Cryptography Engineering Midterm

Problem 1

In $\mathbb{GF}(2^4)$ Compute $f(x)+g(x)$, $f(x)-g(x)$, $f(x)\times g(x)$ *mod* $P(x)$, where $P(x)=x^4+x+1$.

a) $f(x)=x^2+1$, $g(x)=x^3+x^2+1$.

b) $f(x)=x^2+1$, $g(x)=x+1$.

(a)

$f(x)+g(x)$: $[0101]+[1101]=[1000] \Rightarrow x^3$ #

$f(x)-g(x)$: $[0101]-[1101]=[1000] \Rightarrow x^3$ #

$f(x)\times g(x)$:

$$\begin{array}{r} x^2+0x+1 \\ \hline x^3+x^2+0x+1 \\ \hline x^5+0+x^3 \\ \hline x^5+x^4+x^3+0x^2+0x+1 \end{array}$$

$$\begin{array}{r|l} & 11 \\ \hline 10011 & 111001 \\ & 10011 \\ \hline & 11100 \end{array}$$

$\Rightarrow x^3+x^2$ #

(b)

$f(x)+g(x)$: $[101]+[011]=[110] \Rightarrow x^2+x$ #

$f(x)-g(x)$: $[101]-[011]=[110] \Rightarrow x^2+x$ #

$f(x)\times g(x)$:

$$\begin{array}{r} x^2+0x+1 \\ \hline x+1 \\ \hline x^2+0+1 \\ \hline x^3+0+x \\ \hline x^3+x^2+x+1 \end{array}$$

$\Rightarrow x^3+x^2+x+1$ #

Problem 2

In $\mathbb{GF}(2^8)$, $f(x) = x^7 + x^5 + x^4 + x + 1$ and $g(x) = x^3 + x + 1$.

a) Calculate $f(x) + g(x)$, $f(x) - g(x)$, $f(x) \times g(x)$ *mod* $m(x)$, where $m(x) = x^8 + x^4 + x^3 + x + 1$.

b) Show that $f(x) = x^4 + 1$ is reducible over $\mathbb{GF}(2^8)$.
Hint: it can be written as the product of two polynomials.

(a) $f(x) + g(x)$: $x^7 + x^5 + x^4 + x^3$ #

$f(x) - g(x)$: $x^7 + x^5 + x^4 + x^3$ #

$f(x) \times g(x)$:

```
  1 0 1 1 0 0 1 1
      1 0 1 1
  -----------------
  1 0 1 1 0 0 1 1
    1 0 1 1 0 0 1 1
      1 0 1 1 0 0 1 1
  -----------------
  1 0 0 0 1 0 0 1 1 0 1
```

```
                  1
  100011011 | 1000100110
             | 100011011
             -----------
               10000 1
```

= $x^5 + 1$ #

(b) $(x^4 + 1) = (x + 1)(x^3 + x^2 + x + 1)$

$f(x)$ 可被表示為兩多項式乘積，∴ 它是 reducible over $GF(2^8)$ #

Problem 3

Consider the field $\mathbb{GF}(2^4)$ with the irreducible polynomial $P(x) = x^4 + x + 1$. Find:

a) the inverse of $f(x) = x$ and

b) the inverse of $g(x) = x^2 + x$ by trial and error.

(a) let inverse of $f(x)$ be $f^{-1}(x)$

$f(x) \cdot f^{-1}(x) \equiv 1 \bmod P(x) \Rightarrow x \cdot f^{-1}(x) \equiv 1 \bmod P(x)$

elements in $GF(2^4)$:

① $x \cdot 0$

② $x \cdot 1$

③ $x \cdot x = x^2$

④ $x \cdot (1+x)$

⑤ $x \cdot x^2$

⑥ $x \cdot (x^2+1)$

⑦ $x \cdot (x^2+x)$

⑧ $x \cdot (x^2+x+1)$

⑨ $x \cdot x^3 = x^4 \bmod (x^4+x+1) = x+1$

⑩ $x \cdot (x^3+1) = x^4+x \bmod (x^4+x+1) = 1$ ✓

Ans: x^3+1 #

(b) $(x^2+x)\, g^{-1}(x) \equiv 1 \bmod (x^4+x+1)$

① $(x^2+x) \cdot 0$

② $(x^2+x) \cdot 1$

③ $(x^2+x) \cdot x$

④ $(x^2+x) \cdot (1+x)$

⑤ $(x^2+x) \cdot x^2 = x^4+x^3$

⑥ $(x^2+x)(x^2+1) \Rightarrow (x^4+x^3+x^2+x) \bmod (x^4+x+1)$
$= x^3+x^2+1$

⑦ $(x^2+x)(x^2+x) \Rightarrow x^4+x^2 \bmod (x^4+x+1) = x^2+x+1$

⑧ $(x^2+x)(x^2+x+1) \Rightarrow x^4+x \bmod (x^4+x+1) = 1$

Ans: x^2+x+1 #

Problem 4

In $\mathbb{GF}(2^8)$, find:

