〈국문초록〉

유한체 위에서의 수열 생성과 영상 암호화

연구자 : 장유준(2학년, happycattony0329@gmail.com)
Binbing Xie(2학년, Huzhang9876@163.com)
Zhang Hui(2학년, xiebinbing897@163.com)

책임지도자 : 조성진(부경대학교, sjcho@pknu.ac.kr)
공동지도자 : 권민정(부경대학교, mjblack02@pknu.ac.kr)

요약문

우리는 인류 역사상 그 어느 때보다도 정보의 보안성이 철저해야하는 시대에 살고 있다. 이메일과 SNS는 우리들의 정체성을 대변하여 현금은 전자신호로 바뀌어 사람들 사이에서 거래가 가능한 이유는 중요한 정보들을 어느 개인의 정보를 도용하지 못하게 만들었기 때문이다. 그러나 정보과학의 기술이 발전하기에 필연적으로 암호 역시 발전해 나가야 한다.

현재 CA를 이용한 암호는 선형점화식을 바탕으로 충분히 긴 키수열로 문장 혹은 이미지를 암호화 한다. 그러나 이 방식은 단순히 반복되는 수열로 암호문을 만들면 그 수열의 주기의 2배가 되는 비트의 정보만 가지고도 암호가 풀려버린다는 문제점이 있다. 이러한 단점을 개선한 키수열이 바로 수축수열이다. 우리는 수축수열 생성함수로 생성한 비선형적 수열을 이용한 영상암호체계를 설계하고자 한다.

주제어 : 유한체, 1차원 CA, 수열 생성기, 암호키, 영상 암호화

Ⅰ. 서론

LFSR이 무엇인지, 기약다항식과 원시다항식이 어떠한 의미를 담고 있는지, 특성다항식 통해 생성한 수열의 일반항을 구하는 방법, 1차원 CA와 다항식이 어떠한 관계를 가지는지, 그리고 CA의 주기를 구하는 방법은 수열키를 이용한 암호학의 기초적인 배경지식이다. 흔히 수열키를 이용한 암호는 평문에 수열키를 XOR하여 암호문을 만드는 가장 기본적인 방식이다. 이러한 방식에 여러 가지 변화와 다양성을 부여한다해도, 근본적으로 키를 더하는 방식이 크게 바뀌지 않는다. 이미지의 암호화 역시 이미지를 행렬로 전환시킨 후 수열키를 차례대로 더하는 것이 보편적인 방식이다. 이때 사용하는 암호키는 일반적으로 선형적인데, 이 경우 주기가 길어지는 대신 선형복잡도가 떨어지기 때문에 암호로서의 보안성이 떨어진다. 우리는 이러한 취약한 부분을 개선하기 위하여 수축 수열을 사용하여 영상을 암호화할 것이다.

Ⅱ. 이론적 배경

1. LFSR (Linear Feedback Shift Register)

1.1 유한체의 정의와 성질

공집합이 아닌 집합 F위에서 정의된 두 연산 $+$와 $\bullet$에 대하여 다음이 성립할 때, $(F, +, \bullet)$을 체라 한다.

(a) $(F, +)$는 항등원 0을 갖는 가환군이 된다.
(b) $(F-\{0, \bullet\})$는 항등원 1을 갖는 가환군이 된다.
(c) 임의의 원소 $a,b,c \in F$에 대하여 $a \bullet (b+c) = a \bullet b + a \bullet c$ 이다,

F의 원소의 개수가 유한하면 F를 유한체라고 한다.

본 연구에서는 $GF(2)$의 연산 방식 $(\oplus, \bullet)$을 적용한 유한체, 그 중에서도 원소의 개수가 2^n개인 유한체만을 다룰 것이다.

$$GF(2) = \{0,1\}$$
$$GF(2^2) = \{0,1,\alpha,\alpha^2\} \ (\alpha^2 = \alpha + 1, \alpha^3 = 1)$$

$$GF(2^2)=\{0,1,\alpha,\alpha^2,\alpha+1,\alpha^2+1,\alpha^2+\alpha,\alpha^2+\alpha+1\}$$
$$=\{0,1,\alpha,\alpha^2,\alpha^3,\alpha^4,\alpha^5,\alpha^6\}\ (\alpha^3=\alpha+1,\alpha^7=1)$$

$GF(2)$는 원소가 0과 1로 이루어진 체이다. 수열키를 구성하는 숫자들이 $GF(2)$의 원소이며 더하기가 XOR 연산을 따르게 되어있다.

$GF(2^2)$는 $\alpha^3-1=(\alpha-1)(\alpha^2+\alpha+1)=0$의 해가 되는 근 $1,\alpha,\alpha^2$ 그리고 0을 원소로 갖는다.

$GF(2^3)$도 마찬가지의 방법으로 $\alpha^7-1=(\alpha-1)(\alpha^3+\alpha+1)(\alpha^3+\alpha^2+1)=0$의 해가 되는 근 $1,\alpha,\alpha^2,\cdots,\alpha^6$ 그리고 0을 원소로 갖는다.

이때 $\alpha^3=\alpha+1$을 만족하는 해와 $\alpha^3=\alpha^2+1$을 만족하는 해 α에 대응되는 벡터는 각각 다르다.

$$GF(2^3)=\{0,1,\alpha,\alpha^2,\alpha+1,\alpha^2+1,\alpha^2+\alpha,\alpha^2+\alpha+1\}$$
$$=\{0,1,\alpha,\alpha^2,\alpha^3,\alpha^4,\alpha^5,\alpha^6\}\ (\alpha^3=\alpha+1,\alpha^7=1)$$
$$\Rightarrow\{000,001,010,100,011,110,111,101\}(\text{벡터화})$$

$$GF(2^3)=\{0,1,\beta,\beta^2,\beta^3,\beta^4,\beta^5,\beta^6\}\ (\beta^3=\beta^2+1,\beta^7=1)$$
$$\Rightarrow\{000,001,010,100,101,111,011,110\}(\text{벡터화})$$

정의 1.

$f(0)\neq0$인 기약다항식 $f(x)$가 있다고 하자. 기약다항식의 최고차항이 n일 때 $\deg(f(x))=n$이라 표현한다. $Min\{a\in N: f(x)\mid x^a-1\}=d$ 일 때 d을 $f(x)$의 주기라고 하고, $\mathrm{ord}(f(x))=d$라 한다.

예제 2.

$$f(x)=x^2+x+1,\ f(x)|x^3-1,\ \mathrm{ord}(f(x))=3 \tag{1}$$
$$g(x)=x^3+x+1,\ g(x)|x^7-1,\ \mathrm{ord}(g(x))=7 \tag{2}$$
$$h(x)=x^4+x^3+x^2+x+1,\ h(x)|x^5-1,\ \mathrm{ord}(h(x))=5 \tag{3}$$

n이 자연수일 때 $x^{2^n-1}-1$은 n의 약수를 차수로 하는 모든 기약다항식들을 곱한 결과물이다. (이때 x^1를 포함하지 않는다)

예제 3.

$$x^{2^4-1}-1=(x-1)(x^2+x+1)(x^4+x+1)(x^4+x^3+1)(x^4+x^3+x^2+x+1)$$
$$x^{2^5-1}-1=(x-1)(x^5+x^2+1)(x^5+x^3+1)(x^5+x^3+x^2+x+1)$$
$$\times(x^5+x^4+x^2+x+1)(x^5+x^4+x^3+x+1)(x^5+x^4+x^3+x^2+1)$$

정의 4.

