특정 AP를 이용한 안드로이드 기반 모바일 보안 메신저 구현

Implementation of Android-Based Mobile Messenger with Security Function Using Specific AP

김지형*, 이충호*
Ji-Hyung Kim*, Choong- Ho Lee*

요 약

안드로이드를 탑재한 스마트 폰에서 사용자 그룹간의 통신 메신저를 사용하는 경우 해킹 또는 검열에 의한 개인 정보 유출이 최근 큰 문제 중 하나로 대두되고 이다. 이 문제를 해결하기 위하여 일반적으로 기업에서는 내부 메신저에 복잡한 보안 절차를 적용시키고 있으나 이는 기존의 업무 효율성을 저하시키는 결과를 초래한다. 본 논문에서는 특정 AP(Access Point)에 접근한 클라이언트만 통신 할 수 있도록 특정 그룹만의 폐쇄된 메신저를 구현함으로써 업무효율성을 높이면서도 보안을 강화할 수 있는 방법을 제안한다. 이 방법은 기존의 일반적 방식과 달리 서버에서 데이터를 보관하지 않으며, 클라이언트가 특정 AP와 접속이 단절되는 순간 메신저의 내용을 자동으로 삭제하도록 구현되어 있어 보안 및 검열문제를 해결하고 있다. 제안된 방법은 안드로이드 메신저로 구현하여 그 유효성을 검증하였다.

ABSTRACT

Personal information leakage is one of the great recent issues, which can be caused by hacking or censorship when a messenger program is used between user groups in Android-based smartphones, In order to solve these problems, companies generally apply complex security procedures inside the messenger, but which degrade the efficiency of work. This paper proposes a method which can improve work efficiency and strengthen security by allowing only the clients that can access a particular AP, and implementing a closed messenger group. The proposed method is validated by implementing on the Android-based smartphones.

Keywords : Wireless Security, Access Point, Android Application, Mobile Messenger

I. 서론

최근 안드로이드 운영체제를 탑재한 스마트폰이 사용되면서 그에 따른 사생활 및 정보 유출 피해가 증가하고 있다. [1]의 내용에 따르면 정부 핵심 자료와 고위 간부들의 대화내용이 민간 모바일 메신저 서비스인 카카오톡에 무방비로 유출되고 있음을 알 수 있다. 그리고 [2]의 논문에 따르면 스마트 단말기의 대중화로 모바일을 통한 업무처리가 선호되고 있는데, 사이버 보안 위협의 특징이 기존의 인터넷에서 발생되는 보안 위협과 동일하게 재현되고 있고, 특히 웹 어플리케이션 보안 표준 기관인 OWASP(The Open Web Application Security Project) 가 경고하는 세션 관리의 취약점은 앞으로 모바일 환경에서도 대비해야 하는 핵심적인 이슈이나 그 대응방안이 매우 미흡한 상황이라고 정의하였다.

이러한 문제를 해결하기 위하여 보안 소프트웨어를 계속 업데이트한다고 하더라도 해킹 기술은 무궁무진하며 삭제된 대화내용을 복원하거나 메시지를 중간에 가로챌 수 있는 챌 수 있는 위험도 상존한다.

따라서 이 문제에 대한 대책으로 많은 기업들에서는 복잡한 보안 절차들을 적용시키고 있다. 하지만 이런 복잡한 보안 절차들은 많은 부작용과 취약점들이 있음이 알려져 있다. 다음 표 1은 한국인터넷진흥원(KISA)에서 발표한 기업의 관리적 보안 결함 최우선순위 10가지 중 핵심 내용을 정리한 것이다.[3]

* 한발대학교
투고 일자 : 2015.6.2 수정완료일자 : 2015.7.22
게재확정일자 : 2015.8.8

표 1. 기업의 관리적 보안 결함.
Table 1. Security flaws of the enterprise management.

