情報保護學會論文誌
第 22 卷 第 1 號, 2012. 2

안드로이드 로그 시스템을 이용한 효율적인 사용자 행위기반 라이브 증거수집 및 분석 시스템 연구

홍 일 영,† 이 상 진‡
고려대학교 정보보호대학원

Research on Efficient Live Evidence Analysis System Based on User Activity Using Android Logging System

Ilyoung Hong,† Sangjin Lee‡
Graduate School of Information Security, Korea University

요 약

최근 안드로이드 모바일 기기의 사용자가 증가함에 따라 디지털 포렌식 분야에서도 안드로이드에 대한 관심이 높아지고 있다. 하지만 해당 플랫폼 및 기기의 고유한 특성을 이용한 증거수집 및 분석에 대한 연구는 부족한 실정이다. 안드로이드 시스템의 특징 중 하나인 안드로이드 로그는 기기의 휘발성 저장매체로부터 수집될 수 있는 휘발성 데이터로서, 안드로이드 하부 시스템에서부터 애플리케이션에 이르기까지 최근의 모든 구동 내역과 관련한 기록이 저장되기 때문에 포렌식적으로 매우 중요한 정보가 될 수 있다. 본 논문에서는 안드로이드 로그의 고유한 특성을 사용자 행위와 연계하여 유의미한 정보를 이끌어내는 안드로이드 로그 수집・분석 시스템을 제안하고 나아가 로그 분석의 효율성을 극대화하여 현장에서 실시간 증거 분석이 가능하도록 해주는 효율적 분석 기법을 제안한다. 실험에서는 제안하는 로그 수집・분석 시스템을 이용함으로써 다양한 사용자 행위 정보를 수집하여 구체적이고 직관적으로 표현 가능함을 보이고, 제안하는 로그 분석 기법이 일반 정규식 검색 방법에 비해 10배 이상 검색시간을 단축함을 보인다.

ABSTRACT

Recently as the number of smartphone user is growing rapidly, android is also getting more interest in digital forensic. However, there is not enough research on digital data acquisition and analysis based on android platform's unique characteristics so far. Android system stores all the related recent systemwide logs from the system components to applications in volatile memory, and therefore, the logs can potentially serve as important evidences. In this paper, we propose a digital data acquisition and analysis system for android which extracts meaningful information based on the correlation of android logs and user activities from a device at runtime. We also present an efficient search scheme to facilitate realtime analysis on site. Finally, we demonstrate how the proposed system can be used to reconstruct the sequence of user activities in a more intuitive manner, and show that the proposed search scheme can reduce overall search and analysis time approximately 10 times shorter than the normal regular search method.

Keywords: Digital Forensic, Mobile Forensic, Android, Log analysis, Volatile Memory

접수일(2011년 8월 8일), 게재확정일(2011년 10월 7일)
† 주저자, hiy2009@korea.ac.kr
‡ 교신저자, sangjin@korea.ac.kr

I. 서 론

최근 모바일 기기가 급속하게 확산되면서 개인 정보의 흐름이 개인용 컴퓨터에서 모바일 기기로 이동하고 있다. 특히 안드로이드, iOS 등을 탑재한 스마트폰 혹은 태블릿 PC는 개인용 컴퓨터가 제공하던 웹서핑, 오피스, 멀티미디어 등 기능에 전화, MMS, 소셜 네트워크 기능이 강화되어 있고, 사용자가 항상 휴대할 수 있다는 장점 때문에 최근 몇 년간 사용자 수가 급증하였다. 이에 따라 수사에서도 모바일 기기에 저장된 데이터는 매우 중요한 증거로 인식되고 있으며, 디지털 포렌식 관점에서 모바일 기기에 대한 연구 역시 활발하게 진행되고 있다.

디지털 포렌식을 수행함에 있어서 운영체제가 활성화 되어 있는 시스템의 경우 휘발성 저장매체 상에 존재하는 데이터를 수집한 후에 비휘발성 저장매체 상에 저장되어 있는 데이터를 수집하는 순서로 진행하는 것이 일반적인 절차이다[2,8,10,11]. 모바일 기기에 대한 포렌식 수행에 있어서도 이와 같이 휘발성 데이터 수집을 선행한 후에 비휘발성 데이터를 수집하는 것이 타당하지만[2,15] 모바일 휘발성 데이터에 대한 연구는 활발하지 못한 상태이다.

휘발성 데이터 중에서도 안드로이드 로그는 플랫폼 하부 시스템으로부터 응용프로그램에 이르기까지 안드로이드 기기 상에서 일어난 상세한 로그를 기록하고 있어 사용자의 행위를 추적하는 데 매우 유용한 자료로 활용될 수 있다. 특히 비휘발성 메모리에는 기록되지 않는 특정 파일 접근 및 재생 시각, UI 조작 시각, 외장 메모리 포맷 시각 등 보다 상세한 정보를 안드로이드 로그를 통해서 확인할 수 있다.

또한 안드로이드 로그는 커널상의 로그 버퍼에 기록되어 사용자에 의해 임의적으로 조작될 수 없기 때문에 신뢰성이 높은 정보이다. 나아가 로그 수집 및 분석이 현장에서 실시간으로 가능할 경우 로그의 활용 가치는 더욱 큰 의미를 지닐 수가 있다. 예를 들어 사용자가 안드로이드 모바일 기기에 장착되어 있던 외장 저장 매체를 분리한 경우 로그를 확인하여 분석하면 최근 저장 매체가 분리되었음을 파악할 수 있고 이로써 누락될 수 있었던 해당 매체를 증거로 확보할 수 있을 것이다.

하지만 안드로이드 로그를 현장에서 분석하는 것은 쉽지 않다. 왜냐하면 안드로이드 시스템은 사용자가 특정 행위를 했을 때 뿐만 아니라 사용자의 행위가 없는 상태에서도 배터리 변화나 스크린 상태 변화와 관련한 로그 등 매우 다양한 로그 메시지를 다량으로 발생시키므로 사용자의 행위와 관련된 로그만을 가려내어 해석하는 데는 많은 시간이 소요되기 때문이다. 특히, 특정한 사용자의 행위에 대해서 단일 로그 엔트리로는 분석이 불가능하여 여러 로그 엔트리를 조합해야 분석이 가능한 경우에는 분석이 더욱 어려워 질 수 있다. 이러한 경우 여러 엔트리를 동시에 검색하는 다중 엔트리 정규식으로 검색이 가능하지만 다중 엔트리 정규식 검색은 단일 엔트리 검색에 비하여 상당히 많은 검색 시간이 소요되기 때문에 현장에서 수많은 검색어를 실시간으로 검색하기 위해서는 효율적인 분석방법이 마련되어야 한다.

본 논문에서는 안드로이드 로그를 이용한 증거 수집· 분석 시스템을 제안하고, 안드로이드 로그 분석에 사용되는 효율적인 분석기법을 제안한다. 증거 수집· 분석 시스템은 검색식 데이터베이스 생성, 증거 로그 획득, 로그 분석, 분석 결과 시각화 단계로 구성된다. 수집된 로그는 로그 정제 및 로그 분할 과정을 거쳐 정규식 검색을 수행하는 로그분석기법을 통해서 분석된다.

본 논문의 구성은 다음과 같다. 2장에서는 안드로이드 모바일 포렌식과 관련한 기존의 연구에 대하여 설명한다. 3장에서는 안드로이드 로그 시스템에 대해서 분석하고, 4장에서는 안드로이드 로그 수집· 분석 시스템 및 분석 기법을 제안한다. 마지막으로 5장에서는 실제 안드로이드 기기에 제안하는 시스템 및 기법을 적용하여 테스트한 결과를 제시한다.

