Rooteo en Android

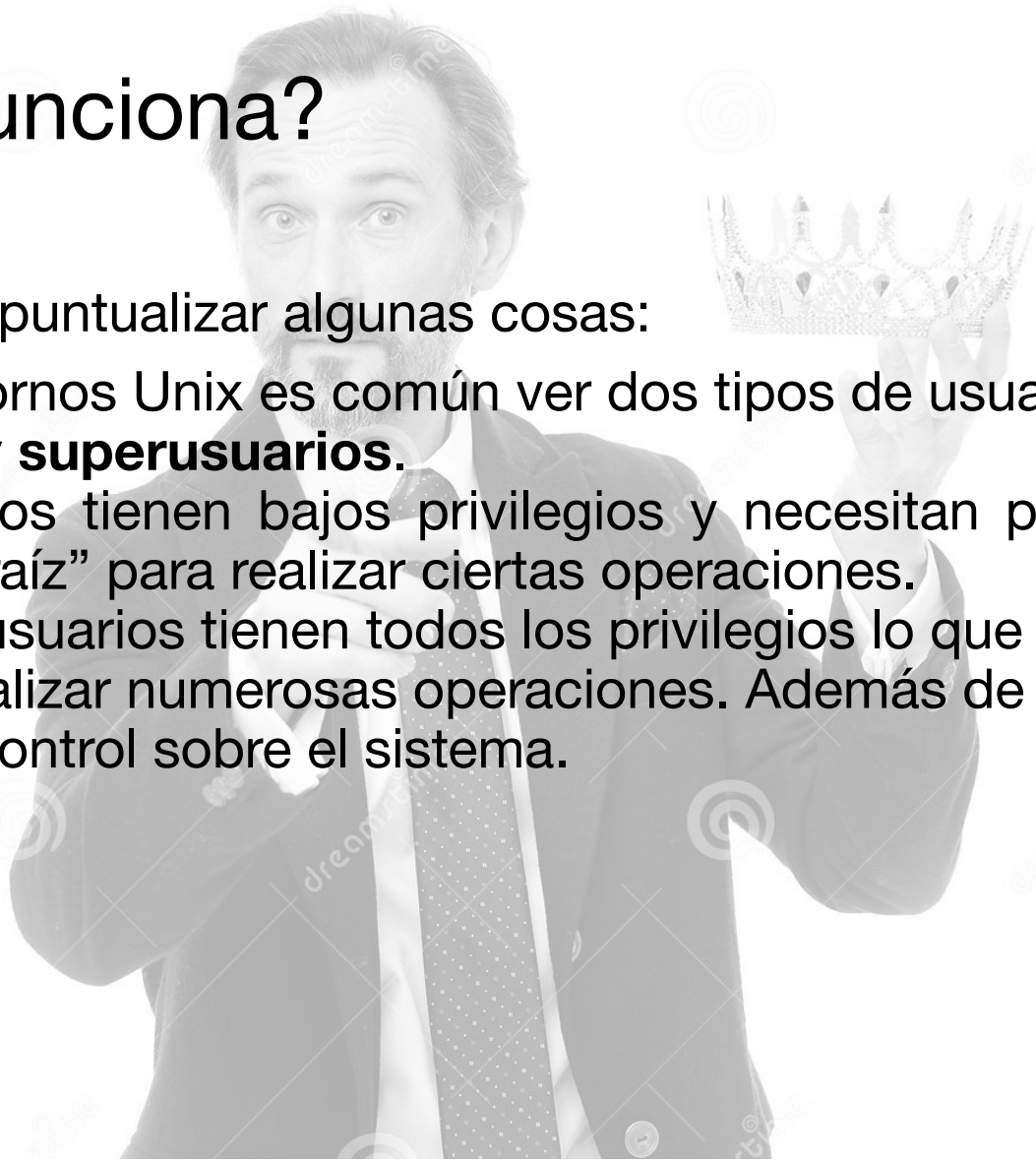
Rivera García Mauricio
Ruiz Flores Laura Andrea

A rasgos generales, ¿qué es?

