



瞬间爆炸

Exploit? Explode!

申迪 (retme)

我是谁？

自我介绍

- ❖ 360的安全研究员
- ❖ 从事过Windows内核研究，Anti Rootkit
- ❖ 现在主攻安卓
 - ❖ 漏洞挖掘与利用：内核& framework & Apps
 - ❖ 攻防技术研究
 - ❖ 逆向工程
- ❖ 喜欢看球赛、玩主机、学日语

The easy way

- ❖ 集成了各种公开/自主挖掘的漏洞利用
- ❖ 支持20000+的机型
- ❖ 对各种机型适配，有时候需要付出很多写漏洞利用额外的辛苦
- ❖ 说了这么多，其实我没参与开发...



所以不要误会...

- ❖ 我并没有参与360一键root产品的开发
- ❖ 本议题中的观点仅代表我个人的看法
- ❖ 安卓内核漏洞的挖掘和利用是我的一个方向，也是兴趣




7.16 Android平台Bootkit高级攻击技术

Slides下载: <http://t.cn/RPUaUPm>

关于FakeID签名漏洞的利用~ <http://t.cn/RP6x39l>


8月4日 17:05 来自微博 weibo.com

阅读(1.9万) 推广 |  (6) | 转发(33) | 收藏 | 评论(4)

