VPN을 위한 IPSec 암호프로세서의 FPGA 구현

*이 광호, 유 수봉, 전 진오, 강 민섭
안양대학교 컴퓨터공학과
e-mail : mskang@anyang.ac.kr

FPGA Implementation of IPSec Crypto Processor for VPN

*Kwang-Ho Lee, Su-Bong Ryu, Jeen-Oh Jun, Min-Sup Kang
Department of Computer Engineering
Anyang University

Abstract

본 논문에서는 VPN을 위한 IPSec 암호 프로세서의 설계 및 구현에 관하여 기술한다. IPSec 암호 프로세서의 기밀성 서비스를 위한 암호엔진은 DES, 3 DES, SEED, 그리고 AES 알고리듬 등을 사용하여 설계하였고, 인증 및 무결성 보안 서비스를 위한 인증엔진은 HMAC(The Hashed Message Authentication Code)-SHA-1을 기본으로 설계하였다.

제안된 암호 프로세서는 Verilog를 사용하여 구조적 모델링을 행하였으며, Xilinx사의 ISE 6.2i 툴을 이용하여 논리 합성을 수행하였다. FPGA 구현을 위해서 Xilinx ISE 6.2i툴과 Modelsim을 이용하여 타이밍 시뮬레이션을 수행하였다.

I. 서론

인터넷은 본질적으로 신뢰할 수 없는 네트워크들의 집합체로, 정보의 흐름을 통제하기가 대단히 어렵기 때문에 내부의 중요한 자원을 인터넷으로부터 보호해 줄 수 있는 인터넷 보안이 가장 심각한 문제로 대두되고 있다[1-3]. VPN(Virtual Private Network)은 적절한 암호기술을 이용하여 한 조직의 내부 사용자들이 사내나 사외에서 서로 안전하게 통신할 수 있는 채널을 형성해 주며 또한 필요한 접근제어 기능을 제공해 준다[1]. IPSec은 데이터의 인증과 무결성을 보장하기 위한 AH와 암호화 기법을 사용하여 데이터의 무결성, 리플레이 방지, 비밀성의 기능을 제공하기 위한 ESP(Encapsulation Security Payload)는 프로토콜로 구성된다[2].

그러나 소프트웨어로 구현된 IPSec 기반 보안관련 제품들 AH와 ESP에서의 인증 데이터의 계산 및 비교 암호화와 복호화를 수행하는데 많은 시간이 요구된다. 뿐만 아니라 단일 SG(security gateway)나 혹은 시스템 서버는 높은 bandwidth와 많은 네트워크 연결로 인해 많은 부하가 걸리게 되므로 처리속도를 따라가지 못하여 병목현상을 유발시킬 뿐만 아니라 보안체계에서도 약점을 가지게 된다[1].

본 논문에서는 IP패킷의 기밀성(Confidentiality), 무결성(Integrity), 인증(Authentication)을 위해 IPSec을 위한 암호 및 인증 프로세서의 FPGA 설계 및 구현에 관하여 기술한다. 기밀성 서비스를 위한 암호엔진은 DES/3DES, SEED, 그리고 AES 알고리듬 등을 사용하여 설계하며, 인증 및 무결성 보안 서비스를 위한 인증엔진은 HMAC-SHA-1을 기본으로 설계한다.

II. 관련 연구

2.1 IPSec 프로토콜

IPSec은 양 종단 간의 안전한 통신을 지원하기 위해 IP 계층을 기반으로 하여 보안 프로토콜을 제공하는 개방 구조의 프레임워크로서 VPN 구현에 가장 널리 사용되는 기술이다[1]. IPSec은 크게 IP 헤더를 포함하는 전체 패킷에 대한 인증 기능을 제공하는 AH(Authentication Header), IP header 이외의 payload에 대한암호화/인증 기능을 제공해 주는ESP(Encapsulating Security Payload) 헤더, 그리고 AH/ESP를 포함한 각종 인터넷 보안 서비스에 필요한 Securi

* 본 연구는 2003년도 산업자원부(제1차 부품.소재 기술개발사업)와 IDEC 지원으로 수행되었음.

ty Association Negotiation 및 Key Management를 담당하는 ISAKMP/IKE(Internet Security Association and Key Management Protocol / Internet Key Exchange)로 구성된다[1,2].

