韓國 컴퓨터情報學會 論文誌
第13卷 第6號, 2008. 11.

2008-13-6-3-3

휴대용 보안시스템에 적합한 MT-Serpent 암호알고리즘 설계에 관한 연구

이 선 근*, 정 우 열**

A Study on MT-Serpent Cryptographic Algorithm Design for the Portable Security System

Seon-Keun Lee *, Woo-Yeol Jeong **

요 약

이동식 시스템의 보안관련 문제점 등을 해결하기 위하여 본 논문은 네트워크 환경과 유무선 통신망에 적합하며 구현의 용이성, 비도 유지, 재수정 및 재사용 할 수 있으며 TCP/IP 프로토콜 아키텍쳐에 적합한 이동식 스마트카드용 MT-Serpent 암호알고리즘을 소프트웨어 기반이 아닌 하드웨어 기반 칩 레벨로 구현하였다. 구현된 MT-Serpent 암호시스템은 크기면에서 4,032이고 throughput은 406.2Mbps@2.44MHz를 가진다.

구현된 MT-Serpent 암호알고리즘은 스마트카드 등과 같은 이동식 시스템의 특징을 살릴 수 있도록 하기 위하여 TCP/IP 프로토콜의 보안 취약성을 보강하며 유무선 환경에서 여러 종류의 서비스가 가능하고 다수의 사용자에 대한 보안을 유지하는데 주요한 목적이 있다.

Abstract

We proposed that is suitable network environment and wire/wireless communication network, easy of implementation, security level preservation, scalable & reconfigurable to TCP/IP protocol architecture to implement suitable smart card MS-Serpent cryptographic algorithm for smart card by hardware base chip level that software base is not implement. Implemented MT-Serpent cryptosystem have 4,032 in gate counter and 406.2Mbps@2.44MHz in throughput. Implemented MS-Serpent cryptographic algorithm strengthens security vulnerability of TCP/IP protocol to do to rescue characteristic of smart card and though several kind of services are available and keep security about many user in wire/wireless environment, there is important purpose.

▸ Keyword : MT-Serpent Cryptographic Algorithm, Smart Card, Portable Security System

---

• 제1저자 : 이선근
• 접수일 : 2008. 8. 22, 심사일 : 2008. 9. 22, 심사완료일 : 2008. 11. 26.
* (주)SODY 수석연구원 ** 한려대학교 멀티미디어정보통신공학과 교수

Ⅰ. 서 론

네트워크 환경 및 정보통신 기기의 발달에 따라서 전자지불, 전자상거래 등과 같은 전자금융계는 매우 급격하게 발달되고 있다. 이러한 시점에서 사용자들의 편리함과 금융업무의 분산을 위하여 도입된 스마트카드의 기능은 매우 중요한 자리매김을 차지하고 있다. 그러나 스마트카드의 기능과 별개로 보안기능의 더딘 발전은 전자 금융 전체를 위협하는 무시할 수 없는 요소가 되었다. 이러한 단점을 보완하고자 스마트카드에 암호알고리즘을 적용하고 있으나 해킹 및 크래킹의 발달은 현재 사용되는 암호알고리즘의 해석을 가능하게 함으로서 스마트카드의 안전성을 보장할 수 없게 만들고 있다[1,2,4].

그러므로 본 논문에서는 NIST의 AES 후보 알고리즘들 중 최종 후보까지 올라갔던 Serpent 암호알고리즘을 스마트카드와 같은 휴대용 보안시스템에 적합하도록 설계함으로서 기존 휴대용 정보시스템의 보안상의 문제점들을 없애고자 한다[3][8].

