한국컴퓨터정보학회 동계학술대회 논문집 제19권 제1호 (2011. 1)

안드로이드 스마트폰의 데이터 수집 방법

안영건°, 김명호*
°*숭실대학교 컴퓨터학과
e-mail:bhythmmaker@gmail.com, kmh@ssu.ac.kr

Method of Collecting Data on Android Smartphone

An Young Geon°, Kim Myoung Ho*
°*Dept. of Computer Science, Soong Sil University

● 요 약 ●

스마트폰에 대한 관심이 높아지면서 스마트폰 사용자가 증가하고 있다. 스마트폰 중에서도 안드로이드 스마트폰의 증가세가 가장 두드러지며 현재 국내 스마트폰 시장에서 가장 많은 점유율을 차지하고 있다. 스마트폰은 전화기능 외에도 다양한 기능을 가지고 있고 다양한 애플리케이션을 실행할 수 있으며 이러한 특징은 사용자들로 하여금 많은 일을 스마트폰으로 처리하게 하였으며 많은 데이터들이 스마트폰에 저장되게 되었다. 본 논문에서는 이러한 데이터들을 수집하는 방법을 제안한다. 이러한 수집방법은 모바일 포렌식 분야에도 도움이 될 수 있으며 복잡해지는 정보들을 사용자가 관리하기 편하게 하는데도 도움이 될 수 있다. 본 논문에서는 논리적인 정보획득 방법에 기반을 두어 안드로이드 프레임워크의 컴포넌트중 하나인 컨텐트 프로바이더를 통해서 안드로이드 스마트폰의 기본 애플리케이션이 제공하는 정보들을 수집하는 방법으로 정보를 수집한다.

키워드: Android(Android), 데이터 수집(collecting data), 컨텐트 프로바이더(content provider)

I. 서 론

아이폰의 국내 출시 이후 스마트폰의 관심이 높아졌다. 이후 여러 제조사에서 다양한 안드로이드 스마트폰을 출시하면서 안드로이드 스마트폰에 대한 관심도 높아지고 있으며 사용자도 크게 증가하고 있다. 현재 국내 스마트폰 시장은 안드로이드 스마트폰이 가장 높은 점유율을 차지하고 있다.

스마트폰은 전화기능뿐만 아니라 다양한 애플리케이션을 실행할 수 있는 환경을 갖추고 있다. 이러한 애플리케이션을 이용해서 다양한 작업을 스마트폰에서 할 수 있다. 이러한 특징은 스마트폰에 다양한 데이터가 저장되도록 한다. 이러한 데이터에 대한 수집은 모바일 포렌식 분야에도 도움이 될 수 있으며 사용자의 정보관리에도 도움이 될 수 있다.

본 논문에서는 논리적인 정보획득 전략[1]을 바탕으로 안드로이드 프레임워크 컴포넌트중 하나인 컨텐트 프로바이더를 이용해서 정보를 수집하는 방법을 제안한다. 이 방법으로 안드로이드 스마트폰에 기본적으로 설치되는 애플리케이션에 의해서 저장되는 정보들을 수집할 수 있으며 수집한 정보는 PC를 통해서 확인할 수 있다.

II. 관련 연구

viaForensics의 운영자인 Andrew Hoog는 DFI News에서 다음의 네가지 안드로이드 포렌식 전략을 소개하였다.

- SD 카드 분석 : SD카드에 저장되는 데이터를 분석하는 방법이다. FAT32 파일시스템을 분석하는 방법을 사용한다.
- 논리적 분석 : 안드로이드 애플리케이션을 이용해서 분석하는 방법이다. 안드로이드 애플리케이션을 설치하고 실행하여 데이터를 수집하고 수집된 데이터는 SD 카드에 저장한다.
- 물리적 분석 : 리눅스의 dd 명령어를 이용해서 NAND 메모리의 덤프 이미지를 분석하는 방법이다. 루트권한이 필요하며 YAFFS2 파일시스템을 분석해야 한다.
- Chip-off : 플래쉬 메모리를 분리하여 하드웨어 적으로 데이터를 분석하는 방법이다.

본 논문에서는 위 4가지 방법 중에서 논리적 분석 방법을 바탕으로 해서 데이터를 수집하는 방법을 제안한다.

III. 데이터 수집 방법

1. 안드로이드

안드로이드 애플리케이션은 컨텐트 프로바이더를 이용해서 기본 애플리케이션이 제공하는 데이터를 수집한다.

