ScanDal/Privacy

안드로이드 앱의
개인정보 누출을 잡아내는
정적 분석기

서울대학교
프로그래밍 연구실
김진영 윤용호 이광근

요약

요약

- 안드로이드 앱의 개인정보 누출 문제

요약

- 안드로이드 앱의 개인정보 누출 문제
- 정적 분석으로 잡아보자

요약

- 안드로이드 앱의 개인정보 누출 문제
- 정적 분석으로 잡아
- 잡았다 요놈!

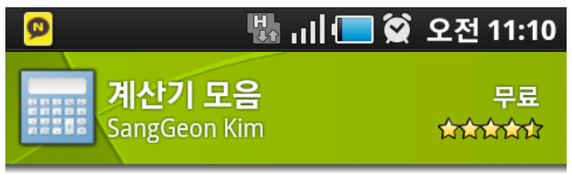
나의 안드로이드 사용기
- 블랙마켓

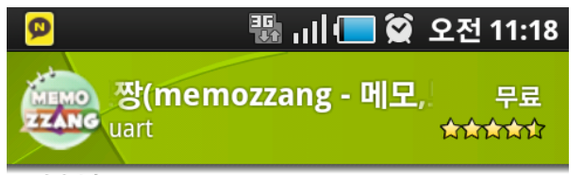
남의 안드로이드 사용기
- 블랙마켓

남의 안드로이드 사용기
- 블랙마켓

나의 안드로이드 사용기 - 확인. 수락. 다음. 예.

- 이런거, 안 읽으시죠?





안드로이드 보안은

- 사실상 무방비 상태
  - OS도
  - 마켓도
  - 사용자도

- 해답은?

TaintDroid?

- OS를 뜯어고쳐서 앱을 동적으로 감시

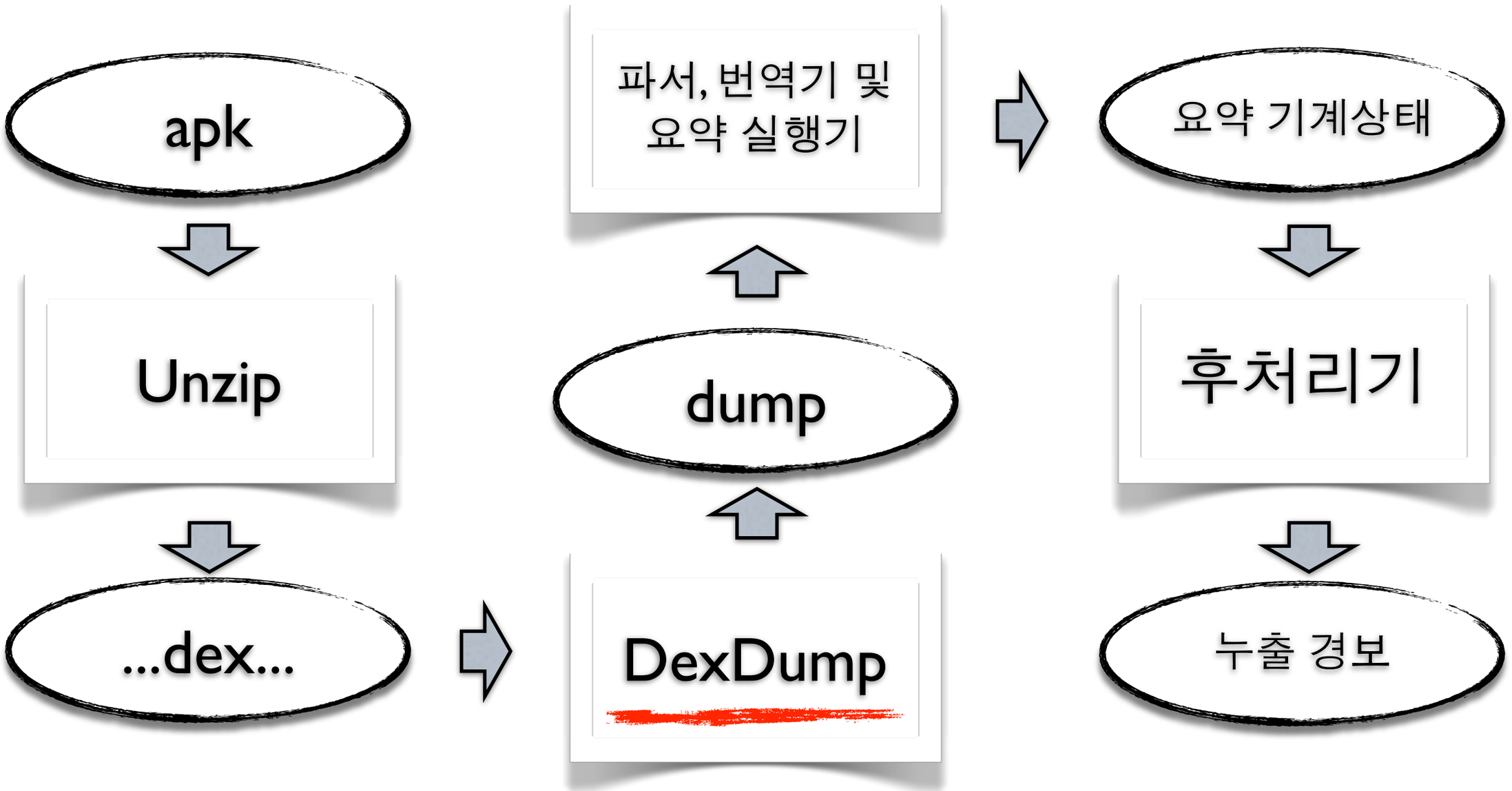
- 기기가 일을 더 하므로
  - (10%까지) 느려진다

- 기기 제조사도 구글도 그다지...

