JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지(J. Korea Inst. Inf. Commun. Eng.) Vol. 19, No. 6 : 1374~1379 Jun. 2015

# 좀비 스마트폰 특징 추출 및 대응기술 연구

이대성*

# A Study on Feature Extraction and Response Technology of Zombie Smartphone

**Daesung Lee***

Department of Computer Engineering, Catholic University of Pusan, Busan 609-757, Korea

## 요 약

DDoS와 같은 악성 네트워크 공격은 뚜렷한 대비책이 없으며, 그 피해 또한 막대하다. 특히, 스마트폰이 감염되어 좀비화 될 경우 통신요금 과금 및 개인정보 유출 등 네트워크 장애와 더불어 다양한 사용자 피해가 예상된다. 본 연구에서는 스마트폰이 악성코드에 감염되어 좀비 서비스가 실행되는 동안 나타나는 현상 및 특징들을 추출하고 좀비 스마트폰을 예방하는 대응기술을 소개한다.

## ABSTRACT

Malicious network attacks such as DDoS has no clear measures, the damage is also enormous. In particular, in addition to a network failure, such as leakage of personal information and damage of the communication charge in the case of zombie smartphone is infected, is expected damages of various users. In this study, we extract the zombie smartphone's phenomena and features that appear while the zombie service is running and introduce a corresponding technique to prevent zombie smartphone.

**키워드** : 좀비 스마트폰, 좀비PC, 분산 서비스 거부 공격, 특징 추출

Key word : Zombie Smartphone, Zombie PC, Distributed Denial of Service, Feature Extraction

**Received** 01 May 2015, **Revised** 30 May 2015, **Accepted** 08 June 2015

*** Corresponding Author** Daesung Lee(E-mail:dslee@cup.ac.kr, Tel:+82-51-510-0653)
Department of Computer Engineering, Catholic University of Pusan, Busan 609-757, Korea

Open Access http://dx.doi.org/10.6109/jkiice.2015.19.6.1374 print ISSN: 2234-4772 online ISSN: 2288-4165

## Ⅰ. 서 론

스마트폰은 지난 2004년 미국 내 사무직 종사자를 중심으로 열풍이 시작되어 미국 전역으로 확대 된 것에 이어 국내에도 2009년 4월부터 방송통신위원회의 WIFI 탑재 의무화 제도 폐지가 확정되면서 표준화된 개발 환경을 제공하는 범용 OS 기반의 스마트폰 시장이 확산되고 있다. PC수준의 기능을 제공하는 스마트폰의 경우 PC와 유사한 보안 취약점이 발생할 수 있다고 보안 전문가들은 우려하고 있는데, 이는 스마트폰이 단순한 휴대폰이 아닌 다양한 네트워크 인터페이스를 가진 컴퓨터로 현재 인터넷 환경에서 발생할 수 있는 모든 보안 위협이 스마트폰 환경에서 동일하게 적용되기 때문이다. 개인 중요 정보가 저장되고 오픈 개발 환경의 제공으로 자유로운 프로그램 개발 및 설치가 가능하다는 점에서 악성코드에 감염될 가능성이 크고, 이른바 좀비 스마트폰이라 불리는 악성코드에 감염된 스마트폰은 해커에게 제어권이 넘어간 경우 DDoS 공격, 악성 스팸 유포, 스파이웨어 설치, 개인정보 유출 등에 악용될 수 있다.

해커는 좀비스마트폰을 만들기 위해서 다양한 방법을 시도하고 있다. 육안으로 보이는 증상의 악성 어플리케이션을 활용한 방법, 루트(root)권한을 획득한 후 스마트폰 사용자의 개인정보 탈취 및 요금을 과금 시키는 현상 그리고 지능형 좀비 서비스를 사용하여 사용자의 스마트폰을 지배하는 등의 방법으로 좀비 서비스를 개발하고 유포시킨다[1, 2, 3, 10]. 본 연구에서는 스마트폰의 영역별 위협과 스마트폰 DDoS 공격의 특징을 알아봄으로써 좀비 스마트폰의 특징을 확인하고 안드로이드 apk 구성 요소에 대한 분석과 주요 클래스 및 안드로이드 어플리케이션 프로그래밍 절차에 대한 취약점 조사를 통해 안드로이드 스마트폰의 좀비화가 어떻게 진행되는지 파악할 것이다.

