DeGuard: Statistical De-obfuscation for Android

Petar Tsankov, ETH Zurich

DeGuard Team

Benjamin Bichsel  Veselin Raychev  Petar Tsankov  Martin Vechev
Why De-obfuscate Android Applications?

Android binaries (APKs) (no code available)

Open-source (code available)

Google Play

F-Droid
Why De-obfuscate Android Applications?

Which APKs are malicious?

Which ones use vulnerable libraries?

Google Play

2.6M APKs

5K APKs

F-Droid
Layout Obfuscation in Android

Descriptive application-specific names

Non-descriptive names

Names of API classes/methods

package com.example.dbhelper

class DBHelper extends SQLiteHelper {
    SQLiteDatabase db;

    public DBHelper(Context ctx) {
        db = getWritableDatabase();
    }

    Cursor execSQL(String str) {
        return db.rawQuery(str);
    }
}

package a.b.c

class a extends SQLiteHelper {
    SQLiteDatabase b;

    public a(Context ctx) {
        b = getWritableDatabase();
    }

    Cursor c(String str) {
        return b.rawQuery(str);
    }
}
Layout Obfuscation in Android

Security Challenges

- Code inspection
- Third-party library detection
- ... many others

Descriptive application-specific names

Non-descriptive names

API names remain non-descriptive

Names of API
Layout Obfuscation in Android

Can we reverse layout obfuscation?

Descriptive application-specific names

Non-descriptive names

Names of API classes/methods

package com.example.dbhelper

class DBHelper extends SQLiteHelper {
  SQLiteDatabase db;

  public DBHelper(Context ctx) {
    db = getWritableDatabase();
  }

  Cursor execSQL(String str) {
    return db.rawQuery(str);
  }
}

package a.b.c

class a extends SQLiteHelper {
  SQLiteDatabase b;

  public a(Context ctx) {
    b = getWritableDatabase();
  }

  Cursor c(String str) {
    return b.rawQuery(str);
  }
}
Layout Obfuscation in Android

Descriptive application - Names of API classes/methods

Non-descriptive names

Yes, with roughly 80% accuracy!
Released in October 2016, so far: > 100GB distinct APKs de-obfuscated

Reddit posts/comments

[-] evantatarka WillowTree • 3 points 2 days ago
Nice! This should help with debugging issues in play services or other libs that are obfuscated just to make my life harder a bit easier.

[-] oleeEnchantado 2 points 4 days ago
Works quite well, I've tried on some small games.

[-] Tycon712 • 3 points 2 days ago
Can someone tell me what the point of using Proguard is if there are tools out there like this?

[-] theheartbreakpug • 6 points 2 days ago
As far as I know, this is brand new. I asked the creator of ProGuard a week ago how hard it is to unobfuscate code after it's run through proguard. He said it strips all the names out of the code so it's essentially impossible. I'm super impressed by what they've done here.

Tweets

D|ARS|IN @dharshin • Oct 17
Deobfuscate Proguard ed APKs: apk-deguard.com. The paper on the inner workings: srl.inf.ethz.ch/papers/deguard... #Android #MobileSecurity

Brian Carpenter @geeknik • 9h
Android Deobfuscation with Machine Learning reverses the effects of ProGuard [PDF] srl.inf.ethz.ch/papers/deguard...

rvivek @vivek_310 • Oct 17
*Android Deobfuscation with Machine Learning reverses the effects of ProGuard* #security #feedly

Android Deobfuscation with Machine... • /r/Reverse... 8 points and 1 comments so far on reddit
How Does DeGuard Work?
DeGuard: System Overview

Open-source, unobfuscated APKs

Obfuscated code

De-obfuscated code

class A extends SQLiteHelper {
    SQLiteDatabase b;
    public A(Context ctx) {
        b = getWritableDatabase();
    }
}

class DBHelper extends SQLiteHelper{
    SQLiteDatabase db;
    public DBHelper(Context ctx) {
        db = getWritableDatabase();
    }
}
Probabilistic Graphical Models
Probabilistic Graphical Models

Class `a` extends `SQLiteHelper` {
    SQLiteDatabase `b`;
    public `a`(Context `ctx`) {
        `b` = getWritableDB();
    }
}

Graph + features define a **probabilistic graphical model**

$$P(a, b | SQLiteHelper, getWritableDB) = \frac{1}{Z} \exp(0.3 \cdot f_1(SQLiteHelper, a) + 0.2 \cdot f_2(SQLiteHelper, a) + \cdots)$$

For details see report on www.apk-deguard.com
Probabilistic Graphical Models

class a extends SQLiteHelper {
    SQLiteDatabase b;
    public a(Context ctx) {
        b = getWritableDB();
    }
}

How are the features and their weights learned?

