스마트폰 환경의 응용 소프트웨어 개발과정에서 보안정책 이슈

홍진근*

Security Policy Issue in Application Software Development Process of Smart Phone Envrionment

Jin-Keun Hong*

요 약 스마트 폰 환경에서 개발되는 어플리케이션 소프트웨어는 시스템 개발방법론이 적용되어오고 있다. 본 논문은 스마트 폰 환경에서 개발되는 주요 응용 프로그램에 요구되는 보안 고려사항을 살펴보았다. 먼저 어플리케이션 프로그램에서 보안 문제를 살펴보았고, 안전한 어플리케이션 프로그램을 위한 보안정책 문제를 고찰하였다.

주제어 : 보안, 소프트웨어, 스마트, IT, 어플리케이션

Abstract The application software, which is developed on smart phone environment, is applied to according to system development methodology. This paper presents security consideration, that is required to major application program, which is developed in smart phone environment. First it reviews security issues in application program, and the next it considered to security policy for secure application program.

Key Words : security, software, smart, IT, application

1. 서론

스마트 폰 환경에서 어플리케이션 소프트웨어를 기반으로 하는 시스템은 시스템 개발방법론을 기반으로 설계 및 구현되어 오고 있다. 보안 소프트웨어의 경우 보안 개발 라이프사이클을 자체적으로 개발 및 검증에 대한 요구가 지속적으로 증가하고 있는 실정이다. 미국 정부는 국방부를 중심으로 security SDLC 개념을 제도화하여 실시해오고 있으며, 특히 상무부 산하 NIST를 중심으로 연방정부 및 민간 지침을 제시하여 적용해오고 있는 실정이다. 상용 분야에서는 MS사가 제품 개발과정에 보안 개발을 위한 라이프 사이클(분석, 설계, 구현, 시험단계)을 자체적으로 개발 적용하고 있으며 이로 인해 취약점의 개수를 60%이하로 감소시키고 있다고 발표되고 있다. 이러한 노력들은 국내 주요 통신사나 은행, 관련 개발사 등에서도 추진되어오고 있는 실정이다.[1-3]

기존 연구에서 Ben Smith 등은 보안 테스트 패턴의 효과적인 사용이라는 주제로 보안 테스트 패턴 내에 포함된 보안 요소에 접근하기 위한 블랙박스 보안 테스트를 효과적으로 생성하는 방안에 대해 연구한 바 있다.[4]

Ansar et. al 등은 CIA를 기초로 보다 나은 결과를 위한 보안 소프트웨어에 대한 방안들에 관해 접근하고 있다. 소프트웨어 개발시에 효과적인 보안 실행 요소들에 대해 제시하고 있다.[5]

llkka et. al 등은 소프트웨어 개발 과정에서 보안 보증을 위한 평가에 대한 연구를 수행한 바 있다.[6] 이 연구는 소프트웨어 개발 라이프 사이클의 올바른 가치 측면에서 가이드라인을 제시하고 있다. 또한 이 휴리스틱한 평가에서는 명시된 특정 상황과 비교하여 소프트웨어 프로세스를 위한 보증 수준을 도출할 수 있는 단계라는 측면에서 의미를 가진다고 볼 수 있다.

Philipp Zech et. al 등은 서비스 중심의 시스템의 위험

*백석대학교 정보통신학부
논문접수: 2012년 10월 23일, 1차 수정을 거쳐, 심사완료: 2012년 11월 20일

위주의 보안 테스팅에 대한 연구를 하였는데, 위험 위주의 보안 테스팅이라는 방법론에 대해 접근하고 있다.[7]

Mariantonietta et. al 등은 모바일 디바이스를 위한 보안 문제에 대해 살펴본 바 있다. 이 연구에서는 현대 모바일 환경에서 주요 보안 솔루션, 위협, 취약성에 대해 다루고 있다.[8]

그러므로 소프트웨어에서 보안 테스트나 소프트웨어 개발시에 보안 실행요소에 대한 고려, 그리고 소프트웨어 개발 프로세스에서 개발 평가 방안이나 위험을 기반으로 하는 보안 테스팅, 모바일 환경에서 보안 문제 등은 여전히 현실적인 이슈로 고려되고 있다.

