UNIVERSIDAD NACIONAL DE LA PLATA

Sistemas Operativos

Práctica 6

Preguntas teóricas

Conceptos generales

1. ¿Qué es Android?

2. ¿Qué versión del kernel de Linux se está utilizando para desarrollar Android?

3. ¿Qué aprovecha Android de Linux?

4. ¿Cuál es la mascota de Android?, ¿Cuál es su etimología?

5. ¿Cuándo y por quién nació Android?

6. ¿Cuándo fue comprado por la empresa Google?

7. ¿Qué es la OHA?

8. ¿En qué año se lanzó la primera versión estable?

9. ¿Cuál es la última versión estable de Android?

10. ¿Con qué frecuencia se lanzan las versiones del SDK de Android?

11. Lea la página "Arquitectura de la Plataforma"[1] y conteste:

    (a) ¿Qué rol cumple la capa del kernel de Linux?

    (b) ¿Qué rol cumple la HAL?

    (c) ¿Qué rol cumple la capa de librerías?

    (d) ¿Qué rol cumple la capa de Android Runtime?

    (e) ¿Por qué fue reemplazada la DVM ?, ¿Qué mejora?

    (f) ¿Qué rol cumple la capa de Application Framework?

Aplicaciones

1. ¿Qué componentes de una aplicación Android existen? ¿para qué sirve cada uno?

2. ¿Qué es Gradle?, ¿Qué tarea permite realizar?

[1] https://developer.android.com/guide/platform/index.html

3. ¿A qué hacen referencia y para qué sirven las variables compileSdkVersion, minSdkVersion, targetSdkVersion y buildToolsVersion? ¿tienen que tener alguna relación entre ellas? ¿Qué significa y qué impacto tiene en la distribución de una aplicación a través de Google Play Store la variable versionCode?

4. ¿Todas las aplicaciones tienen que estar firmadas digitalmente para ejecutarse en un dispositivo?, ¿A través de qué mecanismo se realiza la firma digital?, ¿A través de qué herramienta se puede hacer?

5. ¿Qué es el application sandbox? (Vea la documentación correspondiente en source.android.com[2])

6. ¿Existe la posibilidad de que dos aplicaciones compartan el userId?, ¿Qué debe respetarse?

7. ¿Qué alternativas tienen las aplicaciones para compartir sus datos?.

8. ¿Dónde se define el acceso a los recursos por parte de las aplicaciones?, ¿Pueden realizarse cambios en tiempo de ejecución?, ¿Qué excepción se lanza si una aplicación intenta acceder a un recurso sobre el cual no definió permisos?

9. ¿En qué momento el usuario le da permiso a la aplicación para que esta utilice los recursos del dispositivo? ¿Se pueden definir de manera dinámica?

10. ¿Qué es un APK y qué contiene?

11. ¿Para qué sirve el archivo AndroidManifest.xml?. ¿Qué son los archivos .dex?

12. Detalle el proceso de firma (signing) de un APK.

Procesos

1. ¿Qué comandos de GNU/Linux tenemos disponibles en Android?

2. ¿Android cuenta con un área de intercambio? ¿por qué?

3. ¿Qué alternativa tiene a la hora liberar memoria? ¿Cómo se clasifican los procesos?

4. Lea el artículo sobre procesos y threads en developer.android.com[3] y responda:

(a) Por defecto, ¿en cuántos procesos/threads corren los componentes de una aplicación?

(b) ¿Los componentes de las aplicaciones pueden correr en procesos separados y/o en el mismo proceso?

(c) ¿Se pueden crear n threads de un proceso?

(d) ¿Los componentes de la vista (UI) de una aplicación son thread-safe?, ¿Qué reglas sigue el Android's single thread model?

[2] https://source.android.com/security/app-sandbox
[3] http://developer.android.com/guide/components/processes-and-threads.html

5. ¿Qué es DVM ?, ¿Por qué fue diseñada?. Compare este concepto con el de JVM. Adicionalmente compare el concepto de DVM con el de ART en base a la comparación en addictivetips.com[4] y el artículo de Wikipedia sobre el ART[5].

6. ¿Qué es y de qué se encarga Zygote?

Almacenamiento

1. ¿Qué alternativas tiene una aplicación para almacenar sus datos?. Detalle cada una.