利用fakeid注入浏览器执行代码~成功~✌️



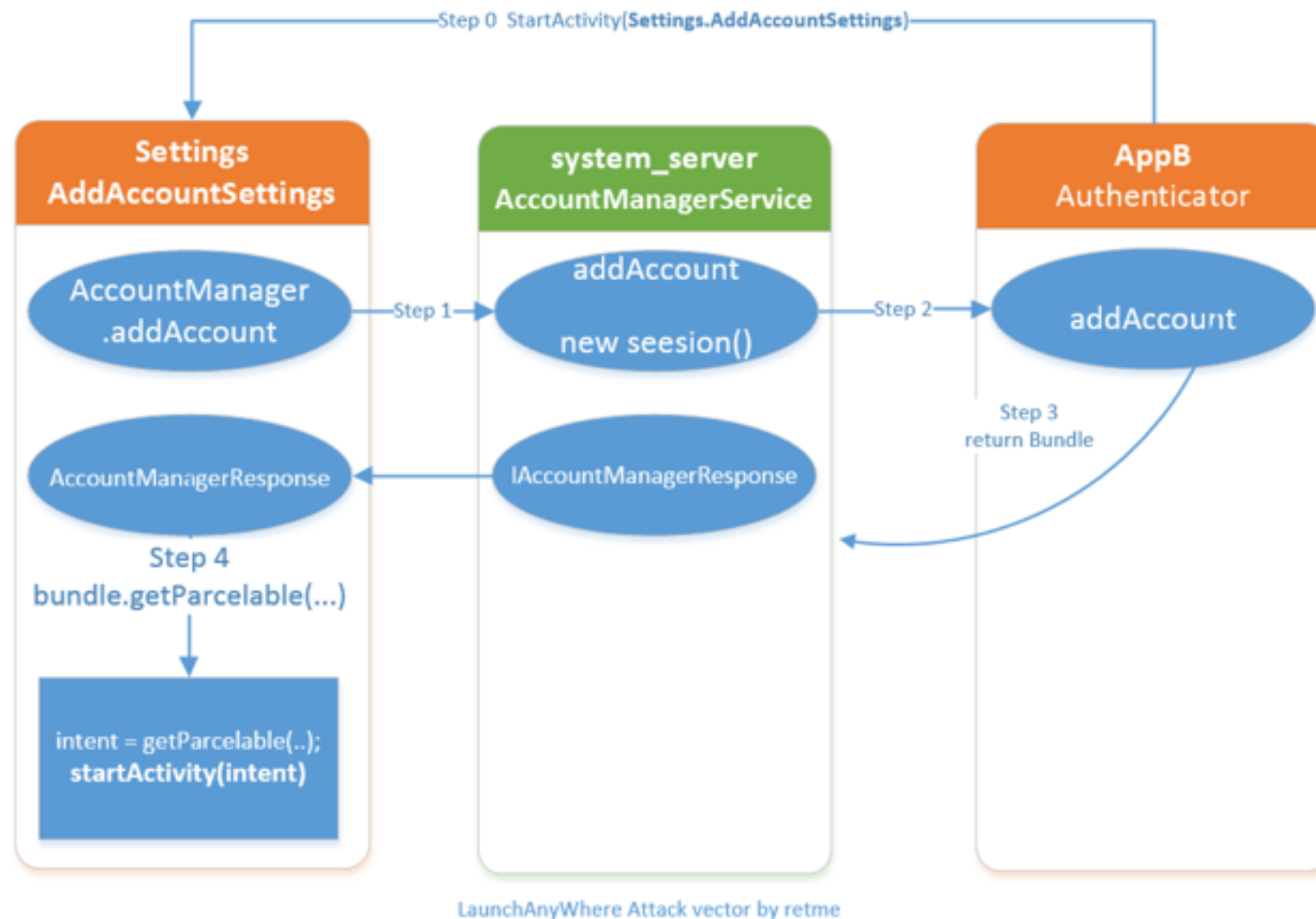
8月1日 20:22 来自微博 weibo.com

阅读(1538) 推广 |  (1) | 转发 | 收藏 | 评论(2)

8.4 分享FakeID签名漏洞的细节和利用方法

详情: <http://t.cn/RP6x39l>

开源代码: <https://github.com/retme7>



8.20 launchAnyWhere

公开Google秘密修复的组建权限绕过漏洞

详情 <http://t.cn/RPEptX0>

