

Shell Trojan Generators / Droppers / Backdoors Notes

From observations of running the “zwshell.exe” dropper on several infected client systems we observed that the dropper creates different service entry names in the registry upon execution. The service entry name is different from the ones we filled in the creation of the dropper in the dropper/C2 application. Some of the new service names we found are called:

- Nwsapagent
- NWCWorkstation

Others seen during the incident included:

- IAS
- ASP.NET
- IPRIP
- 6to4
- Web

With a search on the Internet we've found a reference dated from 2007 of a trojan that produces the same artifacts in the registry as our testing malware clients. The reference can be found here:

<http://www.mcafee.com/threat-intelligence/malware/default.aspx?id=143837>

The malware referenced also connects to a command and control server called oandpsoftware.com. Additional reference relate to malware detected in 2009 and 2010 with similar artifacts.

Microsoft and McAfee have attributed the dropper/C2 application to Gh0st:

<http://www.microsoft.com/security/portal/Threat/Encyclopedia/Entry.aspx?Name=Backdoor%3aWin32%2fRemosh.A>

There are some notable differences between past Gh0st applications and this version (though it is worth noting that one version of the file discovered during the incident is named “ghost”); however functionality is fairly common to these types of “point and click” or (WYSIWYG) remote administration tools. Research into attribution through Trend, McAfee, and Microsoft resources is continuing.

Host-Based Signatures

Related Registry Values

- The malware variant named shelldc.dll is designed to be installed as a service named las with a DisplayName of ASP.NET Service and a Description of “Provides support for out-of-process session states for ASP.NET.”
- The malware variant named recyle64.dll is designed to be installed as a service named las with a DisplayName of ASP.NET Services and no description.
- The malware variant named hpmdp093.dll is designed to be installed as a service named 6to4 with a DisplayName of OfficeScan Support and a Description of “Enables Help and Support Center to run Officescan on this computer”
- The malware variant named ws_18.dll is designed to be installed as a service named Irip with a DisplayName of Network Management and a Description of “Provides network installation services such as Assign, Publish, and Remove”

File System Residue

- The malware binary will contain between at least 200 and 400 bytes of data at the end of the file, beyond the structure of the PE file format. This data is known as EOF data or an Overlay.
- The malware will end 128 bytes of data, each byte incrementing by one from the number 0. The following is an example from the shelldc.dll binary analyzed:

```
00052d0: 0001 0203 0405 0607 0809 0a0b 0c0d 0e0f .....
00052e0: 1011 1213 1415 1617 1819 1a1b 1c1d 1e1f .....
00052f0: 2021 2223 2425 2627 2829 2a2b 2c2d 2e2f !#$%&()'*,./
0005300: 3031 3233 3435 3637 3839 3a3b 3c3d 3e3f 0123456789:<=>?
0005310: 4041 4243 4445 4647 4849 4a4b 4c4d 4e4f @ABCDEFGHIJKLMNO
0005320: 5051 5253 5455 5657 5859 5a5b 5c5d 5e5f PQRSTUVWXYZ[\\]^_
0005330: 60
```

- The malware exports a single function named ServiceMain and imports functions from GDI32.DLL named BitBlt and GetDIBits. These imported functions are considered suspicious in a service dll.

Volatile Evidence

- The malware creates a mutex object to ensure that only one instance of the malware is running at a time. Each variant is configured with a mutex name in its configuration data block at the end of the file. The mutex used by each variant is listed below:
 - shelldc.dll – shelldc
 - recyle64.dll – NT1630
 - hpmdp093.dll – w
 - ws_18.dll – shellsa

MD5 Signatures

- The following MD5 signatures have been discovered through live response and forensic analysis:

Name	Size (Bytes)	MD5 Hash Checksum
ws_data.dll	20,753	1AA038E8AAC50CF1825739389E904B44
ws_data.dll	20,753	549dff76afc0dd9e536a6d9c4d499065

From HBGary Dump (header added by Cryptome)

ws_data.dll	20,753	665E0B1E031460FEA258E674805B6224
ws_18.dll	20,753	79bb3e12cb08240f8d37583b0aebe25d
ws_18.dll	20,753	8C4153A218BD12DB528F46FAB7B2E405
hpcui093.dll	20,753	d39eeef1c14a3349b61ff5d45dd749b3
ws_data.dll	20,888	525386053AF66358A1B938A2BA4CCF8F
recyle32.dll	21,297	0a5d9f5c6ced1f6222416cd13e4b8612
recyle64.dll	21,297	378D6016D32430B31042AE4EF783C117
shelldc.dll	21,297	45829795396a5af6db26fcf30d456f3c
shelldc.dll	21,297	60a25fc31c9360a69cc0535555a0fbff
hpmdp093.dll	21,297	b8735f55d7e0f3b0aaf8574dcfc2fe1a
hpmdp093.dll	21,297	CFDB09811E6FAB420B474D96BE40F371
Server.dll	21,297	d88e930bc3e514519b6c74ea9fa27dcb
recyle64.dll	21,297	E2DB67EACB919D4647E975F9A1BD6C5C
recyle32.dll	21,297	F41A1EDA474C642E5B080B3EFDD6197C
recyle64.dll	21,297	f84839503ec237be4e5ccb045a7c30d8
hpmdp093.dll	21,434	6B57D4A315BFE57DAD3DA74FF116363A
ver.exe	28,672	63f40d2104bbf15accd0a9c36978089c
Server.exe	28,672	a331dee4a6554ef70dc90628558a558a
Server.exe	28,672	c36a3275ae435e3ff1a387f475a0d579
ver.exe	28,672	F46E9C3049F0781779B24C3AB0DDD5BA
Server.exe	79,360	ca915897185e1ee3f811606f364bc995
connect.dll	89,120	6E31CCA77255F9CDE228A2DB9E2A3855
Sver.exe	159,744	36E6BDAE6E9E5004A7313F1D35A56528
sver.exe	159,744	e9b395829f985ce50e64374fd6653cab
zwShellx[1].exe	247,707	18801e3e7083bc2928a275e212a5590e
ghost.exe	631,808	093640A69C8EAFCB60343BF9CD1D3AD3
zwShellx.exe	631,808	093640A69C8EAFCB60343BF9CD1D3AD3
shell.exe	631,808	093640A69C8EAFCB60343BF9CD1D3AD3
zwShell.ini		30fc8ca9012a55f8f1a8953abb992f2

Network-Based Signatures

- The malware variants are designed to communicate to a remote host for command and control. The specific host (either hostname or IP address) and TCP port number are specified in the configuration data block at the end of the binary. The host and port that each variant communicates to is listed below:
 - shelldc.dll – shell.is-a-chef.com:3128
 - recyle64.dll – shell.is-a-chef.com:80
 - hpmdp093.dll – 134.146.82.25:53
 - ws_18.dll – shell.is-a-chef.com:80
- The malware initiates communication with the remote host with a 16-byte beacon packet which begins with the bytes 0x01 0x50 and ends with the bytes 0x68 0x57 0x24 0x13

Details

The malware is designed to be installed as a service by a secondary installer tool and provides no self-installation routine of its own. The first operation taken by the malware is to decode its configuration data block. It reads the last 737 bytes of its own file and decodes this data into a data structure used by the program. The algorithm used to decode the data is to XOR each byte with the remainder of index from the beginning of the structure divided by 128. The following python script illustrates the decoding algorithm:

```
f = open('shelldc.dll', 'rb')
f.seek(-737, 2)
buf = f.read()
buf2 = ""
for i in xrange(len(buf)):
    buf2 += chr(ord(buf[i]) ^ (i % 128))
print buf2 # decoded data
```

Figure 1 - Algorithm to Decode Configuration Block for shelldc.dll

The configuration structure contains the following fields when decoded (fields highlighted with unique colors, descriptions following data fragment).

