1 

新建授权成功，还需要建立默认的系统策略

SQL语句

**StrategyTable**

SQLServerStatement\* stmt=g\_smSqlite.Statement(

TEXT("insert into StrategyTable values (?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?, \

?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)"));

stmt->Bind(0,CSIP.GetBuffer());

//stmt->Bind(0,L"gogo");

stmt->Bind(1,L"系统默认策略");

stmt->Bind(2,STRATEGY\_OPERATE\_SYSTEM);

stmt->Bind(3,L"Windows 2008 Server完整性监测");

stmt->Bind(4,STRATEGY\_WINDOW\_FILE | STRATEGY\_WINDOW\_REGISTRY);

stmt->Bind(38,CSIP.GetBuffer());

stmt->Bind(5,0);

stmt->Bind(6,0);

stmt->Bind(7,TEXT(""));

stmt->Bind(8,TEXT(""));

stmt->Bind(9,TEXT(""));

stmt->Bind(10,TEXT(""));

stmt->Bind(11,0);

stmt->Bind(12,TEXT(""));

stmt->Bind(41,TEXT(""));

stmt->Bind(13,0);

stmt->Bind(14,1);

stmt->Bind(15,1);

stmt->Bind(16,0);

stmt->Bind(17,0);

stmt->Bind(18,0);

stmt->Bind(19,0);

stmt->Bind(20,0);

stmt->Bind(21,0);

stmt->Bind(22,0);

stmt->Bind(23,0);

stmt->Bind(24,StrategyInfo.IsAutoDelete);

stmt->Bind(25,StrategyInfo.IsStopTask);

stmt->Bind(26,StrategyInfo.NumOfHours);

stmt->Bind(27,StrategyInfo.NumOfMunites);

stmt->Bind(28,1);

stmt->Bind(29,StrategyInfo.IsHistoryCopy);

stmt->Bind(30,StrategyInfo.IsSavePlace);

stmt->Bind(31,StrategyInfo.IsReport);

stmt->Bind(32,StrategyInfo.IsJiXianData);

stmt->Bind(33,1);

stmt->Bind(34,CREATE\_FILE | CREATE\_FOLDER |ACCESS\_WRITE |ACCESS\_DELETE| ACCESS\_RENAME);

stmt->Bind(35,0);

stmt->Bind(36,L"D:\\StrBackup");

stmt->Bind(37,L"natogo@gcexe.com");

stmt->Bind(39,1);

stmt->Bind(40,StrategyInfo.StrategyID);

stmt->Bind(42,0);

stmt->Execute() ;

g\_smSqlite.CloseQuery(stmt);

wcscpy\_s(strategyElement.StrategyName,64,CSIP.GetBuffer());

strategyElement.StrategyType = 0;

StrategyInfo.TaskMode =0;

wcscpy\_s(strategyElement.SelectedItem,32, L"Windows 2008 Server完整性监测(64)");

strategyElement.ItemAttribute =STRATEGY\_WINDOW\_FILE | STRATEGY\_WINDOW\_REGISTRY;

wcscpy\_s(strategyElement.LastTime, 32, L"");

strategyElement.StrategyStatus = TRUE;

switch(StrategyInfo.TaskMode)

{

case EVERYDAY:

{

strategyElement.TaskMode = EVERYDAY;

strategyElement.taskset.day.NumOfDay = StrategyInfo.taskset.day.NumOfDay;

strategyElement.taskset.day.StartTime = StrategyInfo.taskset.day.StartTime;

}

break;

case WEEKLY:

{

strategyElement.TaskMode = WEEKLY;

strategyElement.taskset.week.NumOfWeek = StrategyInfo.taskset.week.NumOfWeek;

strategyElement.taskset.week.StartTime = StrategyInfo.taskset.week.StartTime;

strategyElement.taskset.week.WhichWeek = StrategyInfo.taskset.week.WhichWeek;

}

break;

case PERMONTH:

{

strategyElement.TaskMode = PERMONTH;

strategyElement.taskset.month.DayOfMonth = StrategyInfo.taskset.month.DayOfMonth;

strategyElement.taskset.month.StartTime = StrategyInfo.taskset.month.StartTime;

strategyElement.taskset.month.WhichMonth = StrategyInfo.taskset.month.WhichMonth;

