ScanDal:
개인정보 누출 분석기에서 악성 앱 분석기로

김진영 윤용호 이승중
서울대학교 ROPAS

제 8회 ROSAEC 워크샵
2012년 7월 27일

개요

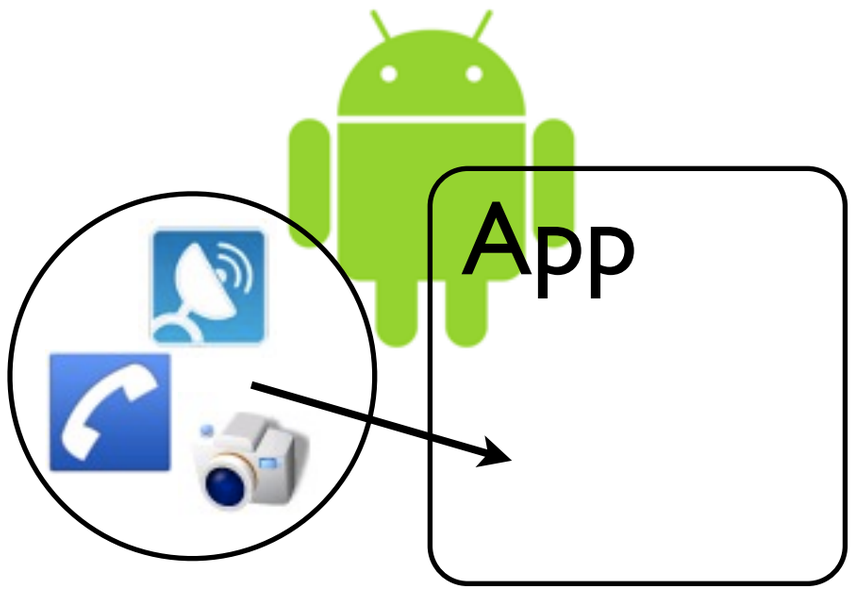
- 한 일
  - 개인정보 누출 분석기 (SPW MoST '12)
- 할 일
  - 악성 앱 분석기

