IPsec의 Message Authentication Module을 위한 HMAC의 설계

하진석, 이광엽, 곽재창
서경대학교
전화 02-940-72402/핸드폰 018-399-4258

Design of a HMAC for a IPsec's Message Authentication Module

Jae-Chang Kwak, Jin-Suk Ha, Kwang-Youb Lee
Dept of Computer Engineering, Seo-Kyeong University
E-mail : power@home.skuniv.ac.kr

Abstract

In this paper, we construct cryptographic accelerators using hardware implementations of HMACs based on a hash algorithm such as MD5.It is basically a secure version of his previous algorithm, MD4 which is a little faster than MD5.The algorithm takes as input a message of arbitrary length and produces as output a 128-bit message digest. The input is processed in 512-bit blocks. In this paper, new architectures, iterative and full loop, of MD5 have been implemented using Field Programmable Gate Arrays(FPGAs). For the full-loop design, the performance is about 500Mbps @ 100MHz. .

I. 서론

현재 인터넷은 TCP/IP 프로토콜 중 네트워크 layer의 프로토콜로 IPv4(Internetworking Protocol, version 4)를 사용하고 있다. 비록 IPv4가 설계가 잘되어 있다고는 하지만 데이터 통신은 1970년대에 IPv4가 나온 이래에 발전을 거듭하여 왔다. IPv4는 빠르게 발전하는 인터넷에 비해 주소공간이 부족하고, 최소지연과 자원의 예약을 요구하며, 정보보호가 필요로하는 분야에서 원하는 데이터의 암호화와 인증을 제공하지 않는다. 이러한 결점을 보완하기위해 IPng(Internetworking Protocol, next generation)이라고도 알려진 IPv6(Internet Protocol version 6)가 제안되었고 현재 표준이 되었다. IPv6는 엄청나게 발전하는 인터넷을 수용하기 위해 많이 수정되었며 IPv6에서는 암호화와 인증옵션들은 패킷의 신뢰성과 무결성을 등을 제공한다. 인터넷에서의 정보보호는 여러 분야에서 여러 가지 암호방식이 사용되고 있으며, 인터넷을 구성하는 여러 계층에서 이루어 질 수 있지만, 디지털 서명분야에는 해쉬함수를 이용한 서명 방식이 널리 이용되고 있다. 해쉬함수는 본래의 메시지를 축약하는 방식으로 이로 인해 디지털서명의 효율성을 높일 수 있다. 즉 디지털 서명 때 해쉬함수에 의해 축약된 메시지를 서명하게 되는데 이로 인해 서명을 위해 필요한 계산, 메모리, 전송량이 크게 줄어든다.

IPsec은 기존 application level에서의 적용되던 보안을 application 과는 독립적으로 보안이 가능하도록 고안된 패킷처리 보안기술로서, 앞으로 VPN(Virtual Private Network)을 구성하는 장비들도 IPsec를 지원할 것으로 추정되고 있다. IPsec에서는 두 개의 protocol(AH , ESP)과 두 개의 mode(turnel , transport)를 지원하는데 protocol 속성은 데이터 패킷이 기밀성 또는 메시지 무결성(또는 둘다)에 의해서 안전한지를 나타내고 mode 속성은 얼마나 많은 데이터 패킷이 승인되어 안전한지를 나타낸다. protocol과 mode의 조합으로 네 개중 하나의 데이터 패킷형태를 선택할 수 있으며 두 프로토콜 모두 무결성을 제공하

게 되는데, Message Authentication을 위해 HMAC(Keyed-Hashing for Message Authentication)을 사용한다. MAC는 MDC에 비해 훨씬 느린데 HMAC는 속도의 향상을 위하여 MAC와 MDC를 결합시킨 형태이며 MD5는 SHA-1과 더불어 HMAC에 사용되는 MAC 중의 하나이다. MD5의 취약성에 대해서 연구된 바가 있고 앞으로의 사용에 우려를 표명하는 견해가 있으나 HMAC-MD5에 대해서는 안전한 것으로 결론 짓고 있다.[5]

