好久不发帖了，今天来点干货

1、配置调试环境

首先，配置好调试环境。

Androidserver 的调试方法是 android_server -pxxxx 自定义port

Jdb的调试方法是 jdb -connect com.sun.jdi.SokcetAttach:hostname=xxxx,port=xxxx

2、断点

需要断在link中，主要是so的init函数。则找到BLX R4，十六进制代码是A0 47注意是thumb指令。由于高版本的安卓系统实行了ALSR技术，所以每次BLX R4指令加载的位置都不一样，但是可以进行搜索。

打开module，在libc.so中，断在fork上，监视子进程的建立。断在pthread_create上，监视子线程的建立。新的进程或者线程，都很有可能有anti代码的出现。

3、**子进程调试技术**

思路，是修改plt表，got表，使用libc.so的sleep函数，使子进程一建立就sleep。不能使用一个死循环，使用死循环后，由于时间片被占用完了，导致IDA attach不到子进程上面。然后就可以再用一个IDA来attach上子进程，进行双IDA调试。这个需要再在android终端启动一个android_server，同时指定不同的端口。大概就是./android_server -p34568

子进程从sleep中恢复出来后，只需要把堆栈修和寄存器修复好，就可以了。注意要恢复堆栈，否则会运行失败。这个时候，父进程的堆栈和子进程的堆栈和寄存器应该是一样的，或者大部分是一样的。

在调试的过程中，可以把某些不重要的线程给挂起来。往往还是很有用的。

顺便说一句，在linux其他的版本中，是可以通过设置内核的一些参数，来进行子进程调试的。

4、anti-debug技术

动态跟踪，慢慢的找吧。

查找/proc/xxx/android_server,gdbserver,trace

tracepid

notify fd了等等

还有一些利用IDA对thumb指令，arm指令翻译不好的问题等。

5、下面的指令是IDA插入的端点

thumb breakpoint

0xDE10

thumb2 breakpoint

0xA000F7F0

arm breakpoint

0xEF9F0001

查了一下rtld_db_dlactivity函数，这个函数实则默认情况下为空函数，这里的值应该为0，而当有调试器时，这里会被改为断点指令，0xDE10实则为thumb指令的断点。函数的功能是用于处理一些调试器的特殊情况的。比如调试器对模块的某个地址进行下断，但这个地址实际不存在，得在模块加载后才存在等的特殊情况下的函数。起的作用就是类似于协商解决特殊问题。具体的各位可以自己去查询相关资料。