- Es una técnica que permite a un usuario de Android tener un control privilegiado del equipo.
- De forma predeterminada, los dispositivos Android no tienen estos privilegios activados.

Rivera García Mauricio
Ruiz Flores Laura Andrea

¿Cómo funciona?



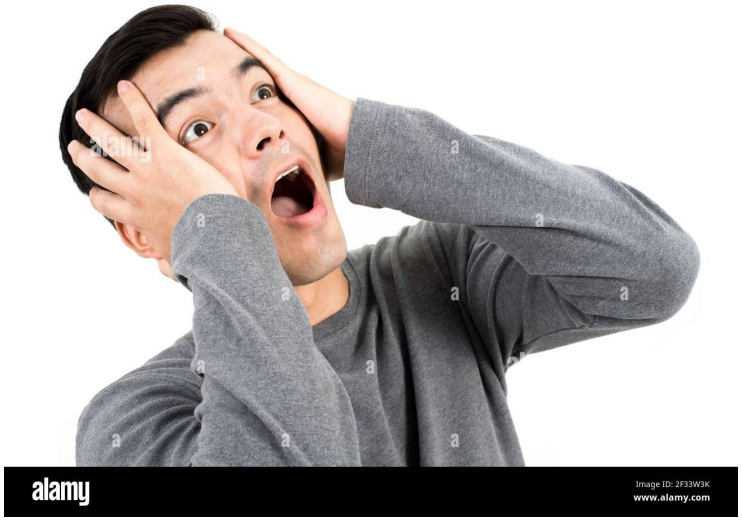
Es importante puntualizar algunas cosas:

- En los entornos Unix es común ver dos tipos de usuario: **usuarios** y **superusuarios**.
- Los usuarios tienen bajos privilegios y necesitan permisos desde la “raíz” para realizar ciertas operaciones.
- Los superusuarios tienen todos los privilegios lo que les permite realizar numerosas operaciones. Además de tener un fuerte control sobre el sistema.

Rivera García Mauricio
Ruiz Flores Laura Andrea

- Android es, de hecho, un sistema operativo basado en el kernel de Linux.
- Muchas de las cosas que se ven en los sistemas Linux tradicionales se pueden ver también en los sistemas Android, como lo es esta separación de privilegios.
- Pero en Android no se puede acceder a este modo de inicio.

Es decir que, técnicamente, no eres el dueño de tu dispositivo



Rivera García Mauricio
Ruiz Flores Laura Andrea

Entonces, el rooteo trata de obtener total control sobre tu dispositivo Android.

Rivera García Mauricio
Ruiz Flores Laura Andrea

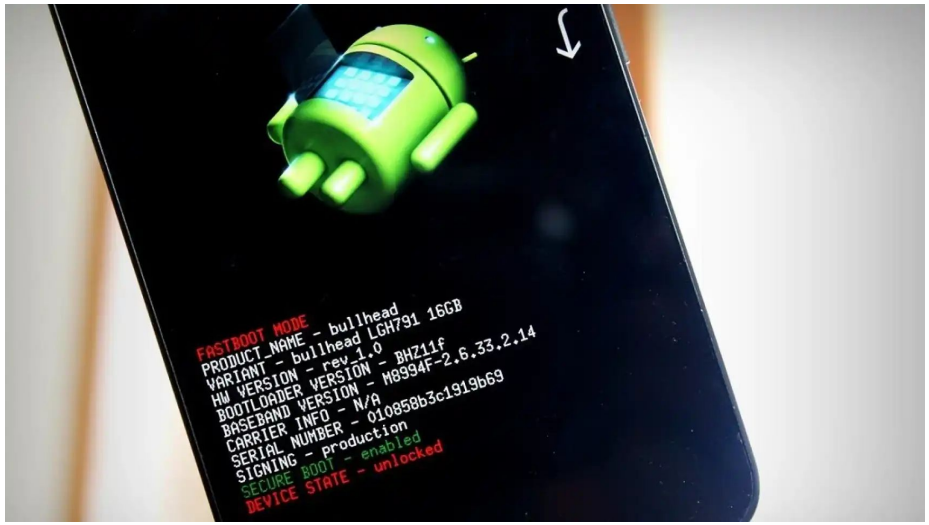
Pero…

Es importante mencionar los mecanismos de protección que tiene Android sobre algunos métodos de rooteo.