源码: <http://t.cn/RPduhQi>



8.23 @北京GDG 和大家分享手机Root的话题

议程

- ❖ SU命令的背后
- ❖ ROOT手机的多种方法
- ❖ 利用内核漏洞
- ❖ AOSP/Samsung KNOX对抗手机Root
- ❖ 展望Android L

议程

- ❖ SU命令的背后
- ❖ ROOT手机的多种方法
- ❖ 利用内核漏洞
- ❖ AOSP/Samsung KNOX对抗手机Root
- ❖ 展望Android L

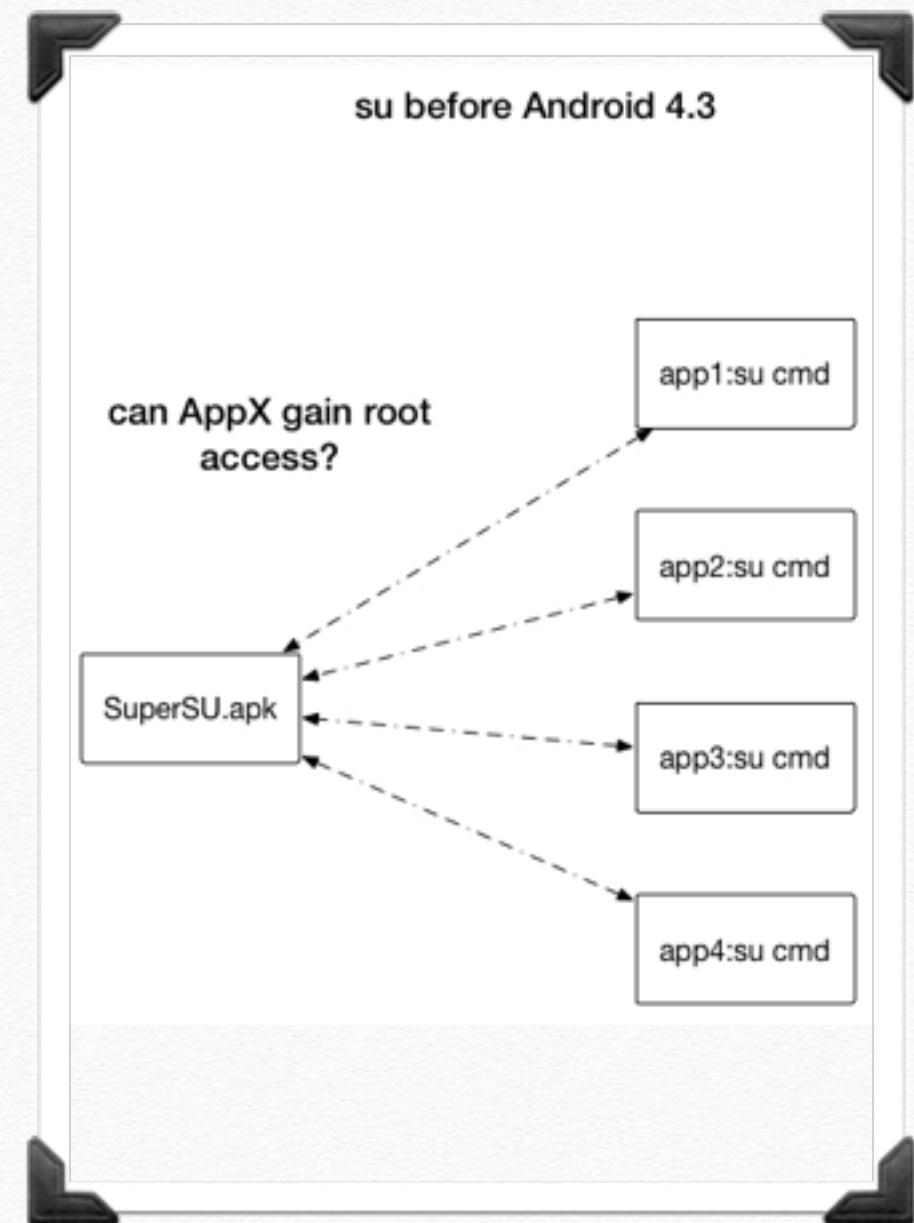
“获得特权，
然后向/system分区安装su程序”

