

WMI Diagnosis Tool 2.0

Alain Lissoir
Senior Program Manager
Microsoft Corporation
Version 2.0
Monday, January 22, 2007

WHAT IS THE WMI DIAGNOSIS TOOL?	4
REQUIREMENTS	4
GETTING STARTED	4
HOW DO I RUN THE WMI DIAGNOSIS TOOL?	4
WHERE DOES THE WMI DIAGNOSIS TOOL REPORT?	5
HOW CAN I SEND THE WMI DIAGNOSIS TOOL RESULTS TO MICROSOFT?	5
HOW DO I RUN WMI DIAGNOSIS TOOL TO TROUBLESHOOT WMI?.....	5
WMI DIAGNOSIS TOOL RETURN CODES CLASSIFICATION	8
SAMPLE COMMAND LINE ARGUMENTS	10
WMI DIAGNOSIS TOOL GENERAL QUESTIONS	12
1. WHERE CAN I GET THE WMI DIAGNOSIS TOOL?	12
2. ARE THERE ANY PLANS TO INCLUDE THE WMI DIAGNOSIS TOOL IN NEW VERSIONS OF WINDOWS?	13
3. IS THERE A WMI DIAGNOSIS TOOL VERSION FOR LONGHORN SERVER?.....	13
4. IS IT POSSIBLE TO "OFFICIALLY" ENGAGE MICROSOFT FOR FEEDBACK ON THIS TOOL?	13
5. DID MICROSOFT DEVELOP THIS TOOL AS A SCRIPT KNOWING THAT A LOT OF US WILL ENHANCE IT FOR OUR OWN USES?	13
6. IS THERE ANY DOCUMENTATION DESCRIBING WHAT THE WMI DIAGNOSIS TOOL DOES NOT CHECK?	13
7. HOW MUCH DEVELOPMENT TIME WENT INTO CREATING WMI DIAGNOSIS TOOL?	13
8. ARE THERE PLANS TO DOCUMENT THE TOP PROBLEMS CUSTOMERS FIND WITH WMI DIAGNOSIS TOOL?	13
9. DO YOU PLAN TO WRITE FUTURE VERSIONS OF WMI DIAGNOSIS TOOL IN COMPILED CODE INSTEAD OF A SCRIPTED VERSION?	14
10. IS IT POSSIBLE TO STORE WMI DIAGNOSIS TOOL INFORMATION IN A SQL DATABASE INSTEAD OF A LOG FILE?.....	14
11. IS THERE A RELEASE DATE PLANNED FOR THE NEXT WMI DIAGNOSIS TOOL VERSION?	14
12. IS THERE AN E-MAIL ADDRESS TO SEND FEATURE REQUESTS FOR FUTURE VERSIONS?	14
13. WHAT ARE THE NEW FEATURES PLANNED FOR THE NEXT WMI DIAGNOSIS TOOL VERSION?	14
14. IS THERE AN INTERNET BLOG FOR WMI DIAGNOSIS TOOL?	14
WMI DIAGNOSIS TOOL USAGE QUESTIONS	14
1. HOW DO YOU USE THE EXCEL SPREADSHEET THAT COMES WITH THE WMI DIAGNOSIS TOOL?	14
2. HOW DO I KNOW WHAT TO FIX ONCE THE WMI DIAGNOSIS TOOL HAS EXECUTED SUCCESSFULLY?	16
3. CAN YOU RUN THE WMI DIAGNOSIS TOOL IN A RUNAS COMMAND?	16
4. CAN THE WMI DIAGNOSIS TOOL DIAGNOSE A REMOTE COMPUTER?	16
5. CAN THE WMI DIAGNOSIS TOOL BE REMOTELY EXECUTED WITH WINDOWS REMOTE SHELL (WINRS) ON WINDOWS VISTA?.....	16
6. CAN WMI DIAGNOSIS TOOL BE REMOTELY EXECUTED WITH PSEXEC.EXE FROM ON A REMOTE COMPUTER?.....	17
7. CAN WMI DIAGNOSIS TOOL BE RUN USING "WMIC PROCESS CALL CREATE"?	17
8. DOES THE WMI DIAGNOSIS TOOL FIX PROBLEMS IT DISCOVERS?	17
9. BY LIMITING WMI DIAGNOSIS TOOL TO CHECK ONLY ONE NAMESPACE, ARE YOU MISSING SOME POSSIBLE ERROR DETECTIONS?.....	17
10. IF THE REPOSITORY HAS BEEN DELETED AND THE COMPUTER STILL REPORTS WMI ERRORS, CAN THIS SITUATION BE RESOLVED WITH THE WMI DIAGNOSIS TOOL?	18
11. CAN THE WMI DIAGNOSIS TOOL IDENTIFY THE PROBLEM AND ROOT CAUSES OF THE REPOSITORY THAT IS GROWING VERY LARGE (+300 MB)?18	18
12. DOES THE WMI DIAGNOSIS TOOL REPORT ON ANY SECURITY SETTINGS OR PROBLEMS WITH RIGHTS?.....	18
13. DOES THE WMI DIAGNOSIS TOOL OUTPUT ERRORELEVEL CODES?	18
14. HOW CAN I DETERMINE IF I AM USING THE LATEST VERSION OF THE WMI DIAGNOSIS TOOL?.....	19
15. IF VERSION A OF WMIY.DLL AND VERSION B OF WMIX.DLL ARE PRESENT ON A SYSTEM, DOES THE WMI DIAGNOSIS TOOL TOOL DETECT THAT SCENARIO?.....	19
16. WILL THE LOG HAVE A UNIQUE NAME, OR CAN YOU SPECIFY A LOG NAME (PERHAPS WITH COMPUTER NAME AS A VARIABLE)?.....	19
17. WHEN RUNNING THE WMI DIAGNOSIS TOOL, CAN WE COMBINE MULTIPLE COMMAND LINE PARAMETERS?	19
18. HOW FREQUENT SHOULD THE WMI DIAGNOSIS TOOL SHOULD BE USED?	20
19. IS THE WMI DIAGNOSIS TOOL A GOOD UTILITY TO RUN ON WINDOWS 2000 COMPUTERS THAT ARE GOING TO BE UPGRADED TO WINDOWS SERVER 2003 USING THE UPGRADE PATH?	20
WMI DIAGNOSIS TOOL AND SMS QUESTIONS	20
1. DOES THE WMI DIAGNOSIS TOOL WORK WITH THE SMS 2003 ADVANCED CLIENT?	20
2. CAN THE WMI DIAGNOSIS TOOL FIND SERIOUS WMI PROBLEMS IF THE SMS CLIENT NEEDS A HEALTHY WMI INFRASTRUCTURE TO WORK?	20
3. IS THERE A BEST PRACTICE FOR WMI DIAGNOSIS TOOL PARAMETERS IN ORDER TO FIND COMMON ERRORS WHEN DEPLOYED VIA SMS 2003?.....	20
4. DOES MICROSOFT RECOMMEND THAT CUSTOMERS SEND IN ANY WMI FAILURES THAT OCCUR FROM SMS CLIENTS?.....	21

WMI DIAGNOSIS TOOL AND MOM QUESTIONS	21
1. ARE THERE ANY WMI DIAGNOSIS TOOL MANAGEMENT PACKS FOR MOM 2005?.....	21
WMI QUESTIONS	21
1. WHY IS WMI BROKEN AS OFTEN AS IT IS?	21
2. WHY DID MICROSOFT DEVELOP THE WMI DIAGNOSIS TOOL INSTEAD OF FIXING WMI?	22
3. IS THERE A TOOL OR A METHOD TO UNINSTALL/REINSTALL (REFRESH) ALL COMPONENTS (AND NON-CORE-OS DEPENDENCIES) OF WMI?	22
4. CAN YOU SHED SOME MORE LIGHT ON WHY WMI RETURNS AN 0x8004001 ERROR SO FREQUENTLY?	22
5. WHERE ARE WMI ERRORS DOCUMENTED?.....	23
6. IF WE HAVE BEEN DELETING OUR REPOSITORY AS A WAY TO REMEDY PROBLEMS, COULD WE HAVE DAMAGED ANYTHING IN THE PROCESS?	23
7. HOW CAN WE DETERMINE IF THE REPOSITORY IS ACTUALLY CORRUPTED?.....	24
8. HOW DOES THE WMI REPOSITORY GET RE-CREATED?.....	24
7. WMI REQUESTS HAVE LONG LATENCIES, UP TO 3 SECONDS. WILL THIS CHANGE IN THE FUTURE?	24
8. WHAT ARE ALL THE MOF FILES MARKED WITH ‘#PRAGMA AUTORECOVER’?	25
9. CAN YOU MODIFY WMI/DCOM SECURITY REMOTELY THROUGH COMMAND-LINE UTILITIES?	26
10. IS THERE A LIST ON MICROSOFT.COM THAT SHOWS ALL KNOWN APPLICATIONS USING WMI?	27
WMI DIAGNOSIS TOOL 2.0 UPDATES.....	28
WMI DIAGNOSIS TOOL NEW COMMAND LINE PARAMETERS	28
DCOM AND WMI NAMESPACE SECURITY ANALYSIS	29
WINZIP SUPPORT	31
WINDOWS VISTA SUPPORT.....	32
<i>User Account Control (UAC) sensing</i>	32
<i>Vista Firewall</i>	32
EVENT LOG QUERY TO LOCATE DCOM, WMI AND WMIADAPTER EVENTS FOR THE LAST 20 DAYS	33
RUN-TIME ENVIRONMENT VERIFICATION	33
<i>Analyze EventLog to detect suspicious improper system shutdown</i>	33
<i>Enhanced registry checking</i>	34
<i>Cscript timeout</i>	34
<i>WBEM presence in path and maximum PATH length</i>	34
THE WMI DIAGNOSIS TOOL REPORT IS AVAILABLE IN A SEPARATE FILE	34
SMS WMI INFORMATION	34
PERMANENT SUBSCRIPTIONS AND TIMERS.....	35
MOF FILE VERIFICATION.....	37
LOADED WMI PROVIDERS	37
MOFCOMP AND REGSVR32.....	40
WMI DIAGNOSIS TOOL KNOWN LIMITATIONS.....	40
THE SEQUENCE OF OPERATIONS	40
NON-WMI OPERATIONS OF THE WMI DIAGNOSIS TOOL (STEP 1)	40
WMI OPERATIONS OF THE WMI DIAGNOSIS TOOL (STEP 2).....	41
NON-WMI OPERATIONS OF THE WMI DIAGNOSIS TOOL (STEP 3)	41
WMI DIAGNOSIS TOOL LOG CONTENT	41
COLLECTING RUN-TIME ENVIRONMENT INFORMATION	42
VERIFYING RUN-TIME ENVIRONMENT (UPGRADE, PRIVILEGES, WSH TIMEOUT, FILES PRESENCE).....	43
VERIFYING REPOSITORY FILE PRESENCE	44
VERIFYING ADDITIONAL (UNEXPECTED) FILES IN WBEM FOLDER	44
VERIFYING MOF FILES IN AUTO-RECOVERY REGISTRY KEY PRESENCE.....	44
VERIFYING ‘#PRAGMA AUTORECOVER’ STATEMENT PRESENCE IN MOF FILES	44
VERIFYING DCOM CONFIGURATION	45
VERIFYING WMI DCOM COMPONENT REGISTRATIONS.....	45
VERIFYING DCOM SECURITY	45
VERIFYING WMI PROGID REGISTRATIONS	46
VERIFYING WINDOWS FIREWALL SETTINGS.....	46
VERIFYING WINDOWS VISTA FIREWALL SETTINGS	46
VERIFYING WMI CORE REGISTRY SETTINGS.....	48
VERIFYING WMI SERVICE REGISTRY SETTINGS	48
VERIFYING LOADED PROVIDERS BEFORE/AFTER WMI DIAGNOSIS TOOL EXECUTION	49

