GF(2^8) 곱셈

https://youtu.be/6ZKK0jSyYiE

HSU 한성대학교 HANSUNG UNIVERSITY

CryptoCraft LAB
https://crypto.modoo.at

갈루아 체(Galois field)

- 체 : 덧셈, 뺄셈, 곱셈, 나눗셈의 사칙연산을 집합 안에서 소화할 수 있는 집합

- 유한체 : 유한개의 원소를 갖고 있는 체 GF(p^n), $\mathbb{F}_{p^n}$

- 유한체 = 갈루아체 = , GF(p^n)는 p^n 개의 원소를 갖는 유한체이다.

p는 소수이고, n 은 양정수

GF(7) {0,1,2,3,4,5,6}

덧셈 곱셈

- n-bit 워드들을 표현하는 다항식들은 두 개의 체 GF(2) 와 GF(2^n)를 사용

n-bit word: 1 0 0 1 1 0 0 1

Polynomial: $1x^7 + 0x^6 + 0x^5 + 1x^4 + 1x^3 + 0x^2 + 0x^1 + 1x^0$

First simplification: $1x^7 + 1x^4 + 1x^3 + 1x^0$

Second simplification: $x^7 + x^4 + x^3 + 1$

- GF(2^n)의 다항식들의 집합들에 대해서,
  차수 n의 다항식의 어떤 집합은 군은 모듈러로서 정의된다
- 모듈러는 기약 다항식 사용

$$\mathbb{F}_{p^n} \cong \mathbb{F}_p[t]/(f(t))$$

덧셈

- 다항식의 덧셈과 뺄셈 연산은 같은 연산 = exclusive-or (XOR)연산을 의미

GF(2^8)에서 $(x^5 + x^2 + x) \oplus (x^3 + x^2 + 1)$

$$
\begin{array}{ll}
0x^7 + 0x^6 + 1x^5 + 0x^4 + 0x^3 + 1x^2 + 1x^1 + 0x^0 & \oplus \\
0x^7 + 0x^6 + 0x^5 + 0x^4 + 1x^3 + 1x^2 + 0x^1 + 1x^0 & \\
\text{---} & \\
0x^7 + 0x^6 + 1x^5 + 0x^4 + 1x^3 + 0x^2 + 1x^1 + 1x^0 & \rightarrow \quad x^5 + x^3 + x + 1
\end{array}
$$

곱셈

- x^i 에 x^j 를 곱하면 결과로 x^{i+j}을 얻는다.
- 곱셈은 n – 1보다 큰 차수를 가지는 항을 생성할 수도 있다.
  따라서 모듈러 다항식을 사용하여 곱셈한 결과를 줄일 필요가 있다.
- $(x^8 + x^4 + x^3 + x + 1)$을 갖는 GF($2^8$) 에서 $(x^5 + x^2 + x) \otimes (x^7 + x^4 + x^3 + x^2 + x)$

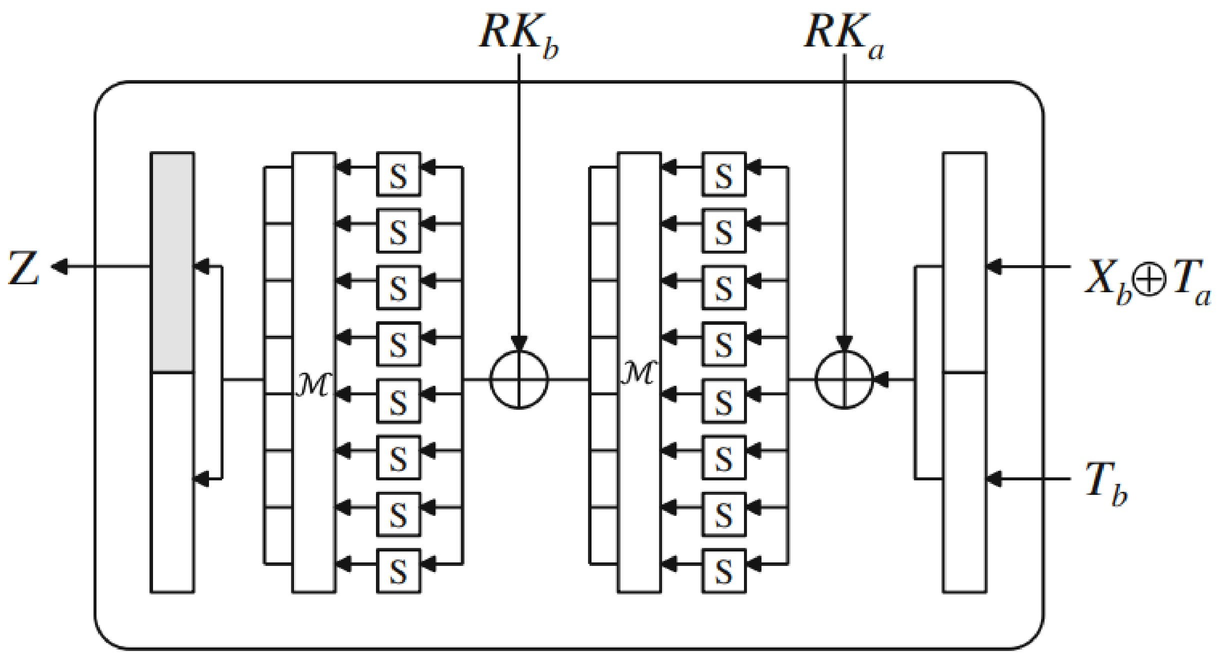
$$P_1 \otimes P_2 = x^5(x^7 + x^4 + x^3 + x^2 + x) + x^2(x^7 + x^4 + x^3 + x^2 + x) + x(x^7 + x^4 + x^3 + x^2 + x)$$