Rivera García Mauricio
Ruiz Flores Laura Andrea

Bootloader

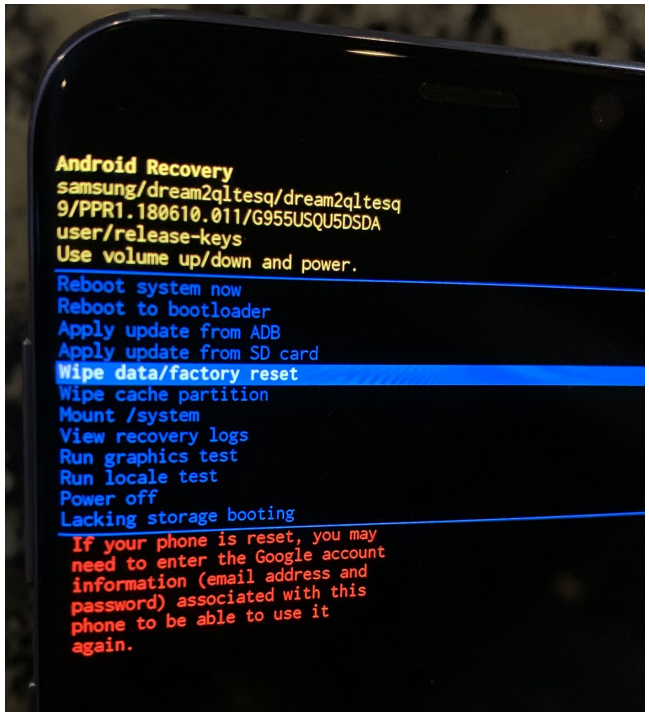
- Es el pedazo de código que se encarga de arrancar el sistema operativo
- Con cierta configuración de teclas se podría acceder ahí y llegar al Recovery System.
- Puede estar **bloqueado** o **desbloqueado** en dispositivos Android



Rivera García Mauricio
Ruiz Flores Laura Andrea

Recovery System

- En este punto es donde básicamente se puede acceder a una versión alternativa del sistema operativo, por lo general **aprobada por el fabricante.** Esto depende de si el bootloader está bloqueado o no.



Cuando está bloqueado, sólo permite arrancar las versiones del sistema que el fabricante instale, asegurándose que el móvil no pueda ser manipulado.

Si está bloqueado, entonces no se puede rootear el dispositivo mediante el Bootloader :(

Rivera García Mauricio
Ruiz Flores Laura Andrea

adb shell

- Intérprete de comandos de Android
- Se accede a él por una PC o una Mac.
- Se pueden ejecutar ***ciertos*** comandos, **dependiendo de *ciertos* parámetros**.

Rivera García Mauricio
Ruiz Flores Laura Andrea

- **¿dependiendo de ciertos parámetros?**

`ro.secure`

```
root@Obi_S454:/system/etc/permissions # getprop ro.secure
0
```

Si **`ro.secure=0`** *entonces se puede ejecutar cualquier comando en ADB con privilegios de superusuario*

```
walleye:/ # getprop ro.secure
1
```

Si **`ro.secure=1`** *los comandos se ejecutarán como un usuario sin privilegios de superusuario*

Rivera García Mauricio
Ruiz Flores Laura Andrea

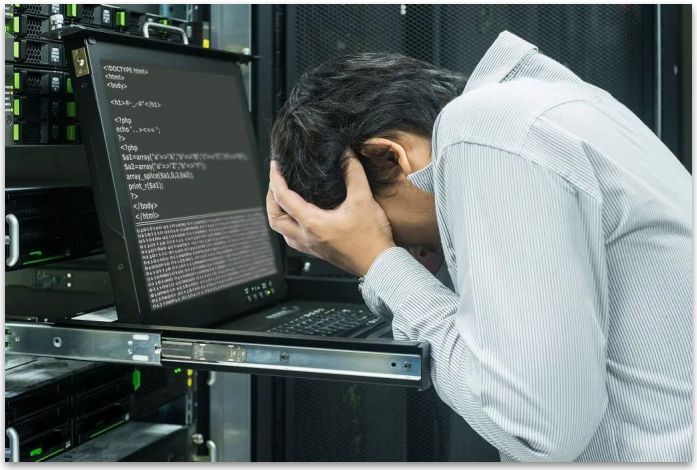
Pero no es posible pasar el ro.secure de uno a cero

