

Problem 1

In $\mathbb{GF}(2^4)$ Compute $f(x) + g(x)$, $f(x) - g(x)$, $f(x) \times g(x) \bmod P(x)$, 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)

$$(1) f(x) + g(x) = (x^2 + 1) + (x^3 + x^2 + 1) \\ = x^3 \quad \cancel{\neq}$$

$$(2) f(x) + g(x) = f(x) - g(x) = x^3 \quad \cancel{\neq}$$

$$(3) f(x) \times g(x) = (x^2 + 1) \cdot (x^3 + x^2 + 1) \\ = x^5 + x^4 + x^3 + 1$$

$$(x^5 + x^4 + x^3 + 1) \bmod (x^4 + x + 1)$$

$$= x^3 + x^2 \quad \cancel{\neq}$$

$$\begin{array}{r} 11 \\ 10011 \overline{) 11001} \\ \underline{10011} \\ 11111 \\ \underline{10011} \\ 1100 \end{array}$$

b)

$$(1) f(x) + g(x) = (x^2 + 1) + (x + 1) = x^2 + x + 2$$

$$(2) f(x) - g(x) = f(x) + g(x) = x^2 + x + 2$$

$$(3) f(x) \cdot g(x) = (x^2 + 1)(x + 1) \\ = x^3 + x^2 + x + 1$$

$$x^3 + x^2 + x + 1 \mod (x^4 + x + 1) = x^3 + x^2 + x + 1$$

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 $f(x) + g(x)$, $f(x) - g(x)$, $f(x) \times g(x) \bmod m(x)$, 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.

$$\begin{aligned}
 a) \quad f(x) + g(x) &= (x^7 + x^5 + x^4 + x + 1) + (x^3 + x + 1) \\
 &= x^7 + x^5 + x^4 + x^3 \\
 f(x) - g(x) &= f(x) + g(x) = x^7 + x^5 + x^4 + x^3 \\
 f(x) \cdot g(x) &= (x^7 + x^5 + x^4 + x + 1) \cdot (x^3 + x + 1) \\
 &= x^{10} + x^6 + x^3 + x^2 + 1 \\
 (x^{10} + x^6 + x^3 + x^2 + 1) \bmod (x^8 + x^4 + x^3 + x + 1) \\
 &= x^5 + 1
 \end{aligned}$$

$$\begin{array}{r}
 100011011 \overline{) 10001001101} \\
 \underline{100011011} \\
 100001
 \end{array}$$

$$\begin{aligned}
 b) \quad f(x) = x^4 + 1 &= (x^2 + 1)(x^2 + 1) \\
 x^4 + \cancel{x^2} + 1 \\
 f(x) \text{ 可以拆成两个多项式在 } \mathbb{GF}(2^8)
 \end{aligned}$$

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)

$$f(x) f^{-1}(x) \equiv 1 \pmod{P(x)}$$

$$f(x) \cdot f^{-1}(x) = x^4 + x + 1 + 1$$

$$x \cdot (x' + 1) = x^4 + x$$

$$f^{-1}(x) = x^3 + 1 \quad \text{A}$$

(b)

$$x^4 + x + 1 + 1 = (x^2 + x) g^{-1}(x)$$

$$(x^2 + x) g^{-1}(x) = x^4 + x$$

$$g^{-1}(x) = x^2 + x + 1 \quad \text{A}$$

Problem 4

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

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

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

$$\begin{aligned} (x^3 + x^2)^{-1} (x^3 + x^2) &= 1 \neq (x^8 + x^4 + 1) \\ (x^3 + x^2)(x^3 + x^2)^{-1} &= x^8 + x^4 \\ (x^3 + x^2)^{-1} &= x^5 + x^4 + x^1 + x^2 \\ (x^3 + x^2 + x)(x^3 + x^2)^{-1} &= (x^3 + x^2 + x)(x^5 + x^4 + x^1 + x^2) \\ &= x^8 + x^6 + x^5 + x^3 \\ &= x^8 + x^6 + x^5 + x^3 \bmod x^8 + x^4 + 1 \\ &= x^6 + x^5 + x^4 + x^3 + 1 \end{aligned}$$

$$\begin{array}{r} 1 \\ 1001001 \overline{) 1101000} \\ \underline{1001001} \\ 1110001 \end{array}$$

b)

$$\begin{aligned} (x^6 + x^3 + 1)(x^4 + x^3 + 1) &= x^{10} + x^9 + x^6 + x^7 + x^6 + x^3 + x^4 + x^3 + 1 \\ &= x^{10} + x^9 + x^7 + x^4 + 1 \\ &= x^{10} + x^9 + x^7 + x^4 + 1 \bmod x^8 + x^4 + 1 \\ &= x^2 + x^6 + x^5 + x^4 + x^2 + x + 1 \end{aligned}$$