$$P_1 \otimes P_2 = x^{12} + x^9 + x^8 + x^7 + x^6 + x^9 + x^6 + x^5 + x^4 + x^3 + x^8 + x^5 + x^4 + x^3 + x^2$$

$$P_1 \otimes P_2 = (x^{12} + x^7 + x^2) \bmod (x^8 + x^4 + x^3 + x + 1) = x^5 + x^3 + x^2 + x + 1$$

$$\begin{array}{r|l}
 & x^4 + 1 \\
\hline
x^8 + x^4 + x^3 + x + 1 & x^{12} + x^7 + x^2 \\
 & \underline{x^{12} + x^8 + x^7 + x^5 + x^4} \\
 & x^8 + x^5 + x^4 + x^2 \\
 & \underline{x^8 + x^4 + x^3 + x + 1} \\
\text{Remainder} & \boxed{x^5 + x^3 + x^2 + x + 1}
\end{array}$$

FEA MDS matrix



$$\mathcal{M} = \begin{pmatrix} 28 & 1a & 7b & 78 & c3 & d0 & 42 & 40 \\ 1a & 7b & 78 & c3 & d0 & 42 & 40 & 28 \\ 7b & 78 & c3 & d0 & 42 & 40 & 28 & 1a \\ 78 & c3 & d0 & 42 & 40 & 28 & 1a & 7b \\ c3 & d0 & 42 & 40 & 28 & 1a & 7b & 78 \\ d0 & 42 & 40 & 28 & 1a & 7b & 78 & c3 \\ 42 & 40 & 28 & 1a & 7b & 78 & c3 & d0 \\ 40 & 28 & 1a & 7b & 78 & c3 & d0 & 42 \end{pmatrix}$$

8차 기약 다항식 $p(t)=t^8 + t^6 + t^5 + t^4 + 1$을 이용하여 정의된 $GF(2^8)$ 상의 곱셈

IoT 환경을 위한 곱셈

곱셈 연산 수행

```c
unsigned char GF_mul(unsigned char a, unsigned char b)
{
    unsigned char result = 0, t;
    while (a != 0)
    {
        if ((a & 1) != 0)
            result ^= b;
        t = (b & 0x80);
        b <<= 1;
        if (t != 0)
            b ^= 0x71;
        a >>= 1;
    }
    return result;
}
```

```c
unsigned char GF_mul(unsigned char a, unsigned char b)
{
    unsigned char p = 0, mask;
    for (int i =0 ;i<8;i++)
    {
        p^=-(b&1)&a;
        mask=-((a>>7)&1);
        a=(a<<1)^(0x71&mask);
        b>>=1;
    }
    return p;
}
```

룩업 테이블

- 모든 수의 곱셈 $2^8 * 2^8 = 2^{16}$ 6kbyte 사용
- 행렬 M 원소 $2^8 * 8 = 2^{12}$ 2kbyte 사용
- 아두이노 우노(uno)