Si está en 1, entonces no se puede hackear el dispositivo mediante ADB :(

Rivera García Mauricio
Ruiz Flores Laura Andrea

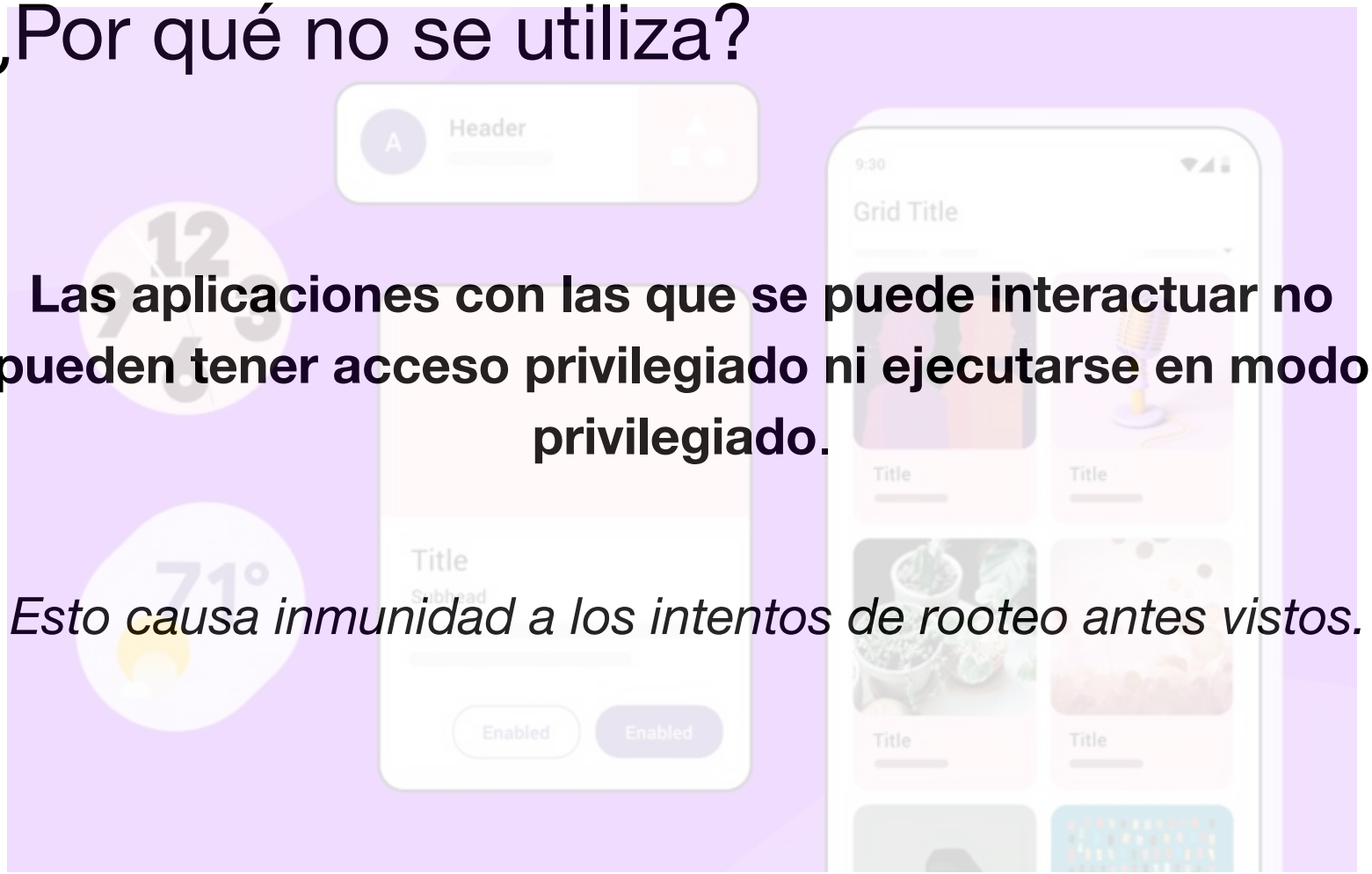
su y **sudo**

- En Linux, estos comandos hacen llamadas al sistema que permiten acceder a privilegios de superusuario.
- Algunas aplicaciones usan esto para tener ciertos permisos desde la raíz.
- Para Android, en las versiones de los móviles que salen a la venta nunca se utiliza este comando.