순위	발견된 취약점	통제내용
5	보안사고 예방 및 대응 절차 미흡	보안사고의 정의 및 범위, 긴급연락체계 구축, 보안사고 발생 시 보고 및 대응 절차, 사고 복구조직의 구성, 교육계획 등을 포함한 보안사고 대응 계획을 수립
10	기업의 주요정보 유출방지를 위한 비밀유지서약서 미징구	직원으로부터 비밀유지 서약서에 서명을 받아야 하며, 임시직원이나 제3자에게 정보에 대한 접근권한을 부여할 경우에도 비밀유지 서약서를 서명을 그들로부터 받아야 할 필요가 있다.

또한 업무등과 관련된 내용을 일반 메일이나 기타 메신저등을 사용하여 통신을 하는 것에 있어서 보안사고를 통해 문제가 생겼을 경우 그에 대한 대책등이 거의 존재하지 않는다고 볼 수 있기 때문에, 안전하게 기밀 사항등도 외부로 유출될 걱정 없이 통신할 수 있도록 개발된 커뮤니케이션 수단이 필요하다고 파악하였다.

본 논문에서는 보안성 강화와 업무적 효율성간의 트레이드오프(trade-off) 관계형성에 있어서 최적화하기 위해 특정 공유기에서만 접근을 허용하고 그 외의 공유기를 소프트웨어 적으로 식별하여 공유기에 접속한 사용자에게만 서비스를 허가해주는 방식을 제안한다.

본 논문에서 제시하는 메신저 프로그램은 연결과정에 있어서 보안성을 유지하기 위하여 특정 AP에서만 접속이 가능하도록 접속을 제한한다. 또한, 접속 중단 시에 데이터를 서버에 저장하거나 단말기에 저장하지 않는다. 이러한 방법은 일반 통신 메신저처럼 사용하기에는 상용화에 있어서 불편한 점을 가지나 통신하는 내용이 외부로 유출될 가능성을 차단하고 보안성이 뛰어나다. 제안된 방식은 대기업, 공공기관 혹은 기밀 내용 등을 주고받는 기업들에게 있어서 효율적인 통신 방법이 될 수 있다.

Ⅱ. 제안하는 통신 방식

2.1 기존의 메신저 프로그램과의 차이점

기존의 메신저 프로그램의 대표적인 사례인 카카오톡과 비교하여 표 2에 정리하였다.

표 2. 기존의 메신저프로그램(카카오톡)과의 차이점.
Table 2. Differences with existing Messenger Program(Kakaotalk)

카카오톡	제안하는 프로그램
특정 AP에 제한 없이 프로그램 구동	특정 AP에서만 프로그램 구동
특정 기기에 대한 사용자 인증 후 인증 지속	프로그램 구동 시 사용자 인증
메시지를 읽기 전 3개월간 데이터 유지 및 채팅방에서 나가지 않을 경우 데이터보존	특정 AP와 연결이 종료되거나 프로그램 종료 시 데이터 삭제

기존의 카카오톡은 특정 AP에 제한 없이 프로그램이 구동 가능하고 특정 기기에 대한 사용자 인증 후에 인증을 지속하는 방식을 사용하고 데이터 저장을 위해서 메시지를 읽은 후에는 채팅방에서 나가지 않을 경우 대화 시에 사용가능한 텍스트나 이미지 등의 데이터가 계속 남아 있기 때문에 보안상의 우려가 있다.

본 논문에서의 메신저는 특정 AP에서만 프로그램의 구동을 제한하고 특정 AP와 연결이 끊어지거나 프로그램 종료 시대화시에 사용가능한 텍스트나 이미지 등의 데이터를 자동으로 삭제되게 하여 정보 유출을 막는다.

2-2. 프로그램 설계 구성

프로그램은 크게 4개의 과정으로 나눌 수 있다. 이 과정을 그림 1로 나타내었다.

그림 1. 프로그램 설계 구성.
Fig, 1. Composition of Program design.