2.2 암호 알고리듬

암호시스템은 대칭형(symmetric) 암호시스템과 비대칭형(asymmetric) 암호시스템으로 분류할 수 있다. 대칭형 암호 알고리듬은 암호화키와 복호화 키가 동일한 암호 알고리듬을 말하며, DES, 3DES(Triple Data Encryption Standard), SEED, AES(Advanced Encryption Standard)등이 있다[3-5]]. 비대칭형 암호 알고리듬은 암호화키와 복호화 키가 동일하지 않은 암호 알고리듬을 말하고, RSA(Rivest-Shamir-Adleman), ECC등이 이에 속한다[1-3].

가. DES/3DES 알고리듬

대표적인 블럭 암호화 알고리듬인 DES는 64 비트의 데이터와 56 비트 길이의 키를 사용하여 64 비트의 암호화 결과를 생성한다[3]. 암호화를 위해 64 비트의 평문이 초기 치환(initial permutation)후에 32 비트씩 좌), 우 부분으로 나뉘게 되며 그 다음 16라운드의 계산을 거치게 된다. 16라운드 후에는 오른쪽과 왼쪽 부분이 합쳐져서 역 초기 치환(inverse initial permutation)을 거침으로써 암호문(Ciphertext)이 생성된다.

3DES는 두 개의 암호 키를 사용하여 첫 번째 키로 암호화하고 다시 두 번째 키로 복호화 한 다음 또 다시 첫 번째 키로 암호화하여 강력한 암호를 얻는 방식이다.

나. SEED 알고리듬

SEED는 대칭키 암호알고리듬으로, 블럭 단위로 메시지를 처리하는 블럭 암호알고리듬이며, 16개의 라운드를 가진 Feistel 구조를 가진다[4,9]. SEED의 F 함수는 수정된 64 비트의 Feistel 구조를 갖추고 있으며, 32 비트 단위의 2개의 블럭(C, D)을 입력으로 받아, 32 비트 단위의 2개의 블럭(C′, D′)을 각각 출력한다. G 함수는 F함수 및 라운드키 생성시에 사용되는 주요 함수이다.

다. AES 알고리듬

AES암호 알고리듬은 가변 블럭길이와 가변 키 길이를 갖는 반복구조(non-Feistel)의 블럭 암호 방식이다[5,8]. 블럭길이와 키 길이는 128, 192, 256 비트로 독립적으로 지정될 수 있으며 라운드 수는 블럭길이와 키 길이에 따라 결정된다. 암호화 과정은 8bit 단위의 입력에 대한 치환과정을 거치는 ByteSub함수와, 32bit 단위의 입력에 대한 행변환 과정을 거치는 ShiftRow함수, 32bit 단위의 입력에 대한 열변환 과정을 거치는 MixColumn함수, 128bit 단위의 입력에 대해 키 블럭과 xor 연산을 거치는 AddKey함수의 합성을 1회전으로 구성된다. AddKey에 사용되는 키 블럭은 매 라운드마다 새로이 생성된 키 블럭을 사용한다.

라. HMAC-SHA-1 알고리듬

HMAC은 암호학적 해쉬함수 H와 비밀키 K를 요한다[6,7]. H는 데이터 블럭에 기본압축함수를 반복하여 데이터를 해쉬하는 암호학적 해쉬함수라고 가정한다. B는 그러한 블럭의 바이트 단위의 길이이며, 앞서 언급된 해쉬함수들 에서는 B=64이다. L은 해쉬 출력의 바이트 단위의 길이이며, SHA-1에서 L=20이다. 인증키 K의 길이는 해쉬함수의 블럭 길이인 B이하의 임의의 값이다. B 바이트를 초과하는 키를 사용하는 응용 프로그램들에서는 먼저 H를 사용하여 키를 해쉬하여 출력되는 L 바이트 결과를 다시 HMAC의 실제 키로 사용한다.

SHA-1은 512비트 단위의 메시지 블럭으로 분할한 후, 각 메시지 블럭에 대해 20 스탭으로 이루어진 4 라운드의 총 80 단계의 연산 수행과 최종 연쇄변수 갱신 과정을 거쳐서 160비트의 해쉬 코드를 생성하는 알고리듬이다.