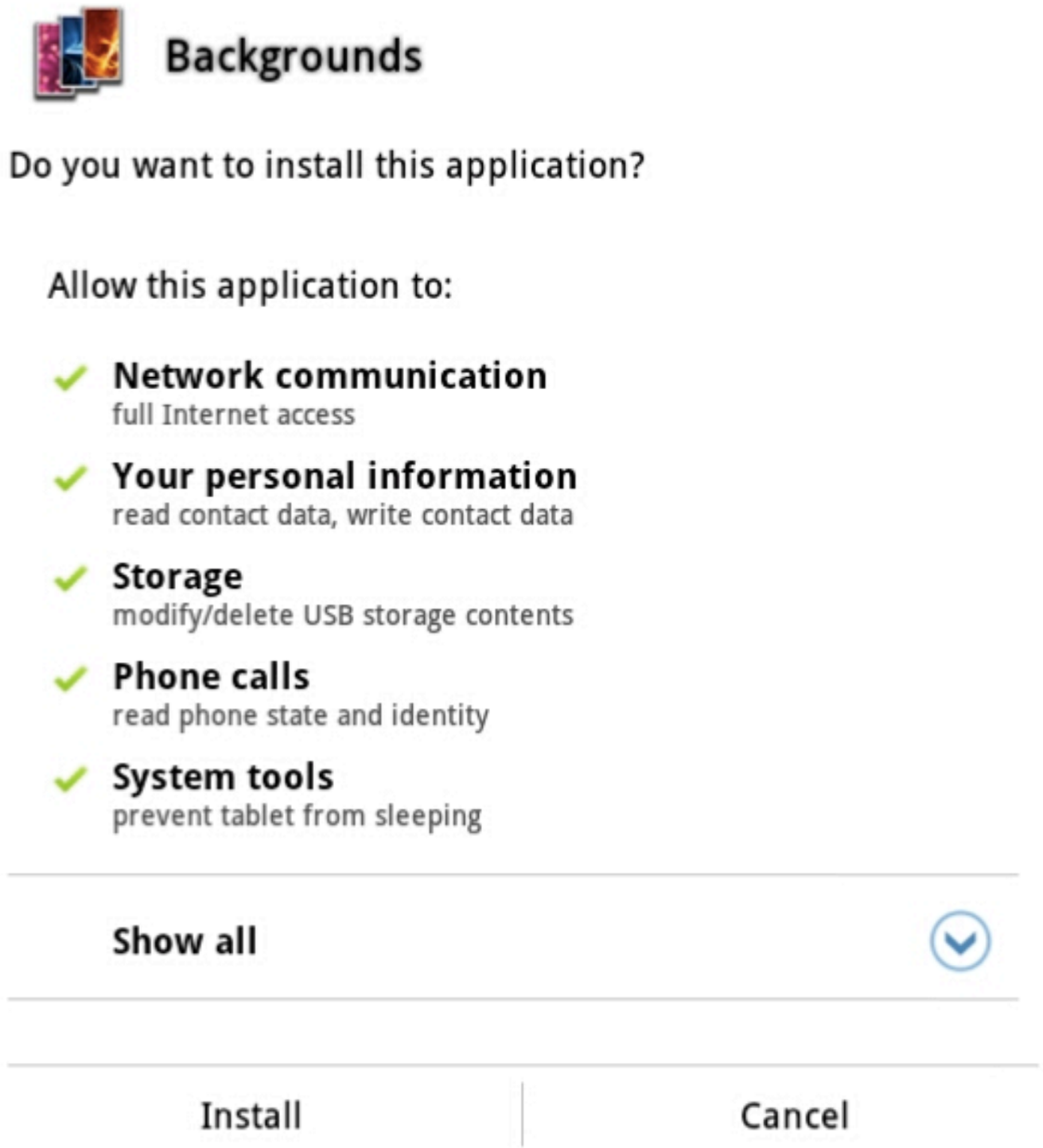
문제

- 안드로이드 앱의 개인정보 누출
- 단말기 식별번호, 위치정보 등의 사례
- 현재는 권한 체계

권한

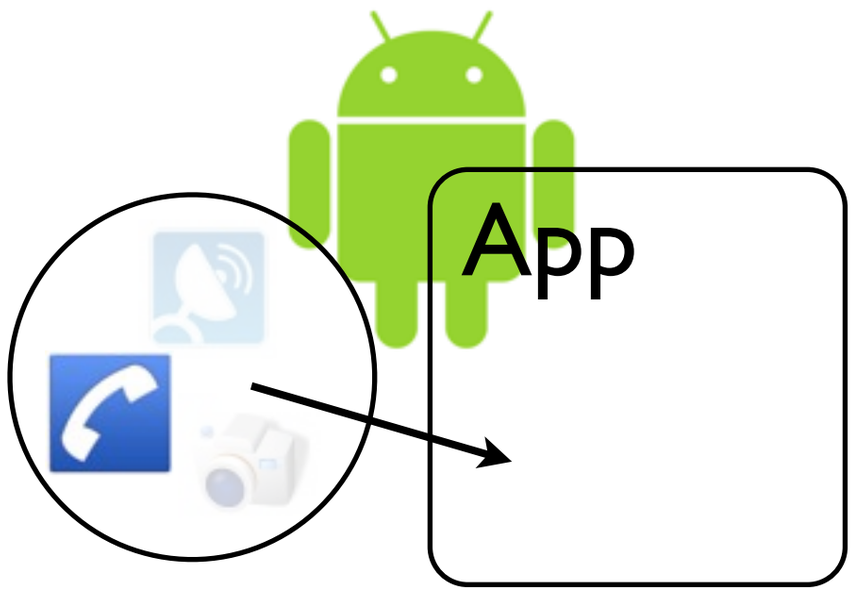
- 과도한 선언
- 포괄적 정의
- 맥락이 없음

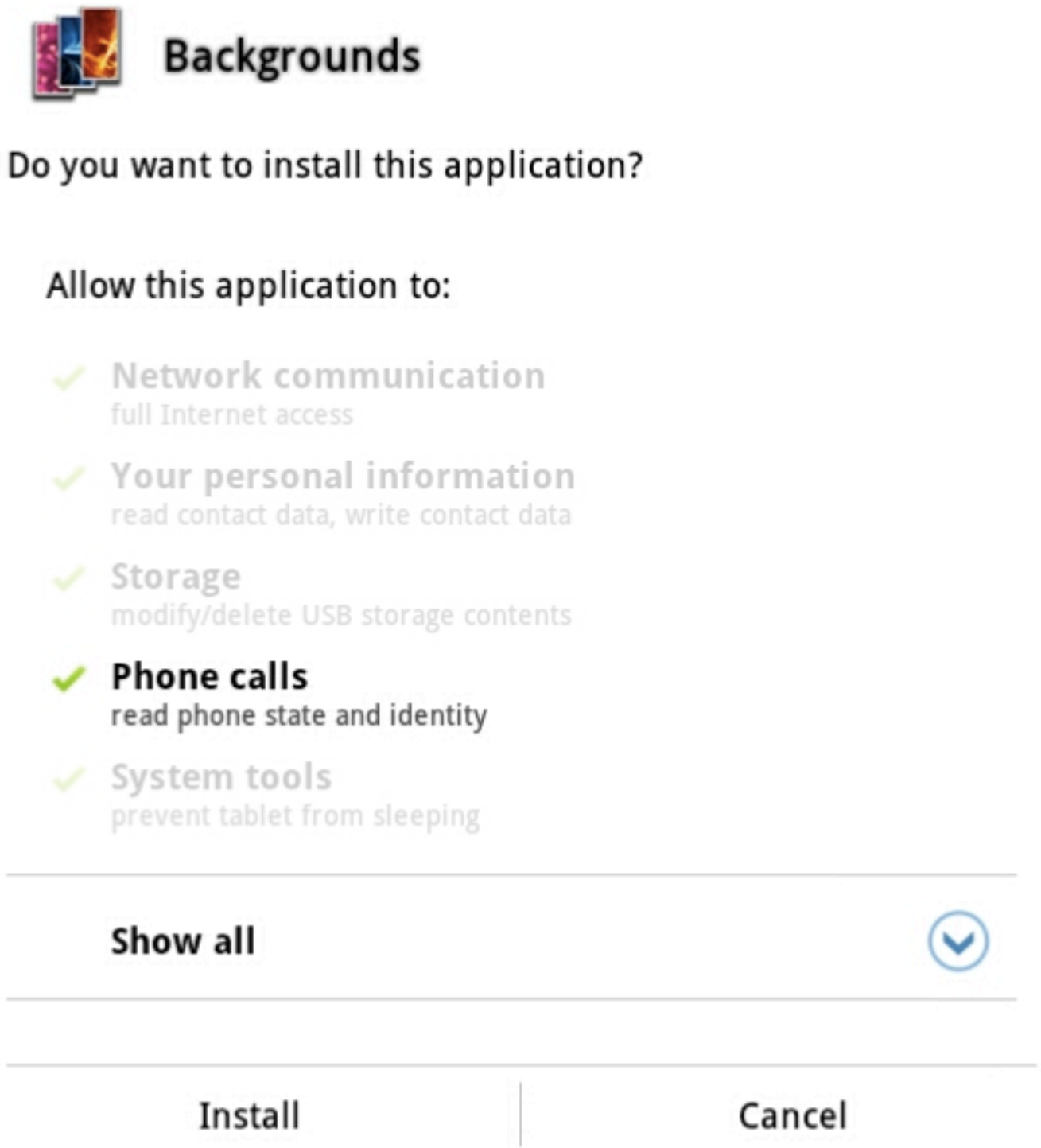




권한

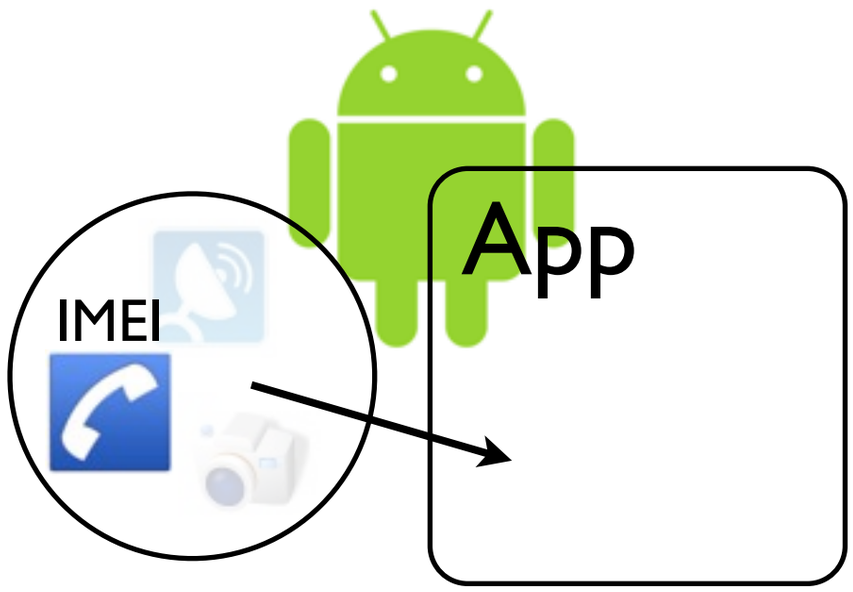
- 과도한 선언
- 포괄적 정의
- 맥락이 없음

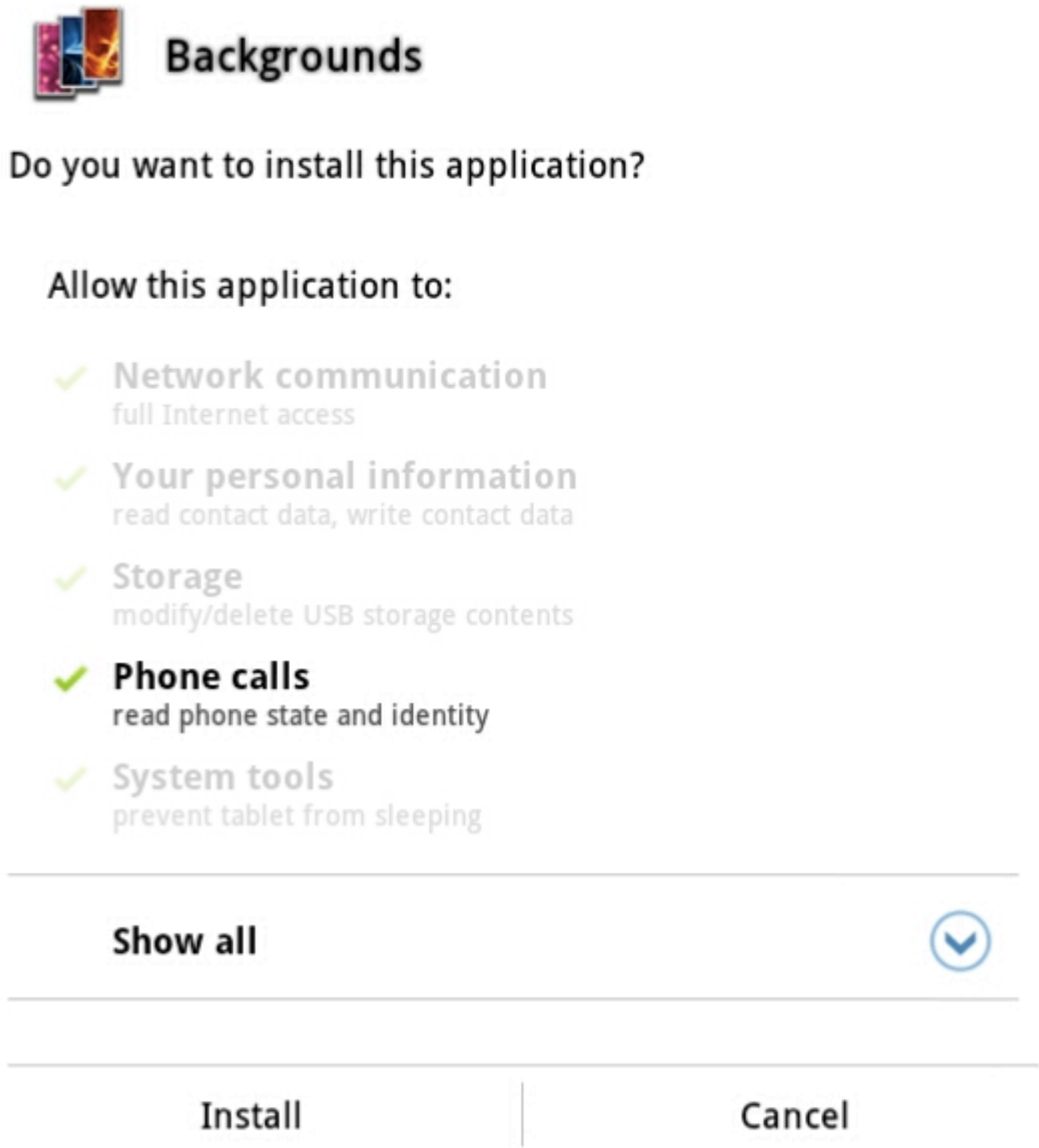




권한