첫 번째로 특정 AP식별을 하여 프로그램 실행의 유무를 결정하고 타이머를 통해 연결을 안정화한다.

두 번째로 사용자 인증을 하는 로그인 과정을 거치면서 데이터베이스를 통한 사용자 인증을 한다. 이 과정에서 무분별한 접속을 통해 불법적인 접속을 할 수도 있기 때문에 사용자 인증 횟수를 제한하는 방식을 사용하기로 하였다.

세 번째는 데이터 송·수신 과정을 거친다. 이 과정에서는 TCP Socket을 통한 메시지 및 사진 등 데이터 교환을 하는 방식을 사용한다.

마지막으로 AP의 연결이 끊어지거나 프로그램을 종료할 경우 데이터를 삭제하는 과정을 거친다. 이 과정에서 데이터 유출을 방지하기 위하여 메시지, 사진 등 데이터를 삭제하는 과정으로 프로그램을 마친다. [4][5]

3.3 AP식별 방식

AP식별은 WifiManager API(Application Program Interface)를 통하여 Wi-Fi 와 3G / 4G 망을 비교한 후에 Wi-Fi 사용할 경우 무선 AP의 SSID(Service Set Indentifier)를 획득하고 그 후 SSID를 통해 특정 AP를 식별한 후에 프로그램을 실행하는 방식으로 진행하였다. Wifimanager를 통해 비교 후 추출하는 소스코드는 다음 그림 2와 같다.

```
public void onTick(long millisUntilFinished) {
System.out.println(millisUntilFinished);

ConnectivityManager cm = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);

WifiManager wifimanager = (WifiManager)
getSystemService(Context.WIFI_SERVICE);

NetworkInfo activeNetwork =
cm.getActiveNetworkInfo();
String m_strName = new String();
String AP_match_SSID = new String();
AP_match_SSID = '"'+"security_ap"+'"'
int m_iNetworkType
m_iNetworkType = activeNetwork.getType();
m_strName =
wifimanager.getConnectionInfo().getSSID();
m_iNetworkType = activeNetwork.getType();
m_strName =
wifimanager.getConnectionInfo().getSSID();
if (m_iNetworkType ==
ConnectivityManager.TYPE_WIFI) {
if (m_strName.equals(AP_match_SSID)) {
timer();
}
}
```

그림 2. AP 식별을 위한 소스코드.
Fig. 2. Source code of AP indentification.

그림 3는 특정 AP가 연결이 되었을 경우와 연결이 되지 않은 경우를 비교하였다.[6]

그림 3. AP 연결 유무를 비교.
Fig, 3. Compare whether the AP connection.

그림 3과 같이 해당 AP가 아닐 경우 특정 AP를 식별할 수 없다는 오류를 Toast로 띄우게 되고 연결이 될 경우 해당 AP의 SSID 의 이름과 Type을 띄우고 그 다음 로그인 화면으로 넘어가게 된다.[6]

이러한 방식을 이용하여 AP를 구별할 경우 프로그램 접근을 차단하여 외부의 접근을 일차적으로 차단할 수 있다는 장점을 가진다.

3.4 회원관리 방식

특정 AP연결을 한 후에 로그인을 하는데 있어서 회원관리가 필요하다. 표 3에 회원관리에 필요한 테이블을 구성하였다.

표 3. 메신저 회원관리 테이블 구조.
Table 3. Structure of messenger member management table.

