Mobile Apps and How To Pentest Them.

OWASP
The Open Web Application Security Project

About Me

- $whoami

Ing. Gustavo M. Sorondo (Puky)

CTO @ Cinta Infinita|Information Security

http://www.cintainfinita.com

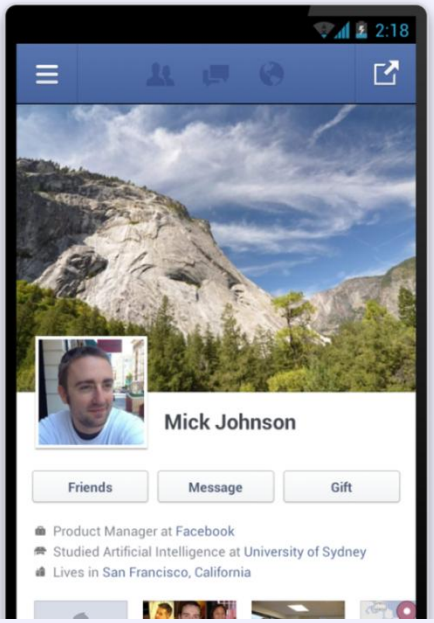
Cinta Infinita
Information Security

OWASP
The Open Web Application Security Project

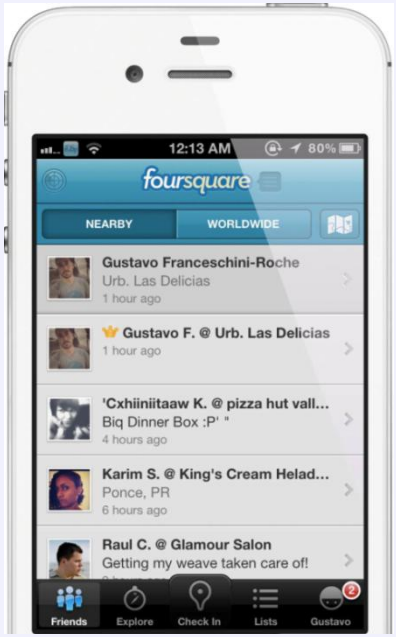
Tipos de Aplicaciones Mobile

(Según su programación)

Web Based App



Native App



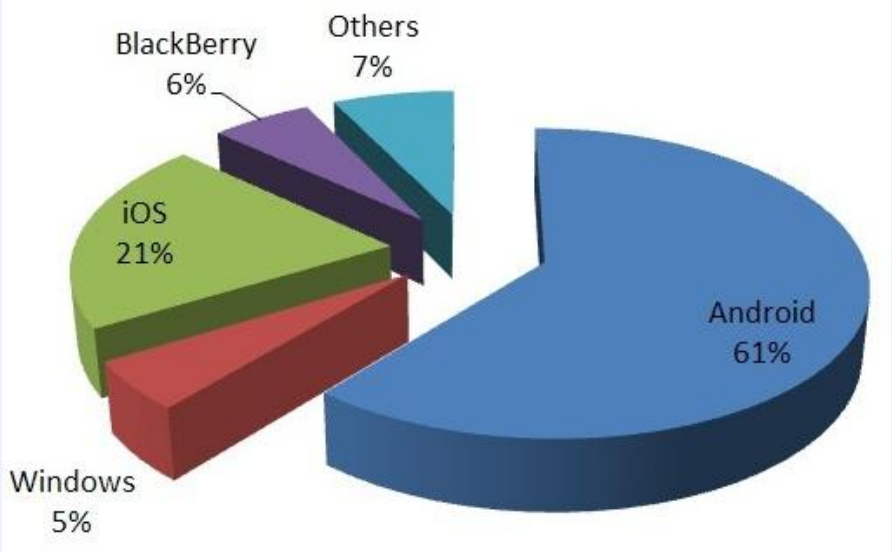
Hybrid App

¿Comunicaciones? Básicamente HTTP(S)

OWASP
The Open Web Application Security Project

Plataformas Actuales

(Market Share por Sistema Operativo)