SU命令的背后

- ❖ root过的手机需要授权管理来保护su不被恶意应用调用
- ❖ 4.3 以后system被mount为nosetuid，zygote孵化出的应用也不再能调用setuid程序
- ❖ 6.7.55的/system/bin/su已经不能独立完成提供root权限的任务
- ❖ 现在的su程序包含多个模块

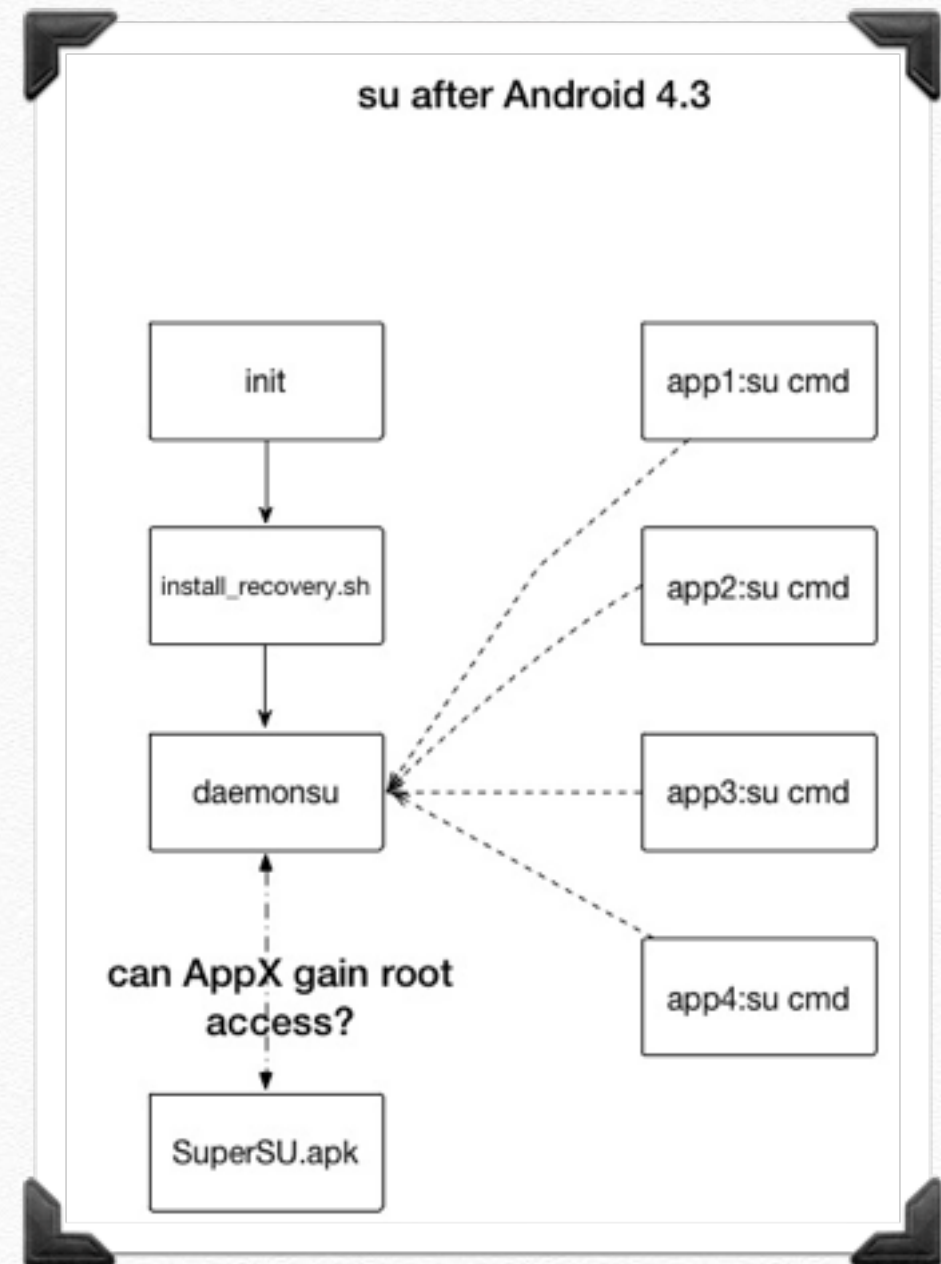
4.3以前的root程序

- ❖ su去询问授权管理是否向应用开放root权限
- ❖ 若放行，su执行root命令



4.3以后的root程序

- ❖ daemonsu必须作为服务在系统初始化早期启动
- ❖ daemonsu负责询问root授权许可（弹窗）
- ❖ daemonsu最终执行root命令



install-recovery

```
root@hammerhead:/ # cat /etc/install-recovery.sh
#!/system/bin/sh

# If you're implementing this in a custom kernel/firmware,
# I suggest you use a different script name, and add a service
# to launch it from init.rc

# Launches SuperSU in daemon mode only on Android 4.3+.
# Nothing will happen on 4.2.x or older, unless SELinux+Enforcing.
# If you want to force loading the daemon, use "--daemon" instead

/system/xbin/daemonsu --auto-daemon &

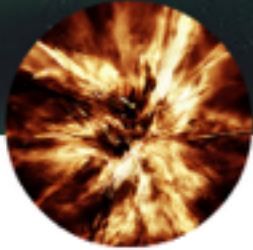
# Some apps like to run stuff from this script as well, that will
# obviously break root - in your code, just search this file
# for "install-recovery-2.sh", and if present, write there instead.


/system/etc/install-recovery-2.sh
root@hammerhead:/ #
```



daemonsu:0:x


```
127|root@hammerhead:/ # ps -Z | grep daemon
u:r:init:s0          camera    220      1      /system/bin/mm-qcamera-daemon
u:r:init:s0          system    221      1      /system/bin/time_daemon
u:r:init:s0          root      232      1      daemonsu:mount:master
u:r:init:s0          root      252      232    daemonsu:master
u:r:init:s0          root      1751     252    daemonsu:10114
u:r:init:s0          root      2050     252    daemonsu:0
u:r:init:s0          root      2338     252    daemonsu:10118
u:r:init:s0          root      2780     252    daemonsu:10087
u:r:init:s0          root      5365     252    daemonsu:10082
u:r:init:s0          root      7576     2050   daemonsu:0:7573
[1] + Broken pipe
```


Chainfire's Blog



Chainfire 

 关注对象

 **Chainfire**
公开分享 · 2013年7月28日

Some specifics about the 4.3 SuperSU (ramblings)

All this information is subject to change without notice. The current SuperSU at time of writing is v1.45.

Apart from a lot of text, this post also includes stuff a dev would want to know regarding daemon startup and mount namespaces.

Pre-cursor

I was surprised by the large number of negative (sometimes even flaming) responses from even techies that SuperSU didn't magically work perfectly out of the

