# lab10\_oops实验说明

本实验的目的：编写一个简单的模块，并且人为编造一个空指针访问错误来引发oops错误。

## 基本实验步骤

1. 进入rlk\_lab/rlk\_basic/chapter\_10/lab1目录。

|  |
| --- |
| # export ARCH=arm  # export CROSS\_COMPILE=arm-linux-gnueabi-  # make BASEINCLUDE=/home/figo/work/runninglinuxkernel/runninglinuxkernel\_4.0 |

这里BASEINCLUDE指定你当前runninglinuxkernel\_4.0的目录路径。

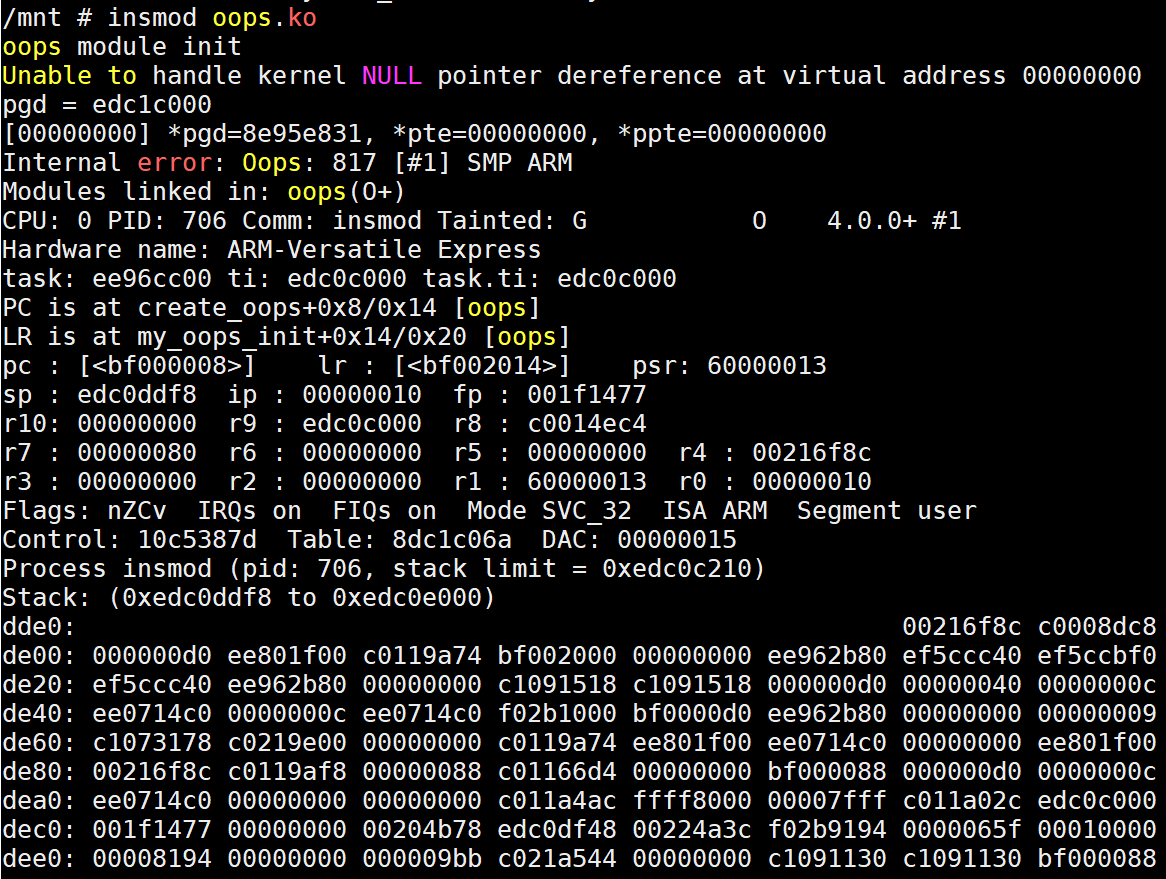
然后把\*.ko拷贝到runninglinuxkernel\_4.0/kmodules目录下面。