VERIFYING WMI ROOT NAMESPACE SETTINGS.....	50
VERIFYING EXISTING STATIC INSTANCES	51
VERIFYING TIMER INSTRUCTIONS.....	51
VERIFYING THE NAMESPACE SECURITY.....	52
VERIFYING THE PROVIDER CIM AND DCOM REGISTRATION.....	53
VERIFYING PERMANENT SUBSCRIPTIONS	53
VERIFYING ADAP STATUS.....	54
GET OPERATIONS.....	55
WMI ENUMERATION OPERATIONS	55
WMI EXECQUERY OPERATIONS	56
VERIFYING SMS WMI TYPICAL OPERATIONS	56
COLLECTING SYSTEM INFORMATION ABOUT THE DISK SUB-SYSTEM.....	57
COLLECTING SYSTEM INFORMATION ABOUT THE NETWORK ADAPTERS	58
COLLECTING SYSTEM INFORMATION ABOUT DMA.....	59
COLLECTING SYSTEM INFORMATION ABOUT IRQ	60
COLLECTING SYSTEM INFORMATION ABOUT MEMORY	61
COLLECTING SYSTEM INFORMATION ABOUT PROCESSOR	61
COLLECTING SYSTEM INFORMATION ABOUT THE OPERATING SYSTEM.....	61
COLLECTING INFORMATION ABOUT THE QFE	63
COLLECTING SYSTEM INFORMATION ABOUT WINDOWS SERVICES.....	64
COLLECTING SYSTEM INFORMATION ABOUT WMI BINARY FILES	64
COLLECTING SYSTEM INFORMATION ABOUT DCOM,WMI AND WMIADAPTER EVENT LOG ENTRIES	64
COLLECTING SYSTEM INFORMATION ABOUT SUSPICIOUS SHUTDOWNS	65
WMI REPORT	65

What is the WMI Diagnosis Tool?

The WMI Diagnosis Tool is a VBScript based-tool for testing, validating and analyzing a WMI installation. The tool collects data from WMI installations on computers Windows 2000 (all service packs), Windows XP (all service packs), Windows 2003 (all service packs) and Windows Vista including, as well Longhorn Server.

The tool is targeted towards advanced IT pros, PSS organizations, and WMI developers to help diagnose WMI installations and to then fix problems based on the diagnostic report created by the tool.

Requirements

- You must be a full administrator to run the WMI Diagnosis Tool. Under Windows Vista you must be an elevated Administrator.
- Windows Script Host (WSH) must be enabled. WSH 5.1 is installed by default in Windows 2000, while WSH 5.6 is installed by default in Windows XP and Window Server 2003.Latest versions of Windows have WSH 5.7 installed.
- The WMI Diagnosis Tool must be run on the local computer to be tested but can however be launched remotely (See FAQ further in this document).

Getting started

How do I run the WMI Diagnosis Tool?

The WMI Diagnosis Tool can be run from Windows Explorer or from the command line. As shown below, no specific command line arguments are required to run the tool in default mode:

```
C:\>WMIDiag.vbs
```

The WMI Diagnosis Tool can also be run from management agents such as SMS. However, to prevent user interaction the *Silent* switch is required in these situations.

Where does the WMI Diagnosis Tool report?

Each time it runs, the WMI Diagnosis Tool creates three files:

- A .LOG file containing all the WMI Diagnosis Tool activity as well as a WMI report at the end of the .LOG file.
- A .TXT file containing the WMI Diagnosis Tool report (the same report found at the end of the .LOG file).
- A .CSV file containing statistics that can be used to measure trends and issues. This CSV file can be imported into the Excel sheet found in the WMI Diagnosis Tool download. The file can also be used to consolidate data in a database.

The three files are created, by default in the %TEMP% directory, contain information related to both the operating system configuration and WMI. They contain information like the system name, the user name, the network configuration, the hardware configuration, the software configuration and all the WMI system parameters with associated test results mentioned further in this paper. You can change the WMI Diagnosis Tool default behaviors by specifying additional command line arguments. Refer to the section of this paper titled “Sample command line arguments” for a complete list of WMI Diagnosis Tool arguments.

When the WMI Diagnosis Tool terminates, the ERRORLEVEL environment variable is set to one of the following values:

- 0 = SUCCESS
- 1 = ERROR
- 2 = WARNING
- 3 = Command Line Parameter errors
- 4 = User Declined (Clicked the Cancel button when getting a consent prompt)

See the section of this paper titled “WMI Diagnosis Tool return codes classification” for information about these return codes.

How can I send the WMI Diagnosis Tool results to Microsoft?

You can send WMI Diagnosis Tool information to Microsoft by configuring the SMTP command-line arguments. Be aware that, **by specifying relay information, you give explicit consent to send this information to Microsoft.** See the section of this paper titled “Sample command line arguments” to see the complete list of WMI Diagnosis Tool arguments.

How do I run WMI Diagnosis Tool to troubleshoot WMI?

Step 1:

When you run the WMI Diagnosis Tool on a new computer you should initially run the tool in default mode (no command-line arguments):

```
C:\>WMIDiag.vbs
```

This enables you to identify the potential problems as well get a better understanding of the system installation.

Step 2:

Analyze the output of the WMI report generated by WMI Diagnosis Tool. At the end of the .LOG file you will find the WMI report which is the same information found in the .TXT file. An example of this report follows. The report contains two types of information:

- Information that is useful if certain actions are executed (for example, the consequence of stopping the WMI service). This type of information is reported as “WARNING”
- Problems that need to be solved to avoid errors reported by WMI (for example, missing provider registration for a specific set of WMI classes). This type of information is reported as “ERROR”.

```
(0) ** -----
(0) ** ----- WMI REPORT: BEGIN -----
(0) ** -----
(0) ** -----
(0) ** Windows XP - Service pack 2 - 32-bit - User 'PC-ALAINL-02\ALAINL' on computer 'PC-ALAINL-02'.
(0) ** -----
(0) ** There is no missing WMI system files: ..... OK.
(0) ** There is no missing WMI repository files: ..... OK.
(0) ** - The WMI repository has a size of: ..... 28 MB.
(0) ** -----
(2) !! INFO: Windows Firewall status: ..... ACTIVE.
(0) ** => Windows Firewall 'RemoteAdmin' status is INACTIVE.
(0) ** - This will prevent any WMI connectivity to this computer.
(0) ** - You can adjust the configuration by executing the following command:
(0) ** - 'NETSH.EXE FIREWALL SET SERVICE REMOTEADMIN ENABLE SUBNET'
(0) ** => Windows Firewall application exception for 'UNSECAPP.EXE' is missing.
(0) ** - This will prevent any script and MMC application asynchronous callbacks to this computer.
(0) ** - You can adjust the configuration by executing the following command:
(0) ** - 'NETSH.EXE FIREWALL SET ALLOWEDPROGRAM %SYSTEMROOT%\SYSTEM32\WBEM\UNSECAPP.EXE WMICALLBACKS ENABLE'
(0) ** -----
(0) ** WMI Service setup: ..... OK.
(2) !! INFO: WMI service has dependents: ..... WARNING.
(0) ** - Security Center (WSCSVC, StartMode='Automatic')
(0) ** - Windows Firewall/Internet Connection Sharing (ICS) (SHAREDACCESS, StartMode='Automatic')
(0) ** - SMS Agent Host (CCMEEXEC, StartMode='Automatic')
(0) ** - IPv6 Helper Service (6TO4, StartMode='Automatic')
(0) ** => If the WMI service is stopped, the listed service(s) will have to be stopped as well.
(0) ** (Note: If the service is marked with (*), it means that the service may use WMI but
(0) ** there is no hard dependency on WMI. However, if WMI is stopped, this can prevent
(0) ** the service to work as expected).
(0) ** RPCSS service: ..... OK.
(0) ** WINMGMT service: ..... OK.
(0) ** -----
(0) ** WMI service DCOM setup: ..... OK.
(0) ** WMI components DCOM registrations: ..... OK.
(1) !! ERROR: WMI provider DCOM registrations missing for the following provider(s): ..... 1 ERROR(S)!
(0) ** - ROOT/FAKE, FakeProv (e.g. WMI Class 'FakeClass')
(0) ** => This is an issue because there are still some WMI classes referencing this list of provider(s)
(0) ** while the DCOM registration is wrong or missing.
(0) ** => You can correct the DCOM configuration by executing the 'REGSVR32.EXE <Provider.DLL>' command.
(0) ** WMI provider CLSIDs: ..... OK.
(1) !! ERROR: Some WMI providers EXE/DLL file(s) are missing: ..... 1 ERROR(S)!
(0) ** - ROOT/COMPAQ, CompaqCustomerRegistration,
(0) ** C:\Program Files\HPQ\Help and Support Center\CustomerRegInfo.dll
(0) ** => This will make any operations related to the WMI class supported by the provider(s) to fail.
(0) ** => You must restore a copy of the missing provider EXE/DLL file(s) as indicated by the path.
(0) ** WMI Registry key setup: ..... OK.
(0) ** -----
(0) ** All WMI connections succeeded: ..... OK.
(1) !! ERROR: Some WMI enumeration errors occurred: ..... 2 ERROR(S)!
(0) ** - ROOT/SECURITYCENTER, InstancesOf, __Win32Provider, did not return any provider instance
(0) ** registration while at least 1 instance is expected.
(0) ** - ROOT/MICROSOFT/HOMENET, InstancesOf, __Win32Provider, did not return any provider instance
(0) ** registration while at least 1 instance is expected.
(0) ** All WMI get operations succeeded: ..... OK.
(0) ** All WMI execquery operations succeeded: ..... OK.
(0) ** All WMI qualifier access operations succeeded: ..... OK.
(0) ** -----
(0) ** -----
(0) ** -----
(0) ** ----- WMI REPORT: END -----
(0) ** -----
```

Step 3:

Fix all reported problems as suggested by the WMI Diagnosis Tool. Follow the suggested steps in order.

