e-seal 보안 프로토콜을 위한 효율적인 Pseudo Random Function[1)]

민정기[1O], 강석훈[1], 정상화[2], 김동규[3]
[1,2]부산대학교 컴퓨터공학과
[3]한양대학교 전자통신컴퓨터공학부*
[1]{jkmin, shkang}@islab.ce.pusan.ac.kr [2]shchung@pusan.ac.kr [3]dqkim@hanyang.ac.kr

Efficient Pseudo Random Functions for the e-seal Protection Protocol

Jung Ki Min[1O], Seok Hun Kang[1], Sang-Hwa Chung[2], Dong Kyue Kim[3]
[1,2]Dept. of Computer Engineering, Pusan National University
[3]Dept. of Electronics and Computer Engineering, Hanyang University

요 약

e-seal은 RFID(Radio Frequency IDentification) 기술을 사용하여 원격에서 자동으로 봉인상태를 확인할 수 있는 컨테이너 봉인 장치를 말한다. RFID의 특징상 반도체 칩에 기록된 정보를 제 삼자가 쉽게 판독 및 변조할 수 있다는 취약점으로 인하여 활성화되지 못하고 있는 실정이다. ISO에서는 RFID의 취약점을 보안하기 위한 표준작업(ISO 18185)을 진행 중이다. 이 중, ISO 18185-4는 e-seal에 저장되는 자료나 리더와의 RF통신에서 데이터 보호를 위한 표준이다. 이와 관련된 연구로는 인증 프로토콜과 ISO 18185-4를 위한 보고서로 제출된 보안 프로토콜이 있다. 제안된 e-seal 보안 프로토콜을 적용하기 위해서는 e-seal과 리더 간의 데이터를 암/복호화할 키가 필요하지만, 키 서버를 통해 전달받은 마스터 키를 데이터 암/복호화 키로 바로 사용하는 것은 보안 상의 문제점을 야기할 수 있기 때문에 PRF(Pseudo Random Function)을 이용하여 마스터 키로부터 MTK(Mutual Transient Key)를 유도하고, MTK를 암/복호화 키로 사용해야 한다. 기존의 PRF는 일방향 해시 함수(MD5, SHA 등)를 기반으로 하는 HMAC[2, 3]을 일반적으로 사용하였다. 그러나 일방향 해시 함수는 e-seal과 같은 제한된 자원을 갖는 환경에 적합하지 않다. 따라서, 본 논문에서는 e-seal 보안 프로토콜을 위한 효율적인 PRF을 제안한다. 기존의 일방향 해시 함수 기반이 아닌 블록 암호화 알고리즘을 기반으로 하는 MAC을 이용하여 PRF을 보다 효율적으로 구현하였고, 블록 암호화 알고리즘은 AES를 선택 합성체 $GF((2^4)^2)$을 통해 하드웨어 모듈을 최적화 하였다. AES를 기반으로 하는 MAC은 HMAC에 비해 면적 및 처리율에서 뛰어난 결과를 보여주었다.

1. 서 론

e-seal은 RFID(Radio Frequency IDentification) 기술을 사용하여 원격에서 자동으로 봉인상태를 확인할 수 있는 컨테이너 봉인 장치를 말한다. e-seal은 미국의 911테러 이후, 국제 유통 네트워크를 통한 테러위협 방지에 관심이 높아지면서 그 역할과 관심이 증가하고 있다. 그러나 RFID의 특징상 반도체 칩에 기록된 정보를 제 삼자가 쉽게 판독 및 변조할 수 있다는 취약점이 있는 실정이다.