Rivera García Mauricio
Ruiz Flores Laura Andrea

¿Por qué no se utiliza?



Las aplicaciones con las que se puede interactuar no pueden tener acceso privilegiado ni ejecutarse en modo privilegiado.

Esto causa inmunidad a los intentos de rooteo antes vistos.

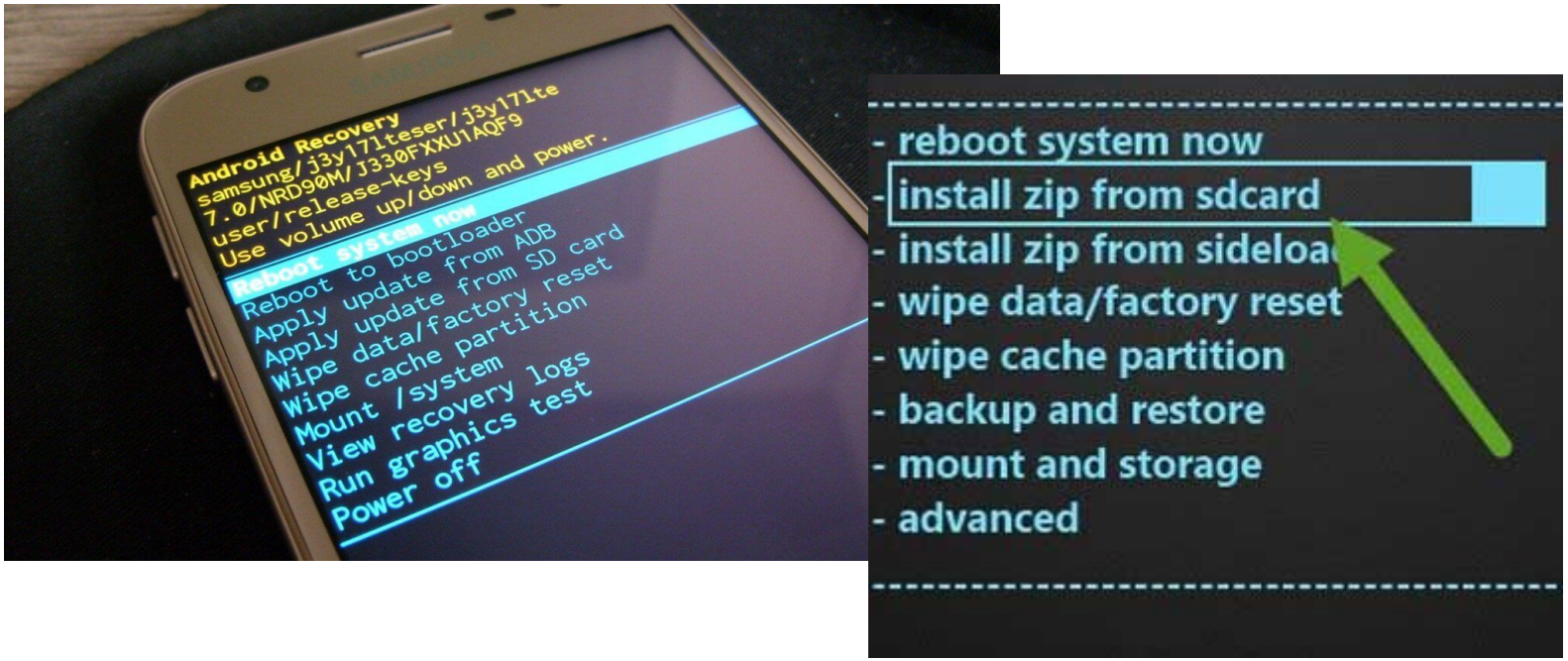
Rivera García Mauricio
Ruiz Flores Laura Andrea

Con todas estas restricciones, ¿cómo se realiza entonces el rooteo?

