ISSN : 1976-5304
https://kdfs.jams.or.kr

안드로이드 에뮬레이터에 대한 포렌식 기법

남 우 환
서울지방경찰청 (경장)

Android Emulators Forensic Analysis Technique

Woohwan Nam
Seoul Metropolitan Police Agency (Senior Policeman)

요 약

안드로이드 에뮬레이터는 Windows, MacOS 등 PC 운영체제에서 안드로이드 환경을 구축하여 모바일 앱 실행을 가능하게 해주며, 가상 안드로이드 환경에서 루팅, 스크린샷, GPS 위치 변경, 단말기식별번호(IMEI) 값 수정과 같은 각종 설정 변경 등 다양한 기능을 제공한다. 그러나 안드로이드 에뮬레이터의 기능을 오용하거나 범죄에 악용하는 사례가 발생하고 있다. 본 논문에서는 주요 안드로이드 에뮬레이터별 획득 가능한 포렌식 아티팩트들을 분석하고, 이를 바탕으로 안드로이드 에뮬레이터 포렌식 절차를 제시한다. 또한, 실제 범죄사건 사례를 통하여 안드로이드 포렌식 기법의 활용 방법을 제안한다.

주제어 : 안드로이드 에뮬레이터, 앱 플레이어, 블루스택, 녹스, 디지털 포렌식

ABSTRACT

Android emulator allows to run mobile apps by building an Android environment in PC operating systems such as Windows and MacOS. In addition, it provides various functions and settings such as rooting, screenshots, changing GPS location, and IMEI(International Mobile Equipment Identity) value modification in virtual Android environment. However, there are cases of misusing these functions or using it for crime. This paper analyzes the forensics artifacts that can be obtained for major Android emulators and presents the forensic process. Lastly, this paper suggests how to use Android forensic techniques in criminal cases.

Key Words : Android Emulators, App Players, BlueStacks, Nox, Digital Forensics

Ⅰ. 서 론

2008년 구글에서 발표한 안드로이드는 현재까지 스마트폰, 태블릿 컴퓨터, PDA 등 다양한 기기에 적용되면서 대표적인 모바일 운영체제가 되었다. 2007년 애플의 아이폰 출시를 기점으로 스마트폰과 태블릿의 대중화가 시작되었지만, 그보다 더 후발주자인 안드로이드가 오픈소스의 이점을 내세워 다양한 하드웨어 회사를 공략함으로써 모바일 운영체제 시장에 성공적으로 진출하였다. 실제로 2019년 7월 기준 스탯카운터(Statcounter)의 세계 모바일 운영체제 점유율 발표에 따르면 안드로이드가 76.08%, iOS가 22.01%를 점유하고 있는 것으로 나타났다[1].

안드로이드 에뮬레이터(android emulator: app player)는 Windows, MacOS 등 PC 운영체제에서 안드로이드 운영체제 환경을 구축하여 모바일 앱의 실행을 가능하게 해주는 소프트웨어이다. 오라클 버추얼박스(Oracle VirtualBox)와 같은 내장된 가상화 프로그램을 이용하여 .vmdk 파일 등 안드로이드 운영체제가 설치된 가상디스크 파일을 불러와 구동시킴으로써 안드로이드 환경을 PC에서 이용할 수 있도록 해준다. 안드로이드 에뮬레이터 출시 초기에는 다양한 모바일 환경에서 개발된 앱의 호환성 테스트와 소프트웨어 디버깅 목적으로 이용되었으나, 모바일 게임 시장의 급속한 성장과 더불어 정교한 조작과 고품질 그래픽을 필요로 하는 모바일 게임 출시가 늘어나면서 키보드 인터페이스와 고성능 PC 환경에서 모바일 게임을 구동하고자 하는 사용자가 급증하기 시작했다. 실제로 2012년 세계 게임시장에서 모바일 게임의 비중은 18%에 불과하였지만, 모바일 기기의 급속한 보급으로 2019년에는 54%까지 성장하였다[2].

▪ Received 18 October 2019, Revised 30 October 2019, Accepted 17 December 2019
▪ 제1저자(First Author) : Woohwan Nam (Email : whnam87@naver.com)

이러한 성장세는 블루스택(BlueStacks), 녹스(Nox), 미뮤(Memu) 등 다양한 안드로이드 에뮬레이터 출시로 이어졌다. 세계 최대 규모의 안드로이드 에뮬레이터인 블루스택에 의하면 2018년 블루스택 다운로드 수는 약 3억 5천만 건이며, 블루스택을 이용해 매달 실행되는 전 세계 모바일 게임 실행 횟수는 20억 회에 육박하는 것으로 나타났다[3]. 특히 블루스택은 '모바일 게이밍 플랫폼'을 표방하며 넥슨이 출시한 모바일 MMORPG 게임인 '트라하'의 PC방 서비스를 정식 지원한 데 이어 삼성전자와의 협업을 통해 갤럭시스토어 전용서비스를 제공하는 등 안드로이드 에뮬레이터 생태계를 확대하고 있다[4]. IT서비스 업체 민앤지는 중국 내 1위 안드로이드 에뮬레이터인 텐센트(Tencent)사의 게이밍 버디(Gaming Buddy)를 국내에 도입하여 출시하는 등 안드로이드 에뮬레이터 시장은 더욱 확대되고 있다[5].

안드로이드 에뮬레이터를 이용하는 사용자가 증가하면서 사용자의 편의를 위한 다양한 기능이 추가되고 성능도 향상되었다. 가상 안드로이드 환경에서 루팅, 스크린샷, GPS 위치 변경, 단말기식별번호(IMEI) 값 수정 등 각종 정보의 변경은 물론, 가상 인스턴스(virtual instance) 복제 및 동시 실행, PC와 파일 공유 등 다양한 기능을 지원하고 있다.

문제는 이러한 안드로이드 에뮬레이터의 기능을 오용하거나 범죄에 악용하는 사례가 발생하고 있다는 것이다. 대표적으로 스크린샷 기능을 이용한 전자책(eBook) 복제 및 유포, GPS와 기기명 변경을 통한 알리바이 조작, 인스턴스 복제를 이용한 다계정 생성 및 운영, 멀티 실행과 매크로 기능을 이용한 게임 작업장 운영 등을 꼽을 수 있다. 특히 안드로이드 에뮬레이터의 원격 코드 실행 취약점을 공격하거나 가상화폐 채굴 악성코드를 설치하는 사례도 발생하였다[6].

안드로이드 에뮬레이터상의 흔적들은 안드로이드 환경이 구축된 가상디스크 파일 안에 저장되어 있기 때문에, 일반적인 하드디스크 매체에 대한 디지털 포렌식 분석만으로는 발견이 불가능하다. 따라서 윈도우 호스트와 가상디스크 파일 내부에 존재하는 안드로이드 에뮬레이터와 관련된 아티팩트들을 수집 및 분석함으로써 안드로이드상에서의 사용 흔적을 확인하고 범죄와 관련된 증거파일을 획득하는 방법이 필요하다.

본 연구에서는 국내에서 대표적으로 사용되는 블루스택(v4.90.0.1046), 녹스(v6.3.0.8), 미뮤(v6.1.1) 등 3종의 안드로이드 에뮬레이터의 구조를 분석하고 테이터 수집 절차에 따른 각 안드로이드 에뮬레이터별 주요 포렌식 항목을 도출하여 분석함으로써 획득 가능한 아티팩트들을 확인하고 실제 사건 사례를 통해 유효성을 검증해 보고자 한다.