ISO에서는 화물 컨테이너용 E-Seal에 대한 표준을 제정하는 하기 위해 TC104 SC4 WG2를 두고, RFID의 취약점을 보안하기 위한 표준작업(ISO 18185)을 진행 중이다. 이 중, ISO 18185-4는 e-seal에 저장되는 자료나 리더와의 통신에서 데이터 보호를 위한 표준이다. 이와 관련된 연구로는 인증 프로토콜[1, 2]과 ISO 18185-4를 위한 보고서로 제출된 보안 프로토콜[3]이 있다.

e-seal 보안 프로토콜[3]을 적용하기 위해서는 e-seal과 리더 간의 데이터를 암/복호화할 키가 필요하다. 하지만, 키 서버를 통해 전달받은 마스터 키를 데이터 암/복호화 키로 바로 사용하는 것은 보안 상의 문제점을 야기할 수 있다. 따라서 PRF(Pseudo Random Function)을 이용하여 마스터 키로부터 MTK(Mutual Transient Key)를 유도하고, MTK를 암/복호화 키로 사용해야 한다.

기존의 PRF는 일방향 해시 함수(MD5, SHA 등)를 기반으로 하는 HMAC(Keyed-Hash Message Authentication Code)[4]을 일반적으로 사용하였다. 하지만, 일방향 해시 함수는 하드웨어로 구현하였을 때, 많은 면적을 차지하고 낮은 처리율을 갖는 단점으로 인해 e-seal과 같은 제한된 자원을 갖는 환경에는 적합하지 않다.

본 논문에서는 e-seal 보안 프로토콜을 위한 효율적인 PRF을 제안한다. 기존의 일방향 해시 함수 기반이 아닌 블록 암호화 알고리즘을 기반하는 CBC-MAC(Cipher Block Chaining -MAC)[6], XCBC-MAC[7], CMAC[8]을 이용하여 PRF을 효율적으로 구현하였다. 블록 암호화 알고리즘은 AES(Advance d Encryption Standard)-128[5]을 선택하고, AES 하드웨어 모듈은 합성체 $GF((2^4)^2)$을 이용하여 최적화하였다[9]. AES -128을 기반으로 하는 MAC은 기존의 HMAC 보다 면적 및 처리율에서 뛰어난 결과를 보여주었다.

논문의 구성은 2장에서 CBC-MAC, XCBC-MAC, CMAC을 설명하고, 3장에서 구현 내용을 설명한다. 4장에서 구현 결과 및 기존의 연구 결과와 비교한다. 마지막으로 5장에서 결론 및 향후 연구 방향에 관하여 기술한다.

2. 기본지식

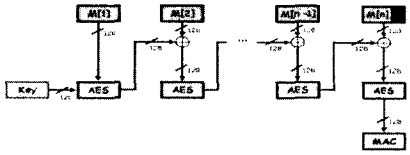
MAC은 키와 메시지를 입력으로 일정한 길이의 MAC을 계산한다. 본 장에서는 블록 암호화 알고리즘을 이용한 MAC에 대해서 설명한다.

2.1 CBC-MAC

CBC-MAC은 128 비트의 키와 임의 길이의 메시지를 이용하여 MAC을 생성한다. [그림 1]은 CBC-MAC을 생성하는 과정을 보여준다.

입력 메시지(M)를 128비트 크기의 n개 블록(M_1, M_2, ..., M_n)으로 나눈다. M_n의 크기가 128이 되지 않으면 {10i}로 패딩하여 128 비트 블록으로 만든다. 그리고 입력된 키를 이용하여 메시지를 CBC 모드로 암호화하고, 마지막 블록의 결과를 MAC으로 출력한다.

1) 이 논문은 교육인적자원부 지방연구중심대학육성사업(차세대물류IT 기술연구사업단)의 지원에 의하여 연구되었음.



[그림 1] CBC-MAC 생성과정

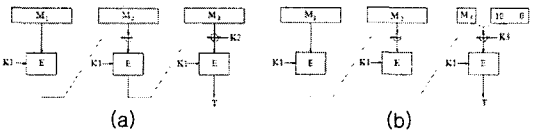
2.2 XCBC-MAC

XCBC-MAC은 CBC-MAC을 기반으로 동작한다. XCBC-MAC은 입력된 키를 이용하여 계산되어지는 SubKey(K_1, K_2, K_3)를 사용한다. K_1은 암호화 과정에서 블록 암호화 알고리즘의 키로 사용되고, K_2와 K_3는 마지막 메시지를 암호화 할 때 메시지와 XOR 연산된다. [그림 2]는 3개의 SubKey를 생성하는 방법이다.

