+ \\
&(02)(s_3y^3+s_2y^2+s_1y+s_0) \pmod{y^4+1}
\end{aligned}$$

Where;

$$\begin{aligned}
y^6 &= y^4y^2 = y^2 \\
y^5 &= y^4y = y
\end{aligned}$$

Therefore we get:

$$
\begin{aligned}
t(y) = {} & (03)(s_3y^2 + s_2y + s_1 + s_0y^3)+ \\
& (01)(s_3y + s_2 + s_1y^3 + s_0y^2)+ \\
& (01)(s_3 + s_2y^3 + s_1y^2 + s_0y)+ \\
& (02)(s_3y^3 + s_2y^2 + s_1y + s_0) \pmod{y^4 + 1} \\
\\
& = ((03)s_3y^2 + (03)s_2y + (03)s_1 + (03)s_0y^3)+ \\
& ((01)s_3y + (01)s_2 + (01)s_1y^3 + (01)s_0y^2)+ \\
& ((01)s_3 + (01)s_2y^3 + (01)s_1y^2 + (01)s_0y)+ \\
& ((02)s_3y^3 + (03)s_2y^2 + (02)s_1y + (02)s_0) \pmod{y^4 + 1} \\
\\
& = ((02)s_3y^3 + (01)s_2y^3 + (01)s_1y^3 + (03)s_0y^3))+ \\
& ((03)s_3y^2 + (03)s_2y^2 + (01)s_1y^2 + (01)s_0y^2))+ \\
& ((01)s_3y + (03)s_2y + (02)s_1y + (01)s_0y))+ \\
& ((01)s_3 + (01)s_2 + (03)s_1 + (02)s_0) \pmod{y^4 + 1} \\
\\
& = ((02)s_3 + (01)s_2 + (01)s_1 + (03)s_0))y^3+ \\
& ((03)s_3 + (03)s_2 + (01)s_1 + (01)s_0))y^2+ \\
& ((01)s_3 + (03)s_2 + (02)s_1 + (01)s_0))y+ \\
& ((01)s_3 + (01)s_2 + (03)s_1 + (02)s_0) \pmod{y^4 + 1}
\end{aligned}
$$

ii. $t(y)$ written in matrix form:

$$
\begin{bmatrix} t_0 \\ t_1 \\ t_2 \\ t_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 03 & 03 \\ 03 & 01 & 01 & 01 \end{bmatrix} \begin{bmatrix} s_0 \\ s_1 \\ s_3 \\ s_3 \end{bmatrix}
$$

Problem 3 — Error propagation in block cipher modes, 12 marks

a.
   i. **ECB mode**: only M_i will be affected. ECB mode is simply a shift cipher, each block is a substitution.
   ii. **CBC mode**: M_i and M_{i+1} will be affected, because $M_i = D_K(C_i) \oplus C_{i-1}$, where D_K is a decryption function.
   iii. **OFB mode**: only M_i is affected, because decryption for M_i is done using previous state.
   iv. **CFB mode**: M_i and M_{i+1} will be affected, because decryption of M_{i+1} depends on C_i, which is corrupted.
   v. **CTR mode**: only M_i will be affected, because current counter value is XOR with C_i, which will result in corrupted M_i.

b. Since the error occured **before** any encryption, the message will be encrypted and decrypted with no errors, however, the corresponding decrypted plaintext M_i' will be affected.

Problem 4 — Flawed MAC designs, 24 marks

a. i. Given $(M_1, \texttt{PHMAC}_K\texttt{(}M_1\texttt{)})$, where $\texttt{PHMAC}_K\texttt{(}M_1\texttt{)}$ is the hash for $K||M_1 = K||P_1||P_2||\dots||P_L$ and $M_2 = M_1||X$, X is a n bit block.

$$\begin{aligned}\texttt{PHMAC}_K\texttt{(}M_2\texttt{)} &= \texttt{ItHash(}K||M_2\texttt{)}\\ &= \texttt{ItHash(}K||M_1||X\texttt{)}\\ &= \texttt{ItHash(PHMAC}_K\texttt{(}M_1\texttt{)}||X\texttt{)}\end{aligned}$$

Therefore, this defeats computational resistance of $\texttt{PHMAC}$, because $\texttt{PHMAC}_K\texttt{(}M_2\texttt{)}$ can be computed without knowledge of the key K.

ii. Given $(M_1, \texttt{AHMAC}_K\texttt{(}M_1\texttt{)})$, where $\texttt{AHMAC}_K\texttt{(}M_1\texttt{)}$ is the hash for $M_1||K = P_1||P_2||\dots||P_L||K$, we can find $(M_2, \texttt{AHMAC}_K\texttt{(}M_2\texttt{)})$ without knowledge of the key K.