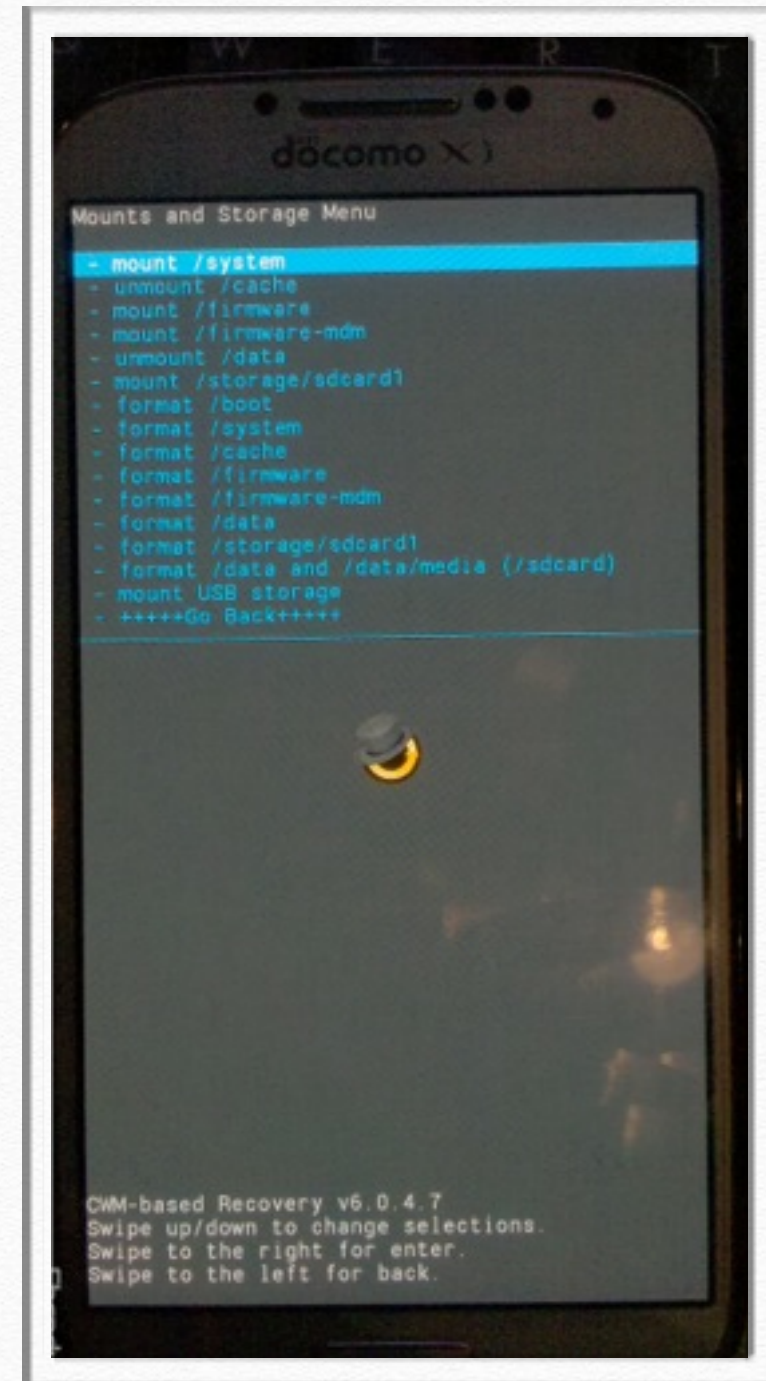
<https://plus.google.com/+Chainfire/posts/LYNiJCrWBGQ>

议程

- ❖ SU命令的背后
- ❖ **ROOT手机的多种方法**
- ❖ 利用内核漏洞
- ❖ AOSP/Samsung KNOX对抗手机Root
- ❖ 展望Android L

刷recovery

- ❖ recovery有一个独立Linux内核
- ❖ 没有如安卓沙箱一样的权限限制，可以随意操作系统分区，安装root
- ❖ 刷分区需要先解锁



利用漏洞获取ROOT

- ❖ 通过adb/APP执行漏洞利用程序获取root
- ❖ 随着版本更新，漏洞总是在不断修复，所以要不断的利用新漏洞
- ❖ 有些framework的漏洞可以利用，也有不少厂商固件中的漏洞，但因厂商而异
 - ❖ <http://theroot.ninja/PAE.pdf>
- ❖ 但现在更流行使用内核漏洞进行root