II. 관련 연구

안드로이드 포렌식에 있어서 비휘발성 데이터 수집과 관련하여서는 ADB, Nandroid 백업, 모바일 포렌식 프로그램 등을 이용한 포렌식 기법[4], FTK Imager를 이용한 외장 메모리 이미지 획득, 루팅(Rooting) 및 리눅스 dd 명령을 통한 내장 메모리 획득 · 분석 기법[9], 여러 형태로 저장되는 위치 정보 수집 기법[12], 안드로이드 SQLite 데이터베이스 파일 분석 기법[1] 등 비교적 다양한 연구가 진행되어 왔으나 휘발성 데이터 포렌식에 대한 연구는 상대적으로 적게 진행되고 있다.

안드로이드 휘발성 데이터 수집과 관련한 연구로서 X. Lee[16] 등은 포렌식 프로그램이 내장된 포렌식 SD 카드를 이용하여 IMEI, 구동 중인 프로세스, 배터리 상태, 메모리 상태, WiFi MAC 주소와 같은 휘

발성 데이터도 수집 가능함을 보였다.

D. Mohindra[7]는 안드로이드 모바일 기기와 관련한 침해대응 및 포렌식 방법에 대하여 포괄적으로 소개하면서 파일 시스템 상의 데이터 뿐만 아니라 내장 명령어들을 통하여 시스템 시간, 시스템 uptime, 구동 중인 프로세스, 네트워크 접속 정보, IP 주소와 같은 휘발성 정보들을 수집 가능함을 보였다. 또한 저자는 안드로이드 로그시스템에 접근하여 안드로이드 로그를 추출해 주는 내장 프로그램인 로그캣 혹은 Dumpstate를 이용하여 휘발성 로그 데이터를 가져올 수 있음을 소개하고 있으나 구체적인 방안에 대해서는 언급하지 않고 있다.

V. Thing[14]는 실시간으로 애플리케이션 간 통신 데이터를 획득하여 분석하는 기법을 제안하였다. 실험 시나리오에서는 메시지 길이, 메시지 전송 간격, 메모리 덤프 간격, 메시지 작성 시간에 편차를 두고 메시지를 송수신한 후 각각의 시나리오 수행 시 마다 휘발성 메모리를 덤프하여 덤프된 메모리에 송수신 메시지가 기록되어 있는지 여부를 조사하였다. 이 연구는 사용자의 특정 애플리케이션 구동 행위에 따라 발생하는 휘발성 메모리 상태를 수집하여 증거로 활용할 수 있음을 제시한 데에 의미를 가지고 있으나 애플리케이션 간 송수신 메시지에 국한된 연구에 그치고 있다.

최근에는 시스템 활성 상태 안드로이드 기기로부터의 휘발성 데이터 수집에 대한 관심이 높아지고 있으나[3,13], 안드로이드 로그 분석을 통해 사용자 행위를 구체화하는 정도의 연구는 이루어지지 않고 있다.

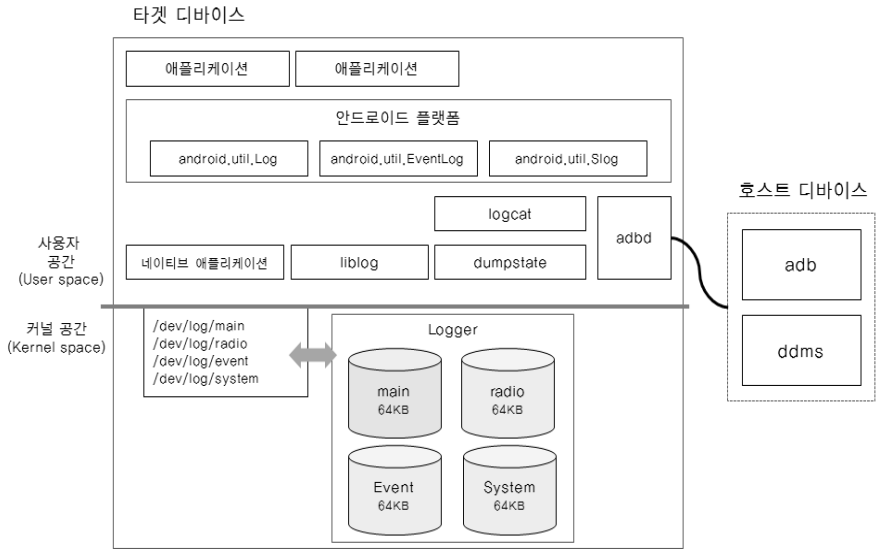
III. 안드로이드 로그 시스템 분석

안드로이드 로그 시스템은 플랫폼 하부 시스템으로부터 응용프로그램까지 시스템 전반적으로 로그 기록을 남길 수 있도록 설계되어 있다. 이러한 로그는 'dmesg' 혹은 '/proc/kmsg'를 이용하여 접근할 수 있는 리눅스 커널 로그와는 구분된다. 본 장에서는 안드로이드 로그 시스템에 대해서 소개하고 포렌식 관점에서 로그 시스템을 분석한다.

3.1 안드로이드 로그 시스템

안드로이드 로그 시스템은 다음과 같이 4가지 구성요소로 이루어진다[6].

- 로그 메시지 저장을 위한 커널 드라이버와 커널 버퍼
- 로그 엔트리 생성과 로그 메시지 접근을 위한 C, C++, Java 클래스 라이브러리
- 커널버퍼로부터 로그메시지를 읽어오는 프로그램(logcat)
- 타겟 디바이스로부터 로그메시지를 확인하고 필터링하는 기능



[그림 1] 안드로이드 로그 시스템

〔그림 1〕과 같이 안드로이드 커널 공간에는 main, radio, event, system 네 종류의 로그 버퍼들이 있으며, 로거(Logger)로 불리는 커널 드라이버가 이를 관리하고 있다. 유저공간의 애플리케이션은 커널의 버퍼에 저장된 로그들에 직접 접근할 수 없기 때문에 안드로이드 시스템은 /dev 디렉토리에 리눅스 디바이스 노드들을 제공하여 애플리케이션이 로그를 읽고 쓸 수 있도록 하고 있다.

또한, 안드로이드 시스템은 liblog라는 네이티브 라이브러리에서 디바이스 노드를 읽고 쓸 수 있는 API들을 제공하고 있다. liblog는 시스템 전체에서 사용할 수 있으며, C로 작성된 네이티브 애플리케이션 혹은 라이브러리에서 사용된다. Dalvik 가상머신 상에서 동작하는 자바로 작성된 플랫폼 및 애플리케이션들은 플랫폼 로그, 즉, android.util.Log, android.util.EventLog 및 android.util.Slog 클래스들을 통해서 로그시스템에 접근 가능하며 내부적으로는 liblog를 사용하게 된다. 즉, 안드로이드 플랫폼은 제조사 및 애플리케이션 개발자들에게 플랫폼 로그 및 liblog를 제공함으로써 자신들이 개발한 기능이 정상 동작하는지 확인할 수 있도록 하고 있다.

로그를 저장하는 것과 반대로 증거수집에 사용되는 타겟 디바이스에서의 로그 수집은 두 가지 방법을 사용할 수 있다. 첫 번째는 /dev/log에서 제공하는 디바이스 노드들을 읽는 하위레벨의 방법이다. 이 경우 데이터는 바이너리 형태로 추출이 된다. 두 번째 방법은 안드로이드에서 제공하는 네이티브 프로그램인 logcat을 이용하는 방법이 있다. logcat은 다양한 쿼리를 인자로 하여 원하는 로그를 선별적으로 획득할 수 있는 기능을 제공하며, 일반적으로는 로그시각, 로그태그, 로그메시지 등을 텍스트 형태로 출력하여 준다. 하지만 logcat은 안드로이드 단말 내부에서만 실행되는 프로그램이며 호스트에서 logcat을 실행하기 위해서는 ADB를 이용하여 접속한 후에 사용해야 한다. 안드로이드에서는 개발자들의 디버깅을 위해 상기의 기능들을 사용자 인터페이스를 이용하여 편리하게 확인할 수 있도록 호스트에서 실행 가능한 ddms라는 툴을 제공하고 있다.