Assume that $\texttt{ItHash}$ is not weakly collision resistant, therefore it is computationally feasible to find such M_2 that $M_2 \neq M_1$ and $\texttt{AHMAC}_K\texttt{(}M_2\texttt{)} = \texttt{AHMAC}_K\texttt{(}M_1\texttt{)}$. Given that pair $(M_1, \texttt{AHMAC}_K\texttt{(}M_1\texttt{)})$ is already known and $\texttt{AHMAC}_K\texttt{(}M\texttt{)}$ doesn't depend on K, just M, there exists $M_2 \neq M_1$ and $\texttt{AHMAC}_K\texttt{(}M_1\texttt{)} = \texttt{AHMAC}_K\texttt{(}M_2\texttt{)}$. Which therefore defeats computational resistance.

b. i. Given that the attacker knows $(M_1, \texttt{CBC-MAC}_K\texttt{(}M_1\texttt{)})$ and $(M_2, \texttt{CBC-MAC}_K\texttt{(}M_2\texttt{)})$ with $M_2 = \texttt{CBC-MAC}_K\texttt{(}M_1\texttt{)}$, we can find $\texttt{CBC-MAC}_K\texttt{(}M_3\texttt{)}$, where $M_3 = M_1||0^n$. Since M_1 and M_2 are single block messages, we can show their $\texttt{CBC-MAC}$ values:

$$\begin{aligned}\texttt{CBC-MAC}_K\texttt{(}M_1\texttt{)} &= E_K(0^n \oplus M_1)\\ \texttt{CBC-MAC}_K\texttt{(}M_2\texttt{)} &= E_K(0^n \oplus M_2)\\ &= E_K(0^n \oplus \texttt{CBC-MAC}_K\texttt{(}M_1\texttt{)})\end{aligned}$$

Therefore, $\texttt{CBC-MAC}_K\texttt{(}M_3\texttt{)}$ evaluates to:

$$\begin{aligned}\texttt{CBC-MAC}_K\texttt{(}M_3\texttt{)} &= \texttt{CBC-MAC}_K\texttt{(}M_1||0^n\texttt{)}\\ &= E_K(0^n \oplus M_1)\\ &= E_K(E_K(0^n \oplus M_1) \oplus 0^n)\\ &= E_K(0^n \oplus E_K(0^n \oplus M_1))\\ &= E_K(0^n \oplus \texttt{CBC-MAC}_K\texttt{(}M_1\texttt{)})\\ &= E_K(0^n \oplus \texttt{CBC-MAC}_K\texttt{(}M_1\texttt{)})\\ &= \texttt{CBC-MAC}_K\texttt{(}M_2\texttt{)}\end{aligned}$$

This violates computational resistance, because there's message $M_3 \neq M_2$ and $\texttt{CBC-MAC}_K\texttt{(}M_3\texttt{)} = \texttt{CBC-MAC}_K\texttt{(}M_2\texttt{)}$.

ii. Given $(M_1, \texttt{CBC-MAC}_K\texttt{(}M_1\texttt{)})$, $(M_2, \texttt{CBC-MAC}_K\texttt{(}M_2\texttt{)})$, $(M_3, \texttt{CBC-MAC}_K\texttt{(}M_3\texttt{)})$, $M_3 = M_1||X$ (X is n bit block), we can find $\texttt{CBC-MAC}_K\texttt{(}M_4\texttt{)}$, where $M_4 = M_2||Y$ and Y is:

$$Y = \texttt{CBC-MAC}_K\texttt{(}M_1\texttt{)} \oplus \texttt{CBC-MAC}_K\texttt{(}M_2\texttt{)} \oplus X$$

Computing $\texttt{CBC-MAC}_K(M_1)$, $\texttt{CBC-MAC}_K(M_2)$, and $\texttt{CBC-MAC}_K(M_3)$:

$$\begin{aligned}
\texttt{CBC-MAC}_K(M_1) &= E_K(0^n \oplus M_1) \\
\texttt{CBC-MAC}_K(M_2) &= E_K(0^n \oplus M_2) \\
\texttt{CBC-MAC}_K(M_3) &= E_K(0^n \oplus M_1) \\
&= E_K(E_K(0^n \oplus M_1) \oplus X) \\
&= E_K(\texttt{CBC-MAC}_K(M_1) \oplus X)
\end{aligned}$$

Computing $\texttt{CBC-MAC}_K(M_4)$:

$$\begin{aligned}
\texttt{CBC-MAC}_K(M_4) &= E_K(0^n \oplus M_2) \\
&= E_K(E_K(0^n \oplus M_2) \oplus Y) \\
&= E_K(E_K(0^n \oplus M_2) \oplus \texttt{CBC-MAC}_K(M_1) \oplus \texttt{CBC-MAC}_K(M_2) \oplus X) \\
&= E_K(\texttt{CBC-MAC}_K(M_2) \oplus \texttt{CBC-MAC}_K(M_1) \oplus \texttt{CBC-MAC}_K(M_2) \oplus X) \\
&= E_K(\texttt{CBC-MAC}_K(M_1) \oplus X) \\
&= \texttt{CBC-MAC}_K(M_3)
\end{aligned}$$

Therefore, we can see that $\texttt{CBC-MAC}_K(M_4) = \texttt{CBC-MAC}_K(M_3)$ given that $M_4 \neq M_3$. This defeats computational resistance.