논문 2013-08-37

모바일 클라우드 스토리지 서비스에서의 데이터 보안을 위한 데이터 접근 제어 및 보안 키 관리 기법

(A Method for Data Access Control and Key Management in Mobile Cloud Storage Services)

신 재 복, 김 윤 구, 박 우 람, 박 찬 익*
(Jaebok Shin, Yungu Kim, Wooram Park, Chanik Park)

Abstract: Cloud storage services are used for efficient sharing or synchronizing of user's data across multiple mobile devices. Although cloud storages provide flexibility and scalability in storing data, security issues should be handled. Currently, typical cloud storage services offer data encryption for security purpose but we think such method is not secure enough because managing encryption keys by software and identifying users by simple ID and password are main defectives of current cloud storage services.

We propose a secure data access method to cloud storage in mobile environment. Our framework supports hardware-based key management, attestation on the client software integrity, and secure key sharing across the multiple devices. We implemented our prototype using ARM TrustZone and TPM Emulator which is running on secure world of the TrustZone environment.

Keywords : Mobile Security, ARM TrustZone, TPM, Cloud Storage

I. 서 론

스마트 폰, 태블릿 PC와 같은 모바일 기기가 대중화 되면서 개인이 여러대의 모바일 기기를 소지하는 일이 흔해짐에 따라 효율적인 기기들간의 데이터 공유 또는 동기화를 위하여 클라우드 스토리지 서비스를 많이 사용한다. 하지만 사용자의 데이터가 원격지에 저장되는 클라우드 스토리지의 특성으로 인하여 사용자 데이터의 유출 및 변경 등의 보안 위협이 발생 할 수 있다. 따라서 보안성을 중요시하는 개인 사용자나 기업 환경에서 클라우드 스토리지 서비스의 확산이 저하되는 원인이 된다.

사용자 데이터 유출은 클라우드 스토리지 서비스의 서버 측 또는 클라이언트 기기에서 발생할 수 있다. 서버 측에서는 클라우드 서비스를 구성하는 가상화 환경의 취약점 [1]을 이용한 악의적인 공격이나 [2, 3] 서비스 관리자와 같은 내부인에 의해서 사용자 데이터가 유출된다. 클라이언트에서는 키로거로 인한 사용자 인증정보 유출 그리고 악성 코드 등에 의해서 데이터가 유출된다.

따라서 현재 많이 사용되는 클라우드 스토리지 서비스들은 사용자 데이터 보안을 위하여 암호화 기법을 사용하고 있다. 대표적인 예로 Amazon S3 [4]는 사용자의 선택에 따라 서버 기반 데이터 암호화 또는 클라이언트 기반 데이터 암호화를 지원하고 Dropbox [5]는 서버 기반 데이터 암호화 방식을 채택하였다. 이러한 방식을 사용하여 데이터 유출의 위협을 줄일 수 있지만 여전히 유출 위협이 존재하며 그 이유는 다음과 같다.

첫 번째는 암호화에 사용되는 데이터 키는 소프트웨어적으로 관리된다는 점이다. 따라서 악성 코드, 악의적인 관리자, 또는 크래킹에 의한 데이터 키의 유출로 암호화 된 데이터를 복호화 할 수 있다. 두 번째로 서버 기반 암호화의 경우 실제 암호화가 일어나기 전에는 데이터가 플레인텍스트로 존

* Corresponding Author (cipark@postech.ac.kr)
Received: 15. Feb. 2013, Revised: 18 Mar. 2013, Accepted: 5 Sep. 2013.
J. Shin, Y. Kim, W. Park, C. Park : POSTEC
※ "본 연구는 미래창조과학부 및 정보통신산업진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음" (NIPA-2013-H0301-13-3002)

ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2013.8.6.303

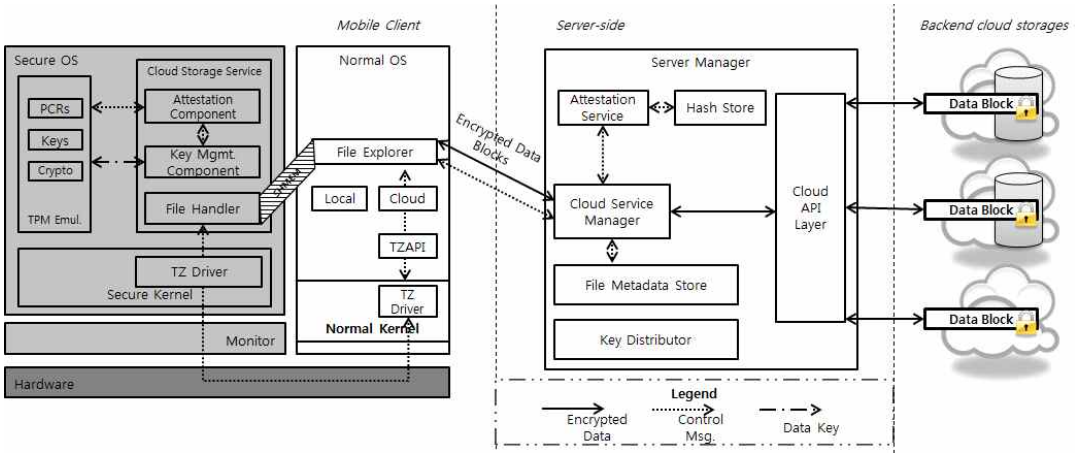

그림 1. 시스템 구조 (빗금친 부분은 공유메모리를 나타낸다.)
Fig. 1 System Architecture

재한다는 점이다. 따라서 서버의 취약점을 이용한 공격에 의해 데이터가 유출될 수 있다. 세 번째는 사용자의 인증 과정이 간단한 ID와 패스워드 만으로 진행이 되기 때문에 키로거에 의한 인증정보 유출이 여전히 가능하다는 점이다.

따라서 본 논문에서는 모바일 클라우드 스토리지 서비스에서 데이터 유출을 최소화하는 안전한 데이터 관리 기법을 제안한다. 본 기법은 하드웨어 기반의 키 관리, 클라이언트 소프트웨어의 무결성 검증 그리고 여러 기기간의 안전한 키 공유 방법을 제공한다. 이를 위하여 ARM TrustZone [6] 기술과 TrustZone 환경의 Secure World에서 동작하는 TPM 에뮬레이터 [7, 8]를 이용해서 프로토타입을 구현하였다 [9].

Ⅱ. 위협 모델

본 논문에서 제안하는 프레임워크를 적용한 모든 모바일 클라이언트들은 ARM TrustZone 기술이 적용되어 있다고 가정한다. ARM TrustZone 기술은 ARM 어플리케이션 프로세서 (AP)에 하드웨어적으로 구현이 되어 있으며 Normal World, Secure World 두 개의 운영체제들이 독립적으로 동작하도록 지원한다. Secure World는 하드웨어적으로 보호되는 안전한 영역이다. 따라서 보안이 요구되는 작업을 Secure World에서 동작시켜 사용자 정보 보호, 암호화 키 관리 등을 안전하게 할 수 있다. 본 프레임워크는 ARM TrustZone 기술을 이용하여 하드웨어 기반의 키 관리 방법을 제공한다.

모바일 클라이언트의 Normal World에는 악성 프로그램이 상주할 수 있으므로 Normal World는 신뢰할 수 없는 환경을 가정한다. 또한 클라우드 스토리지 서비스 제공자도 앞서 서론에서 언급한 데이터 유출 위협으로 인하여 신뢰 할 수 없다. 본 프레임워크는 클라우드 스토리지 서비스에서 일어날 수 있는 서버 측 그리고 클라이언트 측 데이터 유출에 초점을 두었다.