Lenguaje de Programación Utilizado

Java

Objective-C

Java

.NET C#
Silverlight

OWASP
The Open Web Application Security Project

OWASP
Mobile Security Project

OWASP Mobile Security Project

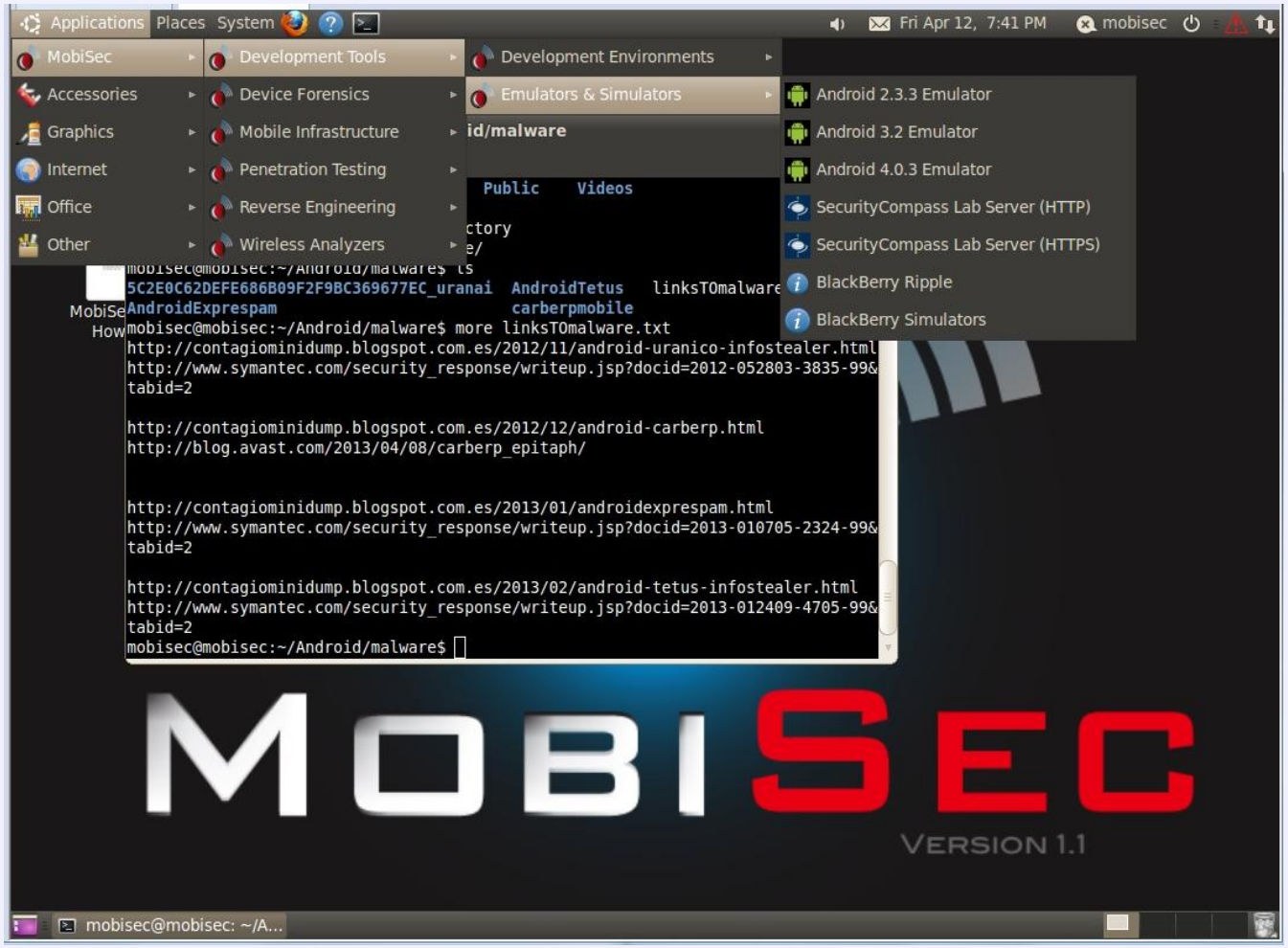
OWASP
The Open Web Application Security Project

- Presentado en AppSec 2011
- Mobile Threat Model.
- Mobile Testing Tools.
  - GoatDroid / iGoat. / DVIA
  - MobiSec.
  - Androick / NowSecure App Testing / Seraphimdroid
- Mobile Top 10 Risks. (Version final 2014)
- Mobile Top 10 Controls & Design Principles.
- Testing Guide.
- Development Guide. (muy verde todavía…)

https://www.owasp.org/index.php/OWASP_Mobile_Security_Project

MobiSec v1.3 (Nov. 2014)

OWASP
The Open Web Application Security Project



http://sourceforge.net/projects/mobisec/

OWASP Mobile Top 10 Risks (2014)

Call to Action for 2015

M1: Weak Server Side Controls

OWASP
The Open Web Application Security Project

- Nada nuevo por aquí...

- DO NOT TRUST THE CLIENT!

M2: Insecure Data Storage

OWASP
The Open Web Application Security Project

- Almacenamiento de información sin protección.
- Local / Nube.
- Cache / Malas Prácticas.

M3: Insufficient Transport Layer Protection

OWASP
The Open Web Application Security Project

- Aplicaciones que se comunican sin cifrado.
- Uso de cifradores débiles..
- No responder correctamente ante errores de certificados.

M4: Unintended Data Leakage

OWASP
The Open Web Application Security Project

- Leaking de información por usar librerías de terceros que no se controlan completamente.
  - Logs
  - Caches
  - Archivos temporales
  - Etc.

M5: Poor Authorization and Authentication

OWASP
The Open Web Application Security Project

- Un poco mas de lo mismo, pero con algunas variantes…
- Uso de datos del equipo para autenticación:
  - IMEI / IMSI
  - UUID
- Problema? Persisten con el equipo.