마지막으로 안드로이드 내부의 모든 로그와 메모리 상황, 프로세스 현황, 프로그램 설치 정보 등의 모든 로그들을 한번에 출력해주는 dumpstate라는 프로그램 또한 단말에 탑재되어 있다.

3.2 안드로이드 로그 분류

3.2.1 로그 저장 위치에 따른 분류

안드로이드 로그 시스템은 로그 저장 위치 또는 커널 로거에 대응되는 디바이스 노드에 따라 〔표 1〕과 같이 4가지 종류로 분류할 수 있다. 마켓 및 인터넷으로부터 다운로드하여 설치하는 애플리케이션의 경우 대부분 main 로그 버퍼에 로그를 기록하며, 제조사에서 기록하는 로그는 system 로그 버퍼에 기록된다. 또한, 배터리 상태 등 플랫폼 내부의 제한적인 정보는 events 로그 버퍼에 저장된다. 하지만 모든 안드로이드 시스템이 동일하게 main, event, radio, system 로그 버퍼를 제공하는 것은 아니다. 제조사별 모델별로 메모리 크기 및 필요에 따라 제공되는 로그 시스템이 제거 또는 추가될 수 있다.

3.2.2 로그 생성원에 따른 분류

로그 생성원에 따라 안드로이드 로그 시스템은 다음과 같이 안드로이드 기본 로그, 제조사 로그, 애플리케이션 로그로 분류될 수 있다.

- 안드로이드 기본 로그 : 안드로이드 플랫폼 내부에 내장된 로그로써, 제조사가 수정하지 않는 한 모든 안드로이드 디바이스에 동일하게 나타나는 로그
- 제조사 로그 : 안드로이드 플랫폼을 하드웨어에

〔표 1〕 로그저장 위치에 따른 분류

종류	내용
main	메인 애플리케이션 로그. 애플리케이션 혹은 플랫폼 내부에서 android.util.Log 클래스를 이용하여 기록하는 로그로서, 포렌식적으로 유의미한 로그는 대부분 main에 담겨져 있음
event	시스템 이벤트 정보를 위한 로그. 애플리케이션 혹은 플랫폼 내부에서 android.util.EventLog 클래스를 이용하여 기록한 로그. 로그 엔트리는 바이너리 형태의 파라미터와 태그 코드로 이루어져 있으며, 태그 코드는 시스템의 /system/etc/event-log-tags에 저장됨.
radio	이동통신망 접속 관련 이벤트 정보
system	안드로이드 플랫폼 내부의 하위 레벨 시스템 메시지와 디버깅을 위한 로그.

탑재하여 출시하는 제조사는 안드로이드 기본 플랫폼 외에 추가로 탑재한 소프트웨어의 버그를 수정할 목적 등으로 로그를 기록함. 제조사 로그에는 제조사에서 수정한 플랫폼 로그와 기기 출시시 탑재한 애플리케이션의 로그가 있음.

- Application 로그 : 상기 2가지 생성원을 제외한 마켓 혹은 인터넷으로부터 다운받아 설치하는 애플리케이션이 생성하는 로그

상기와 같은 로그 시스템 분류는 4장에서 제안하는 시스템에서의 검색식 데이터베이스의 기본 단위가 된다.

3.3 안드로이드 로그 포맷

main, radio, system은 텍스트 메시지로 구성되는 로그 버퍼이며, event는 바이너리로 구성되는 로그 버퍼이다. 따라서 event 로그의 경우 작은 크기로 저장되지만, 로그 버퍼로부터 로그를 읽어 들이거나 이벤트 스트링을 디코딩할 때 추가적인 처리 과정이 필요하다. 각 엔트리는 기본적으로 〔표 2〕와 같이 메시지 페이로드 길이, 패딩, 프로세스 ID, 쓰레드 ID, 타임스탬프, 메시지의 내용으로 구성된다.

다만 로그 엔트리의 구조 및 메모리 상에 저장될 수 있는 엔트리의 최대 개수는 제조사별로 상이할 수 있다.

로그캣 명령을 수행할 경우 안드로이드의 로그 엔트리는 텍스트 형태로 가공되어 출력된다. 이 때 각 로그의 내용은 해당 메시지가 발생한 시스템 혹은 애

〔표 2〕 로그 엔트리 구조체

```
struct logger_entry {
  __u16    len;    /* length of the payload */
  __u16    __pad;  /* 2 bytes of padding */
  __s32    pid;    /* generating process's pid */
  __s32    tid;    /* generating process's tid */
  __s32    sec;    /* seconds since Epoch */
  __s32    nsec;   /* nanoseconds */
  char     msg[0]; /* the entry's payload */
};
```

플리케이션을 나타내는 태그, 타임스탬프, 메시지 로그의 레벨 혹은 메시지에 의해 표현되는 이벤트의 우선 순위, 로그 메시지 자체를 표시하는 텍스트로 구성된다.

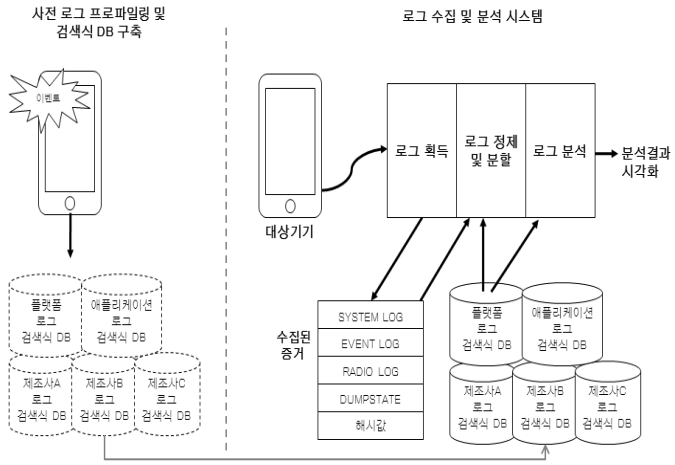
IV. 안드로이드 로그 획득 및 분석 시스템

제안하는 시스템은 〔그림 2〕와 같이 검색식 데이터베이스, 로그 획득 모듈, 로그 정제 및 분할 모듈, 로그 분석 모듈, 분석 결과 시각화 모듈로 구성된다.

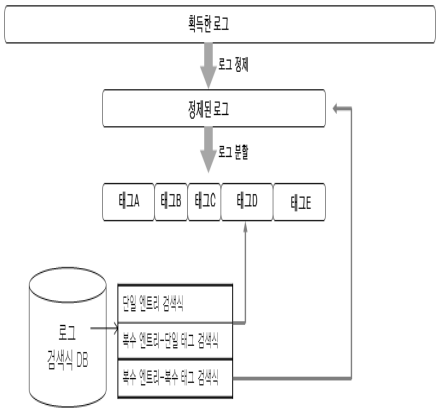
4.1 사용자 행위 기반 로그 패턴 프로파일링 및 데이터베이스 구축

4.1.1 사용자 행위의 정의

안드로이드 기기 사용자의 행위 중 포렌식적으로 유의미하게 해석될 수 있는 행위들을 아래와 같이 정의할 수 있다.