Ⅲ. 시스템 구조

1. 시스템 개괄

본 프레임워크 (그림 1)는 ARM TrustZone 기술이 적용된 모바일 클라이언트, 중앙 서버, 그리고 클라우드 스토리지 제공자의 구성요소를 가진다. 사용자는 클라이언트의 Normal World에서 동작하는 파일 탐색기 응용프로그램을 통하여 사용자 인증, 로컬 및 클라우드 파일 탐색, 그리고 업로드와 다운로드, 삭제, 이동 등 파일 오퍼레이션을 수행한다. 한편으로 모바일 클라이언트 상에서 데이터 키 관리, 플랫폼 무결성 검증, 파일 암호화 및 복호화 기능을 제공하는 보안 프로세스들은 모두 Secure World에서 동작한다. 사용자가 파일 탐색기를 통하여 클라우드 파일에 접근하면 TrustZone API(TZAPI)가 호출되고 TrustZone 모니터에 의해 두 World가 전환되면서 해당하는 요청 메시지가 전달된다. Secure World의 클라우드 스토리지 서비스 프로세스가 전달받은 메

시지에 대한 처리를 하며 키 관리 기능들은 Secure World의 TPM Emulator를 이용하여 처리한다.

중앙 서버는 클라이언트의 플랫폼 무결성 검증을 수행하고 암호화된 사용자 데이터를 클라이언트로부터 전송받아 클라우드 스토리지 API를 사용하여 클라우드 스토리지에 저장하며 유저 ID, 파일명, 그리고 실제 저장된 클라우드 스토리지 위치로 이루어진 파일 메타데이터를 기록한다. 클라이언트에서 다운로드가 수행될 때 메타데이터를 검색하여 파일이 실제로 저장된 클라우드 스토리지 위치를 클라이언트로 전송함으로써 클라이언트로부터 해당 클라우드 스토리지에 직접 접근이 가능하도록 한다. 또한 Key Distributor를 통하여 서로 다른 사용자 또는 다른 기기들 간에 데이터 암호화 키 교환 프로토콜을 지원한다. 클라우드 API 계층은 여러개의 다른 상용 클라우드 스토리지를 선택적으로 사용할 수 있게 한다. 따라서 사용자 데이터는 클라우드 스토리지 선택 정책에 따라 임의적인 위치에 저장될 수 있다.

2. 사용자 인증 및 플랫폼 검증

본 프레임워크에서는 Core Root of Trust for Measurement (CRTM)을 Secure World로 가정한다. 모바일 클라이언트의 부팅과정에서 Secure World의 Attestation Component가 Normal 부트로더 및 OS 이미지의 무결성을 검증하고 검증 결과 값을 TPM Emulator의 Platform Configuration Register (PCR)에 저장한다. Normal OS의 부팅 이후에 Integrity Measurement Architecture (IMA)에 의해서 라이브러리, 실행된 프로세스 및 시스템 설정 파일들의 무결성을 측정하고 TZAPI를 통하여 PCR에 extend 한다.

사용자는 클라우드 파일 탐색기를 실행하여 자신의 인증정보 (ID/password)로 로그인하게 되고 서버와 클라이언트간에 nonce 값 을 교환함으로써 세션이 시작된다 (그림 2 위). 원격 검증 프로토콜에 의하여 TPM Emulator에서 quote 명령으로 PCR에 저장된 무결성 검증 정보와 Measurement Log (ML) 값, 그리고 N 값을 서버에 전송함으로써 플랫폼 검증을 수행한다. 서버의 Attestation Service는 Hash Store에 저장된 값과 전송 받은 무결성 정보를 비교한 다음 현재 모바일 클라이언트가 안전한 상태에 있는지 검사하고 결과를 클라이언트로 전송한다. 인증이 성공하면 서버가 보내는 원격 검증 결과 메시지에는 사용자에게 부여된 인증 값 H 가 같이 전송되고 이는 클라이언트 TPM Emulator의 PCR에 저장한다. 이는 추후 사