설계된 MT-Serpent(Modified Table-Serpent) 암호알고리즘은 암/복호화의 동시수행, 용도에 따른 입력 데이터들의 가변길이 암호화, 라운드 처리 등과 같은 특징을 가진다. 이러한 특징은 스마트카드를 보다 넓은 응용범위에 사용할 수 있도록 하며 보안성을 유지하는데 매우 효율적으로 사용될 수 있다. 또한 이러한 특징은 대용량 멀티미디어 데이터들에 대한 실시간 처리가 가능하다는 것을 의미한다. 그러므로 스마트카드를 적재한 이동식 멀티미디어 시스템인 PDA, PDP, PMP 등의 응용분야에 매우 적합할 것으로 사료된다.

Ⅱ. 스마트카드

스마트카드(smart card)의 발전은 1974년 프랑스의 Roland Moreno가 IC 카드로 특허를 출원한 이후 많은 기업들에서 경쟁적으로 연구 및 개발이 시작되었으며, 이러한 기업들 중 특히 Bull S&T사와 모토롤라사가 칩 카드를 개발하였고 1981년부터 단일 칩으로 구성된 스마트카드가 개발되었다[3][5][6].

ISO/ITE JTC1 7816 IC 카드 표준은 스마트카드의 물리적 특성, 기본명령어, 보안구조, 데이터 객체 정의, 보안기능 명령어 및 접근통제를 위한 보안속성에 대하여 정의하고 있다.

스마트카드는 크게 데이터를 저장하는 메모리 영역과 데이터 처리를 담당하는 마이크로프로세서 영역으로 분류된다. 메모리 영역을 이용한 메모리 카드는 1990년대 초반에 정보보호 인식이 확산되어 데이터를 자체적으로 가공·처리할 수 있는 장치가 요구되면서 정보 통신망 환경의 응용분야에 사용되었다. 마이크로프로세서 영역을 이용한 마이크로프로세서 카드는 1990년대 후반에 비인가된 접근 및 훼손, 변조 공격으로부터 메모리에 대한 접근통제 기능, 패스워드 방식 및 스마트카드 운영체제 및 보안 논리 시스템을 통해서만 접근이 가능하도록 접근통제 메커니즘을 구현하였다. 그리고 데이터에 대한 암·복호화 처리기능도 제공함으로서 정보 통신망의 응용분야에 사용되고 있다[7].

스마트카드는 매우 큰 기억용량과 고도의 안전성을 위하여 IC 칩에 마이크로프로세서를 포함하고 있으며 메모리 카드보다 높은 보안성을 제공하며 정보처리 능력도 가지고 있다.

일반적으로 스마트카드는 IC 칩이 내장되어 정보처리 및 저장능력을 갖는 다목적이며 범용적인 TRM(Tamper Resistant Module) 카드를 의미한다. 외부의 응용프로그램을 통해서 카드 내부로 저장될 데이터들은 마이크로프로세서를 반드시 거쳐야 하며 응용프로그램과의 상호동작을 통하여 이루어진다. 대부분의 스마트카드에 삽입된 칩들은 폰노이만 구조를 채택하고 있고 메모리 분할과 분리된 메모리에 대한 접근을 통제하기 위한 보안 논리 회로가 스마트카드의 핵심이다. 또한 부가적으로 보안성 증대 및 네트워크 환경에 적합하도록 하기 위한 계산 능력 향상 및 상호인증 기능을 수행하기 위하여 요구되는 난수생성 등 암호 기능을 지원하는 coprocessor를 추가할 수 있다[3][7].

Ⅲ. 스마트카드에 적합한 MT-Serpent 암호알고리즘

스마트카드와 같이 이동성을 가지며 금융거래와 같은 매우 중요한 응용분야에 적용하기 위한 블록 암호알고리즘은 처리되는 데이터량, 처리시간, 시스템 복잡도로 인한 성능 저하 및 비도유지가 가장 중요한 파라미터가 된다. 그러므로 스마트카드에 적합한 블록 암호알고리즘은 비도를 위한 충분한 라운딩 횟수를 가지며 128 비트 처리를 수행할 수 있으며 외부로부터 해킹 및 크래킹을 방지하기 위한 재사용 할 수 있도록 구현되어야 한다.