II. IPsec

2.1 IPsec의 보안

IETF에서는 네트워크 보안 프로토콜의 표준화를 위하여 크게 두 가지 방향으로 진행 중인데, 그림 2.1에서 보는 바와 같이 TCP/IP 프로토콜의 IP layer의 보안을 위한 IPsec과 TCP-Layer 위에서 Server/Client Application 사이에 보안 서비스를 제공하기 위한 TLS(Transport Layer Security)가 있으며, IPsec은 IETF에서 1995년 8월 IPsec을 RFC로 채택된 이후 IPSEC W/G에서 현재까지 표준화가 진행이다. TLS는 현재 널리 이용되고 있는 Netscape사에서 개발한 SSL V3를 개정하여 IETF TLS W/G에서 Internet draft로 채택된 상태이다. 현재 internet에서 이용되고 있는 IP protocol은 packet-switched network에서 단순히 데이터의 신뢰성있는 전송만을 염두에 두고 개발한 것이기 때문에 , 설계당시 보안은 고려되지 않았다. 따라서 IP Spoofing, IP Sniffing과 같은 보안 허점이 생겨나고, 이를 악용하는 경우가 많아지고 있다. 이러한 문제점을 해결하기 위한 방안으로 IP layer에서 보안 서비스를 제공할수 있는 IPsec이 등장하였다.

IPsec을 통해 제공되는 보안 서비스는 Authentication, integrity, confidentiality, access control, replay attack등의 5가지가 대표적이며, IPsec은 하나 이상의 연결에 대하여 호스트와 호스트사이, 호스트와 보안 게이트웨이 사이, 보안게이트웨이와 보안게이트웨이 사이 세 부분에서 보안 서비스를 제공할 수 있다

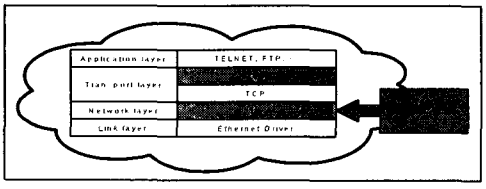


그림 2.1 IPsec과 TLS 프로토콜

IPsec은 IP-layer에서 다양한 보안 서비스를 제공하기 위한 Security Architecture for the Internet Protocol을 정의하고 있다.

III. MAC

3.1 MAC

MAC은 1970년대부터 출발한 현대 암호학 보다도 더 오랜 역사를 가지고 있는 금융분야에서 이미 오래 전부터 사용되어 온 개념이다. 하지만, 안전하고 효율적인 암호학적 특성을 지니는 MAC은 현대 암호학이 본격적으로 도입되면서 소개되기 시작하였다.

MAC 해쉬함수, $h()$는 메시지의 작성자와 수신자만이 알고 있는 비밀키 k를 이용하여 메시지 m에 대한 인증자 MAC = $h(k, m)$을 계산해 낸다. 메시지와 함께 MAC이 수신자에게로 전달되면 수신자 측에서는 전달된 메시지를 이용하여 자기 자신이 계산한 MAC과 수신된 MAC을 비교함으로써 메시지에 대한 무결성을 확인하게 된다. 송신자와 수신자만이 비밀키 k를 알고 있고, 수신된 MAC이 계산된 MAC과 일치한다면 수신자는 그 메시지가 정당한 메시지 작성자로부터 온 것이며 또한 전송 도중에 변조되어지지 않았다는 것을 확신할 수가 있게 된다. 능동적인 공격자는 비밀키 k를 모르기 때문에 메시지 변조를 수행하 수 있다고 할지라도 그 변조된 메시지에 해당하는 정확한 MAC을 계산해 내기는 계산적으로 거의 불가능하다. 그림 2-2은 MAC을 이용한 무결성 확인 과정을 보여주고 있다.

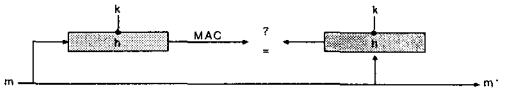


그림 2-2 MAC의 무결성확인 과정

메시지에 대한 무결성 뿐만 아니라 메시지에 대한 기밀성도 동시에 요구되어지는 경우에는 그림 2-3에서와 같이 메시지 m과 인증자 MAC= $h(k_1, m)$을 함께 암호화할 수도 있다. $m \| h(k_1, m)$은 메시지 m과 MAC의 연결(concatenation)을 의미한다. 그림 2-3 에서는 MAC의 생성에는 비밀키 k_1이 적용되고 MAC을 포함한 메시지의 암호화에는 비밀키 k_2가 이용된다. 물론 메시지의 암호화와 MAC의 생성에 동일한 비밀키를 사용할 수도 있다. 이 문제는 보안적인 측면과 키의 생성 및 분배와 연계된키 관리의 효율적인 측면을 고려하여 선택할 수 있게 된다.