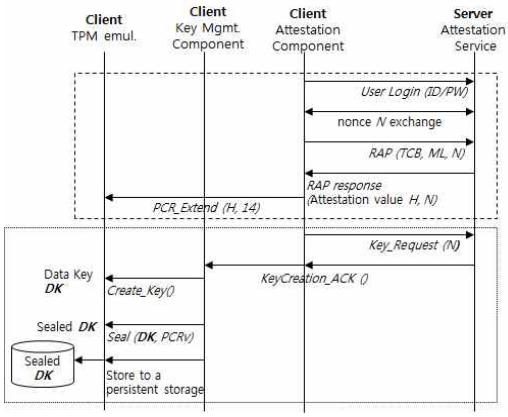

그림 2. 사용자 및 플랫폼 인증과정 (위)과 데이터 키 생성 및 관리 프로토콜 (아래)

Fig. 2 User&platform attestation protocol and data key creation&Management protocol

용자 암호화 키를 모바일 기기 내 저장소에 안전하게 보관하기 위한 프로토콜에서 사용된다. 인증이 실패하면 사용자는 자신의 클라우드 스토리지에 접근 할 수 없다.

3. 데이터 암호화 키 생성 및 관리

사용자 인증 및 플랫폼 검증이 성공적으로 끝나면 데이터 암호화에 사용할 암호화 키를 TPM에 불러오는 과정이 시작 된다. 먼저 서버에 사용자의 키 생성 기록을 조회한 후 (Key_Request), 서버의 응답에 따라 키 생성 (Key_Creation_ACK), 키 로드 (Key_Loading_ACK), 또는 키 공유 프로토콜 (Key_Remote_ACK)을 수행한다. 인증을 통과한 사용자가 처음 접속한 사용자이면 (그림 2 아래) 서버에 키 생성 정보가 없으므로 클라이언트에서 자신의 암호화 키를 생성하여야 한다. 따라서 Key Mgmt. Component가 TPM Emulator에게 키 생성 요청을 하고 TPM Emulator에 의해서 사용자의 클라우드 스토리지 데이터 암호화에 사용할 데이터 키 DK를 생성한다. 클라이언트는 키가 생성 되면 서버에 키 정보와 모바일 기기 정보를 등록한다.

사용자 클라우드 스토리지로의 접근이 종료되어 데이터 키가 사용되지 않을 때 TPM Emulator내 즉, Secure World의 메모리 영역에 로드되어 있는 데이터 키를 모바일 기기의 로컬 스토리지에 안전한 형태로 보관 해한다. 따라서 TPM Emulator의 Seal 명령을 통하여 sealing 된다. Sealing 과정에서 사용되는 값은 현재 기기의 무결성 정보를 담은 PCR 값과