안드로이드 시스템은 다른 애플리케이션의 데이터에 접근할 수 없도록 샌드박싱(sandboxing) 되어있다. 대신 안드로이드 시스템에는 다른 애플리케이션에 데이터를 제공할 수 있도록 하기 위해서 컨텐트 프로바이더를 이용한다. 대부분의 기본 애플리케이션은 컨텐트 프로바이더를 통해 데이터를 제공해주고 있다. 데이터의 요청은 URI를 이용해서 요청한다. 표 1은 본 논문에서 수집하는 정보와 URI이다. 대부분의 URI는 해당 클래스의 기호상수로 정의가 되어 있지만 그렇지 않은 경우는 직접 문자열을 입력해줘야 한다. 표 1에서 SMS, MMS, 일정의 경우는 문자열을 사용하였고 나머지 정보들은 기호상수를 사용하였다.

ContentResolver 클래스의 query 메소드나 Activity 클래스의 managedQuery 메소드의 인자로 표의 URI를 전달하면 해당 정보에 대한 Cursor 객체를 리턴 받는다. Cursor는 어떤 데이터에 대해서 순차적으로 접근할 수 있는 인터페이스를 제공한다. Cursor 객체를 이용해서 순차적으로 데이터를 SD 카드에 저장한다. 저장할 때는 CSV 포맷을 사용해서 문자열들을 콤마(,)로 구분해서 저장한다.

```
/*** 정보 추출을 위해서 다음과 같이 메소드를 사용 ***/
cursor = activity.managedQuery(정보 추출을 위한 URI, 추출할 컬럼 목록, null, null, 정렬 유형);
retVal = cursor.moveToFirst(); // 커서의 처음으로 이동 if(retVal)
  csvWriter = new CSVWriter(파일명); // CSV 파일 저장
```

그림 1. 정보추출을 위한 소스코드
Fig 1. Source code for extracting information

그림 1은 컨텐트 프로바이더에 데이터를 요청하는 소스코드의 예제이다. CSVWriter는 CSV 파일을 만들기 위해 작성된 클래스이다. 이러한 방식으로 URI를 인자로 넘겨서 정보를 수집한다.

표 1. 주요 정보들의 URI
Table 1. URI for main information

정보	URI
연락처	Contacts.CONTENT_URI Contacts.Data.CONTENT_URI
통화내역	Call.CONTENT_URI
SMS	“content://sms”
MMS	“content://mms”
이미지	Images.Media.EXTERNAL_CONTENT_URI
비디오	Video.Media.EXTERNAL_CONTENT_URI
오디오	Audio.Media.EXTERNAL_CONTENT_URI
브라우저	Browser.BOOKMARKS_URI
일정	“content://com.android.calendar/events”

표 1에 있는 정보 중 연락처와 MMS를 제외한 정보는 하나의 테이블의 형태로 구성되어 있기 때문에 한 번의 요청으로 정보를 가져올 수 있다. 하지만 연락처와 MMS의 경우는 여러 개의 테이블로 구성이 되어 있어서 두 번 이상의 요청을 통해서 정보를 가져와야 한다.

연락처의 경우 Contacts.CONTENT_URI를 통해서 연락처의 ID를 가져오고 가져온 ID를 이용해서 Contacts.Data.CONTENT_URI를 통해 각각의 정보를 요청해야 한다. 요청할 때 MIME_TYPE의 값을 지정해서 각각 다른 정보들을 가져온다. 각각의 MIME_TYPE은 표 2에 있는 클래스에 정의되어 있다. 해당 클래스들은 CommonDataKinds의 하위클래스이다.

표 2. 연락처 클래스
Table 2. Contacts classes

정보	클래스
전화번호	Phone
이메일	Email
메신저	Im
주소	StructurePostal
웹사이트	Website
그룹	GroupMembership Groups
조직	Organization
이벤트	Event
메모	Note
사진	Photo

MMS의 경우 “content://mms"를 통해서 메시지의 ID를 가져오고 ID를 이용해서 "content://mms/[ID]/address"로 요청을 하면 문자를 주고 받은 상대방의 전화번호를 알 수 있으며 "content://mms/[ID]/part"로 요청을 하면 문자의 메시지와 첨부파일을 가져올 수 있다.

2. PC

PC에서는 안드로이드 애플리케이션을 설치하고 실행한다. 그리고 안드로이드 애플리케이션이 작업을 완료하고 종료하면 SD 카드에 저장된 수집된 정보들을 가지고 와서 화면에 보여준다. 수집된 정보들은 CSV 형식으로 되어 있기 때문에 가져온 내용은 CSV 형식에 맞춰서 변환을 해준 뒤 화면에 보여준다. 그림 2는 전체 시스템의 설계를 나타낸다.