a) $(x^3+x^2+x)(x^3+x^2)^{-1}\ mod\ (x^8+x^4+1) =$

b) $(x^6+x^3+1)(x^4+x^3+1)\ mod\ (x^8+x^4+1) =$

(a) $(x^3+x^2)\cdot(x^3+x^2)^{-1} \bmod (x^8+x^4+1) = 1$

原式 $= \left[(x^3+x^2)\cdot(x^3+x^2)^{-1} + x\cdot(x^3+x^2)^{-1}\right] \bmod (x^8+x^4+1)$

$= 1 + x\cdot(x^3+x^2)^{-1} \bmod (x^8+x^4+1)$

$s(x)(x^3+x^2) + t(x)\cdot(x^8+x^4+1) = 1$

let $t(x)=1$, $s(x) = x^5+x^4+x^3+x^2 = (x^3+x^2)^{-1}$

$\therefore$ 原式 $= 1+x\cdot(x^5+x^4+x^3+x^2) \bmod (x^8+x^4+1)$

$= x^6+x^5+x^4+x^3+1$ #

(b) 原式 $= x^{10}+x^9+x^6+x^7+x^6+x^3+x^4+x^3+1$

$= x^{10}+x^9+x^7+2x^6+x^4+2x^3+1$

$$
\begin{array}{r|l}
 & \quad\quad\quad\quad 1\ 1 \\
\hline
100010001 & 1\ 1\ 0\ 1\ 2\ 0\ 1\ 2\ 0\ 0\ 1 \\
 & \underline{1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1} \\
 & \quad 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 0 \\
 & \quad \underline{1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1} \\
 & \quad\quad 1\ 1\ 1\ 1\ 0\ 1\ 1\ 1
\end{array}
$$

$\Rightarrow x^7+x^6+x^5+x^4+x^2+x+1$ #

Problem 5

Regarding the mix column operation of the AES round function, it is performed with a pre-defined matrix, i.e.,

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

a) Explain why this **mix column operation** can be implemented with a simple look up table and XOR.

b) Apply the same idea used above, explain why the **byte substitution**, **shift row** and **mix column** can be combined and implemented as a simple look-up table operation. And then the whole AES is implemented by look up table and few XORs.

(a) Mix columns 是由一個 pre-defined matrix 和一個 state matrix 在 galois field 進行矩陣相乘，而 pre-defined matrix 是已知固定的。兩個 matrix 的元素可由二進制多項式表示，Pre-defined matrix 的每一 row 和 state matrix 的一行 column 對應元素相乘後再進行相加，而加法運算在 galois field 中可以 XOR 進行運算，乘法運算則可透過查表的方式。因為 state matrix 的元素值範圍是有限的，通常是 256，因此可透過建表，將 256 種可能值和某數相乘的值存起來，在之後的運算透過查表方式則能較有效率執行。

(b) 在 byte substitution 中，會透過已計算好的 s-box lookup table 去取代 state matrix 中的每個 byte，substitution 的用意是要引入非線性的運算，增加安全性。運算的方式是先對 state matrix 每一 byte 求出其在有限場乘法反元素，再經過 affine transformation，因為 state matrix 的 byte 的數值是有限的，因此可先進行計算將結果存成表，透過查表也可以減少運算的複雜性。