본 연구에서는 실제 100여개의 샘플 악성 어플리케이션을 순정상태의 스마트폰과 루팅된 스마트폰에서의 직접 설치 및 실행을 통해서 어떤 증상이 나타나는지 대표적으로 증상을 분류해 보고, 나아가, 샘플 악성 어플리케이션을 설치 및 실행을 통해 겉으로 드러나는 것들만으로는 분석이 어렵다 판단될 경우, 디컴파일을 통해 악성 어플리케이션의 클래스파일 내부의 소스코드를 분석, 좀 더 깊이 있는 연구를 진행하였다.

## Ⅱ. 관련 연구

본 장에서는 스마트폰이 악성코드에 감염되어 좀비화 되었을 경우 나타나는 다양한 특징과 앱 프로그램에 악성코드를 삽입하는 기존의 연구들에 대해 간략히 언급하고, 실제 안드로이드 앱을 작성할 때 좀비 서비스를 실행시키기 위한 안드로이드 프로그래밍 구조와 관련한 주요 클래스에 대해 살펴본다.

### 2.1. 좀비 스마트폰 연구

사용자 스마트폰이 악성코드에 감염되어 나타나는 대표적인 징후로는 개인정보 유출, 루트(root) 권한 탈취, 전화요금의 과금, 불필요한 앱의 반복적인 실행 등 다양한 현상에 대한 연구사례가 발표되었으며[1-3], 악성코드에 감염된 스마트폰이 원격으로 조정되는 좀비 스마트폰에 대한 사례와 이러한 좀비 스마트폰이 DDoS 공격에 악용될 수 있다는 연구도 발표되었다[4, 6, 7]. 또한, 악성코드를 삽입하는 주요 기법으로는 인기도가 높은 앱을 다운받아 기 작성된 앱에 악성코드를 삽입하고 다시 apk 파일로 패키징하는 리패키징 기법이 발표되었으며[4], 이렇게 리패키징된 악성 앱은 인기도에 높은 앱의 형태로 third-party에 다시 재 배포됨으로써 좀비 스마트폰의 확산을 가중시키는 사례들이 발표되었다[4, 5, 8].

### 2.2. 좀비 서비스 실행을 위한 안드로이드 주요 클래스 구조

본 절에서는 악성코드(좀비 서비스)가 스마트폰 사용자 모르게 살아남기 위해 요구되는 프로그래밍 기법과 관련한 주요 클래스를 살펴본다[9].

- Intent 클래스

Intent 클래스는 수행할 작업에 대한 전체적인 설명을 담고 있는 자료구조 클래스로 그 구조는 action(The general action to be performed), data(The data to operate on), category, type 등으로 구성되며, action은 Standard Activity Action과 Standard Broadcast Action으로 구분된다.

- Standard Activity Action

Activity[1]를 수행하기 위해 Intent 클래스가 결정하는 Action으로, startActivity(Intent) 형태로 호출한다.

- Standard Broadcast Action

시스템에서 발생하는 특정한 이벤트(배터리 부족, SMS 메지지 도착, 전화 등)에 대해 시스템 상에서 수행중인 모든 앱이 관심을 가져야 할 Action이다. 대표적인 Broadcast Action으로는 부팅이 완료된 후에 발생하는 “ACTION_BOOT_COMPLETED”, 시간 설정에 의해 시스템 시간이 변경되었을 때 발생하는 “ACTION_TIME_CHANGED” 등이 있다. 시스템 시간이 변경되면 모든 앱이 이 상황을 감지하고 필요에 따라 수정되어야 하기 때문에 브로드캐스트 형태로 전달된다.