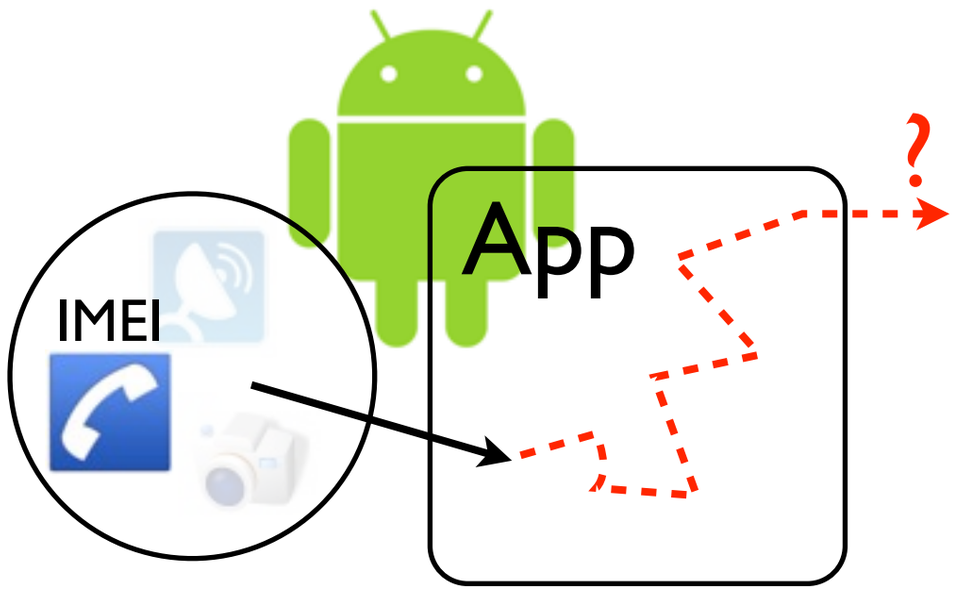
- 과도한 선언
- 포괄적 정의
- 맥락이 없음

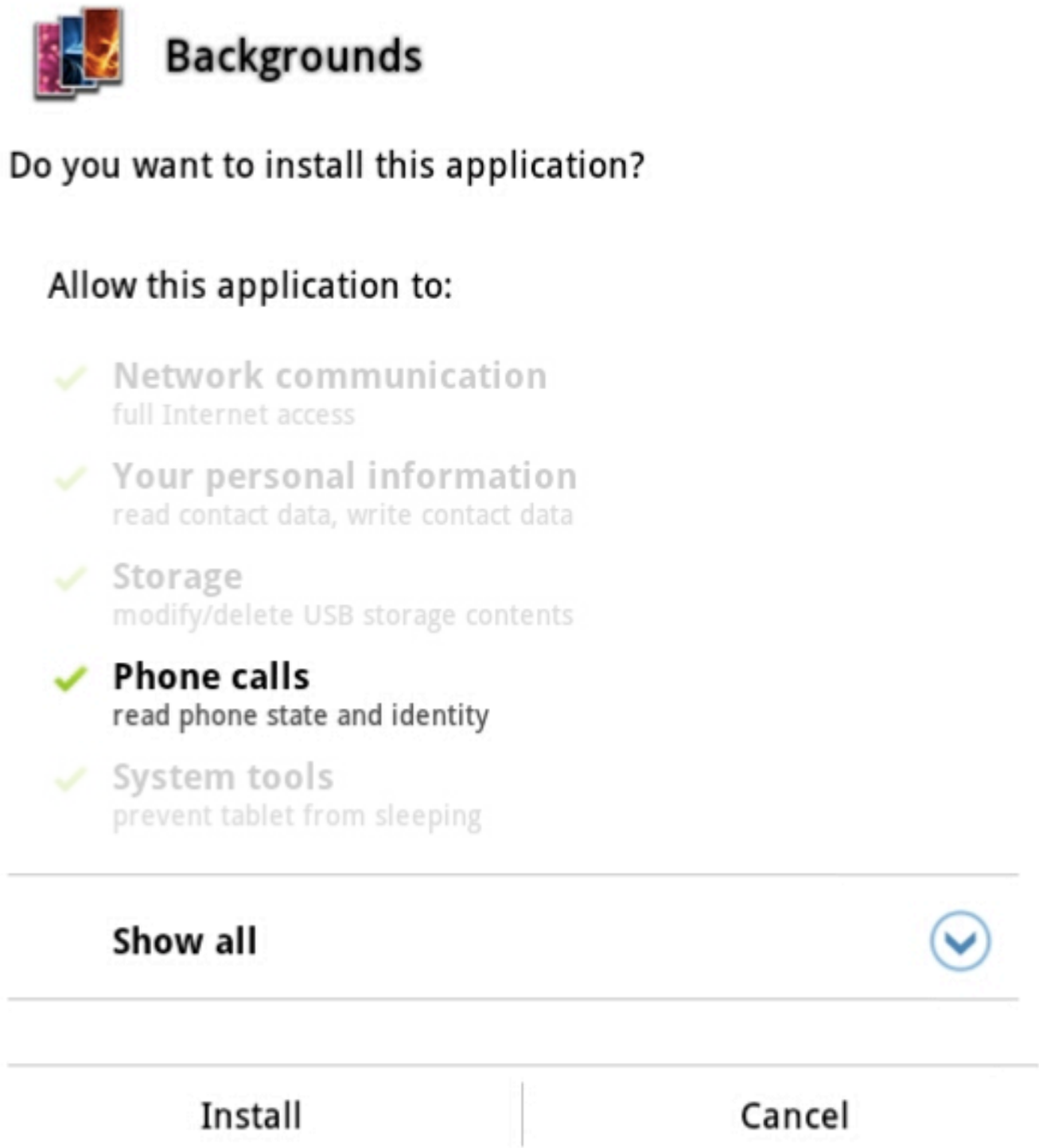




권한

- 과도한 선언
- 포괄적 정의
- 맥락이 없음





해결방법

- ScanDal
  - 요약해석 기반의 정적분석기
  - 앱 패키지 파일을 스캔하여 (.apk)
    - Dalvik 바이트코드
  - 개인정보 누출 여부를 분석

개인정보 누출

- 개인정보 소스 API로부터
  - 위치정보 - `getLastKnownLocation()`,...
  - 기기식별번호 - `getDeviceId()`,...
- 싱크 API까지
  - 네트워크 - `loadUrl()`,...
  - 네트워크/파일 - `write()`,...
  - SMS - `sendTextMessage()`,...

실험결과 (1/2)

- 안드로이드 공식 마켓 무료 인기 앱 90개
  - 11개 앱에서 누출 검출

Application	size (KB)	time (s)	mem (MB)	source	sink
Kids Preschool Puzzle	87	1	62	Location	File
Job Search	167	1	95	Location	Net
Kids Shape	225	2	137	Location	File
Kids ABC Phonics	134	3	109	Location	File
Backgrounds HD Wallpapers	109	4	133	IMEI	Net
Bible Quotes	138	8	265	Location	Net
ES Task Manager	158	20	424	Location	Net
Multi Touch Paint	198	42	718	Location	File & Net
Adao File Manager	255	67	1143	Location	Net
(D-Day) The Day Before	293	225	2648	Location	Net
Kids Number and Math	101	559	176	Location	File

실험결과 (2/2)