Step 4

After you have fixed the problems run the WMI Diagnosis Tool one more time (in default mode) to ensure that the problems have been taken care of. If a problem has been correctly fixed it should disappear from the WMI report.

Step 5

If your applications encounter issues with the WMI repository, you can:

- For Windows XP SP2, Windows Server 2003 SP1 and above

Run the WMI Diagnosis Tool with the *CheckConsistency* argument. Under Windows Server 2003 SP1, this parameter is active all the time. However, under Windows XP SP2, *CheckConsistency* must be explicitly requested. This is due to the fact that under Windows XP SP2, when the repository consistency is checked and if it is found to be inconsistent, a new repository is automatically created. It is important to note that some information can be lost during this process (for example, static data or CIM registration data). However, the original repository will be located at `%SystemRoot%\System32\Repository.001`

- For platforms before Windows XP SP2 and Windows Server 2003 SP1 (Windows 2000, Windows XP RTM, Windows XP SP1, Windows Server 2003 RTM)

Run WMI Diagnosis Tool with the *WriteInRepository* argument, specifying the base namespace where the write operation has to start. This test is non-destructive and can increase the size of the repository by 1 MB.

```
C:\>WMIDiag.vbs WriteInRepository=Root
```

The command line parameter tells WMI Diagnosis Tool to write 250 static instances in each of the examined namespaces starting from the Root namespace.

Specifying no base namespace with the *WriteInRepository* parameter creates 10 temporary namespaces with 250 temporary static classes. This test is non-destructive and can increase the size of the repository by 1 MB.

```
C:\>WMIDiag.vbs WriteInRepository
```

If the actions generated by *WriteInRepository* succeed, then you can have a fair confidence in the consistency of the repository. Note that it is not the ultimate confirmation (only the *CheckConsistency* command line parameter does). In that case, you should also check whether:

- The account that runs the application has the required access rights to save data in the repository. For example, your application must be running under an administrator account when the application tries to save data in the repository.
- All required WMI classes are present in the repository before attempting to write additional data. For example, if you compile a MOF file, ensure that the WMI classes you add do not depend on (inherit from) classes that are missing from the repository.

Step 6

The WMI Diagnosis Tool might not return any errors, other than eventual Access Denied errors, when using the *WriteInRepository* argument. If this is not the case:

1. Restore a copy of the WMI repository (a copy of the repository is always available in the Windows System Restore and in the System State Backup)
2. If you do not have a repository backup, delete the repository. When the WMI service restarts the repository will be rebuilt.

WARNING: Before deleting the existing repository, you must make sure that all the MOF files used to build the repository are available and listed in the Auto-Recovery registry key. To analyze the MOF files run WMI Diagnosis Tool with the *ShowMOFErrors* arguments. See the section of this paper titled “Sample command line arguments” for the complete list of WMI Diagnosis Tool arguments.

WMI Diagnosis Tool return codes classification

The WMI Diagnosis Tool returns a general state of the WMI infrastructure after completing its system analysis. Note that, even if no problems are reported, WMI Diagnosis Tool reports information regarding operations and WMI components. The status is classified in four categories summarized in the table below.

SUCCESS/INFO (ErrorLevel = 0)
WSH has a script execution timeout setup (in machine or system environment).
Machine reports suspicious improper shutdowns (not all event log start events match all event log stop events).
User Account Control (UAC) status is reported (Vista and above).
Local account token filter policy is reported (Vista and above).
Unexpected binaries in the WBEM folder.
The Windows Firewall is enabled.
Some WMI service installed in the machine are dependent on the WMI service (i.e. "SMS Agent", "Live Communication Server", "Exchange Server 2000/2003").
WMI service is setup to run as a STANDALONE host (when it should be a SHARED host as expected by the default setup).
Some WMI providers are running with high privileges (i.e. "LocalSystem"). This information is for support personnel only with the <i>ShowHighPrivProviders</i> command line parameter).
A new ACE with a granted access has been added to a default trustee of a DCOM or WMI security descriptor (This information is for support personnel only with the <i>Debug</i> command line parameter).
A new trustee with a granted access has been added to an actual DCOM or WMI security descriptor (This information is for support personnel only with the <i>Debug</i> command line parameter).
WMI ADAP has a status different than 'running'.
Some WMI namespaces require a packet privacy encryption for a successful connection.
Some WMI permanent subscriptions or timer instructions are configured.
Some information about registry key configurations for DCOM and/or WMI were reported.
ERROR (ErrorLevel = 1)
System32 or WBEM folders are not in the PATH.
WMI system file(s) is/are missing.
WMI repository file(s) is/are missing.
WMI repository is inconsistent (XP SP2, 2003 SP1 and above).
DCOM is disabled.
WMI service is disabled.
The RPCSS and/or the WMI service(s) cannot be started (i.e. not running and startup disabled, registry configuration issues, service host issues, WMI repository issues)
WMI DCOM setup issues (missing CLSID or any other WMI service registry issues related to the DCOM setup)
An expected default trustee has been removed from an actual DCOM or WMI security descriptor (default trustee not found).
An expected default ACE for a default trustee has been removed from an actual DCOM or WMI security descriptor.
The ADAP status is not available.

One or more WMI connections with a moniker failed (winmgmts:).
One or more WMI connections with the .Connect method failed.
Some GET operations failed.
Some WMI class MOF representations failed.
Some WMI qualifier retrieval operations failed.
Some critical WMI ENUMERATION operations failed (i.e. "__Win32Provider", "__NAMESPACE", "__CIMOMIdentification", "__ProviderHostQuotaConfiguration")
Some critical WMI EXECQUERY operations failed (i.e. "Select * From __ClassProviderRegistration", "Select * From Win32_PageFileUsage", "Select * From Win32_Service", "Select * From Win32_LogicalDisk WHERE FreeSpace > 10000000 AND DriveType = 3")
Some critical WMI GET operations failed.
Some WRITE operations in the WMI repository failed.
Some PUT operations failed.
Some DELETE operations failed.
One of the queries of the event log entries for DCOM, WMI and WMIADAPTER failed.
Some critical registry key configurations for DCOM and/or WMI were reported.
WARNING (ErrorLevel = 2)
System32 or WBEM folders are further in the PATH string than the maximum system length.
System drive and/or Drive type reporting are skipped (because of an issue with the WMI provider).
Presence of rogue files at suspicious locations and using WMI system file names (trojan such as WINMGNT.EXE).
DCOM has an incorrect default authentication level (other than 'Connect').
DCOM has an incorrect default impersonation level (other than 'Identify').
WMI service has an invalid host setup (not SHARED or STANDALONE because of some registry key inconsistencies).
WMI service (SCM configuration) has an invalid registry configuration.
Some WMI components have a DCOM registration issue.
WMI COM ProgID cannot be instantiated (COM registration issue).
Some WMI providers have a DCOM registration issue (the corresponding DCOM registration is not found from the actual CIM registration).
Some dynamic WMI classes have a registration issue (WMI classes are supported by a WMI provider that does not have any CIM registration)
Some WMI providers are registered in WMI but their registration lacks a CLSID.
Some WMI providers have a correct CIM/DCOM registration but the corresponding binary file cannot be found.
A new ACE with a denied access has been added to a default trustee of a DCOM or WMI security descriptor.
A new trustee with a denied access has been added to an actual DCOM or WMI security descriptor.
A default trustee has been modified with a denied access to an actual DCOM or WMI security descriptor.
An invalid ACE has been found for an actual DCOM or WMI security descriptor.
WMI ADAP never ran on the examined system.
Some WMI non-critical ENUMERATION operations failed ("References_" for permanent subscriptions, "Associators_" for permanent subscriptions)
Some WMI ENUMERATION operations were skipped (because of an issue with the WMI provider).
Some WMI non-critical EXECQUERY operations failed (i.e. "Select * From __EventConsumer", "Select * From __TimerInstruction", "Select * From MSFT_Providers", "Select * From Win32_VideoController")
Some WMI EXECQUERY operations were skipped (because of an issue with the WMI provider).
Some non-critical WMI GET VALUE operations failed (i.e. "__ObjectProviderCacheControl=@, ClearAfter property", "__ProviderHostQuotaConfiguration=@, HandlesPerHost property")
Some WMI GET VALUE operations were skipped (because of an issue with the WMI provider).
The WRITE operations in the WMI repository were not completed (the write operation started to write temporary classes, but the temporary classes were not removed).
The information collection for the DCOM, WMI and WMIADAPTER event log entries was skipped (because of an issue with the WMI provider).
New event log entries for DCOM, WMI and WMIADAPTER were created during the WMI Diagnosis Tool

execution.

Some non-critical registry key configurations for DCOM and/or WMI were reported.

Some additional registry keys were created for DCOM when they are not supposed to be present.

ABORT (ErrorLevel = 4)

WMI Diagnostics Tool has been run during the same day (only if the *RunOnce* command line parameter is specified)

WMI Diagnostics Tool is started on an unsupported build or OS version

WMI Diagnostics Tool has no Administrative privileges (can be overwritten with the *Force* command line parameter).

WMI Diagnostics Tool is started in Wow environment (64-bit systems only. WMI Diagnostics Tool must be run in the native 64-bit mode, not in the Wow environment).

Sample command line arguments

- To examine a specific WMI namespace with its sub-namespaces:

```
C:\>WMIDiag BaseNamespace=Root\CIMv2
```

- To deploy the WMI Diagnosis Tool across several computers and prevent any popup windows from interfering with the user:

```
C:\>WMIDiag Silent
```

- To prevent the WMI Diagnosis Tool from echoing messages to the user or the console:

```
C:\>WMIDiag Silent NoEcho
```

- To check the CIM repository consistency:

```
C:\>WMIDiag CheckConsistency
```

Note: This switch checks the repository consistency at the page level. However, when used in Windows XP SP2, an inconsistent repository will automatically be rebuilt during the next reboot and after executing this test. **Because of that, you must be careful when you use this switch and ensure that you have a backup of the repository.** In Windows Server 2003 SP1 and above, this switch is not required, as the consistency check is always made (and does not rebuild the repository when it is inconsistent). The CIM repository consistency check is only supported on Windows XP SP2 (optional), Windows Server 2003 SP1 (always executed), and later versions of Windows such as Vista (always executed).

```
C:\>WMIDiag WriteInRepository
```

Note: This test increases the repository size by 1 MB. In addition, the temporary information will be written in 10 temporary dedicated namespaces. You must have a minimum of 10 MB free space available to execute this test.

- To write test information in the CIM repository in existing namespaces:

```
C:\>WMIDiag WriteInRepository=Root
```

Note: This test increases the repository size by 1 MB or more depending of the number of namespaces. You must have a minimum of 10 MB free space available to execute this test.

