

Using static analysis on Android applications to identify private information leaks

progress report
2nd RPE presentation by Kui Luo
computing and information science department

outline

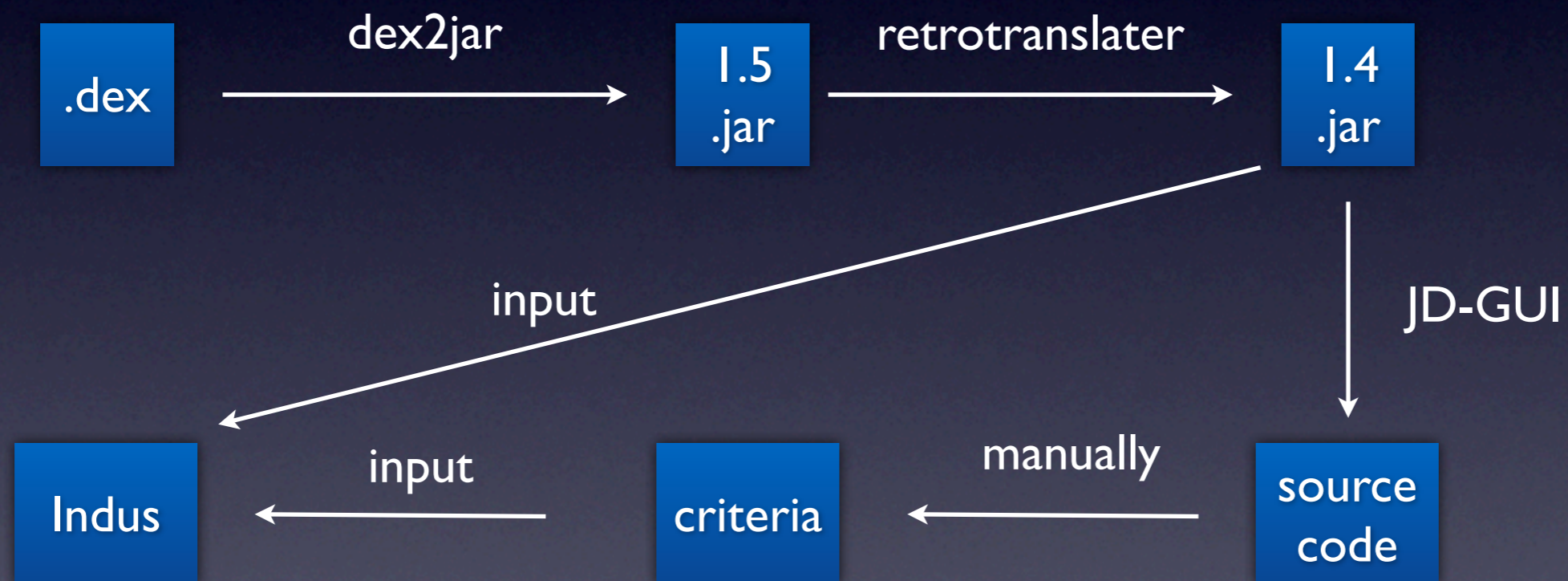
- summary on first phase and second phase
- step by step explanation
- root cause of the problem encountered
- future work discussion

recall on last presentation

- Goal: Identify Android apps that will send out sensitive information without user's acknowledgement.
- `.dex` → `.jar` : using `dex2jar`
- using Indus to slice the jar file and find if there is a flow from the sensitive information to the internet.

summary on second phase

- flow of the application tools



setup the environment

- first step: get indus compiled and running from command line.
 - issues under Mac OS: not support older version swt library, so the configure GUI cannot launch. Ant core class could not be found.
 - solution: compiled and configured under Windows 7, then running in Mac.

get on the track

- Using dex2jar to transform some apps from .dex to .jar file
- indus can only analyze classes which are JVM 1.4 compatible, so we tried to use asm to convert the bytecode into JVM 1.4 compatible format. But this turned out to be complicated.
- Found a tool called Retrotranslator, which does exactly what we wanted.

after downgrade

The screenshot shows a text comparison window titled "Bytecode compare: Tal" comparing two versions of the class `TalkingTomApplication.class`. The left pane shows the original code from `/Test/com/outfit7/talkingtom/TalkingTomApplication.class`, and the right pane shows the code after a downgrade from `/Transform/com/outfit7/talkingtom/TalkingTomApplication.class`. The comparison highlights several differences in the static initialization method `<clinit>()V`.