M6: Broken Cryptography

OWASP
The Open Web Application Security Project

- Ya es conocido pero vale la pena refrescar:
  - Encoding =! Encryption
  - Obfuscation =! Encryption
  - Serialization=! Encryption

- Utilización de algoritmos fuertes pero mal implementados.
  - Almacenamiento de claves junto con los datos.

- Programar su propia criptografía = generalmente malo.

M7: Client Side Injection

OWASP
The Open Web Application Security Project

- Aplicaciones Web / Hybrid.
- ¿XSS? ¿SQL Injection?
- Nuevos recursos. SMS, llamadas, servicios pagos.

“Mandá OWASP al 2020”

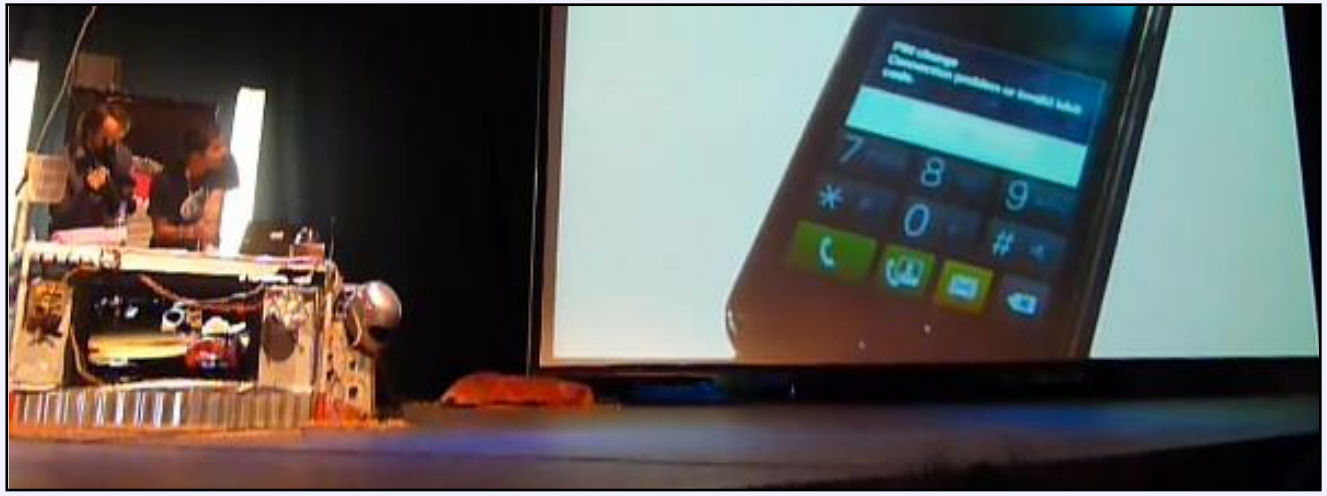
M8: Security Decisions Via Untrusted Inputs

OWASP
The Open Web Application Security Project

- Estas decisiones permiten saltear restricciones o modelos de seguridad.
  - iOS – Abuso de URL Schemes
  - Android – Abuso de Intents

- Vectores de ataque
  - Aplicaciones maliciosas.
  - Client-side Injections.

- Ej: Ravi Borgaonkar - ekoparty 2012
- Explotaba una vulnerabilidad en el intent “tel:” que permitía marcar un número desde otra aplicación y borrar cualquier teléfono Samsung.



```
<iframe src="tel:*2767*3855#" width="320" height="240"></iframe>
```

OWASP
The Open Web Application Security Project

- Esto ya lo conocemos…
  - Cookies, Oauth, Single-Sign-On, etc…
  - Sesiones mas duraderas que en Web.

M10: Lack of Binary Protection

OWASP
The Open Web Application Security Project

- Reversing del binario
- Análisis del código fuente
  - Lógica de la App.
  - Busqueda de vulnerabilidades más simple.
  - Leak de información en el código.

- Ya veremos más sobre esto…

OWASP Top 10 Mobile Controls

OWASP
The Open Web Application Security Project

- Por OWASP y la “European Network and Information Security Agency” (ENISA).
- Controles y buenas prácticas para evitar cada riesgo del Top 10.