	PK	열	데이터유형	NULL?
1	PK	number	int(11)	NOT NULL
2		id	varchar(20)	NOT NULL
3		password	varchar(20)	NOT NULL
4		department	varchar(20)	
5		phone	varchar(20)	

회원가입을 할 때 각 회원마다 고유번호를 지정해주고 이 고유번호를 통해 회원을 관리하기 위해 PK(Private Key)로 지정한다. 그 후에는 id와 password 그리고 phone을 통해 회원관리에 따른 정보를 저장하게 하였고 각 직책마다 권한을 다르게 하기 위하여 직책도 입력하게 되었다. 그리고 NOT NULL 옵션을 준 변수들은 변수 값이 없을 경우 문제가 발생하기 때문에 필수 입력으로 지정하였다.[7]

여기서 안드로이드는 MySQL로 직접 접근을 허용하지 않는다. 그 이유는 데이터베이스에서 MySQL을 직접 사용하게 될 경우 보안상의 문제가 취약해지기 때문이다.. 그렇기 때문에 직접적인 SQL Query를 가져올 수 없다.

이런 이유로 본 논문에서는 PHP(Personal Hypertext Preprocessor)를 이용한 우회접근을 통해 Android에 SQL Query를 전송한다. 이 과정을 그림 4에 나타내었다.

그림 4. Android와 MySQL의 통신과정.
Fig, 4. Process of communication with Android and MySQL.

3.5 통신 연결 방식

통신 연결 방식은 ArrayList를 이용하여 현재 AP식별 과정을 통하여 프로그램에 접속되어 있는 사용자를 데이터베이스의 정보에 따라 ArrayList에 정보를 추가한다. 클라이언트에서는 새로고침을 통해 갱신을 하게 될 경우 현재 접속되어있는 사용자를 ArrayList를 통해 확인하여 통신을 할 수 있다는 것을 명시한다.

ArrayList에서는 id와 현재 수신을 하였지만 확인을 하지 않은 메시지의 개수를 보여주고, 사용자에 따른 해당하는 ArrayList의 ListItem에서는 각 사용자의 AP식별 과정에서 확인한 IP와 데이터베이스에서 가져온 id을 저장하고 있다.[7]

그림 5는 통신 과정을 위한 인터페이스이다.

그림 5. 메신저의 통신 연결 방식.
Fig, 5. Communication connection method of messenger.

그림 5를 통하여 현재 접속자의 수는 ArrayList의 ListItem의 개수를 Count하여 확인하였다. 왼쪽 상단에는 현재 자신이 접속하고 있는 id를 출력하였다.

아래 하단은 ArrayList에서 현재 접속되어 있는 사용자와 개수를 확인할 수 있도록 하였다.

또한 id옆에 '(1)' 과 같은 숫자를 통해 현재 자신에게 전송되어 있는 메시지의 개수 혹은 부재중 메시지에 대한 확인을 한눈에 볼 수 있도록 하였다.

3.6 메신저 통신 과정

그림 6의 인터페이스에서 특정 사용자를 선택하여 될 경우 메시지 수신이 없는 경우 클라이언트로 연결을 하고 메시지를 수신하였을 경우 서버로 연결하는 방식을 If Else구문으로 구분하여 각기 다른 클래스를 Intent하도록 설정하였다.[8]

이는 ListItem에 저장되어 있는 IP와 id를 비교하여 현재 내가 선택한 사용자가 내가 통신을 할 사용자인지 확인 한 후에 채팅방에 접속을 하도록 설정하였다.

서버와 클라이언트간의 통신은 소켓을 이용한 통신을 하였다. 메신저의 인터페이스는 다음 그림 6과 같다.

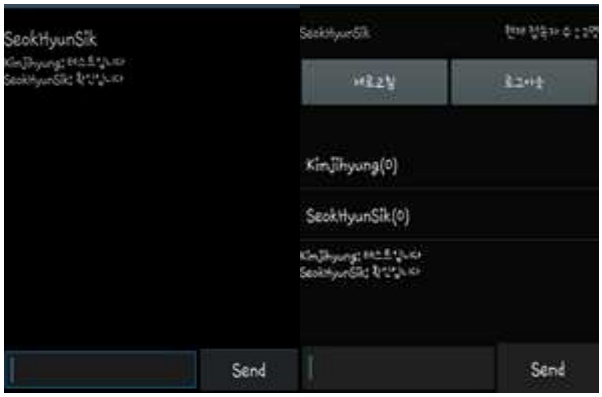

그림 6. 메신저의 클라이언트 및 서버
Fig 6. The Messenger of client and server.