2. ¿En dónde se almacenan las shared preferences de las aplicaciones?, ¿De qué tipo pueden ser?, ¿Una aplicación podría acceder a las shared preferences de otra diferente?

3. ¿En dónde se almacenan las bases de datos de las aplicaciones?

File system

1. ¿Cuáles son los puntos de montaje principales del File System de Android?, ¿Qué contiene cada uno?

2. ¿Con qué tipos de memoria cuenta un dispositivo móvil generalmente? Detalle cada uno y luego relaciónelos con los tipos de file system que cada tipo podría soportar.

3. Detalle las características del YAFFS.

Licencia

1. ¿Bajo qué licencias está el stack de Android?, ¿Por qué?

2. ¿Por qué compañías como Samsung pueden cambiar la interfaz de sus propias versiones de Android?

3. ¿Cómo se llama la libc de Android?, ¿Por qué fue implementada?

Rooteo

1. ¿Qué significa rootear un dispositivo Android?, ¿en qué se diferencia con el jailbreaking?

2. ¿Qué es el bootloader? ¿De qué se encarga? ¿Qué tipos existen? ¿Qué consecuencias trae desbloquearlo?

3. Un bootloader bloqueado permite cargar un sistema operativo particular, ¿Cómo se realiza esta identificación?

4. ¿Qué es el fastboot? ¿Qué acciones permite realizar?

---

[4] http://www.addictivetips.com/android/art-vs-dalvik-android-runtime-environments-explained-compared/

[5] https://en.wikipedia.org/wiki/Android_Runtime

5. ¿Qué es la boot.img?, ¿Con qué herramienta se la puede dividir y con cuál crear?, ¿Qué contiene?, ¿Qué nombre tiene la imagen del kernel de Linux?, ¿y la de Android?. ¿En qué formato está empaquetado y comprimido el initramfs?

6. ¿Para qué sirve el archivo default.prop que está dentro del initramfs?, ¿Qué significa el prefijo ro?, ¿A qué hace referencia la propiedad ro.secure?

Ejercicios prácticos

Requisitos

Para poder realizar los ejercicios prácticos deberá contar con las siguientes herramientas:

- No se recomienda hacer esta práctica en una máquina virtual ya que el emulador de Android requiere virtualización por hardware (si bien existe virtualización por hardware dentro de VirtualBox no es estable y el emulador no arranca correctamente)
- Java SE Development Kit: http://www.oracle.com/technetwork/java/javase/downloads. Puede guiarse con este[6] sitio.
- Última versión de Android Studio: https://developer.android.com/studio/#downloads
- Acceda al Android SDK Manager[7] desde Tools->Sdk Manager
- Descargue e instale/actualice los siguientes paquetes:
  - Android SDK Command-line Tools
  - Android SDK Platform-tools.
  - Android SDK Build-tools.
  - Android 11.0 (30) → SDK Platform.
  - Android Emulator.
- Para facilitar el desarrollo de los ejercicios, agregue los siguientes directorios a la variable de entorno PATH:
  - <android-sdk-directory>/platform-tools/
  - <android-sdk-directory>/tools/[8]

Puesta a punto

- Cree un dispositivo virtual desde el IDE (Tools->AVD Manager). Por ejemplo puede crear un dispositivo “Pixel 4”. Elija como sistema operativo Android R (API level 30) con ABI para x86.

Inicie el emulador desde AVD Manager (AVD significa Android Virtual Device).

Ejecute el siguiente comando para obtener acceso como root al dispositivo:

```
$ adb root
```

Acceda a la shell del dispositivo:

```
$ adb shell
```

[6] https://www.java.com/en/download/help/download_options.xml
[7] Para ejecutar el comando directamente agregue la herramienta “android” a la variable $PATH. Si instalo Android Studio, podrá instalar las dependencias desde una ventana gráfica.
[8] Puede encontrarse con el nombre “emulator-x86”.

Esto creará un cliente adb e iniciará el servidor adb, el cual se comunicará con el demonio adbd del dispositivo ejecutándose en background.

Observe el sonido de notificación configurado con (observe el uso de comillas en el comando grep):

```
# grep '"notification_sound"' /data/system/users/0/settings_system.xml
```

Responda:

1. ¿Qué pasa con este archivo si abrimos la configuración del dispositivo desde la interfaz gráfica (Menú ->Settings ->Sound) y modificamos el sonido de las notificaciones? (Asegúrese de hacer clic en save).