이러한 배경 하에 본 논문에서는 스마트 환경에서 어플을 개발할 때 고려될 수 있는 주요 보안 요소들에 대한 정책적 접근은 나름대로의 의이를 가지는 것으로 고려되므로 이에 대한 연관된 보안 문제와 정책 문제들을 고찰하였다.

본 논문에서는 우선 2장에서 어플 개발과정에서 보안 문제를 살펴보고, 3장에서 안전한 어플을 위한 보안정책에 대해 기술한다. 마지막 4장에서 결론을 맺었다.

2. 어플 개발과정에서 보안 문제

어플의 보안성 개선은 설계단계, 개발단계, 테스트단계, 출시 유지보수 단계에서 적절히 이루어져야 한다.

2.1 설계단계

보안개선을 위해 개념정립에서부터 보안위협 모델을 수립하고 보안 교육을 실시해야 한다. 설계 완성단계가 되면 보안팀과 이 내용을 가지고 검토해야 한다. 구현단계에서 수정은 시간, 비용, 노력 측면에서 설계단계의 10배 이상 요구되므로 설계단계에서의 중요성이 강조된다.

2.2 개발단계

어플 개발 단계에서는 수정이 필요한 버그에 대한 기준을 정하고 단종 계획수립이나, 데이터의 변경여부, 최소권한 테스트를 실시해야 한다. 코드가 완성되면 결함을 체크하고, 안전한 코딩의 가이드 라인을 제시해야 한다. 코드를 작성하고 디버깅을 수행할 때 , 권한관리가 요구된다. 변경된 소스 코드를 용이하게 검토할 수 있도록 지원하는 소프트웨어 또한 요구된다. 안전한 코딩에 대한 가이드라인도 제시되어야 한다. 일반적으로 알려진 security push 작업은 위협모델링 수립, 신규 유형의 문제점 탐지 통보, 버그와 문제점 검토, 취약점 탐지하여 공지, 버그 측정 기준 유지 등의 작업이 될 수 있다.

2.3 테스트단계

이 단계에서는 개발된 어플에 대해 외부 팀과 검토하고, 출시를 준비한다. 시스템 설계 및 코드가 공격에 견딜 수 있는지 여부를 테스트한다.

2.4 유지보수단계

출시 제품에 대한 유지보수, 대응 프로세스를 가동시켜야 한다. 제품의 안전성이나 취약점 침해가능성 여부를 파악하고, 결함 발견시 패치 요구에 대한 조치나 문제점을 탐지하고 조치하도록 한다.

3. 안전한 어플을 위한 보안 정책

일반적으로 보안의 안전성을 평가할 때 보안이 설계된 코드보다 보안을 최우선의 기능으로 설계된 코드를 보다 더 안전한 코드로 제시한다. 또한 보안 기능이 안전한 기능이 아니라는 점이다. 보안 소프트웨어에 보안 버그가 없다고 생각하면 되지 않는다. 코드나 설계에서 보안 문제가 발견된다는 것은 보안 결함이 주위에 여러 개 결함으로 연결되어 발생될 수 있다는 점을 암시한다. 그러므로 취약점이 나타나면 주변 가장 근접한 곳에서 수정해야 한다.

3.1 안전한 어플 개발에서 보안 원칙

설계시 보안 원칙은 보안 위협모델의 완성시점을 고려하여 침해 가능 요소와 개선점을 파악해야 한다. 또한 설계와 코딩 가이드를 수립하고 준수할 것, 위반하는 버그에 대해 수정 및 가이드 갱신, 발견된 취약성에 대한 테스트방법 개발, 코드와 보안 모델 단순화 및 모의 테스트 결과 분석을 실시해야 한다. 기본적인 보안원칙은 디폴트 설치나 어플의 최소권한 설정, 자원에 대한 적합한 방어책을 제시해야 한다. 적용시 보안원칙은 어플이 보안 관리 기능을 갖도록 하고 패치 및 이에 대한 운용매뉴얼이 제시되어야 한다. 앞에서 언급된 안전한 어플 개발에서 요구되는 보안원칙과 함께 고려되어야 할 주요 항

목들을 <표 1>에서 나타내었다.

