Apuntes Android

[Generalidades: 2](#_Toc169124585)

[Androismos 2](#_Toc169124586)

[Medidas de seguridad: 4](#_Toc169124587)

[Personalización: 4](#_Toc169124588)

[Herramientas para desarrolladores: 5](#_Toc169124589)

[Control de privacidad: 5](#_Toc169124590)

[Desarrollado por Google: 5](#_Toc169124591)

[Basado en Linux: 5](#_Toc169124592)

[Código abierto: 6](#_Toc169124593)

[Multiplataforma: 6](#_Toc169124594)

[Google Play Store: 6](#_Toc169124595)

[Integración con servicios de Google: 6](#_Toc169124596)

[Actualizaciones de software: 7](#_Toc169124597)

[Boletines de seguridad de Android: 8](#_Toc169124598)

[INICIALIZACION: 9](#_Toc169124599)

[Procesos: 14](#_Toc169124600)

[Mecanismo Binder en Android: 17](#_Toc169124601)

[File System: 19](#_Toc169124602)

[partición /data: 21](#_Toc169124603)

[Ventajas de tener una partición /data separada: 22](#_Toc169124604)

[Memorias: 29](#_Toc169124605)

[Data 29](#_Toc169124606)

[Boot 29](#_Toc169124607)

[Cache 29](#_Toc169124608)

[System 30](#_Toc169124609)

[Recovery 30](#_Toc169124610)

[Misc 30](#_Toc169124611)

[Retención de Memoria: 31](#_Toc169124612)

[Acceso a la Memoria: 31](#_Toc169124613)

[Proceso Zygote: 31](#_Toc169124614)

[División de Memoria 31](#_Toc169124615)

[MEMORIA VIRTUAL: 31](#_Toc169124616)

[Memoria nombrada (mmapped): 32](#_Toc169124617)

[Memoria anónima: 32](#_Toc169124618)

[Memoria limpia (Clean) vs. Memoria sucia (Dirty): 32](#_Toc169124619)

[Memoria activa vs. Memoria inactiva: 32](#_Toc169124620)

# Generalidades:

* **Relación entre Android y Linux**:
  + Android se basa en el kernel de Linux, pero con modificaciones sustanciales que afectan la compatibilidad con el sistema principal.
  + El árbol fuente del kernel de Android divergió del kernel principal alrededor de la versión 2.6.27, pero ha estado convergiendo desde la versión 3.3.
  + A nivel de kernel, Android y Linux son aproximadamente un 95% similares, aunque hay diferencias (llamadas “Androidismos”) relacionadas con IPC, memoria y registro.
  + A nivel de modo de usuario, la divergencia es mayor, introduciendo componentes como el tiempo de ejecución Dalvik y la Capa de Abstracción de Hardware (HAL).

# Androismos

* **ASHMem (Memoria Compartida Anónima)**: Un mecanismo que permite la memoria compartida en Android.
* **Binder**: El componente central para la comunicación entre procesos (IPC) en Android.
* **Logger**: Proporciona buffers de anillo basados en el kernel para un registro rápido sin archivos.
* **Asignador de Memoria ION**: Ofrece una asignación de memoria eficiente para controladores de kernel y modo de usuario.
* **Low Memory Killer**: Una capa encima del propio killer de OOM (agotamiento de memoria) de Linux.
* **Consola RAM**: Un mecanismo para preservar la salida de pánico del kernel.
* **Controlador de Sincronización**: Introducido para permitir primitivas de sincronización rápidas.
* **Salida Temporizada y GPIO**: Permiten a los programas de modo de usuario acceder a los registros GPIO desde el espacio de usuario.
* **Wakelocks**: Originalmente un “Androidismo” separado para controlar la gestión de energía.
* SDK de Android y sus componentes:
* Herramientas de Desarrollo: Incluye depurador de código, biblioteca, simulador de teléfono, documentación, ejemplos de código y tutoriales.
* Plataformas Soportadas: Compatible con Linux, Mac OS X 10.4.9+, y Windows XP+.
* IDE Oficial: Eclipse con el complemento ADT, aunque se puede usar un editor de texto y comandos en terminal con JDK y Apache Ant.
* Actualizaciones y Soporte: El SDK se actualiza con el desarrollo de Android y soporta versiones antiguas para compatibilidad.
* Aplicaciones Android: Compuestas por archivos. apk que incluyen ficheros. dex y recursos, almacenados en /data/app con permisos de superusuario.
* **Lenguajes y entornos de desarrollo en Android**:
  + **Java**: Ha sido el lenguaje más importante para el desarrollo en Android. Está estrechamente relacionado con Android Studio y se utiliza para crear APIs y entornos de desarrollo de aplicaciones. También se ha utilizado junto con la plataforma Eclipse.
  + **XML**: Un lenguaje de marcado extensible utilizado principalmente para diseñar interfaces de usuario en Android.
  + **Kotlin**: Lenguaje estático y de código abierto diseñado para ser totalmente interoperable con Java. Google declaró a Kotlin como su lenguaje de programación preferido para el desarrollo de aplicaciones Android en mayo de 2019.
  + **Android Studio**: El entorno de desarrollo integrado oficial para Android, anunciado en 2013. Reemplazó a Eclipse como IDE oficial y está basado en IntelliJ IDEA Community Edition de JetBrains.

