DOI: 10.3745/KIPSTC.2011.18C.3.151

안드로이드 플랫폼에서 유연한 응용프로그램 권한관리 기법 설계 및 구현

김 익 환† · 김 태 현††

요 약

대표적인 스마트폰 플랫폼의 하나인 구글 안드로이드는 응용프로그램 권한기반 보안모델을 채택하고 있다. 이는 응용프로그램의 시스템 자원에 대한 부적절한 접근을 제한하여 보안위협을 줄이기 위한 방법이지만 권한의 선택적 허용이 불가능한 점, 한번 권한이 부여된 경우 이를 되돌릴 수 없는 점, 사용자 ID 공유에 따른 권한의 공유를 사용자가 알 수 없는 점 등 문제점이 존재한다. 본 연구에서는 기존 안드로이드 보안 모델의 한계점을 개선하기 위해 사용자가 응용프로그램이 요구하는 권한을 유연하게 설정할 수 있는 기법을 설계, 구현하였다. 본 연구에서 제안한 기법의 설계목표는 기존 보안모델의 수정을 최소화하면서 보안성과 사용자 편의성을 향상하는 것이며, 구현된 기법의 동작 검증은 안드로이드 에뮬레이터 상에서 실제 응용프로그램 수행을 통해 이루어졌다.

키워드 : 안드로이드, 보안 모델, 유연한 권한 관리, 권한 상승

Design and Implementation of a Flexible Application Permission Management Scheme on Android Platform

Ikhwan Kim† · Taehyoun Kim††

ABSTRACT

Google Android, which is one of the popular smart phone platforms, employs a security model based on application permissions. This model intends to reduce security threats by protecting inappropriate accesses to system resources from applications, but this model has a few problems. First, permission requested by an application cannot be granted selectively. Second, once the permission has been granted it is maintained until the application is uninstalled. Third, applications may acquire powerful permissions through user ID sharing without any notice to users. In order to overcome these limitations, we designed and implemented a flexible application permission management scheme. The goal of our scheme is to enhance security and user convenience while keeping compatibility to original platform. We also verified the operation of our scheme with real applications on Android emulator.

Keywords : Android, Security Model, Flexible Permission Management, Permission Promotion

1. 서 론

안드로이드 플랫폼은 기본적으로 시스템자원 또는 개인정보 등에 대한 부적절한 접근을 제한하기 위한 응용프로그램 권한기반 보안모델을 채용하고 있다. 하지만 권한기반 보안모델은 보안성 측면과 사용자 편의성 측면에서 몇 가지 문제점을 가지고 있다. 첫 번째, 사용자가 설치하고자 하는 응용프로그램이 여러 가지 권한을 요구할 경우에 각 권한별로 선택적으로 허용 여부를 결정할 수 없다. 두 번째, 응용프로그램에 한번 권한이 부여되면 해당 응용프로그램을 삭제하지 않는 한 이미 부여된 권한을 이용하는 것을 제한할 수 없다. 세 번째로 사용자 ID 공유 기능이 활성화된 응용프로그램들을 설치할 때 각 응용프로그램에 부여된 모든 권한이 서로 공유되지만 시스템차원에서 이에 대한 공지가 없다. 특히 이 문제는 앞의 두 가지 문제점과 결합될 때 심각한 보안문제를 초래할 수 있다.

본 연구에서는 안드로이드 플랫폼의 기본 보안모델 중 응용프로그램 권한의 특징과 관련된 잠재적인 보안 위협들을 소스코드 분석을 통해 수행하고 응용프로그램들의 권한을 유연하게 설정할 수 있는 기법을 안드로이드 SDK 버전 2.2 Froyo 기반에서 설계, 구현하였다. 본 연구에서 제안한 기법

※ 이 논문은 2009년도 서울시립대학교 교내학술연구비에 의하여 연구되었음.
† 준 회 원 : 서울시립대학교 기계정보공학과 박사과정
†† 종신회원 : 서울시립대학교 기계정보공학과 부교수(교신저자)
논문접수 : 2011년 2월 28일
수 정 일 : 1차 2011년 4월 1일
심사완료 : 2011년 4월 4일

은 기존 시스템에서 제공하는 보안모델의 단점을 보완하기 위해 사용자에게 응용프로그램의 악의적인 행동 여부를 판단할 수 있는 정보를 추가적으로 제공함으로써 사용자에게 편의성을 제공하며 보안성을 향상시킬 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구와 안드로이드 기본 보안모델 중 하나인 응용프로그램 권한의 특징에 대해서 기술한다. 3장은 안드로이드 플랫폼의 권한기반 보안점검과정을 분석하고 이에 따른 설계안과 구현내용을 제시한다. 4장에서는 실제 구현에 따른 동작 검증 결과를 제시하며 마지막으로 5장에서 결론을 맺는다.