Original Code (Left)	Downgraded Code (Right)
<code>// access flags 0x2 private J m</code>	<code>// access flags 0x41008 static Ljava/lang/Class; class\$com\$outfit7\$tall</code>
<code>// access flags 0x8 static <clinit>()V</code>	<code>// access flags 0x8 static <clinit>()V</code>
<code>LDC Lcom/outfit7/talkingtom/TalkingTomApplication</code>	<code>GETSTATIC com/outfit7/talkingtom/TalkingTomAp</code>
<code>INVOKEVIRTUAL java/lang/Class.getName()Ljava/lang</code>	<code>DUP</code>
<code>PUTSTATIC com/outfit7/talkingtom/TalkingTomAppli</code>	<code>IFNONNULL L0</code>
<code>NEW com/outfit7/talkingtom/b</code>	<code>POP</code>
<code>DUP</code>	<code>ICONST_0</code>
<code>INVOKESPECIAL com/outfit7/talkingtom/b.<init>()V</code>	<code>ANEWARRAY com/outfit7/talkingtom/TalkingTomAp</code>
<code>PUTSTATIC com/outfit7/talkingtom/TalkingTomAppli</code>	<code>INVOKEVIRTUAL java/lang/Object.getClass()Ljav</code>
<code>LDC "AYDC4C2AVHKLZFX9NYK7"</code>	<code>INVOKEVIRTUAL java/lang/Class.getComponentTyp</code>
<code>PUTSTATIC com/outfit7/talkingtom/TalkingTomAppli</code>	<code>DUP</code>
<code>NEW java/lang/StringBuilder</code>	<code>PUTSTATIC com/outfit7/talkingtom/TalkingTomAp</code>
<code>DUP</code>	<code>L0</code>
<code>INVOKESPECIAL java/lang/StringBuilder.<init>()V</code>	<code>INVOKEVIRTUAL java/lang/Class.getName()Ljava/</code>
<code>LDC "http://outfit7-affirmations.appspot.com/res</code>	<code>PUTSTATIC com/outfit7/talkingtom/TalkingTomAp</code>
<code>INVOKEVIRTUAL java/lang/StringBuilder.append(Lja</code>	<code>NEW com/outfit7/talkingtom/b</code>
<code>ASTORE 0</code>	<code>DUP</code>
<code>INVOKESTATIC com/outfit7/talkingtom/b.e()Ljava/l</code>	<code>INVOKESPECIAL com/outfit7/talkingtom/b.<init></code>
<code>ASTORE 1</code>	<code>PUTSTATIC com/outfit7/talkingtom/TalkingTomAp</code>
<code>ALOAD 0</code>	<code>LDC "AYDC4C2AVHKLZFX9NYK7"</code>
<code>ALOAD 1</code>	<code>PUTSTATIC com/outfit7/talkingtom/TalkingTomAp</code>
<code>INVOKEVIRTUAL java/lang/StringBuilder.append(Lja</code>	<code>NEW java/lang/StringBuffer</code>
<code>LDC "/talking/"</code>	<code>DUP</code>

Find the source and sink

- Obtain the TaintDroid source code, and learn how to define the source and sink.
 - They actually modified the operating system classes, define their own classes for connivence.
 - tracking the classes where taint tag has been placed.

Find the source and sink

```
//-----  
// Audio data supply  
//-----  
/**  
 * Reads audio data from the audio hardware for recording into a buffer.  
 * @param audioData the array to which the recorded audio data is written.  
 * @param offsetInBytes index in audioData from which the data is written expressed in bytes.  
 * @param sizeInBytes the number of requested bytes.  
 * @return the number of bytes that were read or or {@link #ERROR_INVALID_OPERATION}  
 *         if the object wasn't properly initialized, or {@link #ERROR_BAD_VALUE} if  
 *         the parameters don't resolve to valid data and indexes.  
 *         The number of bytes will not exceed sizeInBytes.  
 */  
public int read(byte[] audioData, int offsetInBytes, int sizeInBytes) {  
    if (mState != STATE_INITIALIZED) {  
        return ERROR_INVALID_OPERATION;  
    }  
  
    if ( (audioData == null) || (offsetInBytes < 0 ) || (sizeInBytes < 0)  
        || (offsetInBytes + sizeInBytes > audioData.length)) {  
        return ERROR_BAD_VALUE;  
    }  
  
    // begin WITH_TAINT_TRACKING  
    //return native_read_in_byte_array(audioData, offsetInBytes, sizeInBytes);  
    int tag = Taint.TAINT_MIC;  
    int ret = native_read_in_byte_array(audioData, offsetInBytes, sizeInBytes);  
    Taint.addTaintByteArray(audioData, tag);  
    return ret;  
    // begin WITH_TAINT_TRACKING  
}
```

