常见修改(机器码) PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/287/2021_2022__E5_B8_B8_E8_A7_81_E4_BF_AE_E6_c98_287057.htm 74=>75 74=>90 74=>EB 75=>74 75=>90 75=>EB jnz->nop 75->90(相应的机器码修改) jnz -> jmp 75 -> EB(相应的机器码修改) jnz -> jz 75->74 (正常) 0F 85 -> 0F 84(特殊情况下,有时,相应的机器码修改) 四.两种不同情况的不同修改方法 1.修改为jmp je(jne,jz,jnz) =>jmp相应的机器码EB （出错信息向上找到的第一个跳转）jmp的作用是绝对跳，无条件跳，从而跳过下面的出错信息 xxxxxxxxxxx 出错信息，例如：注册码不对，sorry,未注册版不能...，"Function Not Avaible in Demo" 或 "Command Not Avaible" 或 "Can ’t save in Shareware/Demo"等 （我们希望把它跳过，不让它出现）。。。。。。。 xxxxxxxxxxx 正确路线所在 2.修改为nop je(jne,jz,jnz) =>nop相应的机器码90 （正确信息向上找到的第一个跳转） nop的作用是抹掉这个跳转，使这个跳转无效，失去作用，从而使程序顺利来到紧跟其后的正确信息处 xxxxxxxxxxx 正确信息，例如：注册成功，谢谢您的支持等（我们希望它不被跳过，让它出现，程序一定要顺利来到这里）。。。。。。。 xxxxxxxxxxx 出错信息（我们希望不要跳到这里，不让它出现）它们在存贮器和寄存器、寄存器和输入输出端口之间传送数据. 1. 通用数据传送指令. MOV 传送字或字节. MOVSX 先符号扩展,再传送. MOVZX 先零扩展,再传送. PUSH 把字压入堆栈. POP 把字弹出堆栈. PUSHA 把AX,CX,DX,BX,SP,BP,SI,DI依次压入堆栈. POPA 把DI,SI,BP,SP,BX,DX,CX,AX依次弹出堆栈. PUSHAD

把EAX,ECX,EDX,EBX,ESP,EBP,ESI,EDI依次压入堆栈. POPAD 把EDI,ESI,EBP,ESP,EBX,EDX,ECX,EAX依次弹出堆栈. BSWAP 交换32位寄存器里字节的顺序 XCHG 交换字或字节.( 至少有一个操作数为寄存器,段寄存器不可作为操作数) CMPXCHG 比较并交换操作数.( 第二个操作数必须为累加器AL/AX/EAX ) XADD 先交换再累加.( 结果在第一个操作数里 ) XLAT 字节查表转换. —— BX 指向一张 256 字节的表的起点, AL 为表的索引值 (0-255,即 0-FFH). 返回 AL 为查表结果. ( [BX AL]->AL ) 2. 输入输出端口传送指令. IN I/O端口输入. ( 语法: IN 累加器, {端口号 | DX} ) OUT I/O端口输出. ( 语法: OUT {端口号 | DX}, 累加器 ) 输入输出端口由立即方式指定时, 其范围是 0-255. 由寄存器 DX 指定时, 其范围是 0-65535. 3. 目的地址传送指令. LEA 装入有效地址. 