임베디드 보드를 이용한 암호화 성능측정

정윤호° 최태영
레드게이트 연구소 응용기술팀
금오공과 대학교 컴퓨터 공학부
jyh07@redgate.co.kr, choety@kumoh.ac.kr

Encryption Efficiency Analysis using Embedded Board

Yunho Jung°, Taeyoung choe
Technical Engineering, RedGate R&D Center
Dept of Computer Engineering, Kum-Oh Institute of Technology University

요 약

나날이 발전하는 임베디드 시스템의 기술은 임베디드 시스템 간의 정보통신 시 보다 안전하고 효율적인 데이터 통신을 필요로 하고 있다. 이를 위해 최근 임베디드 간의 통신 시 암호화 된 데이터 통신으로 보다 안전한 통신보안 시스템을 구축하고 있다. 하지만 임베디드 상에서 충분히 안전한 보안 시스템을 구축하기에는 다소 부하가 큰 것으로 알려져 있다. 따라서 임베디드 시스템에 알맞은 암호화 시스템을 구축하는 것이 필요하다. 본 논문에서는 PXA255 기반의 임베디드 보드에서 RSA/SHA1 암호화 시스템과 DH/RC4 암호화 시스템을 구현하고 그 성능을 평가하였다.

1. 서 론

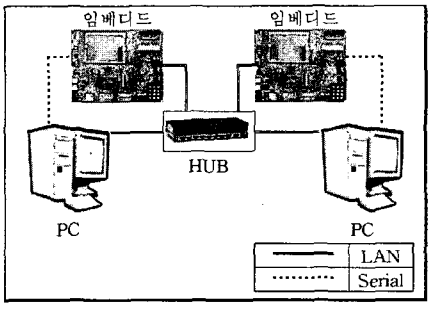
임베디드 시스템이란 마이크로프로세서 혹은 마이크로 컨트롤러를 내장하여 원래 제작자가 의도했던 특정 기능만을 수행하도록 제작된 장치를 말한다. 임베디드 시스템의 개발 및 발전은 임베디드 시스템 간의 정보 처리와 정보 통신의 다양화를 가져왔고 기존에 문제가 되지 않았던 정보 보호가 중요한 문제로 부각되고 있다. 이에 최근 정보 시스템 내에서의 정보 보호와 통신 상태의 정보 보호 및 사용자 합법성 확인을 위한 방법으로 암호가 크게 주목을 받고 있다. 암호란 평문을 해독 불가능한 형태로 변형하거나 또는 암호화된 통신문을 해독 가능한 형태로 변환하기 위한 원리, 수단, 방법 등을 취급하는 기술 또는 과학을 말한다. [1].

임베디드 시스템에서의 암호화 된 데이터 전송관련 논문으로 인터넷 폰 보안 시스템의 설계 [3]와 보안 팩스 서버 개발 [4]이 있었다. 인터넷 폰 보안 시스템의 경우 DES 알고리즘과 EDE 알고리즘을 이용하여 데이터 암호화와 전자서명 기술을 접목한 시스템을 구현하였다. 실험결과 DES 알고리즘을 이용할 경우 1.46초, EDE 알고리즘을 이용할 경우 3.25초의 전송지연 시간을 보였다. 또한 임베디드 보안 팩스 서버 개발에서는 SEED 알고리즘을 이용 하였으며, 실험결과 전송 페이지에 상관없이 일반 팩스 전송보다 2페이지 정도의 전송지연만이 발생하였다. 이와 같이 임베디드 시스템을 통한 데이터를 암호화하여 전송하는 방식은 일반 데이터 전송에 비하여 암•복호화에 따르는 시간에 의해 전송지연 시간이 증가한다.

본 논문은 임베디드 보드 상에서 Data를 암호화하여 통신하는 프로그램을 구현하였으며 암•복호화에 따른 전송지연 시간을 측정하였다. 암호화 알고리즘으로는 RSA 암호화 알고리즘과 SHA1 해쉬 알고리즘 그리고 Diffie-Hellman의 키 분배 알고리즘과 RC4 암호화 알고리즘을 구현하여 사용하였다. 본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 구현한 암호화 시스템의 구성환경과 암호화 알고리즘 구조를 보여준다. 3장에서는 시스템의 환경과 구현결과의 내용을 설명하며, 4장에서는 구현된 시스템의 성능을 평가하고 마지막으로 5장에서는 결론을 맺는다.