From HBGary Dump (header added by Cryptome)

```
0000210: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
0000220: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
0000230: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
0000240: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
0000250: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
0000260: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
0000270: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
0000280: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
0000290: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
00002a0: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
00002b0: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
00002c0: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
00002d0: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
00002e0: 00
```

Figure 2 - Decrypted Configuration Block for shelldc.dll

Legend:

- Magic Number 0x13245768
- Service GroupName
- Mutex
- Service Name
- Service DisplayName
- Service Description
- Command and Control Host
- TCP Port number
- Disable IPSEC Flag
- File to delete (upon reboot)

When the malware is launched as a service (hosted in an svchost.exe process) its exported function ServiceMain is called. This function begins by checking the existence of a mutex object. The name of the mutex is specified in the configuration data block. If the mutex exists, the malware exits. This ensures that only one instance of the malware is running on the current host at a time.

After creating the mutex the malware begins its backdoor loop functionality. It first waits 30 seconds before initiating a connection to the remote command and control server. To communicate with the the remote server it sends a beacon packet.

Below you'll find the codes used by the malware that we've found in our network traces.

"All malware communications with the server follow a well-defined protocol. Each transmission begins with a 16-byte packet containing several fields such as a message type code. Message type code 0x5001 denotes the beacon packet. The following table describes the structure of 16-byte message packet."

Offset	Length	Description
0	2 Bytes	Message Type Code (50XX for outbound beacon and responses, 60XX for inbound commands)
2	1 Byte	Flag to signify that additional data will follow this packet
3	4 Bytes	Length of additional data (if flag is set)
7	4 Bytes	Variable 1 (unused by the program)
11	1 Byte	Variable 2 (for commands that need to transmit a small amount of data)

From HBGary Dump (header added by Cryptome)

12 4 Bytes Magic Number (0x13245768)

Figure - Message Packet Structure for shelldr.dll

The following table describes the commands accepted by the malware:

Code	Description
6001	Exit the command loop
6002	Causes the client to respond with a 5003 message with no data (heartbeat)
6003	Set the service name and group name in the registry to names specified in the additional data payload.
6004	Move the malware DLL to the same filename plus ".cnt" and sets the current ServiceDII registry value to the DLL name. It then schedules both the malware DLL and ".cnt" file to be deleted upon reboot.
6005	Retrieve a "+" delimited list of usernames that are currently connected to the host via a Remote Desktop session. (Response message code 5004 with additional data)
6011	Retrieve a list of currently connected drive letters with basic filesystem information including total size, free space and volume ID. (Response message code 5011 with additional data)
6012	Retrieve a list of directory entries for a specified directory. Directory specified with additional data following the command message packet. (Response message code 5012 with additional data)
6013	Shell Execute . Command or document to open is specified in the additional data following the command message packet. (Response message code 5100 with Variable 2 byte set to 2 if execution was successful)
6014	Move File . Filenames from and to are specified in the additional data following the command message packet. (Response message code 5100 with Variable 2 byte set to 4 if the move failed and 3 if it succeeded)
6015	Delete File . Filename to delete is specified in the additional data following the command message packet. (Response message code 5100 with Variable 2 byte set to 6 if the delete failed and 5 if it succeeded)
6016	Copy File . Filenames from and to are specified in the additional data following the command message packet. (Response message code 5100 with Variable 2 byte set to 8 if the copy failed and 7 if it succeeded)
6017	Set File Attributes . The First 4 bytes of the additional data payload following the command message packet specify the file attributes and data starting at offset 4 in the additional payload specifies the filename. (Response message code 5100 with Variable 2 byte set to 10 if the operation failed and 9 if it succeeded)
6018	Enter File Management Loop . Commands 6019 – 601D accepted.
6019	Exits File Management Loop . (Response message code 5013)
601A	Upload File to Remote Server . Must be in File Management Loop. File name to upload is specified in the additional data payload following the command message packet. The file transfer begins and ends with message code 5014. When the Variable 2 byte is set to 1 it denotes the beginning of the file transfer session and when it is set to 0 it denotes the end of the session. All content following the initial 5014 message to begin the transfer will be sent with response message code 5015 with up to 4K of additional data per message.
601B	Download A File . Must be in File Management Loop. This command begins a download from the remote server. Additional data may be downloaded into the file with command message code 601C.
601C	Download Data To Current File . Must be in File Management Loop. This command is used after a 601B command to download up to 4K of additional data per message.
601D	Retrieve Directory Entries . Must be in File Management Loop. (Response message 5014 to begin the list and 5015 for each entry)
601F	Retrieve a Recursive Directory Size . This command causes the malware to traverse the directory and calculate the total size of data under the directory. Due

From HBGary Dump (header added by Cryptome)

to improper coding techniques used by the malware author the program will produce inaccurate results in directories containing large files (greater than 4GB). (Response message code 501F returned)

6021 Launch Interactive Command Shell.

6031 Enter Registry Browser Mode. Registry key to start with specified in additional data payload. (Responds with message code 5001 with Variable 2 set to 4 to begin)

6032 Enumerate Registry Subkeys and Values. Must be in registry browser mode (command 6031). (Responds with return code 5100 with Variable 2 set to 10 if the operation failed and 9 if it succeeded)

