The Inner Workings of Mobile Cross-Platform Technologies

BLACK HAT ASIA
Singapore, March 2014
ME?

Simon Roses Femerling

• Founder & CEO, VULNEX  [www.vulnex.com](http://www.vulnex.com)

• Blog:  [www.simonroses.com](http://www.simonroses.com)

•  @simonroses  |  @vulnexsl

• Former Microsoft, PwC, @Stake

• DARPA Cyber Fast Track award on software security project

• Black Hat, RSA, OWASP, SOURCE, AppSec, DeepSec, TECHNET
TALK OBJECTIVES

• Existing mobile cross-platform tech
• Better or worst security?
• How and what to audit
AGENDA

1. Too Many Platforms
2. Cross-Platform Technologies
3. Auditing Apps
4. Conclusions
1. Too Many Platforms
## 1. Mobile Platform Madness

<table>
<thead>
<tr>
<th>Leaders</th>
<th>Contenders</th>
</tr>
</thead>
<tbody>
<tr>
<td>iOS</td>
<td>BlackBerry</td>
</tr>
<tr>
<td>Android</td>
<td>Tizen</td>
</tr>
<tr>
<td>Windows</td>
<td>Sailfish</td>
</tr>
</tbody>
</table>

---

*Image: Mobile platform logos and icons.*
### 1. TRADITIONAL VS. CROSS-PLATFORM DEVELOPMENT

<table>
<thead>
<tr>
<th>Cross-Platform</th>
<th>Dev Language</th>
<th>ANDROID</th>
<th>iPhone</th>
<th>Windows Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ximian (Mono)</td>
<td>.NET</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Corona SDK</td>
<td>LUA</td>
<td>YES</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>PhoneGap</td>
<td>HTML / CSS / JavaScript</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>RhoMobile</td>
<td>JavaScript / HTML / CSS / Ruby</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
</tbody>
</table>
1. EXPANDING TOOLKIT

**TRADITIONAL**
- apktool
- Dex2jar
- JD-GUI
- IDA PRO
- Debugger

**NEW**
- .NET decompiler / disassemblers
- Ruby decompiler / disassemblers
- JavaScript static analysis
- Custom tools (parse smali and extract info)
2. Cross-Platform Technologies
2. WE WILL EXPLORE

- Basic4android: http://www.basic4ppc.com/
- PhoneGap: http://phonegap.com/
- Corona SDK: http://coronalabs.com/
- MonoDroid: http://xamarin.com/android
- MonoTouch: http://xamarin.com/ios
2. BASIC4ANDROID

- Writes Android & Desktop apps using BASIC
- Code gets translated from BASIC to Java, so no dependencies / native code
- Includes 33 Java libraries
2. BASIC4ANDROID: EXAMPLE

Sub Activity_Create(FirstTime As Boolean)
    MsgBox("Hello Folks!", ")
End Sub

public static String _activity_create(boolean paramBoolean)
    throws Exception
{
    Common.Msgbox("Hello Folks!", ", mostCurrent.activityBA);
    return ");
}
2. BASIC4ANDROID: PERMISSIONS DEFAULT

- By default 4 permissions:
  - android.permissionINTERNET
  - android.permission.BLUETOOTH
  - android.permission.WRITE_EXTERNAL_STORAGE
  - android.permission.BLUETOOTH_ADMIN
2. BASIC4ANDROID: KUDOS, OBFUSCATION

- Strings obfuscation
- Variables renaming
2. PHONEGAP

- Writes Apps using HTML / CSS & JavaScript
- Platforms: iOS, Android, Windows, Blackberry, bada, webOS
- Many Apps!
2. PHONEGAP APP STRUCTURE

- PLATFORM BINARY
  - config.xml
  - www/ Folder
    - index.html
    - js/ Folder
    - img/ Folder
    - css/ Folder
    - plugins/ Folder
    - Misc. Files & Folders
2. PHONEGAP: ASK FOR PERMISSIONS & YOU SHALL RECEIVE