- BroadcastReceiver 클래스

스마트폰 상에서 모든 앱이 상황을 인지해야 할 필요가 있는 많은 이벤트가 발생(배터리 부족, SMS 도착, 전화, 스크린 화면 꺼짐 등)하는 데 이러한 이벤트를 모든 앱이 직접 체크하는 것은 어려운 작업이기 때문에, 안드로이드에서는 해당 이벤트에 대해 처리하고자 하는 기능을 BroadcastReceiver 클래스를 통해 구현한다.

앱이 해당 이벤트를 받기 위한 설정은 2가지 방법이 있다.

- AndroidManifest.xml 에 <receiver> </receiver>를 설정해 주는 방법
- onReceive( ) 함수에서 해당 이벤트를 확인하는 방법

AndroidManifest.xml에서 IntentFilter에 등록된 Intent들은 onReceive 함수에서 호출되므로 그 Intent의 Action값에 따라 코딩을 해주면 된다. 예를 들어, SMS 메시지가 도착하면 시스템에서는 "android.provider.Telephony.SMS_RECEIVED" 라는 Broadcast Action을 방송한다. 이 Action을 Filter에 등록한 Receiver 클래스들은 onReceive( ) 함수에서 해당 Action을 사용할 수 있다. 해당 Receiver 클래스를 사용하기 위해서는 퍼미션을 사용하는 경우가 대부분이므로 확인이 필요하다.

- Service 클래스

background에서 수행되는 프로세스이며 사용자 화면을 갖지 않기 때문에 사용자 인터페이스가 제공되지 않는다. 예를 들어, 사용자가 다른 앱을 실행하는 동안 Service가 백그라운드에서 음악을 재생하거나, 아니면 사용자 개입 없이 네트워크를 통해 데이터를 가져올 수 있다. Service 클래스는 백그라운드에서 실행되므로 사용자 모르게 직접 또는 간접적으로 다른 앱을 실행시키는 것이 가능하다.

startService(Intent) 호출로 Service가 시작되며, onCreate( ) -> onStartCommand( ) 순으로 실행된다. onCreate( ) 함수는 Service가 만들어졌을 경우 실행될 로직을 기록한다. stopServiece(Intent) 호출로 Service가 종료하며, onDestroy( ) 함수가 호출된다. Service 사용을 위해서는 AndroidManifest.xml 내용에 <service> </service> intent-filter action 을 등록해야 한다.

## Ⅲ. 좀비 스마트폰 분석 실험 및 특징 추출

### 3.1. 좀비 스마트폰 분석 실험

본 연구에서는 100개의 악성 어플리케이션 샘플을 실행하고 그 수행결과를 분석하였으며, 경우에 따라서는 소스코드 분석을 통해 분석에 대한 정확도를 높였다. 분석결과는 [그림 1]과 같이 동적 서비스 등록 61%, 정적 서비스 등록 39%로 나타났으며, 동적 서비스 등록과 정적 서비스 등록 방식을 사용한 어플리케이션들을 각각 분류하였을 때, 각 서비스 내에서는 비슷한 수치를 보이고 있었다.

어플리케이션을 실행 시켰을 경우 눈에 보이는 증상과 코드 상에서의 특이사항이 없는 경우가 동적 서비스 등록이 61%, 정적 서비스 등록이 72%로 소폭 차이를 보이고 있었다. 그 외에도 악성 광고를 실행시키는 어플리케이션은 비슷한 수치로 각각 12%, 11%를 나타내고 있다.

홈 화면 실행 어플리케이션 생성은 동적 서비스 등록이 20%, 정적 서비스 등록이 11%로 차이를 보이고 있

1) 앱 실행 시에 사용자와 인터페이스 하는 일종의 foreground process

었는데, 이 차이는 생성된 실행 어플리케이션이 불법 사이트나 공식적이지 않은 사이트가 링크 되어 있어서 그에 해당하는 서비스, URL을 감추기 위해서 서비스를 동적으로 등록한 것이 많다고 볼 수 있다. 이어서 SMS Service에 해당하는 부분은 동적 서비스 등록과 정적 서비스 등록은 각각 5%, 3%, Alarm Service 등록은 동적 서비스 등록과 정적 서비스 등록이 각각 2%, 3%를 차지하고 있었다.

