南京邮电大学

实 验 报 告

（2025/ 2026 学年 第 二 学期）

课程名称	计算机系统基础I
实验名称	炸弹实验
实验时间	2026 年 6 月 4 日
指导单位	基础教学中心
指导教师	

学生姓名		班级学号	
学院(系)	计算机学院	专　业	软件工程

实 验 报 告

实验名称	炸弹实验			指导教师	廖磊瑶
实验类型	设计	实验学时	2	实验时间	

一、 实验目的和要求

实验目的：

通过实现缓冲区溢出攻击，熟悉理解过程调用规则，包括控制的传递、参数的传递以及栈帧结构、缓冲区溢出问题等，了解缓冲区溢出保护，并进一步加深对理论课程中

关于程序的机器级表示各方面知识点的理解，以及反汇编、跟踪/分析/调试等常用技能

的掌握。

实验要求

作为实验目标的存在缓冲器溢出漏洞的可执行程序共有两个：ctarget 和 rtarget。对目标程序实施缓冲区溢出攻击（buffer overflow attacks）。通过造成缓冲区溢出来破坏目

标程序的栈帧结构，继而不再返回原来的调用程序，而是转向执行指定的子程序，要求

level1 转向 touch1 子程序，level2 转向 touch2 子程序，level3 转向 touch3 子程序

二、实验环境(实验设备)

硬件：微型计算机

软件：linux 操作系统、Gcc编译套件

三、实验代码及结果

中文五号宋体，英文五号 Times new roman 字体，1.25 倍行距

说明：这部分内容主要包括：

1.截屏实验运行结果

2.截屏显示记分牌上自己的实验成绩。

3.写出 5 个阶段攻击思路和过程。

四、实验小结（包括问题和解决方法、心得体会、意见与建议等）

中文五号宋体，英文五号Times new roman字体，1.25倍行距

说明：这部分内容主要包括：在编程、调试或测试过程中遇到的问题及解决方法、本次实验的心得体会、进一步改进的设想等。

(一)实验中遇到的主要问题及解决方法

1. phase_3 跳转表的间接寻址机制

phase_3 里的跳转表条目是 32 位有符号相对偏移，不是绝对地址。第一次直接把 .rodata 里的 4 字节当绝对地址来用，算出来的目标地址根本不存在。后来看清楚汇编：lea 0x1ad5(%rip), %rdx 把表基址 0x4031c0 装进 rdx，movslq (%rdx,%rax,4),%rax 是符号扩展的相对偏移，add %rdx,%rax 才得到实际跳转地址。明白了「基址加偏移」这套机制，才能正确计算各 case 的入口。这是 switch 语句在 x86-64 下的典型实现，以后碰到 notrack jmp *%rax 这种模式就要立刻想到跳转表。

2. func4 的算术右移与有符号除法语义

分析 func4 的时候我在比较方向上栽了跟头。汇编里 cmp %edi,%ecx 比较的是 ecx 和 edi（mid 和 x），随后的 jg 是「mid > x 跳转」走左分支，jl 是「mid < x 跳转」走右分支。我最初按 x 和 mid 顺手写反了递归方向，导致手算 func4(x,0,14) 的所有结果全偏。重新对照 cmp 操作数顺序后才弄清，调用 func4(0,0,14) 才能让返回值等于 7，第一个数应该是 0 而不是别的。另外 sar $1,%ecx 是算术右移，对负数会向负无穷取整，编译器为了模拟 C 的「向零取整」除法，专门在前面加了 shr 31 + add 来修正符号位。这是有符号除以 2 的标准代码模式。

3. phase_6 链表节点的非连续存放

phase_6 的 6 个链表节点 node1~node5 连续存放在 0x405330 开始的位置，但 node6 单独被链接器放到了 0x405210，没有紧跟在 node5 后面。一开始我以为节点是连续的，按地址 +0x10 读到第六个时拿到的是别的数据。后来发现只能通过 node5 的 next 指针字段（偏移 +8 处存的 0x405210）才能找到 node6 的真实位置。这让我意识到「结构体在内存里的物理布局」和「逻辑上的链表顺序」是两回事 —— 编译器和链接器有自己的内存分配策略，不能想当然按顺序读。

（二）实验心得

这次 BombLab 是一次真正的「逆向思维」训练，和 DataLab 完全相反 —— DataLab 是已知目标写代码，BombLab 是拿着黑盒反推程序逻辑，思维方式要彻底翻过来。

收获最大的是理解了高级语言控制结构在汇编层面的真实形态。switch 在汇编里是跳转表（phase_3），if-else 是条件跳转链（phase_2），递归函数是栈帧一层层累叠（phase_4 的 func4）。phase_1 到 phase_4 把这些结构都过了一遍，看完之后对「C 代码是怎么翻译成机器指令」有了直观的认识，不再是抽象的概念。

phase_5 的数组索引技巧让我印象深刻。and $0xf,%edx 这一行就把字符变成了 0~15 的下标，整个查表逻辑压缩在三行汇编里。理解了这个之后再回头看 DataLab 里的位掩码技巧，感觉是一脉相承的 —— 位运算在底层代码里无处不在，往往比看起来更高效。

GDB 的使用熟练度提升了很多。b *0xADDR 设断点、x/s 读字符串、x/24wx 看内存布局、info registers 看寄存器状态、si 单步指令、c 继续执行 —— 这套工作流跑完 5 个 phase 之后基本

五.支撑毕业要求指标点
六、指导教师评语 (含学生能力达成度的评价)

成 绩		批阅人		日 期	

评分细则	评分项	优秀	良好	中等	合格	不合格
	遵守实验室规章制度					
	学习态度					
	算法思想准备情况					
	程序设计能力					
	解决问题能力					
	课题功能实现情况					
	算法设计合理性					
	算法效能评价					
	回答问题准确度					
	报告书写认真程度					
	内容详实程度					
	文字表达熟练程度					

	其它评价意见					
	本次实验能力达成评价（总成绩）					