이와 같은 조건을 만족시키는 암호알고리즘은 AES 중에서 Serpent가 가장 적합하다. 그러나 원래 Serpent 암호알고리즘은 알고리즘 자체에 다음과 같은 문제점들을 가지고 있

기 때문에 멀티미디어화 되어 가고 있는 스마트카드에 바로 적용하기에는 무리가 따른다[7][10].

첫째, LT 처리부분이 매우 복잡한 와이어드 로직으로 구현할 수 밖에 없다. 둘째, 충분한 비도 유지를 위하여 Serpent 암호알고리즘에서는 라운딩 횟수를 32번 수행한다. 마지막으로 키 스케쥴러에서 서브키를 발생시키기 위한 부분이 32 비트씩 132개로 구성된 4,224 비트에 대한 처리를 수행해야 하기 때문에 LT 부분과 비슷한 단점을 가진다.

그러므로 본 논문은 기존 Serpent 암호알고리즘의 단점을 줄이기 위하여 변형된 MT-Serpent 암호알고리즘과 암호시스템을 설계하였다.

설계된 MT-Serpent 암호시스템은 대용량, 실시간, 비도 유지 등의 장점을 유지할 수 있도록 암호화 및 복호화가 동시에 가능하도록 하였으며 각 기능블록에서 사용되는 S-box, InvS-box, LT, InvLT, IP, FP를 하나의 인덱스에서 해결 가능하도록 하였다. 또한 구현된 MT-Serpent 암호시스템을 재구성 할 수 있도록 설계함으로서 비인가자로부터의 해킹 및 크래킹으로부터 안전할 수 있도록 구성하였다.

변형된 MT-Serpent 암호알고리즘은 다음과 같이 크게 4가지 부분 즉, 초기 치환(IP) 및 마지막 치환(FP), 32 라운드 연산, LT(InvLT) 및 S(InvS) 변환, 키 스케쥴러로 분류된다.

1. 초기 치환 및 마지막 치환

IP 및 FP는 1:1 대응되는 변환표를 이용하여 데이터를 변환시킨다. 또한 IP 및 FP는 비도에 영향을 주는 파라미터이므로 기존 Serpent에서 사용되는 IP 및 FP 내용을 재사용한다. 그러나 변형된 MT-Serpent 암호알고리즘에 사용되는 IP 및 FP는 기존 Serpent에서 사용되는 IP 및 FP를 하나의 표를 이용하여 구성하였다.

IP와 FP의 관계는 IP1과 IP2의 관계와 같고 동시에 표현하면 식 (1)과 같다.

$$\begin{aligned} 4n(IP1) &\xleftrightarrow{n\leftarrow 0\sim 31} i(IP2) \\ 4n+1(IP1) &\xleftrightarrow{n\leftarrow 32\sim 63} i(IP2) \\ 4n+2(IP1) &\xleftrightarrow{n\leftarrow 64\sim 95} i(IP2) \\ 4n+3(IP1) &\xleftrightarrow{n\leftarrow 96\sim 127} i(IP2) \end{aligned} \quad \cdots\cdots (1)$$

여기에서 $0 \leqq i \leqq 127$이다. 식 (1)은 IP1과 IP2를 동시에 표현한 것으로서 시스템 설계시 IP 및 FP를 하나의 인덱스에서 구현할 수 있다는 것을 보여준다.

2. 32 라운드 연산

기존 Serpent 암호알고리즘은 32번의 라운드 연산을 수행하였다. 이러한 라운드 연산에 맞추어 키 스케쥴러에서도 33개의 서브키를 생성하게 된다[10].