Note: Specifying a namespace enables the write operations in existing namespaces. Therefore, if *WriteInRepository=Root* is specified, all namespaces from Root and below will be used to write temporary information. Specifying *WriteInRepository=Root\CIMv2* will only write temporary information in the Root\CIMv2 namespace and below.

- To increase the amount of information displayed on the screen while running the WMI Diagnosis Tool:

```
C:\>WMIDiag LoggingLevel=4
```

Note: All information is logged in the LOG file regardless of the value of the *LoggingLevel* setting.

- To create the LOG, TXT and CSV files in a location other than %TEMP%:

```
C:\>WMIDiag LogFilePath=\\RemoteServer\RemoteShare
```

Note: You must have write permission at the selected location. If you do not, this switch prevents WMI Diagnosis Tool from running.

- To log WMI Diagnosis Tool error messages in the event log as well as the LOG file:

```
C:\>WMIDiag LogNTEventErrors
```

Note: Too many encountered errors could overflow the application event log.

- To log all WMI Diagnosis Tool messages (including error messages) in the event log as well as the LOG file:

```
C:\>WMIDiag LogNTEvents
```

Note: Due to the verbose nature of the WMI Diagnosis Tool, this could overflow the application event log.

- To create an event log entry containing the WMI Diagnosis Tool execution state (Success, Warning or Error) in the application event log:

```
C:\>WMIDiag LogWMIState
```

- To deploy the WMI Diagnosis Tool with SMS and capture LOG, TXT and CSV created for each machine in a central location, while logging the final state of the WMI status in the local machine event log:

```
C:\>WMIDiag SMS LogFilePath=\\RemoteServer\RemoteShare Silent NoEcho LogWMIState
```

- To send WMI Diagnosis Tool information to Microsoft Corporation

```
C:\>WMIDiag SMTPServer=MyIntranetSMTPRelay.MyDomain.Com SMTPFrom=YourEmailAddress@MyDomain.Com
```

Note: The target SMTP address (WMIDiag@Microsoft.Com) is preset. WMIDiag@Microsoft.Com is the SMTP address where the LOG, TXT and CSV files must be sent. **By specifying the *SMTPServer* and sending WMI Diagnosis Tool data to this address you explicitly give consent to send this information to Microsoft. You must specify your email address (*SMTPFrom* command line parameter).**

- To send WMI Diagnosis Tool information to a mailbox in your organization over SSL:

```
C:\>WMIDiag SMTPServer=MyIntranetSMTPRelay.MyDomain.Com SMTPSSL  
SMTPFrom=YourEmailAddress@MyDomain.Com SMTPTo=SMTPTargetAlias@MyDomain.Com
```

- To send WMI Diagnosis Tool information to a mailbox in your organization only when the WMI state is in warning or error:

```
C:\>WMIDiag SMTPServer=MyIntranetSMTPRelay.MyDomain.Com  
SMTPFrom=YourEmailAddress@MyDomain.Com SMTPTo=SMTPTargetAlias@MyDomain.Com SmtpWMIInvalidState
```

- To perform a WMI Diagnosis Tool full test pass (CPU intensive, may take several hours)

```
C:\>WMIDiag WriteInRepository RequestAllInstances ShowMOFErrors
```

Note: The *RequestAllInstances* parameter is **very time-consuming and CPU-intensive**. It is recommended that you run this test only in situations where the WMI infrastructure and associated installed applications need to be extensively tested. It is **NOT** recommended that you execute this test in a wide deployment scenario such as SMS. In addition, this test should be run interactively. You can limit the number of instances requested by specifying the nature of instance requested. For instance *RequestAllInstances=Static* will requests only the static instances, *RequestAllInstances=Dynamic* will requests only the dynamic instances and *RequestAllInstances=StaticAndDynamic* will requests both the static and dynamic instances (the same as *RequestAllInstances* without any additional parameters)

- To run WMI Diagnosis Tool despite the lack of privileges:

```
C:\>WMIDiag Force
```

Note: Using the *Force* parameter could produce a number of access denied errors if the user executing WMI Diagnosis Tool does not have the rights required to perform the operations executed by the tool.

- When deployed using SMS you should turn the ERRORLEVEL return code off and ensure that no popup window occur during the WMI Diagnosis Tool execution. The *SMS* parameter ensures that ERRORLEVEL is always set to 0 at the end of the tool execution. It also ensures that the *NoEcho* and *Silent* parameters are enabled.

```
C:\>WMIDiag SMS
```

- The *OldestLogHistory* parameter ensures that the LOG, TXT and CSV files created in the *LogFilePath* folder for the examined system are not older than a specified number of days. In the following example the WMI Diagnosis Tool deletes all the LOG, TXT and CSV files for the examined system that are older than 10 days:

```
C:\>WMIDiag OldestLogHistory=10
```

Note: This parameter can be used to perform housekeeping when running the WMI Diagnosis Tool on a regular basis.

WMI Diagnosis Tool general questions

1. Where can I get the WMI Diagnosis Tool?

The WMI Diagnosis Tool can be downloaded from the Microsoft Download Center at

<http://go.microsoft.com/fwlink/?LinkId=62562>. More information about the WMI Diagnosis Tool usage can be found at these Internet locations:

- WMI Diagnosis Tool webcast:
<http://msevents.microsoft.com/CUI/EventDetail.aspx?EventID=1032290320&Culture=en-US>
- WMI Diagnosis Tool usage:
http://www.microsoft.com/technet/scriptcenter/topics/help/WMI_Diagnosis_Tool.aspx
- WMI Troubleshooting:
<http://www.microsoft.com/technet/scriptcenter/topics/help/wmi.aspx>

2. Are there any plans to include the WMI Diagnosis Tool in new versions of Windows?

There is no plan to include the WMI Diagnosis Tool in the future versions of Windows. However, the WMI Diagnosis Tool package will remain available from the Microsoft Download Center.

3. Is there a WMI Diagnosis Tool version for Longhorn Server?

WMI Diagnosis Tool version 2.0 supports Longhorn Server. Note that, under Longhorn Server, false positives might be reported because the operating system is still under development. WMI Diagnosis Tool version 2.0 supports Vista and all previous versions of Windows since Windows 2000.

4. Is it possible to "officially" engage Microsoft for feedback on this tool?

There is no official support for WMI Diagnosis Tool. However, feedback for the tool is welcome and can be sent to WMIDdiag@microsoft.com.

5. Did Microsoft develop this tool as a script knowing that a lot of us will enhance it for our own uses?

The WMI team decided to develop this tool as a script simply because the development cycle and maintenance are easier and faster. Moreover, because the scripting community is quite large, we thought that some people would be interested in re-using or enhancing some of the functionality of WMI Diagnosis Tool. Some people have already provided feedback and enhanced the tool on their own.

6. Is there any documentation describing what the WMI Diagnosis Tool does not check?

No. However, this information – for WMI Diagnosis Tool version 1.1 – is briefly mentioned in the [WMI Diagnosis Tool webcast](#). You can refer to section “WMI Diagnosis Tool known limitations” for more information.

7. How much development time went into creating WMI Diagnosis Tool?

The project started in October 2005, a few weeks after the MVP summit in Redmond. The number of hours spent developing WMI Diagnosis Tool has not been tracked because it was a side project and not one of our normal activities. However, we estimate a development time of 4 months, with an average of 16-to-20 hours per week for one person for version 1.1. This includes the analysis, the problem detection pattern studies, the development and the testing of the tool. An additional period of 4 months (10 hours a week) has been invested to enhance the tool based on customer feedback.

Today, the WMI Diagnosis Tool is a script of more than 30,500 lines, including 12,000 lines of code logic (implementing 142 functions) and over 18,500 lines of WMI hard-coded information describing expected setup information.

8. Are there plans to document the top problems customers find with WMI Diagnosis Tool?

Currently, there are no such plans. It is important to note that obtaining a valid list of top problems requires customer feedback. Therefore, we are counting on the community to provide feedback. If we get enough feedback, we may revisit this.

9. Do you plan to write future versions of WMI Diagnosis Tool in compiled code instead of a scripted version?

There are no such plans. New versions of WMI Diagnosis Tool will be written in VBScript.

10. Is it possible to store WMI Diagnosis Tool information in a SQL database instead of a LOG file?

Technically, anything is possible. Because the WMI Diagnosis Tool is a script, nothing prevents you from adding this feature to meet your needs. Today, there are no plans to include this capability in WMI Diagnosis Tool. However, the CSV file created by the WMI Diagnosis Tool can be easily loaded into a pre-created SQL database. The Excel spreadsheet included with the WMI Diagnosis Tool download can help you understand the structure of the CSV file, which is also described further in this document.

11. Is there a release date planned for the next WMI Diagnosis Tool version?

There is no release date planned for the next version of WMI Diagnosis Tool after version 2.0

12. Is there an e-mail address to send feature requests for future versions?

All communication related to WMI Diagnosis Tool should be sent to WMIDiag@Microsoft.Com.

13. What are the new features planned for the next WMI Diagnosis Tool version?

Although none of the features planned were confirmed at the time of the [WMI Diagnosis Tool web cast](#) for version 1.1, we mentioned that features such as the following would be considered:

- NT Event Log Events analysis for WINMGMT messages
- Support for DCOM and WMI namespace security deciphering and analysis (Security Descriptor)
- Tighter integration with SMS by adding tests for specific SMS WMI classes.

The feedback has been well received and these features have been implemented for version 2.0. Refer to the section “WMI Diagnosis Tool 2.0 Updates” for more information.

14. Is there an Internet BLOG for WMI Diagnosis Tool?

There is no BLOG specifically for WMI Diagnosis Tool. However, the WMI team has a BLOG at <http://blogs.msdn.com/wmi/default.aspx> if time permits.

WMI Diagnosis Tool usage questions

1. How do you use the Excel spreadsheet that comes with the WMI Diagnosis Tool?

Each time the WMI Diagnosis Tool is executed a LOG, a TXT and a CSV file are created. By default, these files are created in the %TEMP% directory. To easily collect all the CSV files created on all computers, it is recommended that

you redirect the files to a central location. This can be done with the *LogFilePath* command-line parameter by specifying a UNC path (share). Once all the CSV files are centrally available, it is very easy to load these files into the WMI Diagnosis Tool Excel spreadsheet. Simply do the following:

- From the central location, execute

```
Copy /a \\MyCentral_Server\MyShare\*.CSV All.CSV
```

- Load the file ALL.CSV in Excel
- Load the WMI Diagnosis Tool Excel spreadsheet in Excel (WMIDiag.xls comes with the WMI Diagnosis Tool package)
- Copy the content of the ALL.CSV Excel spreadsheet in the clipboard
- Paste the clipboard content in the DATA tab of the WMI Diagnosis Tool Excel spreadsheet

Once completed, the WMI Diagnosis Tool Excel spreadsheet automatically adjusts all calculations and associated graphics in other Excel sheet panes.

The CSV structure is described in the table below.