# Medidas de seguridad:

* + Android incorpora diversas medidas de seguridad, como el cifrado de datos.
  + Realiza escaneos de aplicaciones en busca de malware mediante Google Play Protect.
  + Ofrece autenticación biométrica, como el reconocimiento facial y la huella dactilar.

# Personalización:

* + Los usuarios pueden personalizar no solo la interfaz de usuario, sino también otros aspectos del sistema.
  + Pueden ajustar configuraciones de notificación, permisos de aplicaciones y accesos directos rápidos.

# Herramientas para desarrolladores:

* + Android proporciona herramientas avanzadas de desarrollo, como Android Studio.
  + Android Studio incluye un editor de código robusto, emuladores de dispositivos, herramientas de depuración y perfiles de rendimiento.

# Control de privacidad:

* + Los usuarios tienen un mayor control sobre su privacidad y datos personales.
  + Pueden revisar y ajustar fácilmente la configuración de privacidad, gestionar permisos de aplicaciones y controlar qué datos comparten con las aplicaciones.

# Desarrollado por Google:

* + Android es un sistema operativo móvil desarrollado por Google.
  + Inicialmente, fue creado por Android Inc., que Google adquirió en 2005.

# Basado en Linux:

* + Android utiliza el núcleo del sistema operativo Linux como base.
  + Esto proporciona una plataforma sólida y estable para su funcionamiento.

# Código abierto:

* + Android es de código abierto, lo que significa que su código fuente está disponible públicamente.
  + Los desarrolladores pueden estudiarlo, modificarlo y distribuirlo según las licencias de código abierto (como la Licencia Pública General de GNU y la Licencia Apache).

# Multiplataforma:

* + Android es compatible con diversos dispositivos, como teléfonos inteligentes, tabletas, relojes inteligentes, televisores y dispositivos IoT.
  + Su flexibilidad ha contribuido a su popularidad y adopción generalizada.

# Google Play Store:

* + La tienda de aplicaciones Google Play Store es la fuente principal para descargar aplicaciones en dispositivos Android.
  + Ofrece una amplia variedad de aplicaciones, juegos, libros, música y películas.

# Integración con servicios de Google:

* Android está estrechamente integrado con servicios como Google Search, Google Maps, Gmail, Google Drive y Google Photos.
* Los usuarios pueden acceder directamente a estos servicios desde sus dispositivos Android.

# Actualizaciones de software:

* + Las actualizaciones de software son gestionadas por fabricantes y operadores de telefonía móvil.
  + Google proporciona actualizaciones periódicas con mejoras de seguridad, correcciones de errores y nuevas características.

arquitectura del sistema operativo Android y el Proyecto Treble:

* Arquitectura Modular:
* Android está diseñado con una arquitectura modular que permite actualizar componentes específicos sin afectar todo el sistema operativo.
* A partir de Android 8.0 (Oreo), se rediseñó el marco del sistema operativo para facilitar y abaratar las actualizaciones de dispositivos a nuevas versiones de Android.
* Interfaz HAL (HIDL):
* El lenguaje de definición de interfaz HAL (HIDL) especifica la comunicación entre una HAL y sus usuarios.
* Permite reemplazar el marco de Android sin reconstruir las HAL.
* Proyecto Treble:
* Treble separa la implementación del proveedor (software específico del dispositivo) del marco del sistema operativo.
* Los proveedores construyen HAL una vez y la colocan en la partición /vendor del dispositivo.
* El marco se puede actualizar sin recompilar las HAL, lo que facilita las actualizaciones inalámbricas (OTA).
* Diferencia con la Arquitectura Heredada:
* En versiones anteriores a Android 8.0, no existía una interfaz de proveedor formal, lo que dificultaba las actualizaciones.
* A partir de Android 8.0, una nueva interfaz de proveedor estable permite a los fabricantes ofrecer nuevas versiones de Android sin modificar gran parte del código.