2. 연구 배경

안드로이드 보안정책중 하나인 응용프로그램 권한은 다른 스마트폰 플랫폼의 보안모델과 비슷한 개념으로 특정 응용프로그램이 내부의 다양한 데이터, 하드웨어 장치, 또는 다른 응용프로그램의 컴포넌트에 접근하려 할 때 적절한 권한(Permission)을 획득해야만 이를 허용하도록 하는 개념이다[1, 2]. 권한은 특정한 문자열 형태로 정의되며, 개발자가 응용프로그램 작성할 때 기재하고 사용자가 응용프로그램을 설치하는 과정에서 확인할 수 있다. 권한은 시스템 차원에서 미리 정의된 권한과 응용프로그램 개발자가 자신의 컴포넌트 보호를 위해 새롭게 설정한 권한으로 구분할 수 있으며, 해당 권한이 시스템의 동작에 미칠 수 있는 영향의 정도에 따라 normal, dangerous, signature, signatureOrSystem 등 4가지 보호 단계 (Protection Level)로 구분된다[1].

응용프로그램이 권한을 획득하기 위해서는 개발자가 응용프로그램 별로 정의되는 메타파일인 AndroidManifest.xml 파일 내에 <uses-permission> 태그를 이용해 권한명을 지정해야 한다[2]. 만약 응용프로그램이 적절한 권한을 획득하지 못한 상태에서 시스템 자원이나 다른 응용프로그램의 컴포넌트를 접근하려고 시도할 경우, 보안 예외상황 (Security Exception)이 발생하여 정상적인 수행을 할 수 없다[1]. 또한, 각 응용프로그램은 시스템 차원에서 미리 정의된 권한이나 개발자가 직접 정의한 권한명들을 AndroidManifest.xml 파일의 <permission> 태그에 명시함으로써 내부에 정의된 컴포넌트나 데이터를 외부의 허가되지 않은 접근으로부터 보호할 수 있다[1, 2]. 특정 컴포넌트 혹은 데이터 접근을 위한 권한을 얻고자 하는 응용프로그램은 AndroidManifest.xml 파일 내의 <uses-permission> 태그에 해당 권한명을 명시하여야 한다.

사용자 ID 공유(Shared User ID) 기능은 동일한 개발자가 작성한 응용프로그램 간 상호협업을 원활히 할 수 있도록 지원되는 기능으로 이는 동일한 시스템에 설치될 경우 같은 사용자 ID를 부여하고 같은 가상머신 상에서 구동을 하게 하는 기법이다[1,2]. 하지만 사용자 ID 공유를 사용할 경우 각 응용프로그램이 가지고 있는 권한까지 사용자에게 공지 없이 공유되는 현상이 발생 한다[1]. 특히 설치과정에서 사용자 ID 공유가 이루어져 발생할 수 있는 보안 취약점에 대해 사용자에게 어떠한 공지도 없다는 점 또한 큰 문제점이다.

안드로이드 권한 기반 보안모델의 제약사항을 보완하기 위한 기존 연구로는 크게 설치과정에서 미리 정의한 보안 규칙을 적용해 사용자 개입을 최소화하는 방안[3], 응용프로그램의 소스분석을 통해 데이터 흐름을 파악하여 보안상 민감한 데이터 전송동작을 추출해 내는 방안[4], 개발자가 정의한 보안정책에 따라 실제 응용프로그램의 설치과정과 런타임에 요청되는 권한이 정의된 보안정책에 위배되는지를 검증하는 방안[5], 사용자가 직접 응용프로그램 설치 시 요구되는 권한들에 대해 정책을 설정할 수 있고 이를 변경할 수 있도록 하여 보안성을 강화하는 방안[6] 등이 있다. 그러나, 기존 연구들은 각각 사용자가 개입할 여지가 없어 낮은 편의성을 가지며 수행 중 악의적 행위가 탐지되더라도 이에 대처가 불가능하거나[3], 응용 개발자의 부담이 늘어나고 기존 시스템과의 호환성 이 떨어지며 [4], 응용프로그램의 소스 코드를 얻기 힘든 현실적인 문제[5], 권한설정 과정에서 사용자에게 지나친 부담이 지워지는 등의 문제 [6]를 가지고 있다.