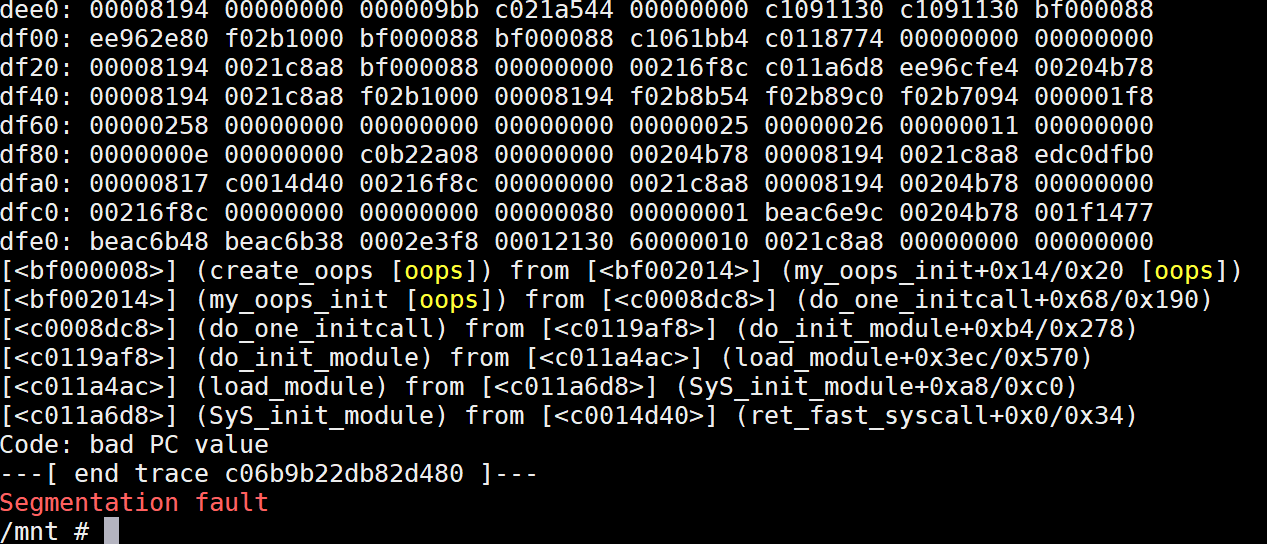
运行如下脚本启动Qemu。

|  |
| --- |
| #cd runninglinuxkernel\_4.0  # sh run.sh arm32 #启动虚拟机 |

在Qemu虚拟机:

