[**动态加载EXE文件到内存执行**](http://blog.csdn.net/zzz3265/article/details/1824662)

http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// loadEXE.cpp : Defines the entry point for the console application.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// Proof-Of-Concept Code  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// Copyright (c) 2004  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// All rights reserved.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// Permission is hereby granted, free of charge, to any person obtaining a  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// copy of this software and associated documentation files (the  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// "Software"), to deal in the Software without restriction, including  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// without limitation the rights to use, copy, modify, merge, publish,  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// distribute, and/or sell copies of the Software, and to permit persons  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// to whom the Software is furnished to do so, provided that the above  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// copyright notice(s) and this permission notice appear in all copies of  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// the Software and that both the above copyright notice(s) and this  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// permission notice appear in supporting documentation.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT,  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// Usage:  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// loadEXE <EXE filename>  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// This will execute calc.exe in suspended mode and replace its image with  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// the new EXE's image.  The thread is then resumed, thus causing the new EXE to  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// execute within the process space of svchost.exe.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif#include <stdio.h>  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif#include <windows.h>  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif#include <tlhelp32.h>  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif#include <psapi.h>  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifstruct PE\_Header   
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long signature;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short machine;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short numSections;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long timeDateStamp;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long pointerToSymbolTable;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long numOfSymbols;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short sizeOfOptionHeader;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short characteristics;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif};  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifstruct PE\_ExtHeader  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short magic;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned char majorLinkerVersion;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned char minorLinkerVersion;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sizeOfCode;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sizeOfInitializedData;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sizeOfUninitializedData;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long addressOfEntryPoint;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long baseOfCode;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long baseOfData;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long imageBase;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sectionAlignment;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long fileAlignment;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short majorOSVersion;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short minorOSVersion;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short majorImageVersion;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short minorImageVersion;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short majorSubsystemVersion;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short minorSubsystemVersion;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long reserved1;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sizeOfImage;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sizeOfHeaders;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long checksum;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short subsystem;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short DLLCharacteristics;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sizeOfStackReserve;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sizeOfStackCommit;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sizeOfHeapReserve;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sizeOfHeapCommit;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long loaderFlags;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long numberOfRVAAndSizes;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long exportTableAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long exportTableSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long importTableAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long importTableSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long resourceTableAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long resourceTableSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long exceptionTableAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long exceptionTableSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long certFilePointer;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long certTableSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long relocationTableAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long relocationTableSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long debugDataAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long debugDataSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long archDataAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long archDataSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long globalPtrAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long globalPtrSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long TLSTableAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long TLSTableSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long loadConfigTableAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long loadConfigTableSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long boundImportTableAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long boundImportTableSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long importAddressTableAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long importAddressTableSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long delayImportDescAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long delayImportDescSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long COMHeaderAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long COMHeaderSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long reserved2;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long reserved3;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif};  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifstruct SectionHeader  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned char sectionName[8];  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long virtualSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long virtualAddress;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long sizeOfRawData;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long pointerToRawData;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long pointerToRelocations;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long pointerToLineNumbers;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short numberOfRelocations;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short numberOfLineNumbers;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long characteristics;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif};  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifstruct MZHeader  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short signature;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short partPag;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short pageCnt;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short reloCnt;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short hdrSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short minMem;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short maxMem;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short reloSS;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short exeSP;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short chksum;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short exeIP;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short reloCS;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short tablOff;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned short overlay;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned char reserved[32];  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long offsetToPE;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif};  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifstruct ImportDirEntry  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    DWORD importLookupTable;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    DWORD timeDateStamp;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    DWORD fowarderChain;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    DWORD nameRVA;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    DWORD importAddressTable;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif};  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// This function reads the MZ, PE, PE extended and Section Headers from an EXE file.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifbool readPEInfo(FILE \*fp, MZHeader \*outMZ, PE\_Header \*outPE, PE\_ExtHeader \*outpeXH,  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif                SectionHeader \*\*outSecHdr)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    fseek(fp, 0, SEEK\_END);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    long fileSize = ftell(fp);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    fseek(fp, 0, SEEK\_SET);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(fileSize < sizeof(MZHeader))  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("File size too small ");          
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        return false;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    // read MZ Header  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    MZHeader mzH;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    fread(&mzH, sizeof(MZHeader), 1, fp);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(mzH.signature != 0x5a4d)        // MZ  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("File does not have MZ header ");  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        return false;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    //printf("Offset to PE Header = %X ", mzH.offsetToPE);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if((unsigned long)fileSize < mzH.offsetToPE + sizeof(PE\_Header))  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("File size too small ");          
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        return false;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    // read PE Header  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    fseek(fp, mzH.offsetToPE, SEEK\_SET);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    PE\_Header peH;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    fread(&peH, sizeof(PE\_Header), 1, fp);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    //printf("Size of option header = %d ", peH.sizeOfOptionHeader);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    //printf("Number of sections = %d ", peH.numSections);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(peH.sizeOfOptionHeader != sizeof(PE\_ExtHeader))  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("Unexpected option header size. ");  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        return false;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    // read PE Ext Header  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    PE\_ExtHeader peXH;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    fread(&peXH, sizeof(PE\_ExtHeader), 1, fp);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    //printf("Import table address = %X ", peXH.importTableAddress);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    //printf("Import table size = %X ", peXH.importTableSize);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    //printf("Import address table address = %X ", peXH.importAddressTableAddress);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    //printf("Import address table size = %X ", peXH.importAddressTableSize);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    // read the sections  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    SectionHeader \*secHdr = new SectionHeader[peH.numSections];  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    fread(secHdr, sizeof(SectionHeader) \* peH.numSections, 1, fp);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    \*outMZ = mzH;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    \*outPE = peH;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    \*outpeXH = peXH;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    \*outSecHdr = secHdr;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    return true;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif}  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// This function calculates the size required to load an EXE into memory with proper alignment.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifint calcTotalImageSize(MZHeader \*inMZ, PE\_Header \*inPE, PE\_ExtHeader \*inpeXH,  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif                       SectionHeader \*inSecHdr)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    int result = 0;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    int alignment = inpeXH->sectionAlignment;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(inpeXH->sizeOfHeaders % alignment == 0)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        result += inpeXH->sizeOfHeaders;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        int val = inpeXH->sizeOfHeaders / alignment;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        val++;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        result += (val \* alignment);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    for(int i = 0; i < inPE->numSections; i++)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        if(inSecHdr[i].virtualSize)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            if(inSecHdr[i].virtualSize % alignment == 0)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                result += inSecHdr[i].virtualSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif            {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                int val = inSecHdr[i].virtualSize / alignment;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                val++;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                result += (val \* alignment);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif            }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    return result;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif}  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// This function calculates the aligned size of a section  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifunsigned long getAlignedSize(unsigned long curSize, unsigned long alignment)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{      
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(curSize % alignment == 0)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        return curSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        int val = curSize / alignment;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        val++;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        return (val \* alignment);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif}  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// This function loads a PE file into memory with proper alignment.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// Enough memory must be allocated at ptrLoc.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifbool loadPE(FILE \*fp, MZHeader \*inMZ, PE\_Header \*inPE, PE\_ExtHeader \*inpeXH,  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif            SectionHeader \*inSecHdr, LPVOID ptrLoc)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    char \*outPtr = (char \*)ptrLoc;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    fseek(fp, 0, SEEK\_SET);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long headerSize = inpeXH->sizeOfHeaders;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    // certain PE files have sectionHeaderSize value > size of PE file itself.    