Field #	Description	Field #	Description	Field #	Description
1	WMIDiagVersion	32	FirewallActive	63	WMIExecQueryErrors
2	Started	33	RemoteAdmin	64	WMIExecQuerySkipped
3	Date	34	UNSECAPP	65	WMIGetValuePropertyErrors
4	Started Time	35	InvalidDCOMSetup	66	WMIGetValuePropertySkipped
5	LastRun	36	InvalidWMIServiceSetup	67	WriteInRepositoryFailures
6	IsServerOS	37	WMIServiceDependents	68	WMIPutErrors
7	Is64	38	InvalidWMIServiceDCOMSetup	69	WMIDeleteErrors
8	IsWow64	39	MissingWMIDCOMRegistrations	70	DynamicInstances
9	ProcessorArchitecture	40	WMIMissingProgID	71	StaticInstances
10	ProcessorIdentifier	41	HighPrivWMIProvider	72	AsyncCancellations
11	NTVersion	42	InvalidWMIProviderDCOMRegistration	73	DCOM Events before execution
12	NTBuild	43	InvalidWMIProviderCIMRegistration	74	WMI Events before execution
13	NTServicePack	44	InvalidWMIProviderCLSID	75	WMIAdapter Events before execution
14	ProductFullName	45	MissingWMIProviderFile	76	DCOM Events after execution
15	ProductShortName	46	UserAccountControl	77	WMI Events after execution
16	Locale	47	LocalAccountFiltering	78	WMIAdapter Events after execution
17	MachineName	48	DCOMSecurityErrors	79	MissingAutoRecoveryMOFFiles
18	DomainUsername	49	DCOMSecurityWarnings	80	MissingWBEMMOFFilesFromAutoRecovery
19	NoPrivilege	50	WMI SecurityErrors	81	MissingWMIProviderMOFFile
20	BaseNamespace	51	WMI SecurityWarnings	82	MissingPragmaAutorecoverMOFFile
21	Environment	52	PermanentSubscriptions	83	Ended Date
22	DiskInterface	53	TimerInstructions	84	Ended Time
23	FilePresenceIssues	54	ADAPStatus	85	# of InProcProviders
24	AdditionalBinariesInWBEMFolder	55	WMIConnectionEncryption	86	# of OutOfProcProviders
25	MissingWMI SytemFiles	56	WMI MonikerConnectionErrors	87	# of DecoupledProviders
26	MissingRepositoryFiles	57	WMIConnectionErrors (Connect)	88	ExecutionTime (sec)
27	WMIRepositoryConsistency	58	WMIGetErrors	89	Total warnings
28	RepositorySize (MB) - BEFORE WMIDiag	59	MOFRepresentationErrors	90	Total errors
29	freespace (MB) - BEFORE WMIDiag	60	QualifierAccessError	91	WMICriticalityLevel
30	RepositorySize (MB) - AFTER WMIDiag	61	WMIEnumerationErrors	92	LOGFile
31	Disk freespace (MB) - AFTER WMIDiag	62	WMIEnumerationSkipped	93	LOGFilePath

2. How do I know what to fix once the WMI Diagnosis Tool has executed successfully?

The WMI Diagnosis Tool creates a LOG file in the %TEMP% folder by default. This LOG file contains a WMI Report (search for the “WMI REPORT: BEGIN” string). The same report is also created in the .TXT file. By default, any error or warning reported by the tool will automatically load this .TXT report in Notepad. If you carefully read that report, the WMI Diagnosis Tool will suggest which kind of action can be taken to fix the reported problems; some of the suggested actions even contain command line samples you can execute in order to fix a problem. Sometimes several actions can be reported for a single problem. These actions are usually reported in order of importance and should be undertaken in the same order as presented in the report.

3. Can you run the WMI Diagnosis Tool in a RUNAS command?

Yes. Just make sure that you correctly specify the RUNAS parameters, the CSCSCRIPT.EXE parameters (the Windows Script Host engine running WMIDdiag.vbs), the WMI Diagnosis Tool script full access path, and the WMI Diagnosis Tool command-line parameters. For instance, the RUNAS command might look as follows (the command line below must be typed on one single line):

```
Runas /User:MyDomain\MyAdminUser
      "Cscript.Exe C:\WMIDdiag\WMIDdiag.vbs
      Basenamespace=Root\Default"
```

4. Can the WMI Diagnosis Tool diagnose a remote computer?

The WMI Diagnosis Tool is not designed to diagnose remote computers. This is due to the fact that WMI remote access is mainly based on the WMI infrastructure. Because the aim of WMI Diagnosis Tool is to diagnose WMI, the WMI Diagnosis Tool does not use WMI to perform its core operations. That’s why the WMI Diagnosis Tool must be run locally. However, the WMI Diagnosis Tool can be deployed remotely using Group Policy, Systems Management Server (SMS), or Microsoft Operations Manager (MOM) via a Management Pack. With Windows Vista, the WMI Diagnosis Tool can also be remotely executed through WinRM/WinRS, provided you configure and enable these features (WinRM/WinRS are not enabled by default). Microsoft SysInternals tool PSEXEC.EXE can also be used.

5. Can the WMI Diagnosis Tool be remotely executed with Windows Remote Shell (WinRS) on Windows Vista?

Yes. For example, you can execute WMI Diagnosis Tool on a remote computer with WinRS from a network share, meaning that you do not need to copy WMI Diagnosis Tool to the remote computer. This assumes that WinRM is properly configured and that you have access to the remote share on the remote computer.

For example, to store LOG, TXT and CSV files on a dedicated share and send WMI Diagnosis Tool failures only (Warnings and Errors) to Microsoft Corporation via email, the command line must be as follows (note that the command line must be typed on one single line):

```
WinRS -r:http://MyRemoteServer
      -u MyRemoteServer\Administrator -p MyPassword
      "Cscript.Exe
      \\OtherRemoteServer\RemoteShare\WMIDdiag.vbs
      SMTPServer=MyIntranet.SMTPRelay.MyDomain.Com
      SMTPFrom=YourEmailAddress@MyDomain.Com
      LogFilePath=\\OtherRemoteServer\RemoteShare
      SmtpWMIInvalidState"
```

In this example, WINRS uses the Cscript.Exe WSH engine to execute on \\MyRemoteServer the WMI Diagnosis Tool located on the \\OtherRemoteServer\RemoteShare share.

6. Can WMI Diagnosis Tool be remotely executed with PSEXEC.EXE from on a remote computer?

Yes. For example, you can execute WMI Diagnosis Tool on a remote computer with PSEXEC.EXE from a network share, meaning that you do not need to copy WMI Diagnosis Tool to the remote computer. This assumes that you have access to the remote share on the remote computer.

For example, to store LOG, TXT and CSV files on a dedicated share and send WMI Diagnosis Tool failures only (Warnings and Errors) to Microsoft Corporation via email, the command line must be as follows (note that the command line must be typed on one single line):

```
Psexec.exe \\MyRemoteServer
-u MyRemoteServer\Administrator -p MyPassword
Cscript.Exe
\\OtherRemoteServer\RemoteShare\WMIDdiag.vbs
SMTPServer=MyIntranetSMTPRelay.MyDomain.Com
SMTPFrom=YourEmailAddress@MyDomain.Com
LogFilePath=\\OtherRemoteServer\RemoteShare
SmtPWMIInvalidState
```

In this example, PSEXEC.EXE uses the Cscript.Exe WSH engine to execute on \\MyRemoteServer the WMI Diagnosis Tool located on the \\OtherRemoteServer\RemoteShare share.

7. Can WMI Diagnosis Tool be run using “WMIC process call create”?

Yes. For example, you can execute WMI Diagnosis Tool on a remote computer with WMIC from a network share, meaning that you do not need to copy WMI Diagnosis Tool to the remote computer. This assumes that you have access to the remote share on the remote computer.

For example, to store LOG, TXT and CSV files on a dedicated share and send WMI Diagnosis Tool failures only (Warnings and Errors) to Microsoft Corporation via email, the command line must be as follows (note that the command line must be typed on one single line):

```
WMIC.EXE /NODE:MyRemoteServer /User:MyRemoteServer\Administrator /PASSWORD:MyPassword
Process Call Create "Cscript.Exe
\\OtherRemoteServer\RemoteShare\WMIDdiag.vbs
SMTPServer=MyIntranetSMTPRelay.MyDomain.Com
SMTPFrom=YourEmailAddress@MyDomain.Com
LogFilePath=\\OtherRemoteServer\RemoteShare
SmtPWMIInvalidState"
```

In this example, WMIC.Exe uses the Cscript.Exe WSH engine to execute on \\MyRemoteServer the WMI Diagnosis Tool located on the \\OtherRemoteServer\RemoteShare share.

8. Does the WMI Diagnosis Tool fix problems it discovers?

No. The WMI Diagnosis Tool executes in read-only mode. Even though the WMI Diagnosis Tool diagnoses the situation and provides procedures to fix problems, at no time does the tool automatically fix a problem. This is by design, because the correct repair procedure depends on the context, the usage, and the list of applications installed on the computer.

9. By limiting WMI Diagnosis Tool to check only one namespace, are you missing some possible error detections?

Yes. However, after running WMI Diagnosis Tool once without any namespace limitations, and if WMI Diagnosis Tool reports problems for only one single namespace, it is safe to limit the verification to the namespace reporting

problems. However, this approach assumes that no new problems are created on the computer. The *BaseNamespace* command-line parameter scopes the browse operation of the WMI repository to the namespace specified. Therefore only the WMI provider registrations related to that namespace will be verified. However, irrespective of the *BaseNamespace* command-line parameter, WMI Diagnosis Tool always performs the verification of all WMI core components in the system. Therefore, the *BaseNamespace* command line parameter only affects the WMI operations related to the repository browsing. Refer to the section of this paper titled “The sequence of operations” for more information about non-WMI operations and core WMI operations.

10. If the repository has been deleted and the computer still reports WMI errors, can this situation be resolved with the WMI Diagnosis Tool?

Yes, depending on the problem. If the repository has been deleted, it will be rebuilt when the WMI Diagnosis Tool is run, because the tool starts the WMI service. This, in turn, rebuilds the repository. The list of MOF files recompiled during the reconstruction re-registers the WMI providers in the WMI repository. If a DCOM registration is missing for a provider, WMI Diagnosis Tool is capable of detecting the error even after a repository reconstruction.

However, some static classes specific to a third-party application may be missing because the MOF file was not compiled during the reconstruction process. In this case WMI Diagnosis Tool will not be able to discover the error because the tool has no knowledge of third-party applications other than the WMI infrastructure itself. MOF files that do not contain the [#PRAGMA AUTORECOVER](#) statement are not recompiled during the auto-recovery process. The missing classes from a third-party MOF can cause the associated applications to fail.

11. Can the WMI Diagnosis Tool identify the problem and root causes of the repository that is growing very large (+300 MB)?