원시다항식은 기약다항식들 중에서도 더 특수한 다항식들을 말하는데, n이 자연수이고 $\deg(f(x))=n, \mathrm{or} d(f(x))=2^n-1$이면, $f(x)$를 원시다항식이라고 한다.

식 (1), (2)의 $f(x), g(x)$는 원시다항식이고, 식 (3)의 $h(x)$는 원시다항식이 아니다. 원시다항식은 같은 최고차항을 가지는 다항식들보다도 주기가 가장 길다. 그러하기 때문에 원시다항식을 이용하여 긴 수열을 생성하기가 쉽다.

정의 5.

어떤 다항식 $f(x)$가 n차 다항식일때, 상반다항식은 식 (4)와 같이 정의된다.

$$f^*(x)=x^n f(\frac{1}{x}) \tag{4}$$

$f(x)=a_n x^n+a_{n-1}x^{n-1}+\cdots+a_1x^1+a_0$일 때, $f^*(x)=a_0x^n+a_1x^{n-1}+\cdots+a_{n-1}x^1+a_n$이다.

예제 6.

$$f(x)=x^3+x+1, f^*(x)=x^3+x^2+1$$
$$f(x)=x^8+x^5+x^2+1, f^*(x)=x^8+x^6+x^3+1$$

또한, n차 다항식 $f(x)$에 대하여 $(f(x))^m$ (m은 자연수)의 상반다항식을 구하면 식 (5)가 된다.

$$((f(x))^m)^*=x^{nm}(f(\frac{1}{x}))^m=(x^n f(\frac{1}{x}))^m=(f^*(x))^m \tag{5}$$

이 성질은 나중에 수열의 일반항을 구할 때 쓰이게 된다.

1.2 LFSR (Linear Feedback Shift Register)

LFSR은 0과 1로 이루어진 수열을 만들어내는 기계장치이다.

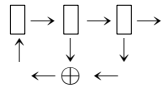

그림 [1]

1	0	0→0
0	1	0→0
1	0	1→1
1	1	0→0
1	1	1→1
0	1	1→1
0	0	1→1
1	0	0→0

그림 [2]

그림 [1]의 상자 안의 숫자들은 기계 내부의 화살표를 따라서 동시에 움직이면서 연산이 된다. 그림 [1]에서는 가장 왼쪽의 상자의 숫자가 중간 상자로 들어갔고, 중간 상자는 오른쪽 상자로, 그리고 오른쪽 상자에 있던 숫자가 출력이 된다. 가장 왼쪽의 상자에는 중간 상자와 가장 오른쪽 상자의 숫자가 XOR 연산으로 처리되어 들어가진다. 이런 과정을 반복하면 이 기계에서 일정한 주기를 갖는 수열이 출력이 된다.

그림 [2]에서 초기값 처음에 $1,0,0$으로 시작하여 7번의 연산 후에 다시 $1,0,0$으로 되돌아왔다. 이때 출력되는 수열은 그림 [2]에서 볼 수 있듯이 주기가 7인 수열 $\underline{0010111}001\cdots$이다.

1.3 기약다항식과 LFSR 사이의 호환관계

특정한 기약다항식에 대응하는 특정한 LFSR을 만들 수 있다.

x^3+x+1에 대응하는 LFSR을 만든다고 해보자. 우선 세 개의 상자를 그리고 각 상자 위에 순서대로 $x^2, x, 1$이라고 적는다. 이 상자 위의 항들은 오른쪽으로 한 칸씩 자리를 옮기고, 왼쪽 상자에는 x^3이 새로 들어온다. 이때 오른쪽으로 한 칸씩 자리를 옮기는 행위를 한 턴이 지났다고 말하자. 이제 우리는 $x^3=x+1$을 만드는 것이 목표이다. 한 턴이 지났을 때, x와 1이 더해져서 x^3 이 되도록 $\oplus$하면, 그림 [3]의 LFSR과 같이 된다.

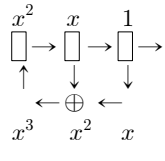

그림 [3]

항등식은 각 턴이 지날 때마다 가장 왼쪽에 있는 상자 내의 다항식을 나타낸다. 매 턴마다 오른쪽의 두 상자 안의 항들이 더해져서 가장 왼쪽 상자 항을 나타낸다. 처음 $x^3 = x+1$을 바탕으로 $x^7 = 1$이라는 결론으로부터 이 LFSR은 주기가 7인 수열을 만든다는 사실을 알 수 있다. 마찬가지의 방식으로, n차 원시다항식 $f_n(x)$에 대응하는 LFSR은 예제 3에 의하여 $f_n(x) | x^{2^n-1} - 1$이기 때문에 생성된 수열의 최대주기는 $2^n - 1$이다.

초깃값은 $000, 001, 010, 011, 100, 101, 110, 111$을 넣을 수 있다. 각 상자들 안에 어떤 배열의 초깃값들을 넣어도 000이 초깃값인 경우를 제외하고는 시작점만 다르고 숫자의 배열과 주기가 똑같다. LFSR 상자가 n개 있을 때 넣을 수 있는 수열의 종류는 2^n이다.

1.4 LFSR의 행렬화

LFSR은 선형대수적인 기계장치이므로 LFSR을 행렬과 대응시킬 수 있다.

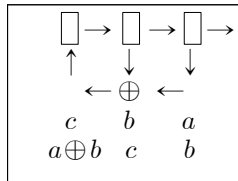

그림 [4]

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} b \\ c \\ a \oplus b \end{pmatrix}$$

그림 [5]

그림 [4]의 LFSR은 1턴에 각 상자 안에 있는 수 a, b, c를 $b, c, a \oplus b$로 바꾼다. 그림 [5]의 행렬이 그림 [4]의 LFSR을 행렬로 표현한 것이다. 1×3행렬을 곱하면 그림 [4]의 LFSR과 같다. 단위행렬은 아무런 영향을 안 주기 때문에 이를 이용하여 LFSR을 만들면 같은 수열들만 나오게 된다.

정의 7.