〈표 1〉 기타 보안원칙

항목	내용
어플 코드 오류 방지	보안 오류 발생경로 파악조치
	코드 동일형태오류 복사여부
	사전 예방가능성 여부파악
	동일형태오류 예방방법
개발후 검토	품명/버전/담당/버그번호관리
	취약점 설명 및 영향정도파악
	디폴트 환경에서 취약점여부
	결함예방 위한 조치수행여부
	코드수정시 내용, 변경 관리
공격영역 최소화	개방된 소켓/파이프/RPC 종단 수
	서비스 수(기본/높은 권한 실행)
	어플 수
	파일, 폴더, 레지스터리 키 수

또다른 보안 원칙으로 다음 표와 같은 어플 코드의 오류 방지하기 위한 고려, 개발후 검토과정이나 공격 가능한 영역에서 이를 최소화하는 방안이 고려되어야 한다.

3.2 안전한 어플 개발을 위한 정책 고려사항

스마트폰 어플이 안전하지 않을 때 장기적인 측면에서 이를 패치하기 위해 훨씬 많은 작업시간이 요구된다는 점이다. 또한 어플이 보안 수준이 회사의 이미지와 영업 이익에 영향을 미친다. 개발자 입장에서는 안전하게 동작하는 어플을 기대하지만 실제 공격받는 어플은 어느 시점에 정상적으로 동작하지 않을 수 있는 가능성을 가지고 있다. 우리가 이해하는 안전한 어플은 현재 동작 환경에서 에 충분한 유연성과 안전성을 가진 소프트웨어 즉 안전성과 안정성을 가진 소프트웨어를 말한다.

보안 패치를 고려할 때, 패치 작업은 높은 비용을 지불해야 하므로 이에 대한 제반 비용을 산정에 산정해야한다. 이들 비용 가운데는 취약점이 있는 코드 탐지를 위한 개발자 비용이나, 취약점 수정에 필요한 비용, 교정된 결과를 테스트하는데 필요한 비용, 설치나 테스트 비용, 국가별 언어에 따른 보안 패치를 생산하고 테스트하는데 소요되는 비용, 보안 패치를 디지털 서명하는데 요구되는 비용, 패치를 웹에 게시하는데 요구되는 비용, 패치에 대한 설명이나 사용자 설명서, 회사 신용 개선비용 등을 포함한 제반 소요비용을 고려해야 한다.

3.3 안전한 어플을 제공하기 위한 요구사항

스마트 환경에서 안전한 어플에 요구되는 주요 보안 항목들에 대해 다음 <표 2>에서 나타내었다.

〈표 2〉 어플이 제공해야 하는 보안 항목

항목	내용
사용범위	사용하지 않은 보안 어플이 응용소프트웨어에 영향을 주지 않도록 적합한 범위 제한
최소권한원칙	어플 실행에서 취약점 노출을 위한 최소권한의 원칙 수립
기밀성	최적의 암호기법, 안전한 저장, 저장메모리 관리 등이 제공
무결성	악의적인 컴포넌트에 노출되지 않도록 실행기능을 제한
가용성	식별,권한,인증 가이드라인가지고 어플 테스트
식별및인증	어플에 접근통제 정책구현인증
허가권한	객체 사용권리에 대한 무결성
감사및로깅	어플 활동 모니터링 및 추적
입력유효성	입력유효 및 버퍼 범위 등 체크

3.4 보안 위협 모델링 문제

위협모델은 어플에서 가장 취약한 보안 위험이 무엇인지, 이를 공격자가 어떻게 이용할지를 분석하는 것이다. 사실 버그는 50%가 위협모델로부터 탐지되고, 코드 분석이나 테스트로부터 50%가 탐지되는 것으로 알려져 있다. 위협모델은 대응할 위협을 파악하고 방안을 정하기 위한 일련의 작업이라 할 수 있다. 위협모델링은 실행팀 조직, 어플 분석, 어플 위협요소 정의, 정해진 위협의 우선순위 선정과 대응기법을 선정한다.