No. The WMI Diagnosis Tool is intended to verify the setup and the core elements of WMI based on information in the registry and the WMI repository. To trace all activities of WMI at runtime WMI logging/tracing must be used instead. Tracing runtime activities is not the purpose of the WMI Diagnosis Tool.

12. Does the WMI Diagnosis Tool report on any security settings or problems with rights?

Yes, the version 2.0 of the WMI Diagnosis Tool verifies the DCOM and WMI namespace security settings.

13. Does the WMI Diagnosis Tool output *ERRORLEVEL* codes?

Yes, it does. When the WMI Diagnosis Tool terminates, the *ERRORLEVEL* environment variable is set to a specific value:

- 0 = SUCCESS
- 1 = ERROR
- 2 = WARNING
- 3 = Command Line Parameter errors
- 4 = User Declined (Clicked the Cancel button when getting a consent prompt)

Note that if you use the *SMS* parameter, then WMI Diagnosis Tool does not set the *ERRORLEVEL* parameter. This functionality exists to prevent confusion between the failure of the execution of the tool and the state of WMI as reported by the tool after a complete and successful execution.

14. How can I determine if I am using the latest version of the WMI Diagnosis Tool?

The WMI Diagnosis Tool version 1.10 is dated from the 8th of March 2006 at 13:26.

The latest WMI Diagnosis Tool version is 2.00 and is dated from the 16th of January 2007 at 3:34pm.

It is recommended that you use the latest WMI Diagnosis Tool version (2.0). Many enhancements have been made in this version for all operating system from Windows 2000 on. See the section of this paper titled “WMI Diagnosis Tool 2.0 Updates” for more information.

15. If Version A of WMIy.dll and Version B of WMIx.dll are present on a system, does the WMI Diagnosis Tool tool detect that scenario?

No. However, Windows File Protection handles problems regarding invalid versions of the Windows system files. Note that the LOG file contains an inventory with more detailed information about the actual WMI binaries. See the section “Collecting system information about WMI binary files” for more information.

16. Will the log have a unique name, or can you specify a log name (perhaps with computer name as a variable)?

No. WMI Diagnosis Tool names the LOG, TXT and CSV files itself by using a combination of the computer name and several other parameters to guarantee the uniqueness of the created files. However, the location used to create the files can be overwritten. By default, LOG, TXT and CSV files are created in %TEMP%. This can be modified with the *LogFilePath* command-line parameter by specifying a folder or share if required. The LOG, TXT and CSV filename format is as follows:

```
WMIDIAG-vv.vv_wwwww.ppp.sss.aa_COMPUTERNAME_yyyy.mm.dd_hh.mm.ss
```

Where

- vv.vv is the WMI Diagnosis Tool version (for example v1.11 or v2.00)
- wwwwww is the Windows version (for example XP___, 2003_, 2000_, VISTA)
- ppp is the service pack level (for example SP1)
- sss is the platform type (for example SRV for server, CLI for client).
- aa is the processor architecture (for example 32 or 64).
- COMPUTERNAME is the computer name of the system analyzed.
- yyyy.mm.dd is the date of the WMI Diagnosis Tool run
- hh.mm.ss is the time of the WMI Diagnosis Tool run

For instance,

```
WMIDIAG-V2.00_XP___.CLI.SP2.32_SERVER02_2006.03.08_04.52.42
```

17. When running the WMI Diagnosis Tool, can we combine multiple command line parameters?

Yes. For example, to deploy the WMI Diagnosis Tool with SMS, store LOG, TXT and CSV files on a share and send WMI Diagnosis Tool failures (Warnings and Errors) to Microsoft Corporation via email, use the following command line (the command line below must be typed on one single line):

```
WMIDIag SMTPServer=MyIntranetSMTPRelay.MyDomain.Com SMTPFrom=YourEmailAddress@MyDomain.Com  
SMS LogFilePath=\\RemoteServer\RemoteShare SmtPWMIInvalidState
```

18. How frequent should the WMI Diagnosis Tool should be used?

It is recommended that you run WMI Diagnosis Tool at least once in your organization. By doing so you will be able to assess the situation, identify and fix problems that have not yet been discovered. If any problems are found and fixed, you should then run WMI Diagnosis Tool once a week or once a month as a preventive action to detect and fix any new issue as it happens that some applications are modifying the security policies and/or the configuration of the system in such a way that they impact WMI.

19. Is the WMI Diagnosis Tool a good utility to run on Windows 2000 computers that are going to be upgraded to Windows Server 2003 using the upgrade path?

If you plan to upgrade Windows 2000 computers to Windows XP SP2, Windows Server 2003 and/or Windows Vista, there is no need to run WMI Diagnosis Tool before upgrading. During the upgrade process, the DCOM registration of all WMI components is executed, which automatically fixes all DCOM issues if any. Because the WMI repository format changes from Windows 2000 to Windows XP SP2/Windows Server 2003 and Windows Vista, a new repository is created. For example, if you upgrade to Windows Vista, the repository is converted from the old repository format to the new format. Therefore, the actual repository is verified during the conversion process. Under Windows Vista, the old repository content is actually migrated into the native Windows Vista repository. Once the upgrade is completed, it is a good idea to run the WMI Diagnosis Tool to assess the WMI setup. However, this is not mandatory.

WMI Diagnosis Tool and SMS Questions

1. Does the WMI Diagnosis Tool work with the SMS 2003 Advanced Client?

The WMI Diagnosis Tool works regardless of any installed software consuming or providing WMI information. SMS 2003 Advanced Client is no exception. The SMS 2003 Advanced Client registers WMI providers in the WMI repository; thus the registration will be verified like any other registration. The version 2.0 of WMI Diagnosis Tool has specific knowledge of the SMS 2003 Advanced Client; therefore, if an SMS 2003 Advanced Client WMI class is missing, the WMI Diagnosis Tool will report the problem as it would for the core WMI classes. Last but not least, the WMI Diagnosis Tool also executes a series of WMI tests regarding the SMS Advanced Client. These tests are expected to succeed if the SMS Advanced Client is healthy. Any WMI failure of the SMS Advanced Client will be reported.

2. Can the WMI Diagnosis Tool find serious WMI problems if the SMS client needs a healthy WMI infrastructure to work?

Yes, WMI problems can exist that do not impact the SMS 2003 Advanced Client. For instance, Resultant Set of Policy (RSOP), which is based on WMI, can suffer from DCOM registrations that do not impact the SMS 2003 Advanced Client. This principle can be the same for any components or software using WMI. WMI coverage is quite large in Windows, and some components can work fine while others do not simply because their dependencies are different.

3. Is there a best practice for WMI Diagnosis Tool parameters in order to find common errors when deployed via SMS 2003?

The WMI Diagnosis Tool default settings are configured in such a way that the tool finds most common errors. When deployed with SMS, the *SMS* command-line parameter is recommended, mainly to prevent screen echoes and window popup during execution. If you want to collect the WMI Diagnosis Tool LOG, TXT and CSV files in a central location, use the *LogFilePath* command-line parameter. This makes it easy to consolidate all results in the WMI Diagnosis Tool Excel spreadsheet as described previously in this document.

4. Does Microsoft recommend that customers send in any WMI failures that occur from SMS clients?

Yes. It is always helpful to get direct feedback from our customers. This is also a good way for customers to directly contribute to the development of both WMI and the WMI Diagnosis Tool. By leveraging customer feedback and experience we will improve customer satisfaction. Listening to problems and seeing how we can improve the product across service packs and new versions of Windows is our most important goal.

WMI Diagnosis Tool and MOM Questions

1. Are there any WMI Diagnosis Tool Management Packs for MOM 2005?

At this time there is no specific MOM Management Pack for the WMI Diagnosis Tool. Because the WMI Diagnosis Tool is a tool that uses advanced WMI features such as asynchronous calls. Therefore, it can't be executed as a management pack. However, the WMI Diagnosis Tool could be leveraged by a management pack acting as a parser on the WMI Diagnosis Tool results produced in the LOG. We know that some customers did this by using third-party management software and reporting tools.

Based on feedback we may eventually provide a WMI Diagnosis Tool MOM Management Pack but there is no plan to do so at this time.

WMI questions

1. Why is WMI broken as often as it is?

WMI is a pervasive technology in Windows, one that interacts with a very large number of Windows components. These components provide information to WMI or consume WMI information. In both cases, the components rely directly or indirectly on WMI. If one of these components fails, even for a simple reason such as insufficient access privileges or an invalid request, WMI will return an error. However, the error returned to the WMI consumer or the provider does not necessarily indicate that WMI is broken. To isolate the error source, you must differentiate between WMI native errors and errors returned by WMI that originate in other components. To give a brief example of error categories:

- 0x800410xx and 0x800440xx are WMI errors. These error codes mean that a specific WMI operation failed. This could be due insufficient privileges to perform the WMI requested operation; due to the nature of the request itself (for example, no WMI rights, bad WQL query); or due to a WMI infrastructure issue, such as CIM or WMI DCOM registration preventing a provider from being loaded.
- 0x8007xxxx are Win32 errors, not WMI errors. WMI may return these types of error due to an external failure (for example, DCOM security).
- 0x80040xxx are pure DCOM errors, not WMI errors. WMI may return these types of error due to an external failure (for example DCOM configuration)
- 0x80005xxx are ADSI errors (LDAP), not WMI errors. WMI may return these types of errors due to an external failure, such as an Active Directory access failure when using the WMI Active Directory providers.

Because WMI can return errors from other Windows components, WMI appears to be the root cause of all the failures reported, which is a misperception. For instance, if a consumer issues a WMI request that fails due to a DCOM registration issue, WMI returns an error stating that the WMI provider cannot be loaded (WBEM_E_PROVIDER_LOAD_FAILURE). Obviously, this prevents the WMI operation from succeeding even though the problem is in the DCOM registration (which in this case can be fixed with the REGSVR32.EXE command).

Another example is related to the Wbem_E_Not_Found message. This message generally appears in response to a missing WMI class in the WMI repository. The requested WMI class can be missing for many reasons. For instance:

- A misspelled WMI class name issued by the consumer.
- A correct WMI class name is used but the request attempts to connect to a WMI namespace where this class does not exist.
- An improper or incomplete application installation, which failed to create all expected classes.
- A MOF file containing the class definition that was not (re)compiled after a repository reconstruction because the MOF didn't contain the [#PRAGMA AUTORECOVER](#) statement.
- A WMI repository corruption preventing location of the requested WMI class.

In this last list of examples, WMI returns the same error message because the symptoms are the same. However, only the WMI repository corruption cause is a pure WMI infrastructure issue.

To help people understand the various errors and problems that can be encountered Microsoft developed the WMI Diagnosis Tool to ensure that the right course of action is taken. Deleting the WMI repository **is not** the right course of action, at least not as a **first** action for most reported problems! Deleting the repository can create additional problems if not handled properly. The aim of WMI Diagnosis Tool is to help people to understand and fix problems they encounter. Repository deletion and reconstruction is recommended only when truly needed and must be a **last** resort.