OWASP
The Open Web Application Security Project

Pentesting Mobile Apps

(acá viene lo divertido…)

Tareas de un Pentest Mobile

OWASP
The Open Web Application Security Project

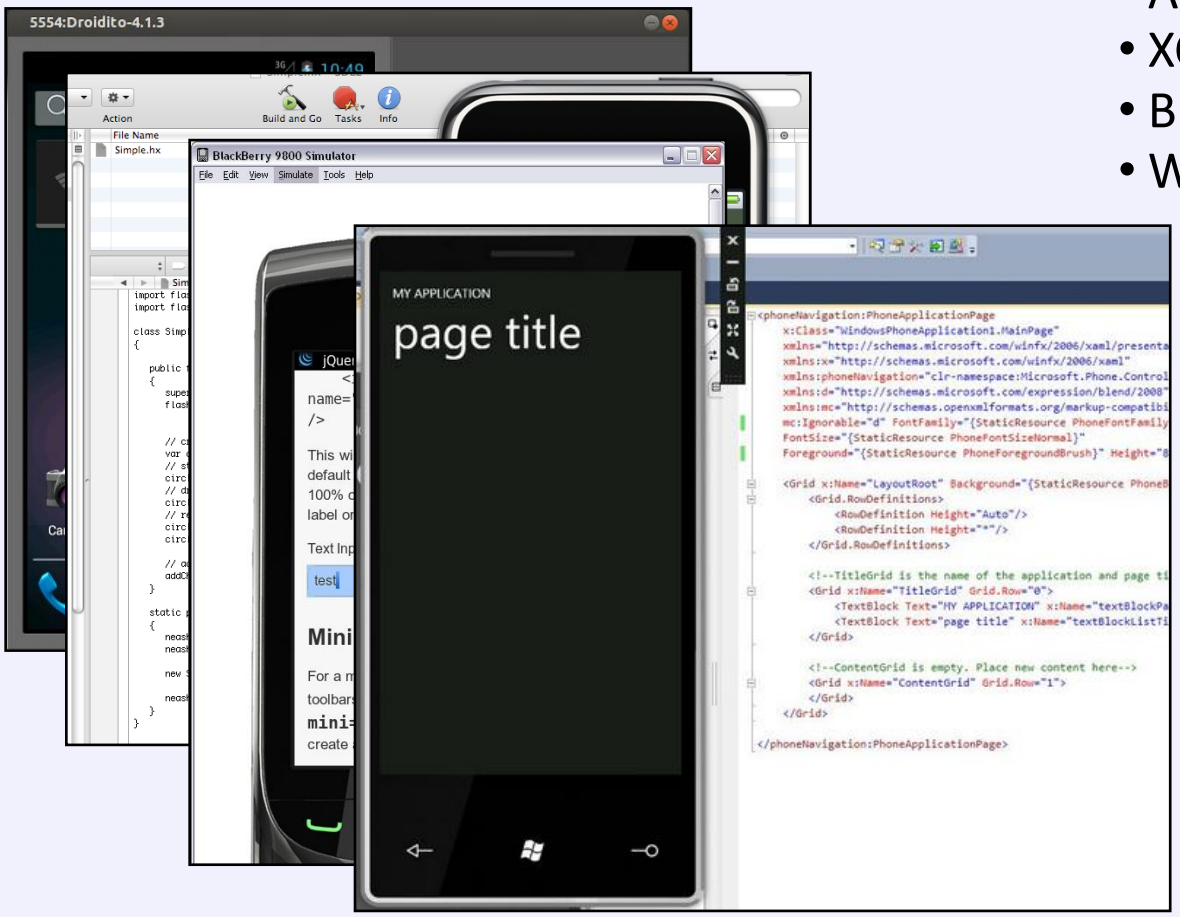
- Armado del laboratorio.
- Análisis y comprensión de la lógica de la app.
- Análisis de seguridad de los servidores.
- Análisis de los archivos creados por la app.
- Análisis de los datos en memoria.
- Ingeniería Inversa de código fuente / búsqueda de información sensible y vulns.
- Búsqueda de vulns a través del análisis y manipulación de la mensajería.

OWASP
The Open Web Application Security Project

Armado del Laboratorio

OWASP
The Open Web Application Security Project

Dispositivos Reales vs. Emuladores



- Android SDK Emulators
- XCode iOS Simulator
- Blackberry Simulators
- Windows Phone Emulator

OWASP
The Open Web Application Security Project

Análisis de Archivos

Análisis de Archivos

OWASP
The Open Web Application Security Project

- ¿Qué se busca?
- ¿Para que?
- Root/ Jailbreak necesario en dispositivos reales.

/data/data/<nombre_pkg>/

Shared Preferences

Base de datos SQLite

/private/var/mobile/ Applications/<id_app>

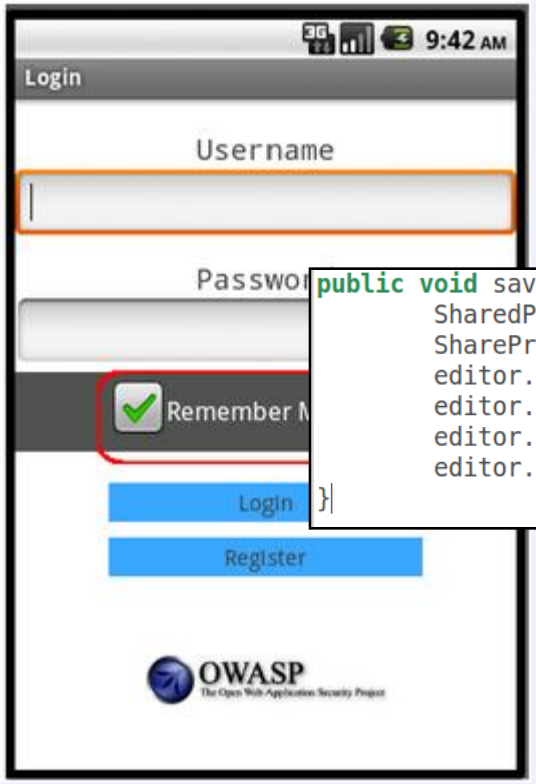
PropertyList (.plist)