应用场景

- ❖ 为手机安装su
- ❖ 为不愿意root的用户提供root后才能使用的功能，无需安装su，尽量不影响保修
- ❖ APT攻击/高级入侵

议程

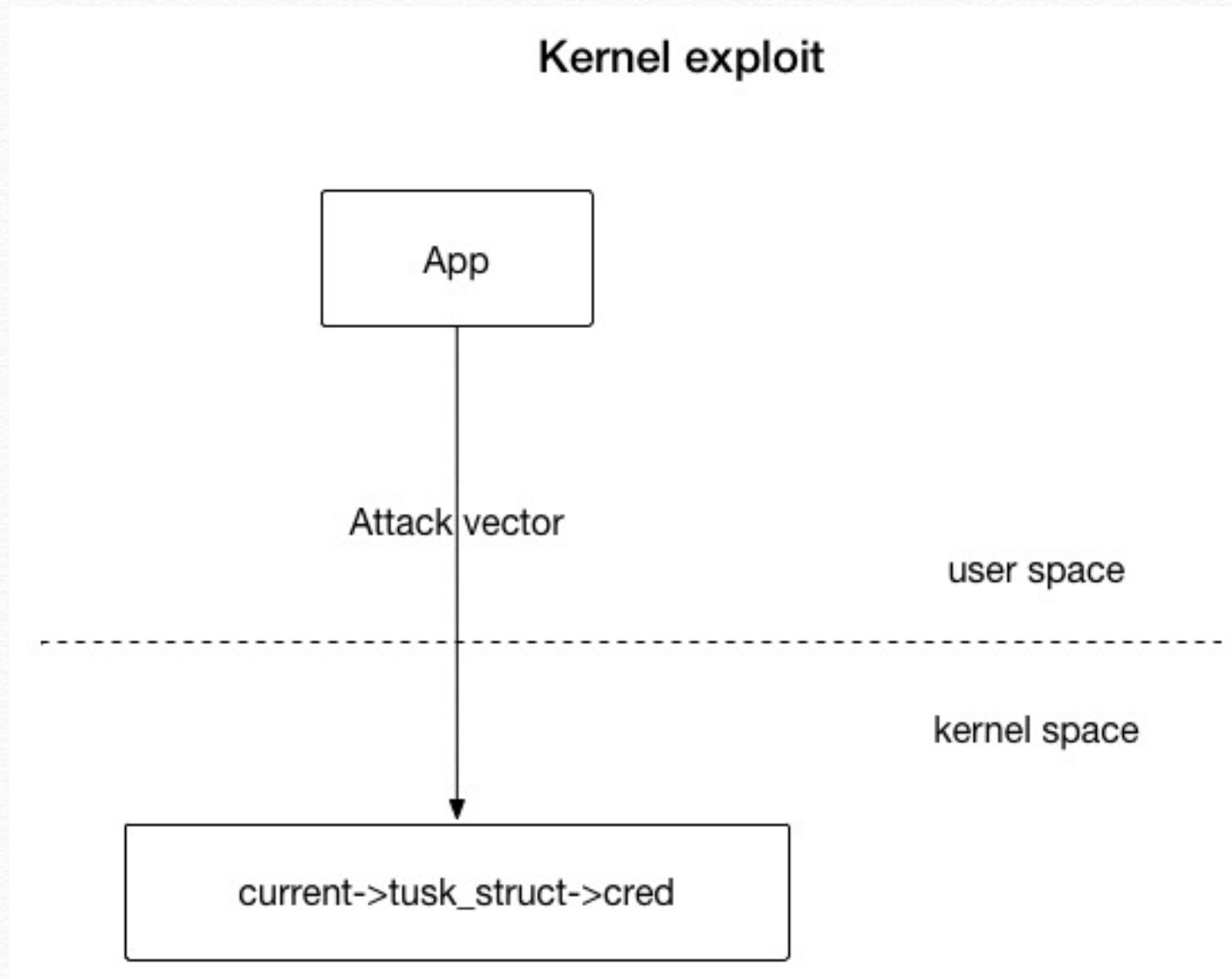
- ❖ SU命令的背后
- ❖ ROOT手机的多种方法
- ❖ 利用内核漏洞
- ❖ AOSP/Samsung KNOX对抗手机Root
- ❖ 展望Android L

利用内核漏洞

- ❖ 与桌面版Linux内核漏洞利用理论上一致
- ❖ 安卓内核与桌面版Linux内核在编译选项开关上有较大差异（binder/ashmem/各家SoC的代码）
- ❖ 高通/MTK/海思等架构引入的驱动代码是安卓内核独有的攻击面（不怕神一样的对手...）
- ❖ 内核漏洞频发，仅用公开漏洞已经可以将6月之前编译的手机内核全破