2. Why did Microsoft develop the WMI Diagnosis Tool instead of fixing WMI?

Microsoft is working on two fronts. One front consists of addressing customer concerns and feedback for the **new** release, the other front consists of addressing customer concerns for **existing** releases. In the first case, Microsoft is making many investments to improve the WMI infrastructure, just like we do with any other Windows components. In the second, Microsoft is both helping customers to address problems they face in existing products and getting feedback on those problems at the same time. This second case is the aim of the WMI Diagnosis Tool. As Microsoft gains additional understanding and feedback about all problems encountered, the appropriate course of action will be taken to address these in both existent and future versions of the products and tools.

3. Is there a tool or a method to uninstall/reinstall (refresh) all components (and non-core-OS dependencies) of WMI?

No. However, the WMI Diagnosis Tool will tell you if there are WMI core files missing or incorrectly registered in the system and what to do to fix the situation. Some WMI dependencies are also analyzed by the WMI Diagnosis Tool such as DCOM and RPCSS.

4. Can you shed some more light on why WMI returns an 0x8004001 error so frequently?

WMI returns a large number of error codes. Despite the fact that this series of error codes covers many specific situations, there are still cases that do not fall into a pre-defined category. Cases outside of these pre-defined categories generally return the 0x8004001 error code. After 10 years of experience deploying WMI (WMI has existed since the Windows NT 4.0 era), the number of cases outside of the pre-defined categories has increased. During the design of the WMI core infrastructure 10 years ago, most of these cases didn't exist or were minimal and considered edge cases. In this context, WMI has obviously improved in many areas. However, it is very complex and risky to revamp the error management sub-system while maintaining a 100% backward compatibility with existing applications. This is why WMI still returns so many "generic" errors today.

5. *Where are WMI errors documented?*

WMI errors are documented on [MSDN](#). The WMI Diagnosis Tool also helps to understand what to do with most common errors reported.

6. *If we have been deleting our repository as a way to remedy problems, could we have damaged anything in the process?*

When you delete the repository WMI recovers most of the information the deleted repository contained, provided that the MOF files used to rebuild the repository are part of the auto-recovery process. If there are MOF files not included in this process (because they didn't include the '[#PRAGMA AUTORECOVER](#)' statement at installation time), then some components that utilize WMI may not work anymore. Generally speaking, the type of lost information can be classified into three categories:

1. **Registration.** The registration of the WMI providers supporting a set of WMI classes added to the system at installation time. This information can be recovered by re-compiling the MOF files if these files are not in the auto-recovery process.
2. **Classes .** The set of WMI classes itself, which is used by WMI consumers or the application itself. This information can be recovered by re-compiling the MOF files if these files are not in the auto-recovery process.
3. **Static Information.** Static information created by the application at run-time and stored in the repository. This static information may not exist in the reconstructed repository. This depends on the application itself and the way it uses the repository. Generally, this information is not contained in any MOF files and the auto-recovery process does not re-create this information. Therefore, this information is lost. However, depending on the application behavior, this information might be recreated by the application. However, this depends on the application itself.

Items 1 and 2 can be easily fixed as long as you have access to the MOF files included with the application. These are not necessarily stored in the %SystemRoot%\System32\WBEM folder. Instead they might be located in the folder where the application is installed. If no MOF file available, that means that the repository content for the application was programmatically created during the setup of the application. If this is the case then there is no choice but to reinstall the application.

Some applications provide MOF files but also have their own recovery approach. For instance, the SMS Advanced Client provides MOF files but does not list these MOF files in the WMI auto-recovery process. This is because the CCMRepair.Exe tool, located in %SystemRoot%\System32\CCM, automatically reinstalls the entire agent when information of any kind is missing, including WMI repository information.

As for static information in the repository, if the application stores information in the repository at run-time, this information will be lost and must be recreated.

Note: The WMI repository is part of the System State backup. Even though it is easy to obtain a System State backup of servers, it is generally much more complex to obtain and manage the System State backup of thousands of workstations.

This is why the deletion of the repository must always be the **last** course of action and should be performed only when the WMI repository is truly corrupted. This prevents the administrator from having to locate and recompile the MOF files installed with applications or reinstall those applications.

The WMI Diagnosis Tool checks the repository consistency with the **CheckConsistency** command line parameter. This validation is only supported under Windows XP SP2, Windows Server 2003 SP1 and later versions of Windows.

7. **How can we determine if the repository is actually corrupted?**

Systems running Windows XP SP2, Windows Server 2003 SP1 and later can determine if the repository is corrupted because the required instrumentation to do so is available in the WMI infrastructure.

Other operating system versions and Service Packs do not have the required instrumentation and thus do not offer a way to make that determination. Therefore, it is highly recommended that you upgrade systems to Windows XP SP2, Windows Server 2003 SP1 or later. This, of course, is in addition to the fixes that these platforms bring regarding repository management.

The WMI Diagnosis Tool leverages this instrumentation when available and returns the state of the repository in the report:

```
.4979 20:17:19 (0) ** WMI repository state: ..... CONSISTENT.
```

In Windows XP SP2, if the repository is determined to be inconsistent, it is automatically rebuilt during the next reboot. Because this is a Windows XP SP2 design and not a WMI Diagnosis Tool design, by default WMI Diagnosis Tool verifies the repository consistency under Windows XP SP2 only when the *CheckConsistency* command-line parameter is specified. In Windows Server 2003 SP1 and above, this switch is not required, as the consistency check is always made. On these platforms, the repository is not automatically rebuilt when it is found inconsistent.

8. **How does the WMI repository get re-created?**

The repository gets recreated when it does not exist at the location where WMI expects to find it (at %SystemRoot%\System32\WBem\Repository). The recreation process leverages a registry key called “Autorecover MOFs” located at HKLM\SOFTWARE\Microsoft\WBEM\CIMOM. This key contains the list of MOF files to be recompiled with MOFCOMP when the reconstruction process takes place. For a MOF to be listed in that registry key, it must contain the [#PRAGMA AUTORECOVER](#) statement. When the MOF is compiled and when that statement is found during the compilation, the registry is updated with the MOF path file name. If the statement is missing the MOF will not be listed and the definition it contains will be missing from the reconstructed repository. This is why it is recommended that this statement appear in every MOF file used to configure the system.

7. **WMI requests have long latencies, up to 3 seconds. Will this change in the future?**

This depends on the request issued. Normally, WMI does not have a 3-second latency for most requests issued. If this is the case then there is a problem in the installation and we would like to hear about it. For example, suppose you request the list of all Windows services on the local computer using the following WQL statement:

```
Select * From Win32_Service
```

Or using the following WMIC command:

```
WMIC path win32_Service get /Value
```

The response time for these commands must be almost immediate (unless you are using WMIC for the first time, as it installs itself on the first usage). However, suppose you execute the same operation remotely:

```
WMIC /Node:MyRemoteServer path win32_Service get /Value
```

The response might be slower because the list of Windows services comes from a remote location, which introduces latency. Another example is this:

```
Select * From CIM_DataFile
```

Executed locally, this WQL query sample will be much slower than the previous one because it requests the list of all the files that exist on the system, and a system may have thousands of files. If you narrow the scope to the C: drive and only the files located in the root directory, the response will be almost immediate:

```
Select * From CIM_DataFile Where Drive="C:" And Path="\"
```

The nature of the WMI class queried, its scope, and the location where the query is executed greatly influence WMI responsiveness.

WMI is an abstraction layer on top of native APIs: it provides easy access to the management information because it abstracts the use of native APIs with the WMI providers and the CIM class representations. Although WMI implements a query language close to SQL (WQL), it is not a SQL server database and it should not be considered as such. Like any abstraction layer, WMI eases access to information but also introduces additional hops that can impact the performance. By contrast, native APIs are faster, but much more complex to use.

8. ***What are all the MOF files marked with '#PRAGMA AUTORECOVER'?***

Microsoft does not have a complete list of all MOF files that may contain the '[#PRAGMA AUTORECOVER](#)' statement because many third parties and/or administrators create their own MOF files and decide to make them part of the auto-recovery process (or not). However, MOF files included in the operating system default installation are part of the auto-recovery process. MOF files included in a Windows XP SP2 installation and containing the '[#PRAGMA AUTORECOVER](#)' statement are the followings:

```
C:\WINDOWS\SYSTEM32\WBEM\CIMWIN32.MOF
C:\WINDOWS\SYSTEM32\WBEM\CIMWIN32.MFL
C:\WINDOWS\SYSTEM32\WBEM\SYSTEM.MOF
C:\WINDOWS\SYSTEM32\WBEM\WMIPCI.MOF
C:\WINDOWS\SYSTEM32\WBEM\WMIPCI.MFL
C:\WINDOWS\SYSTEM32\WBEM\REGEVENT.MOF
C:\WINDOWS\SYSTEM32\WBEM\REGEVENT.MFL
C:\WINDOWS\SYSTEM32\WBEM\NTEVT.MOF
C:\WINDOWS\SYSTEM32\WBEM\NTEVT.MFL
C:\WINDOWS\SYSTEM32\WBEM\SECRCW32.MOF
C:\WINDOWS\SYSTEM32\WBEM\SECRCW32.MFL
C:\WINDOWS\SYSTEM32\WBEM\DSPROV.MOF
C:\WINDOWS\SYSTEM32\WBEM\DSPROV.MFL
C:\WINDOWS\SYSTEM32\WBEM\MSI.MOF
C:\WINDOWS\SYSTEM32\WBEM\MSI.MFL
C:\WINDOWS\SYSTEM32\WBEM\POLICMAN.MOF
C:\WINDOWS\SYSTEM32\WBEM\POLICMAN.MFL
C:\WINDOWS\SYSTEM32\WBEM\SUBSCRPT.MOF
C:\WINDOWS\SYSTEM32\WBEM\WMI.MOF
C:\WINDOWS\SYSTEM32\WBEM\WMI.MFL
C:\WINDOWS\SYSTEM32\WBEM\WMIPDSKQ.MOF
C:\WINDOWS\SYSTEM32\WBEM\WMIPDSKQ.MFL
C:\WINDOWS\SYSTEM32\WBEM\WMIPICMP.MOF
C:\WINDOWS\SYSTEM32\WBEM\WMIPICMP.MFL
C:\WINDOWS\SYSTEM32\WBEM\WMIPIPRT.MOF
C:\WINDOWS\SYSTEM32\WBEM\WMIPIPRT.MFL
C:\WINDOWS\SYSTEM32\WBEM\WMIPJOB.J.MOF
C:\WINDOWS\SYSTEM32\WBEM\WMIPJOB.J.MFL
C:\WINDOWS\SYSTEM32\WBEM\WMIPSESS.MOF
C:\WINDOWS\SYSTEM32\WBEM\WMIPSESS.MFL
C:\WINDOWS\SYSTEM32\WBEM\KRNLPROV.MOF
C:\WINDOWS\SYSTEM32\WBEM\KRNLPROV.MFL
C:\WINDOWS\SYSTEM32\WBEM\CLI.MOF
C:\WINDOWS\SYSTEM32\WBEM\TSCFGWMI.MOF
C:\WINDOWS\SYSTEM32\WBEM\TSCFGWMI.MFL
C:\WINDOWS\SYSTEM32\WBEM\LICWMI.MOF
C:\WINDOWS\SYSTEM32\WBEM\LICWMI.MFL
C:\WINDOWS\SYSTEM32\WBEM\HNETCFG.MOF
C:\WINDOWS\SYSTEM32\WBEM\SR.MOF
C:\WINDOWS\SYSTEM32\WBEM\DGNET.MOF
```

```

C:\WINDOWS\SYSTEM32\WBEM\IEINFO5.MOF
C:\WINDOWS\SYSTEM32\WBEM\RSOP.MOF
C:\WINDOWS\SYSTEM32\WBEM\RSOP.MFL
C:\WINDOWS\SYSTEM32\WBEM\SCERSOP.MOF
C:\WINDOWS\SYSTEM32\WBEM\SNMPSMIR.MOF
C:\WINDOWS\SYSTEM32\WBEM\SNMPREG.MOF
C:\WINDOWS\SYSTEM32\WBEM\WSCENTER.MOF
C:\WINDOWS\SYSTEM32\WBEM\CLIEGALIASES.MOF
C:\WINDOWS\SYSTEM32\WBEM\CLIEGALIASES.MFL