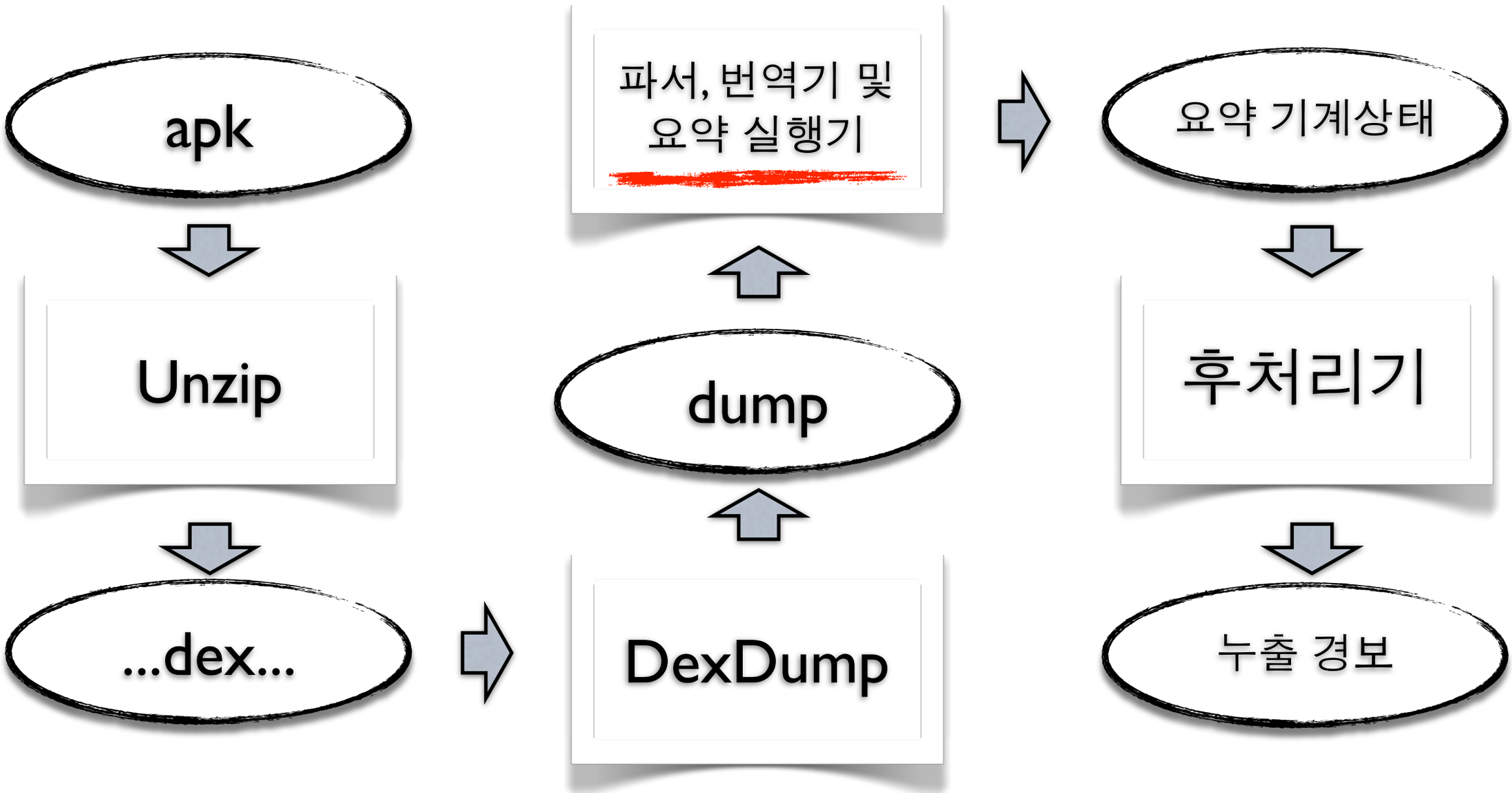
ScanDal / Privacy

- 앱을 미리 Scan하자
- Dalvik 레벨에서
  - Java가 아닌
- 지금은 Privacy 유출에 집중
- “Scandal을 일으켜보자”