〔그림 2〕 시스템 아키텍쳐



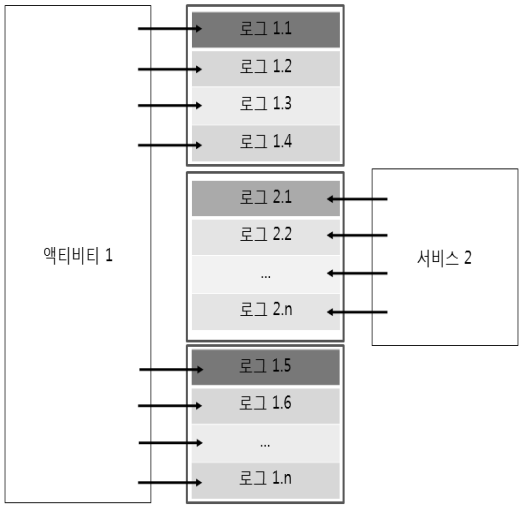
〔그림 3〕 로그 정제, 분할 및 분석

- 부팅 및 망접속 행위 : 부팅, WIFI 접속, GPS 접속, 네트워크 서비스 사업자 확인 등
- 통화 관련 행위 : 다이얼 패드 누르기, 음성 통화 송수신, 영상 통화 송수신, 최근 통화 목록 삭제 등
- SMS/MMS 관련 행위 : 메시지 송수신, 메시지 목록 삭제 등
- 이메일 관련 행위 : 이메일 목록 확인, 이메일 작성, 이메일 발송, 이메일 회신, 이메일 전달, 이메일 삭제 등
- 지도 관련 행위 : 지도 프로그램 실행, 위치 검색, 즐겨 찾기 목록 삭제, 최근 검색 목록 삭제 등
- 인터넷 브라우저 관련 행위 : URL 입력, 사이트 방문, 검색어 입력 등
- 외장 SD 카드 관련 : 외장 SD 카드 삽입, 외장 SD 카드 제거 등
- 카메라 관련 행위 : 카메라 실행, 사진 촬영, 사진 삭제 등
- 파일 관련 행위 : 파일 다운 로드, 파일 실행, 파일 삭제 등
- 프로그램 관리 관련 행위 : 프로그램 다운 로드, 프로그램 업데이트, 프로그램 삭제 등

이 외에도 다양한 애플리케이션 사용 행위들이 로그 패턴 프로파일링을 위해 포함될 수 있다.

4.1.2 사용자 행위에 따른 로그 패턴 프로파일링 고려사항

사용자의 안드로이드 기기 사용 행위는 기기 상에서 애플리케이션의 구성요소인 액티비티, 서비스, 콘텐트 프로바이더, 브로드캐스트 리시버들의 생명주기로 표현된다[5]. 안드로이드 로그는 애플리케이션의 생명주기 진행 과정 중에서 개발자가 사전에 설정해 둔 로그 발생의 시점에 발생된다. 따라서 사용자가 안드로이드 기기 상에 취한 1개의 행위도 여러 개의 로그를 발생시킬 수 있고, 반대로 로그가 전혀 남지 않을 수도 있다. 또한 사용자가 한 애플리케이션의 생명주기의 시작부터 종료에 이르는 전 과정을 기다리지 않고, 다른 애플리케이션을 시작하는 것과 같이 여러 애플리케이션을 중복적으로 실행할 수도 있으며, 기기의 사용자가 특별한 행동을 하지 않아도 안드로이드 디바이스는 [그림 4]와 같이 여러 애플리케이션의 액티비티나 서비스에 의해 발생된 로그를 끊임없이



[그림 4] 애플리케이션과 로그

생성하게 된다. 따라서 방대한 로그 속에서 사용자가 특정 행위를 취하였을 때의 로그를 가려내어 분석할 필요가 있다.

또한 사용자의 행위를 추적하기 위해 로그를 분석하는 방법에 있어서도 단일 로그 엔트리만으로도 간단히 사용자의 행위를 파악할 수 있는 경우도 있으나, 여러 로그 엔트리를 조합하여야만 하는 경우가 존재한

[표 3] 복수 엔트리

07-06 20:41:32.811 V/WindowManager(2516): Delivering toWindow{47d89ed8 com.android.contacts/com.android.contacts.PhoneContactsActivity paused=false}
07-06 20:41:32.815 I/LogsListActivity(2671): fillData id:298, cursor.getCount() : 1
07-06 20:41:32.815 I/LogsListActivity(2671): fillData LogsId:298, strNumber:01234567890
07-06 20:41:35.155 I/LogsListActivity(2671): onContextItemSelected CONTEXTMENU_REMOVELOG Position:1, id:298, mLogCnt:161, c.getCount():161
07-06 20:41:35.155 I/LogsProvider(2706): delete selection : _id=?
07-06 20:41:35.155 I/LogsProvider(2706): delete match : 20
07-06 20:41:35.213 I/LogsProvider(2706): delete count : 1
07-06 20:41:35.221 I/LogsListActivity(2671): CONTEXTMENU_REMOVELOG first log
07-06 20:41:35.221 I/LogsListActivity(2671): CONTEXTMENU_REMOVELOG COLUMN_IDX_SEP:1
07-06 20:41:35.221 I/LogsListActivity(2671): CONTEXTMENU_REMOVELOG next mHighLightId:297

다. 예를 들면 사용자가 안드로이드 맵 애플리케이션을 구동하여 지도 검색에 검색어를 입력하는 행위를 한 경우 단일 로그 엔트리를 통해서도 확인이 가능하다. 그러나 사용자가 특정 통화 목록을 지우는 행위에 대해서는 단일 로그만을 분석하여서는 사용자의 행위를 탐지하기 어렵다. 이 경우 〔표 3〕과 같이, 1) 전화 애플리케이션 실행 2) 최근 통화 목록 탭 터치 3) 삭제 대상 목록 터치 4) 삭제 버튼 터치 5) Database 삭제 순서로 진행되는 연속적인 로그 엔트리가 기록된다. 이러한 경우 하나의 로그 엔트리만을 검색할 경우 오탐지 확률이 높아지기 때문에 복수의 로그 엔트리를 함께 조합하여 분석하여야만 해당 행위를 정확히 탐지할 수 있다.

이와 같은 안드로이드 애플리케이션과 로그의 특성으로 인하여 신속성이 요구되는 현장에서 로그를 분석하여 그 결과를 즉각 이용하는 것은 매우 어려운 일이다. 따라서 신속하고 정확한 분석의 수행을 위해서는 사용자 행위가 발생하였을 때 생성되는 로그 엔트리를 단일 혹은 그룹 단위로 패턴을 파악하여 사전에 프로파일링 하는 과정이 필요하다.

4.1.3 데이터베이스 구축

사전 프로파일링을 통해 사용자의 행위를 파악할 수 있는 로그의 패턴이 확정되면, 해당 패턴을 정규 검색식으로 표현하여 데이터베이스를 구축한다.

데이터베이스 구축시 검색식은 〔그림 3〕과 같이 세 가지의 형식으로 분류되는데, 우선 단일한 엔트리 내에서 검색이 가능한 '단일 엔트리 검색식'이 있으며, 동일한 태그를 가진 여러 개의 엔트리를 조합하여 검색을 수행하여야 하는 '복수 엔트리-단일 태그 검색식'이 있고, 마지막으로 다양한 태그를 가진 여러 개의 엔트리를 조합하여 검색을 수행하여야 하는 '복수 엔트리-복수 태그 검색식'이 있다.

5장에서는 실험을 통해 실제 구축된 검색식과 이를 통해 분석을 수행한 결과를 제시한다.

4.2 로그 획득

안드로이드 기기에서 로그 메시지를 수집하기 위해서는 커널 내의 로그 버퍼에 접근해야 하므로 일반 유저공간의 프로그램을 통해서는 접근이 불가능하며 파일시스템 내의 /dev/log에 위치하고 있는 디바이스 노드들을 통해서만 접근할 수 있다. 하지만 루팅을 하지 않는 이상 접근권한의 제약으로 디바이스 노드들을 읽어오는 것은 어려우며 안드로이드에서 제공하는 로그캣을 이용하는 다음과 같은 명령으로 로그 버퍼의 내용을 텍스트로 추출할 수 있다.