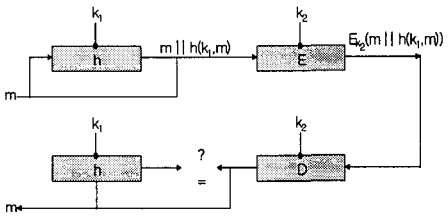


그림 2-3 비밀키를 이용한 MAC의 생성

가장 대표적인 MAC은 CBC(cipher block chaining) 모드로 운영되는 DES에 기반을 두고 있다. 이는 금융권에서 무결성의 목적으로 사용하기 위해 표준화된 MAC으로서 그 구성 방식은 다음과 같다. 메시지 m이 t개의 64 비트들의 블록 $m_1 m_2 .... m_t$로 구성된다면 l비트의 MAC은 다음과 같이 계산된다. 이때, 마지막 블록이 64비트로 구성이 안될 경우에는 부족한 만큼을 '0'으로 채워준다.

$$c_1 = E_k(IV \oplus m_1),\ c_2$$
$$= E_k(c_1 \oplus m_2), \ldots\ldots,\ c_t$$
$$= E_k(c_{t-1} \oplus m_t)$$

초기화 벡터 IV는 0이고 마지막 블록 c_t중에서 l (≤64)비트를 MAC으로 선정한다.

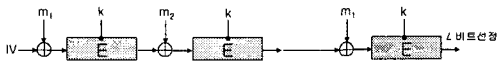


그림 2-4 메세지 축약과정

MAC이 실용적인 인증자로서 사용되고 또한 불법적인 제 3자의 능동적인 공격으로부터 안전하기 위해서는 다음과 같은 제약 조건이 MAC 해쉬함수에 요구된다.

3.2 HMAC 의 구조

HMAC는 기본적으로 어떤 crypto-graphic hash function 도 사용이 가능하며, hash function 의 변형과 성능의 저하되지 없이 사용할 수 있다. HMAC의 구조를 간단히 표현하면 다음과 같다.

HMAC(L) = H(K xor opad , H(K xor ipad, text)

H=cryptographic hash function

K=secret key

ipad = the byte 0x36 repeated B times

opad = the byte 0x5c repeated B times

B = the byte-length of such block

L = the byte-length of hash outputs

즉 , secret key 에 '0'을 덧붙여 B 의 길이로 만든후 (secret key의 길이가 B보다 큰 경우 H(secret key)가 새로운 key 가 된다) ipad 와 xor 연산을 한다. 연산된 결과에 'text'를 덧붙여 hash function을 적용한 값을 secret key에 opad를 xor한 값에 붙여서 다시 hash function을 적용시킨다. 이때 secret key의 길이가 L보다 작을 경우 보안에 문제가 생길 수 있다.[2]

Ⅳ. MD5 회로설계

4.1 HMAC-MD5

본 논문에서는 cryptographic hash function으로서 MD5를 사용하였으며, MD5는 일반적으로 F, G, H, I 4개의 Function으로 구성 되어있다.

F(X,Y,Z) = (X and Y) or (X and Y)
G(X,Y,Z) = (X and Z) or (X and Z)
H(X,Y,Z) = X ⊕ Y ⊕ Z
I(X,Y,Z) = Y ⊕ (X or Z)

MD5의 구현은 적은 면적을 위한 iterative구조와 빠른 속도를 위한 full loop unrolling구조 두 가지로 형태로 구현하였다.

4.1.1 Iterative Architecture

Iterative 구조는 그림3-1과 같이 MD5의 4Round(64step)구성을 공통 부분인 F, G, H, I Function을 12개의 32bit Register와 8개의 32bit Adder, barrel-shifter부분을 64번의 Looping으로 처리하도록 구조를 설계하였다. Looping부분 즉 iterative Core 부분을 제외한 나머지 부분과 이것을 제어하는 FSM모듈을 이용하여 설계하였다. Shifter는 소형의 코어를 위한 구조에 맞추어 일반적인 barrel shifter의 기능을 축소하여 Left Rotate shift의 기능만을 가진 shifter를 2bit mux를 이용하여 구현하였다.