3.5 보안 테스트 문제와 정책

어플의 보안 테스트 실시 목적은 기능이 정상적으로 동작하는가를 확인 하는 것이 아닌 대응 메커니즘이 잘 동작하고 있는지 여부를 판별하는데 있다. 즉 식별된 위협 완화기술이 정상적으로 동작하는지, 그리고 위협모델에 발견되지 않은 문제를 존재하지 않는지를 파악하는데 있다. 테스트 항목에는 컴포넌트, 컴포넌트 위협 타입, 위험으로 분류할 수 있다. 컴포넌트 인터페이스 식별에서 인터페이스를 식별하고 보안 버그를 탐지한다. 인터페이스 테스트 순위선정에서는 인터페이스와 함수를 실행하는 프로세스가 계정과 관련된 문제, 데이터 처리 인터페

이스와 상위 레벨 언어와 관계, 인터페이스가 버퍼 크기와 문자열 관계, 버퍼와 스택과 관계, 접근통제 메커니즘의 취약성 문제, 인터페이스와 리소스와 접근통제의 적합성 문제 등이 고려되어야 한다. 테스트 과정에 탐지된 결함은 제거되어야 하고 초기 동작이 가능하도록 수정되어야 한다. 공격이 이루어지는 컴포넌트에 대해서는 공격벡터와 벡터 성격에 대한 확인, 그리고 예측 가능한 벡터 수를 카운터 해야 한다. 소켓의 경우 자주 공격받는 지점이므로 예측되는 어플 공격 벡터 수에 대한 계산이 요구된다.

어플에서 보안 테스트를 실시하기 위해서는 fuzz 테스팅, 침투테스팅, 런 타임 검증, 필요할 경우 위협 모델에 검토와 함께 업데이트, 표면공격에 대한 재평가 등으로 이루어지고 있다. Fuzz 테스팅은 보안 버그의 특정 클래스를 발견하는데 효율적인 방법으로 알려져 있으며 잘못 생성된 데이터에 대해 어플이 어떻게 반응하는지 여부를 평가한다. 일반적으로 보안 버그의 25%는 퍼징 테스트에 의해 파악된다. 퍼징 테스트는 데이터 구조 분석 실행코드로 그래픽 이미지나 문서나 실행파일 형식에 관한 것이나 네트워크 프로토콜을 분석하는데 요청하기 전에 응답을 수행함으로써 네트워크 오퍼레이션을 퍼지하거나, API 같은 플러그된 프로토콜을 취급하는 3개의 기능을 가진다. 그러므로 퍼징은 파일 형식을 식별하고 파일 퍼지 실행하여 어플이 퍼지된 파일이나 어플을 관측하게 된다. 어플 보안 위협과 테스트 되어야 할 사항들에 대해 <표 3>에서 나타내었다.

〈표 3〉 어플 보안위협과 테스트

항목	내용
스푸핑	어플 인증적용 여부, 인증프로토콜의 적합성, 신용정보 노출, 자격증명 공격과 오류메시지 존재 여부
데이터 변조	권한부여나 접근통제 메커니즘 적용여부, 데이터변조 및 재생성여부, 안전하지 않은 어플로 회귀성 여부
서비스거부	로그기록 거부여부, 적합하지 않은 이벤트 생성여부, 보안체크 없는 작업 여부
정보유출	데이터 노출여부, 프로세스 비정상 종료나 디스크 청소 실행 여부, 저장된 데이터 탐지여부
권한	권한에 따른 적절한 실행여부, 코드로 데이터 실행가능성 여부, 상승된 권한으로 실행가능한 어플 기능
보안 위협	어플 환경에서 새로운 보안 위협에 대한 접근 방법여부
OS환경위협	루트 킷 공격에 대응하는 능력여부
화면 보호	화면 스크레이퍼, 키 로거 취약점에 대한 취약성 여부
코어프로세스 위협	코어 프로세스에 취약점을 기반으로 하는 위협
다양한 디바이스에 따른 위협	다양한 기기에 기반하는 위협으로부터 보호 여부
OS 이외 보안 관리	OS 이외 보안 관리를 위한 기술과 위협 요소 탐지 여부
고정 디바이스 위협	고정형 디바이스 장비에 대한 종단 보안 취약점 여부(허용, 차단)

3.6 보안 어플의 설치 문제

설치 프로그램이 어플을 안전하지 않게 만들고 외부 프로세스를 호출할 수 있다는 점이 지적된다. OS 자체가 디폴트로 보안 홀이 개방되어 있을 가능성이 있는 보안 설정의 경우도 고려되어야 한다.

3.7 어플에 개인정보 포함 문제

어플에 개인정보가 포함될 경우 개인정보보호정책, P3P(platform for privacy preferences) 내용이 포함되어야 하는데, 개인정보구현 여부와 내용에 대한 유효성 검사가 검토되어야 한다.