Ⅱ. 관련 연구

안드로이드 에뮬레이터는 VMware나 VirtualBox와 같은 가상화 기술을 응용한 소프트웨어이기 때문에 기본적으로 가상 환경 포렌식과 안드로이드 운영체제에 대한 포렌식 분석이 기본이 된다.

가상 환경 포렌식 연구와 관련하여, 임성수[7]는 VMware 가상머신 이미지 데이터와 윈도우 호스트 시스템에 남겨지는 데이터를 수집 및 분석하였으며, 손상된 상태의 이미지 파일 조사 방법과 실제 복구사례를 제시하고 분석 절차를 도출하였다. 장상희[8]는 Citrix, VMware, Microsoft 등 전 세계적으로 가장 많이 사용되고 있는 데스크톱 가상화 솔루션을 소개하고 각 솔루션에 대한 디지털 포렌식 조사 절차 및 방법을 제안하였다.

안드로이드 운영체제 포렌식 분석과 관련하여 오정훈[9]은 안드로이드 환경에서 데이터 수집 및 분석 절차에 기반한 루팅 및 ADB, SD카드 등 안드로이드 데이터 수집 방법을 제시하고, 웹 브라우저 기록, 멀티미디어 데이터, 위치 정보 등 표준 안드로이드 운영체제의 사용자 데이터 획득 방법과 데이터 구조를 분석하였다.

안드로이드 에뮬레이터 자체에 대한 연구와 관련해서 윤종성[10]은 안드로이드 에뮬레이터를 이용해 야기될 수 있는 모바일 게임의 보안상 위험성에 대해 언급하면서 주요 안드로이드 에뮬레이터 빌드 정보(build information)를 이용한 클라이언트와 서버 및 네트워크에서 안드로이드 에뮬레이터를 탐지하는 기법을 제시하였다. Teng[11] 등은 안드로이드 에뮬레이터 종류별 설치 경로와 레지스트리 정보 등 기본적인 안드로이드 에뮬레이터 포렌식 항목을 제시하였고, Wicrk, Sureport 등 4개 종류의 데이터 은닉 앱을 이용해 안드로이드 에뮬레이터상에서의 데이터 은닉 사례를 분석하는 방법을 제안하였다. 그러나 해당 연구들은 에뮬레이터의 탐지와 데이터 은닉 앱의 분석에 초점을 맞추었지 안드로이드 에뮬레이터 자체에 대한 포렌식 분석 기법을 다루지 않았다는 한계가 있다.

Ⅲ. 안드로이드 에뮬레이터 포렌식 기법의 역할

디지털 증거 수집·처리에 대한 원칙과 일련의 절차는 그동안 다양한 학술연구와 법률 및 제도의 정비를 통해 발전해 왔다. 특히 최근 경찰청에서는 「디지털 증거 수집 및 처리 등에 관한 규칙」을 바탕으로 일련의 디지털 증거 처리 과정에서 준수해야할 원칙과 업무절차를 규정한 「디지털 증거분석 표준 업무처리 지침」을 2018년 10월에 발간하였다. 해당 지침을 통해 디지털 증거의 수집, 이송, 분석, 보관에 대한 절차와 준수 사항을 규정하였으며, 특히 저장 매체별 '디지털 증거 분석 표준 절차'를 제시하였다. 지침에 제시된 일반적인 PC의 하드디스크에 대한 디지털 증거 분석 표준 절차는 크게 해시값 검증, 시간대 확인 및 분석 프로그램 적용, 복구 수행, 상세 분석, 분석 결과 추출, 해시값 생성의 절차로 구성된다[12]. 상세 분석단계에서는 레지스트리, 파일, 메타데이터 등 각종 아티팩트에 대한 실질적인 분석이 이루어지며 다양한 흔적들이 발견

되게 된다. 만약 분석 작업 중에 안드로이드 에뮬레이터가 설치되어 있거나 있었던 흔적이 발견된다면, 안드로이드 환경에서 이루어진 범죄 행위 확인을 위해 안드로이드 에뮬레이터의 데이터 수집과 분석 절차가 이루어질 필요가 있다. 특히 컴퓨터를 직접적으로 이용한 범죄 혐의로 압수된 하드디스크의 경우에는 안드로이드 에뮬레이터의 설치 여부를 반드시 확인하여 이에 따른 수집 및 분석 작업이 이루어져야 한다.

안드로이드 에뮬레이터 포렌식 기법은 하드디스크 매체에 대한 일반적인 디지털 포렌식 절차로는 확인이 어려운 안드로이드 에뮬레이터상에서의 사용 흔적 분석과 범죄와 관련된 증거파일 획득을 가능하게 해준다. 이는 하드디스크 매체의 완벽한 분석이 이루어 질 수 없는 일반적인 디지털 포렌식 절차에 대한 보완적인 성격을 가질 수 있다. 또한 이 기법은 기존의 태블릿과 스마트폰 중심의 안드로이드 포렌식 기법을 활용한 진보된 분석 절차가 될 수 있다. 안드로이드 에뮬레이터는 기본적인 안드로이드 시스템 구조와 저장 방식을 유지하면서도 에뮬레이터별로 서로 다른 기능과 아티팩트 경로를 가지고 있어 이에 관한 데이터 획득과 분석을 포함하고 있기 때문이다.

Ⅳ. 안드로이드 에뮬레이터 관련 데이터 수집

안드로이드 에뮬레이터 포렌식 분석을 위해서는 사용자의 데이터가 저장되어 있는 가상디스크 파일, PC에 저장되는 각종 로그파일과 레지스트리 정보를 모두 확보해야 한다. 특히 확보 당시 안드로이드 에뮬레이터가 실행 중인 상태라면 RAM 데이터까지 확보해야 한다. 물론 안드로이드 에뮬레이터를 구동 중인 Windows 등 운영체제에서 라이브 포렌식 절차를 이용해 안드로이드 에뮬레이터에 대한 메모리 정보를 확보하는 것이 가장 좋은 방법이지만, 에뮬레이터 내에서 실행 중인 RAM 데이터를 분석하는 것은 현실적으로 불가능하기 때문에 Figure 1.과 같이 ADB(Android Debugging Bridge) shell의 dumpsys meminfo와 같은 명령어를 이용해 RAM 데이터를 확보 및 덤프하는 방법도 사용할만하다.

상황에 따른 안드로이드 에뮬레이터 데이터 수집 및 분석 절차는 Figure 2.와 같다.

```
Applications Memory Usage (kB):
Uptime: 38196 Realtime: 38196

Total PSS by process:
    62420 kB: com.vphone.launcher (pid 1973 / activities)
    61348 kB: com.google.android.gms (pid 2069)
    52386 kB: com.google.android.gms.persistent (pid 2262)
    44456 kB: com.android.systemui (pid 1831)
    40309 kB: system (pid 1705)
    32766 kB: com.android.vending (pid 2601)
    30044 kB: com.android.vending:download_service (pid 3178)
    17248 kB: com.google.android.gms.unstable (pid 2952)
    11383 kB: com.android.phone (pid 1924)
    11319 kB: mediaserver (pid 1453)
    10924 kB: com.google.android.gms.ui (pid 2974)
     8580 kB: com.google.android.play.games (pid 2728)
     7265 kB: com.google.process.gapps (pid 2105)
     6123 kB: android.process.acore (pid 2018)
     5616 kB: android.process.media (pid 2232)
     4722 kB: com.android.inputservice (pid 1866)
     4378 kB: com.android.providers.calendar (pid 2436)
     3821 kB: zygote (pid 1457)
     3523 kB: com.android.keychain (pid 2422)
     3450 kB: com.android.managedprovisioning (pid 2513)
     3430 kB: com.android.onetimeinitializer (pid 2534)
     2825 kB: surfaceflinger (pid 1443)
     1239 kB: logd (pid 1437)
     1027 kB: wpa_supplicant (pid 2059)
      981 kB: drmserver (pid 1452)
      878 kB: netd (pid 1447)
      832 kB: adbd (pid 1439)
      733 kB: vold (pid 1442)
      708 kB: healthd (pid 1438)
      595 kB: sdcard (pid 1399)
```