|  |
| --- |
| #cd /mnt  # insmod oops.ko |





## 进阶思考

对于oops错误的分析，书上已经列举了很多有用的方法。

另外还有一种办法就是使用kdump+crash工具来分析oops导致的死机问题。

笨叔专门录制了一期死机专题的视频节目，长达5小时，包括Centos7.6和arm64两集。

死机专题视频课程分成上下两集，全程5小时高清视频，140多页ppt，8大死机案例：

上集：x86\_64专场，讲述Centos 7.x系统的死机案例分析 （3个多小时）

下集：ARM64专场 （约1.5小时）

全球唯一和独一无二的手把手解决死机问题的视频，只在奔跑吧Linux社区！

笨叔在这一次挑战死机的专题视频中，设计了8个死机案例，分别有：

lab1：简单的空指针引发的panic

lab2：访问已经被删除的list head链表

lab3：复杂一点的空指针引发的panic

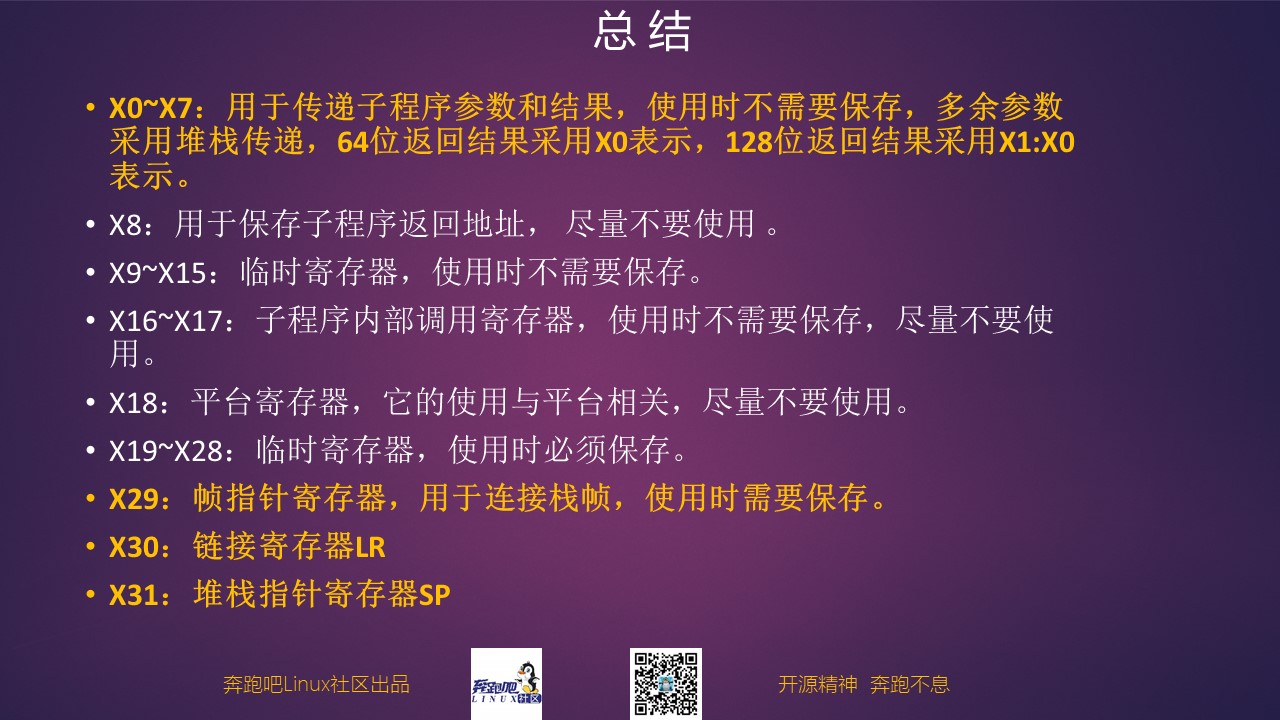
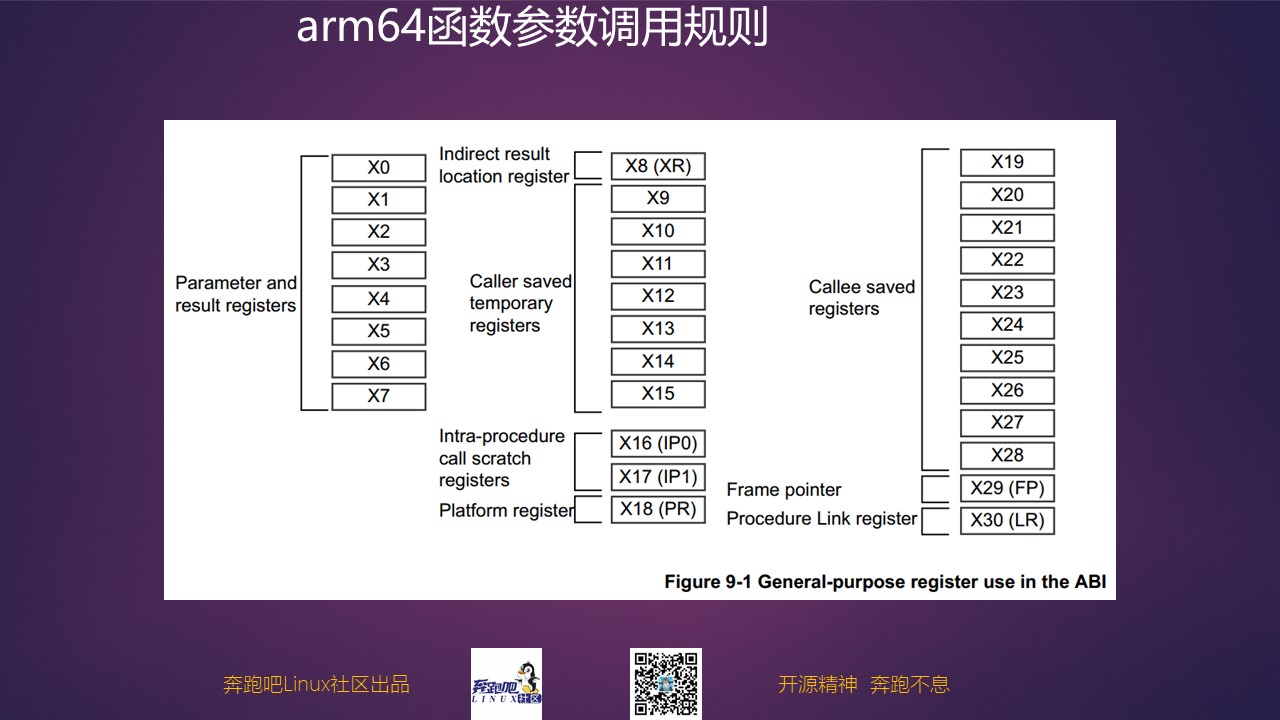
lab4：一个真实的驱动引发的死机