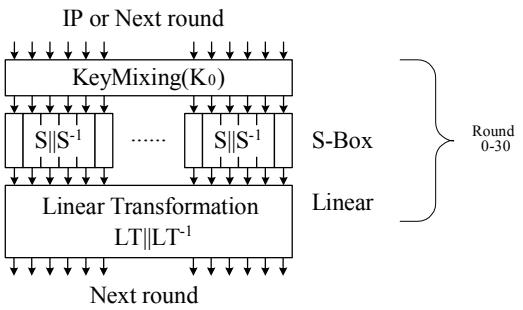
이때 라운드 연산을 수행함에 있어서 라운드를 처리하는 방법은 크게 4가지가 있다. 즉, IL(Iterative Looping), ILPLU(Iterative Looping with Partial Loop Unrolling), FLU(Full Loop Unrolling), FP(Full Pipelining)이다.

변형된 MT-Serpent 암호알고리즘도 32번의 라운드 연산을 수행한다. 라운드 처리방법 4가지는 처리속도 면에서 거의 차이가 없다. 그러나 시스템 복잡도 및 동기화에 영향을 주기 때문에 암호시스템 전체 성능에 매우 많은 영향을 미치게 된다.

또한 기존 라운딩 구조를 스마트카드에 적용하지 못하는 이유는 암호화와 복호화에 사용되는 LT, S-box가 독립적으로 존재한다는 것이다. 그러므로 MT-Serpent는 FLU 구조를 사용하여 식 (2)를 만족시켜 라운딩 구조에 관여되는 LT, S-box 등을 변화시켜준다.

그림 1은 변형된 MT-Serpent 암호알고리즘을 위한 라운드 구조이다. 암호화와 복호화를 동시에 수행하기 위하여 LT와 LT^{-1}, S와 S^{-1}을 하나의 인덱스에 포함시켰다.

$$\begin{aligned} LT &\xleftrightarrow[de]{en} LT^{-1} \\ S &\xleftrightarrow[de]{en} S^{-1} \end{aligned} \quad \cdots\cdots (2)$$



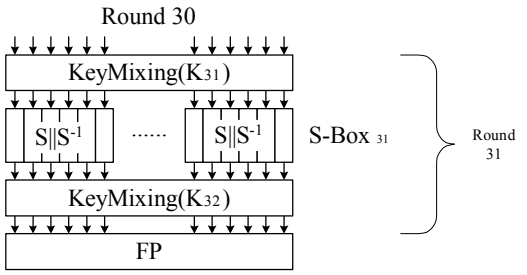


그림 1. FLU를 적용한 변형된 Serpent 라운드 구조
Fig.1 FLU applied Modified Serpent Round Structure

3. LT(InvLT) 및 S(InvS) 변환

표 1은 LT와 InvLT를 하나의 인덱스로 구현한 표의 일부분이다. LT 및 InvLT 변환은 128 비트들에 대한 1:1 선형 변환이다. 표 1은 LT와 InvLT를 하나의 인덱스 안에 구현함으로서 와이어드 로직 복잡도가 감소되도록 하였다. 또한 암호화와 복호화가 동시에 가능함으로서 처리속도의 증가를 가져올 수 있다.

기존 Serpent 암호알고리즘과 동일하게 변형된 MT-Serpent 암호알고리즘은 32개의 라운드를 수행하는 중, 각 라운드는 Key Mixing을 수행하게 되는데 이때 서브키 K_i는 현재 데이터 B_i와 배타적 논리합을 수행한다[10].

표 2는 S-box와 InvS-box에 대하여 하나의 인덱스로 표현한 표이다. 각 셀들은 각각의 S-box들에 대하여 1:1 대응관계를 가지므로 4 비트 입력에 4 비트 출력값을 가진다. 이때 각 셀들 중 하나의 값을 가지는 경우는 S-box와 InvS-box의 값이 동일한 경우이고, a~z, α~δ으로 표현한 값들은 각각의 S-box들에 대한 동일한 변환을 의미한다.