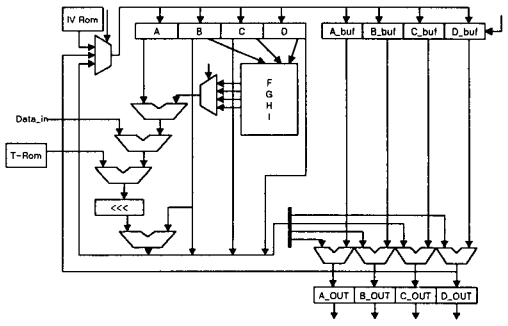
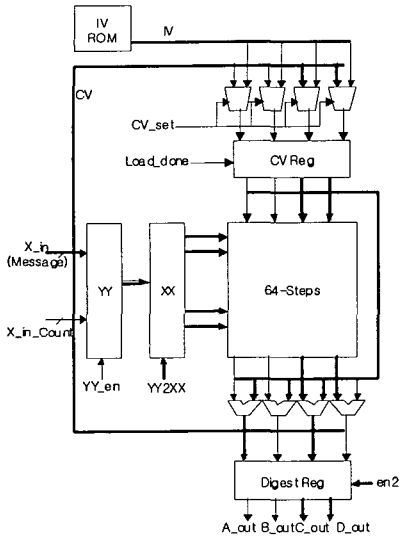


그림 3-1 Iterative Core 구조

4.1.2 Full loop unrolling Architecture



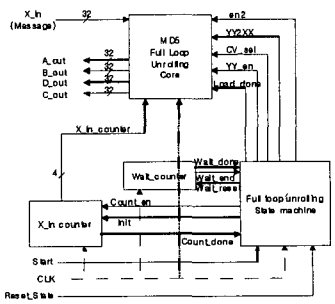
<MD5 full loop-unrolling core>

그림 3-2 Full loop 구조

full loop unrolling 구조는 그림3-2와 같이 MD5의 4round(64step)을 combination logic으로 설계한 것으로서, Iterative Architecture를 64step 모두를 중복사용 없이 Logic 순서대로 나열하여 보다 빠른 동작속도를 가진 logic으로 설계하였다. 또한 주로 software에 의존하였던, MD5의 초기과정인 padding과 appending length의 기능을 가진 모듈을 함께 설계하였다. 이 모듈로 인해서 software의 부담을 줄일 수 있고, 보다 독립적으로 동작할 수 있도록 설계하였다. Iterative Architecture에서 필요하였던 barrel shifter는 combination logic에서 직접 shifting 한 값을 할당해 줌으로서 제거할 수 있다. padding 모듈은 message의 입력 시에 counting하는 counter를 lastbyte signal의 변화로 counting을 멈추게 하여 message의 byte를 count하고, count한message의 byte에 따라서 "1000...00"을 할당한다. appending length는 count된 byte 수를 3bit left shift 하여 append하였다. 전체블록도는 그림3-2과 같다.

V. 결론

본 논문에서는 IPv6에 포함된 IPsec의 authentication으로서 사용되는 HMAC구현에 있어서 MD5를 Cryptographic hash function 으로 사용하여 Iterative 와 Full loop unrolling architecture 두 가지



< Block diagram of full-loop-unrolling design>

그림 3-3. 전체 블록도

로 모델을 구현하였다. 설계와 Simulation은 Synopsys와 Xillinx-Foundation3.1을 이용하여 설계하였다.

Interactive Architecture는 vertex-XCV800HQ-240(80만 Gate)에서 Utilization이 5%의 Size를 가지며, 동작주파수는 10㎒이다.

Full loop unrolling은XCV800HQ-240(80만 Gate)에서 Utilization이 slice 5065/9408로 58%의 size를 가지며, 동작주파수는 50㎒로 동작한다.

*본 논문은 *시스템 2010사업과 부분적으로 IDEC 지원장비로 작성됨*

참고문헌

[1] ETRI, "암호학의 기초", 경문사, 1999년 03월.

[2] William Stallings, "Cryptography And Network Security" second edition, prentice hall, 1998

[3] 이만영 외 공역, "전자상거래 보안기술", 생능출판사, 1999.8

[4] Kent, S., and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.

[5] H. Dobbertin, "The Status of MD5 After a Recent Attack", RSA Labs' CryptoBytes, Vol. 2 No. 2, Summer 1996.

[RFC-2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[RFC-1321] Rivest, R., "MD5 Digest Algorithm", RFC 1321, April 1992.

[ESP] Kent, S., and R. Atkinson, "IP Encapsulating Security Payload", RFC 2406, November 1998.