SubKey를 이용하여 CBC-MAC과 비슷한 과정을 통하여 MAC을 생성한다. [그림 3]은 XCBC-MAC의 동작과정을 보여준다. 입력 메시지를 CBC-MAC과 같이 n개 블록(M_1, M_2, …, M_n)으로 나눈다. M_1부터 M_{n-1}까지 CBC 모드로 암호화한다. M_n이 128비트 이면 M_{n-1}까지 암호화한 결과와 K_2, M_n을 XOR 연산하고 결과를 암호화한다[그림 3-(a) 참조]. 만약 M_n이 128비트 미만이면, {10i}로 패딩하고, M_{n-1}까지 암호화한 결과와 K_3, M_n을 XOR 연하고 결과를 암호화한다[그림 3-(b) 참조].

$$K_1 = E_{InputKey}(\text{Const1})$$
$$K_2 = E_{InputKey}(\text{Const2})$$
$$K_3 = E_{InputKey}(\text{Const3})$$

[그림 2] XCBC-MAC의 SubKey 생성



(a) (b)

[그림 3] XCBC-MAC의 동작 과정

2.3 CMAC

CMAC도 CBC-MAC을 기반으로 동작하는 모드이다. CMAC은 두 개의 SubKey(K_1, K_2)를 생성하고, XCBC-MAC의 M_n블록을 처리할 때와 같이 사용된다. [그림 4]는 SubKey를 생성하는 과정을 보여준다.

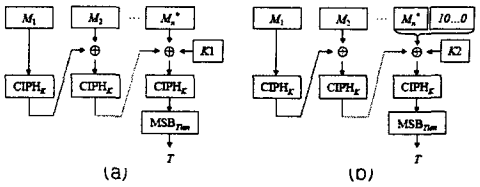
[그림 5]에서 CMAC의 생성과정을 보여준다. 동작은XCBC-MAC과 같고, XCBC-MAC의 K_2, K_3대신 CMAC의 K_1, K_2를 사용한다. 그리고 마지막 암호화 결과 중 일부 또는 전체를 MAC으로 출력한다.

1. $L = E_{InputKey}(0^{128})$
2. If $\mathbf{MSB}_1(L) = 0$, then $K_1 = L << 1$;
   else $K_1 = (L << 1)$ xor R_b
3. If $\mathbf{MSB}_1(K_1) = 0$, then $K_2 = K_1 << 1$;
   else $K_2 = (K_1 << 1)$ xor R_b

[그림 4] CMAC의 SubKey 생성

2.4 AES Specification

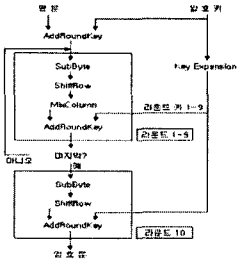
AES의 암호화 과정은 [그림 6]와 같으며, 초기 라운드 키



(a) (b)

[그림 5] CMAC 생성과정

스트림 생성, 9번의 반복 라운드 및 최종 라운드의 순서로 처리 된다. 최종 라운드를 제외한 9번의 라운드는 SubByte, ShiftRow, MixColumn 및 AddRoundKey의 변환 과정으로 구성되며, 각 연산은 4행×4열로 구성된 State단위로 이루진다.



[그림 6] AES 암호화 과정

첫째, SubByte 연산은 State를 구성하는 각각의 바이트에 대해 서로 독립적인 비선형 치환(nonlinear substitution)을 수행한다. 역변환이 가능한 두 단계의 변환 과정 즉, 유한체(Finite Field) $GF(2^8)$에서 곱셈의 역원(multiplicative inverse)을 $x \rightarrow x^{-1}$ 매핑과 affine 변환으로 구성된다.

둘째, ShiftRow 연산은 State의 값을 변경시키지 않으면서 바이트의 위치를 교환한다.