# Boletines de seguridad de Android:

* **Actualizaciones mensuales**: Se lanzan actualizaciones de seguridad mensuales para Android, conocidas como “Android Security Bulletins”. Estos abordan vulnerabilidades y mejoran la seguridad del sistema operativo. Los boletines se publican el primer lunes de cada mes, a menos que sea festivo. En ese caso, se publican el siguiente día hábil.
* **Fabricantes y chipsets**: Además, los fabricantes de dispositivos y chipsets Android pueden publicar detalles específicos de vulnerabilidades de seguridad en sus productos. Algunos ejemplos de fabricantes son Google, Huawei, LG, Motorola, Nokia, OnePlus, Oppo, Samsung y Vivo.
* **Fuentes de correcciones**:
  + Las correcciones de la plataforma Android provienen del Proyecto de código abierto de Android (AOSP) y se fusionan en AOSP entre 24 y 48 horas después de que se publica el boletín de seguridad.
  + Las correcciones del kernel de Linux están vinculadas directamente desde el boletín en el momento del lanzamiento.
  + Las correcciones de los fabricantes de SOC (sistemas en chip) están disponibles directamente de los fabricantes.

# INICIALIZACION:

proceso de arranque en Android:

* **Boot ROM**:
  + Al encender el dispositivo, la CPU se activa y busca la dirección de memoria donde se encuentra el boot ROM.
  + El código del boot ROM inicializa el hardware del dispositivo y busca la partición de arranque en la memoria NAND.
  + Luego, copia el boot loader en la RAM.
* **Boot Loader**:
  + La CPU ejecuta el Boot Loader, cuya función es lanzar el sistema operativo Android.
  + El Boot Loader prepara la inicialización del kernel.
* **Kernel**:
  + El kernel inicializa la memoria, dispositivos, schedulers y el acceso a la partición raíz.
  + Luego, lanza el proceso init.
* **Init**:
  + El proceso init se encarga de ejecutar los scripts de inicio ubicados en /etc/rcX.d.
  + El script principal que se ejecuta es init.rc, que contiene la configuración inicial del sistema.
* **System Server**:
  + Una vez iniciado el servicio, el sistema puede lanzar servicios básicos como telefonía, teclado y otros.
  + Finalizado el inicio de estos servicios, el sistema está listo para interactuar con el usuario.
* Zygote en Android:
* El proceso Zygote es fundamental para el arranque eficiente de las aplicaciones en dispositivos Android. Cuando inicias tu dispositivo, Zygote se inicia antes que cualquier otra aplicación. Su función principal es cargar las bibliotecas comunes de Java y los recursos del sistema que todas las aplicaciones necesitarán utilizar. Esto mejora la eficiencia y velocidad de inicio de las aplicaciones.
* Zygote es el proceso inicial en Android y se inicia después de que el kernel de Linux haya completado su carga y configuración inicial. Carga las bibliotecas del sistema y la máquina virtual Android Runtime (ART), permaneciendo en espera. Cuando se necesita iniciar una nueva aplicación, Zygote crea una copia de sí mismo mediante una técnica llamada “forking”. Este método es eficiente porque permite que todas las aplicaciones compartan el mismo código base de la máquina virtual y las bibliotecas en memoria, reduciendo así el uso de recursos.
* Boot ROMCuando se enciende el dispositivo, lo primero que se pone en marcha es la CPU. Ésta tiene hardcoreada la dirección de memoria donde se encuentra el código de inicio, o boot ROM. Lo que hace este código es inicializar el resto de hardware del dispositivo. Hecho esto, busca la partición de arranque, que suele estar en la memoria NAND, y copia elboot loader en la RAM.

Diagrama

Descripción generada automáticamente

Diagrama

Descripción generada automáticamente

* Kernel: La carga del kernel de Android es igual que cualquier sistema Linux. Inicializa memoria, dispositivos, schedulers, y acceso a la partición raiz para lanzar el proceso init

Diagrama

Descripción generada automáticamente

* init:
* Diagrama

  Descripción generada automáticamenteAl igual que en un sistema Linux, el proceso init se encarga de lanzar los scripts de inicio contenidos en /etc/rcX.d. En este caso el script que se ejecuta es init.rc que se encuentra en raiz, aunque el fichero o la ubicación puede variar dependiendo del dispositivo. Contiene toda la configuración inicial del sistema. Desde el punto de vista del analista forense, es la parte de inicio más suculenta, por lo que, si os pica la curiosidad, os recomiendo que busquéis este archivo y le echéis un ojo. Necesitareis eso si un dispositivo rooteado y un explorador de ficheros que permita navegar por directorios con permisos de root.
* System server  
  Server Iniciado el servicio Zygote, el sistema ya puede empezar a lanzar servicios básicos, como la telefonía, teclado, y otros agentes y servicios de Google. Finalizado el inicio de estos servicios, el sistema está listo para interactuar con el usuario.