Base de datos SQLite

Análisis de Archivos

OWASP
The Open Web Application Security Project

- Ejemplo:




```
public void saveCredentials (String userName, String password){
        SharedPreferences credentials = this.getSharedPreferences("credentials", MODE_WORLD_READABLE);
        SharePreferences.Editor editor = credentials.edit();
        editor.putString("username", userName);
        editor.putString("password", password);
        editor.putBoolean("remember", true);
        editor.commit();
}
```

Análisis de Archivos

OWASP
The Open Web Application Security Project

- Veamos algunas tools…

- adb: Android Debugging Bridge
  - android-sdk/platform-tools/adb
  - Dispositivo Físico = USB Debugging ON
  - http://developer.android.com/tools/help/adb.html
  - Comandos: devices, shell, pull, push, install, …

- SQLiteMan
  - apt-get install sqliteman

OWASP
The Open Web Application Security Project

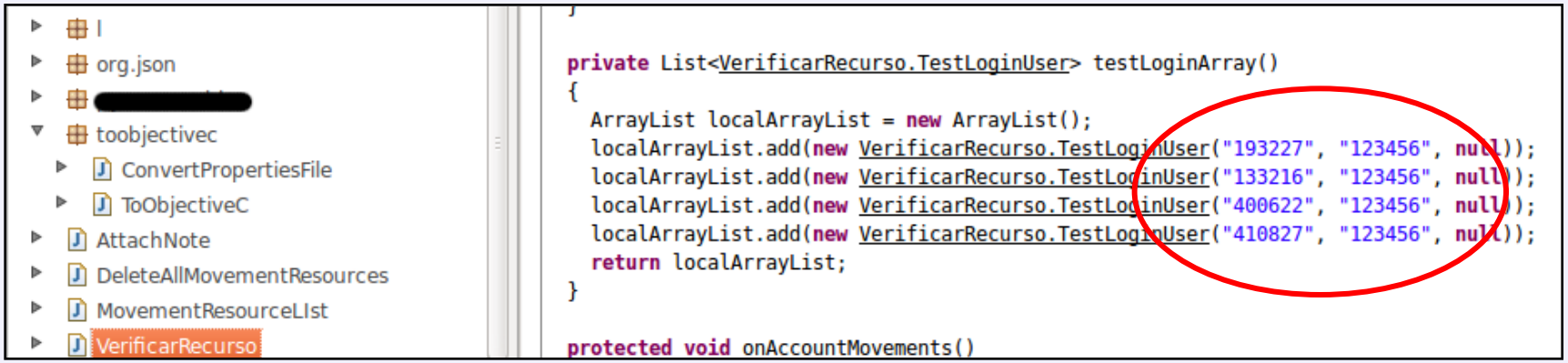
Reversing de Código Fuente

Reversing de Código Fuente

OWASP
The Open Web Application Security Project

- Binario compilado -> Código legible
- Bastante simple en Android
  - Podemos llegar al código Java completo.
  - APK = ZIP
- En iOS solo podríamos llegar a un código Assembler
  - Windows Phone y Blackberry mas simple.

- Ejemplo: Mobile Banking



OWASP
The Open Web Application Security Project

- Ejemplo: Padrón Electoral 2013

http://wsp.mininterior.gov.ar/ws_escuela.php?param=v%23v%23gWHVDcQRVRtNmMWRjY6FjT

```
public static String codificar(String s)
{
    String s1 = (new StringBuffer(
                (new StringBuilder(Base64.encodeBytes((new StringBuffer((
                 new StringBuilder(Base64.encodeBytes(s.getBytes())))).toString()
                 .replace("a", "#t").replace("e", "#x").replace("i", "#f").
                 replace("o", "#l").replace("u", "#7").replace("=", "#g"))).
                 reverse().toString().getBytes()))).toString().replace("a", "#j").
                 replace("e", "#p").replace("i", "#w").replace("o", "#8").replace("u", "#0")
                 .replace("=", "#v"))).reverse().toString();
    String s2;
    try
    {
        s2 = URLEncoder.encode(s1, "utf-8");
    }
    catch(UnsupportedEncodingException unsupportedencodingexception)
    {
        return s1;
    }
    return s2;
}
```

OWASP
The Open Web Application Security Project

¿Cómo hacemos? Vamos a las tools…

- APKDownloader
  - Extensión de Chrome.
  - http://apps.evozi.com/apk-downloader/
- APKTool
  - Permite decompilar y compilar código en formato SMALI (un assembler de DALVIK).
  - Útil para realizar cambios básicos (ej. Modificar valores)
  - https://code.google.com/p/android-apktool/

OWASP
The Open Web Application Security Project

Mas tools…

- dex2jar
  - Permite pasar de formato dex (Dalvik Executable) a JAR.
  - http://code.google.com/p/dex2jar/
- JDGUI
  - Permite decompilar un JAR a código JAVA.