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 column operation 其元素的乘法是在有限域 $GF(2^8)$ 進行的
乘以 2:如果最高位為 0，就 left shift 一位，如果最高位是 1(左移會溢位)需要左移後跟 0x1b 進行 XOR。
乘以 3:為乘 2 的結果跟原始數字 XOR
這些操作因為 $GF(2^8)$ 的有限性。只有 256 可能的字節，因次可以先預先計算並創建 lookup Table，一旦乘法就是用 XOR 來組合每一列的結果這樣就有效率。
- b) byte-substitution:每個字節經過非線性進行替換，可以抵抗線性分析。
shift row:狀態矩陣根據不同的偏移量進行 cyclically shifted.增加了 diffusion of bytes across column.
mix column:如同 a 的描述每個 column 用 $GF(2^8)$ 的固定多項式去 transform 可以用 lookup tables 和 XOR 的操作實現。
這三個操作的結果合併到一個 lookup table AES 的運算就可以透過單一的查表來執行，可以簡化運算過程。

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 過程大致對稱，最後一輪省略了 MIXColumn 而已

1. The order of $InvSubBytes$ and $InvShiftRows$ is indifferent:

操作 $InvSubBytes$ 和 $InvShiftRows$ 是 **SubBytes** 和 **ShiftRows** 的逆操作。在解密時 $InvSubBytes$ 和 $InvShiftRows$ 的順序不影響最後結果，它們獨立的對其輸入的數據進行操作。所以，在下一次操作前執行兩個步驟，可以優化 chip area (for hardware implementation)

2. The order of $AddRoundKey$ and $InvMixColumns$ can be inverted if the round key is adapted accordingly:

在 decryption 時 $AddRoundKey$ 在 $InvMixColumns$ 後面使用，但如果適當的調整 round key，這兩個順序就可以交換，例如可以預先計算 round key 使他們考慮 MixColumn 的操作，將 Mixcolumn 的 transform 用到 key 中，這樣可以在 $InvMixColumns$ 前 $AddRoundKey$ 不影響結果，這樣 Encryption 和 decryption 更一致，就可以在操作這兩個過程中 **share some of the code (for software implementation)**

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.

Under what circumstances could you choose 3DES over AES:

3DES 可以更容易的與系統兼容，不用修改太多，有一些舊設備可能只支持 3DES 加上有些行業的標準也要用 3DES

what are the advantages of choosing AES over 3DES:

AES 安全性比較高，因為他用更大的 key，其在運行也更快更高效，特別是在處理大數據時，因為其是新的標準，其可用性和支持性比較好。

Is 3DES susceptible to Meet-in-the-Middle Attacks like 2DES?

是，3DES 容易遭遇 Meet-in-the-Middle Attacks，雖然 3DES 比 2DES 安全因為它增加了額外的一輪加密，但還是存在被攻擊的風險，AES 比較能抵禦這種攻擊。

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

如果公司將最敏感的數據加密且密鑰由技術總監持有，而該人員被解僱，公司需要更換加密密鑰。首先，識別哪些數據使用了舊密鑰加密並評估影響範圍。然後，生成符合安全標準的新密鑰，並用新密鑰重新加密所有受影響的數據。接著，在所有相關系統中部署新密鑰，替換舊密鑰，並將舊密鑰標記為撤銷。通知所有相關人員和系統管理員密鑰更新事宜，並進行密鑰更新後的審計和檢查，確保新密鑰已正確部署並運行正常。

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:

$$(n, 3), (n, 5), (n, 7), (n, 11), \dots$$

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_1 after p , and q to the next prime q_1 after q , and sets her public key to (n_1, e_1) for a suitable e_1 . Similarly, in March, she advances p_1 to p_2 and q_1 to q_2 , and so on.

Explain how Alice's scheme could be broken.

- a) Alice 只是在公鑰指數 e 從 3 逐步遞增到下一個質數，這樣重用 modulus n ，因為 n 不便所以同一對質數 pq 在多個月中重複使用，然後容易受到攻擊，因為攻擊者會用相同的 n 值不同的 e 值去推斷 pq 是啥
- b) 第二種方法是每個月更新 pq 來生成公鑰，這樣的問題是攻擊者可以推斷出每個月更新質數的關聯，如果讓攻擊者知道一對 pq 就可以推出其他月的 pq ，所以還是有其他隱患。

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?

Hint 1: <https://www.youtube.com/watch?v=dRYHSf5A4lw>

Hint 2: <https://dl.acm.org/doi/abs/10.1145/3405669.3405819>

Hint 3: https://xilinx.github.io/Vitis_Libraries/security/2019.2/guide_L1/internals/aes.html

a) arithmetic in finite $\mathbb{GF}(2^8)$:

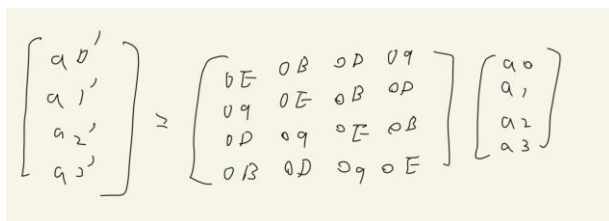
加法:在 $\mathbb{GF}(2^8)$ 中，加法為二進制的 XOR 運算。

乘法:將特定的元素相乘後相加，加法為上面提到的，然後在對特定的不可約多項式來取 mod。

Differ from Standard matrix multiplication:

加法: Standard matrix 用的是常規加法而 $\mathbb{GF}(2^8)$ 為 XOR

乘法: Standard matrix 為常規整數乘法，而 $\mathbb{GF}(2^8)$ 為乘法，以及其加法為 XOR 再取 mod。


$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \times \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

大概是這樣的概念，只是其中運算過程是 $\mathbb{GF}(2^8)$

b) 其中乘法中*2 可以視為左移一

乘 9:

$$\begin{aligned}
 X \cdot 9 &= X(2 \cdot 2 \cdot 2) \oplus X \\
 &= (X \cdot 2^3) \oplus X \\
 &\text{X 左移三位 再 是 X XOR}
 \end{aligned}$$

乘 11:

$$\begin{aligned}
 X \cdot 11 &= (X \cdot 8) \oplus (X \cdot 2) \oplus X \\
 &= (X \cdot 2^3) \oplus (X \cdot 2) \oplus X \\
 &\text{X 左移三位 再 是 X 左移一位和 X XOR}
 \end{aligned}$$

乘 13:

$$\begin{aligned}
 X \cdot 13 &= ((X \cdot 8) \oplus (X \cdot 4) \oplus X) \\
 &= ((X \cdot 2^3) \oplus (X \cdot 2^2) \oplus X) \\
 &\text{X 左移三位 再 是 X 左移二位和 X XOR}
 \end{aligned}$$

乘 14:

$$\begin{aligned}
 X \cdot 14 &= (X \cdot 8) \oplus (X \cdot 4) \oplus (X \cdot 2) \\
 &= (X \cdot 2^3) \oplus (X \cdot 2^2) \oplus (X \cdot 2) \\
 &\text{X 左移三位 再 是 X 左移二位和 X 左移一位 XOR}
 \end{aligned}$$

c) 用 LUTs 的優缺點

優點:

計算速度快:預先計算, 存結果並查表, 加快處理速度

簡化計算:查表就不用複雜的運算

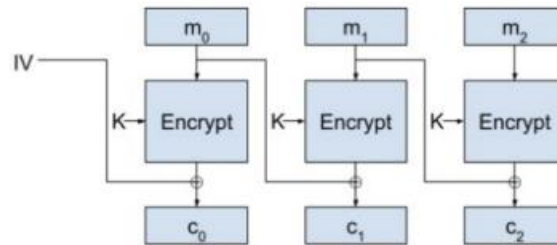
缺點:

空間消耗:因為查表的 **table** 會需要內存空間, 故可能是個問題

靈活性降低:固化了數據, 其算法可能不能適應新的變化

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 .

攻擊者知道 IV, C_0, C_1, C_2 ，也知道 $m_1 = m_2 = x$ ，根據加密步驟可以得到以下方程式：

$$C_0 = E_K(m_0 \oplus IV)$$

$$C_1 = E_K(m_1 \oplus C_0)$$

$$C_2 = E_K(m_2 \oplus C_1)$$

$m_1 = m_2 = x$ 可以改寫成

$$C_1 = E_K(x \oplus C_0)$$

$$C_2 = E_K(x \oplus C_1)$$

1 求 $m_1 \oplus C_0$

$$x \oplus C_0 = D_K(C_1)$$

2 求 $m_2 \oplus C_1$

$$x \oplus C_1 = D_K(C_2)$$

3 求 C_0

因為 x, C_1 已知

$$C_0 = x \oplus D_K(C_1)$$

4 計算 m_0

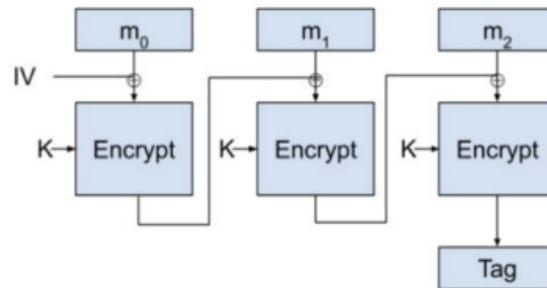
$$m_0 \oplus IV = D_K(C_0)$$

$$m_0 = D_K(C_0) \oplus IV$$

用已知的 IV 和計算出的 $D_K(C_0)$ 可確定 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, \dots in the diagram are distinct from the complete messages M_1, M_2, \dots

我們知道 M_1 和 M_2 和他的輸出 (IV_1, T_1) 和 (IV_2, T_2) 。

我們可以設計一個 M_3 近似於 $M_1 || M_2$ 但是有一個 block 被改動

希望可以調正 $M_1 M_2$ 的某些部份來控制其最終的 MAC 輸出，例如如果 m_2 (M_2 的最後一個塊) m_1 (M_1 的第一個塊)，攻擊者可以預測 T_1 和 M_1 的組合。

如果攻擊者精確地知道人和中間加密塊的輸出，他們可以調整其消息來生成有效的 MAC，這樣整個消息的 MAC 都可以被偽造，這樣可以在不知道 key 的情況下創造有效的 MAC 標籤訊息，破解題目的加密方式。