Rivera García Mauricio
Ruiz Flores Laura Andrea

El caso “fácil”

- El bootloader está desbloqueado.
  - Se realiza cierta combinación de teclas, se obtiene la configuración deseada y se ejecuta desde el bootloader y el sistema de recuperación.



Rivera García Mauricio
Ruiz Flores Laura Andrea

El caso ‘aún más fácil’

- **`ro.secure=0`**
  - Se requiere conectar el móvil a la computadora, ejecutar ADB, montar **`/system`** en modo lectura-escritura e instalar **`su`**.

```
C:\Users\athakur\adt-bundle-windows-x86_64-20140321\adt-bundle-windows-x86_64-20
140321\sdk\platform-tools>adb shell
shell@android:/ $ su
su
root@android:/ # mount -o remount,rw /system
mount -o remount,rw /system
root@android:/ # ls -la /system/bin/sh
ls -la /system/bin/sh
lrwxr-xr-x root     shell                2012-10-29 14:13 sh -> mksh
root@android:/ # chmod 4755 /system/bin/sh
chmod 4755 /system/bin/sh
root@android:/ # ls -la /system/bin/mksh
ls -la /system/bin/mksh
-rwsr-xr-x root     shell      152012 2012-10-29 14:13 mksh
root@android:/ #
```


Rivera García Mauricio
Ruiz Flores Laura Andrea

El caso real

- El bootloader está bloqueado.
- **`ro.secure=1`**

Se hace uso de los procesos que corran en la raíz obligatoriamente

Rivera García Mauricio
Ruiz Flores Laura Andrea

```
C:\Windows\system32>adb shell ps
USER      PID   PPID  VSIZE  RSS     WCHAN    PC         NAME
root      1     0     760    392   ffffffff 00000000 S /init
root      2     0     0      0     ffffffff 00000000 S kthreadd
root      3     2     0      0     ffffffff 00000000 S ksoftirqd/0
root      5     2     0      0     ffffffff 00000000 S kworker/0:0H
root      7     2     0      0     ffffffff 00000000 S migration/0
root      8     2     0      0     ffffffff 00000000 S rcu_preempt
root      9     2     0      0     ffffffff 00000000 S rcu_bh
root      10    2     0      0     ffffffff 00000000 S rcu_sched
root      11    2     0      0     ffffffff 00000000 R migration/1
root      12    2     0      0     ffffffff 00000000 R ksoftirqd/1
root      14    2     0      0     ffffffff 00000000 S kworker/1:0H
root      15    2     0      0     ffffffff 00000000 R migration/2
root      16    2     0      0     ffffffff 00000000 R ksoftirqd/2
root      18    2     0      0     ffffffff 00000000 S kworker/2:0H
root      19    2     0      0     ffffffff 00000000 R migration/3
root      20    2     0      0     ffffffff 00000000 R ksoftirqd/3
root      22    2     0      0     ffffffff 00000000 S kworker/3:0H
root      23    2     0      0     ffffffff 00000000 S khelper
root      24    2     0      0     ffffffff 00000000 S suspend_sys_syn
root      25    2     0      0     ffffffff 00000000 S suspend
root      26    2     0      0     ffffffff 00000000 S writeback
root      27    2     0      0     ffffffff 00000000 S bioset
root      28    2     0      0     ffffffff 00000000 S kblockd
root      29    2     0      0     ffffffff 00000000 S khubd
root      48    2     0      0     ffffffff 00000000 S irq/322-charger
```


Rivera García Mauricio
Ruiz Flores Laura Andrea

- Hay muchos procesos que tienen que ser ejecutados desde root
- Entonces, la forma de rootear aprovecha estos procesos.