3. 권한기반 보안모델 분석 및 개선안 설계

3.1 권한에 근거한 보안점검 과정 분석

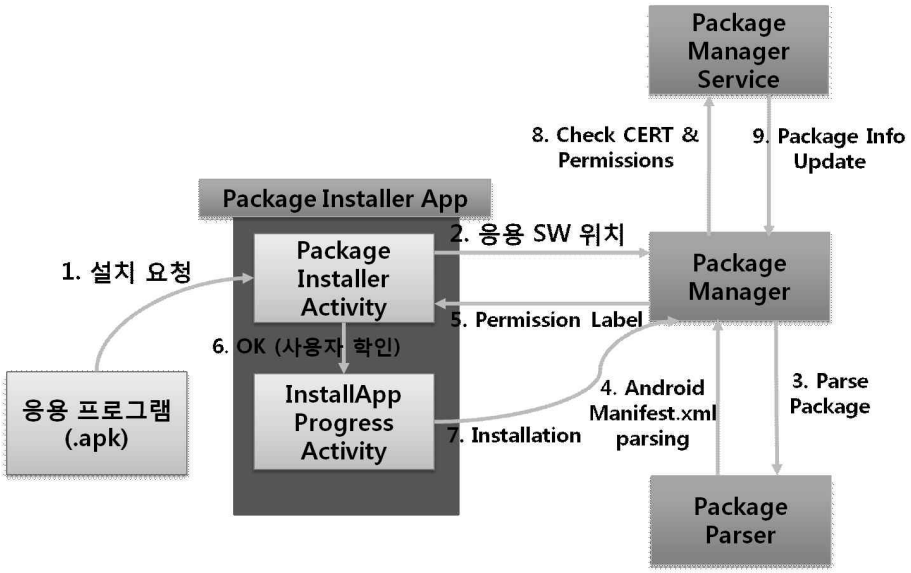
응용프로그램이 가지고 있는 권한이 적절한지 판단하는 것은 응용프로그램이 설치될 때, 실행 중 다른 응용프로그램의 컴포넌트에 접근하거나 시스템 컴포넌트에 특정 작업을 수행 또는 요청할 때이며 크게 보면 응용프로그램 설치 전과 응용프로그램 실행 중으로 구분할 수 있다.

(그림 1)과 같이 응용프로그램이 설치될 때 사용자는 시스템 프로그램인 Package Installer를 통해 응용프로그램이 요구하는 권한을 확인할 수 있다. 사용자가 설치를 허용하면 응용프로그램은 자신이 요구한 특정 기능에 대한 권한을 획득하게 되며, 응용프로그램 프레임워크 계층의 Package ManagerService 컴포넌트가 시스템 내부의 모든 응용에 대해 각각의 권한 정보를 현행화한다. 이 때 권한 보호수준이 signature 혹은 signatureOrSystem인 권한을 요구할 경우 시스템 차원에서 허용조건을 판단 후 권한 부여를 거부할 수 있다.

설치 후 응용프로그램 실행 과정에서 권한으로 보호되는 컴포넌트 호출이 일어날 경우에는 해당 컴포넌트가Package ManagerService의 checkPermission 메소드를 이용하여 시스템이 유지하고 있는 권한정보와 비교 후 코드수행을 허용할지 여부를 판단한다[7]. 그러나 응용프로그램 설치 시 권한이 이미 허용이 된 경우에는 실행 중 점검과정은 보안성 강화 측면에서는 큰 의미가 없다.