Gráfico

Descripción generada automáticamente

* Texto, Aplicación, Chat o mensaje de texto

  Descripción generada automáticamenteDiagrama

  Descripción generada automáticamenteEl proceso de inicialización en Android utiliza muchos componentes esenciales los cuales juegan un papel crucial en el proceso de arranque, el cual es el nucleo en todo el ecosistema de Android.  
  Acá se ve los componentes los cuales van a participar en este proceso.

"**Aplicattion Startups**"

* **Aplicaciones del Sistema**
* Las aplicaciones del sistema son conocidas como aplicaciones integradas, que normalmente se instalan en la partición del sistema del dispositivo durante el proceso de integración del binario final. Estas aplicaciones proporcionan funcionalidades y servicios esenciales para el sistema Android y son responsables de varios aspectos del funcionamiento del dispositivo.
* **Ejemplo**: Launcher, System UI, Configuración
* Las aplicaciones del sistema tienen privilegios y permisos especiales, lo que les permite acceder a recursos y servicios a nivel del sistema. Son esenciales para el funcionamiento adecuado del dispositivo. Durante el proceso de arranque de Android, las aplicaciones del sistema se inician temprano, a menudo por el servidor del sistema (SystemServer). Proporcionan funcionalidades básicas del sistema, y su inicialización es crítica para la estabilidad y la usabilidad del dispositivo.
* **Aplicaciones de Usuario**
* Las aplicaciones de usuario no forman parte del núcleo del sistema operativo Android y no vienen preinstaladas en el dispositivo. Sirven para una amplia variedad de propósitos, desde la productividad y el entretenimiento hasta juegos y utilidades.

# Procesos:

* **Gestión de Procesos en Android**:
* **Multitarea e Hilos**: Android utiliza multitarea con múltiples hilos por proceso, permitiendo un hilo principal y subprocesos adicionales.
* **Tipos de Procesos**:
* **Activos**: En ejecución e interacción con el usuario, como procesos en primer plano o servicios de soporte.
* **Visibles**: En segundo plano, pero aún afectan lo que el usuario ve.
* **Servicio**: Importantes para el usuario, como mantener conexión a internet, no visibles directamente.
* **Segundo Plano**: Ejecutándose sin ser visibles, reciben recursos si están disponibles.
* **Vacíos**: Procesos en espera de recursos para iniciar.
* **Expropiativa**: La planificación en Android es expropiativa, lo que significa que el sistema puede finalizar aplicaciones si no se gestionan adecuadamente los componentes. Esto es importante para mantener la fluidez y liberar recursos para las aplicaciones prioritarias.
* **Algoritmo de Planificación**: Android utiliza un algoritmo Round-Robin con prioridades basadas en la jerarquía de importancia de los tipos de procesos. Por ejemplo, los procesos activos y visibles tienen prioridad sobre los procesos en segundo plano.
* **Lista Pseudo-LRU**: El sistema mantiene una lista pseudo-LRU para evitar que los procesos acaparen recursos. La prioridad de un proceso puede aumentar si es un subproceso de mayor prioridad.
* **Caché (zRAM)**: Los procesos en espera o finalizados se almacenan en caché (zRAM) para acelerar su carga en RAM si se vuelven a solicitar.
* **Comunicación entre Procesos**: Los procesos en Android se comunican mediante paso de mensajes de tipo objeto utilizando el IPC Binder, que permite la sincronización y obtención de información entre procesos independientes del sistema
* La comunicación entre procesos (IPC) en Android es esencial para que las aplicaciones interactúen entre sí. El Administrador de Servicios (servicemanager) desempeña un papel crucial en este proceso.
* **Administrador de Servicios**: Es un directorio centralizado para todos los servicios del sistema operativo Android. Cuando una aplicación o componente necesita interactuar con otro servicio, consulta al Administrador de Servicios para obtener un identificador. Los servicios también se registran con él para que los clientes puedan encontrarlos y comunicarse. Si el Administrador de Servicios se reinicia, todos los servicios dependientes deben reiniciarse.
* **Componente Clave**: El Binder es el componente clave en el modelo de IPC de Android. Es un mecanismo del núcleo al que los servicios de modo de usuario acceden para la comunicación. Se utiliza a través de un nodo de dispositivo de caracteres (/dev/binder). Solo un proceso de modo de usuario puede registrarse como administrador de contexto con el **Binder**. Una vez registrado, actúa como punto focal para procesos clientes y servidores que deben registrar su nombre de servicio e interfaz.
* **Consulta de Clientes**: Los clientes consultan al administrador de contexto para encontrar y comunicarse con los servicios requeridos.

