《Shellcode编码解码》实验报告

姓名：梁婧涵 学号：2112155 班级：1120

实验名称：Shellcode编码解码实验

实验要求：

复现第五章实验三，并将产生的编码后的shellcode在示例5-4中进行验证，阐述shellcode编码的原理和shellcode提取的思想。

在很多的漏洞利用场景中，Shellcode的内容会受到限制，这种限制不仅来自存在漏洞的软件自身，在某些情况下，也来自一些基于特征的IDS（网络入侵检测系统）系统。我们先来看看Shellcode会受到哪些限制。

首先，所有的字符串处理函数都会对NULL字节进行限制，通常我们会选择对Shellcode指令进行优化以避免在Shellcode中直接出现NULL字节。

其次，在某些处理流程中可能会限制0x0D（\r）、0x0A（\n）或者0x20（空格）字符。

最后，有些函数会要求Shellcode必须为可见字符（ASCII值）或者Unicode值。有些时候，我们还会收到基于特征的IDS系统对Shellcode的拦截。

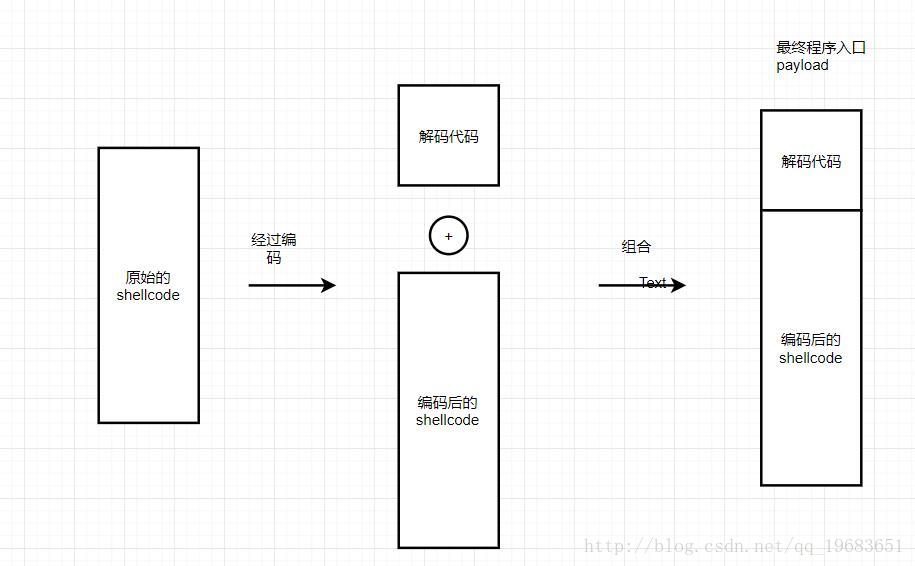
虽然在成功利用漏洞的路上有如此多的阻碍，但对漏洞的利用仍然不是绝对不可能的。目前，我们要想绕过坏字符限制，可以使用两种方法：

第一种就是前面介绍的对Shellcode中的指令进行优化。但由于对Shellcode中的指令进行优化不仅需要强大的汇编语言基础，在关心程序逻辑和流程时，还需要精心挑选合适的指令进行优化，所以，对于功能比较复杂的Shellcode进行指令优化，避开所有的限制字符的过程简直令人崩溃。但幸运的是，我们还有另外一种选择------Shellcode编码技术。

实验过程：

1. 将代码中0的部分异或掉，代码含有编码器，主函数，编码对象，encode函数完成编码，对每一个字节完成异或，写到一个文件中。执行，打开test中的encode，生成编码后的Shellcode。
2. 解码程序：越过decoder记录shellcode起始地址eax，用0x44作为key，末尾放一个0x90作为结束符。call lable运行call指令，反汇编，地址：00401798，下一个地址返回入栈，进行push操作，栈顶改变。F10，pop eax

Shellcode编码原理：



Shellcode编码是指将原始的二进制shellcode进行编码，使其在传输或存储时更难被检测到或防止被修改。常见的编码方法有base64编码、xor编码、rot13编码等。

其中，base64编码是将二进制数据转换为可打印的ASCII字符，可以将shellcode编码为纯文本形式，避免二进制数据在网络中被修改或传输时被误认为是控制字符。而xor编码则是将shellcode和一个密钥进行异或运算，生成新的shellcode。解码时再用相同的密钥进行异或操作，还原原始的shellcode。rot13编码则是将字母表中的每个字母按照顺序向后移动13个位置，可以简单地混淆shellcode，增加其破解难度。

编码后的shellcode需要在执行时进行解码，才能还原为原始的二进制形式。为了实现这一点，编码器通常会在shellcode前面添加一段解码程序，用于解码shellcode。这段解码程序可以直接写入exploit代码中，也可以通过编译器生成。

shellcode提取的思想：

Shellcode提取的思想是指从一个漏洞利用程序中提取出有效负载部分的二进制代码，这个有效负载就是Shellcode。

Shellcode通常是用于利用操作系统或应用程序的漏洞，实现攻击者的目标，比如获取系统权限、执行恶意代码等。在漏洞利用过程中，攻击者通常需要向受害者发送包含Shellcode的数据包或者通过其他方式传递Shellcode到受害者的计算机上。

提取Shellcode的方法取决于漏洞利用程序的具体实现，但通常可以通过以下步骤实现：

1.找到Shellcode在漏洞利用程序中的起始位置和长度。

2.将Shellcode从漏洞利用程序中提取出来，可以使用命令行工具或手动分离代码，以便在后续的利用过程中使用。

3.验证Shellcode是否可用，可以通过编写测试代码或使用反汇编工具进行验证。

4.编码Shellcode，以便在传输或存储时更难被检测到或防止被修改。常见的编码方法有base64编码、xor编码、rot13编码等。

5.将Shellcode注入到目标系统中，可以使用命令行工具、脚本或其他自动化工具实现。