- 메인로그 : # logcat -b main -b system -v threadtime -d *:v
- 이벤트로그 : # logcat -b events -v threadtime -d *:v
- 무선통신로그 : # logcat -b radio -v threadtime -d *:v

현장에서 수집된 로그는 해시값을 계산하여 소유자 등에게 확인을 받고 목록을 교부하여 획득 단계를 마무리한다.

4.3 로그 정제, 분할, 분석

4.3.1 로그 정제

로그 획득단계에서 수집된 로그는 배터리 상태 변화 정보와 같이 포렌식적으로 무의미하고 지속적인 로그들이 매우 많은 양을 차지하기 때문에 한 번의 정제 과정을 거쳐 크기를 줄이게 된다. 정제 과정을 거쳐 로그 크기가 감소하면 이후의 검색단계에서 정규식 검색 시간도 줄어든다. 정제 과정에서는 기기별 정제 데이터베이스 키워드를 이용하여 로그를 검색한 후 매칭되는 라인을 삭제하는 간단한 과정을 거친다.

4.3.2 로그 분할

로그 분할 단계에서는 〔그림 3〕과 같이 정제된 로그를 로그태그 별로 분할한다. 이 과정은 사전에 정의된 태그별로 로그를 분리하며 정의되지 않은 태그를 가진 로그는 하나의 그룹으로 모아서 분리한다. 분할 작업은 약간의 오버헤드를 필요로 하지만 향후 검색단계에서 검색시간을 획기적으로 줄여 준다.

4.3.3 로그 분석

로그 분석 단계에서는 획득한 로그를 대상으로 데이터베이스에 저장된 검색식을 이용해 정규식 검색을 수행한다. 플랫폼 로그는 모델에 따라서 검색식의 양이 한정적이지만 애플리케이션 로그의 경우 매우 다양

하고, 향후에도 지속적으로 늘어날 수 밖에 없다. 따라서 현장에서 즉각적인 대응을 하기 위해서는 검색의 정확성만큼 검색의 효율성이 중요하다.

제안하는 시스템에서의 로그 분석 과정은 〔그림 3〕에 표현한 것과 같다. 먼저, 검색식 데이터베이스로부터 검색식 하나를 선택하여 로그에 대하여 검색을 수행한다. 이 때 선택된 검색식이 4.1.3에서 설명한 세 가지 형식의 분류 중 어느 것에 속하느냐에 따라 검색 대상이 되는 로그 그룹이 달라진다. 만일 선택된 검색식이 단일 엔트리 검색식 혹은 복수 엔트리-단일 태그 검색식인 경우 로그 분할에 의해 분류된 태그별 로그 그룹에서 해당 태그 그룹을 찾아 검색을 수행하게 된다. 선택된 검색식이 복수 엔트리-복수 태그 검색식인 경우에는 정제 후 분할되지 않은 상태의 전체 로그를 대상으로 검색을 수행하게 된다.

4.1.2절에서 소개한 바와 같이 단일 엔트리만으로 행위의 내용을 정확히 판별하기 어려운 경우가 존재하기 때문에 여러 로그 엔트리를 조합하여 특정 행위가 있었음을 확인하기 위해서 복수 엔트리 검색이 필요하다. 그러나 복수 엔트리 검색을 할 경우 검색 속도가 급격히 느려지는 문제가 발생한다. 이러한 문제를 해결하기 위해 복수 엔트리 로그로 구성되는 검색식이 동일한 태그를 사용하는 경우와 다양한 태그를 사용하는 경우를 구분하였다. 동일한 태그를 사용하는 복수 엔트리 검색식의 검색을 해당 태그의 로그 그룹 내에서만 수행할 경우 검색하여야 하는 엔트리 개수가 현저히 줄어들게 되므로 속도가 향상된다. 단일 엔트리 검색식의 경우에도 전체 로그를 대상으로 검색을 수행할 때보다 태그별 로그 그룹을 대상으로 검색할 때 검색 속도가 빨라진다.

따라서 위와 같은 검색식의 분류 및 태그별 로그 그룹의 분할을 통한 검색은 분석의 정확성과 검색 속도를 비약적으로 높여주어 분석 효율성을 증가 시킨다.

V. 실험 결과

본 장에서는 4장에서 제안한 시스템을 이용해 실제 안드로이드 기기 상에서 실험한 결과를 제시한다.

〔표 4〕 실험 환경

CPU	Intel Core2Duo 2.4
OS	Windows7
개발환경	Java SDK 6.0 기반
정규식	Java SDK 6.0 내장
대상단말	안드로이드 2.3 기반

5.1 실험 환경

실험에 사용된 단말과 분석이 수행된 기기의 정보는 〔표 4〕와 같다.

5.2 검색식 데이터베이스 생성

기기 상에서 사전에 정의된 행위를 발생시켰을 때 발생되는 로그의 메시지를 분석하여 프로파일링을 위한 검색식을 〔표 5〕와 같이 선정하고 데이터베이스로 구축하였다. 선정된 검색식은 4장에서 설명한 기준에 의해 단일 엔트리, 복수 엔트리-단일 태그, 복수 엔트리-복수 태그로 분류하였다.

실험 대상 기기에서 부팅 완료, WIFI 접속 IP, 3G 통신망 접속, GPS 연결, 이메일 회신, 이메일 전달, GPS 현재 위치 탐색, 지도 검색, 인터넷 브라우저 URL 입력, 인터넷 사이트 방문 등 관련 로그 엔트리는 단일 엔트리 검색을 통해서 분석이 가능하였다. 그러나 통화 수신, 통화 발신, 통화 목록 삭제, 메시지 전송, 메시지 수신 등과 관련한 검색은 복수 엔트리의 조합을 통하여야 분석할 수 있었다.

5.3 로그 획득, 정제, 분할

안드로이드 기기를 USB 케이블을 통해 분석용 컴퓨터에 연결하고 로그캣 명령을 실행시켜 메인로그, 이벤트로그, 무선통신로그를 텍스트 형태로 획득하였다. 획득된 로그에 대하여는 해쉬값을 산출하여 보존하였다.

이어 획득된 로그에 대하여 사전에 구축된 정제용 데이터베이스를 이용하여 불필요한 로그 엔트리들을 제거하는 과정을 거쳐 정제하였다. 정제의 대상은 사용자의 행위와 무관하게 발생하는 로그들로 선정하였는데, 주로 BatteryService, PowerManagerService, dalvikvm, Zygote 등의 태그를 가진 로그 및 각종 에러메시지 로그가 이에 해당되었다.

실험에 사용된 안드로이드 기기에서 아무것도 실행하지 않고 방치하는 Idle 단말 테스트, 특정 시간 동안 임의의 조작을 수행하는 Random 테스트, 주요 행위 연속 실행 테스트 등 50회 실험을 통해서 획득된 로그에서 정제 후 평균 60%로 엔트리 수가 감소하는 것을 확인하였다.