# Mecanismo Binder en Android:

* **Rendimiento**: El Mecanismo Binder es altamente eficiente y puede manejar grandes cantidades de datos con un bajo overhead computacional.
* **Seguridad**: Proporciona un alto nivel de seguridad al evitar que los procesos accedan directamente a la memoria de otros procesos.
* **Flexibilidad**: Es un marco flexible que se puede utilizar para implementar una amplia gama de tipos de Comunicación entre Procesos (IPC).
* **El Mecanismo Binder** es esencial en el marco de trabajo de Android y se utiliza para implementar funciones como el sistema de mensajería, el marco de actividad y el administrador de ventanas.
* **Proceso del sistema (System Process):** Este proceso es el proceso principal del sistema operativo Android. Controla y coordina las actividades de otros procesos del sistema, como el administrador de ventanas, el administrador de paquetes, el administrador de servicios y otros componentes esenciales del sistema.  
   **Administrador de ventanas (Window Manager)**: Es responsable de la gestión de las ventanas de las aplicaciones y de la interfaz de usuario del sistema. Controla la disposición de las ventanas, la superposición de ventanas y la interacción del usuario con la pantalla táctil.  
   **Administrador de paquetes (Package Manager):**Gestiona la instalación, desinstalación y actualización de aplicaciones en el dispositivo. También se encarga de gestionar los permisos de las aplicaciones y garantizar la seguridad del sistema.
* **El ciclo de vida de la actividad en Android** se extiende desde la creación de la actividad hasta su destrucción, cuando el sistema recupera los recursos de esa actividad. A medida que un usuario navega dentro y fuera de una actividad, esta pasa por diferentes estados en su ciclo de vida. Por ejemplo, cuando el usuario cambia de una pantalla a otra o abre una nueva aplicación, el sistema “pospone” el proceso de la actividad y guarda su estado para recuperarlo más tarde. Aunque el proceso y la actividad siguen existiendo en el sistema, están en modo “standby” y su estado se ha guardado para su posterior uso.
* **-OnCreate (), onDestroy()**

Abarcan todo el ciclo de vida. Cada uno de estos métodos, representan el principio y el fin de la actividad.

* **-OnStart (), onStop()**

Representan la parte visible del ciclo de vida. Desde onStart () hasta onStop(), la actividad será visible para el usuario, aunque es posible que no tenga el foco de acción por existir otras actividades superpuestas con las que el usuario está interactuando. Pueden ser llamados múltiples veces.

* **-OnResume(), onPause()**

Delimitan la parte útil del ciclo de vida. Desde onResume() hasta onPause(), la actividad no sólo es visible, sino que además tiene el foco de la acción y el usuario puede interactuar con ella.

* El proceso que mantiene a esta Activity puede ser eliminado cuando se encuentra en onPause() o en onStop(), es decir, cuando no tiene el foco de la aplicación. (Android nunca elimina procesos con los que el usuario está interactuando en ese momento). Una vez se elimina el proceso, el usuario desconoce dicha situación y puede incluso volver atrás y querer usarlo de nuevo. Entonces el proceso se restaura gracias a una copia y vuelve a estar activo como si no hubiera sido eliminado.

# File System:

* + particiones del sistema de archivos en Android:
* **/system**: Contiene los archivos del sistema operativo Android, incluyendo binarios del sistema y bibliotecas compartidas.
* **/data**: Almacena datos de aplicaciones y usuarios, como archivos de configuración, bases de datos y otros datos generados por las aplicaciones.
* **/cache**: Utilizado para almacenar datos temporales y cachés que no necesitan persistencia a largo plazo.
* **/sdcard** o **/storage**: Punto de montaje para el almacenamiento externo, como tarjetas SD o memoria interna.
  + Además:
* Android utiliza un **sistema de archivos virtual (VFS)** que permite a las aplicaciones acceder a recursos de almacenamiento independientemente de su ubicación física.
* Existe un **almacenamiento compartido** donde las aplicaciones pueden acceder a los archivos de otras aplicaciones con los permisos adecuados.
* Las aplicaciones acceden al sistema de archivos a través de la **API de Android File System**, que permite leer y escribir archivos.
* Los dispositivos Android suelen tener **almacenamiento externo adicional**, como tarjetas SD.
* Las aplicaciones también pueden acceder a **otros recursos**, como imágenes y recursos de diseño, almacenados en diferentes directorios dentro de la aplicación.
* Los permisos en Android y Linux se definen utilizando una **notación octal** que representa los permisos para el propietario, el grupo y otros usuarios.
* **Seguridad**:  
    