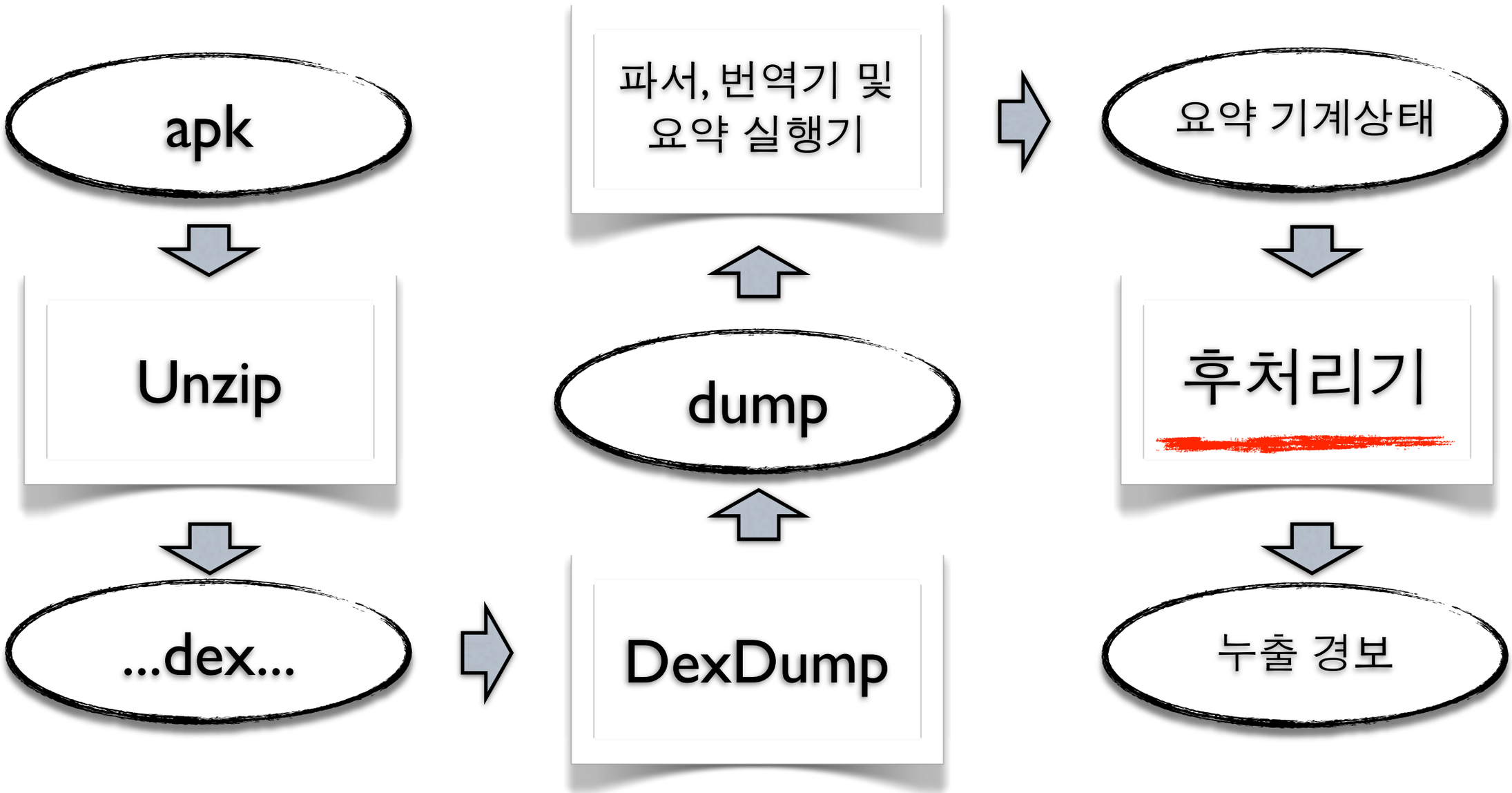
ScanDal 구조, 크게



ScanDal 구조, 크게



ScanDal 구조, 크게



우리가 찾는 것은

- 민감한 정보들이
  - 전화번호부, 문자메시지, 위치정보,
    사진, 영상, 기기고유번호(IMEI 등)

- 어디로 빠져나가는가
  - 인터넷
    - URL에 담아서(WebView.loadUrl)
    - 서버로 직접(OutputStream.write)
  - 문자메시지로

성능

앱 이름	코드 크기 (KB)	분석 시간 (초)	메모리 소모 (MB)	검출된 개인정보 누출 상황
Kids Preschool Puzzle	87	6	67	위치정보 → Flurry
Job Search	167	6	121	위치정보 → 서버
Kids Shapes	225	9	164	위치정보 → Flurry
Kids ABC Phonics	134	12	77	위치정보 → Flurry
Backgrounds HD Wallpapers	109	17	143	기기고유번호 → 서버
Bible Quotes	138	36	278	위치정보 → AdSense
ES Task Manager	158	86	433	위치정보 → AdSense
Multi Touch Paint	198	174	740	위치정보 → AdMob
Adao File Manager	255	220	1160	위치정보 → AdMob
(D-Day) The Day Before	293	626	2761	위치정보 → AdMob
프리즘 FreeSMS	387	708	1249	휴대폰번호 → 서버
Shot Gun Free*	95	36	164	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버
Baseball Superstars 2010*	165	61	285	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버
Monkey Jump 2*	169	74	442	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버
Gold Miner*	191	81	481	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버
Mini Army*	480	174	1292	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버
Xing Metro*	253	23049	1784	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버

공식 마켓의 인기 무료 앱 + 비공식 마켓의 변조된 앱*

AdSense, AdMob : 광고 모듈 서버

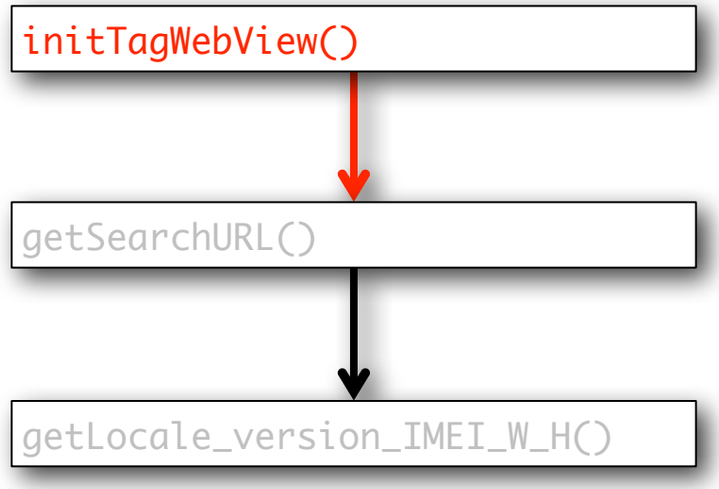
Flurry : 앱 사용 행태 분석 모듈 서버

Google Wallpaper 4.2.2 (1/5)

```
Wallpapers.onCreate(Bundle)
    ...
    callv TelephonyManager.getDeviceId()
    move-result r3
    puts r3 eWallpaperConst.IMEI
```