2. 암호화 시스템 설계

임베디드(LETOK-X255)와 PC는 Serial Cable로 연결 하였으며 Linix 환경에서의 minicom 터미널 에뮬레이터에 의해 임베디드를 제어하였고, 호스트 PC의 크로스 컴파일러를 통해 제작된 프로그램은 NFS(Network File System)를 이용하여 임베디드의 CPU에서 수행시키게 하였다. 통신은 BSD 소켓을 이용하였다. (그림 1)은 시스템의 전체 구성환경을 나타낸 것이다.

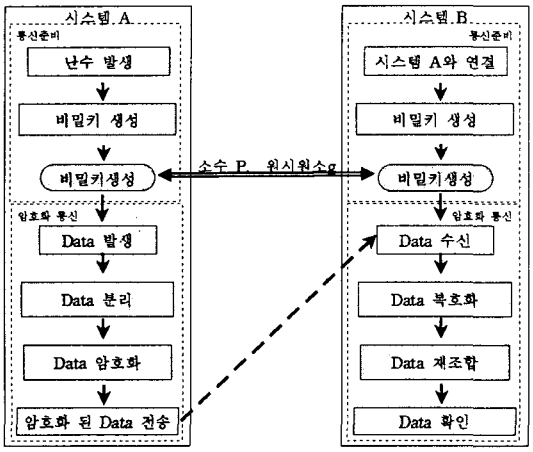


(그림 1) 시스템 전체 구성도

2.1 DH/RC4 암호화 알고리즘 적용과정

대칭키 암호화인 RC4 암호화 알고리즘은 Data의 암•복호화를 위해 공통의 비밀키가 필요하게 된다. 이를 위해

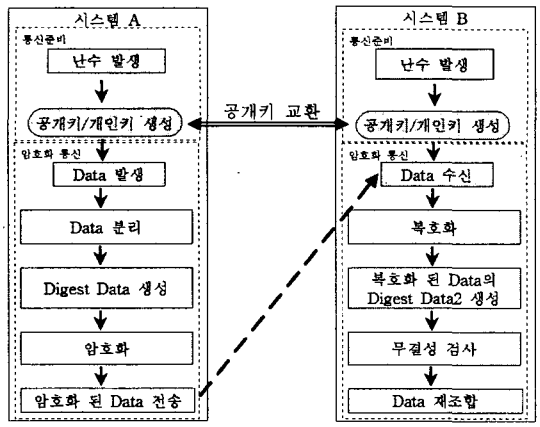
Diffie-Hellman의 키 분배 알고리즘을 이용하여 비밀키를 분배하는 방법을 이용하였다. 1024bit 이상의 수를 생성하기 위하여 GMP 라이브러리를 사용하였으며 [5], Diffie-Hellman의 알고리즘을 이용하여 비밀키를 생성하였다. 이 과정을 (그림 2)에서 도식화 하였다.



(그림 2) DH/RC4 암호화 시스템

2.2 RSA/SHA1 암호화 알고리즘 적용과정

RSA 암호화 시스템에 사용되는 개인키와 공개키는 512bit 이상의 정수로 GMP 라이브러리를 사용하여 큰 수의 소수를 생성한 후, RSA 알고리즘을[2] 이용하여 개인키와 공개키를 생성하였다. RSA 암호화 시 SHA1 알고리즘을 이용하여 160bit의 Digest Data를 생성한 후, Data와 Digest Data을 함께 RSA 암호화 알고리즘으로 암호화하여 통신함으로써 Data의 무결함을 확인하는 방식을 사용하였다. 이 과정을 (그림 3)에서 도식화 하였다.



(그림 3) RSA/SHA1 암호화 시스템

3. 시스템 구현

3.1 임베디드 보드 환경

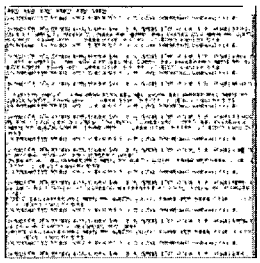
암호화 시스템에 사용되는 임베디드 보드는 LETOK-X255 보드로서 Intel XScale PXA255 Processor가 지원하는 대부분의 주변 기기 셋을 탑재하고 있다. 또한 LETOK-X255 보드에는 4개의 SDRAM 칩을 사용한다. <표 1>은 자세한 LETOK-X255 보드의 하드웨어 사양을 표시하였다.