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    // this loop handles this situation by find the section that is nearest to the  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    // PE header.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    for(int i = 0; i < inPE->numSections; i++)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        if(inSecHdr[i].pointerToRawData < headerSize)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            headerSize = inSecHdr[i].pointerToRawData;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    // read the PE header  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long readSize = fread(outPtr, 1, headerSize, fp);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    //printf("HeaderSize = %d ", headerSize);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(readSize != headerSize)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("Error reading headers (%d %d) ", readSize, headerSize);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        return false;          
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    outPtr += getAlignedSize(inpeXH->sizeOfHeaders, inpeXH->sectionAlignment);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    // read the sections  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    for(i = 0; i < inPE->numSections; i++)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        if(inSecHdr[i].sizeOfRawData > 0)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            unsigned long toRead = inSecHdr[i].sizeOfRawData;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            if(toRead > inSecHdr[i].virtualSize)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                toRead = inSecHdr[i].virtualSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            fseek(fp, inSecHdr[i].pointerToRawData, SEEK\_SET);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            readSize = fread(outPtr, 1, toRead, fp);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            if(readSize != toRead)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif            {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                printf("Error reading section %d ", i);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                return false;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif            }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            outPtr += getAlignedSize(inSecHdr[i].virtualSize, inpeXH->sectionAlignment);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            // this handles the case where the PE file has an empty section. E.g. UPX0 section  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            // in UPXed files.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            if(inSecHdr[i].virtualSize)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                outPtr += getAlignedSize(inSecHdr[i].virtualSize, inpeXH->sectionAlignment);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    return true;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif}  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifstruct FixupBlock  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long pageRVA;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    unsigned long blockSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif};  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// This function loads a PE file into memory with proper alignment.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// Enough memory must be allocated at ptrLoc.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifvoid doRelocation(MZHeader \*inMZ, PE\_Header \*inPE, PE\_ExtHeader \*inpeXH,  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif                  SectionHeader \*inSecHdr, LPVOID ptrLoc, DWORD newBase)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(inpeXH->relocationTableAddress && inpeXH->relocationTableSize)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        FixupBlock \*fixBlk = (FixupBlock \*)((char \*)ptrLoc + inpeXH->relocationTableAddress);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        long delta = newBase - inpeXH->imageBase;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        while(fixBlk->blockSize)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            //printf("Addr = %X ", fixBlk->pageRVA);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            //printf("Size = %X ", fixBlk->blockSize);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            int numEntries = (fixBlk->blockSize - sizeof(FixupBlock)) >> 1;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            //printf("Num Entries = %d ", numEntries);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            unsigned short \*offsetPtr = (unsigned short \*)(fixBlk + 1);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            for(int i = 0; i < numEntries; i++)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif            {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                DWORD \*codeLoc = (DWORD \*)((char \*)ptrLoc + fixBlk->pageRVA + (\*offsetPtr & 0x0FFF));  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                int relocType = (\*offsetPtr & 0xF000) >> 12;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                //printf("Val = %X ", \*offsetPtr);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                //printf("Type = %X ", relocType);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                if(relocType == 3)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                    \*codeLoc = ((DWORD)\*codeLoc) + delta;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif                {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                    printf("Unknown relocation type = %d ", relocType);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif                }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                offsetPtr++;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif            }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            fixBlk = (FixupBlock \*)offsetPtr;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }      
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif}  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif#define TARGETPROC "calc.exe"  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifCHAR    szDefDir[] = "I:/MyProject/tools/Xptools";  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//CHAR    szDeft[] = "E:/WINDOWS/system32/notepad.exe";  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//CHAR    szDeft[] = "I:/MyProject/tools/Xptools/XPtools.exe";  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//CHAR    szDeft[] = "I:/MyProject/Test/Calcu/debug/Calcu.exe";  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifCHAR    szDeft[] = "I:/MyProject/PCBTest/Bin/PCBTest.exe";  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.giftypedef struct \_PROCINFO  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    DWORD baseAddr;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    DWORD imageSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif} PROCINFO;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// Creates the original EXE in suspended mode and returns its info in the PROCINFO structure.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifBOOL createChild(PPROCESS\_INFORMATION pi, PCONTEXT ctx, PROCINFO \*outChildProcInfo)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    STARTUPINFO si = {0};  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(CreateProcess(NULL, TARGETPROC,  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        NULL, NULL, 0, CREATE\_SUSPENDED, NULL, szDefDir, &si, pi))          
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        ctx->ContextFlags=CONTEXT\_FULL;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        GetThreadContext(pi->hThread, ctx);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        DWORD \*pebInfo = (DWORD \*)ctx->Ebx;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        DWORD read;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        ReadProcessMemory(pi->hProcess, &pebInfo[2], (LPVOID)&(outChildProcInfo->baseAddr), sizeof(DWORD), &read);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        DWORD curAddr = outChildProcInfo->baseAddr;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        MEMORY\_BASIC\_INFORMATION memInfo;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        while(VirtualQueryEx(pi->hProcess, (LPVOID)curAddr, &memInfo, sizeof(memInfo)))  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            if(memInfo.State == MEM\_FREE)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                break;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            curAddr += memInfo.RegionSize;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        outChildProcInfo->imageSize = (DWORD)curAddr - (DWORD)outChildProcInfo->baseAddr;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        return TRUE;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    return FALSE;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif}  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// Returns true if the PE file has a relocation table  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifBOOL hasRelocationTable(PE\_ExtHeader \*inpeXH)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(inpeXH->relocationTableAddress && inpeXH->relocationTableSize)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        return TRUE;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    return FALSE;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif}  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.giftypedef DWORD (WINAPI \*PTRZwUnmapViewOfSection)(IN HANDLE ProcessHandle, IN PVOID BaseAddress);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// To replace the original EXE with another one we do the following.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// 1) Create the original EXE process in suspended mode.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// 2) Unmap the image of the original EXE.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// 3) Allocate memory at the baseaddress of the new EXE.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// 4) Load the new EXE image into the allocated memory.    