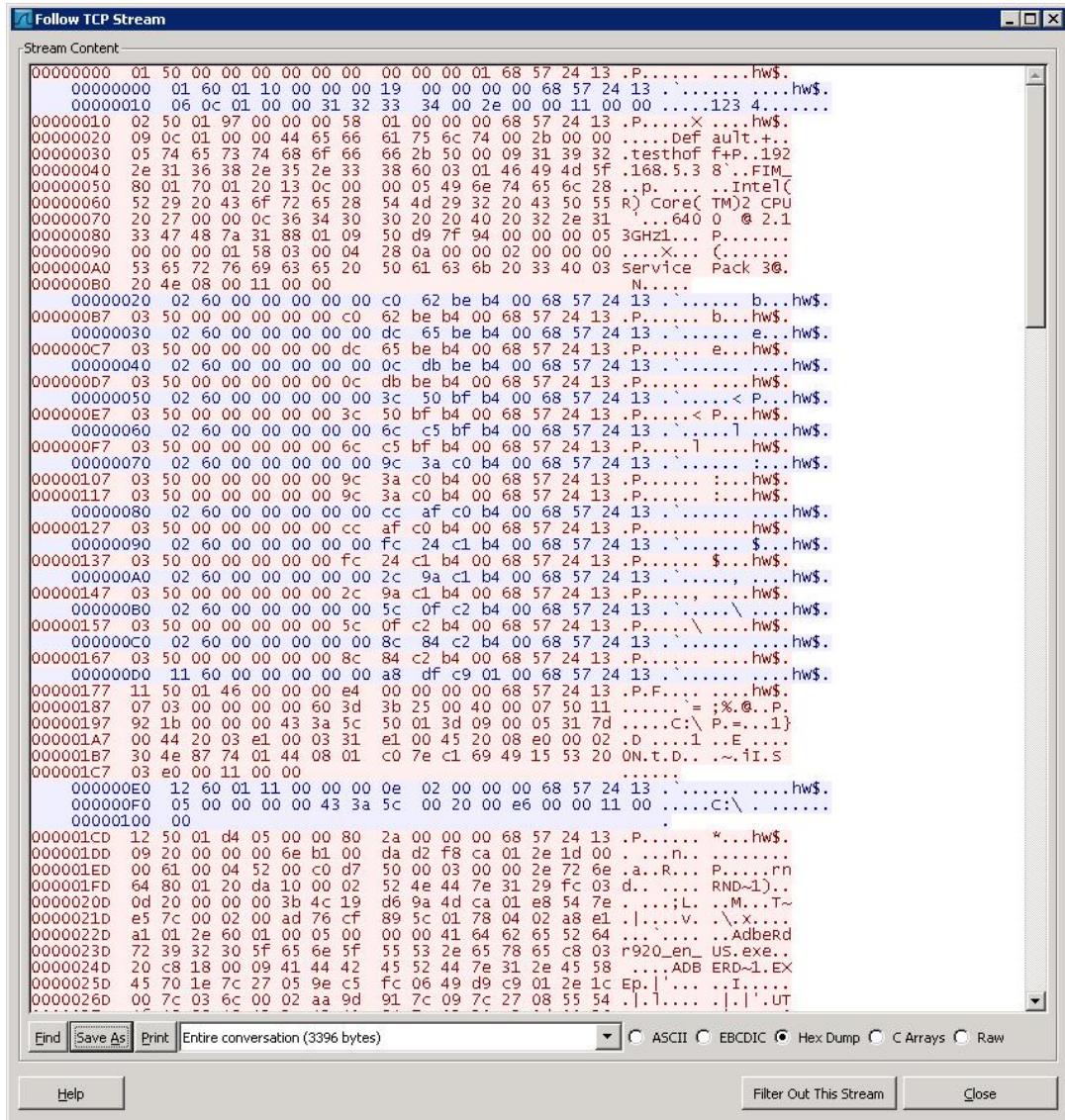
6041 Operate the Desktop Remotely. The additional data payload specifies a command structure with input operations such as mouse and keyboard. Responses contain screen capture information, providing an effective Remote Desktop implementation. Individual input commands are described later in this document.

We've attached a screenshot of a tcp stream of a session between the C&C server and an infected client. All the lines in red are client traffic and the blue lines is the C&C server traffic.

The final value of the message packet is the magic number of 0x13245768 in little endian byte order (hw\$). This is also the same magic number that is at the beginning of the decoded configuration data block of the produced malware clients by the zwShell.exe C&C application.

As you can see on the third line of the tcp stream you will find a transmitted password '1234'. After the handshake the client sends his system information to the C&C server. In this session we traversed a directory and retrieved a file.

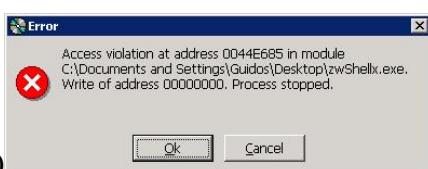
From HBGary Dump (header added by Cryptome)



Once the dropper gets created from the C&C application a port can be specified. This port will be found in the zwShell.ini file. We've already found such references on ports 1026 and 3128 on several systems.

Instructions to run zwShell.exe

For some reason zwShellx.exe breaks with this error:



251658240

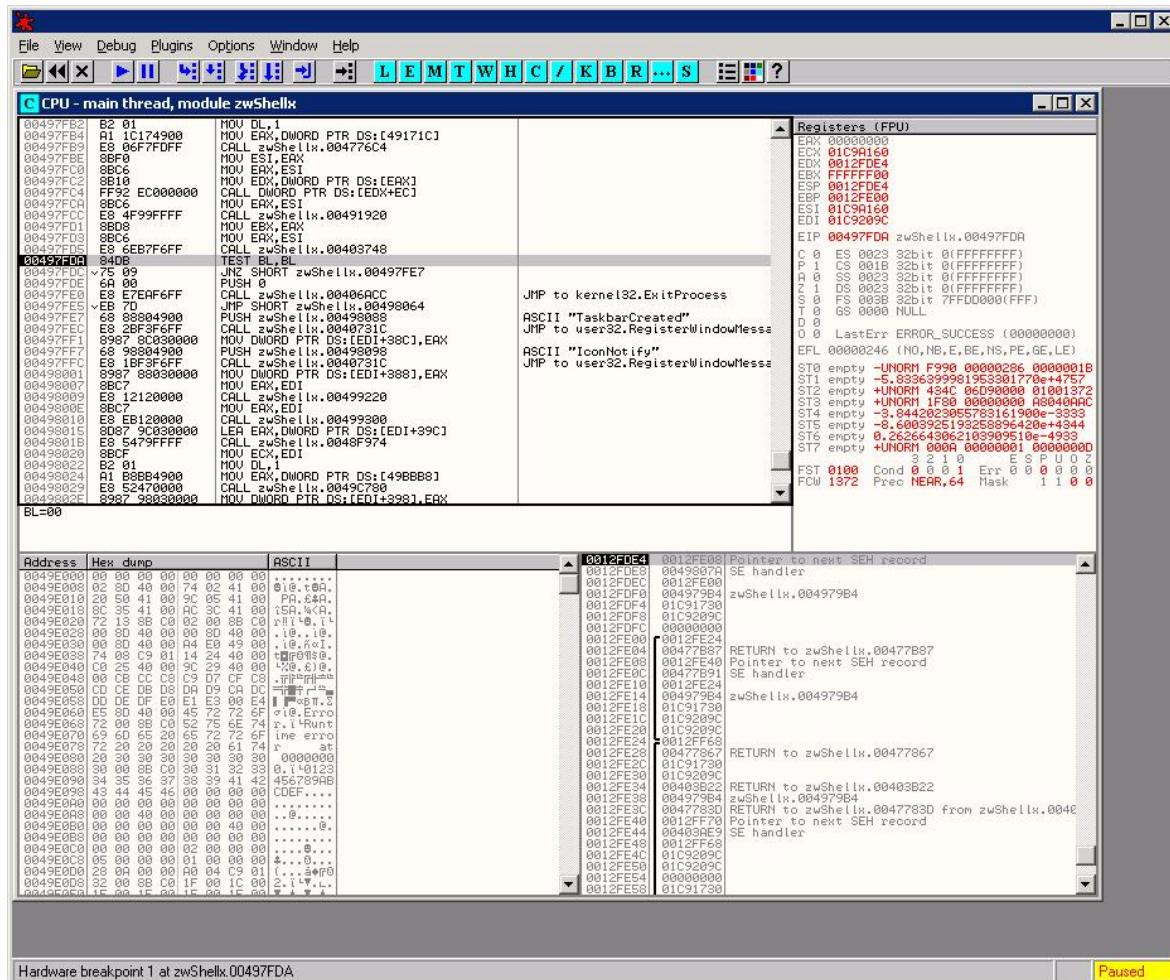
From HBGary Dump (header added by Cryptome)

It's not known yet what causes this error (possibly a missing ini file), but there is a quick workaround to get it to run properly:

1. Load zwShell.exe in Olly Debugger
2. Make sure the debugger is not detected by the ASProtect packer. Olly IsDebuggerPresent plugin works fine.
3. Configure Olly to ignore Access violation errors by adding the C0000005 exception in debugging options -> exceptions -> Ignore also following custom exceptions or ranges
4. Start zwShell
5. When the error message pops up, set a breakpoint at 00497FDA and press ok at the error dialog. See screenshot below.
6. Olly breaks at 00497FDA , where BL=0. Make sure the following JNZ jmp is taken by changing BL to 01.
7. zwShell will now run properly.