즉, a=(3, 13), a′=(13, 3), b=(8, 3), b′=(3, 8), c=(1, 0), c′=(0, 1), d=(11, 12), d′=(12, 11), e=(14, 1), e′=(1, 14), f=(2, 7), f′=(7, 2), g=(7, 15), g′=(15, 7), h=(0, 9), h′=(9, 0), i=(9, 8), i′=(8, 9), j=(12, 2), j′=(2, 12), k=(15, 5), k′=(5, 15), l=(12, 8), l′=(8, 12), m=(7, 14), m′=(14, 7), n=(9, 15), n′=(15, 9), o=(5, 12), o′=(12, 5), p=(10, 3), p′=(3, 10), q=(11, 4), q′=(4, 11), r=(9, 4), r′=(4, 9), s=(12, 14), s′=(14, 12), t=(10, 1), t′=(1, 10), u=(15, 2), u′=(2, 15), v=(13, 0), v′=(0, 13), w=(1, 3), w′=(3, 1), x=(14, 6), x′=(6, 14), y=(11, 8), y′=(8, 11), z=(8, 7), z′=(7, 8), alpha=(9, 14), alpha′=(14, 9), beta=(1, 5), beta′=(5, 1), gamma=(7, 13), gamma′=(13, 7), delta=(0, 11), delta′=(11, 0)로서 S-box와 Inv S-box와의 관계와 같다.

표 2의 변형된 S-box는 기존 Serpent 암호알고리즘에서 사용되는 S-box와 InvS-box를 보다 단순하게 수행하기 위하여 통합한 형태이다. 기존 S 함수들은[10] 모두 8x16=128가지의 2배인 256가지의 경우의 숫자가 존재하였지만 표 2와 같이 통합하는 경우에는 식 (3)과 같은 경우의 수가 존재하게 된다.

$$\begin{aligned} &\{en = de\} := 15 \\ &\{en, de\}!! := 30 \\ &\{en, de\}/! := 48 \qquad \cdots\cdots (3) \\ &\frac{256}{93} \;=\; 2.77 \end{aligned}$$

여기에서 $\{en = de\}$는 암호화와 복호화에 사용되는 S, InvS의 값이 동일한 경우이며, $\{en,de\}!!$는 S, InvS의 값이 여러번 중첩되어 사용되는 경우이고, $\{en,de\}/!$는 동일하지도 않으며 여러번 중첩되지도 않는 함수값 들이다. 또한 $:=$는 S-box에서 실제 계산될 수를 의미한다. 그러므로 기존 S-box, InvS-box들에 비하여 변형된 S-box, InvS-box들은 2.77배의 크기 및 복잡도 감소를 가져온다.

표 1. LT와 InvLT의 중첩
Table 1. Reiteration of LT and InvLT

{LT component}	{Inv LT component}		
{16 52 56 70 83 94 105} {53 55 72}	{72 114 125} {1 5 20 90}	{2 9 15 30 76 84 126} {15 102}	{36 90 103} {3 31 90}
{20 56 60 74 87 98 109} { 57 59 76}	{1 76 118} {5 9 24 94}	{2 6 13 19 34 80 88} {19 106}	{40 94 107} {7 35 94}
...	...	...	...
{8 44 48 62 75 86 97} {45 47 64}	{64 106 117} {12 82 93 121 125}	{1 7 22 76 118 122} {7 94}	{28 82 95} {0 23 82 93 109 111 123}
{12 48 52 66 79 90 101} {49 51 68}	{68 110 121} {1 16 86 97 125}	{5 11 26 80 122 126} {11 98}	{32 86 99} {4 27 86 97 113 115 127}

