정적분석 결과를 이용한 안드로이드 악성 앱 분석을 향해

김진영
서울대학교 프로그래밍 연구실

제 12회 ROSAEC 워크샵
2015년 1월 27일

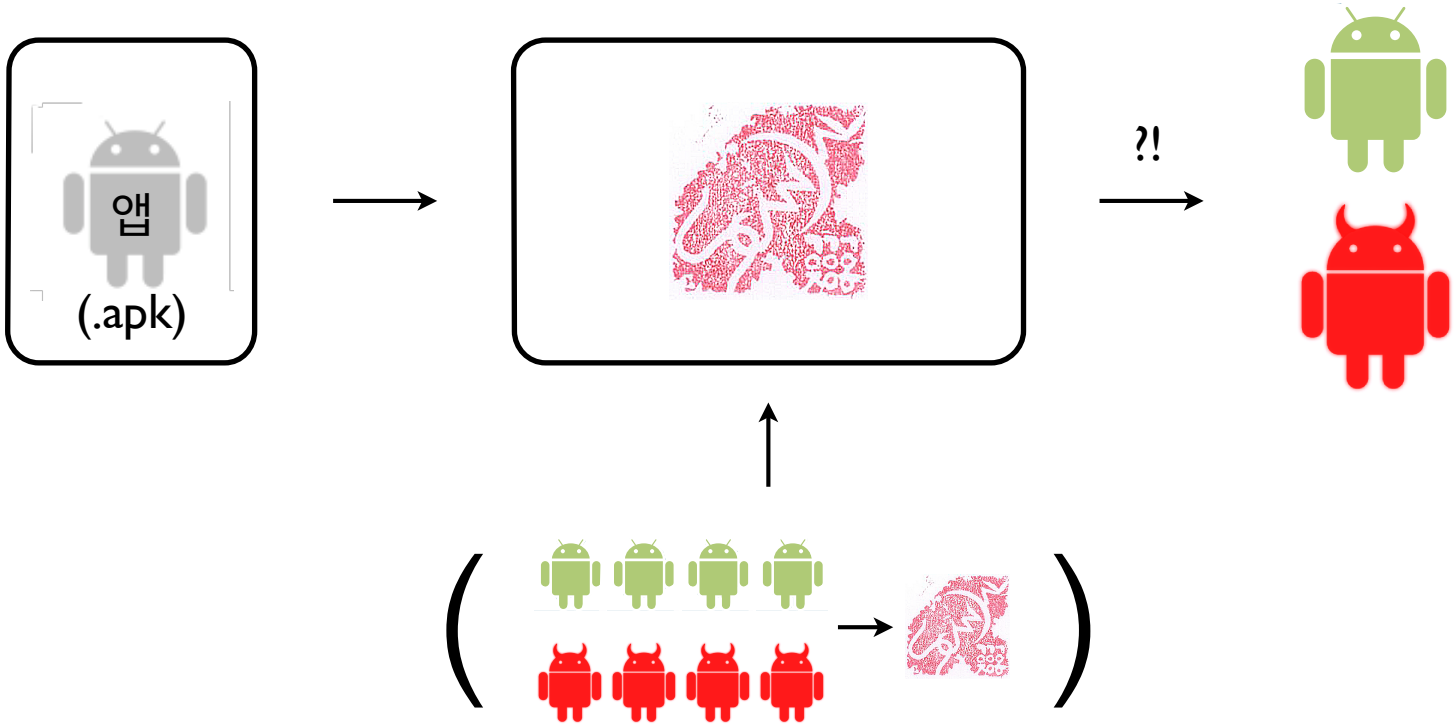
배경

- ScanDal: 안드로이드 앱 요약해석기
- 개인정보 유출 분석
- (소스, 싱크) 경보 결과

배경

- ScanDal: 안드로이드 앱 요약해석기
- 개인정보 유출 분석 → 다른 악성 행동?
- (소스, 싱크) 경보 결과 → 진짜 악성 유출? 경보의 진위?

큰그림



목표

- 악성 앱 판독기
  - 개인정보 유출 이상으로
  - 악성인지 아닌지 가를 수 있는 특징 바탕
  - 알려진 악성 앱과 일반 앱으로 공부
  - ScanDal의 정적분석 결과 이용

어떤 앱으로

- 살피고 공부시킬 앱
  - 알려진 악성 앱 1260개*
  - 마켓 무료 인기 앱 & 갤럭시 내장 앱 516개

- 많은 앱
  - 악성 앱 2만여개**
  - 수많은 마켓 무료 앱

*Malware Genome Project **VirusShare Database

개인정보 흐름 - API

- 분석결과에서 개인정보가 흐르는 모든 API 호출 빈도
- 일반 앱에서 빈번
  - 로그 남기기 `Log.d(), Log.i()` …
  - Empty 검사 `TextUtils.isEmpty()` …
- 악성 앱에서 빈번
  - Intent로 값 넘기기/가져오기 `Intent.putExtra()` …
  - 문자열로 URL 만들기 `URLEncoder.encode()` …
  - 런타임 명령어 실행 `Runtime.exec()`

개인정보 흐름 - Object

- 분석결과에서 개인정보가 흐르는 Object 단위 빈도
- 악성 앱에서 매우 빈번
  - `Service` 클라스
    - 백그라운드 동작 담당
  - `SharedPreferences` 클라스
    - 앱의 설정 정보 담당

개인정보 흐름 - 맥락

- 분석결과에서 개인정보의 읽고 쓰기가 일어나는 맥락
- 악성 앱에서 매우 빈번
  - 기기 부팅 직후 `BOOT_COMPLETED`
  - 첫 번째 액티비티의 생성 `onCreate()`
  - 타이머가 설정된 태스크 `TimerTask`

다른 악성행동과 맥락

- 특정 이벤트 하에서 일어나는 악성행동
- 메시지 수신 후 `SMS_RECEIVED`
  - 브로드캐스트 가로채기 `abortBroadcast()`
- 전화를 걸 때 `NEW_OUTGOING_CALL`
  - 네트워크 접근
- 다양한 의심스러운 악성행동을 (맥락, API호출)로 표현 가능

기타 악성 앱에서 더 빈번한 API

- 기기 및 기기에 설치된 모든 패키지 정보 획득
- 작동중인 액티비티/서비스 정보 획득
- 런타임 명령어 실행
- Stream 획득
- 달빅 Class 직접 적재
- 백그라운드 프로세스 죽이기
- 앱 재시작
- 문자열 암/복호화
- SMS 메시지 획득
- SMS 메시지 전송
- …

리플렉션

- 메소드 리플렉션 `Java.getMethod()`
  - 문자열을 인자로 받아 해당 이름의 메소드 호출
  - 지금까지 앱 정적 분석에 걸림돌 중 하나

- 리플렉션 사용 빈도는 악성 앱과 일반 앱이 비슷
- 인자가 Top으로 분석되는 경우는 대부분 악성앱에서만
  - 일반앱에서는 1%

문자열

- 악성 앱에서 빈번한 문자열
  - 위험할 수 있는 명령어 `su, pm install, root, mount, …`
  - 위험할 수 있는 경로 `/sh, /bin, /dev/net/, …`
  - 스트림 `stdout, stderr, stdin`
  - 유니코드 상수
  - 알려진 exploit의 이름
- 정적분석 중 생성되는 모든 문자열을 관찰할 것

정리

- 악성 앱 판독기
  - 악성인지 아닌지 가를 수 있는 특징들로 공부시켜 판독
  - 특징은 정적분석의 결과를 활용

감사합니다