3.2 권한기반 보안모델의 개선안 설계

본 연구에서는 기존 안드로이드의 권한기반 보안모델의 특징과 점검과정을 확인한 후 <표 1>과 같은 설계 기준을 수립하였으며, 이를 토대로 응용프로그램 계층의 Package



(그림 1) 응용프로그램 설치 시 권한확인 과정

〈표 1〉 설계 기준

항 목	기존 정책의 한계	보완책
All or Nothing 정책	권한들의 개별적 설정 불가	설치 시 사용자가 각 권한을 부여할 선택권 부여
플랫폼 권한검사 과정	런타임 권한검사 과정은 의미 없음	수행 중 사용자가 부여된 각 권한을 설정할 수 있는 기능 제공
사용자 ID 공유	사용자 ID 공유 활성화 응용들이 권한을 공유하는 것에 대한 사용자 공지가 없음	설치 시 공지 및 해당 기능 활성화 응용 현황 제공
네트워크 접속 현황	현재 없음	응용의 패킷 송수신 현황 및 연결정보 제공

Installer와 Application Management, 응용프로그램 프레임워크 계층의 PackageManagerService를 수정하였다.

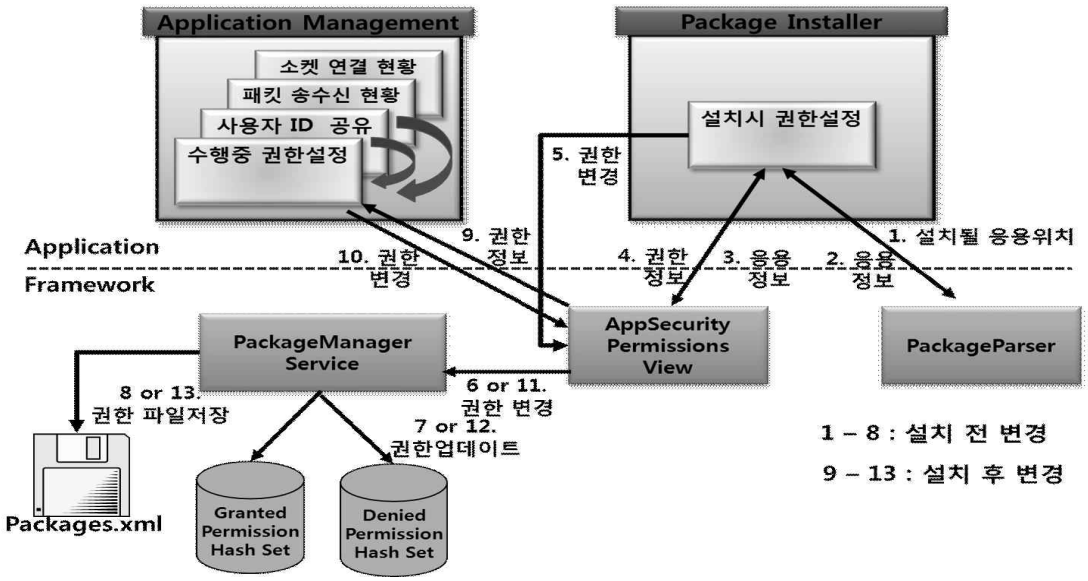
안드로이드 플랫폼 보안 정책의 핵심 구현부인 Package ManagerService는 설치 과정에서 사용이 허가된 권한사항 등 응용프로그램의 특징을 나타내는 자료구조들을 유지하며 이를 제어할 수 있는 메소드들을 제공한다[7]. 또한, 이미 설치된 응용에 대한 정보를 권한 해쉬집합과 XML 파일형태로 파일시스템에 저장하여 이를 시스템 전역정보로 유지하고 있다. 이 중 grantedPermissions 해쉬집합은 설치과정에서 PackageParser 컴포넌트에 의해 업데이트가 되며 해당 응용프로그램이 삭제될 때까지 유지된다.

사용자가 선택적으로 권한을 허가하고 이후 수행과정에서 자유롭게 권한설정을 할 수 있도록 하기 위해서는 denied Permission 해쉬집합을 추가하여 설치 과정에서 허용되지 않은 권한현황을 유지하도록 하고 사용자의 선택에 따라 두 해쉬집합을 변경할 수 있는 API를 작성하는 작업이 필요하다. 또한, 시스템이 리셋된 후에도 변경된 권한을 유지할 수 있도록 XML 파일에 변경된 권한현황을 저장하도록 API를 수정하는 작업이 필요하다.