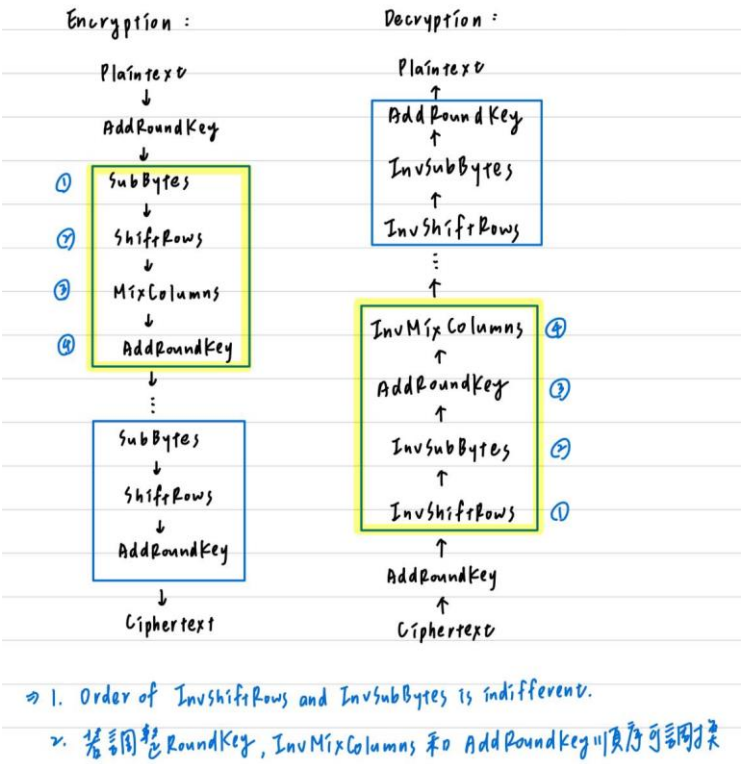
而在 shift row 中，每一列會向左位移一定的位數，因為移動的距離是固定的，因此也可以透過事先計算好位移結果並建表儲存來達成。透過 XOR 和查表的運算，可以同時保證 AES 的安全性和效能。

Problem 6

In order to make AES encryption and decryption more similar in structure, the MixColumns operation is missing in the last round.

Explain how to take **advantage** of this property to share some of the code (for software implementation) or chip area (for hardware implementation) for AES encryption and decryption.

Hint: Denote InvSubBytes, InvShiftRows, and InvMixColumns as the inverses of SubBytes, ShiftRows, and MixColumns operations, respectively. Try to show:
1. The order of InvSubBytes and InvShiftRows is indifferent;
2. The order of AddRoundKey and InvMixColumns can be inverted if the round key is adapted accordingly.



Encryption 和 decryption 在 operation 上若有相似的結構，則可以用相同的 code base 或是 hardware module 來實現，降低設計上的複雜度並可以減少 chip area 或 code size，節省開發所需的資源和成本。

Problem 7

Under **what circumstances** could you choose 3DES over AES? Also, what are the **advantages** of choosing AES over 3DES? Is 3DES **susceptible** to Meet-in-the-Middle Attacks like 2DES? Please explain.

1. 甚麼時後會選擇用 3DES 而非 AES？

可能像是有些電子支付、信用卡仍是使用過去較舊的系統，因此系統可能無法支援 AES，只有支持 3DES 的加密方式，此時選擇 3DES 可確保系統兼容性。

2. AES 相較於 3DES 有甚麼優點？

3DES 的加密區塊大小是 64bits，而 AES 的則是 128bits，加密區塊較大，可以不用那麼頻繁的更換 key，且資料切成的 block 長度較大也可使密文被破解的機率較低。另外 AES 的 key 長度可以是 128, 192, 256 bits，不像 3DES 只限於 168bits。總體來說，AES 有較高的安全性、效率和性能。

3. 3DES 也會容易受到 meet in the middle attack(MITM)的影響嗎？

是，3DES 仍可能受 MITM 攻擊的影響，雖然相較於單一 DES，3DES 的 key 長度較長，在攻擊的成本、時間和複雜性較高，但仍可以透過 MITM 進行攻擊。

$$P = D_{K1}(E_{K2}(D_{K3}(C))$$

$$C = E_{K3}(D_{K2}(E_{K1}(P))$$

DES 的一個 key 長度是 56，總共會需要$2^{56} + 2^{112} \approx 2^{112}$個 operations 和$2^{56}$個 storage 去破解。

Problem 8

If a company has encrypted its most sensitive data with a key held by the chief technology officer and that person was fired, the company would want to change its encryption key. Describe **what would be necessary** to revoke the old key and deploy a new one.
Hint: NIST Special Publication 800-57 Part 1

首先，要將舊密鑰在所有系統中撤銷其訪問權限並標記為無效，因此可能需要更新公司的金鑰管理系統，接著需要生成一個足夠長且夠隨機的新密鑰，它要符合規定的密碼強度需求。

生成新密鑰後，要將其安全的分發給需要訪問該數據的授權方，可能會需要用到授權用戶的公共金鑰來加密新的密鑰。另外，新密鑰也要安全地儲存在公司系統中並實施存取權的控制。