From HBGary Dump (header added by Cryptome)

251658240

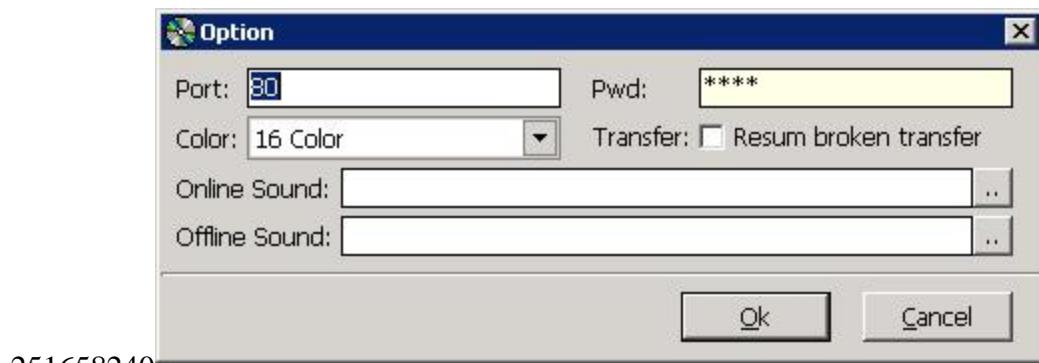
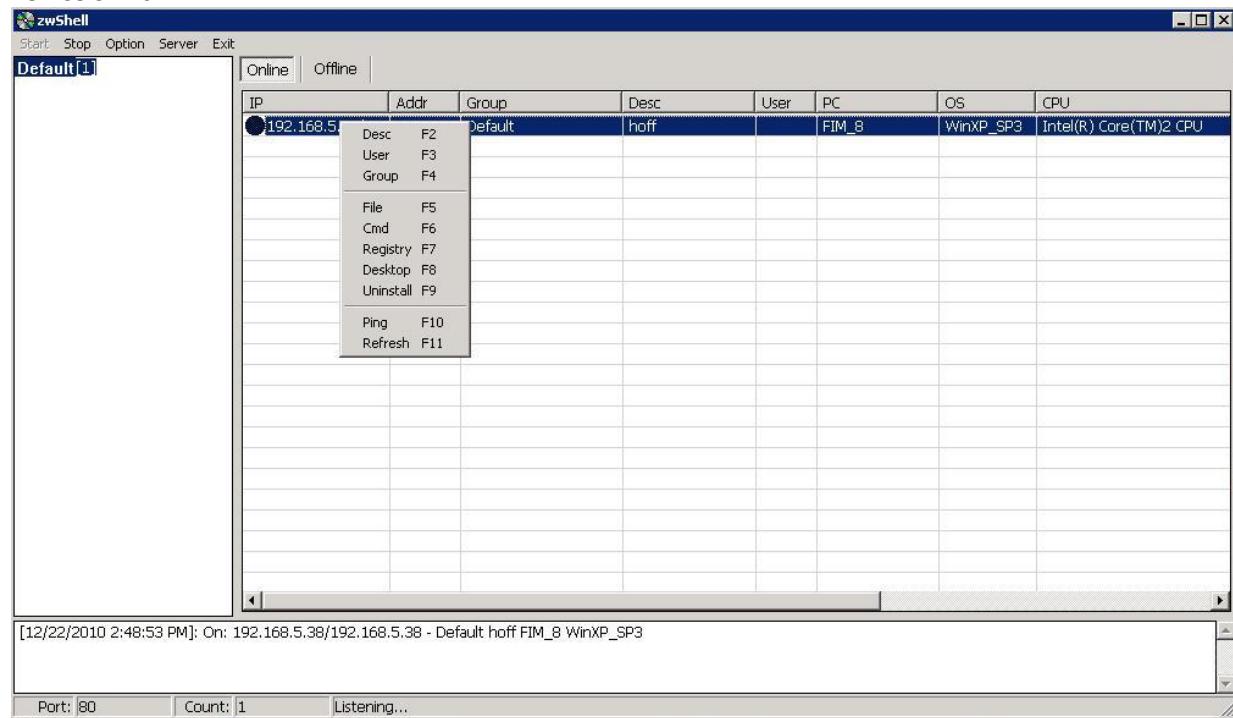


Screenshots

The following screen shots were collected from a system where the dropper/C2 was run in debug mode and used to infect a client machine.

From HBGary Dump (header added by Cryptome)