}

break;

}

**同时发送新增策略消息 至服务**

**g\_InterConnect.SendOperationLocal(0,SERVER\_STRATEGYGUARD\_ADD,&strategyElement,sizeof(STRATEGY\_ELEMENT));**

**1**

1. **但凡有以下这种 获取路径的，有两条交互信息**
2. **SERVER\_LIST\_DRIVE, //列出盘符**
3. **SERVER\_LIST\_FILEDIRECTORY, //列出目录下所有文件和文件夹**

附：列出盘符 C++解析代码 （消息为一XML 格式BUFFER）

LRESULT CDirectoryDlg::OnListDrive(UINT uMsg, WPARAM wParam, LPARAM lParam)

{

DWORD dwDriveMap;

DWORD dwDriveCount;

CString csItemName;

TiXmlDocument doc;

TiXmlElement\* RootElement;

TiXmlElement\* DirElement;

TiXmlAttribute\* dirattr;

LPCWSTR pBuffer = NULL;

PMSGINFO nMsgInfo = (PMSGINFO)wParam;

pBuffer = (LPCWSTR)::HeapAlloc(::GetProcessHeap(), 0, nMsgInfo->dwBufferLen);

if (NULL == pBuffer)

{

CString csMsg;

int dwRet = ::GetLastError();

::GetErrorMessage(dwRet, csMsg);

IGMessageBox(csMsg, TEXT("严重错误"), MB\_OK|MB\_ICONERROR);

return NULL;

}

RtlCopyMemory((WCHAR \*)pBuffer,(WCHAR\*)lParam,nMsgInfo->dwBufferLen);

//doc.Parse((PSTR)lParam, NULL, TIXML\_DEFAULT\_ENCODING

USES\_CONVERSION;

doc.Parse(W2A(pBuffer), NULL, TIXML\_DEFAULT\_ENCODING);

RootElement = doc.RootElement();

DirElement = RootElement->FirstChildElement();

if (NULL != DirElement)

{

dirattr = DirElement->FirstAttribute();

dwDriveMap = dirattr->IntValue();

dwDriveCount = 0;

while (dwDriveMap)

{

if (dwDriveMap & 1)

{

csItemName.Format(TEXT("本地磁盘(%c:)"), 'A'+ dwDriveCount);

//AddItem(csItemName, RULE\_TYPE\_DRIVER, m\_hRootItem, TRUE);

AddItem(csItemName, TREEITEM\_DRIVE, m\_hRootItem, TRUE);

}

dwDriveMap >>= 1;

dwDriveCount ++;

}

}

::HeapFree(::GetProcessHeap(), 0, (LPVOID)pBuffer);

return TRUE;

}

附**列出目录下所有文件和文件夹 代码**

LRESULT CFileDlg::OnListFile(UINT uMsg, WPARAM wParam, LPARAM lParam)

{

BOOL blIsDir;

BOOL blHasChild;

DWORD dwItemType;

CString csItemName;

TiXmlDocument doc;

TiXmlElement\* RootElement;

TiXmlElement\* DirElement;

TiXmlAttribute\* dirattr;

doc.Parse((PSTR)lParam, NULL, TIXML\_ENCODING\_UTF8);

RootElement = doc.RootElement();

DirElement = RootElement->FirstChildElement();

USES\_CONVERSION;

while (NULL != DirElement)

{

dirattr = DirElement->FirstAttribute();

csItemName = A2W(dirattr->Value());

dirattr = dirattr->Next();

blIsDir = dirattr->IntValue();

if (TRUE == blIsDir)

{

dwItemType = TREEITEM\_FILE;

}

else

{

dwItemType = TREEITEM\_DIRECOTRY;

}

dirattr = dirattr->Next();

blHasChild = dirattr->IntValue();

AddItem(csItemName, dwItemType, m\_hCurrentExpandedItem, blHasChild);

DirElement = DirElement->NextSiblingElement();

}

return 0;

}