“无非是想使用最直接的方法将自己的uid改成0”

修改内核结构体



修改内核中的结构

```
615 my_cred->cap_permitted.cap[0] = -1;
616 my_cred->cap_permitted.cap[1] = -1;
617 my_cred->cap_effective.cap[0] = -1;
618 my_cred->cap_effective.cap[1] = -1;
619
620 my_real_cred->uid = 0;
621 my_real_cred->gid = 0;
622 my_real_cred->suid = 0;
623 my_real_cred->sgid = 0;
624 my_real_cred->egid = 0;
625 my_real_cred->euid = 0;
626 my_real_cred->fsgid = 0;
627 my_real_cred->fsuid = 0;
628 my_real_cred->securebits=0;
629 my_real_cred->cap_bset.cap[0] = -1;
630 my_real_cred->cap_bset.cap[1] = -1;
631 my_real_cred->cap_inheritable.cap[0] = -1;
632 my_real_cred->cap_inheritable.cap[1] = -1;
633 my_real_cred->cap_permitted.cap[0] = -1;
634 my_real_cred->cap_permitted.cap[1] = -1;
635 my_real_cred->cap_effective.cap[0] = -1;
636 my_real_cred->cap_effective.cap[1] = -1;
637
638
639 if(isSelinux){
640
641     tsec = my_cred->security;
642
643     if(tsec && tsec > 0xBFFFFFFF){
644         tsec->sid = 1;
645         tsec->exec_sid = 1;
646
647         ret = 15;
648     }
```


各有各的利用方法



公开资料

- ❖ <A Guide to Kernel Exploitation Attacking the Core>
- ❖ <Android Hacker's Handbook >
- ❖ https://github.com/fi01/android_run_root_shell

获取漏洞

- ❖ 关注CVE漏洞库
- ❖ 搜索commit log
- ❖ 审计/搜索源代码
- ❖ Fuzz testing

议程

- ❖ SU命令的背后
- ❖ ROOT手机的多种方法
- ❖ 利用内核漏洞
- ❖ **AOSP/Samsung KNOX对抗手机Root**
- ❖ 展望Android L

内核的攻击面

- ❖ 系统调用
- ❖ Linux标准驱动设备，比如tty
- ❖ 安卓引入的binder、ashmem等驱动设备
- ❖ 高通、MTK等厂商引入的驱动设备
- ❖ 网络协议驱动

AOSP对内核漏洞利用的防御

- ❖ 对某个特定漏洞利用的技巧进行针对性防御
- ❖ 缩小攻击者能接触到的攻击面
- ❖ 对root之后的程序进一步进行权限限制

AOSP对内核漏洞利用的防御

- ❖ 2.3禁止映射mmap_min_addr以下的内存，防止对内核空指针的引用问题的利用
- ❖ 4.1以后限制了通过dmesg获取内核日志信息;限制通过kallsymbols获取内核符号地址
- ❖ 4.3重新设计了照相机管理服务，一般应用无法在攻击相机相关的驱动
- ❖ 4.4 正式开启SE Linux，强化对于应用访问权限的控制
- ❖ L SE Linux 策略库全面升级，进一步进行限制；引入部分KNOX的特性