2. ¿Y qué pasa si ejecutamos el comando "rm /data/system/users/0/settings_system.xml" y luego reiniciamos el dispositivo con "reboot"?, ¿Sigue funcionando el sistema operativo?, ¿Qué pasa con las configuraciones?

Tipo de memoria y File system

¿Qué filesystems utiliza el dispositivo virtual creado? (Tip: utilice el comando mount)

Aplicaciones

En Android, la forma de ejecutar aplicaciones es mediante la herramienta am. La misma provee la funcionalidad necesaria para ejecutar activities, services, entre otras cosas. Para su ayuda, ejecute:

```
# am
```

Ejecute la actividad main de la aplicación com.android.chrome:

```
# am start -a android.intent.action.MAIN -n
com.android.chrome/com.google.android.apps.chrome.Main
```

¿Cómo funciona?:

• El primer parámetro indica que se va a hacer. En este caso, se ejecuta una aplicación (start).
• El segundo parámetro indica el tipo de acción que se ejecutará (android.intent.action.MAIN).
Nota: para ver más tipos de acciones visite esta página[9].
• El tercer parámetro es la actividad que se desea mostrar (com.android.chrome/com.google.android.apps.chrome.Main).

Inicie la aplicación settings del dispositivo:

```
# am start -a android.intent.action.MAIN -n com.android.settings/.Settings
```

[9] http://developer.android.com/guide/topics/intents/intents-filters.html

Procesos y Usuarios

Abra, desde la interfaz gráfica, el browser que viene instalado por defecto. ¿Cómo identificamos al proceso? ¿Podemos matarlo? ¿Cómo?

Ejecute, desde la interfaz gráfica, más de una instancia del browser que viene instalado por defecto. ¿Qué puede deducir al respecto? (Tip: utilice el comando ps)

Entendiendo el sandbox:

1. Cree tres proyectos desde el IDE (por ejemplo con una basic activity en el lenguaje Java):

- nombre: “App uno” dominio: unlp.so.android.uno
- nombre: “App dos” dominio: unlp.so.android.dos
- nombre: “App tres” dominio: unlp.so.android.tres

2. Agregue, al tag manifest del archivo AndroidManifest.xml (<directorio-raíz-proyecto>/app/src/main/AndroidManifest.xml), el atributo sharedUserId con el mismo valor a las aplicaciones uno y dos:

```
android:sharedUserId="unlp.so.android.uno"
```

3. Adicionalmente agregue, al archivo AndroidManifest.xml, permisos de Internet para la aplicación uno:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="unlp.so.android.uno" >
...
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
...
</manifest>
```

4. Construya las aplicaciones mediante el IDE usando “Build->Make project”. Podrá ver los .apk generados en (<directorio-raíz-proyecto>/app/build/outputs/apk/).

5. Instale las aplicaciones en el dispositivo (requiere que el dispositivo/emulador esté conectado/iniciado):

```
$ adb install <debug-apk-generado-proyecto-uno>
$ adb install <debug-apk-generado-proyecto-dos>
$ adb install <debug-apk-generado-proyecto-tres>
```

6. Ejecute cada una de las aplicaciones.

7. ¿Qué información puede extraer como resultado de ejecutar los siguientes comandos? Compare los userId de las aplicaciones uno, dos y tres:

```
$ adb root
$ adb shell
# ps -fA
# dumpsys package unlp.so.android.uno
# dumpsys package unlp.so.android.dos
# dumpsys package unlp.so.android.tres
```

8. Acceda al archivo android_filesystem_config.h[10] (en el cual se definen los UID y GID del sistema) y responda:

a. ¿Cuál es el userId del usuario system?
b. ¿A partir de qué userId Android concede identificación para las aplicaciones de usuario?
c. ¿Qué userId se le asigna al demonio adb (adbd)?

Nota: El que desee hacerlo puede leer el artículo (https://users.encs.concordia.ca/~clark/papers/2012_spsm.pdf) en donde se explica en detalle la asignación del userId al momento de instalar una aplicación en Android.

10 https://android.googlesource.com/platform/system/core.git/+/master/libcutils/include/private/android_filesystem_config.h