  Android implementa mecanismos de seguridad para proteger los datos del usuario y garantizar que las aplicaciones solo accedan a los recursos a los que tienen permiso.  
  Los permisos de almacenamiento son esenciales para que las aplicaciones accedan a recursos específicos del sistema de archivos.
* **La partición /system**es el hogar de todos los componentes de Android, según lo proporcionado por Google y/o el fabricante. El directorio y su contenido son propiedad de root:root, y todos tienen permisos de 0755 (rwxr-xr-x), pero el sistema de archivos está montado de forma de solo lectura.  
    
  Un montaje de solo lectura tiene sentido por dos razones:  
  **Estabilidad**: Debido a que el sistema de archivos está montado de solo lectura, prácticamente no hay posibilidad de que se corrompa, incluso si el dispositivo se apaga abruptamente. Esto reduce la posibilidad de un error que podría "brickear" el dispositivo al evitar que Android se inicie.  
    
  **Seguridad**: Un montaje de solo lectura es otra capa de defensa para proteger los componentes del sistema Android de ser manipulados. En la práctica, sin embargo, es trivial volver a montar la partición como de escritura.
* La partición /system es (en su mayor parte) la misma en la mayoría de los dispositivos.

# partición /data:

* **¿Qué es la partición /data?** La partición /data es donde se almacenan todos los datos personales del usuario en un dispositivo Android. Esto incluye aplicaciones, configuraciones, archivos multimedia y otros datos relevantes.

## Ventajas de tener una partición /data separada:

* + **Independencia del sistema operativo:** La partición /data está desacoplada de la versión subyacente del sistema operativo Android. Esto significa que las actualizaciones del sistema o la recuperación pueden borrar y reescribir la partición /system sin afectar los datos del usuario en /data. Durante un “restablecimiento de fábrica”, solo se formatea /data, lo que permite una rápida eliminación de los datos personales.
  + **Encriptación opcional:** /data puede estar encriptado si el usuario lo requiere. Aunque la encriptación agrega latencia (debido a la desencriptación y encriptación), es útil para proteger datos sensibles. Dado que /system no contiene información personal, no es necesario encriptarlo, evitando así esta latencia.
  + **Modelo de seguridad:** Los permisos en /data y sus subdirectorios permiten que las aplicaciones realicen operaciones en ellos, pero no accedan a su contenido. Por ejemplo, las aplicaciones pueden cambiar de directorio y operar en archivos dentro de /data, pero no pueden acceder al contenido de otros directorios en /data. Las etiquetas SELinux refuerzan este enfoque en versiones recientes de Android, asegurando que solo los usuarios con acceso root puedan acceder a los datos de todas las aplicaciones en /data.
* **PARTICION /CACHE:**

La partición /cache está definida por Android para su uso durante las actualizaciones del sistema. Las actualizaciones del sistema se descargan a esta ubicación, y el administrador de arranque es consciente de esta partición, especialmente cuando se arranca en modo de recuperación/actualización. De lo contrario, la partición normalmente está vacía.

* **EXT4 (Extended File System):**
  + **Capacidad**: Puede manejar grandes volúmenes de datos y archivos.
  + **Optimización**: Está diseñado para trabajar con discos duros de gran capacidad y alta velocidad.
  + **Características avanzadas**: Incluye comprobaciones de integridad y recuperación automática de errores.
* **VFAT:**
  + **Utilización**: Se emplea en algunos dispositivos Android para formatear la partición de almacenamiento externo (como tarjetas SD).
  + **Mejora de FAT32**: Es una versión mejorada del sistema de archivos FAT32 y permite almacenar archivos de más de 4 GB.
  + **Montaje**: La partición de almacenamiento externo formateada en VFAT se monta en el directorio /mnt/sdcard o /sdcard, según la versión de Android y el fabricante del dispositivo.
* **YAFFS2 (Yet Another File System 2):**
  + **Diseño específico**: Está diseñado para trabajar con dispositivos de memoria flash.
  + **Características únicas**: Aborda el desgaste de las celdas de memoria mediante un algoritmo especial.
  + **Estructura de árbol**: Organiza los datos en bloques y nodos para una lectura y escritura más rápida y una mejor gestión de la memoria.