PC와 안드로이드간의 통신은 ADB(Android Debug Bridge)를 이용한다. ADB는 안드로이드 애플리케이션의 디버깅에 사용되는 프로그램으로 SDK를 통해서 제공된다. ADB는 커맨드라인을 통해서 안드로이드 시스템에 명령을 내리는 것이 가능하다. 안드로이드 애플리케이션 설치, 실행, 삭제와 데이터를 가져오고 제거하는 동작들은 모두 ADB를 통해서 실행시킨다.

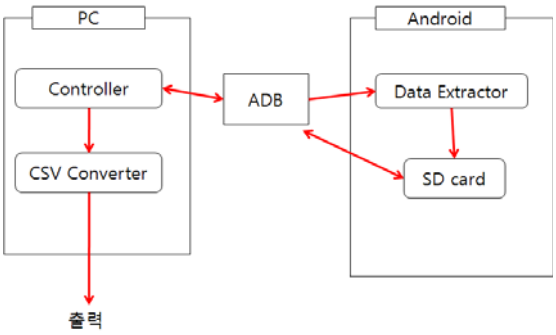

그림 2. 시스템 구성
Fig 2. System Architecture

PC에서 안드로이드 애플리케이션이 종료되었음을 확인할 때는 ADB와 파일을 이용한다. PC에서는 ADB를 이용해 SD 카드에 특정 파일이 생성되었는지를 주기적으로 확인한다. 안드로이드 애플리케이션은 작업이 완료되면 파일을 생성한다. 파일이 생성된 것을 확인함으로써 작업이 완료되었는지를 확인할 수 있도록 한다.

```
executeADBCommand(L"/c adb install ₩"Logical Forensics.apk₩" 2
〉result");  // 폰에 애플리케이션 설치
executeADBCommand(L"/c adb shell am start -a
android.intent.action.MAIN -n
kr.ac.ssu.logicalforensics/kr.ac.ssu.logicalforensics.activities.LogicalForen
sicsActivity");  // 애플리케이션 실행

/* 1초에 한 번씩 주기적으로 작업이 완료되었는지 검사 */
do{
    sleep(1000);
    executeADBCommand(L"/c adb pull /sdcard/extract_complete .");
}while(!CFile::GetStatus(L"extract complete", status));

/* CSV 파일 수신 */
executeADBCommand(L"/c adb pull /sdcard/sms.csv .");
```

그림 3. PC 애플리케이션 소스코드
Fig. 3. PC application source code

그림 3은 이러한 PC 애플리케이션의 동작에 대한 소스코드이다. ADB를 이용해서 안드로이드 애플리케이션을 설치 및 실행하고 1초에 한 번씩 파일이 생성되었는지 여부를 확인한다. 파일이 생성되었으면 ADB 명령을 이용해서 생성된 CSV 파일들을 가져온다.

Ⅳ. 결 론

스마트폰이 발전함에 따라서 스마트폰에 저장되는 정보들도 더욱 복잡해지고 다양해졌다. 스마트폰 사용자 또한 계속 증가하는 추세이고 특히 안드로이드 스마트폰의 사용자가 가장 가파르게 늘어나는 추세이며 시장 점유율 또한 가장 높은 상태다.

본 논문에서는 논리적인 정보획득 전략을 기반으로 한 데이터 수집 방법을 제안하였다. 이 방법은 안드로이드 안드로이드 애플리케이션을 이용해서 SD 카드에 정보를 수집하는 방법이다. 안드로이드 애플리케이션은 안드로이드 프레임워크의 컴포넌트중 하나인 컨텐트 프로바이더를 이용해서 안드로이드 스마트폰의 기본 애플리케이션이 제공하는 정보들을 수집한다. 이러한 동작들은 모두 ADB를 이용해서 PC에서 제어할 수 있도록 구현할 수 있다.

안드로이드 스마트폰의 데이터 수집 방법에 대한 연구는 모바일 포렌식 분야에도 도움이 될 수 있으며 갈수록 복잡해지는 데이터를 사용자가 쉽게 관리할 수 있도록 도와줄 수 있다. 본 논문에서는 아직 기본 애플리케이션이 제공하는 데이터만 수집할 수 있지만 더욱 많은 연구가 진행된다면 더욱 많은 데이터를 수집할 수 있을 것이고 이를 이용해서 훌륭한 관리 도구를 만드는 것도 가능할 것이다.

참고문헌

[1] Andrew Hoog, “An Introduction to Android Forensics”, http://goo.gl/0GDBO, DFI News