- 알려진 악성 앱 8개

Application	size (KB)	time (s)	mem (MB)	source	sink
Shot Gun	95	36	164	Phone # IMEI IMSI ICC-ID Location	Network
Baseball Superstars 2010	165	61	285		
CacheMate for Root Users	174	67	242		
Monkey Jump 2	169	74	442		
Protector	107	75	209		
Gold Miner	191	81	481		
Mini Army	480	174	1292		
Xing Metro	253	23049	1784		

실험결과 (2/2)

- 알려진 악성 앱 8개

Application	size (KB)	time (s)	mem (MB)	source	sink
Shot Gun	95	36	164	Phone # IMEI IMSI ICC-ID Location	Network
Baseball Superstars 2010	165	61	285		
CacheMate for Root Users	174	67	242		
Monkey Jump 2	169	74	442		
Protector	107	75	209		
Gold Miner	191	81	481		
Mini Army	480	174	1292		
Xing Metro	253	23049	1784		
Monkey Jump 2 (Original)	79	2	73	-	-

Repackaged

예제

Google Wallpapers 4.2.2

Timeline

```
Wallpapers.onCreate(Bundle)
   r=TelephonyManager.getDeviceId()
   eWallpaperConst.IMEI=r
```

Store IMEI

Memory

예제

Google Wallpapers 4.2.2

Timeline

Memory

```
SearchTagsActivity.initTagWebView()
   v=this.mWebView
   p=this.mSharedPreferences
   r=XMLTools.getSearchURL(p)
   WebView.loadUrl(v,r)
```

예제

Google Wallpapers 4.2.2

Timeline

Memory

```
SearchTagsActivity.initTagWebView()
   v=this.mWebView
   p=this.mSharedPreferences
   r=XMLTools.getSearchURL(p)
```

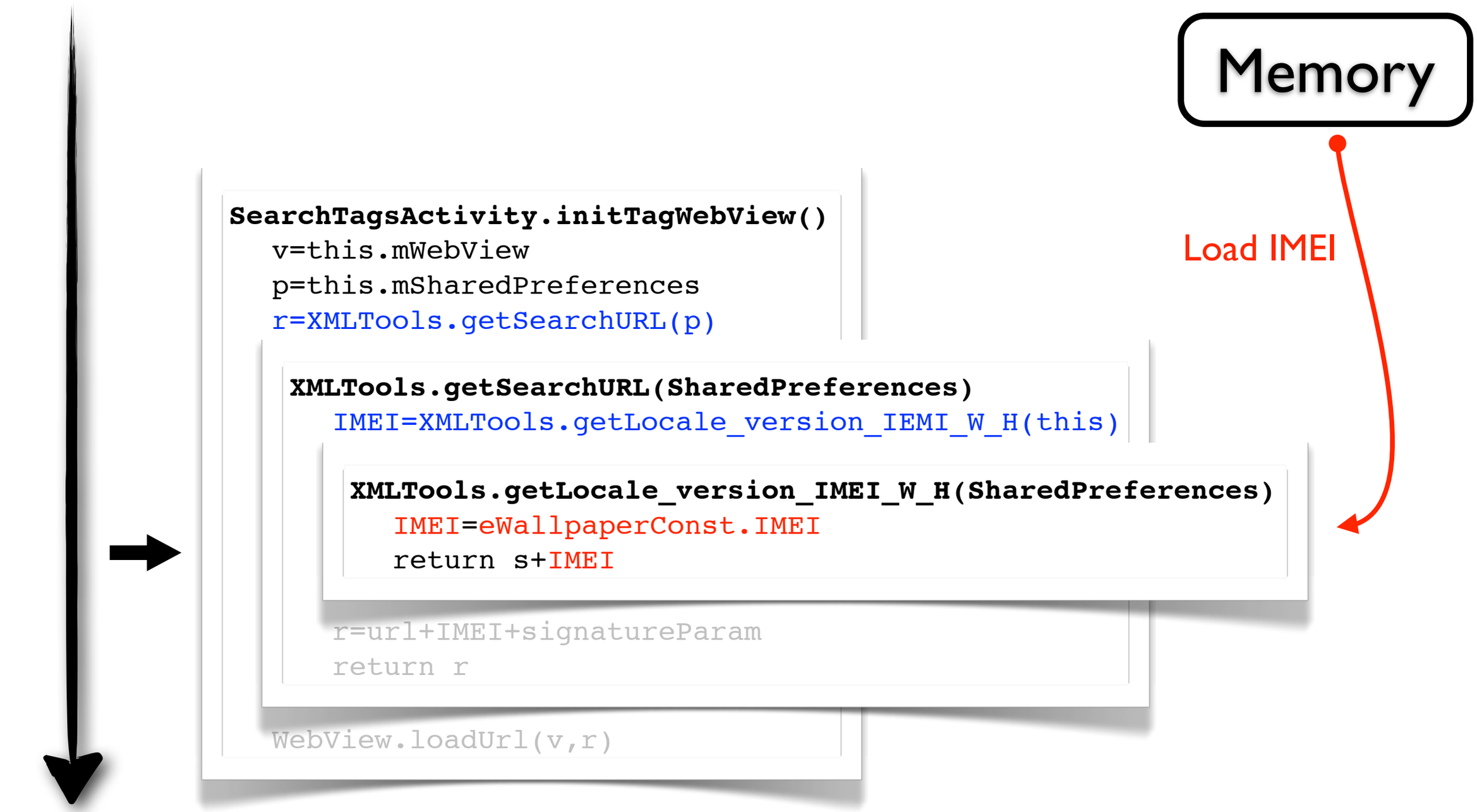
```
XMLTools.getSearchURL(SharedPreferences)
   IMEI=XMLTools.getLocale_version_IEMI_W_H(this)
   r=url+IMEI+signatureParam
   return r
```

```
   WebView.loadUrl(v,r)
```