lab5：一个真实的驱动引发的死锁，导致系统假死

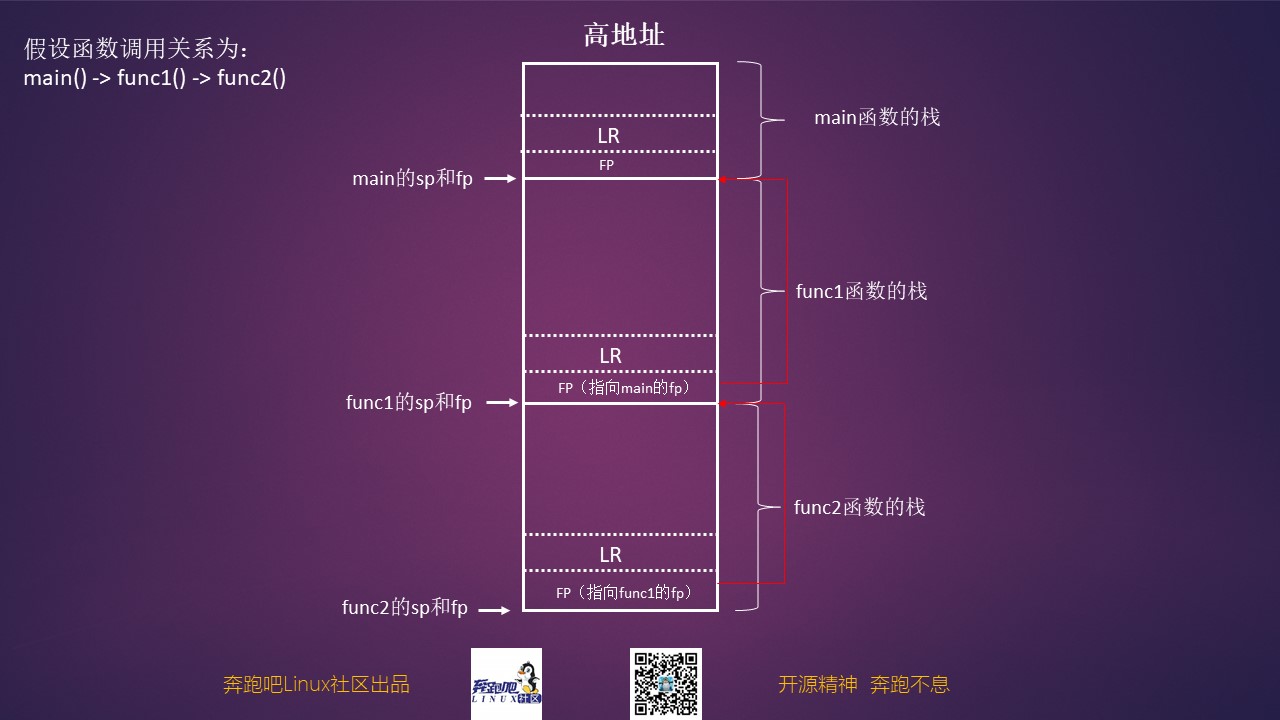
lab6：如何找到函数调用参数的在栈中地址然后获取具体的值

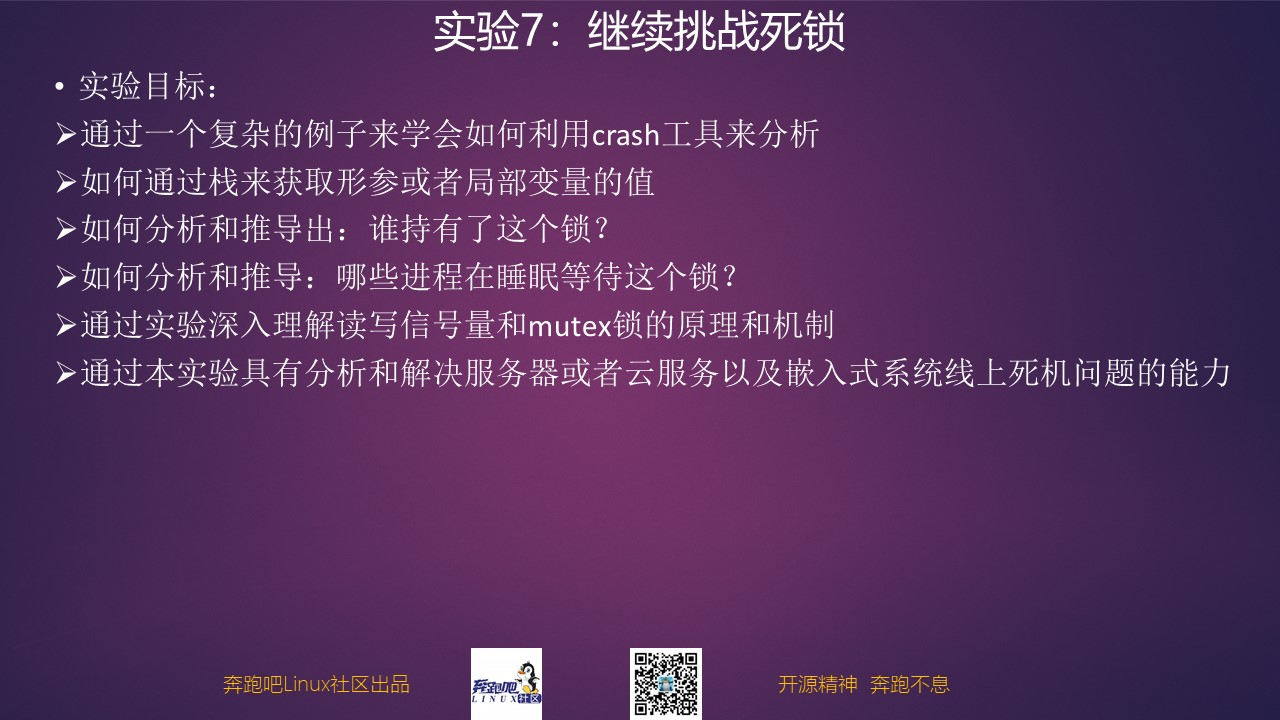
lab7：分析一个复杂的线上死锁导致的死机黑屏例子

lab8: 手工恢复函数调用栈backtrack （arm64）