어플에서 개인정보 설계 템플릿은 기능 설계 템플릿에 개인정보에 대한 내용이 문서화되어야 한다. 사용되는 데이터가 무엇인지, 사용은 누가하는지, 얼마나 오랫동안 저장되며, 이 기능으로 사용자가 얻는 가치가 무엇인지, 데이터를 전송하거나 수정가능한 능력이나 데이터 저장 이전에 사용자 권한이 주어져 있는지 여부나, 데이터 저장이나 사용전에 마지막 사용자 설정이 적용되고 있는지, 데이터 접근에 대한 보호가 적용되는지, 데이터 암호화가 이루어졌는지, 3자와 데이터 공유가 이루어지고 있는지 등이 검토되어야 한다. 어플에서 민감한 정보와 트랜잭션에 대해서는 차폐된 구조를 갖는 프로세스가 제공되어야 한다. 민감한 정보라면 추적 가능하도록 해야 하고, 성능이 뛰어난 알고리즘이나 보호 키 관리 방안 등이 제공되어야 한다.

3.8 어플과 연동 서버에서 보안 설계 원칙

어플과 연동되는 서버 사이에는 여러 가지 고려되어야 하는 보안 요소들이 있다. 먼저 민감한 정보는 어플에 의도적으로 저장하지 않아야 한다. 민감한 정보는 SSL/TLS를 통해 안전하게 전송되어야 하며, 전송된 정

보는 로컬로 캐시 되도록 할 것, API는 민감한 연결 스트링과 암호 키를 보호하기 위해 서버 상에서 적용되도록 설계되도록 할 것, DB 서버는 인증정보를 보관할 것, DB 서버는 서버 허가권한 기술과 암호를 사용하여 인증정보를 적합하게 보호할 것, IIS, ASP.net에 강제 인증 적용할 것, 웹 서비스 보안 설정에 파일 보관과 적법한 관리자에 의해 처리되며 OS 접근통제리스트에 의해 강제 적용될 것, 서버 어플 설정 프로그램이 접근통제리스트 설정할 것, 적법한 관리자가 임의서버 관리나 물리적 접속 가능하도록 할 것, 적법한 관리자가 웹과 DB 서버에 물리적 접속가능하도록 설계할 것, 서버 상에서 사용가능한 어플에 대한 여부를 확인할 것, DB는 고유의 인증프로토콜 설계하여 사용할 것, DB 연결 정보는 어플 특정 파일에 저장하고 보호할 것 등이다.

3.9 보안 어플에 대한 보안공격 문제

어플은 프로그램 상의 민감한 데이터가 악의적인 행동자에 의해 오염되는 스택 스매싱과 같은 공격을 받게 된다. 이로 인해 스택 내 리턴 주소가 오염된다. 어플이 악의적인 입력 파일을 실행함으로써 버퍼 오버플로우를 발생시키고 이로 인해 로컬 변수가 오염되고 스택 상에 저장된 함수의 리턴 주소가 오염되는 것이다. 그러므로 어플 설계시에 이 부분에 대응 방안이 검토되어야 한다.

민감한 변수, 함수 포인터 등 프로그램 힙 상에 저장된 민감한 데이터 구조가 힙 오버플로우 공격에 의해 오염될 수 있다. 이 공격은 인접 주소에 할당된 주소에 위치한 버퍼를 오버플로우시켜 데이터나 포인터를 변경하고 임의 파일에 접근하거나 임의 코드를 실행하므로 이 부분에 대한 대응방안이 검토되어야 한다.

변수 입출력 문에서 출력 문에 적합하지 않은 방법으로 실제 메모리 주소를 공격하여 요구하는 값으로 변경하거나 시스템 루터 권한을 획득하게 되는 포맷 스트링 공격에 대한 대응방안이 검토되어야 한다.

무효한 입력에 대한 오류 즉 정확하지 않은 문법이나 관계 없는 입력 필드를 받을 때 필드 값의 상호관계 오류 문제에 대한 대응이 검토되어야 한다.

하나의 매크로 스텝 안에 변수 상태 값이 두 번 이상 변경되거나 출력 이벤트가 두 번 이상 출력되는 것으로 정의되는 레이스 조건에 대한 문제로, 출력 이벤트와 변수 상태 값이 하나의 실행동안 지속되어야 하나 두 번 이상 변경되어 위반이 일어날 수 있다. 이 문제가 검토되어야 한다.