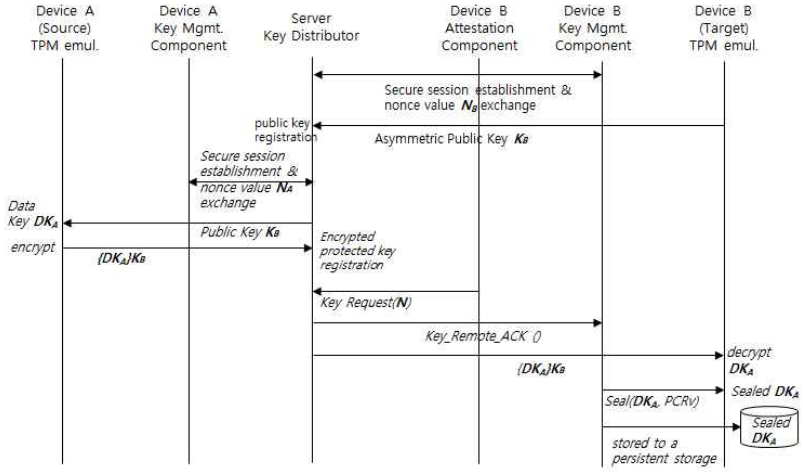

그림 3. 사용자 데이터 암호화 키 공유 프로토콜
Fig. 3 Remote key sharing protocol

사용자 및 플랫폼 인증의 결과로 서버로부터 전송받은 인증 값 가 쓰인다. Sealing된 키가 다시 사용되어야 할 때에는 (이미 키 생성 기록을 가진 사용자이고 해당 모바일 기기가 키를 가지고 있는 경우) unsealing 과정을 통하여 TPM Emulator에 불러 온다. Sealing 때와 마찬가지로 unsealing 될 때의 클라이언트 기기 무결성 정보 (PCR 값)와 서버로부터 전송 받은 인증 값 H 가 사용되며 sealing 될 때와 다를 경우 사용자는 데이터 키를 사용할 수 없다.

4. 데이터 암호화 키 공유

사용자 암호화 키가 생성이 되었지만 현재 모바일 기기에 sealing 되어 보관되지 않은 경우 원격 키 공유 프로토콜을 사용하여 키를 마이그레이션 해야 한다. 또한 클라우드 스토리지의 사용자간 또는 공유그룹간의 공유가 필요할 때에도 동일한 키 공유 프로토콜을 사용할 수 있다. 데이터 키를 가진 클라이언트는 해당 키가 다른 기기 또는 다른 사용자 (그룹)에게 공유할 필요성 (요청)이 있으면 이를 중앙 서버에 전송하여 저장을 하고 공유 대상 클라이언트에서 이를 전송 받아 공유 클라우드 스토리지의 암호화된 데이터에 접근 할 수 있다.

먼저 공유 대상 클라이언트 (target, Device B)는 서버와의 secure network session을 통하여 자신의 public 키 K 를 중앙 서버에 등록 한다 (그림 3). 공유할 키를 가진 클라이언트 (source, Device A)는 공유 요청에 따라 Device B의 K_B를 서버로부터 전송 받고 sealing 된 키를 unsealing 과정을 거쳐 TPM Emulator로 불러 들인 후 (DK_A), K_B로 암호화 한다. 암호화 된 Device A의 데이터 키는 서버에 임시 저장된다. Device B는 그림 2의 인증과정을 거친 후 서버에 Key_Request 요청을 하고 서버로부터 Key_Remote_ACK 메시지와 함께 $\{DK_A\}K_B$를 전송 받는다. TPM Eulator에서 Device B의 private key로 복호화된 DK_A는 공유된 클라우드 스토리지 데이터에 접근할 때 사용되고 클라우드 스토리지 사용이 종료 되면 sealing 과정을 거쳐 Device B의 로컬 스토리지에 안전하게 보관된다.

5. 파일 업로드 및 다운로드

사용자가 로컬 파일을 클라우드 스토리지에 업로드할 때 먼저 Secure World와 Normal World의 공유메모리에 데이터를 불러들인다. TZAPI를 통하여 파일 암호화 요청을 하면 Secure World로 전환되고 File Handler에 의하여 공유 메모리상의 데이터를 암호화 키로 암호화 한 후 완료 메시지와 함께 Normal World로 전환한다. 파일 탐색기는 공유 메모리 상의 암호화된 데이터를 서버로 전송한다. 클라우드 스토리지의 데이터를 다운로드 할 때에는 서버로부터 전송받은 데이터의 실제 위치로 직접 접근하여 공유메모리상으로 불러들인다. 암호화 과정과 비슷한 과정을 거쳐 Secure World의 File Handler에 의해 복호화 된 이후 로컬 스토리지에 저장한다.

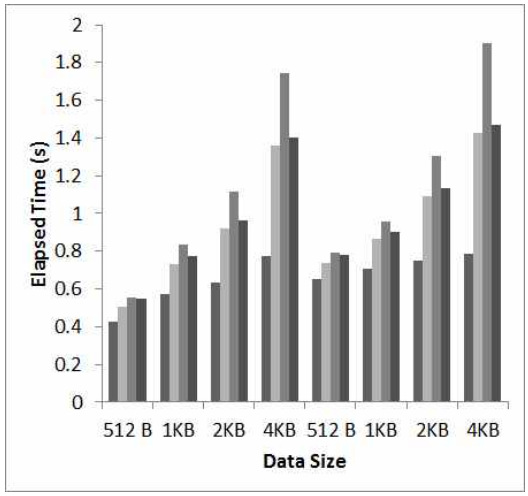

그림 4. 실험 결과
Fig. 4 Experimental Result
그래프의 x 축을 기준으로 왼쪽 4개는 업로드, 나머지는 다운로드 시의 실행시간을 나타냄.
■ : w/o Encryption, □ : Normal World Encryption, ▩ : Secure World Encryption, ■ : w/ Shared Memory