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// 5) Windows will do the necessary imports and load the required DLLs for us when we resume the suspended   
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//    thread.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// When the original EXE process is created in suspend mode, GetThreadContext returns these useful  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// register values.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// EAX - process entry point  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// EBX - points to PEB  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// So before resuming the suspended thread, we need to set EAX of the context to the entry point of the  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif// new EXE.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifvoid doFork(MZHeader \*inMZ, PE\_Header \*inPE, PE\_ExtHeader \*inpeXH,  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif            SectionHeader \*inSecHdr, LPVOID ptrLoc, DWORD imageSize)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    STARTUPINFO si = {0};  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    PROCESS\_INFORMATION pi;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    CONTEXT ctx;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    PROCINFO childInfo;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(createChild(&pi, &ctx, &childInfo))   
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {          
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("Original EXE loaded (PID = %d). ", pi.dwProcessId);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("Original Base Addr = %X, Size = %X ", childInfo.baseAddr, childInfo.imageSize);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        LPVOID v = (LPVOID)NULL;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        if(inpeXH->imageBase == childInfo.baseAddr && imageSize <= childInfo.imageSize)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            // if new EXE has same baseaddr and is its size is <= to the original EXE, just  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            // overwrite it in memory  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            v = (LPVOID)childInfo.baseAddr;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            DWORD oldProtect;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            VirtualProtectEx(pi.hProcess, (LPVOID)childInfo.baseAddr, childInfo.imageSize, PAGE\_EXECUTE\_READWRITE, &oldProtect);              
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            printf("Using Existing Mem for New EXE at %X ", (unsigned long)v);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            // get address of ZwUnmapViewOfSection  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            PTRZwUnmapViewOfSection pZwUnmapViewOfSection = (PTRZwUnmapViewOfSection)GetProcAddress(GetModuleHandle("ntdll.dll"), "ZwUnmapViewOfSection");  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            // try to unmap the original EXE image  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            if(pZwUnmapViewOfSection(pi.hProcess, (LPVOID)childInfo.baseAddr) == 0)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif            {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                // allocate memory for the new EXE image at the prefered imagebase.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                v = VirtualAllocEx(pi.hProcess, (LPVOID)inpeXH->imageBase, imageSize, MEM\_RESERVE | MEM\_COMMIT, PAGE\_EXECUTE\_READWRITE);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                if(v)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                    printf("Unmapped and Allocated Mem for New EXE at %X ", (unsigned long)v);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif            }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        if(!v && hasRelocationTable(inpeXH))  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            // if unmap failed but EXE is relocatable, then we try to load the EXE at another  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            // location  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            v = VirtualAllocEx(pi.hProcess, (void \*)NULL, imageSize, MEM\_RESERVE | MEM\_COMMIT, PAGE\_EXECUTE\_READWRITE);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            if(v)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif            {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                printf("Allocated Mem for New EXE at %X. EXE will be relocated. ", (unsigned long)v);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                // we've got to do the relocation ourself if we load the image at another  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                // memory location                  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                doRelocation(inMZ, inPE, inpeXH, inSecHdr, ptrLoc, (DWORD)v);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif            }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("EIP = %X ", ctx.Eip);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("EAX = %X ", ctx.Eax);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("EBX = %X ", ctx.Ebx);        // EBX points to PEB  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("ECX = %X ", ctx.Ecx);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("EDX = %X ", ctx.Edx);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        if(v)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {              
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            printf("New EXE Image Size = %X ", imageSize);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            // patch the EXE base addr in PEB (PEB + 8 holds process base addr)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            DWORD \*pebInfo = (DWORD \*)ctx.Ebx;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            DWORD wrote;                          
