快速通道

文档分类

- 红盟公告
- 新手上路
- 漏洞分析
- 工具介绍
- 安全配置
- **编程技术 <=**
- 安全技术
- 业界动态

本类热门

- 初探数据包分析程序设计
- 杀毒软件的简单实现
- readasm的使用范例
- 半开扫描程序
- 注册表操作（RingZ_RgeEdit）
- proftpd 1.2.7/1.2.9rc2 remote root exploit

杀毒软件的简单实现

提交文章： dahubaobao
提交日期： 2004-06-16
文章属性： 原创
文档类别： 编程技术
相关下载： **=>下载<=**
浏览次数： 今7次/总8631次

```
#include "FunDef.h"

int main (int argc, char *argv[])
{
if (argc==1)
{
Usage(argv[0]);
return 0;
}

if (!(ScanFileVXER(argv[1])))
{
printf("ScanFileVXER() GetLastError reports %d\n",erron);
return 0;
}

if (!(ProcessVXER()))
{
printf("Processes() GetLastError reports %d\n",erron);
return 0;
}

if (!(RegDelVXER()))
{
printf("RegDelVXER() GetLastError reports %d\n",erron);
return 0;
}

return 0;
}

BOOL ScanFileVXER (char *FileName)
{
int count=LOW;

WIN32_FIND_DATA FindFileData;
HANDLE hFind;
BOOL returnvalue=FALSE;
DWORD lpBufferLength=HIGH;
char lpBuffer[HIGH]={LOW};
char DirBuffer[MAX_PATH];
```

```
long FileOffset=0x1784; //偏移地址
int FileLength=0x77; //长度

unsigned char Contents[]={
0x49, 0x20, 0x6A, 0x75, 0x73, 0x74, 0x20, 0x77, 0x61, 0x6E, 0x74, 0x20, 0x74, 0x6F, 0x2
0x61, 0x79, 0x20, 0x4C, 0x4F, 0x56, 0x45, 0x20, 0x59, 0x4F, 0x55, 0x20, 0x53, 0x41, 0x4
0x21, 0x20, 0x62, 0x69, 0x6C, 0x6C, 0x79, 0x20, 0x67, 0x61, 0x74, 0x65, 0x73, 0x20, 0x7
0x79, 0x20, 0x64, 0x6F, 0x20, 0x79, 0x6F, 0x75, 0x20, 0x6D, 0x61, 0x6B, 0x65, 0x20, 0x7
0x69, 0x73, 0x20, 0x70, 0x6F, 0x73, 0x73, 0x69, 0x62, 0x6C, 0x65, 0x20, 0x3F, 0x20, 0x5
0x6F, 0x70, 0x20, 0x6D, 0x61, 0x6B, 0x69, 0x6E, 0x67, 0x20, 0x6D, 0x6F, 0x6E, 0x65, 0x
0x61, 0x6E, 0x64, 0x20, 0x66, 0x69, 0x78, 0x20, 0x79, 0x6F, 0x75, 0x72, 0x20, 0x73, 0x6
0x74, 0x77, 0x61, 0x72, 0x65, 0x21, 0x21};
//具体内容，十六进制

//获取系统目录的完整路径
if (GetSystemDirectory(DirBuffer,lpBufferLength)!=LOW)
{
if (SetCurrentDirectory(DirBuffer)!=LOW) //设置为当前目录
{
hFind=FindFirstFile(FileName,&FindFileData); //查找文件
if (hFind==INVALID_HANDLE_VALUE)
{
printf("FindFirstFile() GetLastError reports %d\n",erron);
FindClose(hFind);
return returnvalue;
}
else
{
count++;

//获得文件的完整路径
if (GetFullPathName(FindFileData.cFileName,lpBufferLength,lpBuffer,NULL)!=LOW)
printf("FilePath:%s\n",lpBuffer);
else
{
printf("GetFullPathName() GetLastError reports %d\n",erron);
FindClose(hFind);
return returnvalue;
}
}

//进行特征码的匹配工作
ScanVXER(FindFileData.cFileName,FileOffset,FileLength,Contents);
}
}

while (FindNextFile(hFind,&FindFileData)) //继续查找文件
{
count++;

//以"."和".."除外
if (strcmp(".",FindFileData.cFileName)==LOW||strcmp("..",FindFileData.cFileName)==LOW
{
printf("File no include \".\" and \"..\"\n");
exit(0);
}

if (GetFullPathName(FindFileData.cFileName,lpBufferLength,lpBuffer,NULL)!=LOW)
printf("Next FilePath:%s\n",lpBuffer);
else
{
printf("GetFullPathName() GetLastError reports %d\n",erron);
FindClose(hFind);
exit(0);
}

ScanVXER(FindFileData.cFileName,FileOffset,FileLength,Contents);

}

printf("File Total:%d\n",count); //打印出查找到的文件各数
```

```
FindClose(hFind); //关闭搜索句柄
returnvalue=TRUE;
return returnvalue;
}

BOOL ScanVXER (
char *V_FileName, //文件名
long V_FileOffset, //偏移地址
int V_Length, //长度
void *V_Contents) //具体内容
{
int cmpreturn=LOW;
char FileContents[HIGH]={LOW};
BOOL returnvalue=FALSE;
FILE *fp=NULL;

fp=fopen(V_FileName,"rb"); //以二进制只读方式打开
if (fp==NULL)
{
printf("File open FAIL\n");
fclose(fp);
return returnvalue;
}

fseek(fp,V_FileOffset,SEEK_SET); //把文件指针指向特征码在文件的偏移地址处
fread(FileContents,V_Length,1,fp);//读取长度为特征码长度的内容
cmpreturn=memcmp(V_Contents,FileContents,V_Length);
//进行特征码匹配。失败返回FALSE