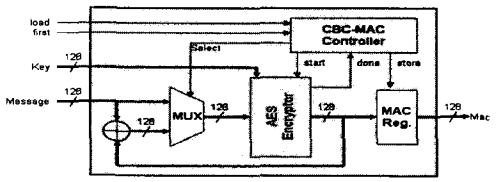
셋째, MixColumn 변환은 State의 행을 유한체 $GF((2^8)^4)$의 다항식으로 생각하여 $b(x) = c(x) \otimes a(x)$의 다항식 곱을 연산한다.

마지막으로 AddRoundKey는 State의 모든 바이트에 라운드 키를 가산하여 이를 EXOR 연산으로 처리한다.

3. PRF의 효율적인 하드웨어 구현

앞에서 언급하였듯이 e-seal과 같은 제한된 자원을 같은 환경에서 PRF는 소프트웨어보다 하드웨어로 구현하는 것이 적합하다. 이번 장에서는 앞에서 설명한 블록 암호화 알고리즘을 기반으로 하는 MAC 알고리즘을 하드웨어로 구현한 결과에 대해서 설명한다.

3.1 CBC-MAC



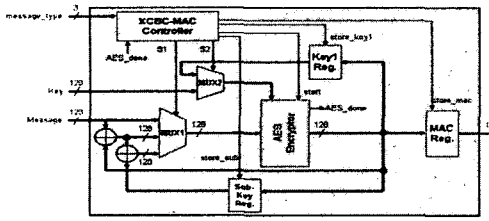
[그림 7] CBC-MAC 구조

CBC-MAC은 [그림 7]과 같은 구조를 갖는다. 블록 암호화 알고리즘은 AES의 암호화 모듈만 사용하였다. MAC 레지스터는 현재까지 계산된 MAC값을 저장하고 있다. AES의 입력 앞에 위치하는 MUX의 경우 first 시그널을 활성화 되었을 때 입력 메시지를 바로 AES로 전달하고, 아닐 경우에는 현재까지

계산된 MAC값과 XOR 연산을 한 결과를 AES로 전달한다. 성능 향상을 위하여 AES의 출력을 MAC 레지스터를 거치지 않고 입력 메시지와 XOR하는 부분으로 피드백한다. 마지막으로 Controller는 load와 first 시그널을 바탕으로 Select와 start 시그널을 생성하고, AES 모듈의 done 시그널을 바탕으로 store 시그널을 생성한다.

3.2 XCBC-MAC

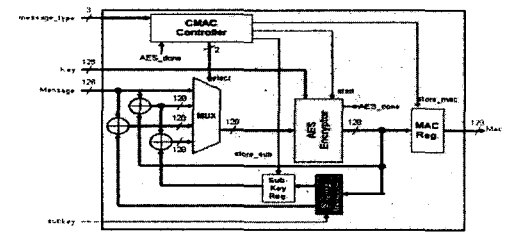
XCBC-MAC은 [그림 8]과 같은 구조를 갖는다. CBC-MAC과 같이 블록 암호화 알고리즘은 AES의 암호화 모듈만을 사용하였다. XCBC-MAC은 2개의 레지스터와 1개의 MUX를 추가하였다. Key1레지스터는 Const1와 Key를 입력으로 암화된 결과를 저장하여 이후 메시지를 암호화 할 때, AES의 Key로 사용된다. SubKey는 Const2또는 Const3과 Key를 입력으로 생성된 값을 저장한 후, 마지막 메시지와 XOR된다. XCBC-MAC Controller는 Message_type과 AES_done 시그널을 바탕으로 MUX의 Select 시그널(S1, S2)과 각 레지스터의 store 시그널(store_key1, store_sub, store_mac), 그리고 AES의 start 시그널을 생성한다.



[그림 8] XCBC-MAC 구조

3.3 CMAC

CMAC은 [그림 9]과 같은 구조를 갖는다. 앞의 두 MAC 알고리즘과 같이 블록 암호화 알고리즘은 AES의 암호화 모듈만을 사용하였다. CMAC은 CBC-MAC와 비교하였을 때, SubKey 레지스터와 SubKey 생성 모듈이 추가된다. SubKey 생성 모듈은 AES의 결과와 subkey 시그널을 바탕으로 Key1 또는 Key2를 생성하여 그 결과를 바로 AES의 입력으로 피드백하거나 SubKey 레지스터로 보낸다. Controller는 XCBC-MAC과 비슷하게 Message_type 시그널을 바탕으로 select 및 store, start 시그널을 생성한다.