정제를 거친 로그에 대하여는 검색식 분류에 따른 로그 검색 대상 그룹을 나누기 위해 태그별 로그 분할

〔표 5〕 실험에 사용된 검색식 데이터베이스

분류	행위유형	행위내용	태그	정규 검색식
단일 엔트리	부팅	부팅	Setting	BOOT_COMPLETED
단일 엔트리	WIFI 접속	WIFI 접속	WifiStateTracker	IP configuration: ipaddr.*gateway.*netmask
단일 엔트리	통신망 연결	통신망 연결	GsmServiceState Tracker	Poll ServiceState done
단일 엔트리	GPS 연결	GPS연결	libgps	HSLP host ip
단일 엔트리	다이얼 누르기	다이얼 누르기	DialerActivity	keyCode value in keyPressed
복수 엔트리 -단일 태그	통화발신	발신	PhoneUtils	placeCall
		전화걸기 시도	PhoneUtils	incoming: false state: DIALING
		통화종료	PhoneUtils	incoming: false state: DISCONNECTED
복수 엔트리 -단일 태그	통화수신	전화 수신 요청 신호	PhoneUtils	startGetCallerInfo: number
		전화 응답	PhoneUtils	answerCall: call state = INCOMING
		통화종료	PhoneUtils	connection: incoming: true state: DISCONNECTED
복수 엔트리 -단일 태그	통화목록 삭제	통화 기록 메뉴	LogsListActivity	bindview onTouch mPrevLogType:100, mLogType100
		통화 기록 선택	LogsListActivity	fillData LogsId:.*strNumber
		선택된 목록삭제	LogsListActivity	onContextItemSelected CONTEXTMENU_REMOVELOG
복수 엔트리 -복수 태그	메시지 전송	메시지 전송	ConversationViewGroup	Send SMS
		수신자 번호, 전송된 메시지	LogsProvider	insert data - logtype : 300.*number :.*contactid :.*name :.*messageid :
복수 엔트리 -복수 태그	메시지 수신	메시지 수신	System.out	TEXT SMS
		메시지 내용	System.out	000000
		발신자 번호	SmsReceiver Service	notifyMms:sFrom
단일 엔트리	이메일 목록	이메일 목록	Email	MsgControl.*time.*from.*contactid.*messageid.*m_subject
복수 엔트리 -단일 태그	이메일 확인	목록 중 특정 이메일 선택	Email	MessageList.*onItemClick
		첨부 파일	Email	MessageView.*attach file name
			Email	EmailContent.*restore.*name
복수 엔트리 -단일 태그	이메일 작성	이메일 작성	Email	Compose.*MsgCompose:LogProvider.*time.*number.*messageid.*m_subject
		이메일 전송	Email	SOCKETSmtpSender.*open::smtpSendOK
단일 엔트리	이메일 회신	이메일 회신 버튼 선택	ActivityManager	com.android.email.intent.action.REPLY.*cmp=com.android.email.*MessageCompose
단일 엔트리	이메일 전달	이메일 전달 버튼 선택	ActivityManager	com.android.email.intent.action.FORWARD.*cmp.*com.android.email.*MessageCompose
단일 엔트리	GPS	현재 위치	libgps	sec_gps_inject_location.*latitude.*longitude
단일 엔트리	지도	지도 검색	SearchDialog	com.google.android.maps.MapsActivity.*S.query=
단일 엔트리	인터넷 브라우저	URL 입력	browser	value for url
단일 엔트리	인터넷 브라우저	사이트 방문	ActivityManager	android.intent.category.BROWSABLE
복수 엔트리 -복수 태그	SD카드 삽입	SD카드 삽입	MountService	/mnt/sdcard/external_sd disk inserted
		SD카드 스캔	MediaScanner	MediaScanner.*Start Nazca.*/mnt/sdcard/external_sd
		파티션 크기	/system/bin/newfs_msdos	/dev/block/vold/.*sectors in
복수 엔트리 -단일 태그	SD카드 포맷	포맷 중	Vold	Volume.*sdcard.*/mnt/sdcard/external_sd state.*Formatting
		디스크 사이즈	Vold	Volume::getDiskInfo -> disk_size
		포맷 완료	Vold	Filesystem formatted OK
단일 엔트리	SD카드 제거	SD카드 제거	DirectVolume	Volume sdcard.*/mnt/sdcard/external_sd.*removed

작업을 수행하였다. 로그 분할의 기준이 되는 태그의 종류는 〔표 5〕의 검색식 데이터베이스 상의 검색어들과 관련한 태그들로 선정되었다.

5.4 로그 분석 및 결과 가시화

로그 분석은 4.3.3에서 설명한 바와 같이 검색식 분류별로 검색 대상 그룹을 다르게 선택하여 검색하는 방식으로 수행하였다. 실험 결과 사전에 정의된 사용자의 각 행위와 관련한 로그가 검색식을 통해 완전하게 검출되는 것을 확인할 수 있었다.

분석 결과는 〔표 6〕과 같이 시간, 행위 유형, 행위 내용, 상세 정보로 구성하여 분석자가 안드로이드 기기 사용자의 행위를 직관적으로 인식할 수 있도록 자동화된 프로그램을 통해 가시화 하였다. 특히 상세 정보에서는 각 로그로부터 통화 수·발신자 전화 번호, 메시지 송·수신자 번호, 수신 메시지 내용, GPS 현재 위치(위도·경도), 삭제된 통화 목록, 지도 검색어, 인터넷 방문 사이트 URL, SD 카드 내의 폴더 목록, 이메일 작성 시각, 첨부된 파일명 등 관련 내용을 표시하였다. 이상의 상세 정보에 표시되는 내용은 사용자의 행위 내용을 더욱 구체적으로 확인할 수 있는 정보로서 검색식을 통해 검출된 로그 엔트리 내의 내용을 분석자가 직관적으로 인식할 수 있는 형태로 가공하여 출력한 것이다.

예를 들면 사용자가 다이얼패드를 누를 때 기록되는 로그 엔트리에는 각 버튼에 따라 정해져 있는 기호(번호)가 저장되는데 이러한 기호는 실제의 번호와는 다르기 때문에 로그 메시지만을 보고 어떠한 번호가 눌러진 것인지 알기 어렵다. 상세정보에는 분석자의 가독성을 높이기 위해 이러한 기호를 실제 누른 버튼의 번호로 대치하여 표시함으로써 로그 분석의 효용성을 높일 수 있게 된다.

제안 시스템을 사용하기 전 후의 분석 성능에 대한 비교 결과는 다음 절에서 제시한다.

5.5 효율성 분석

검색 시간에 영향을 미치는 요인은 검색 대상이 되는 로그 엔트리의 개수와 검색어 데이터베이스 내의 검색어 개수가 있다. 이 중 검색 대상이 되는 로그 엔트리의 개수와 관련하여 정제 과정은 불필요한 로그를 제거함으로써 로그 엔트리의 절대적 개수를 감소시키므로 검색 시간을 단축한다. 또한 태그별 분할 과정을 거친 이후 해당 태그 그룹 내에서 검색을 수행하게 되면 해당 태그 그룹 내의 엔트리 개수 내에서 검색이 이루어지게 되므로 검색 시간이 단축된다. 실험 결과 정제 후에는 정제 전 총 엔트리의 약40% 가량이 소거되었다. 정제 후 로그 엔트리를 태그별 그룹으로 나눌 경우, 검색 대상이 되는 각 태그 그룹별 평균 엔트리 개수는 정제 후 총 엔트리 개수의 1%로 감소된다. 예를 들면 총 엔트리 개수가 12,000개인 경우 정제를 거치게 되면 약 7,200개의 엔트리가 남게 되고, 이를 다시 태그별 그룹으로 엔트리들을 구분하게 되면 태그 그룹 내에서의 평균 엔트리 수는 약 72개가 된다.

본 실험에서는 검색대상 로그 엔트리의 수를 달리하면서 검색 속도를 측정하여 제안 분석 기법과 일반 정규식 검색 간의 수행 속도를 비교하였다. 〔그림 5(좌)〕와 〔표 7〕은 검색 대상 엔트리 수를 4,000에서 20,000까지 증가시키면서 단일 엔트리 정규식 1,000개, 다중 엔트리-단일 태그 정규식 500개, 다중 엔트리-다중 태그 정규식 100개를 검색하는 속도를 각각 측정한 결과이다. 일반 정규식 검색을 수행할 경우 다중 엔트리 검색 수행 속도가 단일 엔트리 검색에 비하여 100배 이상 차이가 나는 것을 볼 수 있다.