그림 1. ADB Shell을 이용한 메모리 덤프 결과 일부

Figure 1. Part of Result for Memory Dump Using ADB Shell

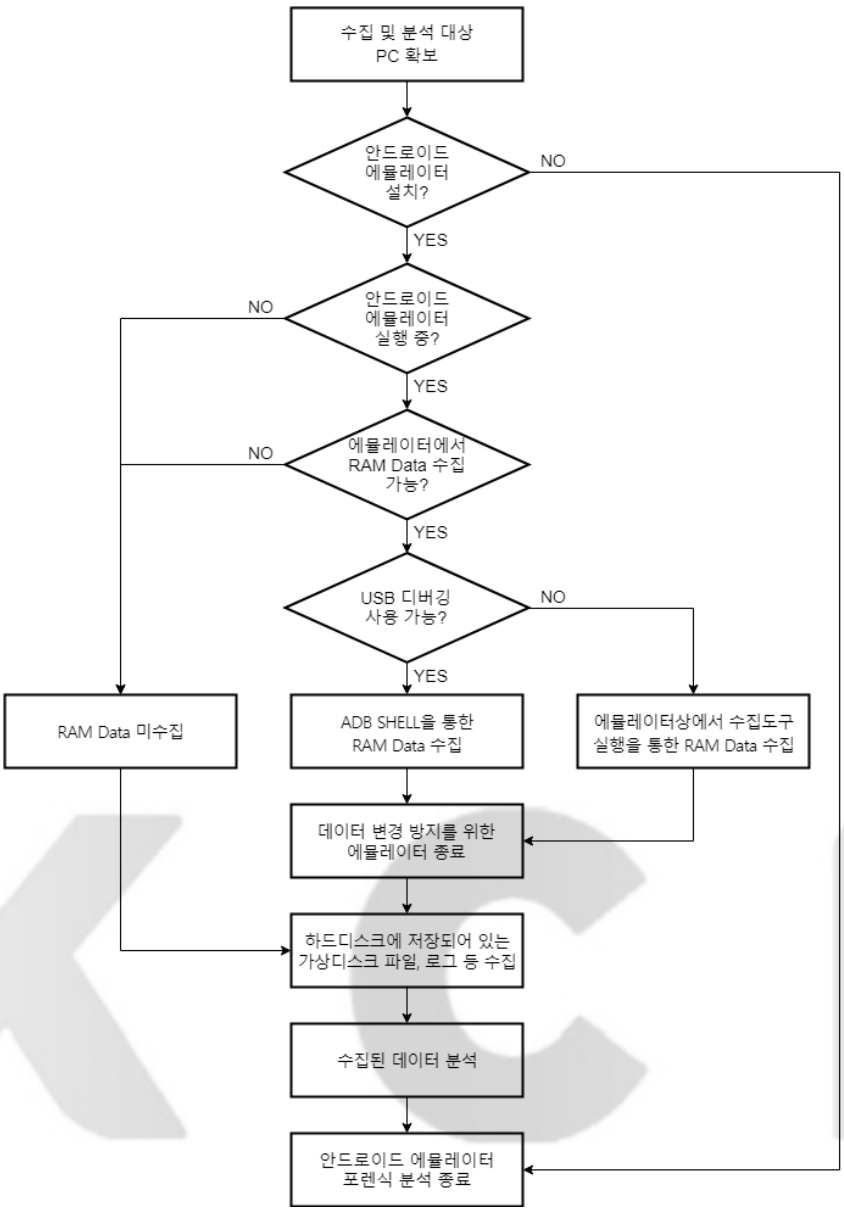

그림 2. 안드로이드 에뮬레이터 데이터 수집 절차
Figure 2. Data Collection Process for Android Emulators

안드로이드 에뮬레이터 분석을 위해서는 사용자 데이터가 저장되는 가상디스크 파일을 확보해야 하며, 특히 에뮬레이터 설치 및 실행 로그와 레지스트리 정보까지 확보하여 분석할 필요가 있다. 안드로이드 에뮬레이터는 운영체제가 설치되어 있는 로컬디스크에 설치되는 것이 보통이지만, D드라이브와 같은 추가적인 보조저장매체가 존재할 경우에는 수십 기가바이트에 달하는 가상 디스크파일의 여유 공간 확보를 위해 설치경로가 보조저장장치의 드라이브로 변경되는 경우가 있다. 이에 따라 다양한 설치환경의 안드로이드 에뮬레이터 포렌식 분석을 위해서는 에뮬레이터별 설치 및 실행 로그와 레지스트리 분석 등이 선행되어야 한다.

윈도우 시스템에 안드로이드 에뮬레이터를 설치하면 안드로이드 운영체제 데이터가 저장되어 있는 가상디스크 파일과 사용자의 데이터를 저장하는 가상디스크 파일 등 '기본 가상디스크 파일(base virtual file)'이 설치된다. 안드로이드 운영체제가 설치되어 있는 가상디스크 파일은 안드로이드 환경을 구축 및 구동(boot)하기 위해 사용되며, 사용자의 데이터를 저장하는 가상디스크에는 설치된 앱, 각종 앱 데이터, 환경설정 등을 저장하는데 사용된다. 최초 안드로이드 에뮬레이터 실행 시 사용자의 데이터가 저장되는 '기본 가상디스크 파일'이 복제되고 이를 통해 새로 생성된 가상디스크 파일에 향후 사용자가 설치하는 앱, 앱 데이터, 환경설정 등 각종 사용자 데이터가 저장된다. 이 사용자 데이터가 저장되는 가상디스크 파일은 사용자가 생성하는 인스턴스 수만큼 복제되어 만들어지고 사용자 데이터가 저장된다.

블루스택의 가상디스크 파일 중 Root.vdi, Prebundled.vdi, Fastboot.vdi는 에뮬레이터 구동과 안드로이드 운영체제의 부팅을 위해서 사용되며, 실제 사용자의 데이터가 저장되는 가상디스크 파일은 Data.vdi(Data_0.vid)이다. 마찬가지로 녹스의 base5-disk1.vmdk는 구동과 부팅을 위해 사용되는 가상디스크 파일이며, 기본 가상디스크 파일인 base5-disk2.vmdk를 복제하여 생성된 nox-disk2.vmdk에 각종 사용자의 데이터가 저장된다. 미뮤 또한 기본 가상디스

크 파일에서 복제 및 생성한 MEmu51-{Build Version}-disk2.vmdk 가상디스크 파일에 사용자의 데이터를 저장한다.

안드로이드 운영체제가 설치되어 있는 가상디스크 파일은 최초 에뮬레이터 구동과 안드로이드 운영체제의 부팅을 위해서만 사용되기 때문에, 이와 관련된 가상디스크 파일에는 추가적인 데이터의 저장은 발생하지 않는다. 그러나 사용자가 설치한 앱이나 각종 데이터 및 환경설정이 저장되는 가상디스크 파일에는 사용자의 모든 데이터가 계속적으로 저장되기 때문에 해당 가상디스크 파일의 확보가 중요하다.