주어진 n차 다항식 $f(x) = x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_0$에 대하여 그림 [6]의 $n \times n$ 정방행렬을 $f(x)$의 동반행렬이라고 한다.

$$\begin{pmatrix} 0 & 0 & 0 & \cdots & a_0 \\ 1 & 0 & 0 & \cdots & a_1 \\ 0 & 1 & 0 & \cdots & a_2 \\ \vdots & \vdots & \vdots & \vdots & \\ 0 & 0 & 0 & \cdots & a_{n-1} \end{pmatrix}$$

그림 [6] $f(x)$의 동반행렬

어떤 행렬 A에 대하여 $|A-xI|$를 구하면 이것을 특성다항식 (characteristic polynomial)이라고 한다. 즉, $f(x)$의 동반행렬이 행렬 A일 때 행렬 A의 특성다항식이 $f(x)$이다. 케일리-헤밀턴 정리에 따라서 행렬 A는 특성다항식의 해가 되고 이 특성다항식이 n차 원시다항식이면 주기가 2^n이 되기 때문에 $A^{2^n}=I$가 된다.

2. Celluar Automata

2.1 CA의 정의

1차원 CA는 각 셀(cell)이 자기 자신과 양 옆 셀의 영향을 받아 값이 바뀐다.

자기 자신과 양 옆, 총 8가지의 초기조건에 따라서 나오는 결과가 각각 다르므로 총 $2^8=256$가지 경우가 있고, 이들에 번호를 매길 수 있다. 전 셀의 양 옆 수를 합한 것이 다음 셀의 수가 되는 규칙은 $rule90$이다. 그리고 전셀의 양 옆와 자기 자신을 합한 것이 다음 셀의 수가 되는 규칙은 $rule150$이다.

표 1

*Rule*150	*Rule*90	이전 셀과 양옆 이웃
0	0	000
1	1	001
1	0	010
0	1	011
1	1	100
0	0	101
0	1	110
1	0	111

↓ 0|011000101110|0 ↓ 0|111101001011|0 ↓ 0|100100110011|0	↓ 0|011000101110|0 ↓ 0|100101100101|0 ↓ 0|111100011101|0
그림 [7]	그림 [8]

그림 [7]의 수열들은 $rule90$을 적용하여, 그리고 그림 [8]의 수열들은 $rule150$을 적용하여 만든 시간에 따른 수열의 흐름이다. 선 밖의 0들은 null boundary를 나타내며 수열의 왼쪽 끝 셀의 왼쪽 이웃, 그리고 오른쪽 끝 셀의 오른쪽 이웃을 0으로 설정한다는 의미를 가진다.

2.2 CA의 행렬

본 연구에서는 1차원 CA를 다루겠다. 앞서 언급했듯이 1차원 CA는 각 셀(cell)마다 256가지 규칙들을 적용할 수 있다. 이 규칙은 모든 셀에 한가지로 통일하여 적용할 필요 없이 각 셀마다 다른 규칙을 적용할 수 있다. 예를 들어서 셀이 3개가 있는 CA는 각 칸마다 순서대로 $90, 150, 150$을 적용할 수 있고 $< 1,0,0 >$이라 표기한다.

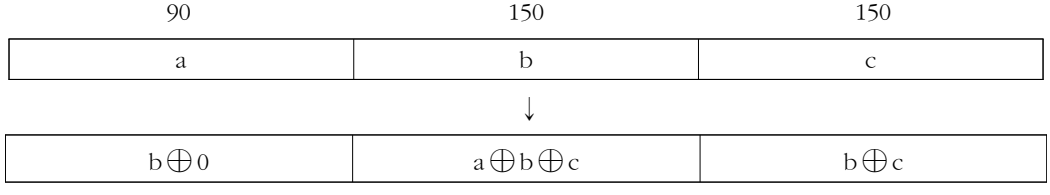

그림 [9] Null boundary

그림 [7]의 CA를 나타내는 행렬은 $\begin{pmatrix}0\,1\,0\\1\,1\,1\\0\,1\,1\end{pmatrix}$이다.

$$\begin{pmatrix}0\,1\,0\\1\,1\,1\\0\,1\,1\end{pmatrix}\begin{pmatrix}a\\b\\c\end{pmatrix}=\begin{pmatrix}b\oplus c\\a\oplus b\oplus c\\b\oplus c\end{pmatrix}$$

그림 [10]

CA 규칙을 행렬로 바꾸려면 대각선의 각 원소에 그 규칙에 대응하는 $0(rule\,90)$ 혹은 $1(rule 150)$을 넣어주면 된다.

$$< a_1,a_2,\cdots,a_n > \rightarrow \begin{pmatrix} a_1 & 1 & 0 & \cdots & 0\\ 1 & a_2 & 1 & \cdots & 0\\ 0 & 1 & \vdots & \cdots & 0\\ \vdots & \vdots & 1 & a_{n-1} & 1\\ 0 & 0 & 0 & 1 & a_n \end{pmatrix}$$

그림 [11]

2.3 Trace를 이용한 수열의 일반항 표현

1) 수열과 이들을 표현하는 벡터, 그리고 생성함수

LFSR로 만든 수열은 각 항을 순서대로 $S_n\,(n=0,1,2,\cdots)$이라 이름을 붙일 수 있다. S_n들 간

의 관계는 그 수열을 만드는 특성다항식에 의하여 결정된다.

$f(x)=x^3+x+1=0$으로 수열을 만들면 $S_{n+3}=S_{n+1}+S_n$이다. 이때 S_n을 결정하는 초기 값 S_0,S_1,S_2에 따라 다른 S_n들의 값들이 정해진다. 초기 값들은 $(000,001,010,011,100,101,110,111)$가 있고 이들이 생성하는 수열의 마디들의 집합을 벡터의 모임이라 부르고 $\Omega(f)$라 표기한다.

$$\Omega(x^3+x+1)=\begin{cases}000000,\ \ 0010111,\\ 0101110,0111001,\\ \cdots\\ 1110010\end{cases}$$

$S(x)$는 수열 생성함수이다. 어떤 수열 $10010111001011\cdots$을 수열 생성함수로 표현하면 식 (6)이다.

$$\begin{aligned}S(x)&=1+0x+0x^2+1x^3+0x^4+1x^5+\cdots\\&=1+x^3+x^5+x^6+x^7+x^{10}+x^{12}+\cdots\end{aligned}\tag{6}$$

여기서 주목해야 할 점은 $S(x)\bullet f^*(x)=x^n$ (n은 유한한 어떤 정수)이다.

$$S(x)\bullet(x^3+x^2+1)=x^2+1=x^3\tag{7}$$

$$S(x)=\frac{x^3}{x^3+x^2+1}\tag{8}$$

식 (9)의 분자는 초깃값에 따라 지수가 달라지기 때문에 일반성을 잃지 않고 1로 둘 수 있다.

2) Trace

기약다항식 $x^3+x+1=0$을 인수분해 하자. 우선 한 근을 α라고 하면 $(x-\alpha)$를 인수로 갖는다. 또한, $(\alpha^3+\alpha+1)^2=\alpha^6+\alpha^2+1=(\alpha^2)^3+(\alpha^2)^1+1=0$이므로 α^2도 근이다. 이런 과정을 반복하면 α^4도 분해된다. 따라서 $x^3+x+1=(x-\alpha)(x-\alpha^2)(x-\alpha^4)=0$

이것을 통하여 우리는 어떠한 n차 기약다항식의 한 근을 기본 원소가 θ라 하면
$(x-\theta^1)(x-\theta^2)\cdots(x-\theta^{2^{n-2}})(x-\theta^{2^{n-1}})$로 인수분해가 된다는 것을 알 수 있다.