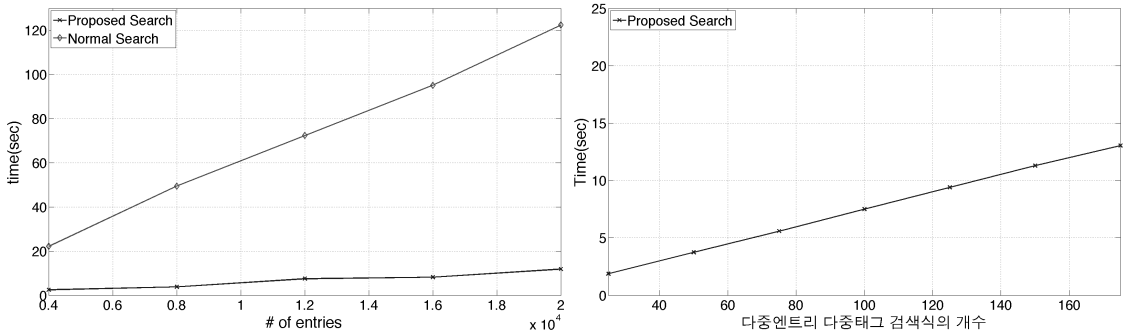
제안 기법을 적용할 경우 일반 정규식 검색에 비하여 단일 엔트리 정규식 검색 속도는 평균 143배 가량, 다중 엔트리-단일 태그 검색속도는 평균 15,631배 가량, 다중 엔트리-다중 태그 검색속도는 평균 1.8배 가량 빨라졌다.

다만 제안하는 기법을 적용하면 정제 및 분할 작업을 수행하게 되므로 정제 및 분할 작업에 소요되는 시간만큼 수행 시간이 늘어나게 된다. 그러나 20,000개의 엔트리를 정제하여 분할하는 데 0.04초 미만의 시간이 소요될 정도이므로 전반적으로 속도 저하에 크게 영향을 미치지는 않는다고 볼 수 있다.

이 외에 본 실험에서는 검색 데이터베이스 내의 검색식 개수가 분석 속도에 미치는 영향을 확인하고자 제안하는 분석 기법을 적용한 상태에서 검색식 데이터베이스 내의 검색식 개수를 달리하면서 검색 속도를 측정하였다. 파라미터로 적용한 검색식 개수는 실제 기기 상에서 수집할 수 있었던 검색어 개수 이상으로 적용하였는데, 〔표 8〕과 같이 단일 엔트리 검색식의 개수를 250개에서 1,750개 까지, 다중 엔트리-단일 태그 검색식의 개수를 125개에서 875개까지, 다중 엔트리-다중 태그 검색식의 개수를 25개에서 175개까지 2배씩 증가시켜가면서 검색을 수행하였다. 실험 결과 〔그림 5(우)〕 및 〔표 8〕과 같이 단일 엔트리 검색

〔표 6〕 분석 결과

시간	행위유형	행위내용	상세 정보
07-30 16:59:22.015	부팅	부팅 완료	
07-30 16:59:23.456	WIFI로 접속한 IP	WIFI 접속	**IP configuration** ipaddr 192.168.123.xxx gateway 192.168.123.xxx netmask 255.255.255.0 dns1 203.xxx.162.xxx dns2 219.xxx.0.1 DHCP server 192.168.xxx.xxx
07-30 16:59:30.671	통신망 연결	통신망 연결	SKTelecom 45005 UMTS CSS
07-30 16:59:45.882	GPS 연결	GPS연결	HSLP host ip : 74.XXX.127.XXX
07-30 17:13:11.100	다이얼 누르기	다이얼 누르기	**keyPressed : 0**번
07-30 17:13:11.636			**keyPressed : 1**번
07-30 17:13:11.995			**keyPressed : 2**번
07-30 17:13:12.417			**keyPressed : 3**번
07-30 17:13:16.229	통화발신	발신 대상 번호	01234567890
07-30 17:13:16.249		전화걸기 시도	
07-30 17:13:26.886		통화종료	
07-30 17:22:12.729	통화수신	전화 수신 요청 신호	CallerInfo : 010XXXXxxxx
07-30 17:32:01.097		전화 응답	
07-30 17:32:08.54		통화종료	
07-30 17:40:00.047	통화목록 삭제	통화 기록 메뉴 보기	
07-30 17:40:00.054		통화 기록 선택	선택된 목록 LogsId : 312 전화 번호 : 01123456890
07-30 17:40:02.710		선택된 목록삭제	
07-30 17:41:30.100	메시지 전송	메시지 전송	
07-30 17:41:30.835		수신자 번호, 전송된 메시지 ID	수신자 번호 : 1234567890 messageid : 3092
07-30 17:42:31.499	메시지 수신	메시지 수신	
07-30 17:42:31.499		메시지 내용	Hello.This is android test.
07-30 17:42:31.780		발신자 번호	010xxxxXXXX
07-30 17:43:35.792	이메일 목록	이메일 목록	time : 1309310065000(UNIX TIME) from : gXXX@nXXe.com contactid : 2 messageid : 163 m_subject : This is E-mail Title
07-30 17:43:04.981	이메일 확인	목록 중 특정 이메일 선택	MessageList onItemClick : 6
07-30 17:43:06.190		첨부 파일	EmailContent : android.jpg
07-30 17:43:06.240			attach file name : android.jpg | image/pjpeg
07-30 17:44:14.554	이메일 작성	이메일 작성	time : 1310190467835(UNIX TIME) number : XXXXX@dXXX.net contactid : 2 messageid : 169 m_subject : This is E-mail Title2
07-30 17:44:25.690		이메일 전송	
07-30 17:45:07.784	이메일 회신	이메일 회신 버튼 선택	
07-30 17:47:58.686	이메일 전달	이메일 전달 버튼 선택	
07-30 17:50:45.592	GPS	GPS 현재 위치	강남역 latitude 37.497844 longitude 127.027531 accuracy 59.000000
07-30 17:51:46.967	지도	지도 검색	query=%EA%B0%95%EB%82%A8%EC%97%AD 강남역
07-30 17:52:43.342	인터넷 브라우저	URL 입력	http://www.daum.net
07-30 17:52:49.666	인터넷 브라우저	사이트 방문	http://m.media.daum.net/media/sisa/newsview/20110709092017566
07-30 18:53:27.585	외장 SD 카드 삽입	외장 SD 카드 삽입	
07-30 18:53:30.554		SD 카드 내의 폴더 스캔	/mnt/sdcard/external_sd/Test_Folder1
07-30 18:53:30.589		파일시스템, 파티션 크기	/dev/block/vold/179:9: 15517952 sectors in 242468 FAT32 clusters (32768 bytes/cluster)
07-30 18:57:00.022	외장 SD 카드 제거	외장 SD 카드 제거	/mnt/sdcard/external_sd partition 179:9 removed



〔그림 5〕 일반 검색과 제안하는 검색 기법의 성능(좌)과 검색식 개수에 따른 검색시간(우)

〔표 7〕 일반 검색과 제안하는 검색 기법의 성능 비교
(다중엔트리 다중태그 검색식 개수=100, 다중엔트리 단일태그 검색식 개수=500, 단일엔트리 검색식 개수=1,000)

검색 기법	총 엔트리수	정제시간 (sec)	다중엔트리 다중태그 검색시간 (sec)	다중엔트리 단일태그 검색시간 (sec)	단일 엔트리 검색시간 (sec)	총 검색시간 (sec)
일반검색 기법	4,000	-	3.703	18.088	0.312	22.103
	8,000	-	8.147	40.636	0.622	49.405
	12,000	-	12.004	59.425	0.913	72.342
	16,000	-	15.574	78.255	1.248	95.077
	20,000	-	20.183	100.59	1.635	122.408
제안하는 기법	4,000	0.007	2.454	0.001	0.002	2.464
	8,000	0.011	3.774	0.002	0.005	3.792
	12,000	0.023	7.468	0.004	0.007	7.502
	16,000	0.033	8.153	0.005	0.008	8.199
	20,000	0.036	11.811	0.007	0.011	11.865