[그림 9] CMAC 구조

3.4 AES Encryptor

AES는 $GF(2^8)$상의 연산으로 구성된다. 본 논문에서는 [9]에서와 같이 State의 각 원소를 합성체 $GF((2^4)^2)$로 변환하여 연산을 수행하는 모듈[9]로 구현하였다.

4. 실험 결과 및 성능 분석

하드웨어 모듈은 VHDL로 구현하였고, Xilinx ISE 7.1.04i를 이용하여 Synthesis 및 P&R 하였다. 그리고, Vertex2P xc2vp30-6 FPGA에서 동작을 검증하였다.

[표 1] MAC의 성능 비교

MAC	Core	면적 (CLBs)	f_{max} (MHz)	처리율 (Mbps)
HMAC	MD5	3161	40.7	70.77
	SHA1	2648	77.9	144.7
CBC-MAC	AES	1900	73.4	939.5
XCBC-MAC	AES	2530	81.9	700.0
CMAC	AES	2458	82.1	840.7

[표 1]은 기존의 HMAC과 본 논문에서 제안하는 블록 암호화 알고리즘 기반의 MAC의 면적 및 처리율에 관한 비교한 것이다. 처리율의 계산은 입력 데이터의 크기가 512 비트로 가정하였다.

기존의 HMAC은 블록 암호화 알고리즘을 이용하는 MAC에 비해, 면적 면에서 약 1.04 ~ 1.66 배 정도 많은 면적을 차지하는 좋지 못한 결과를 나타낸다. 또한, 처리율 면에서는 블록 암호화 알고리즘을 이용하는 MAC이 HMAC에 비해서 약 4.84 ~ 13.3 배 정도 좋은 성능을 보였다.

5. 결론 및 향후 연구

본 논문은 효율적인 PRF의 하드웨어 구현을 위하여 기존의 HMAC 기반의 PRF와 블록 암호화 알고리즘(AES)를 이용한 MAC 기반의 PRF를 하드웨어로 구현 및 성능 분석을 하였다. AES를 기반으로 하는 MAC은 기존의 HMAC에 비하여 면적 및 성능 측면에서 뛰어난 결과를 보였다.

향후 연구 방향으로 제한된 자원을 가지는 환경에서 PRF의 저전력 하드웨어 모듈 구현에 관한 연구가 필요할 것이다.

[참고문헌]

[1] Seongsoo Park, Mun-Kyu Lee, Dong Kyue Kim, Kunsoo Park, Yousung Kang, Sokjoon Lee, Howon Kim, Kyoil Chung, "Design of an Authentication Protocol for Secure Electronic Seals", CIS(International Conference on Cybernetics, Informatics and Systemics)'05, 2005

[2] 박성수, 이문규, 김동규, 박근수, 김호원, 정교일, "안전한 전자봉인을 위한 인증 프로토콜 설계", 한국정보과학회 추계학술발표대회논문집, 8권 2호, pp.279-282, 2005

[3] ETRI, "Report of ePP(eSeal Protection Protocol) for ISO 18185-4", October 2005

[4] NIST FIPS PUB 198, "The Keyed-Hash Messge Authentication Code(HMAC), March 2002

[5] NIST FIPS PUB 197, "Advanced Encryption Standard(AES)", November 2001

[6] NIST FIPS PUB 113, "Computer Data Authentication", May 1985

[7] J. Black, P. Rogaway, "eXtended Cipher Block Chaining MAC", NIST Proposed Mode of Operation

[8] NIST SP800-38B, "Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication", MAY 2005

[9] Charanjit S. Jutla, Atri Rudra, and Pradeep K. Dubey, "Efficient Implementation of Rijndael Encryption with Composite Field Arithmetic, "in CHES, CHES 2001:Paris, France, May 14-16 2001, pp. 171-.184, Springer.