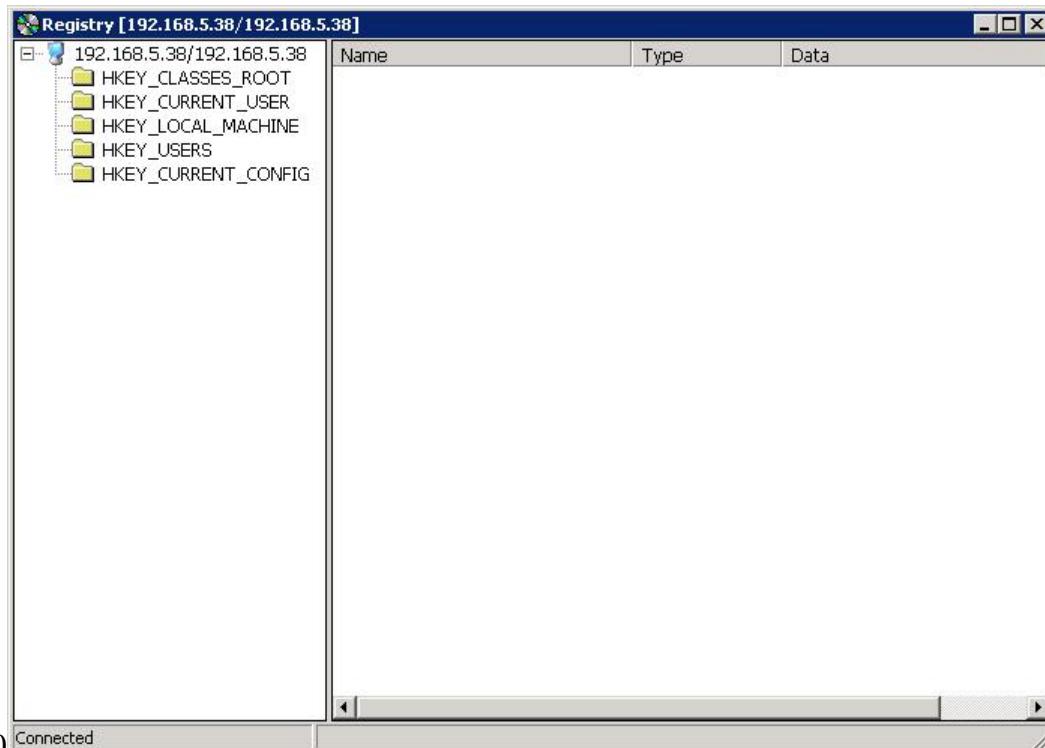
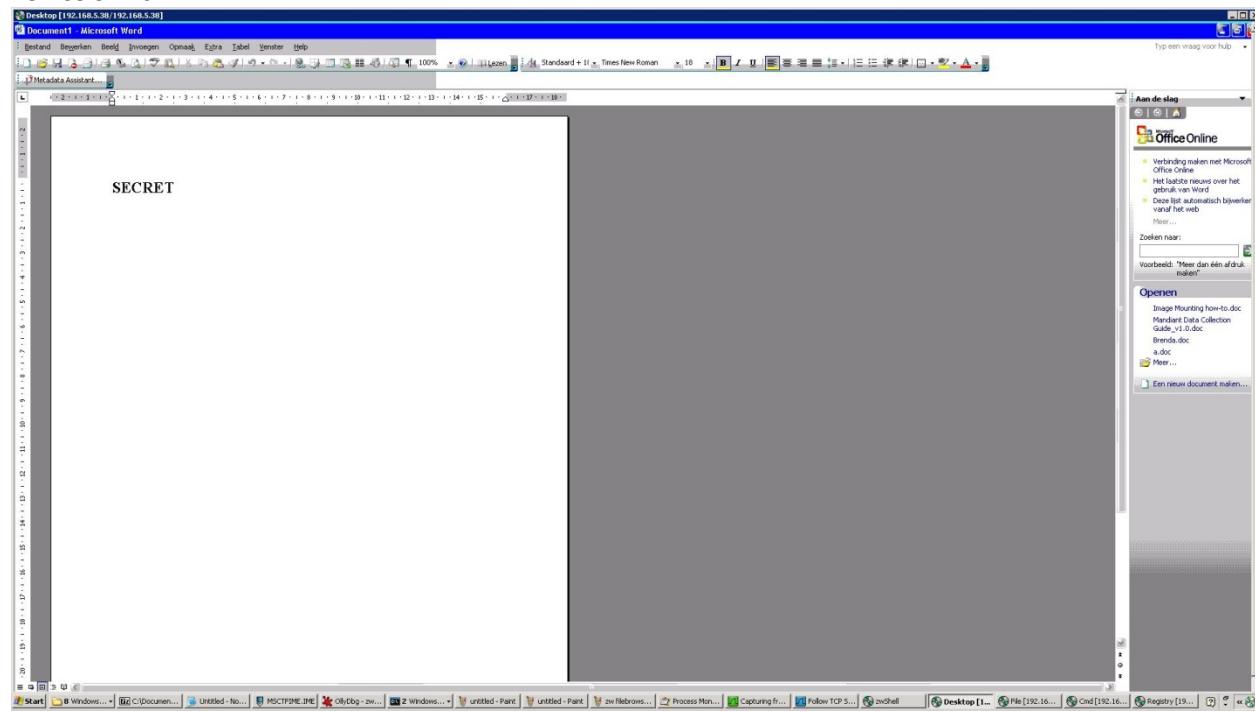
251658240



251658240

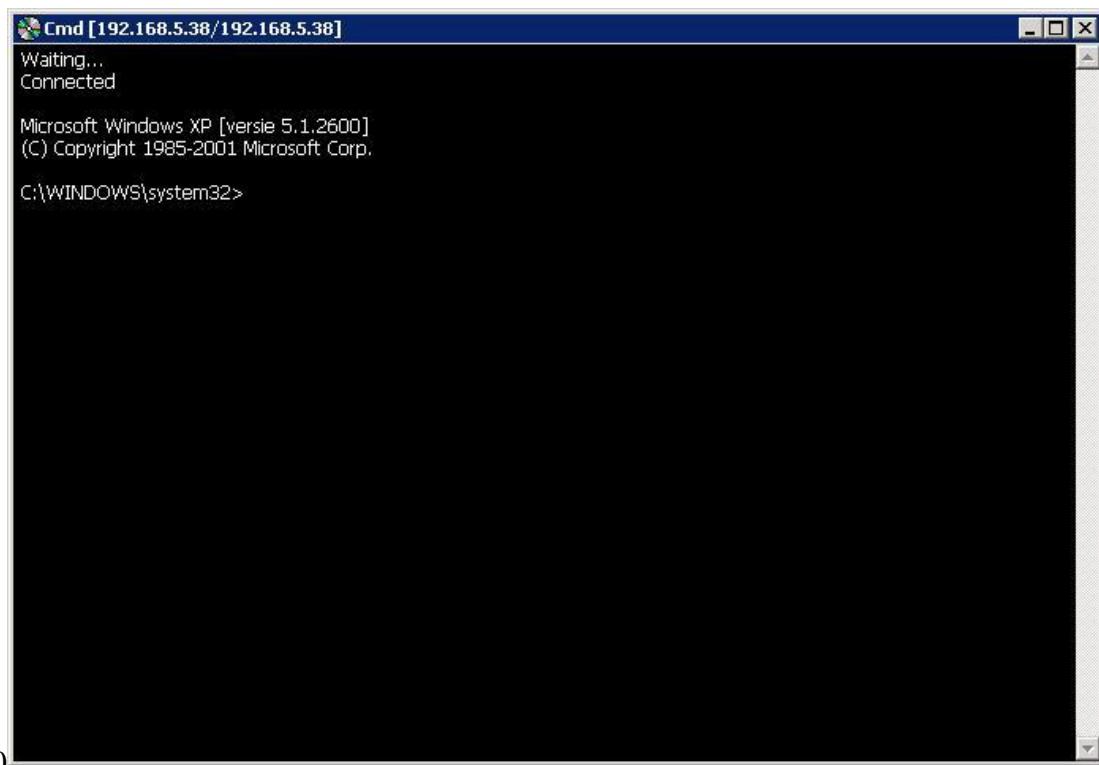
From HBGary Dump (header added by Cryptome)

