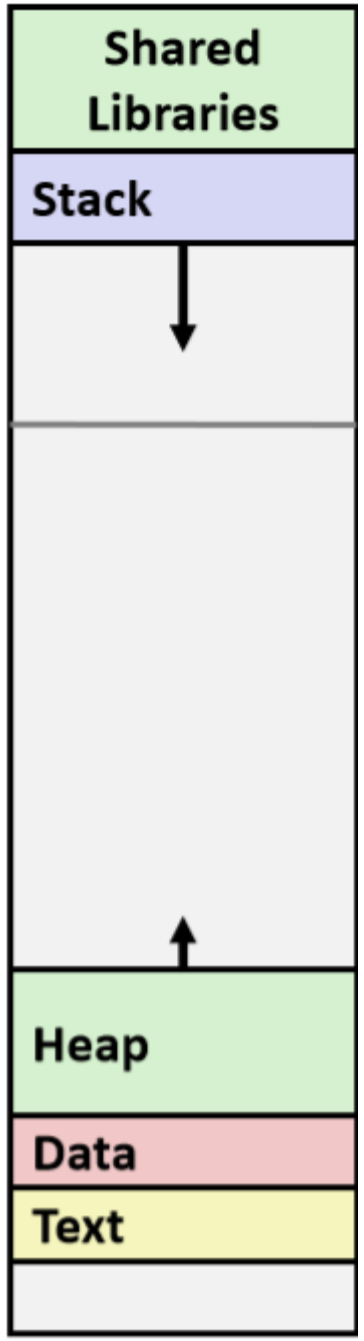
10.25

主要内容

- Machine Advanced
  - Unions
  - Mem layout
  - Buffer overflow attack
- Architecture
  - Y86 ISA
  - Logic Design

Memory layout

- 记住是什么样子的
- 每个进程都有自己独立的地址空间
- 最上面的那个部分内存是内核空间
- 为什么Heap要分成两个部分?