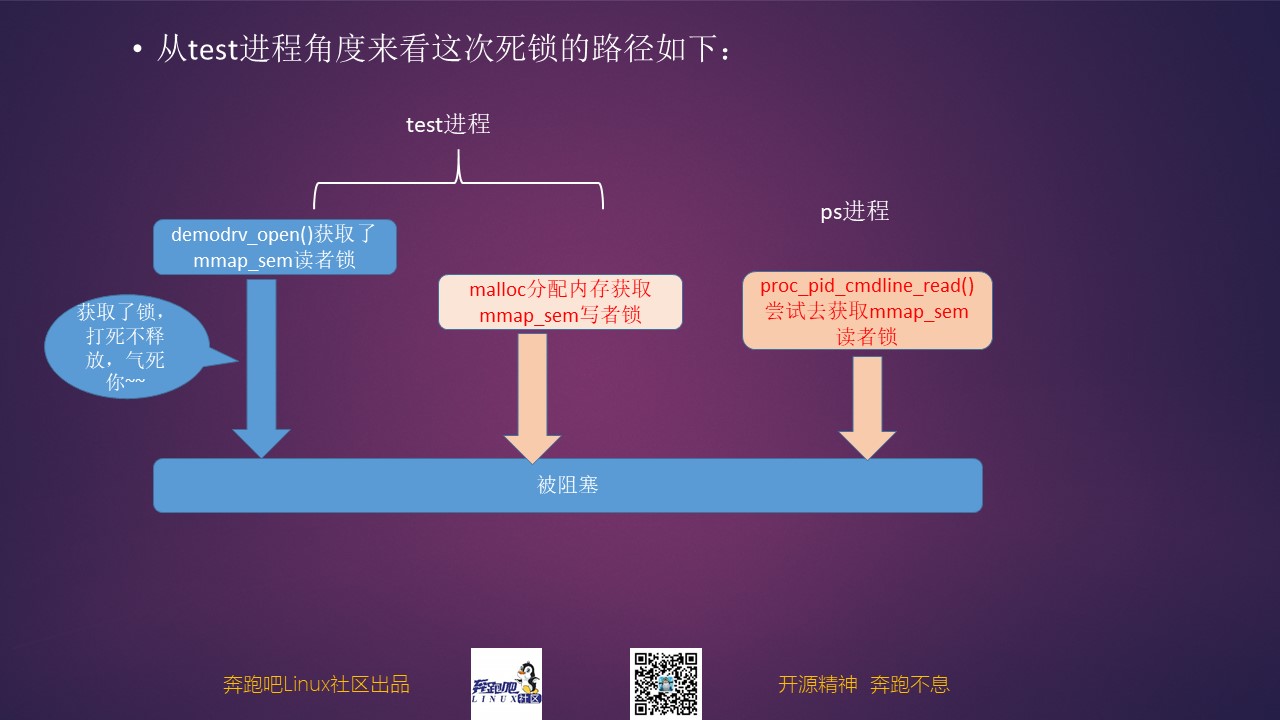
















***订阅第一季或者第二季旗舰篇的小伙伴都可以免费参加死机主题视频课程。***

如果大家对这些问题感兴趣，可以关注笨叔的旗舰篇视频，笨叔会在视频中和大家详细解答。



订阅和购买旗舰篇视频可以到：

1. 微店：



1. 淘宝店：https://shop115683645.taobao.com/

微信公众号：

