Homework6

吴承泽 SA23011083

Chapter9

1 简述shellcode 的概念以及编写shellcode的步骤。

shellcode的概念：Shellcode是一种特殊类型的计算机程序，主要设计用于利用软件漏洞或实现攻击。它通常以二进制形式存在，直接由操作系统加载和执行。Shellcode的目标是在受害者系统上执行指定的操作，例如提取敏感信息、建立远程访问通道或执行特定的恶意功能。

编写shellcode的步骤：

要成功地利用缓冲区溢出漏洞，必须解决三个技术问题：

- 跳转地址放在攻击串的什么位置（偏移，offset）；
- 跳转地址的值（调试目标进程，确定(或猜测)目标缓冲区的起始地址+偏移）；
- 编写期望(能实现某些功能)的shellcode。

具体为：

1. **了解目标系统：** 在编写Shellcode之前，需要深入了解目标系统的体系结构和操作系统。不同的系统可能有不同的指令集和系统调用。
2. **选择合适的汇编语言：** Shellcode通常是用汇编语言编写的。选择适合目标系统的汇编语言，并确保你了解它的语法和特性。
3. **编写Payload：** 这是Shellcode的核心，包含实际执行的指令。这可能涉及到获取系统权限、执行操作、建立连接等。在编写Payload时，考虑目标系统的特性，以确保Shellcode在不同系统上都能正确执行。
4. **处理空字符：** 在Shellcode中，通常需要注意避免使用空字符，因为在许多情况下，空字符可能导致字符串截断或其他问题。这可能需要使用编码技术或其他手段来规避。
5. **测试和调试：** 在将Shellcode用于实际攻击之前，务必进行充分的测试和调试。这可能包括在虚拟环境中模拟目标系统，以确保Shellcode的正确性和可用性。
6. **绕过防御措施：** 防病毒软件和其他安全措施可能会检测并阻止Shellcode的执行。编写Shellcode时，可能需要使用各种技术（如多态性、加密、混淆）来规避这些防御措施。
7. **选择注入方式：** 如果你计划通过漏洞注入Shellcode，需要了解目标程序的结构，并选择适当的注入方式，如缓冲区溢出、格式化字符串漏洞等。
8. **生成二进制代码：** 将汇编代码汇编为目标系统的机器码。这可能需要使用汇编器和链接器等工具，具体取决于你选择的汇编语言和目标系统。
9. **测试攻击：** 在实际攻击之前，通过模拟目标环境进行最终测试。确保Shellcode按预期执行，并且能够达到攻击者的目的。

2 Linux环境下的shellcode为什么不调用libc中的库函数，而是利用系统调用?

这主要有以下几个原因：

1. **大小和简洁性：** 系统调用的直接使用可以使Shellcode更加紧凑，因为避免了加载和调用大型libc库所需的额外开销。Shellcode通常需要非常小的空间，以便更容易在攻击中传输和执行。
2. **避免依赖：** 在某些情况下，目标系统上可能没有完整的libc库，或者在受限制的环境中无法使用。直接使用系统调用可以确保Shellcode在没有额外依赖的情况下正常运行。

3. **逃避防御措施：** 部分安全防御措施，如堆栈保护（Stack Protector）和地址空间布局随机化（ASLR），可以增加攻击者调用库函数时的难度。通过直接调用系统调用，攻击者可以更容易地规避这些防御措施。
4. **难度较高：**调用libc需要对应机器中提高动态链接加载器进行链接，才能够正常使用，而利用系统调用可以直接使用操作系统中的内核函数进行攻击，相对来说技术复杂性较低。

3 在攻击字符串中4字节的RET除了其取值范围要猜测准确外，还有什么需要考虑的（或者说有什么限制）？

1. **字符串结束字符**：shellcode中需要注意字符'\x00'，因为'\x00'是字符串结束标志。由于shellcode是要作为字符串拷贝到缓冲区中去的，在'\x00'之后的代码将丢弃。因此，shellcode中不能包含'\x00'。
2. **ASLR（地址空间布局随机化）：** 如果目标系统启用了ASLR，攻击者必须考虑返回地址的随机性。攻击者可能需要使用其他漏洞或技术绕过ASLR，或者尝试通过其他手段获取目标地址的信息。
3. **地址对齐：** 操作系统通常要求函数调用的返回地址是按照某种规则对齐的。在x86架构上，典型的要求是4字节对齐（即地址的最低两位是00）。如果返回地址没有正确对齐，可能导致程序崩溃或执行未知的行为。
4. **堆栈溢出：** 攻击字符串中的RET通常用于利用堆栈溢出漏洞。攻击者需要确保输入的数据能够溢出到目标函数的返回地址，并将其覆盖为攻击者控制的地址。
5. **大小缓冲区不同策略**：如果被攻击的目标缓冲区较小，不足以容纳shellcode，则将shellcode放在被溢出缓冲区的后面；如果目标缓冲区较大，足以容纳shellcode，则将shellcode放在被溢出缓冲区中。
6. **攻击指令ShellCode的位置：** 返回地址指向的地址通常是Shellcode的开始地址。攻击者需要确保Shellcode在正确的位置，以便被正确执行。

Chapter10

1 简述进程跳转攻击方法的基本思想

从系统必须加载的动态链接库(如ntdll.dll，kernel32.dll)中寻找call esp和jmp esp指令，记录下该地址（溢出攻击的跳转地址），将该地址覆盖函数的返回地址，而将shellcode放在返回地址所在单元的后面。