particiones en Android y la estructura del sistema de archivos:

* **Partición del sistema:**
  + La partición “system” contiene los programas y configuraciones proporcionados inicialmente por el fabricante u operador móvil.
  + Su contenido se monta en modo de lectura exclusiva, lo que significa que los usuarios pueden leer los archivos pero no modificarlos.
  + El directorio “app” incluye aplicaciones de sistema como el lanzador de aplicaciones, contactos, calendario y reproductor de música.
  + El directorio “etc” contiene configuraciones estáticas del sistema y sonidos de notificaciones y tonos de llamada predeterminados.
  + Los directorios “bin” y “xbin” contienen archivos binarios y scripts utilizados por el sistema.
* **Partición de datos:**
  + La partición “data” almacena programas instalados por el usuario, sus datos y los datos de aplicaciones de sistema.
  + El sistema puede leer y escribir archivos en esta partición, lo que permite la modificación dinámica de su contenido.



* **Directorio “app”:**
  + Este directorio contiene las aplicaciones instaladas por el usuario, ya sea desde la Play Store de Google o desde otras fuentes alternativas.
  + Antiguamente, existía un directorio llamado “app-private” que incluía algunas aplicaciones ocultas, pero este enfoque está en desuso debido al nuevo sistema de comprobación de licencias en línea.
* **Directorio “data”:**
  + En esta ubicación, Android almacena los archivos asociados a cada aplicación, ya sean aplicaciones del sistema o instaladas por el usuario.
  + Para cada aplicación, Android crea un directorio específico y asigna permisos de acceso para que solo la aplicación correspondiente pueda acceder a sus propios archivos.
* **Partición del kernel**:
* Montada generalmente desde la carpeta “sys”, esta partición contiene el kernel del sistema, junto con los módulos y bibliotecas asociados.

También almacena datos y archivos específicos de cada dispositivo, como información sobre la CPU (por ejemplo, archivos relacionados con las velocidades máxima y mínima de la CPU en la carpeta “/sys/devices/system/cpu”).

* **Partición “efs” (sistema de archivos extendido)**:
* Presente en los teléfonos Samsung, almacena información relevante sobre el hardware de comunicaciones del dispositivo.

Esto incluye direcciones MAC para Bluetooth y redes inalámbricas, así como el IMEI del teléfono.

Se recomienda encarecidamente hacer una copia de seguridad de la carpeta “efs” después de rootear el dispositivo, ya que su corrupción puede causar problemas graves.

* **Partición “sd-ext”**:

Algunos dispositivos y programas utilizan esta partición para permitir la instalación de aplicaciones en la tarjeta de memoria.

Las últimas versiones del sistema operativo Android ya admiten esta funcionalidad de forma nativa.

* **Comandos del sistema**:

El sistema proporciona varios comandos para verificar el estado y el tamaño de las particiones.

Estos comandos se pueden ejecutar a través de un shell utilizando una aplicación de terminal en el propio dispositivo o mediante el comando “adb” (Android Debug Bridge).

“adb” permite interactuar con el sistema Android, como instalar o desinstalar programas, copiar o eliminar archivos y obtener registros del estado del sistema.

* El comando “df” muestra la lista de particiones y su espacio total, libre y ocupado.
* El comando “mount” permite montar y desmontar particiones, pero su uso debe ser cauteloso y solo por usuarios con conocimientos precisos.
* **Almacenamiento adoptable (Adoptable Storage):**
  + A partir de Android 6.0 (Marshmallow), Google introdujo la función de almacenamiento adoptable.

Permite que los medios de almacenamiento externo (como tarjetas SD) se formateen y cifren para funcionar como almacenamiento interno.

Los medios adoptados están vinculados al dispositivo Android específico y pueden almacenar aplicaciones y datos privados.

Los usuarios pueden elegir adoptar o usar los medios sin formatearlos para almacenamiento de archivos simple.

Al adoptar, Android ofrece migrar contenidos desde el almacenamiento principal (generalmente montado en “/sdcard”) a los medios recién adoptados, liberando espacio en el almacenamiento interno.

* **SELinux (Security-Enhanced Linux):**
  + Android utiliza SELinux para aplicar control de acceso obligatorio (MAC) a todos los procesos, incluso aquellos con privilegios de superusuario.

SELinux protege y confina los servicios del sistema, controla el acceso a datos de aplicaciones y reduce los efectos del software malicioso.

Opera según el principio de denegación predeterminada: todo se deniega a menos que se permita explícitamente.