- android.permission.VIBRATE
- android.permission.ACCESS_COARSE_LOCATION
- android.permission.ACCESS_FINE_LOCATION
- android.permission.ACCESS_LOCATION_EXTRA_COMMANDS
- android.permission.READ_PHONE_STATE
- android.permission.INTERNET
- android.permission.RECEIVE_SMS
- android.permission.RECORD_AUDIO
- android.permission.MODIFY_AUDIO_SETTINGS
- android.permission.READ_CONTACTS
- android.permission.WRITE_CONTACTS
- android.permission.WRITE_EXTERNAL_STORAGE
- android.permission.ACCESS_NETWORK_STATE
- android.permission.GET_ACCOUNTS
- android.permission.BROADCAST_STICKY
2. CORONA SDK

- Writes Apps using LUA
- Platforms: iOS, Android, Kindle Fire & NOOK
- Mostly games!
2. CORONA SDK APP STRUCTURE

- PLAFORM BINARY
  - resource.car

- Platform Files & Folders

- Misc. Files & Folders

- Lib/ - Android
  - libcorona.so
  - liblua.so
  - Misc. libraries
2. CORONA SDK DEFAULT PERMISSIONS

• It’s a start!
  • android.permission INTERNET
2. RHOMOBILE

• Writes Apps using Ruby & HTML / JS / CSS

• Platforms: iOS, Android, Windows Phone and Windows Desktop

• Limited set of Apps but improving
2. RHOMOBILE APP STRUCTURE

iOS bin: rhorunner

Android bin: rhodes

lib/ Folder:
- librhodes.so

Misc. Files & Folders

Platform Files & Folders

apps/ Folder
- app/ Folder
  - app_manifest.txt
  - public/ Folder
    - rhoconfig.txt

db/ Folder
- syncdb_java.triggers
- syncdb.schema
- syncdb.triggers

Misc. Files & Folders
2. RHOMOBILE SECURITY

- Developers must declare permissions (11 perms available)
- Security Token: restricts access to App
- JavaScript & CSS Obfuscation
2. MONODROID

- Writes Apps using C# and .NET (Android)

- Platforms: iOS, Android, Windows Phone & MacOS

- Becoming popular
using System;
using Android.App;
using Android.Content;
using Android.Runtime;
using Android.Views;
using Android.Widget;
using Android.OS;

namespace MyTest
{
    [Activity (Label = "MyTest", MainLauncher = true)]
    public class MainActivity : Activity
    {
        int count = 1;

        protected override void OnCreate (Bundle bundle)
        {
            base.OnCreate (bundle);

            // Set our view from the "main" layout resource
            SetContentView (Resource.Layout.Main);

            // Get our button from the layout resource,
            // and attach an event to it
            Button button = FindViewById<Button> (Resource.Id.myButton);

            button.Click += delegate {
                button.Text = string.Format ("{0} clicks!", count++);
            };
        }
    }
}
2. MONODROID APP STRUCTURE

**assemblies/ Folder**
- Mono.Android.dll
- Mono.Security.dll
- System.dll
- App DLLs
- Misc. DLLs

**lib/ Folder**
- armeabi/
- armeabi-v7a/
- X86/

Platform Files & Folders

PLAFTORM
BINARY
2. MONOTOUCH

• Writes Apps using C# and .NET (iOS)

• Platforms: iOS, Android, Windows Phone & MacOS

• Same as MonoDroid
2. MONOTOUCH APP STRUCTURE

- Platform
- Files & Folders
- <APP NAME>.EXE
- DLLs
3. Auditing Apps
3. FINGERPRINT BASIC4ANDROID

- Apktool or unzip apk
  - Search Folder: “anywheresoftware”

- All b4a Apps contain this folder
3. BASIC4ANDROID REVERSING

• If App was published in debug mode, we can recover BASIC code!

```java
.method public static _activity_create(Z)Ljava/lang/String;
   .line 226
   const/16 v0, 0x18
   sput v0, Lanywheresoftware/b4a/BA->debugLineNum:I
   const-string v0, "Sub Activity_Create(FirstTime As Boolean)"
   sput-object v0, Lanywheresoftware/b4a/BA->debugLine:Ljava/lang/String;
   .line 227
   const/high16 v0, 0x80
   invoke-static {v0}, Lanywheresoftware/b4a/debug/Debug;->ShouldStop(I)V
   .line 228
   const/16 v0, 0x19
   sput v0, Lanywheresoftware/b4a/BA->debugLineNum:I
   const-string v0, "If fbLogin.AccessToken = \"\" Then"
   sput-object v0, Lanywheresoftware/b4a/BA->debugLine:Ljava/lang/String;
   .line 229
   const/high16 v0, 0x100
```
3. BASIC4ANDROID BAL FILES