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            WriteProcessMemory(pi.hProcess, &pebInfo[2], &v, sizeof(DWORD), &wrote);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            // patch the base addr in the PE header of the EXE that we load ourselves  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            PE\_ExtHeader \*peXH = (PE\_ExtHeader \*)((DWORD)inMZ->offsetToPE + sizeof(PE\_Header) + (DWORD)ptrLoc);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            peXH->imageBase = (DWORD)v;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            if(WriteProcessMemory(pi.hProcess, v, ptrLoc, imageSize, NULL))  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif            {      
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                printf("New EXE image injected into process. ");  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                ctx.ContextFlags=CONTEXT\_FULL;                  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                //ctx.Eip = (DWORD)v + ((DWORD)dllLoaderWritePtr - (DWORD)ptrLoc);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                //ctx.Eip = 0x007B9640;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                if((DWORD)v == childInfo.baseAddr)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif                {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                    ctx.Eax = (DWORD)inpeXH->imageBase + inpeXH->addressOfEntryPoint;        // eax holds new entry point  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif                }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif                {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                    // in this case, the DLL was not loaded at the baseaddr, i.e. manual relocation was  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                    // performed.  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                    ctx.Eax = (DWORD)v + inpeXH->addressOfEntryPoint;        // eax holds new entry point  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif                }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                printf("\*\*\*\*\*\*\*\*> EIP = %X ", ctx.Eip);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                printf("\*\*\*\*\*\*\*\*> EAX = %X ", ctx.Eax);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                SetThreadContext(pi.hThread,&ctx);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                ResumeThread(pi.hThread);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                printf("Process resumed (PID = %d). ", pi.dwProcessId);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif            }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif            {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                printf("WriteProcessMemory failed ");  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                TerminateProcess(pi.hProcess, 0);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif            }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            printf("Load failed.  Consider making this EXE relocatable. ");  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            TerminateProcess(pi.hProcess, 0);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("Cannot load %s ", TARGETPROC);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif}  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifint main(int argc, char\* argv[])  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockStart.gif{  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    POINT        pt;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    GetCursorPos(&pt);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    FILE \*fp;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(argc != 2)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf(" Usage: %s <EXE filename> ", argv[0]);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf("No Input EXE, use: %s ", szDeft);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        fp = fopen(szDeft, "rb");  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        fp = fopen(argv[1], "rb");  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    if(fp)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif    {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        MZHeader mzH;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        PE\_Header peH;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        PE\_ExtHeader peXH;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        SectionHeader \*secHdr;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        if(readPEInfo(fp, &mzH, &peH, &peXH, &secHdr))  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif        {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            int imageSize = calcTotalImageSize(&mzH, &peH, &peXH, secHdr);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            //printf("Image Size = %X ", imageSize);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            LPVOID ptrLoc = VirtualAlloc(NULL, imageSize, MEM\_COMMIT, PAGE\_EXECUTE\_READWRITE);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            if(ptrLoc)  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockStart.gif            {  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                //printf("Memory allocated at %X ", ptrLoc);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                loadPE(fp, &mzH, &peH, &peXH, secHdr, ptrLoc);                                                  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                doFork(&mzH, &peH, &peXH, secHdr, ptrLoc, imageSize);                                  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif            }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif            else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif                printf("Allocation failed ");  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif        }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        fclose(fp);  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedSubBlockEnd.gif    }  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    else  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif        printf(" Cannot open the EXE file! ");  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/InBlock.gif    return 0;  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/ExpandedBlockEnd.gif}