예제

Google Wallpapers 4.2.2

Timeline



예제

Google Wallpapers 4.2.2

Timeline

Memory

```
SearchTagsActivity.initTagWebView()
   v=this.mWebView
   p=this.mSharedPreferences
   r=XMLTools.getSearchURL(p)
```

```
XMLTools.getSearchURL(SharedPreferences)
   IMEI=XMLTools.getLocale_version_IEMI_W_H(this)
```

```
XMLTools.getLocale_version_IMEI_W_H(SharedPreferences)
   IMEI=eWallpaperConst.IMEI
   return s+IMEI
```

```
   r=url+IMEI+signatureParam
   return r
```

```
   WebView.loadUrl(v,r)
```

예제

Google Wallpapers 4.2.2

Timeline

```
Wallpapers.onCreate(Bundle)
   r=TelephonyManager.getDeviceId()
   eWallpaperConst.IMEI=r
```

Memory

```
SearchTagsActivity.initTagWebView()
   v=this.mWebView
   p=this.mSharedPreferences
   r=XMLTools.getSearchURL(p)
```

```
XMLTools.getSearchURL(SharedPreferences)
   IMEI=XMLTools.getLocale_version_IEMI_W_H(this)
```

```
XMLTools.getLocale_version_IMEI_W_H(SharedPreferences)
   IMEI=eWallpaperConst.IMEI
   return s+IMEI
```

```
   r=url+IMEI+signatureParam
   return r
```

```
   WebView.loadUrl(v,r)
```

flow detected

예제

Google Wallpapers 4.2.2

Timeline

```
Wallpapers.onCreate(Bundle)
   r=TelephonyManager.getDeviceId()
   eWallpaperConst.IMEI=r
```

Memory

```
SearchTagsActivity.initTagWebView()
   v=this.mWebView
   p=this.mSharedPreferences
   r=XMLTools.getSearchURL(p)
```

```
XMLTools.getSearchURL(SharedPreferences)
   IMEI=XMLTools.getLocale_version_IEMI_W_H(this)
```

```
XMLTools.getLocale_version_IMEI_W_H(SharedPreferences)
   IMEI=eWallpaperConst.IMEI
   return s+IMEI
```

```
   r=url+IMEI+signatureParam
   return r
```

```
   WebView.loadUrl(v,r)
```

"http://www.imnet.us/api/wallpapers/photos/search_keywords?"+IMEI+SignatureParamString

ScanDal 큰그림

개인정보 누출 분석기

- 앱 패키지 파일 (.apk) 을 입력으로
  - Java 코드 없이도 분석
    - 대상 언어는 Dalvik VM Bytecode
      - classes.dex (DexDump)
- XML 정보도 분석에 필요
  - Manifest, Layout

- Dalvik VM Bytecode
  - 220여개 명령어
    - 너무 많고 하는 일 비슷
  - 코드에 명시되지 않는 정보
    - 안드로이드 OS가 해결해 주는

- 적절한 번역 & 모델링 필요

- "Dalvik Core"
  - 분석용 핵심언어
- 15개 명령어
- 메소드, 익셉션, 섭타입
- 의미구조 정의
- 번역 정의

$$
\begin{array}{rcl}
instr & ::= & \texttt{move } e\ e \\
& | & \texttt{istype } e\ e\ ty \\
& | & \texttt{new } e\ ty \\
& | & \texttt{get } e\ e\ id \\
& | & \texttt{put } e\ e\ id \\
& | & \texttt{gets } e\ id \\
& | & \texttt{puts } e\ id \\
& | & \texttt{addcallback } id\ e \\
& | & \texttt{call } ty\ id\ e^* \\
& | & \texttt{vcall } id\ e^+ \\
& | & \texttt{return} \\
& | & \texttt{throw } e \\
& | & \texttt{jmpnz } e\ pc \\
& | & \texttt{switch } e\ (e, pc)^+ \\
& | & \texttt{wait}
\end{array}
$$

- 코드에 명시되지 않는 정보
  - Activity 시작점, View들의 ID, ...
    - XML에서 읽어옴
  - Java 라이브러리 함수들
    - 적절히 인코딩
  - 안드로이드 구성요소들의 실행 흐름
    - 2단계 실행모델: **초기화** - (이벤트/콜백)*