정의 8. (Trace 함수)

n차 기약다항식의 한 근 θ라 할 때, 기약다항식은 $(x-\theta^1)(x-\theta^2)\cdots(x-\theta^{2^{n-2}})(x-\theta^{2^{n-1}})$으로 인수분해가 된다. 그리고 Trace 함수를 $Tr_1^m(\theta)$라고 하면, $Tr_1^m(\theta)=\theta^{2^0}+\theta^{2^1}+\cdots+\theta^{2^{m-1}}$이고 $Tr_1^m : GF(2^m)\rightarrow GF(2)$이다.

예제 9.

$x^3+x+1=0$의 한 근 θ에 대하여 $Tr(i), i=0,1,\cdots,6$의 함숫값은 식 (9)과 같다.

$$Tr(1)=1,\ Tr(\theta)=0,\ Tr(\theta^2)=0,\ Tr(\theta^3)=1,\ Tr(\theta^4)=0,\ Tr(\theta^5)=1,\ Tr(\theta^6)=1 \tag{9}$$

여기서의 중요한 사실은 S_t는 $Tr(\theta)$를 이용하여 식 13과 같이 표현할 수 있다.

$$S_t = Tr_1^m(\theta^{t+k})\ (k\text{는 초깃값 역할을 해준다}) \tag{10}$$

경우에 따라서 t에 k를 더해줘서 수열과의 순서를 맞춰줘야 하지만, $Tr(\theta)$이 생성하는 수열의 반복되는 마디가 LFSR을 통해 만들어진 수열의 반복되는 마디와 같다는 사실은 변하지 않는다. 더불어서 $Tr(\theta)$은 선형적인 함수이며 $Tr(\theta^i)=Tr(\theta^{2i})\ (i=0,1,2,\cdots)$이다.

3) $(f(x))^m$ ($f(x)$는 원시다항식, $m\in N$)으로 생성된 수열의 일반항

$F(x)=(f(x))^3=(x^3+x+1)^3$으로 만든 수열을 구해보자

우선 이 다항식으로 생성되는 수열을 식 (5)를 이용하여 식 (14)와 같이 표현할 수 있다.

$$S(x)=\frac{1}{F^*(x)}=\frac{1}{(f^*(x))^3}=\frac{1}{(x^3+x^2+1)^3} \tag{11}$$

$f(x)=x^3+x+1=(x-\theta)(x-\theta^2)(x-\theta^4)$이라고 할 때 상반다항식 $f^*(x)$는 식 (12)로 표현할 수 있다.

$$(x^3+x+1)^*=(x-\theta^{-1})(x-\theta^{-2})(x-\theta^{-4})=(\theta x-1)(\theta^2 x-1)(\theta^4 x-1) \tag{12}$$

식 (12)를 이용하여 $S(x)$를 쪼갤 수 있다.

$$\frac{1}{(x^3+x^2+1)^3}=\frac{\alpha_{0,1}}{\theta x-1}+\frac{\alpha_{0,2}}{(\theta x-1)^2}+\frac{\alpha_{0,3}}{(\theta x-1)^3}+\frac{\alpha_{1,1}}{\theta^2 x-1}+\frac{\alpha_{1,2}}{(\theta^2 x-1)^2}\frac{\alpha_{1,3}}{(\theta^2 x-1)^3}+\frac{\alpha_{2,1}}{\theta^4 x-1}+\frac{\alpha_{2,2}}{(\theta^4 x-1)^2}\frac{\alpha_{2,3}}{(\theta^4 x-1)^3}=\sum_{k=0}^{2}\sum_{j=1}^{3}\frac{\alpha_{k,j}}{(1-\theta^{2^k}x)^j}$$

$$\frac{1}{(f^*(x))^3}=\sum_{k=0}^{2}\sum_{j=1}^{3}\frac{\alpha_{k,j}}{(1-\theta^{2^k}x)^j}$$

($f(x)$의 차수가 n이고 $F(x)=(f(x))^m$의 $S(x)$를 구하면 $\sum_{k=0}^{n-1}\sum_{j=1}^{m}\frac{\alpha_{k,j}}{(1-\theta^{2^k}x)^j}$이 된다)

우리는 $\alpha_{k,j}$의 관계를 구해내야 한다.

$S(x)$는 $GF(2)$의 연산을 따르기 때문에 $S(x^2)=S(x)^2$이다.

$$\sum_{k=0}^{2}\sum_{j=1}^{3}\frac{\alpha_{k,j}}{(1-\theta^{2^k}x^2)^j}=\sum_{k=0}^{2}\sum_{j=1}^{3}\frac{(\alpha_{k,j})^2}{(1-\theta^{2^k}x)^{2j}}=\sum_{k=0}^{2}\sum_{j=1}^{3}\frac{(\alpha_{k,j})^2}{(1-\theta^{2^k}x^2)^j} \tag{13}$$

식 (13)을 풀어서 분모가 $(1-\theta^{2^k}x^2)^j$인 각 항들의 분모를 비교해보면 $\alpha_{k,j}=(\alpha_{0,j})^{2^k}$가 되고, $\alpha_{0,j}$는 α_j로 간략하게 표현할 수 있다.

$$\begin{aligned}\sum_{k=0}^{2}\sum_{j=1}^{3}\frac{\alpha_j^{2^k}}{(1-\theta^{2^k}x)^j}&=\sum_{k=0}^{2}\sum_{j=1}^{3}\alpha_j^{2^k}(1-\theta^{2^k}x)^{-j}\\&=\sum_{t\geq 0}x^t Tr(\sum_{j=1}^{3}\binom{j+t-1}{t}\alpha_j\theta^t)\\&=\sum_{k=0}^{2}\sum_{j=1}^{3}\sum_{t\geq 0}\alpha_j^{2^k}\binom{-j}{t}(-1)^t(\theta^{2^k}x)^t\\&=\sum_{t\geq 0}x^t\sum_{k=0}^{2}\sum_{j=1}^{3}\binom{j+t-1}{t}\alpha_j^{2^k}\theta^{2^k t}\end{aligned} \tag{14}$$

식 (14)의 x^t의 계수가 수열의 일반항, $Tr(\sum_{j=1}^{3}\binom{j+t-1}{t}\alpha_j\theta^t)$을 나타낸다. $\binom{j+t}{t}=\binom{j}{i}\binom{t}{i}$를 이용하면

$$\sum_{j=1}^{3}\binom{j+t-1}{t}\alpha_j=\sum_{j=1}^{2}\binom{j+t}{t}\alpha_{j+1}=\sum_{j=1}^{2}(\sum_{i\geq 0}\binom{j}{i}\binom{t}{i})\alpha_{j+1}=\sum_{i\geq 0}(\binom{t}{i}\sum_{j=1}^{2}(\binom{j}{i}\alpha_{j+1}))=\sum_{i\geq 0}(\binom{t}{i}\gamma_i)$$

이다. 따라서

$$Tr(\sum_{j=1}^{3}\binom{j+t-1}{t}\alpha_j\theta^t) = Tr(\theta^t\sum_{j=0}^{2}\binom{t}{j}\gamma_j) = \sum_{j=0}^{2}\binom{t}{j}Tr(\theta^t\gamma_j)$$

식 (10)에 의하면 원시다항식의 수열을 만드는 경우 그 일반항이 $S_t = Tr_1^m(\alpha^{(t+k)})$이다.