$P(\{a, b\} | SQLiteHelper, getWritableDB) = \frac{1}{Z} \exp(0.3 \cdot f_1(SQLiteHelper, a) + 0.2 \cdot f_2(SQLiteHelper, a) + \cdots)$

Known variables
- SQLiteHelper, getWritableDB
- a, b

Unknown variables
- $f_1, f_2, \ldots, f_7$

Feature functions
Learning
Learning Unobfuscated APKs

> 2,000

Static analysis

Unobfuscated APKs

Feature templates

> 2,000

> 100,000

Dependency graphs

<table>
<thead>
<tr>
<th>name1</th>
<th>name2</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQLiteHelper</td>
<td>DBUtils</td>
<td>0.3</td>
</tr>
<tr>
<td>SQLiteHelper</td>
<td>DBHelper</td>
<td>0.2</td>
</tr>
<tr>
<td>getWritableDB</td>
<td>db</td>
<td>0.7</td>
</tr>
<tr>
<td>getWritableDB</td>
<td>instance</td>
<td>0.4</td>
</tr>
<tr>
<td>DBGUtils</td>
<td>instance</td>
<td>0.5</td>
</tr>
<tr>
<td>DBHelper</td>
<td>db</td>
<td>0.4</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Features (with candidate names)

Actual graphs have > 1,000 nodes

Compute weights that maximize $P(\hat{O} = \hat{o}_i | K = \hat{k}_i)$ for all training samples $(\hat{o}_i, \hat{k}_i)$
DeGuard: System Overview

Learning phase

Static analysis

Training

Probabilistic model $P(\square|\square)$

Prediction phase

Static analysis

MAP inference

Transform

class A extends SQLiteHelper {
    SQLiteDatabase b;
    public A(Context ctx) {
        b = getWritableDatabase();
    }
}

Obfuscated code

Open-source, unobfuscated APKs

class DBHelper extends SQLiteHelper {
    SQLiteDatabase db;
    public DBHelper(Context ctx) {
        db = getWritableDatabase();
    }
}

De-obfuscated code
class \texttt{a} extends \texttt{SQLiteHelper} {
    SQLiteDatabase \texttt{b};
    public \texttt{a}(Context \texttt{ctx}) {
        \texttt{b} = \texttt{getWritableDB}();
    }
}

\textbf{Obfuscated Code}

\textbf{Prediction Phase}

\begin{array}{c|c|c}
\text{name1} & \text{name2} & \text{weight} \\
\hline
\text{SQLiteHelper} & \text{DBUtils} & 0.3 \\
\hline
\text{SQLiteHelper} & \text{DBHelper} & 0.2 \\
\hline
\text{name1} & \text{name2} & \text{weight} \\
\hline
\text{getWritableDB} & \text{db} & 0.7 \\
\hline
\text{getWritableDB} & \text{instance} & 0.4 \\
\hline
\text{DBUtils} & \text{instance} & 0.5 \\
\hline
\text{DBHelper} & \text{db} & 0.4 \\
\hline
\text{DBUtils} & \text{db} & 0.2 \\
\hline
\text{DBHelper} & \text{instance} & 0.2 \\
\end{array}
class a extends SQLiteHelper
SQLiteDatabase b;
public a(Context ctx) {
    b = getWritableDB();
}

Obfuscated Code

\[
\hat{\phi} = \arg \max_{\phi' \in \Omega} P(\hat{\phi} = \phi' | \hat{K} = \hat{k})
\]

Candidate assignment $\hat{\phi}$

| a = DBUtils | b = instance | $P(\hat{\phi} | \hat{k})$ |
|-------------|--------------|--------------------------|
| a = DBHelper | b = db       | 1.3                      |
| a = DBUtils  | b = db       | 0.8                      |
| a = DBHelper | b = instance | 1.2                      |

*Non-normalized*
class `a` extends `SQLiteHelper`

```java
SQLiteDatabase `b`;
public `a`(Context `ctx`) {
  `b` = `getWritableDB`();
}
```

