第一步，要注入的DLL源代码一份,创建步骤打开VC6.0后，选“文件”----“新建”----在弹出的对话框中选”Win32 Dynamic-Link Library”创建一个动态库程序，选择空，然后自己添加源文件PrintMsg.cpp并将代码复制到其中，同样的添加头文件也将源代码复制到其中。  
  
  
源代码如下：  
  
PrintMsg.h头文件  
  
#ifndef \_PRINTMSG\_H\_  
#define \_PRINTMSG\_H\_  
  
#include "PrintMsg.h"  
  
#endif //\_PRINTMSG\_H\_  
  
  
  
  
  
PrintMsg.cpp源文件  
  
#include "stdio.h"  
#include "windows.h"  
#include "PrintMsg.h"  
  
extern "C" \_declspec(dllexport)   
  
void PrintMsg( )  
{  
  
         while(1)  
         {  
                   printf( "恭喜您成功了\n\n" );  
                   Sleep(1000);  
         }  
}  
  
  
  
  
  
完成后直接编译生成PrintMsg.dll文件。第一个工作就完成了。 Easy吧！  
  
  
  
  
  
  
第二步：编写主要程序，用来向远程进程注入DLL并远程创建线程运行注入的DLL函数，  
  
代码为Win32的控制台程序，创建好以后直接将如下源代码复制到源文件当中。  
  
(注意：在测试的时候要配合任务管理器查看要注入进程的ID号，然后手工添加ID号后再运行CreateRemoteDll程序)  
  
  
  
源代码如下:  
  
CreateRemoteDll.cpp源文件  
  
  
  
// CreateRemoteDll.cpp : Defines the entry point for the console application.  
  
//  
  
  
#include "stdafx.h"  
#include "windows.h"  
  
BOOL CreateRemoteDll(const char \*DllFullPath, const DWORD dwRemoteProcessId)  
{  
         HANDLE hToken;  
         if ( OpenProcessToken(GetCurrentProcess(),TOKEN\_ADJUST\_PRIVILEGES,&hToken) )  
         {  
                   TOKEN\_PRIVILEGES tkp;  
                   LookupPrivilegeValue( NULL,SE\_DEBUG\_NAME,&tkp.Privileges[0].Luid );//修改进程权限  
                   tkp.PrivilegeCount=1;  
                   tkp.Privileges[0].Attributes=SE\_PRIVILEGE\_ENABLED;  
                   AdjustTokenPrivileges( hToken,FALSE,&tkp,sizeof tkp,NULL,NULL );//通知系统修改进程权限    
         }  
         HANDLE hRemoteProcess;  
  
         //打开远程线程  
         if( (hRemoteProcess = OpenProcess( PROCESS\_CREATE\_THREAD |    //允许远程创建线程  
  
                   PROCESS\_VM\_OPERATION |                //允许远程VM操作  
  
                   PROCESS\_VM\_WRITE,                    //允许远程VM写  
  
                   FALSE, dwRemoteProcessId ) )== NULL )  
  
         {  
                   MessageBox( NULL, "打开指定进程失败!", "错误提示", MB\_OK );  
                   return FALSE;  
         }  
  
           
  
         char \*pszLibFileRemote;  
  
         //在远程进程的内存地址空间分配DLL文件名缓冲区  
  
         pszLibFileRemote = (char \*) VirtualAllocEx( hRemoteProcess, NULL, lstrlen(DllFullPath)+1,   
  
                   MEM\_COMMIT, PAGE\_READWRITE);  
  
         if(pszLibFileRemote == NULL)  
         {  
                   MessageBox( NULL, "在远程进程的内存地址空间分配DLL文件名缓冲区失败! ", "Error", MB\_OK );  
                   return FALSE;  
         }  
  
           
  
         //将DLL的路径名复制到远程进程的内存空间  
         if( WriteProcessMemory(hRemoteProcess,  
  
                   pszLibFileRemote, (void \*) DllFullPath, lstrlen(DllFullPath)+1, NULL) == 0)  
  
         {  
                   MessageBox( NULL, "将DLL的路径名复制到远程进程的内存空间失败", "Error", MB\_OK );  
                   return FALSE;  
         }  
  
           
  
         //计算LoadLibraryA的入口地址  
         PTHREAD\_START\_ROUTINE pfnStartAddr = (PTHREAD\_START\_ROUTINE)  
  
                   GetProcAddress(GetModuleHandle(TEXT("Kernel32")), "LoadLibraryA");  
  
         if(pfnStartAddr == NULL)  
         {  
                   MessageBox( NULL, "获取函数LoadLibraryA的地址失败", "Error", MB\_OK );  
                   return FALSE;  
         }  
  
           
  
         HANDLE hRemoteThread;  
  
         if( (hRemoteThread = CreateRemoteThread( hRemoteProcess, NULL, 0,   
  
                   pfnStartAddr, pszLibFileRemote, 0, NULL) ) == NULL)  
  
         {  
  
                   MessageBox( NULL, "创建远程线程失败", "Error", MB\_OK );  
  
                   return FALSE;  
  
         }  
  
           
  
         //---------------------------------------------------------------------  
  
         //在远程创建的线程中运行加载的DLL中的函数  
  
         HMODULE hPrint = LoadLibraryA( "PrintMsg.dll" );  
  
         pfnStartAddr = (PTHREAD\_START\_ROUTINE)  
  
                   GetProcAddress(GetModuleHandle(TEXT("PrintMsg")), "PrintMsg");  
  
           
  
         if(pfnStartAddr == NULL)  
  
         {  
  
                   MessageBox( NULL, "获取函数PrintMsg的地址失败", "Error", MB\_OK );  
  
                   return FALSE;  
  
         }  
  
  
  
         if( (hRemoteThread = CreateRemoteThread( hRemoteProcess, NULL, 0,   
  
                   pfnStartAddr, pszLibFileRemote, 0, NULL) ) == NULL)  
  
         {  
  
                   MessageBox( NULL, "创建远程线程PrintMsg失败", "