프로그램이 특정 머신이나 특정 환경 아래 실행되는 경우 어플이 설계된 것과 사용 중인 환경변수가 다를 경우 일어나는 문제에 대해 검토가 있어야 한다.

지금까지 스마트폰 어플 개발과정에서 제반 보안 이슈를 살펴보았다. 스마트 폰에 적용되는 보안 정책은 무엇보다 어플리케이션 프로그램의 유형이나 폰의 종류, 위치, 사용자의 역할, 데이터 전달 모델 등을 정의하고 있어야 한다. 또한 플랫폼 기반 디바이스 인증 프로그램이나 데이터 암호, 보안 인증 방안 등이 함께 정책에서 제시되어야 한다. 물론 보안 통제는 일관된 방식으로 제시되어야 하며 조직의 규모와 관계없이 암호화, 백업 등의 서비스 신뢰성 문제, 보안의 최소권한의 정책 등이 수립되어야 한다. 그러므로 스마트폰 보안 정책에서는 허용 디바이스 유형을 정의하고, 계정을 기반으로 IT 기반 구조와 연동가능 하도록 하고, 모바일 장치를 통해 위험 예측이 가능하도록 하며, 리소스 관리의 수행과 함께 허용 가능한 어플리케이션 디바이스와 유형을 사용할 수 있도록 그리고 안전하게 저장하고 전송하는 방법을 보안 정책에서 명확하게 정의하도록 요구하고 있다.

4. 결 론

보안 소프트웨어의 경우 보안 개발 라이프사이클을 자체적으로 개발 및 검증에 대한 요구가 지속적으로 증가하고 있는 실정이다. 소프트웨어 개발시에 효과적인 보안 실행 요소들에 대한 관심들이 고조되고 있다.

본 논문에서는 스마트 폰 환경에서 응용 소프트웨어 즉 어플을 개발할 때 고려되는 보안 문제와 시스템 개발 운용 전 과정에 고려되어야 하는 보안 정책적인 요소들에 대해 살펴보았다. 본 논문은 어플 개발 과정에서 제기될 수 있는 보안 이슈를 리뷰하였다는 점에서 의미가 있는 것으로 사료되며 향후 연구에서는 스마트 폰 환경에서 세부적인 리버스 엔지니어링 문제에 대해 살펴보고자 한다.

참 고 문 헌

[1] www.ushmm.org/research/doctors/Nuremberg_Code.htm.
[2] Curphey, Araujo(2006). Web Application Security

Assessment Tools, IEEE Security and Privacy archive, Volume 4 Issue 4, pp.32-41.
[3] B. Chess, G. McGraw(2004). Static Analysis for Security, IEEE Security & Privacy, pp.79-84.
[4] Ben Smith and Laurie Williams(2012). On the Effective Use of Security Test Patterns, the proceedings of IEEE 6th SERE 2012, pp.C4.
[5] Ansar UI Haque Yasar, Davy Preuveneers, Yolande Berbers(2008). Best Practices for Software Security : An Overview, the proceedings of IEEE INMIC2008, vol.12, pp.169-173.
[6] Ilkka Uusitalo, Kaarina Karppinen, Pasi Ahonen(2009). Towards Evaluation of Security Assurance during the Software Development Lifecycle, the proceedings of IEEE ARES 2009, pp.817-822.
[7] Philipp Zech, Michael Felderer, Ruth Breu(2012). Towards Risk-Driven Security Testing of Service Centric Systems, the proceedings of IEEE QSIC2012, pp.140-143.
[8] Mariantonietta La Pllla, Fabio Martinelli, and Daniele Sgandurra(2012). A Survey on Security for Mible Device, IEEE communications Surveys and Tutorials, Vol. PP, Issue 99, pp.1-26.

홍 진 근

보통신학부 교수

- 1991년 2월 : 경북대학교 전자공학과 (공학사)
- 1994년 2월 : 경북대학교 정보통신공학 (공학석사)
- 2000년 2월 : 경북대학교 정보통신공학 (공학박사)
- 2004년 3월 ~ 현재 : 백석대학교 정

- 관심분야 : 정보화정책, 연구윤리정책, 금융보안
- E-mail : jkhong@bu.ac.kr