Windows 7/10에서의 각 안드로이드 에뮬레이터별 설치 및 실행 로그, 레지스트리, 그리고 사용자 데이터가 저장되는 가상디스크 파일명과 그 위치는 Table 1.와 같다.

표 1. Windows 7/10에서의 안드로이드 에뮬레이터 주요 포렌식 항목
Table 1. Major Forensic Items for Android Emulators on Windows 7/10

Items	BlueStacks	Nox	Memu
Emulator System Path	%ProgramFiles%\BlueStacks	%ProgramFiles%\Nox\bin	%ProgramFiles%\Microvirt
Execution Log	%ProgramData%\BlueStacks	%ProgramFiles%\Nox\bin\BignoxVMS\nox\Logs %UserProfile%\AppData\Local\Nox	%ProgramFiles%\Microvirt\MEmu\MemuHyperv VMs\MEmu\Logs %UserProfile%\.MemuHyperv
Install Log	%UserProfile%\AppData\Local\BlueStacks	%UserProfile%\.BigNox %UserProfile%\AppData\Local\Nox\vmInstall	%ProgramFiles%\Microvirt\MEmu
Registry Key	HKEY_LOCAL_MACHINE\SOFTWARE\BlueStacks	HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\Nox	HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\MEmu
Base Virtual Disk Files	Root.vdi Data.vdi Prebundled.vdi Fastboot.vdi	base5-disk1.vmdk base5-disk2.vmdk	MEmu51-{Build Version}-disk1.vmdk MEmu51-{Build Version}-disk2.vmdk
Location of Base Virtual Disk Files	%ProgramData%\BlueStacks\Engine\Android	%ProgramFiles%\Nox\bin\data\base5	%ProgramFiles%\Microvirt\MEmu\image\51
User Data Virtual File	Data.vdi + Data_0.vdi	nox-disk2.vmdk	MEmu51-{Build Version}-disk2.vmdk
Location of User Data Virtual Disk File	%ProgramData%\BlueStacks\Engine\Android	%ProgramFiles%\Nox\bin\BignoxVMS\nox	%ProgramFiles%\Microvirt\MEmu\MemuHyperv VMs\MEmu

Ⅴ. 데이터 분석

1. 윈도우 호스트 데이터 분석

윈도우 호스트의 하드디스크상에는 Table 1.와 같이 각 에뮬레이터별 설치 및 실행 로그, 레지스트리와 함께 안드로이드 환경을 구동하기 위한 가상디스크 파일이 저장된다. 안드로이드 운영체제 내의 사용자 데이터는 기본적으로 가상디스크 파일에 저장되지만, 일부 안드로이드 에뮬레이터 기능과 관련된 사용자 데이터는 윈도우 호스트의 하드디스크상에 저장된다. 대표적으로 용이한 게임 조작을 위해 사용되는 키 매핑과 게임패드 인터페이스 등 가상키보드 기능에 대한 사용자 데이터와 사용자가 생성한 반복 작업(macro)이 있다. 가상키보드와 반복 작업 기록은 불법 게임 작업장 운영과 같은 범죄 혐의를 입증할 수 있는 증거로 활용될 수 있다.

또 윈도우 호스트와 안드로이드 에뮬레이터 간 동영상, 사진 등 파일을 용이하게 전송하기 위해 윈도우 호스트상에 생성된 공유폴더를 이용하게 되는데, 이 공유폴더에 남겨져 있는 파일 등을 통해 안드로이드 에뮬레이터와 공유된 파일을 알 수

있다. 특히 안드로이드 에뮬레이터의 공유폴더를 이용한 파일 이동과 복사는 흔히 윈도우상에서 파일의 복사와 이동하는 작업과 다르기 때문에 자료유출방지 솔루션으로 탐지되지 않아 파일 복사 통제가 무력화되기도 한다. 이에 따라 윈도우 호스트상의 안드로이드 에뮬레이터 공유폴더를 분석하면 안드로이드 에뮬레이터를 이용한 데이터 유출 사건 수사에서 도움이 될 수 있다.

각 안드로이드 에뮬레이터별 사용자의 가상 키보드, 매크로가 저장되는 위치와 안드로이드 에뮬레이터 공유폴더의 경로는 Table 2.과 같다.

표 2. 하드디스크상의 사용자 데이터 경로
Table 2. Path of User Data in Hard Disk Drive

Items	BlueStacks	Nox	Memu
Virtual Key	%ProgramData%\BlueStacks\Engine\UserData\InputMapper\UserFiles	%UserProfile%\AppData\Local\Nox\keyboardConfig\com.vphone.launcher\{Screen Resolution}	%ProgramFiles%\Microvirt\MEmu
Macro	%ProgramData%\BlueStacks\Engine\UserData\InputMapper\UserScripts	%UserProfile%\AppData\Local\Nox\record	%ProgramFiles%\Microvirt\MEmu\scripts
Shared Folder (Image)	%UserProfile%\Pictures\BlueStacks	%UserProfile%\Nox_share\ImageShare	%UserProfile%\Pictures\MEmu Photo

2. 가상디스크 파일(.vdi, .vmdk) 데이터 분석

2.1. 안드로이드 에뮬레이터 공통 데이터 분석

블루스택은 대부분의 안드로이드 에뮬레이터가 사용하는 가상화디스크 파일 구조(.vmdk)와 달리 .vdi 구조를 사용한다. 블루스택의 가상디스크 파일 중 Root.vdi, Prebundled.vdi, Fastboot.vdi는 에뮬레이터 구동과 안드로이드 운영체제의 부팅을 위한 파일이므로, 파일의 가상디스크가 읽기전용이고 별도의 사용자 데이터가 저장되지는 않는다. 실제로 사용자 데이터가 저장되는 가상디스크 파일은 Data.vdi 와 Data_0.vdi 로 구성되는데 이 파일들은 서로 부모-자식 관계로 연결되어 있기 때문에 해당 파일을 각각 불러오거나 분석할 수 없는 구조로 되어 있다[13]. 따라서 사용자 데이터가 저장되어 있는 온전한 가상디스크 파일을 획득하기 위해서는 Oracle에서 제공하는 CUI(Character User Interface)형식의 변환 프로그램(VBoxManage Internal Commands)을 이용하여 Data_0.vid 파일을 Data.vid 파일의 자식 관계로 설정하고 한 개의 가상화 파일로 병합 및 변환시키는 작업이 필요하다.

기본적으로 사용자 데이터가 저장되는 가상디스크 구조는 표준 안드로이드 운영체제의 구조를 기본으로 하고 있으며, 안드로이드 에뮬레이터가 제공하는 기능에 따라 조금씩 차이를 보인다. 최근 작업 내역, 최근 이미지 내역, 사용 내역, 시간대, SD 카드 등의 경로와 저장 방식은 모든 안드로이드 에뮬레이터가 동일하다.

최근 작업 내역은 [root]\system\recent_tasks 폴더 내에 앱 고유번호(UID), 최초 실행 시간과 최종 실행 시간 등의 데이터가 앱별로 {Task ID Number}_task.xml 파일 형태로 저장되며, 저장되는 세부 항목은 Table 3.과 같다[14].