$F(x)$도 식 (10)과 상당히 유사한 형태를 보인다. 단지 기약다항식의 중복도만큼 $Tr(\theta)$의 개수가 있고 각 $Tr(\theta)$은 앞에 $\binom{t}{j}$가 곱해져 있는 것이다.

$\binom{t}{j}$은 j값에 따라 t에 대한 수열들은 같다.

예제 10.

$$j=0 : 1111111111\cdots \text{ 주기: } 1$$
$$j=1 : 0011001100\cdots \text{ 주기: } 2$$
$$j=2 : 0101010101\cdots \text{ 주기: } 4$$
$$j=3 : 0001000100\cdots \text{ 주기: } 4$$
$$j=4 : 0000111100\cdots \text{ 주기: } 8$$
$$j=5 : 0000010100\cdots \text{ 주기: } 8$$
$$j=6 : 0000001100\cdots \text{ 주기: } 8$$
$$j=7 : 0000000100\cdots \text{ 주기: } 8$$

j에 따라서 생성된 수열을 $s(x)$와 같은 방식으로 표현한 다항식을 $p(x)$라 하자. $p(x)=\sum_{t\geq 0}\binom{t}{r}x^t$을 계산해보면 $p(x)=\frac{x^r}{(1-x)^{r+1}}$이 되고 더 많은 증명들을 거치면 $p(x)$는 $2^{k-1}\leq r<2^k$일 때 주기가 2^k가 된다.

즉, 일반적으로 다항식 $F(x)$에 의해 생성된 $\sum_{j=0}^{m-1}\binom{t}{j}Tr(\theta^t\gamma_j)$의 주기는 $2^{k-1}\leq j$인 부분이 원시다항식의 주기 d와 맞물려서 $d\times 2^k$가 된다. 만일 $2^{k-1}\leq j$인 부분의 $\gamma_j=0$이면 $\binom{t}{j}Tr(\theta^t\gamma_j)=0\,(2^{k-1}\leq j)$이 되어서 주기가 $d\times 2^k$가 되지 못한다.

3. 수축 수열 (Shrinking generator)

3.1 $F(x)=(f(x))^m$ ($f(x)$: 기약다항식, $m=2,3,4,\cdots$)의 수열

특성다항식 $(x^3+x+1)^3$의 수열을 만들어보면 $1100011000001010011010101100$이 반복되어

서 나온다. 이 수열을 4개씩 끊어서 세로로 쓴 것이 표 1이다.

표 2. $(x^3+x+1)^3$으로 생성된 수열

	0	1	2	3	4	5	6
$S_t^{(0)}$	1	0	0	1	0	1	1
$S_t^{(1)}$	1	1	0	0	1	0	1
$S_t^{(2)}$	0	1	0	1	1	1	0
$S_t^{(3)}$	0	0	0	0	0	0	0

표 1처럼 $(f(x))^m$으로 생성된 수열을 2^f ($2^{f-1} < m < 2^f$)개의 블록으로 쪼갠 후 세로로 정열하면 주기가 d인 수열이 2^f개가 나온다. 이때 표 1처럼 재배치 된 수열의 i번째 줄의 가로를 수열로 고려하면 그 수열을 $s_t^{(i)}$로 부르기로 한다.

$$s_t^{(i)}(i=0,1,\cdots,2^f) = s_{2^f t+i}$$

$$\sum_{j=0}^{m-1}\binom{2^f t+i}{j} Tr(\theta^{2^f t+i}\gamma_j) = \sum_{j=0}^{m-1}\binom{i}{j} Tr(\theta^i(\theta^{2^f})^t\gamma_j) = Tr(\theta^i(\theta^{2^f})^t \sum_{j=0}^{m-1}\binom{i}{j}\gamma_j) = Tr(\theta^i(\theta^{2^f})^t\beta_i) \quad (\sum_{j=0}^{m-1}\binom{i}{j}\gamma_j = \beta_i) \quad (15)$$

(예제 10에서 나왔듯이 $\binom{t}{i}$는 주기가 2^f인 수열이기 때문에 $\binom{2^f t+i}{j}=\binom{i}{j}$가 된다.)

$Tr(\theta^t\alpha_i)$는 $\gcd(2^f,d)=1$이기 때문에 주기가 d가 된다.

식 15는 식 (10)와 유사하다. 단지 단 한 가지 차이점이 있는데, 근 θ가 θ^{2^f}로 대체되었다.

3.2 수축 수열의 정의

두 개의 원시다항식으로 만든 수열을 주기가 짧은 수열을 위에, 주기가 긴 수열을 아래에 놓는다. 그런 다음 주기가 짧은 수열의 항이 1일 때의 바로 밑의 긴 주기의 항을 취해서 새로운 수열을 만들면, 그 수열이 수축 수열(Shrinking sequence)이다.

$$\begin{array}{l} 01101101101101101101\cdots \quad (x^2+x+1) \\ 00101110010111001011\cdots \quad (x^3+x+1) \\ \ 01\ \ 11\ \ 00\ \ 01\ \ 10\ \ 10\ \ 11\cdots \quad ((x^3+x^2+1)^2) \end{array} \quad (16)$$

이때 아래의 수열을 A라 하고 그 차수를 $|A|$, 주기가 2^A-1이라고 하고 마찬가지로 위의 수열을 S, 그 차수를 $|S|$, 주기가 2^S-1이라고 하면 최종적으로 생성된 수축 수열은 주기가 최대 $(2^A-1)(2^{s-1})$이기 때문에 특성다항식이 $(f(x))^{2^n-1}(n\in N)$의 형태임을 알 수 있다.

3.3 선형복잡도 (Linear Complexity)

선형적으로 생성된 수열의 선형복잡도가 l이라고 하자. 그러면 그 수열로부터 $2l$만큼의 비트를 얻으면 Berlekamp-Massey algorithm을 사용하여 그 수열을 생성한 특성다항식과 전체 수열을 구할 수 있다. 원시다항식의 경우 n차 다항식으로 수열을 만들면 선형복잡도는 n이다. 즉, 비록 수열의 주기 자체는 1.3의 마지막 단락에서 설명했듯이 2^n이지만, 해커가 n개의 비트만 훔치면 전체수열을 알아낼 수 있는 것이다.