이 과정에서 데이터를 전송하는 소스코드는 다음 그림 7과 같다.

```
start.setOnClickListener(new OnClickListener(){
        @Override
public void onClick(View V){
        if(et.getText().toString() != null){
   text.append(myname+":
"+et.getText().toString()+"\n");
        return_msg = et.getText().toString();
//newworkWriter의 형식은 BufferedWriter
        if(networkWriter != null) {
        try {
        networkWriter.write(return_msg);
        et.setText("");
        networkWriter.newLine();
        networkWriter.flush();
        } catch (IOException e) {
     text.setText("정상적으로 문자를 전송할
수없었습니다.");
```

그림 7. 데이터 전송을 위한 소스 코드
Fig, 7. Source code of data transform.

메신저에서는 현재 접속되어 있는 채팅방의 이름을 왼쪽 상단에 표기하여 자신이 어느 채팅방에 있는지를 먼저 알 수 있도록 명시하였다.

또한 Intent과정에서 Activity간의 데이터를 전달할 수 있도록 서버는 기존의 UserPage에서 ArrayList의 ListItem을 클릭하게 될 경우 그에 해당하는 채팅창이 활성화 되면서 클라이언트와 마찬가지로 채팅이 가능하도록 설정하였다.

화면의 뷰들은 메인스레드에서 연결되기 때문에 일반 스레드에서 생성된 스트림을 공유하도록 하였다.[9]

채팅방에서 전송버튼을 누르게 될 경우 데이터가 비어있지 않을 경우에는 OnClickListener를 통하여 Thread가 돌아가게 되고 EditText에 입력한 데이터를 특정 변수에 저장한 후에 BufferedWriter에 현재 변수에 저장되어 있는 변수를 입력하고 라인을 생성하여 Flush함수를 이용하여 추가한다. 만약 데이터가 입출력오류에 따라 전송이 되지 않을 경우 '정상적으로 문자를 전송할 수 없었습니다.' 라는 오류문장을 전송하도록 하였다.

메시지를 받는 입장에서는 메신저의 TextView에 BufferedWriter에 저장되어있는 값을 BufferedReader에 전달하여 출력하도록 하였다.[10]

여기서는 채팅방이 나간다 하더라도 데이터가 지워지지 않고 기존의 메신저 프로그램과 동일하게 채팅내용이 남아있고 이어서 채팅이 가능하도록 하였다.

3.7 통신 연결 종료 과정

본 프로그램에서는 통신 연결 종료과정을 크게 2가지로 두었다.

첫 번째로는 로그아웃을 통해 프로그램을 종료하는 방식을 사용하였다. 로그아웃을 누르게 될 경우 로그아웃을 요청한 사용자의 id와 ArrayList의 ListItem과 일치하는 항목을 가져온 후 ListItem을 삭제한 후에 데이터베이스에 접근하여 일치하는 id를 현재 접속 중인 페이지를 관리하는 테이블에서 삭제하는 방식을 통해 구현하였다.

두 번째로는 예외적으로 메신저 통신을 하던 도중 AP의 연결이 끊어지게 될 경우에는 프로그램의 데이터가 밖으로 유출될 가능성이 발생하기 때문에 바로 데이터를 삭제시키도록 하고 현재 데이터가 저장되어 있는 BufferedWriter와 BufferedReader를 close() 함수를 이용하여 비활성화 및 삭제를 시키고 socket을 null값으로 바꾸어 기존의 채팅 내용을 알 수 없도록 하였다.[11]

로그아웃버튼을 이용할 경우 그림 8과 같은 결과를 얻을 수 있다.[12]