표 2. S-box와 InvS-box의 중첩
Table 2. Reiteration of S-box and InvS-box

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S0 InvS 0	a	b	15 11	c	10	6	5	d	e	13 14	4	f	g	h	i	j
S1 InvS 1	k	l	2	m	n	0 6	o	p	1 11	q	m′	i′	6 1	13	p′	4 0
S2 InvS 2	l′	6 9	g	r	3 11	s	t	u	v	w	x	4 13	0 5	y	5 10	f
S3 InvS 3	0	n′	11 10	z	12 11	α	6	a	a′	β	j′	4 2	10 4	z′	k′	e
S4 InvS 4	β	15 0	8	3	12 10	h	11 7	x′	2	o	q′	10 6	r	14 15	γ	13 1
S5 InvS 5	15 8	5 15	2	11 9	4	t	9 13	s	δ	3 6	14 5	b	γ′	6 12	7 10	c
S6 InvS 6	g	2 10	12 1	5 13	8 5	4 3	6	δ′	14 4	9	e′	g′	13 2	3 12	10 8	0 11
S7 InvS 7	w	v	15 6	0 13	α′	8 14	u′	y	7 5	4 12	d′	10 7	9 10	w′	5 4	6 2

4. 키 스케쥴러

기존 키 스케쥴러 블록의 S-box는 암호화와 복호화에 별도로 동작되므로 암호화 및 복호화 기능을 동시에 수행할 수 없다. 그러므로 MT-Serpent의 변형된 키 스케쥴러 블록은 그림 2와 같이 표현된다.

변형된 키 스케쥴러 블록은 기존 Serpent와 같이 32 비트씩 8개로 분할되고 분할된 확장 키 정보는 PGF(Prekey Generation Function) 블록의 입력으로 사용된다.

PGF 블록은 기존 Serpent PGF 블록과 동일하다. 즉, 32 비트씩 132개로 데이터를 변환시키는 기능을 수행한다. 이렇게 생성된 132개, 4,224 비트인 W_j 데이터들은 각각 4 비트씩 1:1 대응되는 비선형함수인 S-box에 각각 적용되어 식 (4)와 같이 128 비트의 subkey를 생성하게 된다. 여기에서 k 및 l 값의 범위는 $0 \le k, l \le 32$이고 $K_{k_{en}}$는 암호화에 사용되는 킷값이고 $K_{k_{de}}$는 복호화에 사용되는 킷값이다. 또한 $\hat{K}_l$은 각 라운드에 적용될 서브키로서 전체 서브키에 대한 부분합을 의미한다.

$$\begin{aligned} K_{k_{en}} &= S-\boxed{(W_j)} \\ K_{k_{de}} &= InvS-\boxed{(W_j)} \\ \hat{K}_l &= IP(K_k) \end{aligned} \quad \cdots\cdots (4)$$

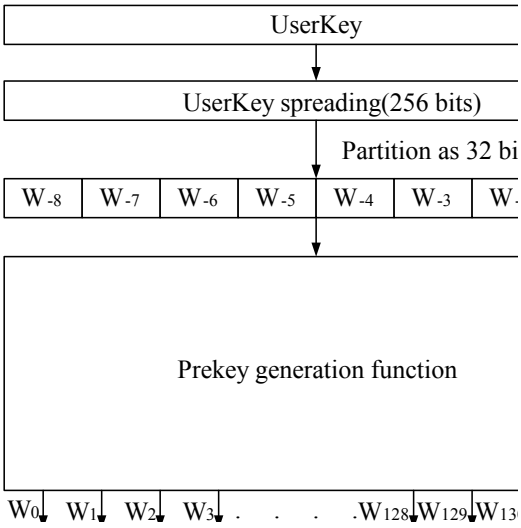


그림 2. 변형된 키 스케줄러 블록
Fig. 2 Modified Key Schedular Block

라운드 키는 비트 슬라이스 모드에서 S-box와 InvS-box를 이용하여 Prekey로부터 계산된다. 즉, 식 (5)와 식 (6)과 같이 서브키 k_i는 Prekey w_i와 S-box, InvS-box를 이용하여 생성된다.