¿Las probamos?

OWASP
The Open Web Application Security Project

¿Protección? ProGuard

com.whatsapp-41547_dex2jar.jar

- a
  - a
    - a : long
    - a(Reader)
    - readLine() : String
- ab
- ac
- b
- bb
- bc
- c
- cb
- d
- db
- e
- eb
- f
- fb
- g
- gb
- h
- hb
- i
- ib
- j
- jb

b.class | a.class | bb.class | c.class | db.class | d.class | cb.class | e.class | **eb.class** | f.class | fb.class

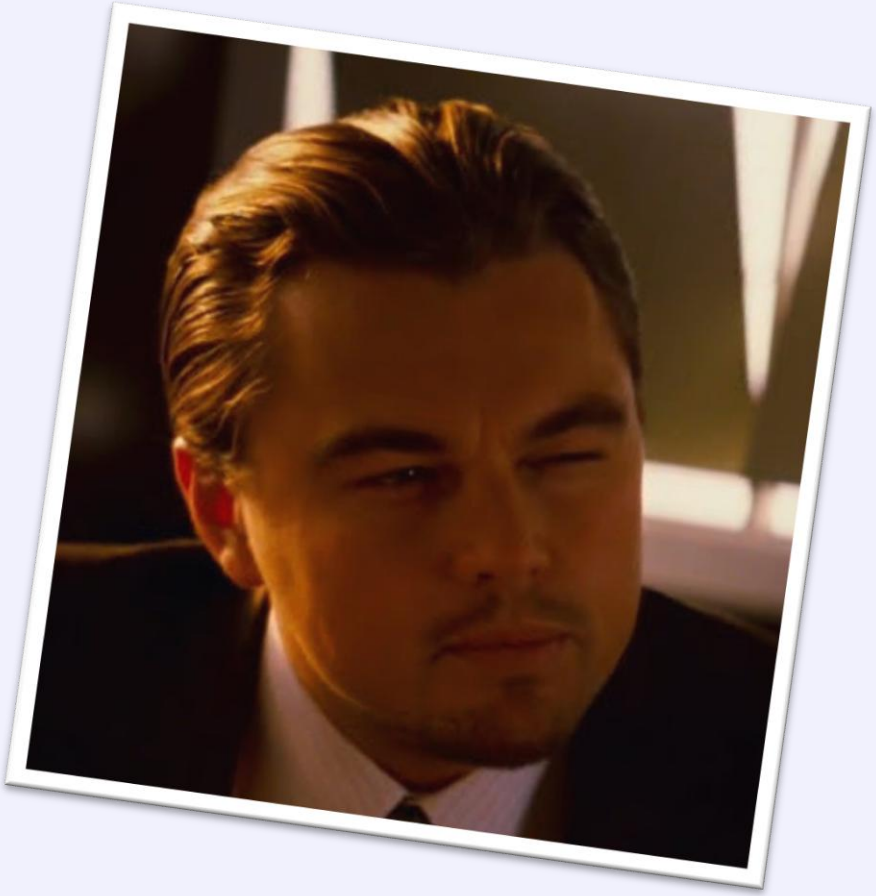
```
import com.whatsapp.App;

public class eb
  implements db, v
{
  private static final x[] a;
  private static final String[] z;
  private x[] b = a;
  private char[] c;
  private int d;
  private int e;
  private Object[] f;
  private int[] g;
  private int[] h;
  private int[] i;
  private int j;

  static
  {
    String[] arrayOfString = new String[13];
    char[] arrayOfChar1 = "t${]\f1t!\t\bt".toCharArray();
    int k = arrayOfChar1.length;
    int m = 0;
    char[] arrayOfChar2;
    int i3;
    char[] arrayOfChar3;
    int i7;
    char[] arrayOfChar4;
    int i11;
    char[] arrayOfChar5;
```

OWASP
The Open Web Application Security Project

Análisis de Lógica de la Aplicación



OWASP
The Open Web Application Security Project

- Activity: Clase que representa una acción que la App hace (generalmente una pantalla).
- Intent: Mensaje que una App puede enviar para ejecutar “activities” (propias o de otra App).
- Las activities disponibles para ejecutarse por Intents estan en el AndroidManifest.xml