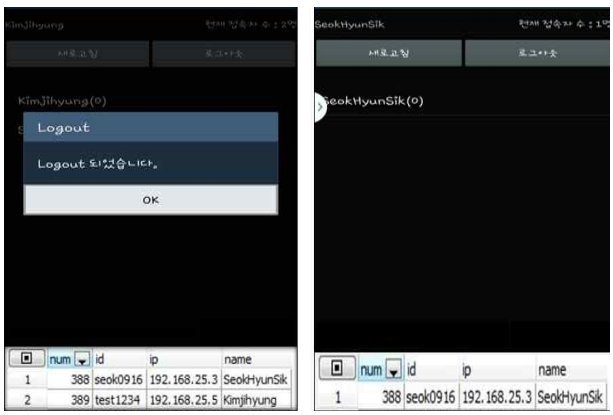

그림 8. 로그아웃 결과창.
Fig, 8. Result of Logout.

Ⅲ. 설계 및 실험결과

본 시스템은 특정 AP를 통한 보안체계 시스템을 구현하기 위하여 안드로이드 기반의 메신저에서 테스트 하였다. 메신저의 통신과정은 크게 3가지로 나누어 진행한다. 그림 9는 다음의 과정들을 도식화한 것이다.

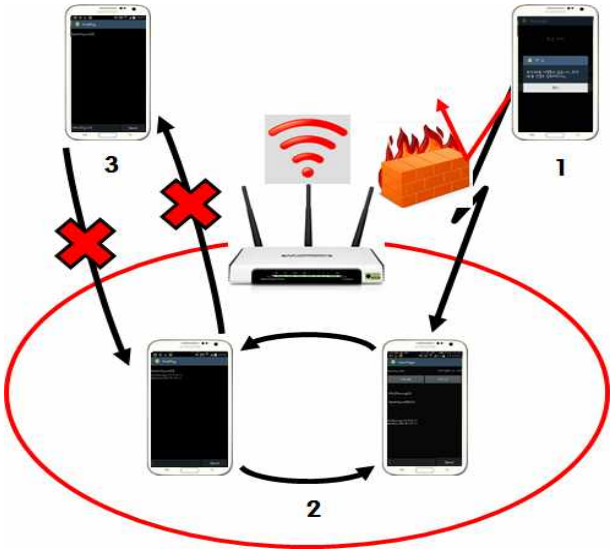

그림 9. 메신저의 통신 과정
Fig, 9. Communication process of messenger.

1번 과정은 클라이언트가 AP 를 통해 접속을 시도하려고 하는 경우 특정한 AP가 아니면 접속을 차단하고 프로그램을 종료시키는 것으로 일차적인 보안을 시행한다.

2번 과정은 접속한 메신저 사이에서 통신을 한다. 이 과정은 AP와의 연결이 되어있을 경우 그 다음 로그인 과정을 통해서 인증을 마친 클라이언트만 통신이 가능하다. AP가 연결되어 있을 경우 자체적인 보안은 무선 LAN 보안 프로토콜중의 하나인 WPA(Wi-Fi Proteced Access) 보안 프로토콜을 이용하여 불법적인 접속을 차단한다.

3번 과정은 통신이 끝나거나 AP와의 연결이 끊어질 경우에는 데이터를 자동으로 삭제하여 외부로 정보가 유출되는 것을 방지하여 보안체계를 완성하였다.

이러한 과정을 가지고 통신을 한 결과 외부에서 비정상적으로 접근하는 것에 대해 일차적으로 대응할 수 있다는 결과를 도출하였고 유효성을 검증하였다. 이 방법은 소켓을 통하여 통신을 하고 데이터를 서버에 저장하지 않아 통신 종료 후 데이터를 해킹당할 위험이 없으며, 통신 종료 후 자동으로 데이터를 프로그램에서 삭제하여 복잡하고 고비용인 기존의 보안체계와는 다르게 저비용 고효율의 효과를 나타내었고, AP를 교환하여 실험을 한 경우에도 동일한 실험결과를 도출하여 프로그램이 재사용성을 가지게 되었다.