표 3. 최근 작업 내역 항목 및 설명
Table 3. Items and Description of Recent Tasks File

Items		Description
Recent Tasks	Task ID number	Used to correlate snapshot and recent image files
	Effective UID	App identifier
	First active time	Timestamp in millisecond epoch time
	Last active time	Timestamp in millisecond epoch time
	Last time moved	Timestamp in millisecond epoch time
	Affinity	Bundle ID name

	Calling package	Bundle ID or process that called the referenced recent task
	Real activity	Gives information on app usage at time of recording and snapshot creation

Table 3.을 통해 최근 실행된 앱에 대한 기본적인 내용과 시간정보 등을 확인할 수 있다. 특히 Task ID number는 최근 작업 내역 .xml 파일이 생성될 때 [root]\system\snapshots 폴더에 {Task ID Number}.jpg 형식으로 저장되는 해당 앱의 스크린캡처된 이미지를 식별하는데 활용될 수 있다.

최근 이미지 내역은 [root]\system\recent_images 폴더 내에 최근 앱에서 이용한 이미지가 저장되며, 해당 이미지 파일의 파일명, 확장자, 크기, 생성 일시, 액세스한 일시, 수정 일시 등을 확인할 수 있다.

사용 내역(usage stats)은 앱 실행, 이벤트 로그, 각종 설정 변경에 대한 시간적 통계 데이터를 기록하며 [root]\system\usagestats\0 폴더 하위에 daily, monthly, weekly, yearly 단위로 저장된다. 기록이 저장되는 .xml 파일은 해당 시간 단위별로 생성되며, 기록되는 내용은 Figure 3.과 같다. 기록되는 시간 값들은 밀리세컨드 단위의 에포크 시간 형태로 기록되며, 앱 및 이벤트 로그별 저장되는 세부 항목은 Table 4.와 같다[15].

```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
- <usagestats version="1" endTime="86399999">
  - <packages>
      <package lastTimeActive="34958353" package="com.minwise.system" timeActive="1106" lastEvent="2" />
      <package lastTimeActive="36154000" package="com.microvirt.launcher" timeActive="1198780" lastEvent="3" />
    </packages>
  - <configurations>
      <config lastTimeActive="86399998" timeActive="51446675" count="1" active="true" fs="1066611507" locale="en-US"
        touch="1" key="1" keyHid="1" hardKeyHid="2" nav="1" navHid="2" ori="2" scrLay="268435491" ui="17" width="1066"
        height="575" sw="600" density="192" />
      <config lastTimeActive="34953262" timeActive="344" count="1" fs="1065353216" locale="en-US" touch="1" key="1"
        keyHid="1" hardKeyHid="2" nav="1" navHid="2" ori="2" scrLay="268435491" width="1066" height="575" sw="600"
        density="192" />
      <config lastTimeActive="34952918" timeActive="21" count="1" fs="1065353216" locale="en-US" touch="1" key="1"
        keyHid="1" hardKeyHid="2" nav="1" navHid="2" ori="2" scrLay="268435491" width="1066" height="600" sw="600"
        density="192" />
      <config lastTimeActive="34953324" timeActive="62" count="1" fs="1065353216" locale="en-US" touch="1" key="1"
        keyHid="1" hardKeyHid="2" nav="1" navHid="2" ori="2" scrLay="268435491" ui="17" width="1066" height="575" sw="600"
        density="192" />
    </configurations>
  - <event-log>
      <event time="34952898" package="android" type="5" fs="1065353216" locale="en-US" touch="1" key="1" keyHid="1"
        hardKeyHid="2" nav="1" navHid="2" ori="2" scrLay="268435491" width="1066" height="600" sw="600" density="192" />
      <event time="34952919" package="android" type="5" fs="1065353216" height="575" />
      <event time="34953263" package="android" type="5" fs="1065353216" ui="17" />
      <event time="34953325" package="android" type="5" fs="1066611507" />
      <event time="34954066" package="com.microvirt.launcher" class="com.microvirt.launcher.Launcher" type="1" />
      <event time="34957210" package="com.microvirt.launcher" class="com.microvirt.launcher.Launcher" type="2" />
      <event time="34957247" package="com.minwise.system" class="com.minwise.adzip.launchadapp.LauncherActivity"
        type="1" />
      <event time="34958353" package="com.minwise.system" class="com.minwise.adzip.launchadapp.LauncherActivity"
        type="2" />
      <event time="34958364" package="com.microvirt.launcher" class="com.microvirt.launcher.Launcher" type="1" />
    </event-log>
</usagestats>
```

그림 3. 사용 내역 파일 내용
Figure 3. Content of Usage Stats File

표 4. 사용 내역 주요 항목 및 설명
Table 4. Major Items and Description of Usage Stats

Items		Description
Package	**Last time active**	Timestamp in millisecond epoch time
	Package	Bundle ID name
	Time active	Timestamp length in millisecond epoch time
	Last event	'0' : No event type '2' : Move to background
Event Log	**Event time**	Timestamp in millisecond epoch time
	Package	Bundle ID name
	Class	Modules and activity name
	Type	'0' : No event type '1' : Move to foreground '2' : Move to background '5' : Configuration change '7' : User interaction '8' : Shortcut Invocation

Table 4.와 같이 .xml 파일에 기록된 앱의 Last event 항목과 이벤트 로그의 Type 항목 값을 통해 앱 실행 및 이벤트 발생 시 포어그라운드(Type 1) 또는 백그라운드(Type 2)로의 이동이나 설정사항 변경(Type 5)을 확인함으로써 실행 당시 일련의 과정과 선후관계에 따른 연관성 등을 파악할 수 있다.

사용자의 시간대 정보는 〔root〕\property\persist.sys.timezone 파일을 통해 확인 가능하며, 사진 앱 폴더(DCIM)와 Downloads, Movie, Music, Picture 폴더 등 SD 카드에 저장되는 사용자 데이터는 〔root〕\media\0 경로를 통해 확인할 수 있다.

2.2. 안드로이드 에뮬레이터별 데이터 분석

기기명, 휴대폰 번호, IMEI 값, 가상 위치(GPS), WiFi MAC 주소 등 기기와 관련된 정보는 안드로이드 에뮬레이터가 제공하는 기능에 따라 가상디스크 파일에 저장되기도 하고 그렇지 않기도 한다. 안드로이드 에뮬레이터는 이동통신사의 셀룰러 네트워크를 이용할 수 없기 때문에 설정된 휴대폰 번호를 이용하여 문자를 보내거나 받을 수는 없지만, 휴대폰 번호가 인식되어야만 이용할 수 있는 앱을 사용하기 위한 용도로 주로 사용된다.

블루스택은 휴대폰 번호 설정, IMEI 값 수정, WiFi MAC 주소 설정 기능은 없기 때문에 이에 대한 기록이 가상디스크에 저장되지 않으나, 기기명 및 제조사 등 데이터는 〔root〕\data\.bluestacks.prop 파일을 통해 확인이 가능하다. 또한 사용자가 가장 최근 이용한 가상 위치(GPS) 값은 〔root〕\downloads\.location_data 파일로 알 수 있다.

녹스는 기기명, 휴대폰 번호, IMEI 값, 가상 위치, WiFi MAC 주소 등 사용자가 이용한 데이터가 〔root〕\property 폴더에 각 파일별로 저장된다.

미뮤는 블루스택이나 녹스와 달리 기기명, 휴대폰 번호, IMEI 값, 가상 위치, WiFi MAC 주소 등의 정보가 가상 디스크 파일이 아닌 윈도우 호스트 하드 디스크에 저장되며, 설치 경로의 MEmu.memu 파일을 통해 모두 확인할 수 있다.

기기 관련 데이터가 저장되는 각 안드로이드 에뮬레이터별 경로는 Table 5.과 같다.

