1. 简述JNZ和JZ指令的用法及区别。

JNZ : jump if not zero 结果不为0则转移

jz=jump if zero，即零标志为1就跳转。jz的另一种写法就是je，je=jump if equal，jz和je的作用是完全一样的

2. 简述TEST指令的用法。

Test命令将两个操作数进行逻辑与运算，并根据运算结果设置相关的标志位。结果本身不会保存。Test命令的两个操作数不会被改变。运算结果在设置过相关标记位后会被丢弃。

3. 简述MOVSX指令的用法。

汇编语言数据传送指令MOV的变体。带符号扩展，并传送。

例如movsx ecx,byte ptr ss:[esp+0x4]，将esp+0x4作为有符号数存入ecx。

4. 简述进行程序爆破的思路（如何定位要爆破的指令？定位后如何爆破？）

暴力破解的一般流程

1、有壳者自然得先脱壳

2、尝试随机输入注册码看看有无提示字符串，如果有即可根据字符串快速定位判断跳转区域。

3、用下断点调试的方法精确定位需要爆破的指令。把判断验证失败的这段逻辑用NOP填充 。

4、复制保存！

5. 在“面向汇编的逐句解析”的示例中，有一条注册码判断指令为cmp byte ptr ss:[esp+0x9],0x2D。请解释：

a) 这条指令是如何访问内存的；

b) 这条指令的含义是什么。

1. 我们计算机的小端模式，是指数据的高位保存在内存的高地址中，而数据的低位保存在内存的低地址中。cmp指令为第一个操作减去第二个操作数,但不影响第两个操作数的值,它影响flag的CF，ZF，OF，AF，PF若执行指令后:ZF=1,则说明两个数相等，

2. 将esp+9处byte减去2D（十进制45），若执行指令后:ZF=1,则说明两个数相等。

6. 简述MD5算法流程。

对MD5算法简要的叙述可以为：MD5以512位分组来处理输入的信息，且每一分组又被划分为16个32位子分组，经过了一系列的处理后，算法的输出由四个32位分组组成，将这四个32位分组级联后将生成一个128位散列值。

　　 在MD5算法中，首先需要对信息进行填充，使其字节长度对512求余的结果等于448。因此，信息的字节长度将被扩展至N\*512+448，即N\*64+56个字节（Bytes），N为一个正整数。填充的方法如下，在信息的后面填充一个1和无数个0，直到满足上面的条件时才停止用0对信息的填充。然后，在在这个结果后面附加一个以64位二进制表示的填充前信息长度。经过这两步的处理，现在的信息字节长度=N\*512+448+64=(N+1)\*512，即长度恰好是512的整数倍。这样做的原因是为满足后面处理中对信息长度的要求。

MD5中有四个32位被称作链接变量的整数参数，他们分别为：A=0x01234567，B=0x89abcdef，C=0xfedcba98，D=0x76543210。

　　 当设置好这四个链接变量后，就开始进入算法的四轮循环运算。循环的次数是信息中512位信息分组的数目。

　　 将上面四个链接变量复制到另外四个变量中：A到a，B到b，C到c，D到d。

　　 主循环有四轮（MD4只有三轮），每轮循环都很相似。第一轮进行16次操作。每次操作对a、b、c和d中的其中三个作一次非线性函数运算，然后将所得结果加上第四个变量，文本的一个子分组和一个常数。再将所得结果向右环移一个不定的数，并加上a、b、c或d中之一。最后用该结果取代a、b、c或d中之一。

所有这些完成之后，将A、B、C、D分别加上a、b、c、d。然后用下一分组数据继续运行算法，最后的输出是A、B、C和D的级联。

为：A=0x01234567，B=0x89abcdef，C=0xfedcba98，D=0x76543210。