ATmega328P AVR 8-bit 마이크로컨트롤러

2kbyte의 램 사용

$$\mathcal{M} = \begin{pmatrix} \texttt{28} & \texttt{1a} & \texttt{7b} & \texttt{78} & \texttt{c3} & \texttt{d0} & \texttt{42} & \texttt{40} \\ \texttt{1a} & \texttt{7b} & \texttt{78} & \texttt{c3} & \texttt{d0} & \texttt{42} & \texttt{40} & \texttt{28} \\ \texttt{7b} & \texttt{78} & \texttt{c3} & \texttt{d0} & \texttt{42} & \texttt{40} & \texttt{28} & \texttt{1a} \\ \texttt{78} & \texttt{c3} & \texttt{d0} & \texttt{42} & \texttt{40} & \texttt{28} & \texttt{1a} & \texttt{7b} \\ \texttt{c3} & \texttt{d0} & \texttt{42} & \texttt{40} & \texttt{28} & \texttt{1a} & \texttt{7b} & \texttt{78} \\ \texttt{d0} & \texttt{42} & \texttt{40} & \texttt{28} & \texttt{1a} & \texttt{7b} & \texttt{78} & \texttt{c3} \\ \texttt{42} & \texttt{40} & \texttt{28} & \texttt{1a} & \texttt{7b} & \texttt{78} & \texttt{c3} & \texttt{d0} \\ \texttt{40} & \texttt{28} & \texttt{1a} & \texttt{7b} & \texttt{78} & \texttt{c3} & \texttt{d0} & \texttt{42} \end{pmatrix}$$

곱셈 규칙

```
#define xtime(x)   ((x<<1) ^ (((x>>7) & 1) * 0x1B))

uint8_t mul2(uint8_t a) {
    return xtime(a);
}
uint8_t mul3(uint8_t a) {  /* 3 = 2 + 1 */
    return xtime(a) ^ a;
}
```

```
uint8_t mul9(uint8_t a) {  /* 9 = 8 + 1 */
    return xtime(xtime(xtime(a))) ^ a;
}
uint8_t mul11(uint8_t a) {  /* 11 = 8 + 2 + 1 */
    uint8_t a2 = xtime(a), a4 = xtime(a2), a8 = xtime(a4);
    return a8 ^ a2 ^ a;
}
uint8_t mul13(uint8_t a) {  /* 13 = 8 + 4 + 1 */
    uint8_t a2 = xtime(a), a4 = xtime(a2), a8 = xtime(a4);
    return a8 ^ a4 ^ a;
}
uint8_t mul14(uint8_t a) {  /* 14 = 8 + 4 + 2 */
    uint8_t a2 = xtime(a), a4 = xtime(a2), a8 = xtime(a4);
    return a8 ^ a4 ^ a2;
}
```

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 14 & 11 & 13 & 9 \\ 9 & 14 & 11 & 13 \\ 13 & 9 & 14 & 11 \\ 11 & 13 & 9 & 14 \end{bmatrix} \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix}$$

곱셈 규칙

- mul2(x), mul26(x), mul40(x), mul64(x)
- 2^8 * 4 1kbyte

		xor	2	26	40	64
0x28 = 40	mul40(x)				1	
0x1A = 26	mul26(x)			1		
0x7B = 123	mul2(mul40(x))^mul40(x)^mul2(x)^x	3	1		2	1
0x78 = 120	mul2(mul40(x))^mul40(x)		1		2	
0xC3 = 195	mul2(mul64(x))^mul40(x)^mul26(x)^x	3	1	1	1	1
0xD0 = 208	mul2(mul40(x))^mul2(mul64(x))	1	2		1	1
0x42 = 66	mul64(x)^mul2(x)	1	1			1
0x40 = 64	mul64(x)					1

log table

- log (U × V) = log (U) + log (V)))

- log_table 256byte
- antilog 255byte

```c
uint8_t log_table[256], antilog[255];
const uint8_t g =4;

void init_log_table(void)
{
    log_table[0] = 0;  /* dummy value */
    for (int i = 0, x = 1; i < 255; x = GF_mul(x, g), i++) {
        log_table[x] = i;
        antilog[i] = x;
    }
}

uint8_t gmul_table(uint8_t a, uint8_t b)
{
    if (a == 0 || b == 0) return 0;

    uint8_t x = log_table[a];
    uint8_t y = log_table[b];
    uint8_t log_mult = (x + y) % 255;

    return antilog[log_mult];
}
```

subfield representation

- $GF(2^8) = GF((2^4)^2)$
- $GF(2^{2m})$

$2^4 * 2^4$ 256kbyte table

$$
\begin{aligned}
\mathbf{a} \cdot \mathbf{b} &= (a_0 + a_1 x)(b_0 + b_1 x) \\
&= a_0 b_0 \ + \ (a_0 b_1 + a_1 b_0) x \ + \ a_1 b_1 x^2 \\
&= a_0 b_0 \ + \ (a_0 b_1 + a_1 b_0) x \ + \ a_1 b_1 (\alpha x + 1) \\
&= (a_0 b_0 + a_1 b_1) \ + \ (a_0 b_1 + a_1 b_0 + \alpha a_1 b_1) x
\end{aligned}
$$

$$(y_0, y_1) = (a_0, a_1) \cdot (b_0, b_1) \qquad \begin{aligned} y_0 &= a_0 b_0 + a_1 b_1 \\ y_1 &= a_0 b_1 + a_1 b_0 + \alpha a_1 b_1 \end{aligned}$$

Q & A