기기 고유번호

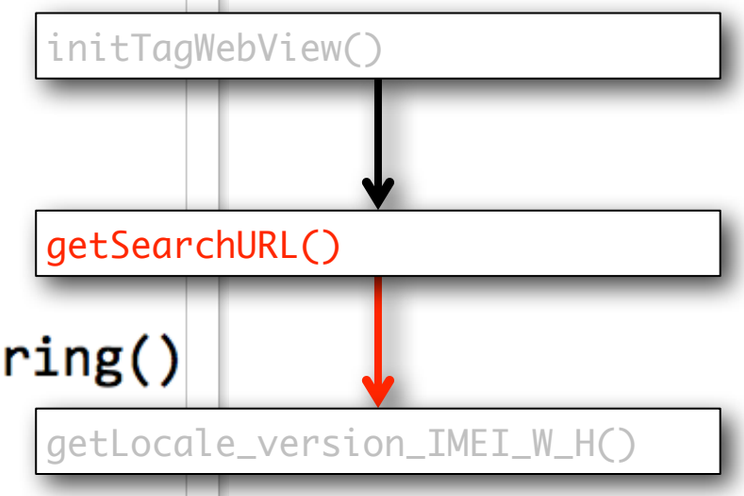
```
SearchTagsActivity.initTagWebView()
    ...
    get r0 r3 SearchTagsActivity.mWebView
    get r1 r3 SearchTagsActivity.mSharedPreferences
    calld XMLTools.getSearchURL(r1)
    move-result r1
    callv WebView.loadUrl(r1)
```



Google Wallpaper 4.2.2 (2/5)

```
Wallpapers.onCreate(Bundle)
    ...
```

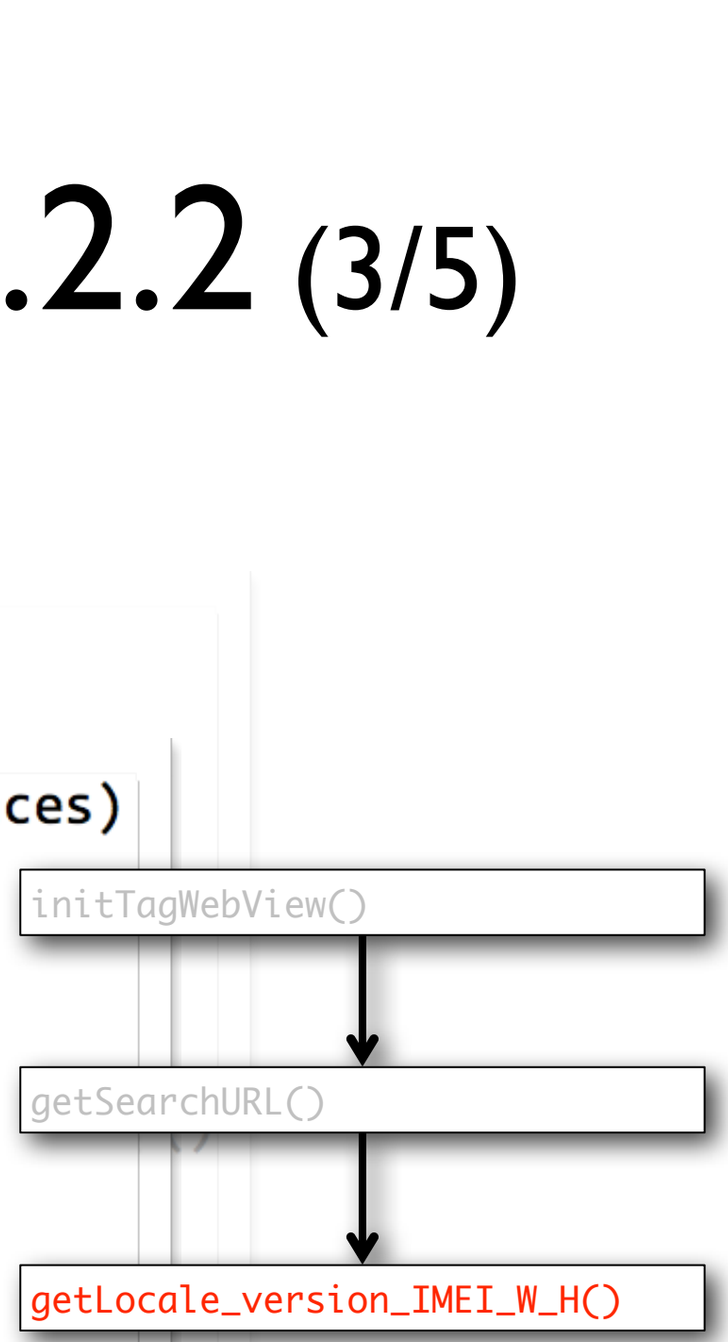
```
XMLTools.getSearchURL(SharedPreferences)
    ...
    calld XMLTools.getLocale_version_IMEI_W_H(r4)
    move-result r2
    callv StringBuilder.append(r1,r2)
    move-result r1
    calld EWallpaperHttpHelper.getSignatureParamString()
    move-result r2
    callv StringBuilder.append(r1,r2)
    move-result r1
    callv Stringbuilder.toString(r1)
    move-result r0
    return r0
```