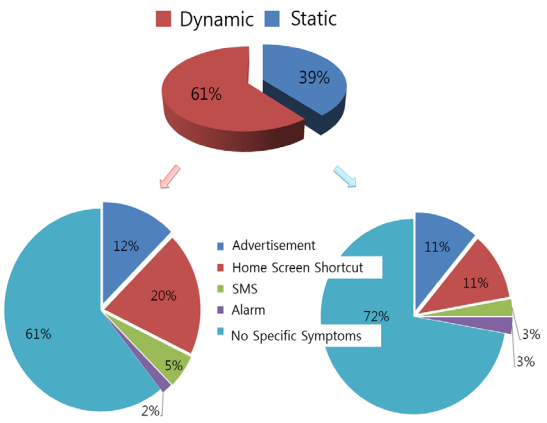


**그림 1.** 100개의 악성 앱 분석 결과
**Fig. 1** Malicious App Analysis Results(100 Samples)

동적 서비스 등록과 정적 서비스 등록의 목적은 같지만 그 방법적인 면에서 큰 차이를 보여 주고 있다. 은폐 엄폐시키기 좋은 동적 서비스 할당 방식이 악성코드 활용 면에서 빈도수가 높다고 판단할 수 있다. 그 이유는 동적 서비스 등록 방식은 AndroidManifest.xml 이외에도 기타 MainActivity, onReceiver, onCreate, onDestroy, onStart, onStop 등 다양한 함수에서 백그라운드 서비스 등록이 가능하며, 추가로 호출되는 다른 많은 함수들에서 좀비 프로세스가 실행되는 서비스를 호출해서 사용할 수 있기 때문인 것으로 분석되었다.

### 3.2. 좀비 스마트폰 특징 추출

악성 앱의 소스코드 분석을 통해 좀비 서비스로 실행될 수 있는 앱의 특징을 분석한 결과로는 첫째, 서비스가 사용자가 알지 못하도록 백그라운드에서 수행 되어야 하며 둘째, 사용자 강제 종료 등에 의해 종료되었을 경우, 다시 살아나기 위한 로직(Logic)이 구현되어 있는 것으로 분석되었다. 결과적으로 좀비 서비스를 제작하는 방법은 다양하지만, 위 2가지 요건을 충족시키면서 좀비 서비스를 제작하는 기본 구성은 아래와 같았다.

- 좀비 서비스를 사용자가 알지 못하도록 Service 클래스를 상속받아 background에서 실행

```
public class zombie extends Service { ... };
MainActivity.onCreate( ) {
        startService(zombie);
}
```

- 좀비 서비스가 종료할 경우 다시 살리기 위한 로직 구현, 다시 살리는 로직(Logic)은 Broadcast Action을 이용해 특정 이벤트가 발생했을 때 그 이벤트에 대해 startService(좀비 Intent)를 호출

- Broadcast Action 특정 이벤트에 대해 준비할 BroadcastReceiver 자식 클래스를 만들고, 좀비 서비스가 종료하게 될 때 호출되는 onDestroy( ) 함수에서 이 자식 클래스가 특정 이벤트를 받을 수 있도록 준비(예를 들어, 알람 등이 일정 간격으로 자동으로 발생하도록 준비)

```
public class call_zombie extends
BroadcastReceiver { ... };
zombie.onDestroy( ) {
        10초 후 알람 발생 시킴;
 }
```

- 자식 클래스는 Broadcast Action 특정 이벤트(알람 등)에 반응하게 되어 있으므로 이벤트가 발생하면, 이때 startService(좀비 Intent)를 호출하여 좀비 서비스를 다시 시작

```
call_zombie.onReceive( ) {
        if (알람 발생) startService(zombie);
}
```

앞서 설명한 좀비 서비스 동작 방식을 그림으로 요약하여 나타내면 [그림 2]와 같다.