표 5. 가상디스크 파일상의 사용자 데이터 경로
Table 5. Path of User Data in Virtual Disk File

Items	BlueStacks	Nox	Memu
Device Name	〔root〕\data\.bluestacks.prop	〔root〕\property\persist.nox.model	%ProgramFiles%\Microvirt\MEmu\MemuHyperv VMs\MEmu\ MEmu.memu (Windows Host)
Phone Number	UNKNOWN (Not Supported)	〔root〕\property\persist.nox.modem.phonumber	
IMEI	UNKNOWN (Not Supported)	〔root〕\property\persist.nox.modem.imei	
Virtual Location (GPS)	〔root〕\downloads\.location_data	〔root〕\property\persist.nox.gps.latitude 〔root〕\property\persist.nox.gps.longitude	
WiFi MAC	UNKNOWN (Not Supported)	〔root〕\property\persist.nox.wifimac	

Ⅵ. 안드로이드 에뮬레이터의 악용 가능성 판단 및 사례 분석

스크린샷, 가상 위치(GPS)값 변경, IMEI 값 수정, 인스턴스 복제 및 동시 실행, 반복 작업 등 안드로이드 에뮬레이터에 내장된 다양한 기능들은 소프트웨어 디버깅이나 사용자의 편의를 위해 제공되고 있다. 그러나 문제는 이러한 안드로이드 에뮬레이터의 기능을 오용하거나 범죄에 악용할 수 있다는 것이다. 대표적으로 스크린샷 기능을 이용한 전자책(eBook) 복제 및 유포, GPS와 기기명 변경을 통한 알리바이 조작, 인스턴스 복제를 이용한 다계정 생성 및 운영, 멀티 실행과 매크로 기능을 이용한 게임 작업장 운영 사례를 들 수 있다.

이러한 점을 고려하여 안드로이드 에뮬레이터 포렌식 분석을 통해 실제 오용 또는 범죄 악용 여부를 판단할 필요가 있으나, 단편적인 분석으로는 악용 여부를 판단하기 어렵기 때문에 윈도우 호스트상에 남겨진 로그 파일과 가상디스크 파일 내의 데이터를 분석하여 종합적으로 판단하여야 한다. 특히 인스턴스 복제 및 동시 실행 기능은 각종 악용행위에 이용될 수 있는 가능성이 매우 높기 때문에 분석 시 사용자 데이터가 저장되는 가상디스크 파일이 상당수 복제 및 생성되어 있다면 실

제 악용 여부 판단을 위한 추가적인 아티팩트별 분석이 진행되어야 한다. 또한 스크린샷 기능과 매크로 기능 역시 악용 여부를 판단하는데 중요한 기준이 될 수 있기 때문에, 해당 파일이 저장되는 폴더와 함께 PC와의 공유폴더도 확인할 필요가 있다.

에스24, 교보문고, 리디북스 등 데스크톱용 eBook 뷰어들은 화면 캡처를 통한 전자책 불법복제 방지와 DRM 보호를 위해 스크린샷 기능을 제한하고 있다. 그러나 모바일 기기용 eBook 뷰어를 안드로이드 에뮬레이터에 설치하면 안드로이드 에뮬레이터에 내장된 스크린샷 기능을 이용해 화면캡처가 가능하다. 일부 eBook 뷰어는 모바일 기기에서의 화면캡처를 제한하는 기능도 내장되어 있지만, 이러한 경우에도 윈도우 호스트상에서 안드로이드 에뮬레이터 화면의 캡처가 가능하다.

이에 대해서는 로그, 최근 작업 내역 등 사용자 활동, 앱 실행 기록과 함께 최근 이미지 내역 및 스크린샷 파일이 저장되는 SD 카드 경로의 DCIM 폴더나 이미지 공유폴더를 확인하여 분석할 수 있다. 실제 녹스 에뮬레이터의 스크린샷 기능을 이용해 eBook 본문을 캡처한 이미지 파일이 공유폴더 경로인 %UserProfile%\Nox_share\ImageShare 에 저장된 모습은 Figure 4.와 같다.

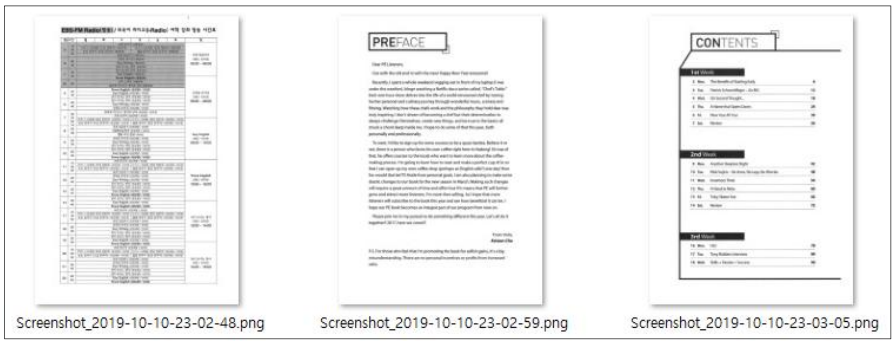

그림 4. eBook 본문이 캡처된 이미지 파일들
Figure 4. Image Files Captured eBook Contents

안드로이드 에뮬레이터의 기기명 및 가상 위치 변경과 VPN 앱을 이용한다면 알리바이 조작이나 모바일 기기의 거짓 해킹 기록 생성이 가능하다. 구글의 경우 계정 로그인시 접속 IP, 기기명, 웹브라우저에 대한 정보가 사용자 기록으로 저장되며, 구글 지도 앱을 통한 GPS 위치 값 등 위치 정보와 각종 앱 사용 기록도 남는다. 안드로이드 에뮬레이터를 이용해 기기명을 설정하고, 가상 위치 기능과 VPN 앱을 이용하여 국외 등 임의의 위치로 지정한 뒤 구글 계정에 로그인하면 구글상에 위치 기록을 남길 수 있다. 이러한 방법을 통해 알리바이를 조작하거나 모바일 기기의 거짓 해킹 기록을 생성할 수 있다. 실제로 조작된 위치정보와 임의로 설정된 기기명으로 구글에 로그인하고, 해당 기록을 근거로 구글측에 계정 해킹을 주장하여 계정과 연결된 모바일 게임 계정의 아이템 구매 결제내역을 환불받는 식으로 악용한 사례가 있었다. 이러한 경우 해당 흔적은 윈도우 호스트 하드디스크에 남아있는 에뮬레이터 사용자 로그와 가상디스크 파일에 저장된 기기명 및 가상위치 기록을 통해 확인할 수 있다. 실제 안드로이드 에뮬레이터를 이용해 러시아의 임의 지역으로 지정된 가상 위치와 기기명이 구글에 기록된 모습은 Figure 5.와 같다.