### Prediction Phase

#### MAP Inference

\[
\tilde{\omega} = \arg\max_{\omega'} P(\tilde{\omega} = \omega' | \tilde{K} = \tilde{k})
\]

\(\omega' \in \Omega\)

**Candidate** assignment \(\tilde{\omega}\)  
**\(P(\tilde{\omega} | \tilde{k})\)**

- `a` = DBUtils  
  `b` = instance  
  1.2
- `a` = DBHelper  
  `b` = db  
  1.3
- `a` = DBUtils  
  `b` = db  
  0.8
- `a` = DBHelper  
  `b` = instance  
  1.2

*Non-normalized*
class a extends SQLiteHelper {
    SQLiteDatabase b;
    public a(Context ctx) {
        b = getWritableDB();
    }
}

Obfuscated Code

class DBHelper extends SQLiteHelper {
    SQLiteDatabase db;
    public DBHelper(Context ctx) {
        db = getWritableDB();
    }
}

Deobfuscated Code
Semantics-Preserving De-obfuscation Constraints

Freely renaming fields/variables/methods may **change** the application’s **semantics**

<//> Syntactic constraints

*e.g. “Fields within a class must have distinct names”*

Semantic constraints

*e.g. “Method overloads must be preserved”*
DeGuard: System Overview

Learning phase

Static analysis

Training

Probabilistic model $P(\square|\square)$

Prediction phase

Static analysis

MAP inference

Transform

Obfuscated code

De-obfuscated code

Open-source, unobfuscated APKs

DeGuard class extends SQLiteHelper{
    SQLiteDatabase b;
    public DeGuard(Context ctx) {
        b = getWritableDatabase();
    }
}

class DBHelper extends SQLiteHelper{
    SQLiteDatabase db;
    public DBHelper(Context ctx) {
        db = getWritableDatabase();
    }
}
DeGuard Implementation

Static Analysis
- Static analysis framework for Java and Android

Learning and MAP Inference
- Scalable open-source framework for structured prediction
- Open-source: http://nice2predict.org
- Training data: 2K open-source, unobfuscated Android applications

www.apk-deguard.com
Evaluation

1. Can DeGuard reverse ProGuard?
2. Can DeGuard detect third-party libraries?
3. Is DeGuard useful for malware inspection?
ProGuard Experiment

<\/> Source Code → Compile → Non-obfuscated APK → ProGuard → Obfuscated APK → DEGUARD → De-obfuscated APK
After Obfuscation

% of program elements

- Fields: 20
- Methods: 4
- Classes: 0
- Packages: 60
- Total: 80

Only 13% known names
Can DeGuard Reverse ProGuard?

80.6% correct names

1.6% known names

Package names can be directly used to predict third-party libraries

<table>
<thead>
<tr>
<th></th>
<th>Known names</th>
<th>Correctly predicted names</th>
<th>Mis-predicted names</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fields</td>
<td>80%</td>
<td>80.6%</td>
<td>13%</td>
</tr>
<tr>
<td>Methods</td>
<td>80%</td>
<td>80.6%</td>
<td>13%</td>
</tr>
<tr>
<td>Classes</td>
<td>80%</td>
<td>80.6%</td>
<td>13%</td>
</tr>
<tr>
<td>Packages</td>
<td>80%</td>
<td>80.6%</td>
<td>13%</td>
</tr>
<tr>
<td>Total</td>
<td>80%</td>
<td>80.6%</td>
<td>13%</td>
</tr>
</tbody>
</table>

80% of the names are identical to the original ones
Can DeGuard Detect Third-Party Libraries?

ProGuard obfuscates library package names

Source Code

Library Code

Obfuscated APK

De-obfuscated APK

Precision: 93.1%
Recall: 91%
Is DeGuard Useful for Malware Inspection?

We de-obfuscated all samples from the Android Malware Genome Project

```java
class d {
    String a = System.getProperty(..)
    char[] b;
    byte [] c;
    byte[] a(String) {..}
}
```

```java
class Base64 {
    String NL = System.getProperty(..)
    char[] ENC;
    byte [] DEC;
    byte[] decode(String) {..}
}
```

Malware Sample

De-obfuscated Malware Sample

- Reveals string decoders
- Reveals classes that handle sensitive data (e.g. Location)
- Hard to handle heavily-obfuscated code (e.g. reflection)
Summary

Try online: [www.apk-deguard.com](http://www.apk-deguard.com)