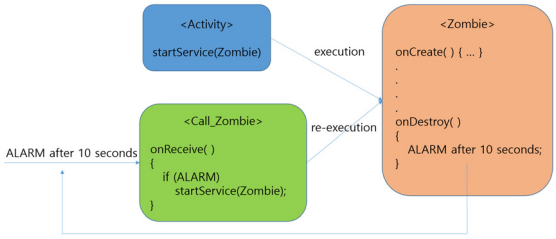


**그림 2.** 좀비 서비스 동작 방식
**Fig. 2** Zombie Service Behavior

## Ⅳ. 좀비 스마트폰 대응 기술

3장에서 살펴본 바와 같이 좀비 서비스는 사용자 모르게 실행되는 경우가 대부분이며, 종료가 될 경우 다시 실행되기 위한 로직을 구현하고 있다. 따라서 본 장에서는 소스코드 분석을 통해 파악된 좀비 서비스 실행 로직을 기반으로 스마트폰 상에서 악성코드나 좀비 서비스가 실행되고 있는 지를 판단하고 그 실행을 차단하기 위한 방법에 대해 살펴본다.

### 4.1. Intent – Broadcast Action을 이용한 좀비 서비스 실행 확인 및 차단

SMS 수신, 알람 등과 같은 Broadcast Action 이벤트를 이용하여 좀비 서비스가 제작되는 경우는 Broadcast Action에 관심이 있는 해당 앱이 BroadcastReceiver 클래스로부터 상속을 받아 정의 되어야 하며, 그 앱이 해당 이벤트를 받기 위한 설정은 AndroidManifest.xml파일에 [그림 3]과 같이 설정되어 있다.

```
<receiver android:name="com.Cpon.tool.AlarmReceiver">
  <intent-filter>
    <action android:name="android.intent.action.BOOT_COMPLETED"/>
     // Broadcast Action will be happened after boot is completed
    <action android:name="com.Copon.tool.alarm.ACTION_ALARM"/>
     /* ACTION_ALARM will be used to perform specific tasks at
        the desired time */
  </intent-filter>
</receiver>
```

**그림 3.** AndroidManifest.xml 파일 내용
**Fig. 3** AndroidManifest.xml file information

[그림 3]과 같은 설정의 경우 Intent-filter에 등록된 Intent들은 onReceive( ) 함수에서 startService(좀비 Intent) 형태로 좀비 서비스가 실행될 가능성이 높으므로 코드 확인을 통해 좀비 서비스 실행 여부를 판별할 수 있다. 특히, 100개의 악성 앱 분석과정에서 "BOOT_COMPLETED" Broadcast Action을 이용한 악성코드가 다수 발견되었으므로 이러한 경우는 시작되는 Intent 확인을 통해 좀비 유무를 확인할 수 있다.

### 4.2. 동적으로 등록되는 좀비 서비스 실행 확인 및 차단

서비스 등록을 런타임 시에 동적으로 등록하는 방법이 있다. 이때 사용하는 함수가 registerReceiver (Receiver, IntentFilter)이며, 사용 예는 아래와 같다.

```
// Intent 필터 생성
IntentFilter smsFilter = new IntentFilter( );
smsFilter.addAction("SMS_RECEIVED");
// Broadcast Receiver 등록
registerReceiver(call_zombie, smsFilter);
```

위 코드의 경우 Broadcast Receiver 인 call_zombie는 onReceive( ) 함수에서 SMS 이벤트에 반응하게 되며, [그림 2]에서와 같이 startService(zombie)를 호출하여 좀비 서비스를 다시 시작하게 된다. 따라서, registerReceiver( ) 함수를 검색하고 등록되어 있는 IntentFilter와 Broadcast Receiver 확인을 통해 좀비 서비스 등록 유무를 파악할 수 있다.