<표 1> 임베디드 보드 사양

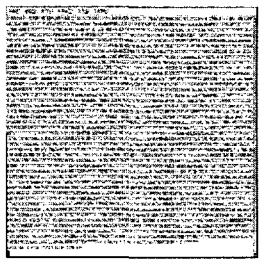
항목	세부항목	비고
CPU	Intel XScale PXA255(400Mhz)	
SDRAM	128MByte, 32Bits Access	
Flash Memory	32MByte, 32Bits Access(Intel)	
Ethernet	10/100Mbps Ethernet Interface 2Port	LAN91C111
Text LCD	2line * 16 Text LCD, Back Light	

3.2 시스템 구현결과

(그림 4)과 (그림 5)는 두 개의 암호화 시스템에서 Data를 암호화 한 Packet의 일부분을 뽑아서 화면에 출력시킨 결과이다. DH/RC4 암호화 시스템은 알 수 없는 문자들로 암호화되어 출력되었고, RSA/ SHA1은 숫자형태로 암호화되어 출력되었다. 이는 DH/RC4 암호화 시스템은 Data를 문자열 형식으로 변환한 후 이 문자열 정보를 RC4로 암호화하여 Packet으로 만들었고, RSA/SHA1은 Data를 바이트 형식으로 변환한 후 이 바이트 정보를 RSA로 암호화하여 Packet으로 만들었기 때문이다. 또한 RSA 암호화 방식은 최대 난수크기의 2배 정도의 Data를 암호화 하였다.



(그림 4) RC4 암호화 Data



(그림 5) RSA 암호화 Data

4 성능 분석

성능분석은 PC와 임베디드 상에서 암•복호화에 따르는 시간지연을 측정하여 그 결과 값을 비교하였다. 비교에 사용된 PC의 사양은 Pentium(R) 4 1.7Ghz의 CPU와 256MB의 SDRAM을 장착하였고 커널 2.4의 Linux 환경에서 측정하였다. 측정에 사용된 Data는 각 10KB, 20KB, 30KB 크기의 텍스트 파일로 이 파일들을 암•복호화를 하면서 지연되는 소요시간을 측정하였다.

<표 2> Linux PC 간 암호화 통신

단위 : 초

알고리즘	Key (bit)	Key 생성시간 (평균/표준편차)	Size (KB)	암호화시간 (평균/표준편차)	복호화시간 (평균/표준편차)
RSA/SHA1	512	1.70(±1.18)	10	0.03(±0.01)	4.95(±0.02)
			20	0.07(±0.01)	9.88(±0.04)
			30	0.09(±0.01)	15.02(±0.04)
	1024	14.00(±5.72)	10	0.05(±0.01)	17.74(±0.02)
			20	0.10(±0.01)	35.41(±0.03)
			30	0.15(±0.01)	53.02(±0.04)
	2048	194.10(±171.44)	10	0.10(±0.01)	67.23(±0.08)
			20	0.20(±0.01)	131.50(±0.40)
			30	N/A	N/A
DH/RC4	1024	1.81(±1.31)	10	0.005(±0.005)	0.005(±0.005)
			20	0.01(±0.01)	0.01(±0.005)
			30	0.01(±0.01)	0.01(±0.005)
	2048	21.01(±15.21)	10	0.005(±0.005)	0.005(±0.005)
			20	0.01(±0.01)	0.01(±0.005)
			30	0.01(±0.01)	0.01(±0.005)
	3072	66.22(±36.13)	10	0.005(±0.005)	0.005(±0.005)
			20	0.01(±0.01)	0.01(±0.005)
			30	0.01(±0.01)	0.01(±0.005)

<표 2>는 PC에서 측정한 암호화 시스템의 성능을 나타낸 것이다. RSA/SHA1 암호화 시스템은 키 크기에 따라 암•복호화의 소요시간이 길어지는 것으로 나타났다. 또한 키 크기가 같은 경우에는 파일의 크기에 비례하여 시간이 일정히 증가하는 것으로 나타났으며 키 생성시간 역시 키 크기가 증가하면 소요시간이 증가하는 것으로 나타났다. 2048bit 키의 30KB의 파일의 암•복호화의 소요시간은 성능이 현격히 느려 측정을 하지 않았다. DH/RC4 암호화 시스템의 경우에는 키 크기에 상관없이 암•복호화에 소요되는 시간이 일정히 빠르게 나타났다.

<표 3> 임베디드 간 암호화 통신

단위 : 초

알고리즘	Key (bit)	Key 생성시간 (평균/표준편차)	Size (KB)	암호화시간 (평균/표준편차)	복호화시간 (평균/표준편차)
RSA/SHA1	512	3.44(±2.60)	10	0.11(±0.01)	11.81(±0.01)
			20	0.22(±0.01)	23.49(±0.03)
			30	0.33(±0.01)	35.35(±0.03)
	1024	35.60(±20.16)	10	0.14(±0.01)	40.22(±0.05)
			20	0.28(±0.01)	80.42(±0.05)
			30	0.43(±0.01)	120.85(±0.20)
	2048	325.69(±135.51)	10	0.21(±0.01)	144.94(±0.50)
			20	0.42(±0.01)	283.17(±0.50)
			30	N/A	N/A
DH/RC4	1024	5.86(±3.94)	10	0.02(±0.01)	0.02(±0.01)
			20	0.05(±0.01)	0.04(±0.01)
			30	0.07(±0.01)	0.06(±0.01)
	2048	71.53(±55.19)	10	0.02(±0.01)	0.02(±0.01)
			20	0.05(±0.01)	0.04(±0.01)
			30	0.07(±0.01)	0.06(±0.01)
	3072	260.08(±173.79)	10	0.02(±0.01)	0.02(±0.01)
			20	0.05(±0.01)	0.04(±0.01)
			30	0.07(±0.01)	0.06(±0.01)