251658240



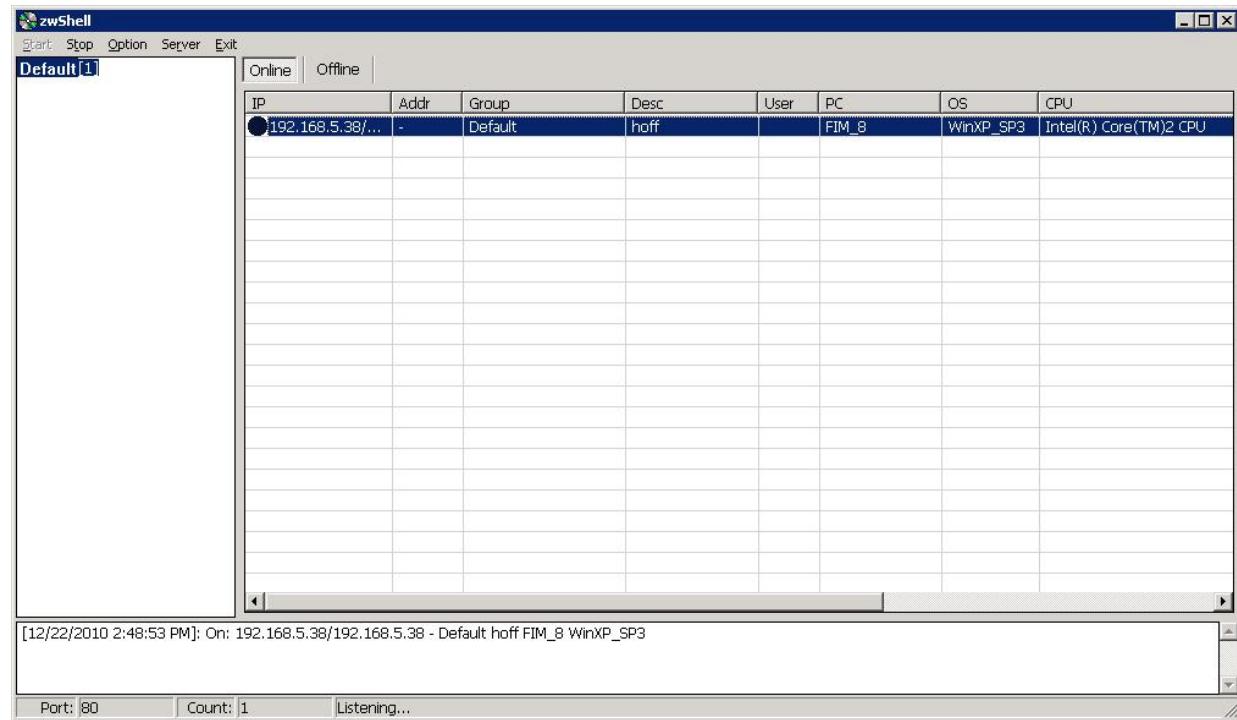
251658240

From HBGary Dump (header added by Cryptome)



251658240

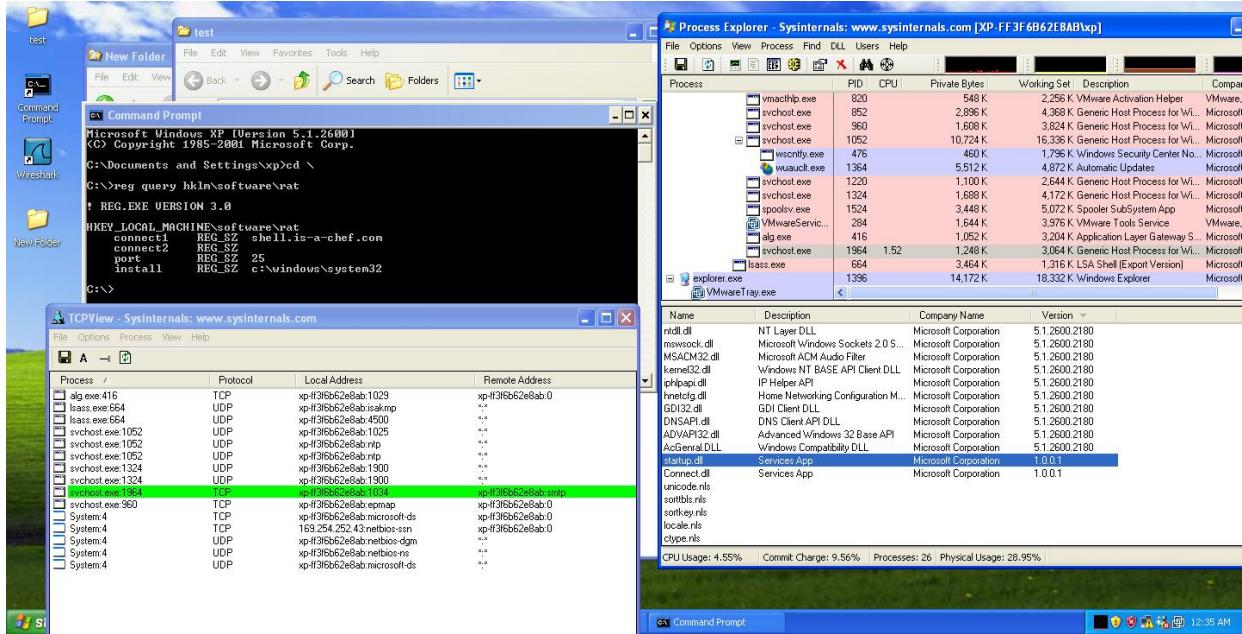
251658240



From HBGary Dump (header added by Cryptome)

The following Screenshot was collected from a VM that was infected with Sver.exe dropper which configured the Connect.dll and Startup.dll backdoor services:

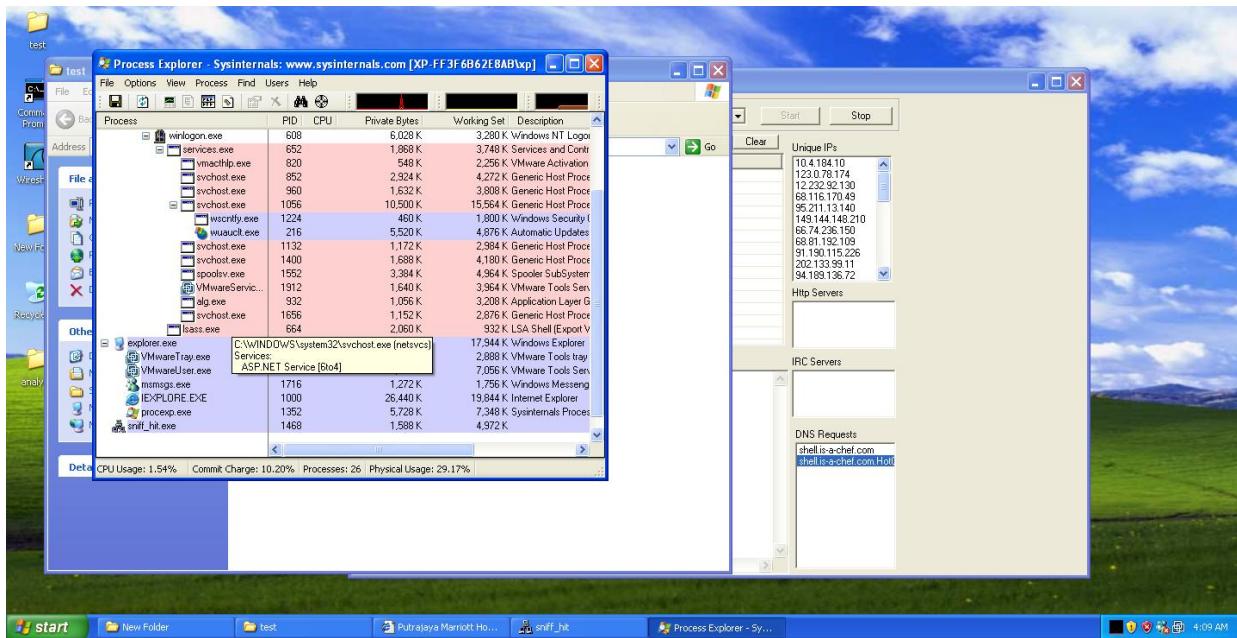
251658240



The following Screenshot was collected from a VM that was infected with the Server.exe dropper, which configured the ShellDC.dll backdoor services:

From HBGary Dump (header added by Cryptome)

251658240



The following was collected from Sandbox analysis of the Server.exe dropper and ShellDC.dll backdoor:

Processes:								
PID	ParentPID	User	Path					
1656	652	NT AUTHORITY:SYSTEM	C:\WINDOWS\system32\svchost.exe					
Ports:								
Port	PID	Type	Path					
shell.is-a-chef.com								
Explorer Dlls:								
DLL Path	Company NameFile Description							
No changes Found								
IE Dlls:								
DLL Path	Company NameFile Description							
No changes Found								
Loaded Drivers:								
Driver File	Company NameDescription							

