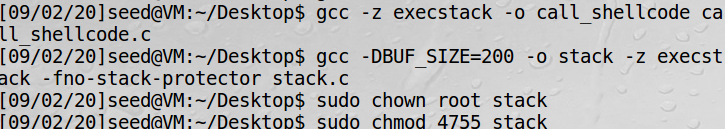
任务一（buffersize设置为200）

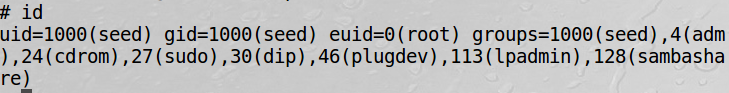


任务二：

填充的值



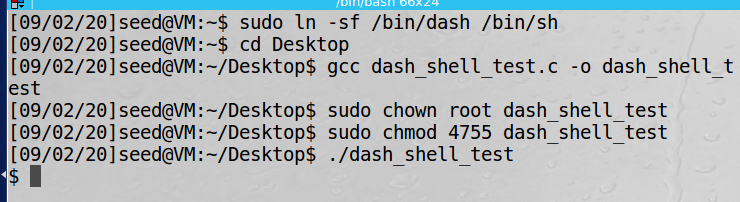
结果：



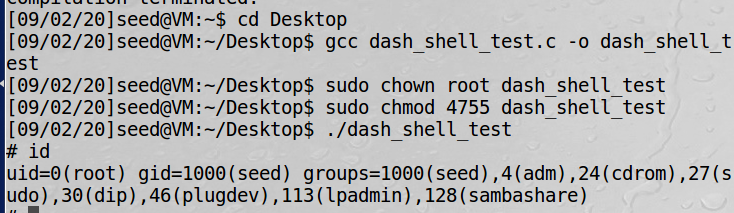
分析：通过运行gdb，计算出攻击代码应该写入的内存地址，并编译运行攻击程序，成功进入root状态

任务三：

注释掉setuid(0)

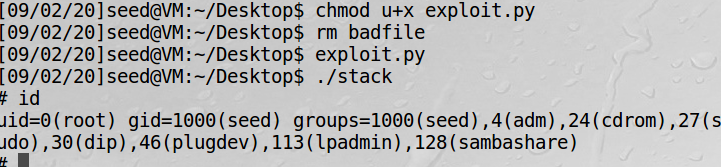


恢复setuid(0)



进入了root状态

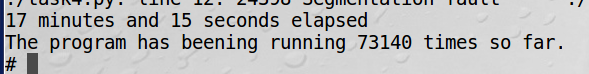
修改task2的文件后运行：



进入了root状态

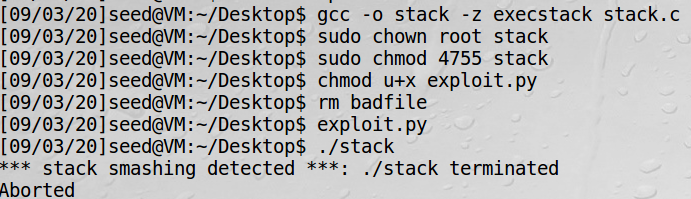
分析：为了避免dash shell对特权程序的降级，在调用dash程序之前通过setuid系统调用修改真正用户的ID，成功进入root状态。

任务四：

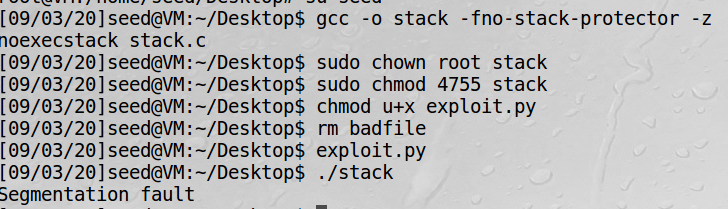


开启了地址随机，通过循环运行程序，最终使得程序地址与攻击代码写入地址匹配，完成攻击

任务五：



分析：开启了stack guard后栈溢出被检测到，进入root状态失败

任务六：  


分析：编译时使得栈中数据不可执行，将向栈中写入的代码当作数据，无法被执行