最後，確保任何使用舊金鑰的系統都更新為使用新金鑰，這可能需要更新系統的加密算法，部署完成後則需要進行測試，確保新系統是可以正常且安全的運作的。

Problem 9

Bitdiddle Inc. requires every employee to have an RSA public key. It also requires the employee to change his or her RSA key at the end of each month.

a) Alice just started working at Bitdiddle, and her first public key is (n, e) where n is the product of two safe primes, and $e = 3$.
Whenever a new month starts, Alice (being lazy) changes her public key as little as possible to get by the auditors. What she does, in fact, is just to advance her public exponent e to the next prime number. So, month by month, her public keys look like:

$$(n, 3), (n, 5), (n, 7), (n, 11), ...$$

Explain how Alice's laziness might get her in trouble.

b) The next year, Alice tries a different scheme.
In January, she generates a fresh public key (n, e) where n is the product of two primes, p and q. In February, she advances p to the next prime p_l after p, and q to the next prime q_l after q, and sets her public key to (n_l, e_l) for a suitable e_l. Similarly, in March, she advances p_l to p_2 and q_l to q_2, and so on.

Explain how Alice's scheme could be broken.

(a) $N = pq$, p and q are prime numbers.

$de \equiv 1(mod\ r) = 1(mod(p-1)(q-1))$

$c^d \equiv n(mod\ N)$

Alice 用的 e 數值很小，且有固定可預測的 pattern，讓 attacker 能夠更容易預測他未來的 keys。若 attacker 知道 Alice 的 public key 則他有 N 和 e，且 Alice 的 e 值是可預測的，而 N 值則是每次都固定的，attacker 可能可以透過多次嘗試，最後破解出 N 是由哪兩個質數相乘的。

假設 Alice 將同一段 message m 用 public keys (n,3), (n,5)和(n,7)加密，則 attacker 可以透過解三個同餘式線性系統來求得 m。

(b) 若 attacker 長期觀察 Alice 的 public key，他可能可以觀察到 N 值的規律並推敲出 p 和 q，且若一開始選擇的 p 和 q 值很小，也會增加破解的容易性。

因為 p2 和 p1 很接近，q2 和 q1 很接近，attacker 可能可以藉由每次相減:

$n_2 - n_1 = (p_2 \times q_2) - (p_1 \times q_1)$ 因為$n_2 - n_1$很小，可能可以透過 Fermat's factorization method 分解出 p,q.

Problem 10

The Advanced Encryption Standard (AES) requires not only the MixColumns step during encryption but also the Inverse MixColumns during decryption.

a) Given the matrix for the Inverse MixColumns operation:

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix}$$

Describe how the matrix multiplication in the Inverse MixColumns step is performed within AES decryption. Your answer should include a **brief explanation of the arithmetic** used in the finite field $\mathbb{GF}(2^8)$ and **how it differs from standard matrix multiplication**.

b) Multiplications in $\mathbb{GF}(2^8)$ can be complex. However, they can be simplified using repeated application of simpler operations. Explain how multiplication by 9, 11, 13, and 14 in $\mathbb{GF}(2^8)$ can be implemented using the simpler multiplication by 2 and addition (XOR). Provide the **mathematical expressions** that represent these multiplications.

c) Discuss how the use of lookup tables (LUTs) can further simplify the implementation of Inverse MixColumns in AES decryption. What are the **advantages** and potential **drawbacks** of using LUTs for this purpose?

(a) 09、0B、0D、0E　　(11) (13) (14)

state matrix

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & \cdots & \\ S_{2,0} & \cdots & & \\ S_{3,0} & \cdots & & \end{bmatrix}$$

$0E \times S_{0,0} \oplus 0B \times S_{1,0} \oplus 0D \times S_{2,0} \oplus 09 \times S_{3,0}$ = 新state matrix的(0,0)

和一般矩陣乘法的位置關係相同，但因為是在GF(2^8)

每个元素是以位的二進制

乘法運算比對於每bit：1×1，0×0=0，1×0，0×1=1

且若乘積超過 x^7 次方，則需要mod運算，控制在8位內

而加法則變為XOR運算，對於每行每列計算得新matrix.