Ⅲ. IPSec 암호프로세서의 설계

그림 1은 PCI 버스 인터페이스, FIFO, packet processor, 그리고 Crypto_processor로 구성된 시스템의 전체 구성도를 나타낸다.

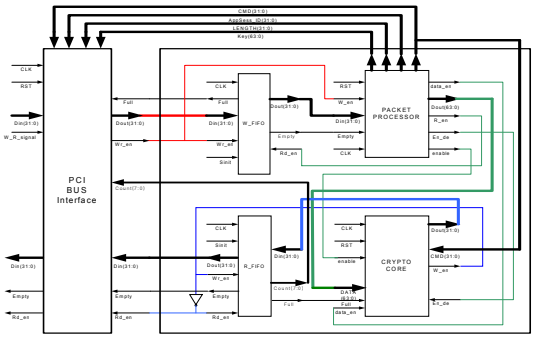

그림 1. 시스템의 전체 구성도

PCI BUS Interface 블럭은 PCI BUS로부터 입력되는 데이터를 Wr_en시그널(FIFO Write Enable)과 함께 W_FIFO 블럭에 저장하는 역할을 한다. Packet Processor 블럭은 입력된 데이터를 Key, CMD, Data, Length, AppSess를 각각 분리하여 PCI BUS Interface 블럭과

Encryption & Authentication 블럭으로 전달하는 역할을 한다. 그리고 Encryption & Authentication을 위한 Crypto_processor는 데이터를 입력받아 암복호 메시지 및 인증메세지 생성을 하여 R_FIFO 블럭에 전달하며, R_FIFO 블럭은 출력데이터를 저장하는 역할을 한다.

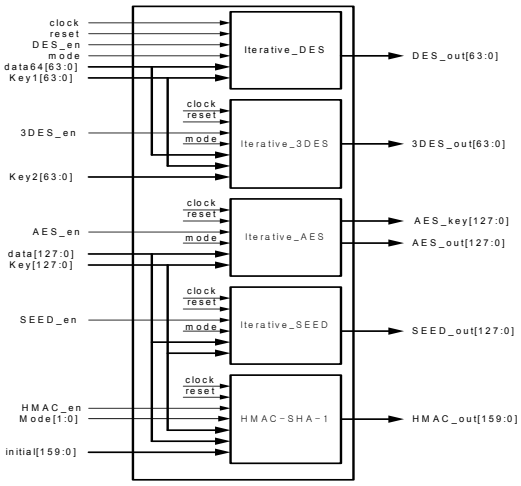

그림 2 암호프로세서의 블럭도

그림 2는 암호프로세서의 블럭도를 나타낸다. 그림 2에서 Iterative_DES, Iterative_3DES, Iterative_SEED, 그리고 Iterative_AES 블록은 제어 신호에 따라 각각 DES/3DES, SEED, 그리고 AES 알고리듬[3-5]을 수행한다. HMAC-SHA-1블록은 HMAC-SHA-1 인증 알고리듬을 수행한다.

3.1 DES/3DES 설계

그림 3은 DES알고리듬의 블럭도를 나타낸다.

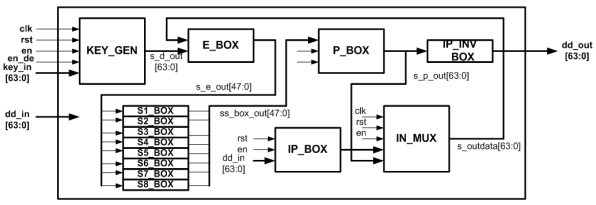

그림 3. DES 내부 블록도

KEY_GEN블럭은 DES의 각각의 라운드키를 생성하는 블럭이며, IP_BOX블럭은 64비트의 평문이 치환입력을 생성하기 위해 비트열의 순서를 재조정 하는 초기순열 블럭이며, IP_INV_BOX블럭은 초기순열의 역초기순열블럭이며, IN_MUX블럭은 DES의 각각의 라운드에 맞게 데이터를 입력해주는 블럭이며, E_BOX, P_BOX, S_BOX 블럭은 DES 알고리듬의 라운드 내에서 각각 확장순열결과와 킷값과 XOR 연산, 순열결과와 상위 32비트와 XOR 연산, 치환선택을 해주는 블록이다.