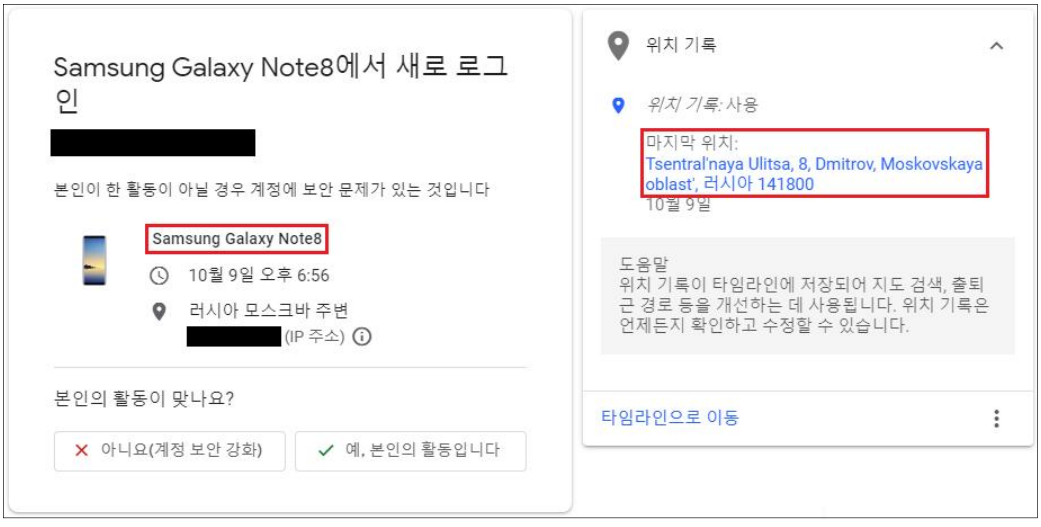

그림 5. 구글 대시보드상에 기록된 기기 정보 및 가상 위치
Figure 5. Device Information and Virtual Location on Google Dashboard

안드로이드 에뮬레이터는 안드로이드 인스턴스 복제 및 동시 실행 기능을 제공하며, PC 사양에 따라 한 대의 데스크톱 PC로 수 십 개 이상의 인스턴스를 동시에 실행할 수 있다. 다수의 안드로이드 환경을 구동하여 여러 모바일 게임 계정을

동시에 접속시키고, 매크로 기능을 이용해 게임 플레이를 자동화하는 방법으로 게임 내 재화와 희귀 아이템 획득을 위한 작업장을 구축하기도 한다. 실제로 2017년 엔씨소프트의 모바일 게임인 리니지M 출시 당시 이러한 방법으로 생성된 작업장 계정 수가 40만 여 개에 이르자 엔씨소프트측은 해당 계정들의 접속을 영구적으로 금지시키기도 하였다[16]. 사용자의 데이터가 저장되는 가상디스크 파일은 생성된 인스턴스 수만큼 생성되기 때문에 윈도우 호스트 하드디스크에 저장된 가상디스크 파일의 개수를 통해 인스턴스 동시 실행을 확인할 수 있다. 또한 각 가상디스크 파일의 실행 로그 및 각종 앱 실행 기록뿐만 아니라 사용자가 생성한 가상 키보드 및 반복 작업 데이터를 분석함으로써 이에 대한 기록을 확인할 수 있다. 윈도우 호스트 하드디스크에 저장된 녹스 에뮬레이터의 사용자 반복 작업 파일 내용은 Figure 6.과 같다.

```
0ScRiPtSePaRaToR1600|900|MULTI:1:0:453:435ScRiPtSePaRaToR305
0ScRiPtSePaRaToR1600|900|MULTI:0:6ScRiPtSePaRaToR406
0ScRiPtSePaRaToR1600|900|MULTI:0:6ScRiPtSePaRaToR406
0ScRiPtSePaRaToR1600|900|MULTI:0:1ScRiPtSePaRaToR406
0ScRiPtSePaRaToR1600|900|MSBRL:0:0ScRiPtSePaRaToR406
0ScRiPtSePaRaToR1600|900|MULTI:1:0:1156:444ScRiPtSePaRaToR656
0ScRiPtSePaRaToR1600|900|MULTI:0:6ScRiPtSePaRaToR741
0ScRiPtSePaRaToR1600|900|MULTI:0:6ScRiPtSePaRaToR741
0ScRiPtSePaRaToR1600|900|MULTI:0:1ScRiPtSePaRaToR741
0ScRiPtSePaRaToR1600|900|MSBRL:0:0ScRiPtSePaRaToR741
0ScRiPtSePaRaToR1600|900|MULTI:1:0:334:534ScRiPtSePaRaToR1600
0ScRiPtSePaRaToR1600|900|MULTI:1:2:334:533ScRiPtSePaRaToR1695
0ScRiPtSePaRaToR1600|900|MULTI:0:6ScRiPtSePaRaToR1695
0ScRiPtSePaRaToR1600|900|MULTI:0:6ScRiPtSePaRaToR1695
0ScRiPtSePaRaToR1600|900|MULTI:0:1ScRiPtSePaRaToR1695
0ScRiPtSePaRaToR1600|900|MSBRL:0:0ScRiPtSePaRaToR1695
0ScRiPtSePaRaToR1600|900|MULTI:1:0:216:410ScRiPtSePaRaToR2055
0ScRiPtSePaRaToR1600|900|MULTI:1:2:217:410ScRiPtSePaRaToR2119
```

그림 6. 녹스 에뮬레이터 사용자 반복 작업 파일의 내용
Figure 6. Content of User Macro File in Nox Emulator

안드로이드 에뮬레이터의 인스턴스 복제 및 동시 실행 기능은 실제로 범죄에 악용되기도 한다. 2018년 온라인 도박사이트 개설 및 운영 혐의로 압수된 피의자의 데스크톱 8대에 대한 포렌식 분석 결과 총 5대에 안드로이드 4.4.2(Kitkat) 환경의 에뮬레이터(녹스)가 설치되어 있었다. 특히 데스크톱 한 대당 4~9개의 복제된 인스턴스들이 있었으며, 압수된 데스크톱 5대에 총 40개의 인스턴스가 확인되었다. 안드로이드 환경에서 이루어진 범죄 행위 확인과 증거파일 발견을 위해 사용자의 데이터가 저장되는 가상디스크 파일(.vmdk)을 포렌식 분석 도구인 FTK Imager에서 마운트 시켜 분석하는 기법을 이용하였다. 각 인스턴스에 해당하는 40개의 가상디스크 파일을 분석한 결과, 피의자가 직접 설치한 앱 중 카카오톡 메신저와 텍스트 플러스(text plus) 등 두 종류의 앱이 여러 인스턴스에서 공통적으로 발견되었다. 텍스트 플러스는 문자 수신이 가능한 가상의 휴대폰 번호를 생성해 주는 앱이다. 이 때문에 텍스트 플러스 앱을 이용해 각종 메신저 계정 생성 등 사용자의 휴대폰 번호를 통한 문자 인증을 우회하기도 한다.

피의자는 카카오톡 메신저와 텍스트 플러스 앱이 설치되어 있는 안드로이드 에뮬레이터 인스턴스를 4~9개가량 복제한 뒤, 각 인스턴스에서 가상 휴대폰 번호를 이용해 카카오톡 메신저의 계정생성 문자 인증을 받는 방식으로 추적이 불가능한 계정을 생성하였다. 피의자들의 이러한 행위는 실제 카카오톡 대화 내용 분석을 통해 재차 확인할 수 있었다. 이와 같은 방법으로 복제된 인스턴스 수만큼 다수의 카카오톡 메신저 계정을 획득하고, 한 대의 데스크톱에서 여러 개의 계정을 동시에 운영할 수 있었다. 사용자가 설치한 앱의 데이터가 저장되는 SD카드 경로([root]\media\0)의 카카오톡 콘텐츠 데이터(Android\data\com.kakao.talk\contents)를 분석한 결과 Figure 7.과 같이 안드로이드 에뮬레이터상에 설치된 카카오톡 메신저를 이용해 도박 사이트 홍보 및 접속 유도에 사용된 승부식 복권 구매 영수증과 같은 각종 증거파일들이 발견되었다. 특히 카카오톡 대화방에서 동일인이 다수의 카카오톡 계정을 이용해 각종 수익 인증사진에 호응하거나 동조의견을 피력하는 방법으로 피해자들이 도박을 하도록 부추기고 도박 사이트 접속을 유도하였다. 어떤 가상디스크 파일의 카카오톡 데이터에서는 Figure 7.과 같이 도박 사이트 주소와 함께 범죄 총책을 특정할 수 있는 중요한 정보를 획득할 수 있었다.