- 요약실행
  - 고정점 계산으로 요약메모리 계산
    - 실제 실행을 모두 포섭하게
- Dalvik Core에 대해 안전
- 대부분의 Domain 단순
  - Flat 또는 Power Set

- 싱크 API의 인자로 넘어온 값 중
  - 소스 API로부터 생성된 값이 있는지 확인

확장

- 새로운 악성 앱은 계속 등장
  - 모바일 멀웨어의 절반가량은 안드로이드
  - 2011년 하반기 안드로이드 악성 앱은 상반기 대비 33배 증가
- 개인정보 누출보다 복잡한 행동을 하는 악성 앱을 유연하게 감지할 수 있다면..

넘어야 할 산

- 더 정교하게
- 더 좋은 성능
- 특정 악성 앱에 특화되지 않는 일반화된 분석

정교한 분석

- 번역시 실행모델 개선 필요
  - 2단계 실행모델은 지나치게 단순
    - 화면전환 관계, 액티비티 lifecycle등을 무시
- 2단계 아닌 액티비티 단위로
  - 문법, 실행의미, 요약실행의미 수정
  - 인텐트 분석
- 간단한 수정으로 수행시간 18% 감소

성능 좋은 분석

- Sparse 분석 적용
  - Java 특성, 기계어 특성의 이슈들 해결

- 다양한 악성 행동 분석 위해 필요에 따라 Domain 확장

일반화된 악성 앱 분석

- 악성 앱을 유연하게 감지
  - 악성 앱의 행동을 질의로 받을 수 있게
  - 요약해석기와 질의를 확인하는 후처리기를 분리

- 악성 행동을 기술할 수 있는 언어 정의
- 기술된 악성행동을 확인하는 후처리기 설계

악성 앱 예제

- 1260개의 악성 앱 샘플
  - http://malgenomeproject.org

- 살펴보며
- 악성행동을 어떻게 기술/검출 할 수 있을지

개인정보 누출

- 60%가 전화번호 등 개인정보 유출

- Source API와 Sink API를 기술하고
- 그 사이에 흐름이 있음을 기술

SMS 가로채기 (1/3)

- zSone에 감염된 3D Cube Horror Terrible
- 특정 번호로부터 SMS가 도착하면 이를 가로챔

```
public class MJReceiver extends BroadcastReceiver {
   public void onReceive(…){
     …
     if(…){
       …
       str = sms.getDisplayOriginatingAddress();
       if(str == 10665123085){
         abortBroadcast();
       }
     }
   }
}
```

SMS 가로채기 (2/3)

- zSone에 감염된 3D Cube Horror Terrible
- 특정 번호로부터 SMS가 도착하면 이를 가로챔

```
<receiver android:name=".MJReceiver">
    <intent-filter>
        <action android:name="...SMS_RECEIVED"/>
    </intent-filter>
</receiver>
```

SMS 가로채기 (3/3)

- SMS_RECEIVED 이벤트 호출을 담당하는
- `onReceive()` 함수에서
- `abortBroadcast()` 가 호출되는가

SMS 몰래 보내기 (1/2)

- zSone에 감염된 3D Cube Horror Terrible
- 앱 시작과 동시에 특정 번호로 SMS 전송

```
// Kube.onCreate() -> Kube.initMainView() -> MJUtils.sendSms() ->
MJUtils.sendCM()

private void sendCM()
{
    SmsManager localSmsManager = SmsManager.getDefault();
    Context localContext = this.context;
    Intent localIntent = new Intent();
    PendingIntent localPendingIntent1 = ...;
    PendingIntent localPendingIntent2 = null;
    localSmsManager.sendTextMessage("10621900",
        null,"M6307AHD", localPendingIntent1, localPendingIntent2);
}
```

SMS 몰래 보내기 (2/2)

- 상수 문자열로 문자를 보내는
- sendTextMessage() 함수가
- 시작점으로부터 어떠한 유저 입력/이벤트 없이 호출되는가

루트권한 상승 (1/2)

- Exploit 이용해 루트 권한 상승
- JNI 사용해서 exploit을 직접 호출
- BaseBridge 악성코드

- exploit을 실행 가능하도록 저장소에 복사
- chmod로 실행퍼미션 추가
- 실행 `Runtime.getRuntime().exec("...").waitFor()`

루트권한 상승 (2/2)

- JNI를 분석하도록 ARM 기계어로 확장
  - 프로그램 실행 시점에서 이름을 얻어와
  - 해당 ARM 프로그램을 분석
  - exploit은 x86 멀웨어 분석과 유사하게 분석

- 또는 exec 함수 호출 자체를 악성행동으로
- 데이터를 파일을 꺼내 복사하고 권한을 준 뒤 exec으로 실행시키는 과정을 악성행동으로

결론

- 안드로이드 개인정보 누출 분석기 만들었고
- 일반화된 악성 앱 분석기로 확장