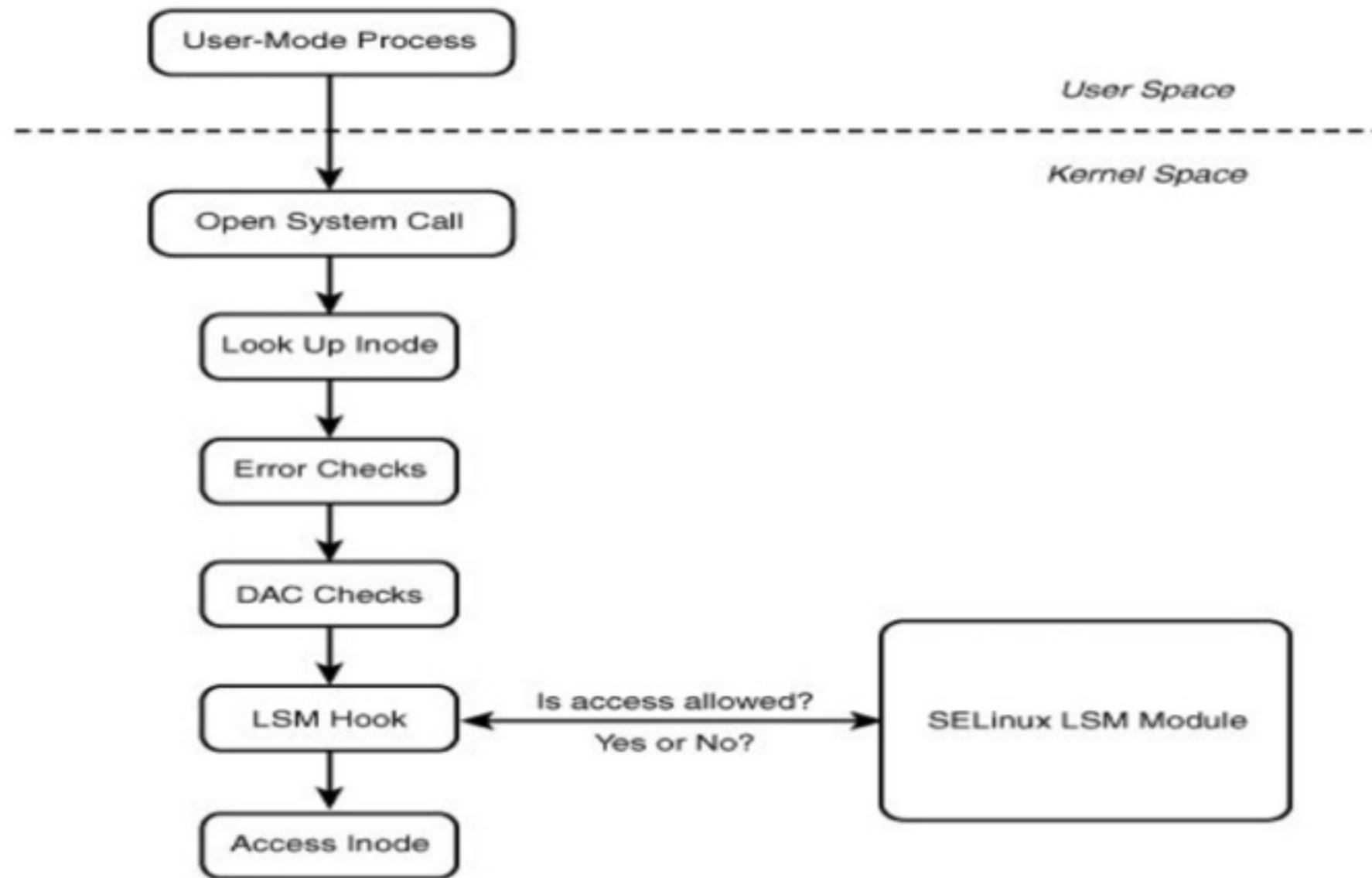
SE Linux开启以前

- ❖ Discretionary Access Control
- ❖ UID 决定 capability
- ❖ 文件属主决定文件访问权限
- ❖ 超级用户拥有绝大多数权限

SE Linux的增强

- ❖ Mandatory Access Control
- ❖ 规则描述细化到：“进程A 对资源 B 进行X操作”是否允许
- ❖ 规则库决定访问权限
- ❖ 最小特权原则，没有超级用户的概念。

Linux Security Module



Samsung Knox

- ❖ 更为严格的SELinux规则
- ❖ Restrict机制：
 - ❖ 只允许init调用setuid
 - ❖ 不允许/data分区的程序以root权限去fork、exec
- ❖ 在TrustZone中实时监控提权行为
- ❖ 开启pxn，内核模式下不能再执行用户态代码/内核栈上得代码，漏洞利用成本提高、通用性降低

议程

- ❖ SU命令的背后
- ❖ ROOT手机的多种方法
- ❖ 利用内核漏洞
- ❖ AOSP/Samsung KNOX对抗手机Root
- ❖ 展望Android L

展望Android L

- ❖ 防御的能力依然有限，root依旧可行。一个合适的漏洞，依然可以绕过以上所有防御机制。
- ❖ 趋势上，漏洞利用的成本在变高
- ❖ 需要破坏系统中的一些全局安全机制，才能让应用完全不受限制

Thank you!
Q&A

retme7@gmail.com

weibo: @retme

blog: <http://retme.net>

<https://github.com/retme7>