play around with indus

- learned how to define slice criteria and scope: no line number in the bytecode. Need Jimple level criteria.
- find out proper source and sink as slice criteria for each apps, and perform backward slice from sink or forward slice from the source

play around with indus

```
j = System.currentTimeMillis();
if (paramContext.checkSelfPermission("android.permission.ACCESS_COARSE_LOCATION") != 0)
    break label417;
if (!Log.isLoggable("AdMobSDK", 3))
    continue;
int i4 = Log.d("AdMobSDK", "Trying to get locations from the network.");
localLocationManager = (LocationManager)paramContext.getSystemService("location");
if (localLocationManager == null)
    break label408;
Criteria localCriteria1 = new Criteria();
localCriteria1.setAccuracy(2);
localCriteria1.setCostAllowed(0);
str = localLocationManager.getBestProvider(localCriteria1, 1);
i5 = 1;
if ((str != null) || (paramContext.checkSelfPermission("android.permission.ACCESS_FINE_LOCATION") != 0))
    continue;
if (!Log.isLoggable("AdMobSDK", 3))
    continue;
int i6 = Log.d("AdMobSDK", "Trying to get locations from GPS.");
localLocationManager = (LocationManager)paramContext.getSystemService("location");
if (localLocationManager == null)
    break label402;
Criteria localCriteria2 = new Criteria();
```

- `localLocationManager = (LocationManager)paramContext.getSystemService("location");`

play around with indus

specify the criteria

```
1 < slicer:criteria xmlns:slicer="http://indus.projects.cis.ksu.edu/slicer"  
2   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">  
3     < slicer:criterion slicer:considerExecution="true">  
4       <!-- This is an example of a stmt level criterion -->  
5       < slicer:className>com.admob.android.ads.AdManager</slicer:className>  
6       < slicer:method>  
7         < slicer:methodName>getCoordinates</slicer:methodName>  
8         < slicer:returnTypeName>android.location.Location</slicer:returnTypeName>  
9         < slicer:parameters>  
10        < slicer:parameterTypeName>android.content.Context</slicer:parameterTypeName>  
11        </slicer:parameters>  
12      </slicer:method>  
13      < slicer:stmt slicer:considerEntireStmt="false" slicer:index="13"/>  
14    </slicer:criterion>  
15  </slicer:criteria>  
16
```

problems encountered in debugging

- Type(12) exception in soot : the parent class for java/lang/Object is null.
- Type cast exception in soot when doing forward slicing : need to turn off the synchronization dependency.
- Null pointer exception in java.

problems encountered in debugging

- sometimes you could have this:

- Considering 33: aload_1[33]

```
Exception in thread "main" java.lang.RuntimeException:  
java.lang.RuntimeException: TypeStack merging failed; unequal stack lengths: 1  
and 0  
    at edu.ksu.cis.indus.tools.AbstractTool.run(AbstractTool.java:287)  
    at edu.ksu.cis.indus.tools.slicer.SliceXMLizerCLI.executeForSingleSlice  
(SliceXMLizerCLI.java:452)  
    at edu.ksu.cis.indus.tools.slicer.SliceXMLizerCLI.main(SliceXMLizerCLI.java:  
301)  
Caused by: java.lang.RuntimeException: TypeStack merging failed; unequal stack  
lengths: 1 and 0  
    at soot.coffi.TypeStack.merge(TypeStack.java:137)  
    at soot.coffi.CFG.jimplify(CFG.java:1392)  
    at soot.coffi.CFG.jimplify(CFG.java:1126)
```

.....

15 apps evaluation

- utilities : skype, androot, antivirus, historyeraser, t800khdv
- entertainment: cestos, solitaire, talkingtom
- social: bump, movies, wertago,dwell, mfwx
- weather: weather_channel, weatherbuglite, mjtqv

15 apps evaluation

- get 2 apps sliced by indus
- 8 apps has this type(12) error
- 5 apps has other exceptions

locate the root problem

- Thinking from the very beginning : which part of the transformation chain breaks?
- we are screwed by dex2jar : after transform to the jar file, the class file could not be verified by JVM 1.4
- It won't work if I do a .dex to .jar transformation using dex2jar, and then use dx.jar to restore to .dex.

locate the root problem

- the author of dex2jar released a new version several days ago, claimed that fixed many issues, include that the parent type of `java/lang/object` is null.
- but still remain incomplete.

possible ways

- try on open source android apps
 - hundreds of open source apps available
 - start from the source code
 - concerns: need to be JVM 1.4 compatible.
And these apps could be less “interesting”.

discussion

- How future work will go?