$$\begin{aligned} \{k_0,k_1,k_2,k_3\} &:= S_{3en}(w_0,w_1,w_2,w_3) \\ \{k_4,k_5,k_6,k_7\} &:= S_{2en}(w_4,w_5,w_6,w_7) \\ \{k_8,k_9,k_{10},k_{11}\} &:= S_{1en}(w_8,w_9,w_{10},w_{11}) \\ \{k_{12},k_{13},k_{14},k_{15}\} &:= S_{0en}(w_{12},w_{13},w_{14},w_{15}) \\ &\cdots \\ \{k_{124},k_{125},k_{126},k_{127}\} &:= S_{4en}(w_{124},w_{125},w_{126},w_{127}) \\ \{k_{128},k_{129},k_{130},k_{131}\} &:= S_{3en}(w_{128},w_{129},w_{130},w_{131}) \end{aligned} \quad \cdots\cdots (5)$$

$$\{k_0,k_1,k_2,k_3\} := S_{3de}(w_0,w_1,w_2,w_3)$$
$$\{k_4,k_5,k_6,k_7\} := S_{2de}(w_4,w_5,w_6,w_7)$$
$$\{k_8,k_9,k_{10},k_{11}\} := S_{1de}(w_8,w_9,w_{10},w_{11})$$
$$\{k_{12},k_{13},k_{14},k_{15}\} := S_{0de}(w_{12},w_{13},w_{14},w_{15})$$
$$\cdots$$
$$\{k_{124},k_{125},k_{126},k_{127}\} := S_{4de}(w_{124},w_{125},w_{126},w_{127})$$
$$\{k_{128},k_{129},k_{130},k_{131}\} := S_{3de}(w_{128},w_{129},w_{130},w_{131}) \quad \cdots\cdots (6)$$

식 (5)와 식 (6)과 같이 생성된 서브키는 식 (7)과 같이 128 비트 라운드 키를 생성한다.

$$K_{ien} := \{k_{(4i)en},k_{(4i+1)en},k_{(4i+2)en},k_{(4i+3)en}\}$$
$$K_{ide} := \{k_{(4i)de},k_{(4i+1)de},k_{(4i+2)de},k_{(4i+3)de}\} \quad \cdots (7)$$

식 (7)은 IP 연산 수행 후 최종 서브키를 생성한다.

Ⅳ. MT-Serpent 암호시스템 설계

MT-Serpent 암호시스템의 경우, FLU 구조를 가지며 암호화와 복호화를 동시에 수행할 수 있도록 하기 위하여 LT 및 S 함수의 암호화/복호화 기능을 하나의 블록으로 처리하였다.

변형된 MT-Serpent에 사용되는 라운딩은 내부에 LT 및 LT-1, S 및 S-1을 포함하고 있다.

라운드 0-30까지는 KeyMixing과 $S \| S^{-1}$ 그리고 $LT \| LT^{-1}$ 연산을 31번 반복연산을 수행하게 된다. 마지막 라운딩인 32번째는 KeyMixing 32번째 연산을 수행한 후 $S \| S^{-1}$ 32번째 연산을 수행하고 다시 KeyMixing 33번째 연산을 마지막으로 수행한 후 IP2를 수행하여 암호화 및 복호화과정을 마무리한다.

그림 3과 그림 4는 MT-Serpent 암호시스템의 합성된 전체 회로 및 모의실험 파형이다. 그림 3과 같이 MT-Serpent 암호시스템은 FLU 구조를 가지며 암호화와 복호화가 동시에 가능하도록 설계되었다. 또한 그림 4는 변형된 MT-Serpent 암호시스템의 모의실험 결과이다. 모의실험 결과에서 32회의 라운딩 연산 후 출력되는 결과값이 기존 Serpent 암호시스템에 비하여 30% 빨리 출력됨을 확인하였고 사용된 게이트 숫자도 거의 1/2배 감소함을 확인하였다.

표 3은 기존 Serpent 암호시스템과 제안된 MT-Serpent 암호시스템과의 성능비교를 수행한 표이다. 표 3에서와 같이 제안된 MT-Serpent 암호시스템은 기존 Serpent 암호시스템에 비하여 throughput이 30% 증가되었으며 게이트 수에서도 거의 1/2배 감소함을 확인할 수 있다.