3DES는 3개의 DES블럭으로 구성이 되며, 각각의 DES블럭을 순차적으로 거쳐서 최종 3DES의 결과 값을 얻는다. 3DES의 킷값은 64비트의 2개의 키를 사용하며, Key1은 첫 번째 DES블럭과 세 번째 DES블럭에 사용되어지고, Key2는 두 번째 DES블럭의 키 값으로 사용되어진다.

3.2 SEED 설계

그림 4는 SEED알고리듬의 내부 블럭의 구성을 나타낸다.

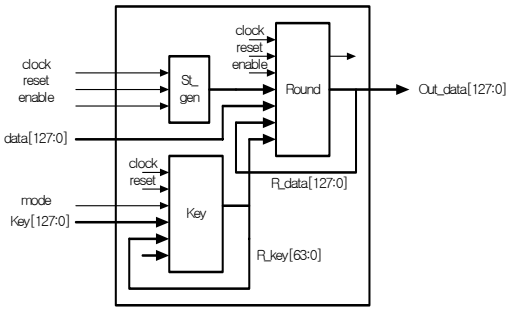

그림 4. SEED의 내부 블록도

SEED의 내부 블럭도는 St_gen, Key, Round 블럭으로 구성되어져 있다. St_gen 블럭은 Round블럭의 데이터 처리에 필요한 제어신호를 발생시키고, Key 블럭은 매 라운드에 필요한 R_key를 생성한다. Round 블럭은 St_gen 블럭에서 나온 제어신호와 Key 블럭에서 생성된 R_key로 SEED의 내부 16 라운드 처리를 수행한다.

3.3 AES 설계

그림 5는 AES의 내부 블록도를 나타낸다. AES의 내부 블럭도는 크게 3개의 블럭으로 구성되어 있다.

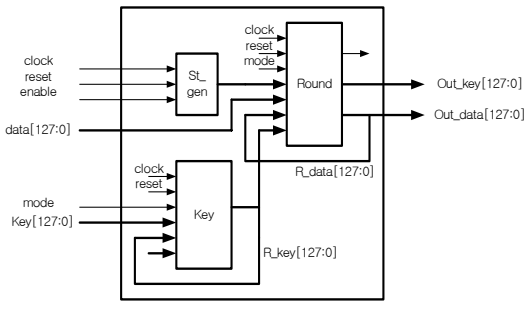

그림 5. AES의 내부 블럭도

St_gen 블럭은 Round블럭의 데이터 처리에 필요한 제어신호를 발생시켜주고, Key 블럭은 매 라운드에 필요한 R_key를 생성한다. Round 블럭은 St_gen 블럭에서 나온 제어신호와 Key 블럭에서 생성된 R_key로 A

ES의 내부 10 라운드 처리를 수행한다.

3.4 HMAC-SHA-1 설계

그림 6은 HMAC-SHA-1 알고리듬의 내부 블럭도를 나타낸다.

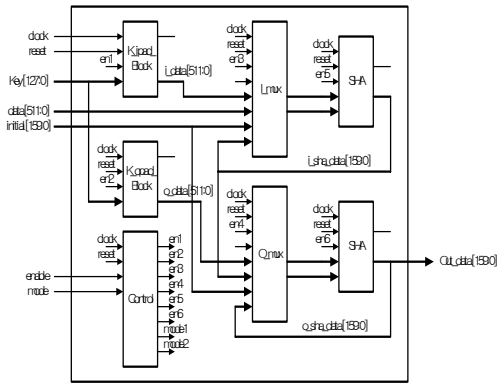

그림 6. HMAC-SHA-1 의 내부 블럭도

Control블럭은 다른 5개의 블럭에 대한 제어신호를 발생시킨다. 제어신호를 받은 K_ipad_block, K_opad_block 은 key 벡터를 입력받아 ipad(0x36), opad(0x5c)에 의해 512bit의 벡터를 생성한다. 이 벡터들은 I_sha와 O_sha의 입력 벡터로 사용되어 새로운 해쉬 벡터를 얻는다. 제어신호를 받은 I_mux와 O_mux는 제어신호에 따라 K_ipad_block, K_opad_block에서 나온 데이터를 출력으로 하거나 입력된 데이터를 축력으로 하는 동작을 수행한다. 마지막으로 제어신호를 받은 두 개의 SHA 블럭은 160bit의 해쉬벡터를 생성하여 HMAC-SHA-1의 동작을 수행하게 된다.