주석의 논문 [1]의 2장(Period and Linear Complexity)는 수축수열의 선형복잡도를 다루면서 두 가지 정의를 이야기한다.

정의 11.

수열 A와 S가 식 21의 방식대로 A를 위에, S를 밑에 배치하고 수축수열 Z를 생성한다고 하자. T_A, T_S를 각각 A와 S의 주기라고 하자. 만일

- A와 S가 최대주기를 가지고 ($i.e.$ 원시다항식으로 생성한 수열이라면)
- (T_s, T_A)

이라면, Z의 주기는 $T_A \cdot 2^{|S|-1} = (2^{|A|-1}) \cdot 2^{|S|-1}$이다.

정의 11.

정의 10의 조건 아래에서, 수축수열 Z의 선형복잡도를 LC라 하자. 그러면, $|A| \cdot 2^{|S|-2} < LC \le |A| \cdot 2^{|S|-1}$이다. ($|A|$는 수열 A가 n차 다항식에 의해 생성되면, 그 n을 의미한다)

위의 두 정의에 대한 증명은 논문 [1]의 2장에 적혀있다. 즉, 수축수열은 원시다항식으로 생성한 수열보다 차수에 비하여 주기가 짧지만, 선형복잡도는 상당히 높다.

Ⅲ. 연구 방법 및 절차

1. 이미지 암호화 (Image Encryption)

이미지 암호화의 알고리즘은 다음과 같다.
단계 1: 이미지를 행렬로 변환시킨다.
단계 2: 특정한 특성다항식 혹은 CA를 이용하여 수열을 만든다.
단계 3: 행렬의 원소들에 차례대로 키를 XOR 연산을 시켜준다.
단계 4: 행렬을 다시 이미지로 변환시킨다.

단계 1은 만일 이미지가 256×256이고 흑백이라면, 각 256×256개의 픽셀(pixel)은 값이 0부터 255까지의 수들로 표기된다. 0은 가장 어두운 검은색, 255는 가장 밝은 흰색이 된다. 이러한 원리를 이용하여 흑백사진의 이미지를 행렬로 표현할 수가 있다. 또한, 우리가 사용하는 암호화 방식에 적합하도록 행렬의 각 원소들은 이진법으로 수를 표기한다.

단계 2에서 생성한 수열이 키가 되어준다.

단계 3은 행렬의 원소들에 순서대로 왼쪽에서 오른쪽으로, 위에서 아래로 차례대로 키수열을 XOR 연산을 시켜준다. XOR 연산을 보통 가로로 더하지만, 암호체계를 쓰는 양쪽이 정한 규칙대로 차례대로 더해주면 되기에 세로로 더하거나 대각선으로 더해도 상관없다.

다시 복호화를 할 경우에는 암호화된 이미지를 행렬로 변환시킨 후, 키수열을 더해주고 이미지로 변환시킨다. 그 외에도 암호를 해킹하기 어렵게 만들기 위하여 다른 여러 연산들을 넣을 수 있지만, 기본적인 이 4가지 단계는 반드시 들어간다.

2. 파이썬(Python)을 이용한 이미지 암호화의 코딩

파이썬을 이용하여 키수열로 이미지 암호화를 하면 다음과 같은 단계들이 필요하다.
단계 1: 이미지를 변수로 불러오기
단계 2-1: 원소를 이진수로 바꾸기
단계 2-2: CA로 수열 생성
단계 3: 이진수로 표현된 수열 $\oplus$ CA 수열

단계 4: 이미지의 모든 원소들이 3)을 실행할 때까지 2)로 되돌아감

단계 5: 암호화된 변수를 이미지로 저장

위의 단계들을 코딩으로 표현하면 다음과 같다.

```
import numpy as np
import cv2
import operator as op
import random as rd
from matplotlib import pyplot as plt
```

그림 [12]

```
def ca(cav,Rul):
    k=len(cav)
    tmp=cav
    cav[0]=(tmp[0]*Rul[0]+tmp[1])%2
    cav[k-1]=(tmp[k-1]*Rul[k-1]+tmp[k-2])%2
    for i in range(1,k-1):
    cav[i]=(tmp[i-1]+tmp[i]*Rul[i]+tmp[i+1])%2

    return cav
```

그림 [13]

```
def crypt(img,cav,Rul):
    for i in range(img.shape[1]):
    for j in range(img.shape[0]):
    nVec=0
    for t in range(8):
    cav=ca(cav,Rul)
    nVec=nVec + (cav[0]*(2**(7-t)))
    img[i,j]=op.xor(img[i,j],nVec)

    return img
```

그림 [14]

```
if __name__ == '__main__':
    file = 'Lenna2.jpg'
    img = cv2.imread(file, 0)

    cav=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1]
    Rul=[1,1,1,1,1,0,1,1,0,1,1,0,0,0,1,1,0,0,0,0,1,1,0,1,1,0,1,1,1,1,1,1]
    img=crypt(img,cav,Rul)
```