Básicamente, hay que modificar estos procesos para colar un código arbitrario entre ellos.

Código arbitrario que, casualmente, podría habilitar los permisos de superusuario en el sistema.

Y es así como nacen los “exploits” o “jailbreaks”

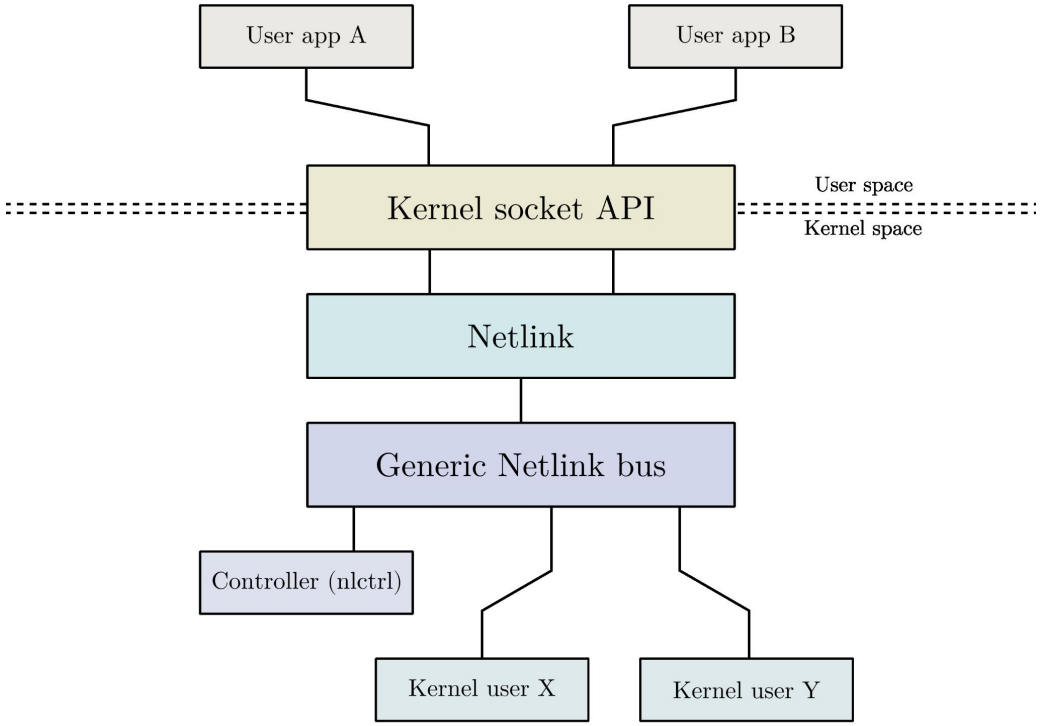
Rivera García Mauricio
Ruiz Flores Laura Andrea

Algunos exploits…

- Exploid / GingerBreak
- RageAgainstTheCage / ZimperLich
- KillingInTheNameOf

Rivera García Mauricio
Ruiz Flores Laura Andrea

Exploid y GingerBreak



La interfaz de Netlink cuenta con una “falta de verificación de fuentes”.

Exploid aprovecha init
GingerBreak aprovecha DirectVolume

Rivera García Mauricio
Ruiz Flores Laura Andrea

KillingInTheNameOf

- Modifica **`ro.secure`** mediante el sistema de memoria compartida de Android.