Google Wallpaper 4.2.2 (3/5)

```
XMLTools.getLocale_version_IMEI_W_H(SharedPreferences)
    ...
    gets r5 eWallpaperConst.IMEI
    callv StringBuilder.append(r4,r5)
    move-result r4
    callv StringBuilder.toString(r4)
    move-result r4
    return r4
```

기기 고유번호



Google Wallpaper 4.2.2 (4/5)

```
Wallpapers.onCreate(Bundle)
    ...
```

```
XMLTools.getSearchURL(SharedPreferences)
    ...
    calld XMLTools.getLocale_version_IMEI_W_H(r4)
    move-result r2
    callv StringBuilder.append(r1,r2)
    move-result r1
    calld EWallpaperHttpHelper.getSignatureParamString()
    move-result r2
    callv StringBuilder.append(r1,r2)
    move-result r1
    callv Stringbuilder.toString(r1)
    move-result r0
    return r0
```

기기 고유번호

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

initTagWebView()

getSearchURL()

getLocale_version_IMEI_W_H()

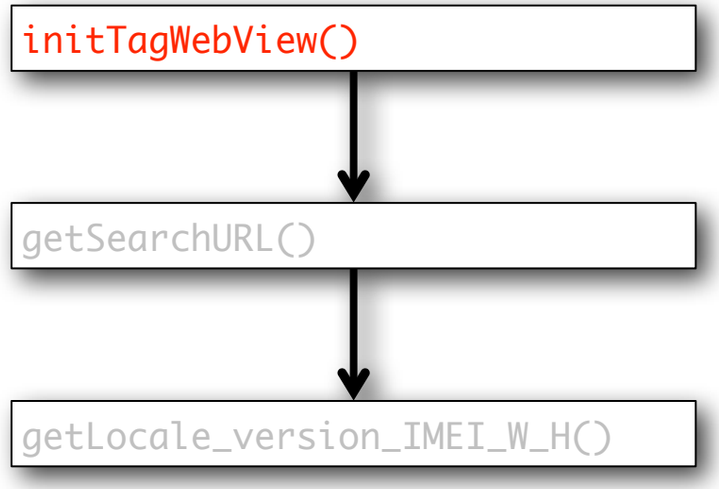
Google Wallpaper 4.2.2 (5/5)

```
Wallpapers.onCreate(Bundle)
    ...
    callv TelephonyManager.getDeviceId()
    move-result r3
    puts r3 eWallpaperConst.IMEI
```

기기 고유번호

```
SearchTagsActivity.initTagWebView()
    ...
    get r0 r3 SearchTagsActivity.mWebView
    get r1 r3 SearchTagsActivity.mSharedPreferences
    calld XMLTools.getSearchURL(r1)
    move-result r1
    callv WebView.loadUrl(r1)
```

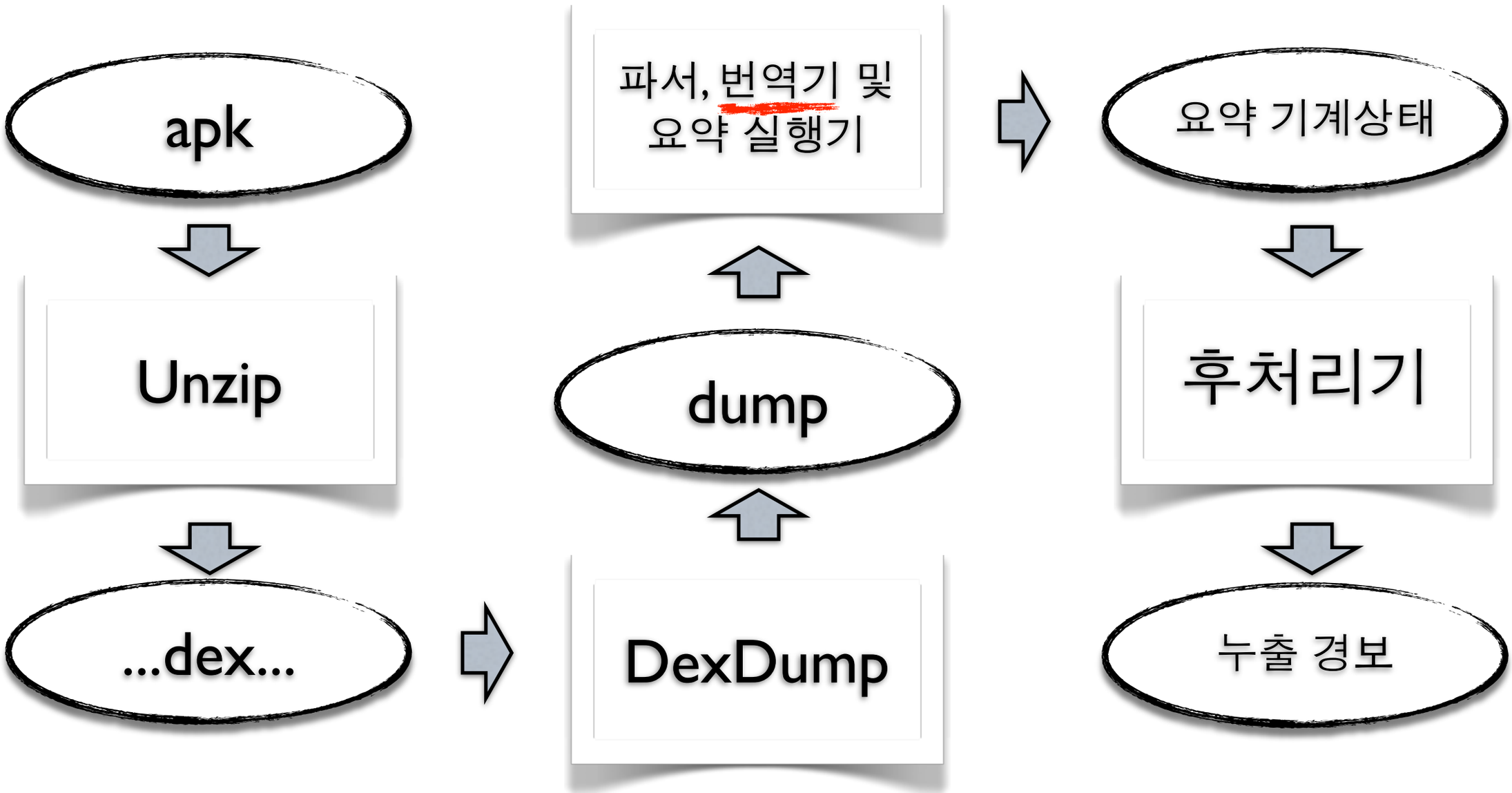
"http://www.imnet.us/api/wallpapers/photos/search_keywords?"+IMEI
+SignatureParamString