- Tools…

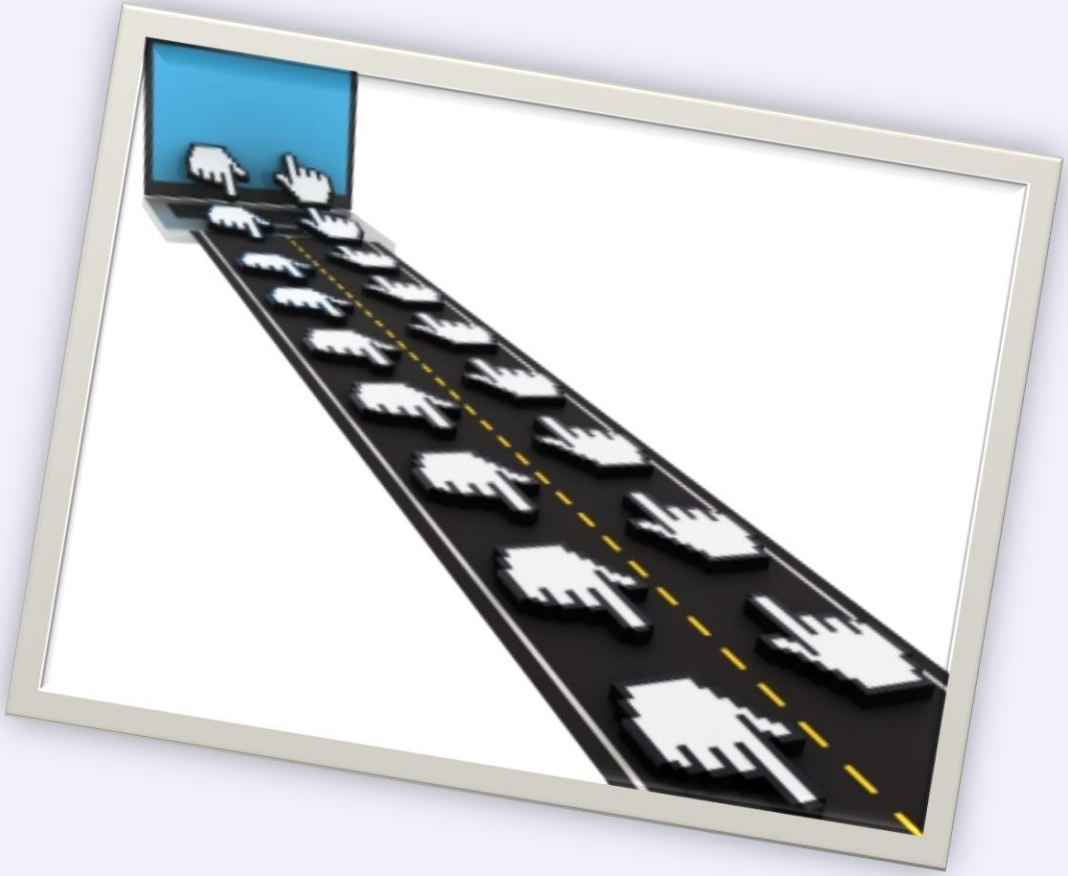
- am – activity manager
  - Incluído con ADB.
  - http://developer.android.com/tools/help/adb.html#am

Ejemplo:

```
$ adb shell
# am start -a android.intent.action.VIEW -d http://www.owasp.org/
```