Ⅳ. 결 론

본 논문에서는 안드로이드 플랫폼 기반 스마트폰에서 발생하는 개인 정보유출을 방지하는 적절한 방법을 제안하고 있다. 이 방법은 높은 보안성과 기존의 업무효율성간의 적절한 트레이드오프를 이루고 있다.

제안된 방법은 AP에 접근할 수 있는 사용자만으로 폐쇄된 메신저그룹을 유지하는 보안방식을 사용하고, 소프트웨어의 특징인 재사용성을 통해 저비용 고효율의 보안체계를 구축한다. 구체적으로는 Wifi manager API를 통해 일차적으로 외부의 접근을 차단하고, 다른 사용자와 통신을 할 경우, AP에서 제공되는 WPA 프로토콜을 이용하여 보안을 유지한다. AP와의 연결이 끊어지거나 사용자가 통신을 종료하면, 각 메신저 프로그램의 데이터를 모두 삭제하는 방식을 통하여 보안체계를 구현하였다. 본 방법은 서버에서 데이터를 보관하지 않고 각 단말기가 메신저 텍스트를 일시 저장하였다가 메신저를 종료하고 나가는 순간, 통신데이터를 삭제하는 방식을 채택하고 있다. 제안된 방법은 안드로이드 폰에서 구현하였고, 실험을 통하여 그 유효성을 검증하였다.

향후 연구과제는 다수의 인원이 통신할 수 있도록 프로그램상의 알고리즘을 추가하는 것과 보안성 강화를 위한 추가 알고리즘을 향상시키는 것이다.

참고문헌

[1] http://www.hankyung.com/news/app/newsview.php?aid=2014010388061, 카톡에 떠다니는 대한민국 정부, 2014.

[2] 김영훈, “모바일 보안을 위한 웹 애플리케이션 서비스의 세션 관리 개선 방안 연구”, 학위논문, 세종사이버대학교, 2015.

[3] http://www.kisa.or.kr, 기업의 관리적 보안 결함 TOP10, 2008.

[4] Belen Cruz Zapata, Android Studio Application Development, 에이콘 출판사, 2014.

[5] Joshua J. Drake, Android Hacker’s Handbook, whily, 2014.

[6] 박현재, 기적을 부르는 안드로이드 통신 프로그래밍 , 투에이치앤에스 출판사, 2013.

[7] 박현재, 안드로이드를 지배하는 통신 프로그래밍, 프리렉 출판사, 2011.

[8] Reto Meier, 프로페셔널 안드로이드2 애플리케이션 개발, 제이펍 출판사, 2010.

[9] 조정원, 안드로이드 모바일 악성코드와 모의 해킹 진단, 2014.

[10] 이두진, 안드로이드 앱 개발 완벽 가이드, PCBOOK , 2010.

[11] Andrew Hoog, 안드로이드 프로그래밍과 보안의 모든 것 세트, 에이콘 출판사, 2013.

[12] Sheran Gunasekera, 안드로이드 앱 보안, 길벗 출판사, 2013.

김 지 형 (Jihyung Kim)

2015년 2월 한밭대학교 정보통신공학전공 (학사)

2015년 3월~현재 : 한밭대학교 정보통신 전문대학원 정보통신공학과 석사과정

※주관심분야: 영상처리, 응용소프트웨어, 컴퓨터네트워크

이 충 호 (Choong Ho Lee)

正會員

1985년 2월 연세대학교 전자공학과(학사)

1987년 2월 연세대학교대학원 전자공학과 (석사)

1998년 3월 토호쿠대학대학원 정보과학 연구과 시스템정보과학전공(공학박사)

1987년 2월~ 2000년 2월 KT 멀티미디어연구소 전임연구원

2000년 2월 ~ 현재 한밭대학교 정보통신공학과 교수

※주관심분야 : 패턴인식, 신호처리, 응용소프트웨어