비공식 마켓의 변조 된 앱

- Monkey Jump 2, Gold Miner, Mini Army, Baseball Superstars 2010, Shot Gun Free, Xing Metro
  - 겉보기엔 공식 마켓 앱과 같으나
  - 악성 코드를 품고 다시 배포됨
- ScanDal로 분석한 결과
  - 공식 마켓의 앱에서는 누출 없음
  - 비공식 마켓의 앱에서는 누출 검출

다시, 좀 더 자세히



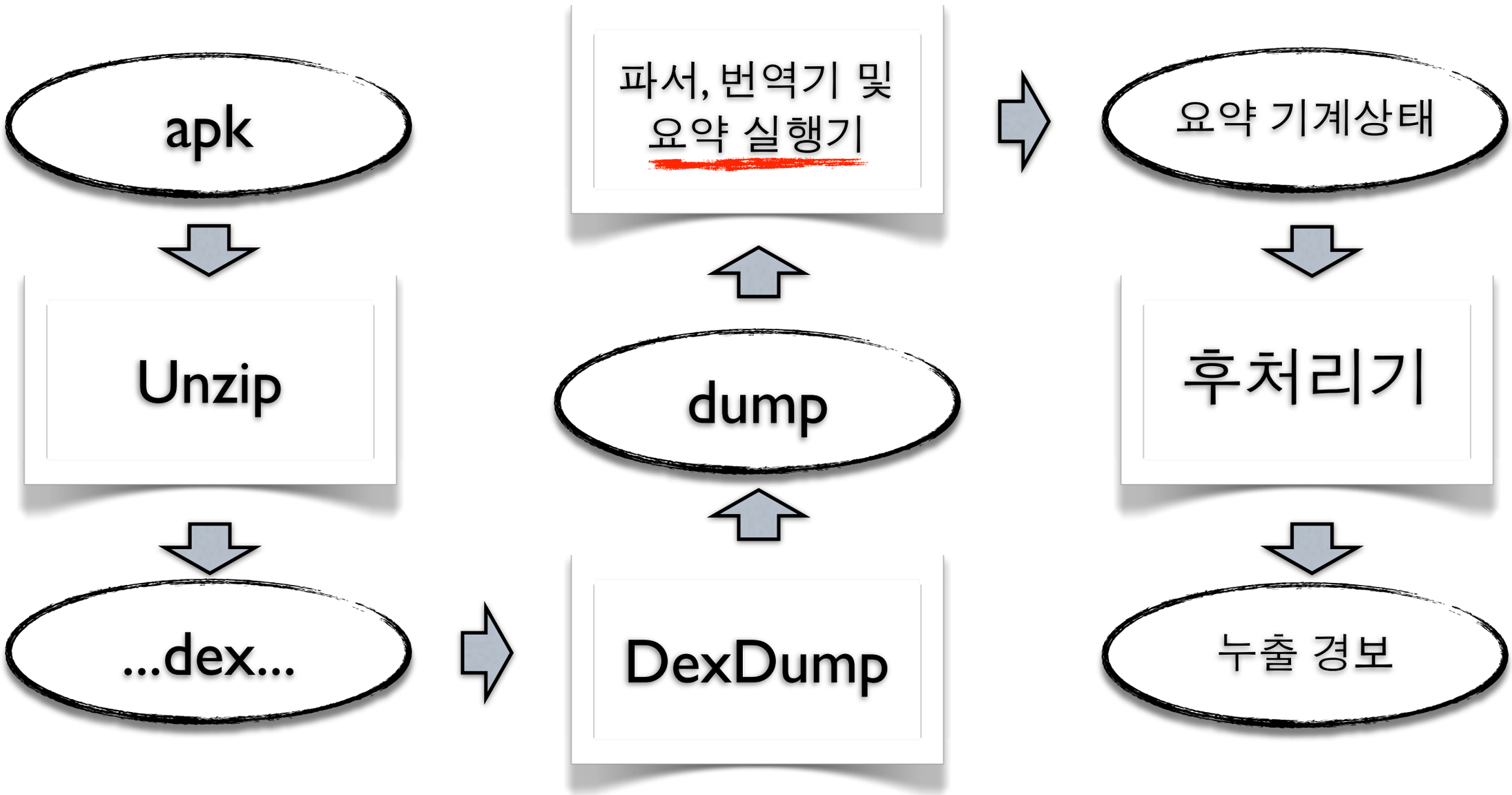
분석용 핵심 언어

- 200여 개의 달빅 명령어를
- 18개의 명령어로 번역
- API 실행의미 하드코딩용 명령어 포함

$$
\begin{array}{rcl}
D & ::= & \texttt{move}\ E\ E \\
& | & \texttt{istype}\ E\ E\ ty \\
& | & \texttt{new}\ E\ ty \\
& | & \texttt{get}\ E\ E\ id \\
& | & \texttt{put}\ E\ E\ id \\
& | & \texttt{gets}\ E\ id \\
& | & \texttt{puts}\ E\ id \\
& | & \texttt{geta}\ E\ E\ E \\
& | & \texttt{puta}\ E\ E\ E \\
& | & \texttt{addcallback}\ ty\ id\ E
\end{array}
$$