Ⅳ. 시스템 구현 및 성능평가

본 논문에서 제안된 암호 및 인증엔진의 각 모듈은 Xilinx ISE 6.2i 툴을 이용하여 VHDL 설계 및 합성을 수행하였다. 또한, 설계 검증을 위한 타이밍 시뮬레이션을 Modelsim을 이용하였고, Xilinx FPGA VertexII(XC8000)를 타겟으로 FPGA를 구현하였다. 표 1은 구현된 암호 프로세서에 대한 성능평가를 나타낸다.

표 1 암호프로세서의 성능 평가

	# gates (Silces)	Throughput (Frequency)
DES	17199 (637)	320Mbps (120Mhz)
3DES	53649 (1987)	106Mbps (120Mhz)
AES	111024 (4112)	546Mbps (59Mhz)
SEED	150336 (5568)	218Mbps (34Mhz)
HMAC-SHA-1	132570 (4910)	233Mbps (43Mhz)

표 1에서 알 수 있듯이 슬라이스의 총수는 약 17214개였으며, 총 게이트 수는 약 464,778 정도였다. 처리속도에 있어서 DES와 3DES는 120Mhz로 동작하며, 처리율은 각각 320Mbps와 106Mbps 정도였다. AES와 SEED 경우 처리속도는 각각 49Mhz 와 34Mhz로 동작하며, 처리율은 각각 546Mbps과 218Mbps 정도를 보였다. 인증모듈의 경우 SHA-1 은 43MHz에서 동작하고, 지연시간은 15 ns으로 약 537Mbps의 처리율을 보였다. HMAC과 연동하여 HMAC-SHA-1으로 동작할 경우 43 Mhz에서 233Mbps의 처리율을 보였다.

Ⅴ. 결론

본 논문에서는 VPN을 위한 IPSec 암호 프로세서의 FPGA 구현에 관하여 기술하였다. 설계된 암호프로세서는 IPSec의 가장 많은 시간을 소요하는 내부 데이터의 암호화 및 인증을 전담하는 코어 부분을 H/W로 구현함으로써 전체 IPSec 처리의 속도 향상이 가능하도록 구현하였다.

제안된 IPSec 암호 및 인증 엔진은 Verilog을 사용하여 구조적 모델링을 행하였으며, Xilinx사의 ISE 6.2i 툴과 Modelsim을 이용하여 시뮬레이션 및 합성을 수행하였다.

참고문헌

[1] 인터넷보안기술포럼 “Implementation Technology for secure VPN in IP Layers”, 2001.

[2] S. Kent, R, Atkinson "Security Architecture for the Internet Protocol". Internet RFC November 1998.

[3] NBS, Data Encryption Standard, FIPS Pub. 46, U.S, National Bureau of Standards, Jan. 1977.

[4] 한국정보보호센터, "128비트 블럭 암호알고리듬(SEED) 개발 및 분석보고서", KISA, 2003.

[5] Joan Daemen, Vincent Rijmen, 『AES Proposal : Rijndael』,(http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf)

[6] C. Madson, R. Glenn, "The use of HMAC-SHA1 within ESP and AH", RFC 2040, November 1998.

[7] 강민섭, 남승용, 김주한, “IPSec 암호 프로세서를 위한 SHA-1 해쉬 엔진의 하드웨어 구현”, 대한전자공학회, 추계학술발표논문집, 2003.

[8] 이광호, 강민섭, 류대현, “병렬처리 기법을 이용한 AES 암호 알고리듬의 FPGA 구현”, 대한전자공학회, 추계학술발표논문집, 2003.

[9] 이광호, 남승용, 강민섭, “개선된 LUT 방식을 이용한 SEED 알고리듬의 FPGA 구현”, 대한전자공학회, SOC설계발표논문집, 2004.