## Ⅴ. 결 론

PC의 DDoS 공격으로부터 시작되어 좀비PC가 생긴 이후로 DDoS 공격은 금융, 방송국, 공공기관 사이트 등을 공격하여 대규모 피해를 일으키고 있다. PC 사용자보다 스마트폰 사용자가 점차 늘어나고 있는 이 시점에서 스마트폰의 보안대책은 아직까지 미비한 상태이다. 스마트폰 상에서도 DDoS 공격뿐만 아니라 악성 앱을 통해서 사용자에게 피해를 발생 시키는 사고사례는 계속 늘어나고 있다.

본 연구에서는 100개의 보고된 악성 앱 샘플을 스

마트폰에 직접 설치해서 실행하고, 실행의 경과를 지켜보면서, 디컴파일을 통해 코드분석을 해본 결과, 소스코드의 다양한 함수에서 좀비 서비스가 백그라운드 형태로 다시 재실행 되는 구조를 구성하고 있는 것으로 대량의 악성 앱 분석을 통해 확인하였다. 이를 기반으로 앱 실행만으로는 판단하기 모호한 좀비 서비스를 소스코드 수준에서 명확하게 판단하는 방법을 제시하였다. 본 연구는 앱의 실행 과정에서 서비스 실행 사이클을 동적으로 분석하고 좀비 서비스 실행여부를 판단하는 향후 악성코드 탐지 연구에 많은 기반 지식을 제공한 것으로 판단된다.


## ACKNOWLEDGMENTS

This paper was supported by RESEARCH FUND offered from Catholic University of Pusan


## REFERENCES


[ 1 ] Ki-Hun Jang, “A Study on Smartphone DDoS Attack Detection and Response System via Source-End Based Packet Monitoring”, M.S Degree. dissertation, Soonchunhyang Univ., 2012.

[ 2 ] Ho-Kyu Maeng, Tae-Won Oh, “Model for Personal Information Protection from Smartphone Synchronization ”, *Korea Computer Domestic Conference* Vol 38, No.1(D). pp 44~47, 2011.

[ 3 ] Ki-Hun Jang, Sang-Myung Choi, Heung-Yul Hyum, “The Attack Trend of Smartphone DDoS”, *Journal of Korea Institute of Information Security* Vol. 21, No. 5, pp 65 ~ 70, 2011.

[ 4 ] Sang-Myung Choi, “Zombie Smartphone and DDoS Attack,” Hauri, Technical Report, 2012.

[ 5 ] National Information Society Agency, “Security Issue and Response Strategy of Smartphone and Mobile Office,” CIO Report, 2010.

[ 6 ] Arun Raj Kumar, P. and S. selvakumar, “Distributed Denial of Service(DDoS) Threat in Collaborative Environment A Survey on DDoS Attack Tools and Traceback Mechanisms”, *IEEE International Advance Computing conference(IACC 2009)*, Mar, 2009.

[ 7 ] Ki-Hun Jang, Heung-Yul Hyum, “The Intrusion Path and Threat Analysis of Smartphone”, *Korea Institute of Information Security Autumn Conf.*, pp 238 ~ 244, Oct, 2010.

[ 8 ] Yong-Hui Jun, Jong-Soo Jang, Jin-Tae Oh, “DDoS Attack and Response Method”, *Korea Institute of Information Security*, Vol. 19, No. 3, pp 46 ~ 57, June, 2009.

[ 9 ] Jae-Nam Woo, Bok-Ki Lee, Android Programming, 1st ed. Hanbit Media, Inc. 2012.

[10] http://www.gartner.com/technology/home.jsp



### 이대성(Daesung Lee)

1992.03 ~ 1999.02 인하대학교 전자계산공학과 공학사
1999.03 ~ 2001.02 인하대학교 전자계산공학과 공학석사
2001.03 ~ 2008.02 인하대학교 정보공학과 공학박사
2008.03 ~ 2012.02 경기대학교 정보보호학과 연구교수
2012.03 ~ 현재 부산가톨릭대학교 조교수
※관심분야 : 클라우드 보안, 사물인터넷 보안, 운영체제 보안