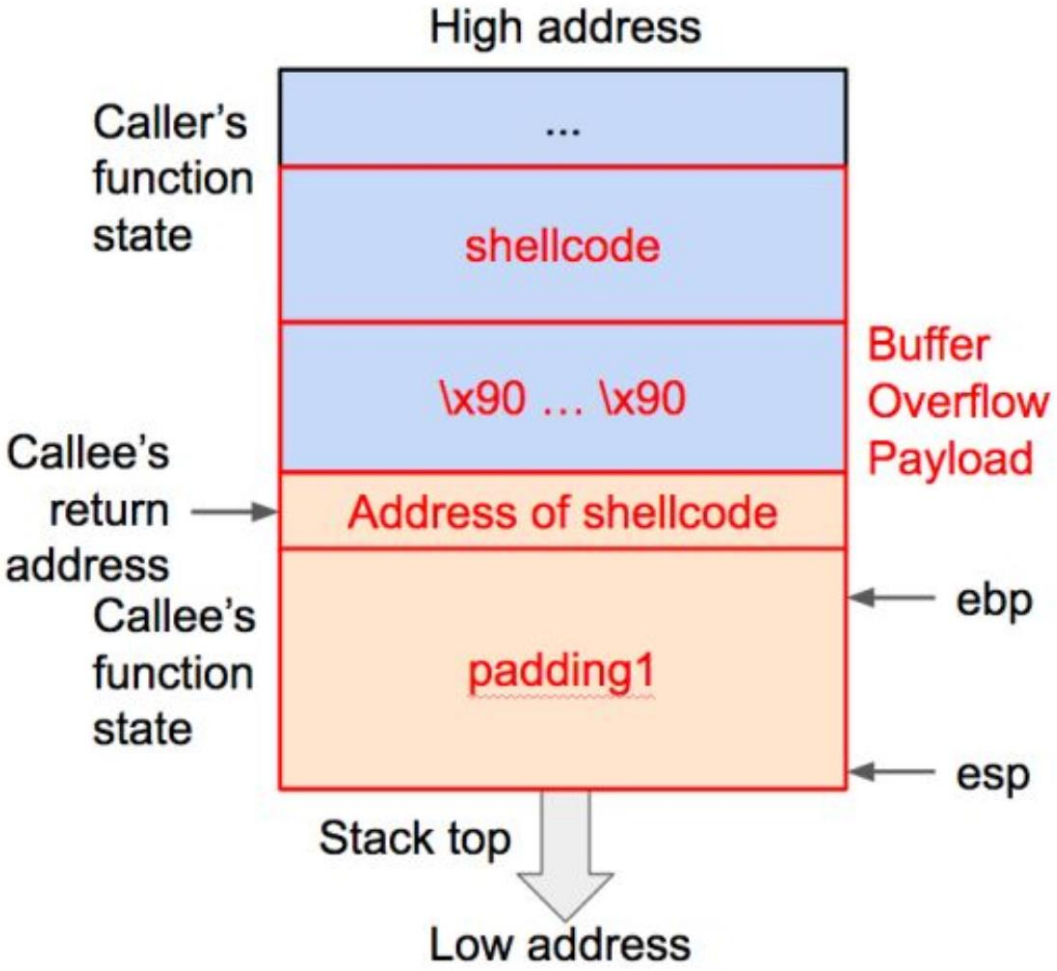


Buffer Overflow Attack

- 栈中插入可执行代码
- 跳转至libc
- ROP
- 绕过金丝雀值

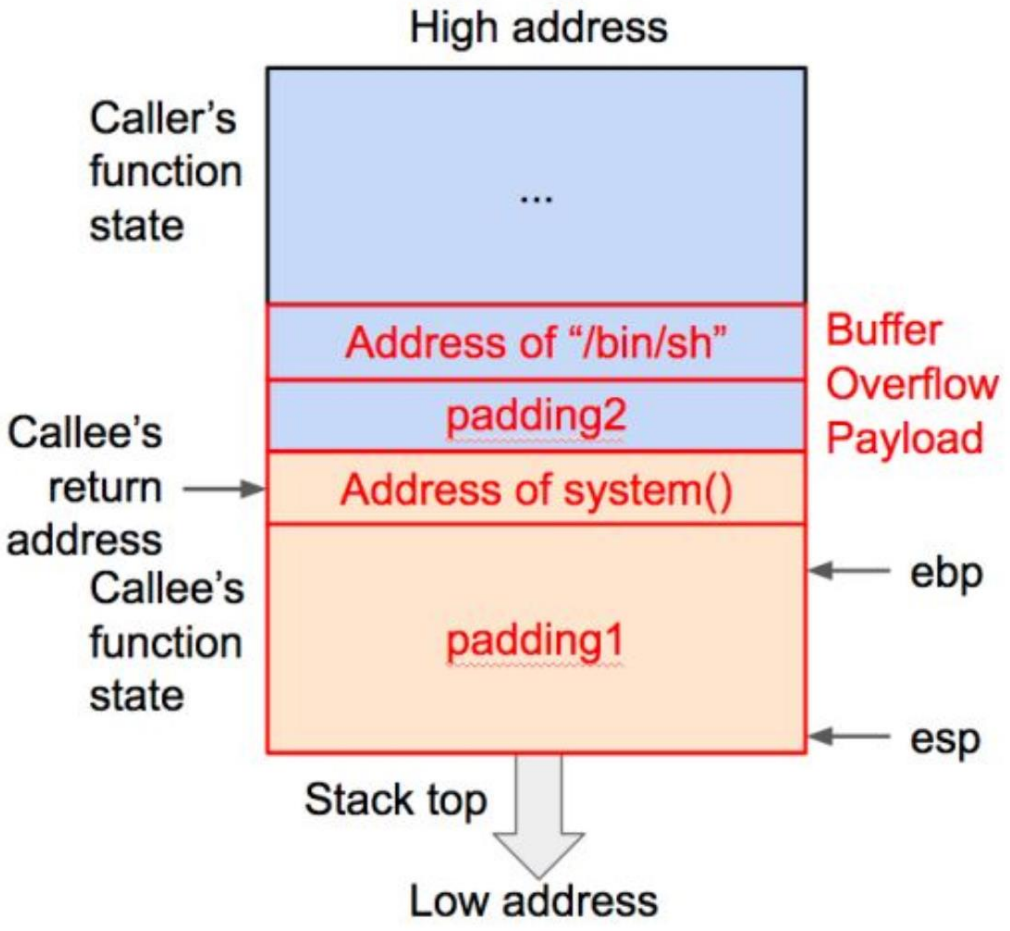
栈中插入可执行代码

- 防御方法1：ASLR
  - 应对：NOP sled
  - 只要我的跳转位置落在NOP的范围里就可以执行
- 防御方法2：栈无法执行
  - 应对：ROP
- 防御方法3：金丝雀值



跳转至libc

- 和刚刚的想法类似
- 但是这次借用本身程序的代码
- 还是会收到ASLR的影响
- 例子：
  - 如果程序中本身有对system()函数的调用，就可以想办法找到这个函数的入口地址。
  - 如果再找到“/bin/sh”的地址，就可以获得shell的执行权限



ROP

- 这个就和书上说的一样
- 有一些小工具可以帮助做ROP攻击

ROPgadget Tool

This tool lets you search your gadgets on your binaries to facilitate your ROP exploitation. ROPgadget supports ELF/PE/Mach-O format on x86, x64, ARM, ARM64, PowerPC, SPARC and MIPS architectures. Since the version 5, ROPgadget has a new core which is written in Python using Capstone disassembly framework for the gadgets search engine - The older version can be found in the Archives directory but it will not be maintained.

绕过金丝雀值

- 这部分我了解的也比较少，简单讲几个
- 利用非格式化的printf试图输出金丝雀值
- 利用多进程程序对金丝雀值进行爆破
- 利用__stack_chk_fail泄露信息

Architecture

- 如何自己设计一个CPU?
- 我们要做的事情:
  - 设计一套可以操纵cpu的指令
  - 让自己的cpu能“认识”我们设计的指令
- 传统的想法：把指令设计成二进制代码
  - cpu可以通过电路来比较二进制数值
  - 通过比较的结果去激活相应的功能模块

Y86 ISA

- 设计了一套指令集体系结构
  - 包括指令格式， 寄存器命名， 操作数等

- 要会写Y86的汇编程序
  - 考试和lab都会涉及到

- CISC v.s. RISC

Y86 Logic Design

- 微电子与电路基础课中学的很多东西会用在这里
  - 内部寄存器：基于D-flip-flop，时钟上跳沿改变状态
  - 组合逻辑电路
- 对于CPU内部寄存器的理解
- 组合逻辑电路 ←→ HCL语言
  - 要会写HCL语言
- 之后的课会涉及到很多这样的模块和图
  - 要会画并且理解，这都是考试重点