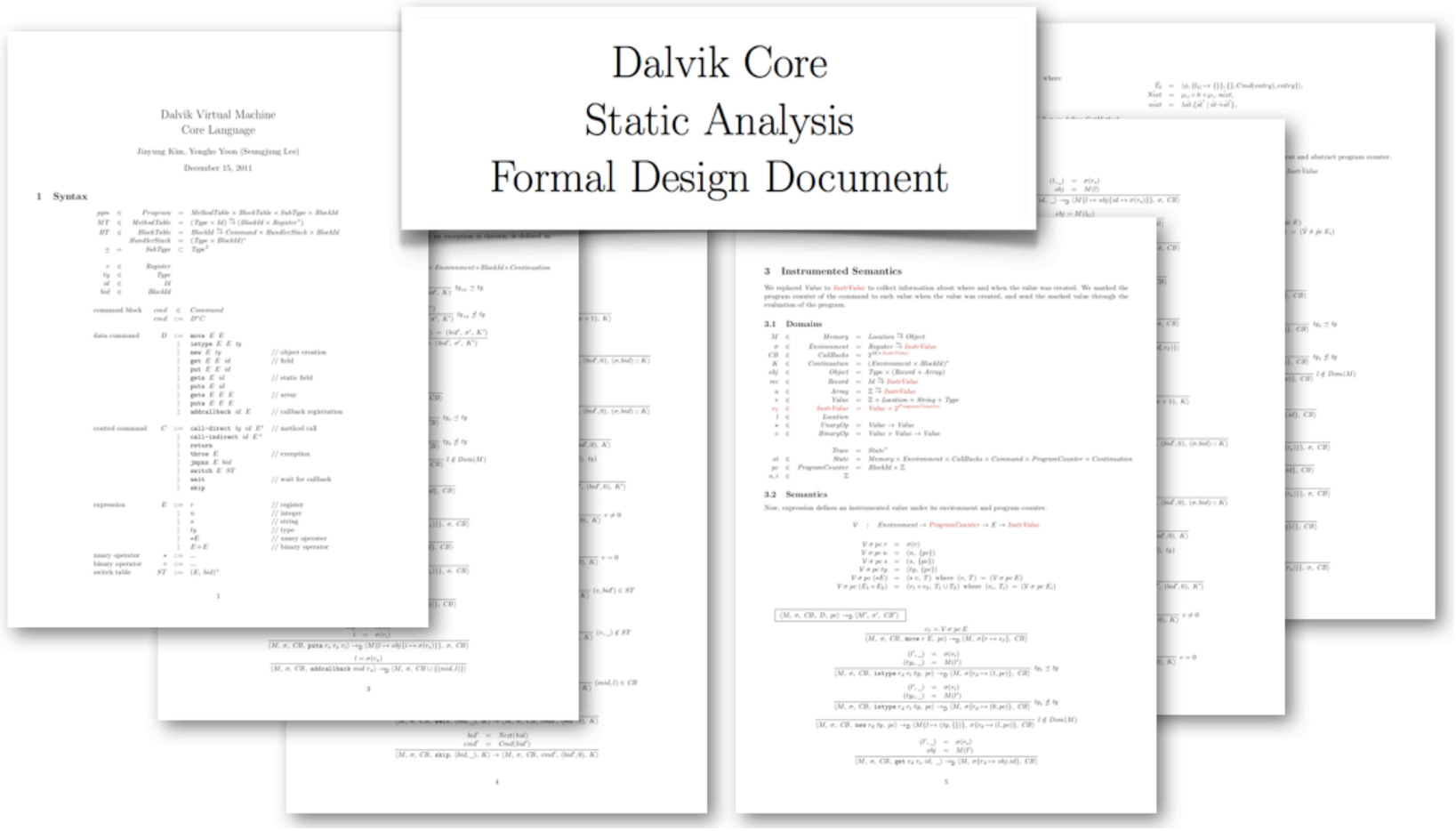
$$
\begin{array}{rcl}
C & ::= & \texttt{call-direct}\ ty\ id\ E^* \\
& | & \texttt{call-indirect}\ id\ E^+ \\
& | & \texttt{return} \\
& | & \texttt{throw}\ E \\
& | & \texttt{jmpnz}\ E\ bid \\
& | & \texttt{switch}\ E\ (E, bid)^+ \\
& | & \texttt{wait} \\
& | & \texttt{skip}
\end{array}
$$

분석기 본체, 요약 실행기



요약 실행기

- 요약해석 이론에 따라
  실행의미 정의, 요약실행 의미 정의



간단한 문자열 분석도

- Prefix Domain
  - 문자열을 공통 접두사로 요약
  - URL, URI 분석에 유용
    - 하길 기대하며

- ropas.snu.ac.kr/~yhyoon
- ropas.snu.ac.kr/~jykim
- →ropas.snu.ac.kr/~*

아까 그 예제에서는

```
Wallpapers.onCreate(Bundle)
    ...
    callv TelephonyManager.getDeviceId()
    move-result r3
    puts r3 eWallpaperConst.IMEI
```