그림 [15]

```
    cv2.imshow('image', img)
```

그림 [16]

```
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

그림 [17]

그림 12와 17는 프로그램의 환경조성을 위한 부분이다.

그림 13은 Rul(규칙)과 cav(초기값)을 입력하면 수열을 생성해준다. 현재 이 코드에서는 초기값[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1]과 규칙[1,1,1,1,1,0,1,1,0,1,1,0,0,0,1,1,0,0,0,0,1,1,0,1,1,0,1,1,1,1,1,1]을 입력했다.

그림 14는 이미지를 행렬로 변환시킨 후, 원소에 수열을 더해준다. 이 함수가 이미지를 암호화하는 함수이다.

그림 15는 실제로 이미지 암호화 함수를 주어진 이미지, 규칙, 그리고 초기값을 대입하여 실행했다.

그림 16은 암호화된 이미지를 불러온다.

위의 파이썬 코딩을 통하여 모든 선형적인 CA들로 이미지를 암호화를 할 수 있다.

3. 히스토그램, 산포도, 그리고 상관계수

암호화된 이미지가 통계학적으로 얼마나 무작위하게 암호화가 되었는지를 알아보기 위하여 히스토그램, 그리고 상관계수와 산포도 그래프를 확인하는 코딩을 만들었다.

3.1 히스토그램

히스토그램은 이미지의 픽셀에 표기된 수의 개수를 그래프로 표현한다. 가로축은 $0 \sim 255$까지의 픽셀의 색깔을 수로 나타내고 세로축은 픽셀의 개수를 의미한다. 이미지를 암호화하기 전과 후를 비교해보았을 때 암호화한 후에는 특정 색깔에 대한 픽셀의 개수가 무작위하게 되어 암호화가 잘 되었다는 것을 알 수 있다. 그림 20은 원문 사진의 히스토그램이고, 그림 21는 암호화된 사진의 히스토그램이다.

그림 [18] Lenna.jpg (256×256)

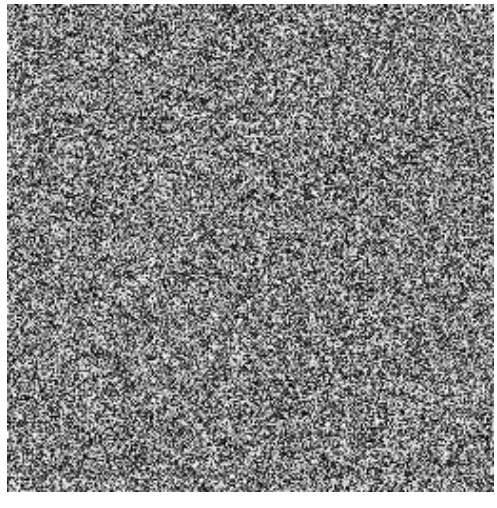

그림 [19] Lenna_encyrpted.jpg (256×256)

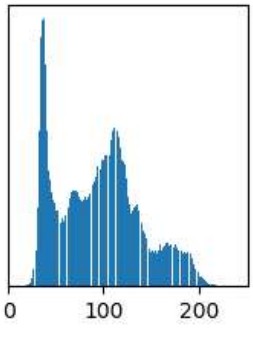

그림 [20]

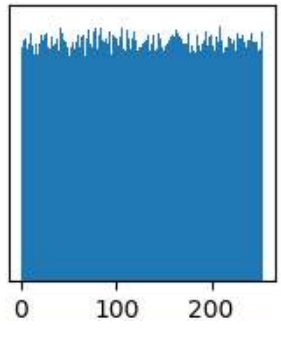

그림 [21]

3.2 산포도 그래프

산포도 그래프는 어떤 두 변수사이의 관계를 살펴보기 위하여 사용한다. 만일 산포도 그래프 위의 데이터들이 규칙성을 보이면 그 데이터들은 x와 y사이의 관계가 더 명확히 들어난다.

그러나 그래프 위의 데이터들이 무작위로 있으면 x와 y사이의 관계는 거의 없다. 이러한 성질을 이용하여 이미지를 암호화 했을 때, 암호화 된 이미지에서 랜덤으로 추출한 두 인접한 픽셀 사이의 값에 차이가 얼마나 나는지를 확인할 수 있다. 산포도 그래프는 가로축을 하나의 픽셀, 그래프의 세로축을 인접한 또 다른 픽셀의 수로 설정한다. 만일 x와 y 사이의 선형적인 관계를 나타내면, 이것은 곧 이미지의 여러 부분들의 색깔이 뭉쳐있다는 것을 의미하고, 암호화가 잘 안 된 것이다. 반면에 암호화가 잘 된 이미지는 모양이 TV의 노이즈와 같은 모습이 되고, x와 y 사이의 관계도 상당히 무작위하게 된다. 그림 20은 원문의 x축으로 이웃한 두 픽셀을 각각 가로축, 세로축으로 지정하여 그래프를 만들었고, 그림 22은 원문의 x축으로 이웃한 두 픽셀을 각각 가로축, 세로축으로 지정하여 만든 그래프이고, 그림 23은 암호문의 x축으로 이웃한 두 픽셀을 각각 가로축, 세로축으로 지정하여 만든 그래프이다.

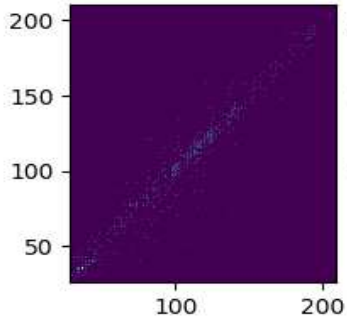

그림 22

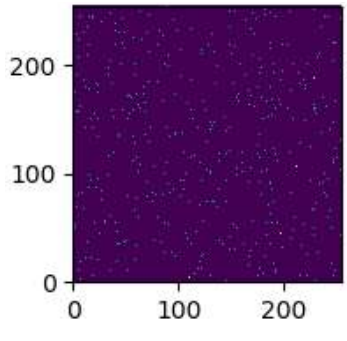

그림 23

3.3 상관계수

상관계수는 두 변수간의 선형관계를 수치화하여 나타낸 것이다. x와 y 사이의 상관관계가 1 혹은 -1에 가까워진다면, 두 변수를 놓고 만든 그래프는 위의 점들은 더욱 직선에 가까워진다. 반면에 x와 y사이의 상관관계가 0에 가까워진다면, 두 변수를 놓고 만든 그래프 위의 점들은 상당히 무작위하다.

이러한 사실이 왜 중요하면 이미지를 암호화 했을 때 두 인접한 픽셀이 서로 값의 차이가 많이 나야지 암호화가 잘 되었다는 지표가 되어준다. 만일 두 인접한 픽셀의 값이 비슷하다면 이미지의 어느 한 부분이 암호화가 잘 안 되어서 윤곽이 들어날 수가 있다.

IV. 연구 결과

그림 24는 원문이다. 픽셀은 256×256이다.

그림 25는 원시다항식 $x^{50}+x^{34}+x^{31}+x^{17}+1$를, 그리고 CA 규칙

그림 24 Lenna.jpg (256×256)

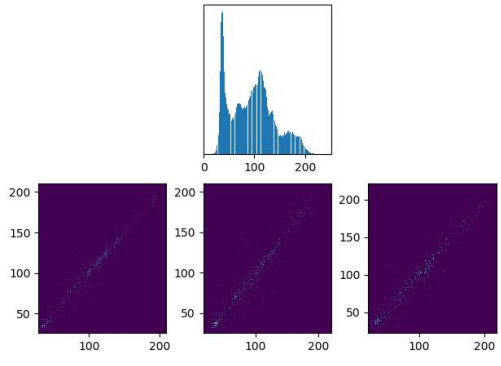

그림 25 Histogram & scatter plot of pic.24

[10011000010000000101101101011010101010101100011001]을 이용하여 키수열을 만든 후 원문에 XOR 연산을 하여 만든 암호문 이미지이다. 히스토그램과 산포도 그래프들 모두 이미지의 암호화가 잘 되었음을 나타내고 상관계수 역시 0에 상당히 근접하여 이미지의 픽셀들이 상당히 무작위하다는 것을 알 수 있다. 키수열의 주기는 $2^{50}-1=1.1258999\times10^{15}$으로 상당히 길다. 그러나 선형복잡도는 50이다.

그림 27은 30차 다항식과 19차 다항식으로 각각 만든 수열을 조합하여 만든 수축수열을 키수열로 설정하고 XOR하여 만든 암호화 이미지다.

키수열의 주기는 $(2^{30}-1)(2^{19-1})=2.8147498\times10^{14}$으로 매우 길다. 그리고 선형복잡도 LC는 $30\cdot2^{19-2}<LC\leq30\cdot2^{19-1}, 3932160<LC\leq7864320$으로 원시다항식과는 비교할 수 없을 정도로 높다.