(b)

value 左移 3 bits

① multiplication by 9 (1001) = (value multiply 2)^3 ⊕ value

② multiplication by 11 (1011) = (value multiply 2)^3 ⊕ value ⊕ (value multiply 2)

③ multiplication by 13 (1101) = (value multiply 2)^3 ⊕ (value multiply 2)^2 ⊕ value

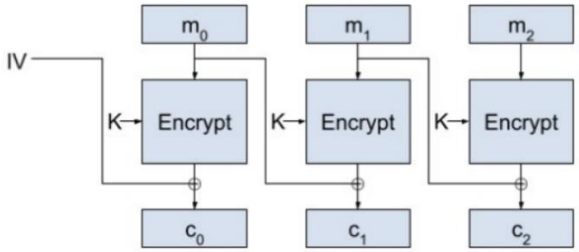
④ multiplication by 14 (1110) = (value multiply 2)^3 ⊕ (value multiply 2)^2 ⊕ (value multiply 2)

上圖的 (value multiply 2)^3 是指 multiply by 2(multiply by 2(multiply by 2 (x)))
(value multiply 2)^2 = multiply by 2(multiply by 2 (x))

(c) 因為 09、11、13、14 都可以用 multiplication by 2 加上 XOR 表示，而 state matrix 的元素值的範圍是固定的，因此我們可以透過事先將 2 和 256 個值做乘法運算，並將結果存成表。
優點是：在之後的每次運算都可以直接查表得到結果，而不需要重複再運算一次，節省時間及運算資源和成本，code implementation 也較為簡易。
潛在的缺點：儲存成 table 需要額外的 memory，若 input 和 output values 的值不斷擴大，可能會達到 memory 的極限。且若數據需要頻繁的更改，則需要不斷更新查表，或是數據關係較複雜，則查表不太有彈性。

Problem 11

In class we saw several modes such as ECB, CBC and CTR mode. Let's look at another possible mode of operation "plaintext block chaining" (PBC) which is similar to cipher block chaining (CBC) but allows for encryption in parallel:



Unfortunately, PBC mode is not secure.

To see this, show how an attacker who knows IV, C_0, C_1, C_2 (which are public) and also knows that $m_1 = m_2 = x$ (for a known x) can easily compute m_0.

已知 $IV, C_0, C_1, C_2, m_1 = m_2 = x$

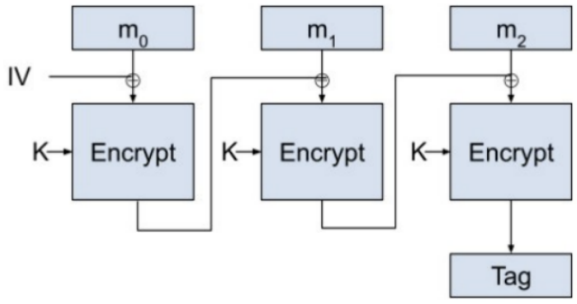
$$\begin{cases} Enc(m_0) \oplus IV = C_0 \\ Enc(m_1) \oplus m_0 = C_1 \\ Enc(m_2) \oplus m_1 = C_2 \end{cases} \Rightarrow \begin{cases} Enc(m_0) \oplus IV = C_0 \\ Enc(x) \oplus m_0 = C_1 \\ Enc(x) \oplus x = C_2 \end{cases}$$

则 $Enc(x) = x \oplus C_2$，因为 x 和 C_2 都已知，可得 $Enc(x)$

$m_0 = C_1 \oplus Enc(x)$ ∵ $Enc(x)$ 和 C_1 已知则可算出 m_0 #

Problem 12

Alice is trying to design a MAC using a block cipher. She decides to use the following construction, which is essentially just CBC encryption, throwing away all but the final block.



Unfortunately, this construction is not secure.

Describe how to produce an existential forgery against this MAC scheme.

Hint 1: Start with two messages M_1 and M_2 (not to be confused with the individual blocks of a message in the diagram above) for which you know the outputs (IV_1, T_1) and (IV_2, T_2). Produce another message M_3 for which (IV_1, T_2) will be the MAC. M_3 will be close to the concatenation $M_1||M_2$, but with one block altered.

Hint 2: There is also a way to produce a forgery with only one known block if you look closely.

Caution: The blocks $m_0, m_1, \ldots$ in the diagram are distinct from the complete messages $M_1, M_2, \ldots$

假設 attacker 知道兩個訊息 M1 和 M2，以及他們對應的 IV1, IV2 和標籤 T1, T2，則 attacker 可以創造一個訊息 M3，使它近似於 M1 和 M2 串接在一起的 string，但將其中一區塊作修改，這個修改仍要確保用 IV1 加密的 M3 仍可以產出和 T2 相符的 tag。

Attacker 會將 M3 連同 IV1 發送給 receiver 作為 MAC，則 receiver 會用 IV1 解密 M3 並將結果產生的 tag 與 T2 比較是否相符，若相符，則 attacker 成功為 M3 偽造一個虛假的 MAC。