기기 고유번호

```
SearchTagsActivity.initTagWebView()
    ...
    get r0 r3 SearchTagsActivity.mWebView
    get r1 r3 SearchTagsActivity.mSharedPreferences
    calld XMLTools.getSearchURL(r1)
    move-result r1
    callv WebView.loadUrl(r1)
```

initTagWebView()

getSearchURL()

getLocale_version_IMEI_W_H()

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

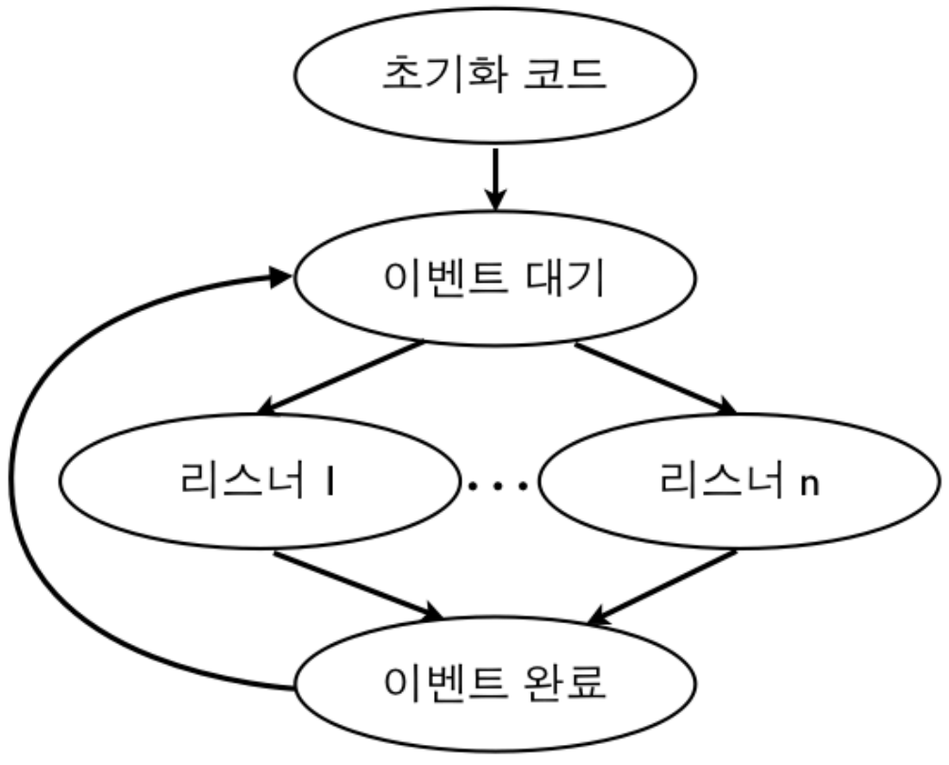
벽, 높지는 않은

- 타입에 따라 다른 메소드 호출
- 예외 상황 처리

- 실행의미 정의만 잘 되면 충분

진짜 벽 하나, Listener

- Java의 이벤트 처리 콜백
- 주로 UI에 관련



- 앱을 모델링
  - 초기화 단계 후
  - 이벤트를 대기하는 형태
  - 이벤트 대기 부분은 flow-insensitive 분석

좀 더 넓게 보면

- 코드에 명시되지 않는 함수 호출
  - Listener 등록을 포함하여
    - Thread, Intent, 기타 여러 API

- 지금은 일일이 실행의미를 하드코딩

노동집약적 벽

- 라이브러리 함수 실행의미
  - 정적분석기라면 누구나 마주치는

- Java 기본 라이브러리도 방대한데...
  - Android API는 클래스가 약 3천개

- 자주 쓰이는 것 위주로 하드코딩

언제나 까다로운

- Reflect
  - (아주 단순한) 다단계 프로그래밍
  - 문자열로 클래스, 메소드를
- 문자열 값도 분석하므로
  - 현재, 문자열로 클래스(java.lang.Class) 객체를 만드는 프로그램은 분석 가능

다른 측면 : 비용

앱 이름	코드 크기 (KB)	분석 시간 (초)	메모리 소모 (MB)	검출된 개인정보 누출 상황
Kids Preschool Puzzle	87	6	67	위치정보 → Flurry
Job Search	167	6	121	위치정보 → 서버
Kids Shapes	225	9	164	위치정보 → Flurry
Kids ABC Phonics	134	12	77	위치정보 → Flurry
Backgrounds HD Wallpapers	109	17	143	기기고유번호 → 서버
Bible Quotes	138	36	278	위치정보 → AdSense
ES Task Manager	158	86	433	위치정보 → AdSense
Multi Touch Paint	198	174	740	위치정보 → AdMob
Adao File Manager	255	220	1160	위치정보 → AdMob
(D-Day) The Day Before	293	626	2761	위치정보 → AdMob
프리즘 FreeSMS	387	708	1249	휴대폰번호 → 서버
Shot Gun Free*	95	36	164	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버
Baseball Superstars 2010*	165	61	285	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버
Monkey Jump 2*	169	74	442	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버
Gold Miner*	191	81	481	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버
Mini Army*	480	174	1292	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버
Xing Metro*	253	23049	1784	위치정보, 휴대폰번호, 기기고유번호 → 악성 서버

- 시간도 메모리도 많이 쓰지만...

앞으로

- 남은 벽들을 마저 넘고
  - Reflect
  - 누출 아닌 누출
    - 콜백 찾기 자동화
- 분석 속도를 높이고
- 발 넓히기

고맙습니다