그림 26와 그림 28 모두 히스토그램과 산포도 그래프를 보면 암호화가 잘 되었다. 그러나 그림 26의 선형복잡도는 그림 25보다 월등히 높았다. 이러한 사실을 통하여 만일 이미지를 암호화한다고 하면 원시다항식을 사용하는 것 보다 수축생성함수를 사용하는 것이 더욱 보안성이 뛰어나다는 것을 알 수 있다.

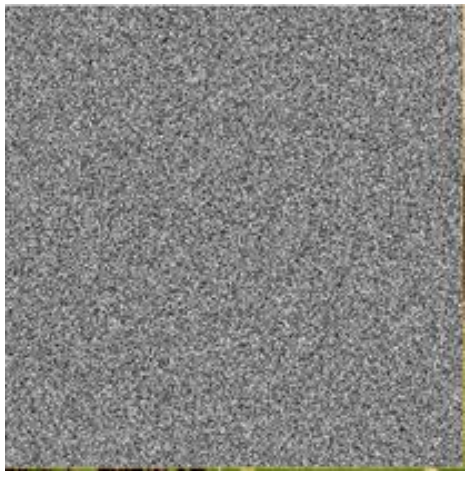

그림 26 Lenna_encrypted_primitive.jpg

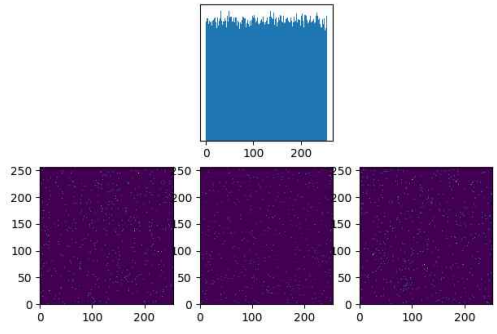

그림 27 Histogram & scatter plot of pic.25

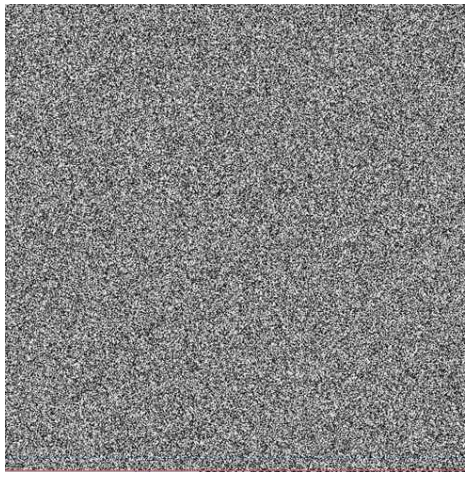

그림 28 Lenna_encrypted_shrinking.jpg

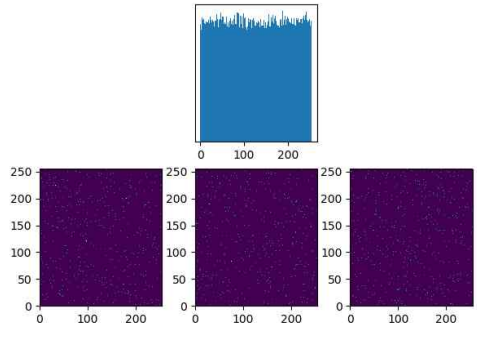

그림 29 Histogram & scatter plots of pic.27

V. 고찰

본 연구의 목적은 수축 수열을 이미지 암호화에 적용시켜 그 실효성을 확인해보는 것이다. 15차 이상의 원시다항식으로 이미지를 암호화하자 히스토그램도 상관계수도 무작위에 가깝게 잘 나왔다. 마찬가지로, 4차와 7차 다항식으로 수축 수열을 만들어 이미지에 암호키로 사용을 해보니 히스토그램과 상관계수가 무작위에 가깝게 잘 나왔다. 그러나 비록 두 암호의 주기

는 원시다항식을 쓰는 암호키가 더 길지만, 선형복잡도는 수축 수열이 더 크다. 둘다 암호화가 잘 되는 상황에서 더욱 안전한 암호키를 고르라고 하면 선형복잡도가 더 높은 암호키를 쓰는 것이 옳다.

Ⅵ. 결론

이미지 암호화를 기약다항식이나 원시다항식에 의해 생성된 선형수열 대신 CA를 사용한 수축수열을 암호키로 정하고 암호화하면 보안성이 더 강하다. 또한, 수축 수열은 두 가지 수열의 조합을 사용하여 만들었기 때문에 더 많은 양의 암호키를 생성할 수 있다. 이로서 수축수열을 키로 사용하는 암호화 방식은 텍스트뿐만이 아니라 이미지 암호화에도 적합하다는 것을 알 수 있다. 또한, 이미지에 암호키를 가로로만 더하는 것이 아니라 세로 혹은 대각선, 그 외 다양한 방법도 모색하면 좋을 것이다.

Ⅶ. 참고문헌

[1] D. Coppersmith, H. Krawczyk, Y. Mansour, The Shrinking Generator, Annual International Cryptology Conference. Springer Berlin Heidelberg, pp. 22-39, 1993.

[2] Ping Ping, Feng Xu, Zhi-Jian Wang,, Image encryption based on non-affine and balanced cellular automata, College of Computer and Information, Hohai University, Nanjing 210098, China, 2014.

[3] M.E. Pazo-Robles, A. Fúster-Sabater, Using Linear Difference Equations to Model Nonlinear Cryptographic Sequecnes, International Journel of Nonlinear Sciences & Numerical Simulation 11(3):165-172, 2010.

[4] Robert J. McEliece, Linear Recurring Sequences over Finite Field, For the Degree of Doctor of Philosophy, California Institue of Technology, Pasadona, California, 1967.

[5] 최언숙, 조성진, 김태홍, Image Encryption Based on One Dimensional Nonlinear Group Cellular Automata, Journal of Korea Multimedia Society Vol.18, No.12, December 2015.

[6] 조성진, 유한체 및 그 응용, 교우사, 2007.

ABSTRACT

Producing Series Solution of Differential Equation to use Polynomial Recurrence

Reseacher : Yujune Jang(sophomore, happycattony0329@gmail.com)
Hui Zhang(sophomore, Huzhang9876@163.com)
Binbing Xie(sophomore, xiebinbing897@163.com)

Supervisor : Sungjin Cho(Pukyoung National University, sjcho@pknu.ac.kr)
Co-Supervisor : Minjung Kwon(Pukyoung National University, mjblack02@pknu.ac.kr)

Abstract

We live in a period where our society is in need of secure encryption. In such era where information takes critical place in our society, the security for our precious information is what we need the most. Currently, password generated by CA is accomplished by making a sequence long enough based on linear algebra so that it takes years to hack the complete key. However, this method is easily deactivated when the hacker obtains a part of the key with bits twices as much as the period of the sequence. Our team is making a cryptograph system made with non-linear sequence.