From HBGary Dump (header added by Cryptome)

Monitored RegKeys	
Registry Key	Value
Hklm\SYSTEM\CurrentControlSet\Services	6to4
Kernel31 Api Log	
***** Installing Hooks *****	
71ab70df	RegOpenKeyExA (HKLM\System\CurrentControlSet\Services\WinSock2\Parameters)
71ab7cc4	RegOpenKeyExA (Protocol_Catalog9)
71ab737e	RegOpenKeyExA (00000005)
71ab724d	RegOpenKeyExA (Catalog_Entries)
71ab78ea	RegOpenKeyExA (000000000001)
71ab78ea	RegOpenKeyExA (000000000002)
71ab78ea	RegOpenKeyExA (000000000003)
71ab78ea	RegOpenKeyExA (000000000004)
71ab78ea	RegOpenKeyExA (000000000005)
71ab78ea	RegOpenKeyExA (000000000006)
71ab78ea	RegOpenKeyExA (000000000007)
71ab78ea	RegOpenKeyExA (000000000008)
71ab78ea	RegOpenKeyExA (000000000009)
71ab78ea	RegOpenKeyExA (000000000010)
71ab78ea	RegOpenKeyExA (000000000011)
71ab78ea	RegOpenKeyExA (000000000012)
71ab78ea	RegOpenKeyExA (000000000013)
71ab2623	WaitForSingleObject(79c,0)
71ab83c6	RegOpenKeyExA (NameSpace_Catalog5)
71ab737e	RegOpenKeyExA (00000004)
71ab7f5b	RegOpenKeyExA (Catalog_Entries)
71ab80ef	RegOpenKeyExA (000000000001)
71ab80ef	RegOpenKeyExA (000000000002)
71ab80ef	RegOpenKeyExA (000000000003)
71ab2623	WaitForSingleObject(794,0)
71aa1afa	RegOpenKeyExA (HKLM\System\CurrentControlSet\Services\Winsock2\Parameters)
71aa1996	GlobalAlloc()
7c80b511	ExitThread()
40144b	GlobalAlloc()
77de5f5e	WaitForSingleObject(7e4,2bf20)
77e9fb8e	RegOpenKeyExA (HKLM\Software\Microsoft\Rpc)
4010c0	RegOpenKeyExA (HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost)
4010c0	RegOpenKeyExA (HKLM\SYSTEM\CurrentControlSet\Services\6to4)
4014f8	CreateMutex(shelldc)
40136f	CreateFileA(C:\WINDOWS\System32\shelldc.dll)
40138e	WriteFile(h=770)
401060	RegCreateKeyExA
(HKLM\SYSTEM\CurrentControlSet\Services\6to4\Parameters,(null))	
401085	RegSetValueExA (ServiceDll)

From HBGary Dump (header added by Cryptome)

```
7c816d55 ExitThread()
7c80cd0c ExitProcess()
***** Injected Process Terminated *****

DirwatchData

-----
WatchDir Initilized OK
Watching C:\DOCUME~1\xp\LOCALS~1\Temp
Watching C:\WINDOWS
Watching C:\Program Files
Modified: C:\WINDOWS\system32\config\system.LOG
Created: C:\WINDOWS\system32\shelldc.dll
Modified: C:\WINDOWS\system32\shelldc.dll
Modified: C:\WINDOWS\system32
Modified: C:\WINDOWS\Prefetch
Created: C:\WINDOWS\Prefetch\SERVER.EXE-234C219D.pf
Modified: C:\WINDOWS\Prefetch\SERVER.EXE-234C219D.pf
Modified: C:\WINDOWS\system32\wbem\Logs\wbemess.log
Created: C:\WINDOWS\Prefetch\SNIFF_HIT.EXE-1AB02EA8.pf
Modified: C:\WINDOWS\Prefetch\SNIFF_HIT.EXE-1AB02EA8.pf
Created: C:\DOCUME~1\xp\LOCALS~1\Temp\JET37D0.tmp
Created: C:\DOCUME~1\xp\LOCALS~1\Temp\JET5.tmp
Deleted: C:\DOCUME~1\xp\LOCALS~1\Temp\JET5.tmp
Deleted: C:\DOCUME~1\xp\LOCALS~1\Temp\JET37D0.tmp
File: svhost.exe
Size: 14336 Bytes
MD5: 8F078AE4ED187AAABC0A305146DE6716
Packer: File not found C:\iDEFENSE\SysAnalyzer\peid.exe

File Properties: CompanyName Microsoft Corporation
FileDescription Generic Host Process for Win32 Services
FileVersion 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
InternalName svhost.exe
LegalCopyright © Microsoft Corporation. All rights reserved.
OriginalFilename svhost.exe
ProductName Microsoft® Windows® Operating System
ProductVersion

Exploit Signatures:
-----
Scanning for 19 signatures
Scan Complete: 24Kb in 0 seconds
Urls
-----
RegKeys
```

Software\Microsoft\Windows NT\CurrentVersion\Svchost

ExeRefs

File: svhost_dmp.exe_
svhost.exe
svhost.exe

Raw Strings:

File: svhost_dmp.exe_
MD5: d7c3d5fc02b6be4acc707157af3e2337
Size: 24578

Ascii Strings:

!This program cannot be run in DOS mode.
5Rich
.text
`.data
.rsrc
ADVAPI32.dll
KERNEL32.dll
NTDLL.DLL

```
F$pj
f9>t f
FFf9>u
f9>t
tof=
tSf=-
f9>t
FFf9>u
ShP$
Wh @
QRPh
u:Vj
PSSj
PSSj
@OSW
PWWj
@hJ.
_ []
u6j3
jWX]
QRPhh2
NETAPI32.dll
ole32.dll
Netbios
CoInitializeEx
CoInitializeSecurity
ADVAPI32.dll
KERNEL32.dll
ntdll.dll
RPCRT4.dll
RegQueryValueExW
SetSecurityDescriptorDacl
SetEntriesInAclW
SetSecurityDescriptorGroup
SetSecurityDescriptorOwner
InitializeSecurityDescriptor
GetTokenInformation
OpenProcessToken
OpenThreadToken
SetServiceStatus
RegisterServiceCtrlHandlerW
RegCloseKey
RegOpenKeyExW
StartServiceCtrlDispatcherW
HeapFree
GetLastError
WideCharToMultiByte
lstrlenW
```

```
LocalFree
GetCurrentProcess
GetCurrentThread
GetProcAddress
LoadLibraryExW
LeaveCriticalSection
HeapAlloc
EnterCriticalSection
LCMapStringW
FreeLibrary
lstrcpyW
ExpandEnvironmentStringsW
lstrcmpiW
ExitProcess
GetCommandLineW
InitializeCriticalSection
GetProcessHeap
SetErrorMode
SetUnhandledExceptionFilter
RegisterWaitForSingleObject
InterlockedCompareExchange
LoadLibraryA
QueryPerformanceCounter
GetTickCount
GetCurrentThreadId
GetCurrentProcessId
GetSystemTimeAsFileTime
TerminateProcess
UnhandledExceptionFilter
LocalAlloc
lstrcmpW
DelayLoadFailureHook
NtQuerySecurityObject
RtlFreeHeap
NtOpenKey
wcscat
wcscopy
RtlAllocateHeap
RtlCompareUnicodeString
RtlInitUnicodeString
RtlInitializeSid
RtlLengthRequiredSid
RtlSubAuthoritySid
NtClose
RtlSubAuthorityCountSid
RtlGetDaclSecurityDescriptor
RtlQueryInformationAcl
RtlGetAce
```