if (cmpreturn==LOW)
{
printf("File Match completely\n"); //打印文件匹配消息
strcpy(name,V_FileName); //将文件名保存在全局变量name中
exit(0);
}
else
returnvalue=FALSE;
}

BOOL ProcessVXER (void)
{
DWORD lpidProcess[1024],cbNeeded_1,cbNeeded_2;
HANDLE hProc;
HMODULE hMod[1024];
char ProcFile[MAX_PATH];
char FileName[FIVE]={LOW};
BOOL returnvalue=FALSE;
int Pcount=LOW;
int i;

EnablePrivilege(SE_DEBUG_NAME); //提升权限

//枚举进程
if (!(EnumProcesses(lpidProcess,sizeof(lpidProcess),&cbNeeded_1)))
{
printf("EnumProcesses() GetLastError reports %d\n",erron);
return 0;
}

for (i=LOW;i<(int)cbNeeded_1/4;i++)
{
//打开找到的第一个进程
hProc=OpenProcess(PROCESS_ALL_ACCESS,FALSE,lpidProcess[i]);
if (hProc)
{
//枚举进程模块
if (EnumProcessModules(hProc,hMod,sizeof(hMod),&cbNeeded_2))
{
//枚举进程模块文件名，包含全路径
if (GetModuleFileNameEx(hProc,hMod[0],ProcFile,sizeof(ProcFile)))
{
printf("[%5d]\t%s\n",lpidProcess[i],ProcFile); //输出进程
```

```
//可以考虑将其注释掉，这样就不会输出进程列表了
Pcount++;

strcpy(FileName,"C:\\WINNT\\system32\\");
strcat(FileName,name);//把文件名+路径复制到FileName变量中

//查找进程中是否包含FileName
if (strcmp(FileName,ProcFile)==LOW)
{
//如果包含，则杀掉。KillProc为自定义的杀进程函数
if (!(KillProc(lpidProcess[i])))
{
printf("KillProc() GetLastError reports %d\n",erron);
CloseHandle(hProc);
exit(0);
}
DeleteFile(FileName); //进程杀掉后，再将文件删除
}
}
}
}
}

CloseHandle(hProc); //关闭进程句柄
printf("\nProcess total:%d\n",Pcount); //打印进程各数
returnvalue=TRUE;
return 0;
}

BOOL KillProc (DWORD *ProcessID)
{
HANDLE hProc;
BOOL returnvalue=FALSE;

//打开由ProcessVXER传递的进程PID
hProc=OpenProcess(PROCESS_ALL_ACCESS,FALSE,ProcessID);

if (hProc)
{
//终止进程
if (!(TerminateProcess(hProc,0)))
{
printf("TerminateProcess GetLastError reports %d\n",erron);
return returnvalue;
}
}

CloseHandle(hProc);
returnvalue=TRUE;
return returnvalue;
}

BOOL EnablePrivilege(PCHAR PrivilegeName)
{
HANDLE hProc,hToken;
TOKEN_PRIVILEGES TP;
hProc=GetCurrentProcess(); //打开进程的一个伪句柄

if(!OpenProcessToken(hProc,TOKEN_ADJUST_PRIVILEGES,&hToken))
{
return FALSE;
}

if(!LookupPrivilegeValue(NULL,PrivilegeName,&TP.Privileges[0].Luid))
{
CloseHandle(hToken);
return FALSE;
}

TP.Privileges[0].Attributes=SE_PRIVILEGE_ENABLED;
TP.PrivilegeCount=1;
```

```
if(!AdjustTokenPrivileges(hToken,FALSE,&TP,sizeof(TP),0,0))
{
CloseHandle(hToken);
return FALSE;
}

CloseHandle(hToken);
return TRUE;
}

int RegDelVXER (void)
{
HKEY hkey;
DWORD ret=LOW;

//打开注册表的Run项
ret=RegOpenKeyEx(HKEY_LOCAL_MACHINE,
"SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\\",
0,
KEY_ALL_ACCESS,
&hkey);

if (!ret==ERROR_SUCCESS)
{
printf("Register Open FAIL\n");
exit(0);
}

//删除键值windows auto update。
ret==RegDeleteValue(hkey,"windows auto update");

if (ret==ERROR_SUCCESS)
printf("Success Delete\n");
else
{
printf("Delete FAIL\n");
exit(0);
}

RegCloseKey(hkey); //关闭打开的注册表项
return 1;
}

void Usage (char *Parameter)
{
char *Path="%SystemRoot%\\system32\\";

fprintf
(stderr,"============================================================
" 杀毒软件的简单实现\n"
"环境：Win2K Adv Server + Visual C++ 6.0\n"
"作者：dahubaobao\n"
"主页：www.RingZ.org\;n"
"OICQ：382690\n"
"邮件：382690@qq.com\n"
"声明：本帖由环行区（RingZ）原创，转载请注明出处，谢谢！\n\n"
"使用方法：\n"
"%s 文件名。例如：%s msblast.exe\n\n"
"注意事项：\n"
"本程序只是简单介绍杀毒软件的编写方法，所以有很多不完善的地方，包括：\n"
"1，本程序是以冲击波蠕虫做的例子\n"
"2，文件遍历只搜索了%s目录下的文件\n"
"3，本程序不能查杀冲击波变种\n\n"
"本程序只是用做代码交流，如有错误，还请多多包含！\n"
"============================================================
,Parameter,Parameter,Path);
}
```

返回上页

Copyright © 2000-2004 **HUC** All Rights Reserved.
中国红客网络技术联盟 www.cnhonker.com
webmaster@cnhonker.net