OWASP
The Open Web Application Security Project

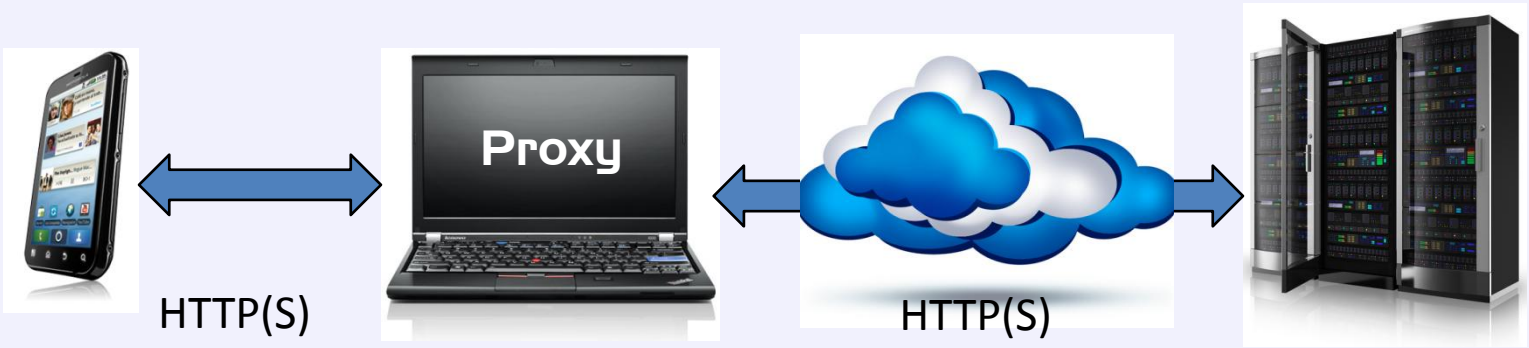
Análisis de Mensajería



Análisis de Mensajería

OWASP
The Open Web Application Security Project

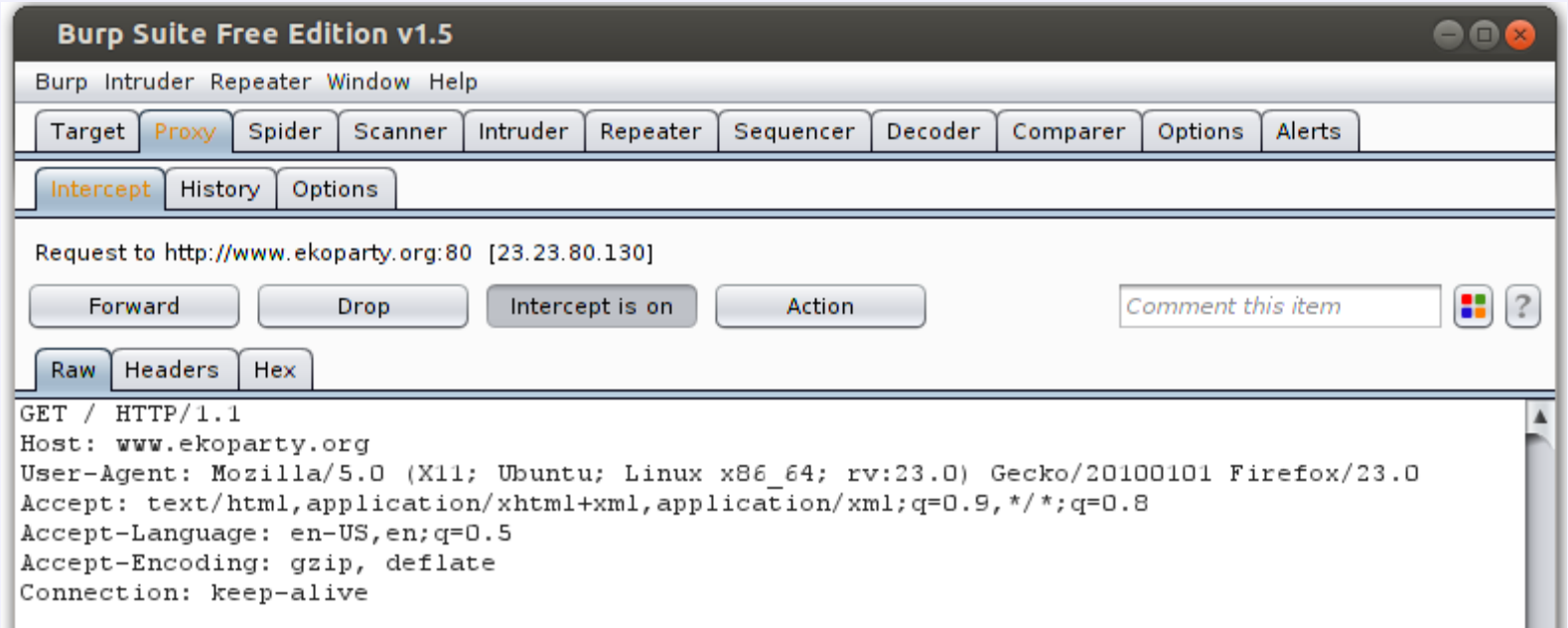
- Man-in-the-middle:



- Configuración de proxy:
  - Soportada por todos los emuladores.
  - Soportada en algunos terminales.
  - DNS Spoof / Packet Forwarding / Hosts.

OWASP
The Open Web Application Security Project

- HTTP(S) Proxy: Burp Suite
  - Free
  - http://portswigger.net/burp/
  - $./emulator @Droidito-4.1.3 -http-proxy 127.0.0.1:8080 -debug-proxy



Análisis de Mensajería

OWASP
The Open Web Application Security Project

- ¿Qué pasa con SSL?
  - El tratamiento de certificados depende de la aplicación.

- Certificate Pinning
  - Moxie Marlinspike:
    - “Your app shouldn't suffer SSL's problems”.
  - CA System vs. Keystores Propias.
  - Pinning del certificado o de la CA.
  - ¿Esto es Bueno o Malo?

Análisis de Mensajería

OWASP
The Open Web Application Security Project

¿Podemos saltear estas restricciones?

- Modo manual
  - Download/Decompile/Modify/Compile/Sign/Install
  - Tools:
    - keytool y jarsigner (incluídas en JDK)
    - Todas las anteriores…

- Modo Automático (todavía verde…)
  - Function hooking
  - Android SSL Bypass by ISEC Partners
    - Black Hat USA 2012
    - También iOS SSL KillSwitch

OWASP
The Open Web Application Security Project

¿Preguntas?

OWASP
The Open Web Application Security Project

Muchas gracias!!

gs@cintainfinita.com.ar

@iampuky

/in/gustavosorondo