RageAgainstTheCage / ZimperLich

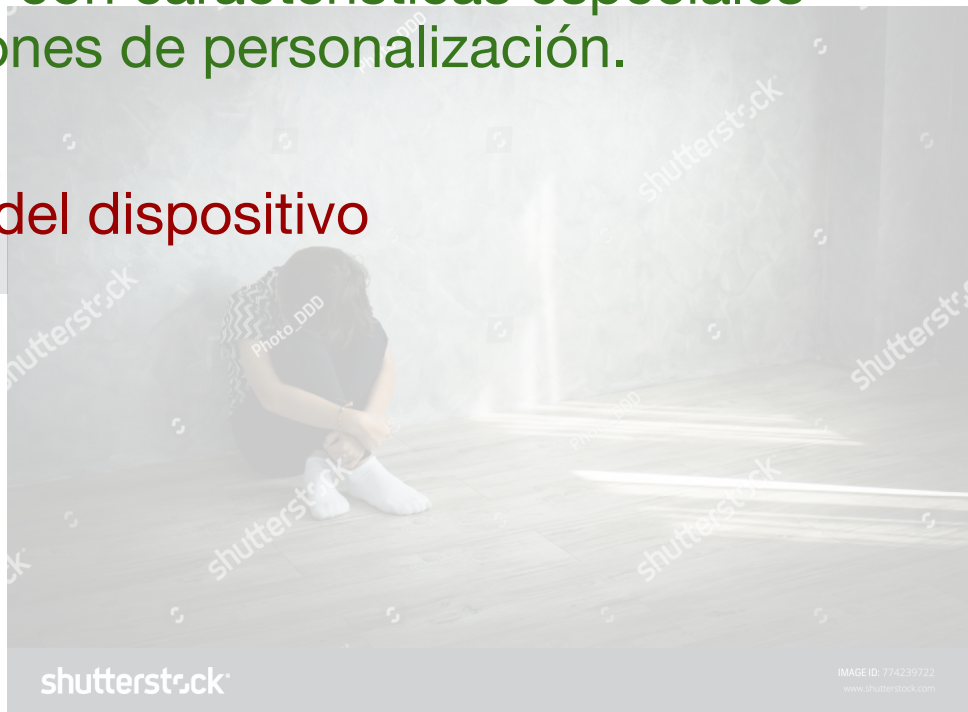
- Infla la cantidad de procesos para generar un error en **`setuid(id)`** y acceder a privilegios con ayuda de ADB

Rivera García Mauricio
Ruiz Flores Laura Andrea

¿Para qué rootear? ¿Qué riesgos tiene?

- Mayor control del dispositivo
- Instalación de aplicaciones con características especiales
- Más características y opciones de personalización.

- Compromete la seguridad del dispositivo
- Bricking
- Puede perderse la garantía



Rivera García Mauricio
Ruiz Flores Laura Andrea

¿Por qué los fabricantes se protegen tanto de los rooteos?

- Seguridad del dispositivo
- Contratos y cuestiones de dinero

Nuevamente comprometiendo la garantía, si el rooteo produce algún problema con el dispositivo esta no será aplicada.

Rivera García Mauricio
Ruiz Flores Laura Andrea

Y ya

Rooteo en Android

Rivera García Mauricio
Ruiz Flores Laura Andrea

Referencias

Ji, Chuan. (2011) *How Rooting Works: A Technical Explanation of the Android Rooting Process*

Enlace:
https://jichu4n.com/posts/how-rooting-works-a-technical-explanation-of-the-android-rooting-process/

Srinivasa Rao Kotipalli y Mohammed A. Imran (2016) *Hacking Android.*

Enlace:

https://www.google.com.mx/books/edition/Hacking_Android/j86qDQAAQBAJ

Documentación de Android Debug Bridge (adb)

Enlace:

https://developer.android.com/studio/command-line/adb

Jon Oberheide y Zach Lanier (2011) *Don't Root Robots!*

Enlace:

https://jon.oberheide.org/files/bsides11-dontrootrobots.pdf

Rashid-Feroze. (2023). *Linux Privilege Escalation Guide.*

Enlace:

https://payatu.com/blog/a-guide-to-linux-privilege-escalation/

Luque S. (2016). *Los fabricantes, el root y la garantía, un mundo de confusión*

Enlace:

https://www.xatakandroid.com/sistema-operativo/los-fabricantes-el-root-y-la-garantia-un-mundo-de-confusion

Long Nguyen-Vu, Ngoc-Tu Chau, Seongeun Kang y Souhwan Jung. (2017) Android Rooting: An Arms Race between Evasion and Detection.

Enlace:

https://www.hindawi.com/journals/scn/2017/4121765/