〔표 8〕 검색식 개수에 따른 검색시간

다중엔트리 다중태그 검색식 수	다중엔트리 단일태그 검색식 수	단일엔트리 검색식 수	정제시간 (sec)	다중엔트리 다중태그 검색시간 (sec)	다중엔트리 단일태그 검색시간 (sec)	단일 엔트리 검색시간 (sec)	총 검색시간 (sec)
25	125	250	0.281	1.879	0	0.001	2.161
50	250	500	0.281	3.743	0.002	0.003	4.029
75	375	750	0.278	5.592	0.003	0.005	5.878
100	500	1000	0.284	7.512	0.004	0.005	7.805
125	625	1250	0.283	9.407	0.004	0.007	9.701
150	750	1500	0.281	11.291	0.005	0.009	11.586
175	875	1750	0.283	13.047	0.007	0.010	13.347

은 0.001초에서 0.01초까지, 다중 엔트리-단일 태그 검색은 0.00001초에서 0.007초까지, 다중 엔트리-다중 태그 검색은 1.879초에서 13.047초까지 수행 시간이 선형적으로 증가하는 것을 확인할 수 있었다.

실험 결과를 종합하여 보면 제안하는 기법을 적용할 경우 일반 검색에 비하여 10.7배 가량 빠른 속도의 검색 성능을 얻을 수 있음을 알 수 있다. 또한 검색식 데이터베이스의 검색어 개수 증가에 따른 검색 속도 상승치가 선형적 증가를 보이고 있음을 볼 때, 향후 실제 상황에서 검색식의 개수가 증가할 경우에도 현장에서 적용 가능한 수준의 성능을 얻을 수 있을 것으로 판단된다.

VI. 결 론

안드로이드 시스템에서 획득할 수 있는 휘발성 데이터 중에서 안드로이드 로그에는 사용자의 최근 기기 사용 내역을 추적할 수 있는 다양한 정보들이 저장되지만 이에 대한 연구는 거의 이루어지지 않았다. 본 논문에서는 안드로이드 로그를 수집하여 현장에서 즉각적으로 활용할 수 있도록 효율성을 높인 안드로이드 로그 수집·분석 시스템을 제안하였다.

실험에서는 안드로이드 로그를 분석하여 사용자의 최근 행위 과정 및 구체적 관련 정보들을 분석자가 쉽게 활용할 수 있도록 가시화 하였다. 이러한 정보 중에는 기존 비휘발성 데이터만을 이용하는 포렌식 기법을 통해서는 파악하기 어려운 다양한 정보들이 포함되어 있어 안드로이드 로그의 활용 가치가 상당함을 알 수 있다.

또한 제안한 분석 기법을 적용할 경우 일반 정규식 검색에 비해 10배 이상 검색 속도를 향상시킬 수 있었다. 검색식 개수 증가에 따른 검색 수행 시간 상승치 실험 결과를 볼 때, 향후 검색식 데이터베이스가 증가한다 하여도 현장 적용 가능한 정도의 속도를 얻을 수 있을 것으로 기대된다.

다만 제안하는 안드로이드 로그 시스템은 안드로이드 기기 및 애플리케이션별로 로그 검색식 데이터베이스를 구축하여야 하는 단점을 지니고 있어, 안드로이드 기기 및 애플리케이션의 업데이트에 따라 검색식 데이터베이스도 지속적으로 증가하게 된다. 향후 이러한 단점을 보완하여 다양한 안드로이드 기기 및 애플리케이션에 일반적으로 적용될 수 있는 검색식 생성 및 검색 기법에 대해서 연구하고자 한다.

참고문헌

〔1〕 구본민, 김주영, 이태림, 신상욱, "Android & iOS 기반 스마트폰의 디지털 증거 수집 및 분석", 정보보호학회논문지, 21(1), pp. 167-175, 2011년 2월.

〔2〕 ACPO, "Good Practice Guide for Computer-Based Electronic Evidence", 2007.

〔3〕 A. Case, "Forensic Memory Analysis of Android's Dalvik Virtual Machine", Source Conference, http://www.slideshare.net/SOURCEConference/forensic-memory-analysis-of-androids-dalvik-virtual-machine, Jun. 2011.

〔4〕 A. Hoog, "Android forensics", Mobile Forensics World 2009, http://www.scribd.com/doc/64763914/MFW2009-HOOG-AndroidForensics, May 2009.

〔5〕 "Android Application Component", http://developer.android.com/guide/topics/fundamentals.html

〔6〕 "Android Logging System", http://elinux.org/Android_Logging_System

〔7〕 D. Mohindra, "Android, Incident Response and Forensics", http://www1.webng.com/dhruv/material/android_report.pdf, 2008.

〔8〕 D. Brezinski and T. Killalea, " Guidelines for Evidence Collection and Archiving", RFC 3227, Feb. 2002.

〔9〕 J. Lessard and G.C. Kessler, "Android Forensics:Simplifying Cell Phone Examinations", Small Scale Digital Device Forensics Journal, Vol. 4, No.1, ISSN# 1941-6164, Sep. 2010.

〔10〕 NIJ, "Electronic Crime Scene Investigation:A Guide for the First Responders, Second Edition", pp. 26, Apr. 2008.

〔11〕 SWGDE, "Best Practices for Computer Forensics", pp. 3-4, Jul. 2006.

〔12〕 S. Maus, H. Höofken and M. Schuba, "Forensic Analysis of Geodata in Android Smartphones", International Conference on Cybercrime, Security and Digital Forensics, http://www.schuba.fh-aachen.de/papers/11-cyberforensics.pdf, Jun. 2011.

〔13〕 T. Maguire, "IR Process & Smart Phones", The 2011 Digital Forensics and Incident Response Summit, https://files.sans.org/summit/forensics11/PDFs/IR%20%20Smart%20Phones.pdf, Jun, 2011.

〔14〕 V.L.L. Thing, Kian-Yong Ng and Ee-Chien Chang, "Live memory forensics of mobile phones", Digital Investigation 7, pp. 74-82, Aug. 2010.

〔15〕 W. Jansen and R. Ayers, "Guidelines on Cell Phone Forensics", NIST, pp. 34, 2007.

〔16〕 X. Lee, "Design and Implementation of Forensic System in Android Smart Phone", The 5th Jointed Workshop on Information Security, http://crypto.nknu.edu.tw/publications/2010JWIS_Android.pdf, Aug. 2010.

〈著者紹介〉

홍 일 영 (Ilyoung Hong) 정회원
2000년 2월: 경북대학교 사회학과 졸업
2009년 3월~현재: 고려대학교 정보보호대학원 석사 과정
2006년 2월~현재: 대검찰청 디지털수사담당관실
〈관심분야〉 디지털 포렌식, 모바일 포렌식, 클라우드 컴퓨팅, 사이버 법률

이 상 진 (Sangjin Lee) 정회원
1987년 2월: 고려대학교 수학과 이학사
1989년 2월: 고려대학교 수학과 이학석사
1994년 2월: 고려대학교 수학과 이학박사
1989년 2월~1999년 2월: 한국전자통신연구원 선임연구원
1999년 2월~2001년 8월: 고려대학교 정보보호대학원 교수
2001년 9월~현재: 고려대학교 정보보호대학원 교수
〈관심분야〉 컴퓨터 포렌식, 모바일 포렌식, 심층 암호, 해쉬 함수