第1章 软件安全概论

1. 软件定义万物的含义，网络空间的两个子空间
2. 牢记任何软件都是不安全的
   1. 为什么软件测试无法保证软件的安全性
   2. 在测试之前尽量多地解决安全问题
   3. 存在问题的代码案例
3. 软件不安全性的外在表现
   1. 运行不稳定，导致软件崩溃或非正常退出；
   2. 恶意攻击，达到信息窃取、系统破坏等目的；
4. 软件安全问题产生的原因（理解基本原理）
   1. 软件缺陷和错误
   2. 从软件开发者的角度看软件不安全的原因
      1. 软件开发没有严格遵守软件工程流程
      2. 大多数软件，结构都相当大并且复杂，不易开发和维护
      3. 开发者没有采用科学的编码方法
      4. 测试不到位
   3. 从软件工程客观角度：
      * 1. **软件复杂性和工程进度的平衡**
        2. **安全问题的不可预见性**
        3. **软件需求的变动**
        4. **软件组件之间的交互的不可预见性**
5. 软件安全防护手段 （理解基本原理）
   1. 安全设计与开发
   2. 保障运行环境
   3. 加强软件自身行为认证
   4. 恶意软件检测与查杀
   5. 黑客攻击防护
   6. 系统还原
   7. 虚拟隔离

第2章 软件安全基础知识

1. 恶意软件与系统引导联系
   1. 熟悉并掌握操作系统引导的过程
   2. 恶意软件可以在哪些阶段重新获取控制权
2. Windows/Linux 虚拟地址空间
   1. 程序在内存中的映像，即程序地址空间分布
   2. 虚拟地址通过页表的方式转换为物理地址
   3. 有关malloc的三个思考题
3. FAT32文件系统
   1. 了解FAT表和簇链的定义
   2. 为什么一般删除的文件可以恢复/如何彻底删除文件
4. PE/ELF 文件的结构
   1. 具体文件格式及常用节头部
   2. PE 文件与内存之间的映射
      1. 掌握文件偏移地址，装载基址，虚拟内存地址等概念

第10章 软件缺陷与漏洞机理基础

1. 软件漏洞定义及其要素
2. 软件漏洞要素
3. 漏洞分类方法
   1. 漏洞成因分类
   2. 漏洞威胁分类（到5种）
   3. 漏洞利用方式分类
4. 软件漏洞生命周期
5. 漏洞利用对软件系统的威胁 （参照威胁分类5种）
6. 典型漏洞类型
   1. 缓冲区溢出
   2. 跨站脚本攻击
   3. 注入类漏洞

第10.1章 栈缓冲区溢出

1. 栈区域的概念
   1. 栈区的内容（参数，返回地址，局部变量等）
   2. 栈操作
   3. 函数调用过程与栈分布图
   4. 栈溢出的根本原因
2. 栈溢出的利用方式
   1. 覆盖返回地址 （JMP ESP 跳转指令）
   2. 覆盖SEH中的handler
   3. Shellcode 的概念与常见功能

第10.2章 堆缓冲区溢出

1. 堆区域的概念
   1. 堆的数据结构和管理（堆表+堆块）
      1. 堆表：空表和块表
      2. 堆块：块首和块身
         1. 占用块 vs 空闲块
2. 堆溢出
3. 堆溢出的利用方式
   1. what→where操作或 Dword Shoot
   2. 堆溢出能够利用成功的前提条件
   3. 堆喷射利用原理及其检测与防御
4. 堆溢出的防御策略
   1. Safe Unlink
5. UAF

a.UAF基本原理与可能的安全问题。

1. 格式化字符串
   1. 格式化字符串函数（printf）的栈结构图
   2. 格式化字符串的危害
      1. 越界读操作 ---> 数据泄露 （Information leak）
      2. 越界写操作 ---> 数据破坏（Data Corruption）
   3. 格式化字符串防御

第10.3整数溢出及其他漏洞类型

1. 整数溢出
   1. 整数溢出的概念
   2. 整数溢出的典型表现形式（需全部掌握）
   3. 整数溢出和缓冲区溢出的经典结合方式
      1. malloc 的参数计算中出现整数溢出，导致后续的堆区域访问出现问题
      2. 内存拷贝的size参数传递出现符号溢出，导致后续的缓冲区访问出现问题
   4. 整数防护手段 （安全意识，避免隐患操作等）
2. C++ 虚函数覆盖：一种漏洞利用的通用方法，用于劫持程序控制流

第10.4章

1. 漏洞利用
   1. 漏洞发现 （0-day vs 1-day vs N-day）
   2. Payload vs Shellcode （Payload与漏洞关联，Shellcode独立于漏洞）
   3. 漏洞利用的目标
   4. 理解漏洞利用的整体流程
   5. 理解Shellcode 的设计与编写流程，掌握ShellCode实例运行原理
   6. 理解为什么需要地址重定位，并能够说明重定位代码含义
2. Return-to-libc (ret2lib)
   1. system(“/bin/sh”)＋exit() 的栈结构设计
   2. 防御方法和相应的攻击方法
      1. ASCII Armoring
      2. Ret2PLT
3. Return-Oriented-Programming (ROP)
   1. 掌握二进制代码重用原理与示例，具备ROP分析与设计能力 （重点考察）
4. 源代码安全审计
   1. 熟悉程序切片、符号执行、污点分析概念
5. 静态分析技术 vs 动态分析技术
   1. 模糊测试 （Fuzz Testing）

第11章

1. 数据执行保护 – DEP / NX
   1. 防御机理及不足
   2. 绕过思路
2. 栈溢出保护 – Stack Guard / Stack Canary
   1. 防御机理及不足
   2. 绕过思路
3. 地址空间分布随机化 – ASLR
   1. 防御机理及不足
   2. 绕过思路
4. SafeSEH / SEHOP
   1. 防御机理及不足
   2. 绕过思路

第12章 构建安全的软件

1). 软件开发生命周期

掌握生命周期的几个阶段。

2).软件设计阶段威胁建模

a.威胁建模过程

b.根据威胁画威胁树

3).安全代码的编写

4).软件的安全性测试

5).漏洞响应和产品的维护