* + Puede funcionar en dos modos globales: permisivo (registra denegaciones, pero no las aplica) y cumplimiento (registra y aplica denegaciones).
  + **Modo permisivo**, en el que las denegaciones de permisos se registran, pero no se aplican.
  + **Modo de cumplimiento**, en el que las denegaciones de permisos se registran y se aplican.

# ****Memorias:****

Android contienen un total de seis particiones de memoria y todas ellas son necesarias en el espacio de almacenamiento en el teléfono.

## Data

Es la partición que contiene todos los datos del usuario. Las aplicaciones y juegos que se instalen, o los archivos que se hayan descargado, se encuentran aquí.

Cuando se restaura el teléfono de fábrica: Se eliminan todos los datos que hay en esta partición en Android.

## Boot

Es la partición que arranca el teléfono Android. Es en la misma donde se encuentra el kernel y la ramdisk (una memoria virtual que ejecuta algunos procesos de encendido en el teléfono).

## Cache

La caché es algo de lo que la mayoría de usuarios en Android ha escuchado hablar en alguna ocasión ya que hay muchos debates sobre la efectividad de borrar la caché del teléfono. Se puede referir a ella como la partición donde se almacenan los datos y archivos a los que se acceden a menudo, tanto por parte del usuario como del sistema operativo. La caché se llena de archivos que se usan a menudo para que no se tengan que descargar cada vez que se acceda a ellos; lo que ahorra tiempo y recursos en el teléfono. Se pueden acumular muchos archivos en esta partición. Por eso, en algunas ocasiones se recomienda borrarla cada cierto tiempo.

## System

Esta partición es aquella que contiene los archivos del sistema operativo. Desde la interfaz de Android a las aplicaciones del sistema, aquellas que vienen instaladas por defecto.

## Recovery

Un requisito obligatorio en todo sistema operativo es tener una partición de recuperación. En caso de que haya problemas con el arranque normal, boot en este caso, se tiene acceso a un menú de recuperación desde el que recuperar el acceso al sistema operativo con normalidad.

## Misc

En esta memoria se guardan los ajustes relacionados con el operador y de aspectos relacionados con el hardware del teléfono. Por eso, es importante que no haya problemas en dicha partición porque pueden provocar problemas en el teléfono.

## Retención de Memoria:

Cuando hay suficiente espacio, el sistema retiene memoria incluso después de cerrar aplicaciones o ponerlas en segundo plano. Esto permite que las aplicaciones se inicien más rápidamente cuando el usuario las vuelve a abrir, ya que los componentes comunes ya están cargados en la memoria.

## Acceso a la Memoria:

El acceso a la memoria por parte del software se logra generalmente a través de los kernels de Linux y los controladores.

## Proceso Zygote:

Zygote es un proceso especial en Android que se encarga de pre-iniciar y cargar componentes comunes de las aplicaciones en la memoria durante el inicio del sistema operativo, esto ayuda a acelerar el tiempo de inicio de las aplicaciones, ya que no necesitan cargar estos componentes comunes desde cero cada vez que se lanzan.

# ****División de Memoria****

La memoria se divide en Virtual y Física, gestionada por la CPU a través de la Gestión de Memoria.

## MEMORIA VIRTUAL:

La memoria virtual se puede clasificar en al menos 4 tipos a continuación detallados.

## Memoria nombrada (mmapped):

Esta memoria proviene de archivos en el almacenamiento, como disco, flash o sistema de archivos de red. Estas páginas se cargan desde un archivo contiguo en la memoria virtual del proceso. El nombre del archivo se utiliza como nombre para la memoria, y por eso se le llama "nombrada".

## Memoria anónima:

Esta categoría de memoria se genera específicamente para satisfacer las demandas instantáneas del proceso en ejecución. Implica la asignación de memoria para propósitos temporales, como la administración de la pila o el montículo durante la creación de estructuras de datos dinámicas o la ejecución de funciones específicas. A diferencia de otros tipos de memoria, no está respaldada por un archivo, lo que significa que no tiene una referencia de nombre asociada.

## Memoria limpia (Clean) vs. Memoria sucia (Dirty):

Una vez que una página de memoria se carga en la RAM, puede permanecer sin cambios (limpia) o puede ser modificada por el proceso (sucia).

## Memoria activa vs. Memoria inactiva:

Las páginas de memoria virtual tienen una "edad" cuando se mapean en la RAM. La memoria activa se refiere a las páginas que se han accedido recientemente, mientras que las memorias inactivas son aquellas que no se han utilizado durante un cierto período de tiempo. Esto es importante para el proceso de liberación y escritura de páginas.