




0. 实验背景

这个实验源自CMU（卡内基梅隆大学）计算机系统经典教材《深入理解计算机系统》的配套实验，是学习程序底层运行的“通关密语”！教材详情可戳官网：[CS:APP3e](#)

实验准则：独！立！作！战！

-  严禁：▸ 搜索引擎抄近道
-  正确姿势：
 - ✓ 对着汇编代码“自言自语”分析流程
 - ✓ 用 `break \x命令\print` 当数字显微镜
 - ✓ 和同学讨论思路（但别交换代码/具体地址！）
-  破关锦囊：
 - 主武器：课本3.1-3.6章
 - 辅助技：疯狂用 `gdb / objdump`

实验概述

1. 实验目的

- 理解C语言函数的汇编级实现及缓冲区溢出原理
- 掌握栈帧结构与缓冲区溢出漏洞的攻击设计方法
- 进一步熟练使用Linux下的调试工具完成机器语言的跟踪调试

2. 实验学时

本实验为4学时。

3. 实验环境

Linux 64-bit, 汇编/C语言

你既可以选择使用**ubuntu虚拟机实验环境**开展实验，也可以利用自己搭建的 Linux 环境完成相关操作。

4. 实验内容

本次实验的目标是对名为“**bufbomb**”的可执行程序展开一系列难度逐步递增的缓冲区溢出攻击。为达成这一目标，同学们需要设计五个不同级别的攻击字符串。每个攻击字符串的设计目的都是通过溢出缓冲区，改变程序运行时的内存映像，特别是栈帧的内容。

在实施缓冲区溢出攻击的过程中，同学们需要对 **bufbomb** 可执行文件程序进行反汇编分析，并借助 gdb 调试器跟踪每一阶段的机器代码。在这个过程中，要深入理解关键机器指令的行为和作用，从而推断出有效的攻击字符串。

具体的五个实验级别如下：

- Level 0: smoke
 - 目标：让目标程序调用smoke函数。
 - 策略：构造一个攻击字符串，其中包含填充数据以覆盖返回地址，并用smoke函数的地址替换它。
- Level 1: fizz
 - 目标：让目标程序使用特定参数调用Fizz函数。
 - 策略：除了覆盖返回地址，还需要在栈上布置正确的参数。攻击字符串将包含填充数据、Fizz函数的地址以及所需参数的值。
- Level 2: bang
 - 目标：让目标程序调用Bang函数并修改全局变量。
 - 策略：此级别的攻击需要更精细的控制。攻击字符串将包括填充数据、Bang函数的地址以及修改全局变量的指令序列。
- Level 3: boom
 - 目标：进行无感攻击，并传递有效返回值。
 - 策略：此级别的攻击需要确保程序在调用函数后能够正常返回，并且不引起任何异常。攻击字符串将包括确保程序稳定性的指令，并可能使用NOP滑块（无操作指令序列）来确保程序执行的流畅性。
- Level 4: kaboom
 - 目标：在栈帧地址变化时进行有效攻击。
 - 策略：由于栈帧地址可能因程序运行时的不同条件而变化，因此此级别的攻击需要更加稳健和灵活。攻击字符串可能使用相对跳转而不是绝对地址，以确保无论栈帧如何

变化，攻击都能成功。