- BAL files contain UI elements
- Open then in b4a designer
3. FINGERPRINT PHONEGAP

• Look for www/ folder

• All app code is HTML & JavaScript 😊
3. PHONEGAP REVIEW

- What permissions?

- Config.xml
  - What plugins are being used?
  - `<access origin="*" />` ?

- JavaScript code
  - Sensitive information?
  - Use of Eval()
  - Cross Site Scripting is back: WebView, Plugins, etc.
  - Use of clear text channels?

- PhoneGap Security Wiki:
3. FINGERPRINT CORONA SDK

- File: resource.car

- Lib/ Folder:
  - liblua.so
  - Libcorona.so
3. FINGERPRINT RHOMOBILE

• iOS
  – File: rhorunner
  – Apps/ folder:
    • rhoconfig.txt file
    • Folders: app, lib and public
  – Lib/ folder:
    • Files *.iseq

• Android
  – Lib/ Folder:
    • Librhodes.so
  – Apps/ folder:
    • rhoconfig.txt file
    • Folders: app and public
3. RHOMOBILE RHOCONFIG.TXT

```plaintext
# startup page for your application
start_path = '/app'

# path to the options page (in this case handled by javascript)
options_path = '/app/Settings'

# location of bundle url (i.e. from rhohub.com); used by desktop win32 simulator
rhubundle_zip_url = ''

# optional password to access bundle (usually not required); used by desktop win32 simulator
rhubundle_zip_pwd = nil

# Rhodes log properties
# log level
# 0-trace, 1-info(app level), 2-warnings, 3-errors
# for production set to 3
MinSeverity = 1

# enable copy log messages to standard output, useful for debugging
LogToOutput = 1

# '*' means all categories, otherwise list them : Cat1, Cat2
LogCategories = '*'

# what categories to exclude
ExcludeLogCategories =

# max log file size in Bytes, set 0 to unlimited size; when limit is reached, log wraps to beginning of file
MaxLogFileSize=50000
```

- App start page
- Any passwords?
- Is HTTP Server for debugging enabled?
- Where are logs going?
- Any URLs?
3. FINGERPRINT MONODROID & MONOTOUCH

- **iOS**
  - `<App Name>.exe`
  - Mono DLLs
  - Xamarin DLLs
  - App DLLs

- **Android**
  - `lib/` folder
    - `(armeabi, armeabe-v7a, x86)` folders
      - `libmonodroid.so`
  - `assemblies` folder
    - Mono DLLs
    - Xamarin DLLs
    - App DLLs
3. NOTHING LIKE THE WTF LOG

- Save to disk error msg in JSON format
- Sends error msg to server using HTTP
3. NO OBFUSCATION!!
3. **USUAL SUSPECTS!**

- Clear Text Communication (OWASP M3)
- Weak Crypto (OWASP M6)
- Use of insecure 3 party libs: HELLO VULNA!
- Sensitive info to SD (OWASP M2)
- App Logic exposed
- Insecure passwords (OWASP M2)
- JavaScript Injection (OWASP M7)
- Sensitive info in config files (OWASP M2)
3. WHERE TO LOOK FOR BUGS

• Native code
  – app
  – libraries

• Cross-Platform App
  – app
  – libraries
  – config files
4. Conclusions
3. SOME APP CASE STUDIES MISSING?
4. CROSS-PLATFORM MOBILE SECURITY RECAP

• Depending on the tech a bit more hard to reverse

• Suffers the same bugs as native apps

• Not offering much additional security
4. Q&A

• Thanks!

• @simonroses | @vulnexssl

• www.vulnex.com