<표 3>은 임베디드에서 측정한 암호화 시스템의 성능을 나타낸 것이다. RSA/SHA1 암호화 시스템의 경우 PC에서의 측정과 같이 키 크기가 커질수록 암•복호화에 소요되는 시간이 길어지고, 키 크기가 같은 경우에는 파일의 크기에 비례하여 소요시간이 일정히 증가되는 것으로 나타났다. 키 생성시간 역시 키 크기가 증가할수록 소요시간이 증가하였고 키 생성 시간의 표준편차가 크게 나타났다. <표 2>에서와 같이 2048bit 키의 30KB 파일의 암•복호화의 성능은 현격히 느려 측정을 하지 않았다. DH/RC4 암호화 시스템의 경우에서는 암•복호화의 지연시간이 PC에서의 측정시간보다 파일의 크기가 커질수록 조금씩 증가하는 것으로 나타났고, 키 크기에는 영향을 받지 않는 것으로 나타났다.

<표 4> RSA/SHA1 100byte 고정블록 암호화

단위 : 초

알고리즘	Key (bit)	Size (KB)	암호화시간 (평균/표준편차)	복호화시간 (평균/표준편차)
RSA/SHA1	512	10	0.12(±0.01)	14.04(±0.01)
		20	0.25(±0.01)	28.09(±0.01)
		30	0.37(±0.01)	42.27(±0.01)
	1024	10	0.27(±0.01)	95.84(±0.01)
		20	0.55(±0.01)	191.70(±0.02)
		30	0.83(±0.01)	288.51(±0.05)
	2048	10	0.53(±0.01)	656.54(±0.10)
		20	1.04(±0.01)	1313.08(±0.20)
		30	N/A	N/A

<표 4>는 임베디드 간의 RSA/SHA1 암호화 시스템에서 100byte 고정블록으로 Data를 암호화하여 측정한 결과이다. 이에 비해 <표 3>의 최대 난수생성 시 키 값의 2배의 Data를 암호화 한 시스템은 512bit에서는 약 16%, 1024bit에서는 약 58%, 2048bit에서는 약 78%의 속도향상을 보였다.

5. 결 론

본 논문은 PXA 255기반의 임베디드 보드 간의 파일 및 Data를 암호화하여 교환하는 시스템을 설계 및 구현하였다. 측정에 사용된 임베디드는 400Mhz의 CPU 클락을 가지고 있고, PC의 CPU은 1.7Ghz로 이는 임베디드 보다 대략 4배 정도 빠른 CPU 클락을 가지고 있다. 위 두 가지 환경에서의 측정을 비교한 결과 RSA/SHA1 암호화 시스템은 PC의 환경에서 보다 임베디드의 환경에서 대략 2.5배 정도 느린 암•복호화 성능을 나타냈다. 또한, 고정블록의 Data를 암호화하는 것보다 키 크기의 2배정도로 Data를 암호화하는 것이 시스템의 속도향상에 크게 영향을 미쳤다. DH/RC4 암호화 시스템은 임베디드 상에서의 측정 시 파일의 크기가 커질수록 암•복호화의 소요시간이 점차로 증가하는 것을 보였다. 하지만 암•복호화의 소요시간이 PC와 임베디드 환경 모두 빠르게 나타났으며, 이는 임베디드 환경에서 사용할 만한 성능으로 사료된다.

참고문헌

[1] 은성배/진성기/장덕순 공저 "임베디드 시스템의 이해", (주)옥타컴, 2005

[2] Niels Ferguson, Bruce Schneier "Practical Cryptography", 사이텍미디어, 2004

[3] 박재희, 김일민 "인터넷 폰 보안 시스템의 설계 및 구현", 정보처리학회논문지, 제9권 제2호, pp.157-162, 2002.

[4] 이상학, 정태충 "이중 방법을 지원하는 임베디드 보안 팩스 서버 개발", 정보처리학회논문지, 제11권 제3호, pp.129-137, 2004.

[5] http://www.swox.com/gmp/