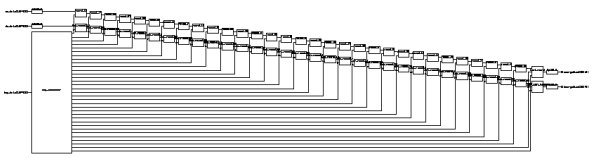


그림 3. MT-Serpent 암호시스템
Fig. 3 MT-Serpent Cryptographic System

그림 4. MT-Serpent 암호시스템의 모의실험
Fig.4 Simulation of MT-Serpent Cryptographic System

표 3. Serpent 암호시스템 성능분석
Table 3. Performance of Serpent Cryptographic System

@2.44MHz	기존 Serpent 암호시스템	제안된 MT-Serpent 암호시스템	Software (C++)
throughput [Mbps]	312.32	406.2	26.90
gatecounting [CLB Slices]	8,103	4,032	.
round counting	32/4/1	1	.
architecture	IL/ILP/ FLU	FLU	IL/ILP/ FLU

Ⅴ.결 론

구현된 MT-Serpent 암호알고리즘은 스마트카드의 특징을 살릴 수 있도록 하기 위하여 TCP/IP 프로토콜의 보안 취약성을 보강하며 유무선 환경에서 여러 종류의 서비스가 가능하고 다수의 사용자에 대한 보안을 유지하는데 주요한 목적이

있다.

MT-Serpent 암호알고리즘은 하드웨어 기반 암호시스템으로 구현되었으며 암/복호화가 동시에 수행되도록 내부 구조를 구성하였으며 구현된 MT-Serpent 암호시스템은 기본 데이터 처리가 128 비트로서 scalable하며 재구성이 가능하기 때문에 실시간 처리 및 대용량 데이터의 암호화에 매우 유리하다.

구현된 MT-Serpent 암호시스템은 크기면에서 기존 Serpent 암호시스템에 비하여 4,032로서 1/2배 감소하였으며 처리율은 기존 암호시스템에 비하여 406.2Mbps @2.44MHz로서 2배의 성능을 가진다.

그러므로 설계된 MT-Serpent 암호시스템은 향후 스마트카드와 같이 휴대성이 강조되는 시스템의 기능을 보다 다양하게 확장함과 동시에 보안상의 문제점이 발생되지 않는 보안상 우수한 성능을 발휘할 것이다.

참고문헌

[1] W. Stallings, Cryptography and Network Security, Prentice Hall, 1998.

[2] Bruce Schneier, Applied Cryptography, Second Edition-Protocols, Algorithms, and Source Code in C. John Wiley & Sons, 1995.

[3] SCSUG, Smart Card PP V1.0, 1999.

[4] Menezes, A. Oorschot, P. and Vanstone, S. "Handbook of Applied Cryptography," CRC Press, 1997.

[5] Euro Smart Card Integrated Circuit with Embedded Software V2.0, 1999.

[6] F&G, Card Industry Directory, 1998.

[7] Integrated Circuit Chip Card Security Guidelines, Version 2, 1997.

[8] Third AES candidate conference, "AES3 Proceedings, http://csrc.nist.gov/encryption/aes/round2/conf3/papers/", pp. 44-54, April, 2000.

[9] NIST, "Draft FIPS for the AES", http://csrc.nist.gov/publications/drafts.html,Feb. 2001.

[10] http://www.cl.cam.ac.uk/~rja14/serpent.html

저 자 소 개

이 선 근(Seon-Keun Lee)

현재 : (주)소디 연구원

※ 주관심분야 : 이동통신시스템, 암호시스템, VLSI 설계

정 우 열(Woo-Yeol Jeong)

현재 : 한려대학교 멀티미디어 정보통신공학과교수

※ 주관심분야 : 이동통신시스템, 암호시스템, VLSI 설계