Ⅳ. 실험결과

본 프레임워크를 적용한 클라우드 스토리지 서비스의 업로드 및 다운로드 성능 측정을 위하여 다음과 같이 구현하였다. ARM Fastmodel [10]을 사용하여 에뮬레이트 된 Cortex-A15 임베디드 보드 위에 Open Virtualization [11] 에서 제공하는 TrustZone 소프트웨어 스택을 사용하였다. Normal World에서는 리눅스 (커널 2.6.38)와 파일 탐색기가 동작하고 Secure World에서는 암호화 모듈이 동작한다. 현재 암호화 알고리즘으로 RC4 알고리즘(256-bit key) [12] 을 사용한다.

실험으로 512B, 1KB, 2KB, 4KB 데이터에 대해서 암호화를 하지 않은 업로드 및 다운로드 시간 측정, Normal World에서의 암호화, Secure World에서의 암호화, 그리고 공유메모리를 사용한 암호화 네 가지에 대해서 수행하였고 결과는 그림 4와 같다. 각각 사용자에 의해서 업로드 또는 다운로드 명령을 내렸을 때로부터 완료되기까지의 시간을 측정하여 나타내었다.

공유메모리를 사용하지 않고 암호화를 수행할 경우 512 바이트의 데이터 마다 World 전환이 일어난다. 즉 TrustZone Secure Monitor를 거쳐서 전달되는 메시지의 최대 크기가 512 바이트 이다. 따라서 512 바이트 데이터 마다 World 간의 전환이 일어나 암호화 오버헤드가 그에 비례하여 커지는 것을 볼 수 있다 (512 바이트의 경우 Normal World 암호화와 Secure World 암호화의 차이는 World 간의 전환에서 발생하는 오버헤드 이고 평균 0.05 초이다.). 이는 World 전환의 메시지 전달 채널 구현상의 한계로 인하여 발생하는 오버헤드이며 이를 해결하기 위한 한 가지 방법으로 본 프로토타입에서는 두 World의 데이터 전달을 위한 공유 메모리 방식을 사용하였다. 실험 결과에서 나타나듯 Normal World에서 암호화를 수행하는 시간에 데이터 전달 오버헤드가 더해지는 Secure World 암호화와는 다르게 공유 메모리를 사용하면 한 번의 컨트롤 메시지 교환만으로 암호화를 수행할 수 있으므로 Normal World에서 암호화를 수행하는 시간과 비슷한 결과를 얻을 수 있다.

Ⅴ. 보안평가

본 프레임워크에서는 모바일 기기에서 클라우드 스토리지 서비스를 사용할 때 발생할 수 있는 사용자 데이터 유출을 막고자하는 데에 중점을 두고 설계되었다. 서버측의 데이터 유출 (서버 인프라의 보안 허점을 악용한)에 대응하고자 클라이언트측 암호화를 사용하였고, 암호화에 사용되는 키를 하드웨어적으로 보호하기 위한 방법으로 ARM TrustZone 기술을 이용하여 일반 사용자의 작업공간과 독립된 실행환경에서 TPM Emulator를 동작시켰다. 데이터 암호화/복호화를 비롯한 보안 관련 모듈을 보안 실행환경에서 동작시키고 키가 실제로 사용될 때 원격 플랫폼 무결성 검증을 거치도록 함으로써 일반 사용자 작업공간에서의 비정상적인 접근을 차단하고 악성프로그램의 설치 유무를 감지할 수 있다.