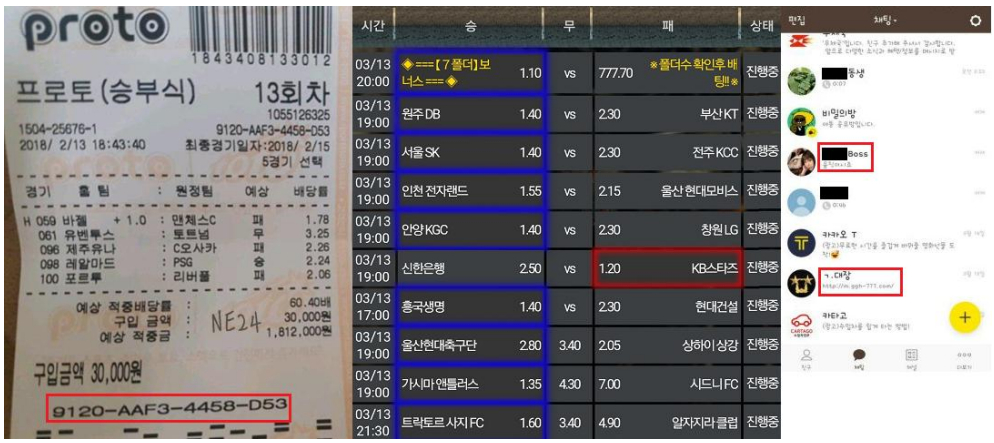

그림 7. 도박 사이트 홍보에 사용된 증거 이미지와 피의자를 특정할 수 있는 이미지 파일
Figure 7. Evidence Images Used for Illegal Gambling Website Touting and Image File Implying the Suspect

Ⅶ. 결 론

각종 스마트 기기의 보급과 모바일 시장의 급속한 성장 등 안드로이드 운영체제의 수요는 꾸준히 증가하고 있으며, 이와 함께 다양한 안드로이드 에뮬레이터의 출시와 이용도 지속적으로 늘어나고 있다. 그러나 사용자의 편의를 위해 제공되는 안드로이드 에뮬레이터의 기능들을 오용하거나 실제로 범죄에 악용하는 사례까지 생겨나고 있다. 안드로이드 에뮬레이터상의 증거들은 사용자의 데이터가 저장되는 가상디스크 파일 안에 저장되어 있기 때문에, 일반적인 디지털 포렌식 분석만으로는 발견이 불가능하다. 특히 블루스택과 같이 사용자 데이터가 저장되는 가상디스크 파일이 부모-자식 관계로 구성되어 있다는 것을 알지 못하면 해당 파일에 대한 분석 자체를 할 수 없다.

앞서 안드로이드 에뮬레이터 포렌식 분석을 통해 범죄 총책에 대한 정보를 획득할 수 있었던 사례처럼 사용자의 데이터가 저장되는 가상디스크 파일에서 수사상 결정적인 증거가 발견될 가능성이 있다. 따라서 데스크톱에 대한 디지털 포렌식 분석 시 안드로이드 에뮬레이터가 설치되어 있을 경우, 본 연구에서 제시한 안드로이드 에뮬레이터 포렌식 절차에 따른 관련 데이터 수집과 분석이 필요하다.

본 논문에서는 국내에서 가장 많이 사용되는 블루스택, 녹스, 미뮤 등 3종의 안드로이드 에뮬레이터의 구조를 분석하고 획득 가능한 아티팩트들을 확인하였으며, 이를 바탕으로 안드로이드 에뮬레이터 포렌식 절차를 제시하였다. 이렇게 확인된 아티팩트들과 분석 기법을 실제 범죄사건 분석에 적용해 봄으로써 유효성을 검증하였다.

본 연구를 통하여 기존에 거의 연구되지 않았던 안드로이드 에뮬레이터를 분석 및 정리함으로써 수사기관에서 이를 활용하여 다양한 사건 수사 시 도움이 될 수 있다고 판단된다. 향후 다양한 안드로이드 에뮬레이터에 대한 체계적인 분석과 연구가 추가적으로 필요하다.

References

[1] Mobile Operating System Market Share Worldwide. Available: https://gs.statcounter.com/os-market-share/mobile/worldwide

[2] Newzoo, Global Games Market Report[Online]. Available: https://newzoo.com/insights/trend-reports/newzoo-global-games-market-report-2019-light-version

[3] BlueStacks Website About Page. Available: https://www.bluestacks.com/ko/about-us/app-player.html

[4] "BlueStacks offers Galaxy Store… Expanding App Player Ecosystem", MoneyS, 2019.5.14. Available: https://moneys.mt.co.kr/news/mwView.php?no=2019051317158011437

[5] "Minwise launches China's No.1 App Player 'Tencent Gaming Buddy' in Korea", Moneytoday, 2019.4.22. Available: https://news.mt.co.kr/mtview.php?no=2019042213420669932

[6] "Remote code execution bug lurked in BlueStacks Android emulator", ZDNet, 2019.6.26. Available: https://www.zdnet.com/article/remote-code-execution-bug-lurked-in-bluestacks-android-emulator

[7] S. S. Lim, "Research on the investigation method of digital forensics for the VMware virtual machine", M.Sc. dissertation, Korea University, 2011

[8] S. H. Jang, "Digital forensic investigation of virtual desktop infrastructure", M.Sc. dissertation, Korea University, 2013

[9] J. H. Oh, "A Study for Android Smartphone Forensic Analysis", M.Sc. dissertation, Korea University, 2012

[10] J. S. Yoon, "A Study on android emulator detection for mobile game security", Journal of The Korea Institute of Information Security & Cryptology, Vol. 25, No. 5, pp. 1067-1075, 2015

[11] S. Y. Teng and C. Y. Wen, "A Study on Common Android Emulators and AntiForensic Message-Hiding Applications", Forensic Science Journal, Vol. 16, No. 1, pp. 3-15, 2017.

[12] 'Digital Evidence Analysis and Standards Process Guidelines', Korea National Police Agency, pp. 51, 2018

[13] "Recover data from .Vdi", Spiceworks, 2019.3.15. Available: https://community.spiceworks.com/topic/2193016-recover-data-from-vdi

[14] Android Recent Tasks XML Parser. Available: https://abrignoni.blogspot.com/2019/02/android-recent-tasks-xml-parser.html

[15] Android Developer Documentation - UsageEvents.Event. Available: https://developer.android.com/reference/android/app/usage/UsageEvents.Event

[16] "NCSOFT permanently suspended 0.4M of LineageM workplace accounts", Joins, 2017.7.14. Available: https://news.joins.com/article/21757590

저 자 소 개

남 우 환 (Woohwan Nam)

2010년 2월 : 인하대학교 졸업
2012년 7월 ~ 2013년 12월 : 외교부
2015년 7월 ~ 현재 : 서울지방경찰청
2018년 9월 ~ 현재 : 고려대학교 디지털포렌식학과 석사과정
관심분야 : 디지털 포렌식, 정보보호, 모바일 보안

KCI

KCI