Package Installer는 PackageManagerService 컴포넌트와 유기적으로 연결되어 있는 시스템 프로그램으로 응용프로그램의 설치와 삭제를 수행한다. 이 프로그램의 가장 중요한 목적은 사용자에게 응용프로그램이 요구하는 권한정보들을 보여주는 것이며 권한부여와 같은 설치 작업은 Package ManagerService 계층에서 처리한다. 따라서 사용자가 각 권한을 선택할 수 있도록 기존의 UI를 변경하고 이벤트 핸들러를 등록하여 발생 이벤트에 따라 추가된 두 개의 해시집합을 변경하는 API를 사용할 수 있도록 한다. 그리고 사용자 ID 공유기능 활성화 응용 설치 시 이를 사용자에게 공지하도록 한다.

Application Management는 시스템에 설치된 모든 응용프로그램에 대한 정보들을 사용자에게 공지하는 역할을 수행한다. 이중 응용에 허가된 권한정보를 보여주는 액티비티는 Package Installer와 비슷한 방식으로 구현되어 있기 때문에 동일한 방식으로 권한현황을 업데이트할 수 있도록 한다.

또한, 사용자 ID 공유 기능을 사용하는 응용프로그램들의 정보를 확인할 수 있도록 하며, 응용의 악의적인 행동 가능성을 모니터링 할 수 있도록 현재 TCP 소켓 연결현황과 각 응용별 패킷 송수신 현황을 추가하도록 한다. 연결현황과 패킷 송수신 현황은 리눅스에서 제공하는 proc 파일 시스템을 이용하여 구현하며, 추가된 기능들은 권한변경 화면과 연결하여 사용자의 편리한 권한설정을 돕도록 한다. (그림 2)는 앞에서 제시한 설계기준에 따른 구현 구조도를 나타낸다.



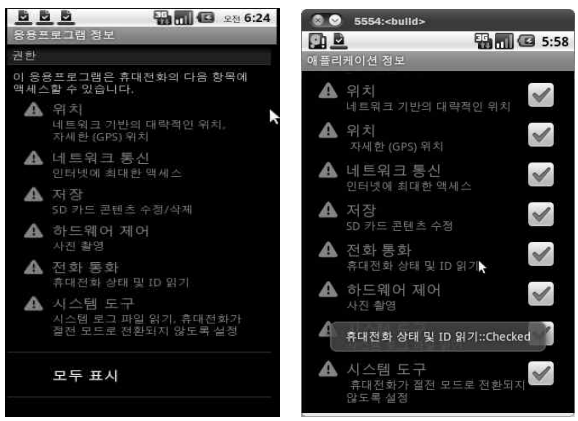
(그림 2) 구현 구조도

4. 권한기반 보안모델 개선안 검증

4장에서는 3장에서 제시한 설계방안을 이용하여 구현한 내용과 검증결과를 보인다.

(a) 플랫폼 수정 전 (b) 플랫폼 수정 후
(그림 3) 응용프로그램 설치 시 권한설정

(그림 3)과 (그림 4)는 각각 본 연구에서 구현한 선택적 권한설정 방안에 따른 동작 결과를 나타낸다. 먼저 (그림 3)은 설치 시에 각 권한을 개별적으로 설정한 결과이다. 대부분의 카메라 응용은 사진을 찍는 기능 이외에 GPS를 사용하여 사진에 현재 위치정보를 저장하고 이를 네트워크를 통해 전송할 수 있는 기능들을 제공한다. 하지만 (그림 3) (b)의 READ_PHONE_STATE 권한을 이용해 응용프로그램이 장치의 전화번호, USIM의 Serial Number와 같은 정보들을 접근이 가능하고, 이러한 정보들이 네트워크를 통해서 외부로 유출될 가능성이 있으므로 이 권한에 대해서 선택적 불허를 함으로써 잠재적인 보안 위협을 줄일 수 있다. READ_PHONE_STATE 권한을 부여하지 않고 실행해본 결과 정상적인 동작을 수행했으며 내부에 포함된 광고 모듈에서만 보안 예외상황이 발생한 것을 확인할 수 있었다. 하지만 해당 권한이 응용의 수행에 있어서 꼭 필요한 권한이라면 보안 예외상황이 발생하여 정상적인 수행이 불가능해지므로 실행 중에도 (그림 4)와 같이 권한을 재설정할 수 있도록 하였다.