Ⅵ. 결 론

본 논문에서는 ARM TrustZone 기반 모바일 환경에서 클라우드 스토리지를 위한 안전한 데이터 관리 기법에 대해서 논의하였다. 클라우드 스토리지 서비스를 사용할 때 발생할 수 있는 데이터 유출 문제에 대응하기 위하여 우리는 클라이언트 측 데이터 암호화 방식을 사용하고 암호화에 사용되는 키를 하드웨어적으로 관리하기 위하여 ARM TrustZone 기술 및 TPM Emulator를 이용하였다. 또한 중앙 서버로부터 모바일 클라이언

트에 대한 원격 플랫폼 검증을 수행한 후 암호화 키를 사용하게 함으로써 악성 코드로 부터의 데이터 유출 위협을 줄일 수 있었다.

성능 오버헤드를 줄이고자 데이터를 암호화 하는데 공유메모리를 사용하였고 두 World 사이의 전환을 최소화하였지만 암호화 자체의 오버헤드를 줄이는 것이 필요하다. 또한 Normal World에서 복호화된 데이터의 보호 방법도 향후과제로 남아있다.

References

[1] National Vulnerability Database, http://nvd.nist.gov/
[2] CVE-2008-0923:http://eve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-0923
[3] The Blue Pill Project: http://bluepillproject.org/
[4] Amazon S3, “Using Data Encryption” http://docs.amazonwebserrvices.com/AamzonS3/latest/dev/UsingEncryption.html
[5] Dropbox. http://www.dropbox.com
[6] ARM, “ARM Securtiy Technology, Building a Secure System using TrustZone Technology,” 2009.
[7] S. Kinney, “Trusted Platform Module Basics-Using TPM in Embedded Systems,” Elsvier, Inc. Oxford, 2006.
[8] M. Strasser, H. Stamer, “A Software-Based Trusted Platform Module Emulator,” TRUST 2008, LNCS, Vol. 4968, pp.33 - 47, 2008.
[9] J. Shin, Y. Kim, W. Park, C. Park, “A Secure Data Management Framwork based on ARM TrustZone for Cloud Storage Services”, Proceeding of Autumn Conference on IEMEK (in Korean).
[10] ARM, “ARM Fast Model Reference Manual”, http://infocenter.arm.com/help/topic/com.arm.doc.dui0423m/DUI0423M_fast_model_rm.pdf
[11] Sierraware, “Open Virtualization for TrustZone Overview,” 2011.
[12] A. Mousa, A. Hamad, “Evaluation of the RC4 Algorithm for Data Encryption,” International Journal of Computer Science & Applications, Vol. 3, No. 2, 2006.

저 자 소 개

신 재 복

2010년 포항공과대학교 컴퓨터공학과 학사.
2010년 ~ 현재, 포항공과대학교 컴퓨터공학과 통합과정.

관심분야: 임베디드 시스템, 시스템 보안.
Email: zstormx@postech.ac.kr

김 윤 구

2011년 포항공과대학교 컴퓨터공학과 학사.
2013년 포항공과대학교 컴퓨터공학과 석사.

관심분야: 임베디드 시스템, 시스템 보안.
Email: pi3wi2@postech.ac.kr

박 우 람

2006년 고려대학교 컴퓨터공학과 학사.
2006년 ~ 현재, 포항공과대학교 컴퓨터공학과 통합과정.

관심분야: 임베디드 시스템, 시스템 보안.
Email: wizrampa@postech.ac.kr

박 찬 익

1983년 서울대학교 전자공학과 학사.
1985년 한국과학기술원(KAIST) 전기 및 전자공학과 석사
1988년 한국과학기술원(KAIST) 전기 및 전자공학과 박사
현재, 포항공과대학교 컴퓨터공학과 교수
관심분야: 임베디드 시스템, 시스템 보안, 시스템 가상화, 스토리지 시스템.
Email: cipark@postech.ac.kr