From HBGary Dump (header added by Cryptome)

```
RtlImageNtHeader
wcslen
RtlUnhandledExceptionFilter
RtlCopySid
RpcServerUnregisterIfEx
RpcMgmtWaitServerListen
RpcMgmtSetServerStackSize
RpcServerUnregisterIf
RpcServerListen
RpcServerUseProtseqEpW
RpcServerRegisterIf
I_RpcMapWin32Status
RpcMgmtStopServerListening
RSDS
svhost.pdb
```

Unicode Strings:

```
Parameters
System\CurrentControlSet\Services
nServiceMain
ServiceDll
ServiceDllUnloadOnStop
eventlog
ncacn_np
\PIPE\
DefaultRpcStackSize
AuthenticationCapabilities
ImpersonationLevel
AuthenticationLevel
CoInitializeSecurityParam
Software\Microsoft\Windows NT\CurrentVersion\Svchost
\Registry\Machine\System\CurrentControlSet\Control\SecurePipeServers\
VS_VERSION_INFO
StringFileInfo
040904B0
CompanyName
Microsoft Corporation
FileDescription
Generic Host Process for Win32 Services
FileVersion
5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
InternalName
svhost.exe
LegalCopyright
Microsoft Corporation. All rights reserved.
OriginalFilename
svhost.exe
```

```
ProductName
Microsoft
Windows
Operating System
ProductVersion
5.1.2600.2180
VarFileInfo
Translation
```

Shelldc.dll Strings info:

```
!This program cannot be run in DOS mode.
```

```
*Rich27
```

```
.text
```

```
`.rdata
```

```
@.data
```

```
.reloc
```

```
uY9E
```

```
uL9E
```

```
uB9E
```

```
u.9E t
```

```
9M u$9E$t
```

```
F@lu
```

```
vfH3
```

```
@AOu
```

```
@ANu
```

```
@ANu
```

```
@GNu
```

```
OAA
```

```
@GNu
```

```
BAOu
```

```
BANu
```

```
BANu
```

```
BGNu
```

```
OAA
```

```
BGNu
```

```
SVf
```

```
D0CPj@
```

```
PWV
```

```
HZx"
```

```
AF;t$
```

```
t3VWh
```

```
WSSj
```

```
jAP
```

```
hPa
```

```
Wh8a
```

```
PSSH
```

```
hx
```

hla
SVW3
PSSWj
SSSSh
SVW3
VSP
SQP
QQQQQ
SVWj
SSj
SSh
VWSP
v}Sj
WSSH
veSS
PSS
SSSSh
SSSSh
SSh
Ht\Ht\$H
SSj
SSh
-SSSSh
SSSSh
SVW
^VSP
RPQ
_WSP
PSSS
SPSSS
@SPj
<SVW3
SSh
SPh
VWP
PPPQ
PPP
VVVVVh1P
VWP
VVVVVh2P
Hff
SUV
(SUV
t\$8WV
[SWP
j(QP
j(QP
FfW

PWW
v6PW
v2WW
vZj
hAP
v6WW
hBP
QQSV
WSSj
ShCP
v6SS
hDP
PWW
vNj
Vj j
UVW
SSSj
SSSj
SSSj
SSSj
PSSH
tSV
SSh
SPhb9
SUV3
PWV
SSSh
SSPSS
SSt
SSWV
SUVWPh
SSh
SSh
uYVPP
YYj
YYV
uRGFGHt
Glu
GJu
wQn
wec
wFX
qMf
lstrcpyA

```
Sleep
GlobalFree
GlobalUnlock
GlobalHandle
GlobalLock
GlobalAlloc
IstrlenA
IstrcpyA
IstrcatA
DeleteFileA
CloseHandle
ReadFile
SetFilePointer
CreateFileA
GetVersionExA
GlobalMemoryStatus
GetComputerNameA
FreeLibrary
GetProcAddress
LoadLibraryA
MoveFileExA
MoveFileA
GetVolumeInformationA
GetDiskFreeSpaceExA
GetDriveTypeA
SetErrorMode
FindClose
FindNextFileA
GlobalReAlloc
IstrcmpA
FindFirstFileA
GetFileSize
WriteFile
TerminateProcess
GetExitCodeProcess
PeekNamedPipe
CreateProcessA
CreatePipe
CopyFileA
GetTempPathA
GetSystemDirectoryA
WaitForSingleObject
IstrcmpiA
CreateThread
GetCurrentThreadId
SetFileAttributesA
GetTickCount
GetModuleFileNameA
```

```
ReleaseMutex
CreateMutexA
OpenMutexA
KERNEL32.dll
CloseDesktop
SetThreadDesktop
OpenInputDesktop
GetThreadDesktop
ReleaseDC
GetDC
PostMessageA
OpenDesktopA
WindowFromPoint
GetCursorPos
SetCursorPos
MapVirtualKeyA
keybd_event
mouse_event
GetSystemMetrics
USER32.dll
ControlService
CloseServiceHandle
OpenServiceA
OpenSCManagerA
RegCloseKey
RegSetValueExA
RegOpenKeyExA
RegQueryValueExA
SetServiceStatus
RegEnumKeyExA
RegEnumValueA
RegisterServiceCtrlHandlerA
ADVAPI32.dll
WS2_32.dll
ShellExecuteA
SHELL32.dll
DeleteObject
DeleteDC
GetDIBits
BitBlt
SelectObject
CreateCompatibleBitmap
CreateCompatibleDC
GDI32.dll
Server.dll
ServiceMain
Oct 17 2005
Start
```

```
ProcessorNameString
HARDWARE\DESCRIPTION\System\CentralProcessor\0
GroupsName
ServiceName
SYSTEM\CurrentControlSet\Services\
open
Description
DisplayName
Type
PolicyAgent
WTSFreeMemory
WTSQuerySessionInformationA
WTSEnumerateSessionsA
wtapi32.dll
ServiceDll
\Parameters
.cnt
svchost.exe
\cmd.exe
exit
Winlogon
c:\windows\system32\shelldc.dll
shell
shelldc
las
ASP.NET Service
Provides support for out-of-process session states for ASP.NET.
shell.is-a-chef.com
C:\Documents and Settings\xp\Desktop\New Folder\Ugtxgt0gzg
hell.is-a-chef.com
=9>E>g>p>{>
0G0X0n0{0
4M4S4e4k4s4
5:5V5q5x5
6"6-6=6C6N6T6
6J7O7X7~7
?D?b?k?{?
8#9)969Q9X9z9
9C:`:g:x:
='>J>[>l>t>{>
0M0W0]0l0r0w0}0
0"1)12181>1D1J1P1V1\1b1h1n1t1z1
2"2(2.242:2@2F2L2R2X2^2d2j2p2v2|2
3$3*30363<3B3H3N3T3Z3`3f3l3r3x3~3
wmckd
!"#$%&'()*+,-./0123456789:;<=>?@AB0, *+, *JKLMNOPQRSTUVW
8)[\]^_`abcdefghijkl,=?^?7'T&
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFG
```

From HBGary Dump (header added by Cryptome)

```
;%=%)<p""#$:$#x?5)|2++M
^>SQ,MAQ(
!"PL@JK
MGUW
P[X6789:;<=>?@ABCDEFGHIJKLMNPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
```