(a) 플랫폼 수정 전 (b) 플랫폼 수정 후
(그림 4) 응용프로그램 설치 후 권한설정

(그림 5)는 구현된 사용자 ID 공유 기능이 활성화된 응용에 대한 정보를 나타낸다. 응용프로그램 설치 시 기존에 설치된 응용이 있는 경우 (그림 5) (a)와 같이 응용의 이름과 해당 응용으로부터 얻게 되는 권한에 대해 공지하며 설치 후에는 (그림 5) (b)와 같이 동일한 사용자 ID를 갖게 되는

응용들의 정보를 Application Management를 통해 확인할 수 있다. 이와 같은 정보를 사용자에게 제공함으로써 사용자 ID 공유 기능을 악용하기 위해 작성된 응용프로그램의 권한 상승을 예방할 수 있다.

(그림 6)은 네트워크를 통한 사용자 정보유출 시나리오를 고려하여 추가한 네트워크 접속 현황 기능을 보여 준다. 사용자는 (그림 6) (a)와 같이 과도한 양의 패킷을 전송한 응용프로그램이 있는지 여부와 (그림 6) (b)와 같이 소켓을 생성한 응용프로그램과 해당 소켓의 현재 상태 등을 알 수 있으며, 이를 근거로 (그림 4) (b)에 제시된 선택적 권한 설정을 할 수 있다.

(a) 권한 공유 공지 (b) 공유기능 활성화 응용현황

(그림 5) 사용자 ID 공유 활성화 현황정보

(a) 송수신 패킷 현황 (b) TCP 소켓 연결현황

(그림 6) 네트워크 접속 현황

5. 결 론

본 연구에서 제안한 기법은 기존 보안모델에 비해 사용자에게 선택의 기회를 제공한다는 점이 가장 큰 장점이다. 또한 사용자에게 권한 설정에 대한 판단 근거를 제공하여 잠재적인 보안 위협을 줄일 수 있는 특징을 가지고 있다. 하지만 여전히 권한부여에 대한 사용자의 책임이 큰 점에서 한계는 존재하기 때문에 추후 연구로는 보안위협이 발생할 수 있는 권한조합을 분류하고 권한설정 시 사용자의 부담을 줄일 수 있는 프레임워크 개발을 고려하고 있다.

참 고 문 헌

[1] A. Shabtai, Y. Fledel, U. Kanonov, Y. Elovii, and S. Dolev, "Google Android: A State-of-the-Art Review of Security Mechanisms", IEEE Security and Privacy, Vol.8, issue2, pp. 35-44, 2010.

[2] Google Android Developers Guide, "Security and Permissions", http://developer.android.com/guide/topics/security/security.html, Aug., 2010.

[3] W. Enck, M. Ongtang, and P. McDaniel, "Understanding Android Security", IEEE Security and Privacy, Vol.7, issue 1, pp.50-57, 2009.

[4] A. Fuchs, A. Chaudhuri, and J. Foster, "SCanDroid: Automated Security Certification of Android Applications", In Proc. of the 31st IEEE Symposium on Security and Privacy, 2010.

[5] M. Ongtang, S. McLaughlin, W. Enck, and P. McDaniel, "Semantically Rich Application-Centric Security in Android", In Proc. of the Annual Computer Security Applications Conference, pp.340-349, 2009.

[6] M. Nauman and S Kahn, "Apex: Extending Android Permission Model and Enforcement with User-Defined Runtime Constraints", In Proc. of the 5th ACM Symposium on Information, Computer and Communication Security, pp. 328-332, 2010.

[7] Android SDK 2.2 Full Source, "Android Application Framework: PackageManagerService", <android_full_ource>/frameworks/base/services/java/com/android/server/PackageManagerService.java, March, 2010.

김 익 환

e-mail : duo830210@uos.ac.kr

2008년 서울시립대학교 기계정보공학과 (학사)

2011년 서울시립대학교 기계정보공학과 (공학석사)

2011년~현 재 서울시립대학교 기계정보공학과 박사과정

관심분야 : 내장형시스템, 실시간시스템, 무선통신

김 태 현

e-mail : thkim@uos.ac.kr

1994년 서울대학교 컴퓨터공학과(학사)

1996년 서울대학교 컴퓨터공학과 (공학석사)

2001년 서울대학교 전기 컴퓨터공학부 (공학박사)

2001년~2005년 (주)지씨티 리써치 책임연구원

2005년~현 재 서울시립대학교 기계정보공학과 부교수

관심분야 : 내장형시스템, 실시간시스템