```

MOF files included in a Windows XP SP2 installation and not containing the ['#PRAGMA AUTORECOVER'](#) are the followings:

```

C:\WINDOWS\SYSTEM32\WBEM\FCONPROV.MFL
C:\WINDOWS\SYSTEM32\WBEM\FCONPROV.MOF
C:\WINDOWS\SYSTEM32\WBEM\NCPROV.MFL
C:\WINDOWS\SYSTEM32\WBEM\NCPROV.MOF
C:\WINDOWS\SYSTEM32\WBEM\SCRCONS.MFL
C:\WINDOWS\SYSTEM32\WBEM\SCRCONS.MOF
C:\WINDOWS\SYSTEM32\WBEM\SMTPCONS.MFL
C:\WINDOWS\SYSTEM32\WBEM\SMTPCONS.MOF
C:\WINDOWS\SYSTEM32\WBEM\TMPLPROV.MFL
C:\WINDOWS\SYSTEM32\WBEM\TMPLPROV.MOF
C:\WINDOWS\SYSTEM32\WBEM\TRNSPROV.MFL
C:\WINDOWS\SYSTEM32\WBEM\TRNSPROV.MOF
C:\WINDOWS\SYSTEM32\WBEM\UPDPROV.MFL
C:\WINDOWS\SYSTEM32\WBEM\UPDPROV.MOF
C:\WINDOWS\SYSTEM32\WBEM\WBEMCONS.MFL
C:\WINDOWS\SYSTEM32\WBEM\WBEMCONS.MOF
C:\WINDOWS\SYSTEM32\WBEM\SCM.MOF
C:\WINDOWS\SYSTEM32\WBEM\FEVPROV.MOF
C:\WINDOWS\SYSTEM32\WBEM\FEVPROV.MFL
C:\WINDOWS\SYSTEM32\WBEM\WMITIMEP.MOF
C:\WINDOWS\SYSTEM32\WBEM\WMITIMEP.MFL
C:\WINDOWS\SYSTEM32\WBEM\EVNTRPRV.MOF
C:\WINDOWS\SYSTEM32\WBEM\CMDEVTGPROV.MOF
C:\WINDOWS\SYSTEM32\WBEM\WHQLPROV.MOF

```

MOF files are also included in other Microsoft products such as Microsoft Office 2003, ASP.NET 1.1 and 2.0, and the Microsoft WMI Tools. These files also contain the ['#PRAGMA AUTORECOVER'](#) statement:

```

C:\WINDOWS\MICROSOFT.NET\FRAMEWORK\V1.1.4322\ASPNET.MOF
C:\WINDOWS\MICROSOFT.NET\FRAMEWORK\V2.0.50727\ASPNET.MOF

C:\WINDOWS\SYSTEM32\WBEM\OUTLOOK_01C56F6C0B3112E6.MOF
C:\WINDOWS\SYSTEM32\WBEM\OUTLOOK_01C64B2E4B127A1E.MOF
C:\PROGRAM FILES\COMMON FILES\MICROSOFT SHARED\MSINFO\OINFOP11.MOF

C:\PROGRAM FILES\WMI TOOLS\EVIEWER.MOF

```

9. Can you modify WMI/DCOM security remotely through command-line utilities?

There is no tool included in the operating system that can be used to modify WMI or DCOM security from the command line. However, this can be done under Windows XP and Windows Server 2003 using scripts. Note, however, that the scripting technique is very complex, because it requires reading the DCOM security from registry keys containing a binary blob representing the security descriptor. The WMI Diagnosis Tool 2.0 actually reads security descriptors from DCOM and WMI as a binary blobs and converts them to an ADSI security descriptor with the *ConvertBinSDtoObjSD()* script function. The conversion of a binary blob can be achieved with the *IADsSecurityUtility* ADSI interface documented on [MSDN](#) as follows:

```

' AdSecurityUtility security descriptor components definitoon (ADS_SECURITY_INFO_ENUM)
Const ADS_SECURITY_INFO_OWNER           = &h1
Const ADS_SECURITY_INFO_GROUP           = &h2
Const ADS_SECURITY_INFO_DACL            = &h4
Const ADS_SECURITY_INFO_SACL            = &h8

' AdSecurityUtility security descriptor format definition (ADS_SD_FORMAT_ENUM)
Const ADS_SD_FORMAT_IID                  = &h1
Const ADS_SD_FORMAT_RAW                  = &h2

```

```

Const ADS_SD_FORMAT_HEXSTRING          = &h3

' Make sure you have the registry key binary blob in arrayBinSD

strHexSD = ConvertBinDataToHexString (arrayBinSD)
Set objADSSecurity = CreateObject ("AdsSecurityUtility")
objADSSecurity.SecurityMask = ADS_SECURITY_INFO_OWNER Or _
                             ADS_SECURITY_INFO_GROUP Or _
                             ADS_SECURITY_INFO_DACL Or _
                             ADS_SECURITY_INFO_SACL

Set objSD = objADSSecurity.ConvertSecurityDescriptor (strHexSD, _
                                                    ADS_SD_FORMAT_HEXSTRING, _
                                                    ADS_SD_FORMAT_IID)

' Here you have an ADSI Security Descriptor representation

' Decipher/Change the ADSI Security Descriptor here!

' -----
Function ConvertBinDataToHexString (arrayBinData)

    Dim strbyte, intIndex

    On Error Resume Next

    For intIndex = 0 to UBound (arrayBinData)
        strbyte = Right ("0" & Hex (arrayBinData (intIndex)), 2)
        ConvertBinDataToHexString = ConvertBinDataToHexString & strbyte
    Next

End Function

```

After you have an ADSI Security Descriptor object you can decipher the security descriptor in an ACL or an ACE by using the ADSI security descriptor object representation documented on MSDN.

10. *Is there a list on microsoft.com that shows all known applications using WMI?*

There is no official list of applications that use WMI. However, the words “using WMI” must be clarified. This means consuming and providing WMI information. It is almost impossible to provide a list of all WMI consumers, because any script or application in the world could consume WMI information. However, the number of WMI applications providing WMI information is fairly well known (outside of the operating system). Here is a list of applications that Microsoft knows about. This list is not exhaustive and anyone is welcome to let us know any application providing WMI information that is not on that list by sending an email to WMIDdiag@Microsoft.Com:

Microsoft	
Systems Management Server	http://www.microsoft.com/smsserver/default.mspix
Microsoft Operations Manager	http://www.microsoft.com/mom/default.mspix
Host Integration Server	http://www.microsoft.com/hiserver/default.mspix
Automated Deployment Services	http://www.microsoft.com/windowsserver2003/technologies/management/ads/default.mspix
Exchange Server 2000/2003	http://www.microsoft.com/exchange/default.mspix
Microsoft Office 2000/2003	http://office.microsoft.com/en-us/default.aspx
SQL Server 2000/2005	http://www.microsoft.com/sql/default.mspix
Microsoft Identity Integration Server	http://www.microsoft.com/windowsserversystem/miis2003/default.mspix
Microsoft Office Live Communication Server (LCS)	http://www.microsoft.com/office/livecomm/prodinfo/default.mspix
Audit Collector Service (ACS)	Not released yet.
Windows 2000 -> Vista	2000: 29 Providers, XP: 57 Providers, 2003: 83 Providers, Vista Client: 70, Longhorn Server: +110
3rd Parties	
DeskView	http://www.fujitsu-siemens.co.uk/rl/servicesupport/techsupport/software/Deskview/Deskview53.htm

BMC Software	http://www.bmc.com/corporate/nr1999/microsoft.html
Bit Defender	http://www.bitdefender.com/PRODUCT-20-en--BitDefender-Enterprise-Manager-v2.html
Intel Network drivers	http://support.intel.com/support/network/sb/cs-006103.htm
IBM Tivoli	http://www-03.ibm.com/certify/tests/obj886.shtml
IBM Director	http://www-03.ibm.com/servers/eserver/xseries/systems_management/agent.html#q11
HP OpenView	http://h20276.www2.hp.com/drc/topics/OpenView.jsp
MetaFrame Enterprise Edition	http://www.citrix.com/lang/English/home.asp
eXctender	http://www.exccsoftware.com/WMI/nonWindows/Architecture.htm
DB2 Integration	http://publib.boulder.ibm.com/infocenter/db2help/index.jsp?topic=/com.ibm.db2.udb.doc/admin/c0007444.htm
Bindview	http://www.bindview.com/Partners/Why-Partner/
HP/Compaq Insight Manager Agents	http://h18000.www1.hp.com/products/servers/management/agents-ga.html#11
Vintela	http://www.vintela.com/products/vmx/product_details.php
Tucows Asset Logger	http://www.tucows.com/get/290318_114163
IPSwitch WhatsUp Professional	http://www.ipswitch.com/products/whatsup/professional/features.asp

WMI Diagnosis Tool 2.0 Updates

WMI Diagnosis Tool new command line parameters

The WMI Diagnosis Tool 2.0 command line parameters are completely compatible with the previous version of the WMI Diagnosis Tool. However, new command line switches are available as described below. New command line switches are in *italic* and described in the specific sections included in this document.

Note: If you use the *SMTPServer* command line parameter, the *SMTPFrom* command line parameter is now mandatory.

```
D:\WMIDdiag>WMIDdiag.vbs ?
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.

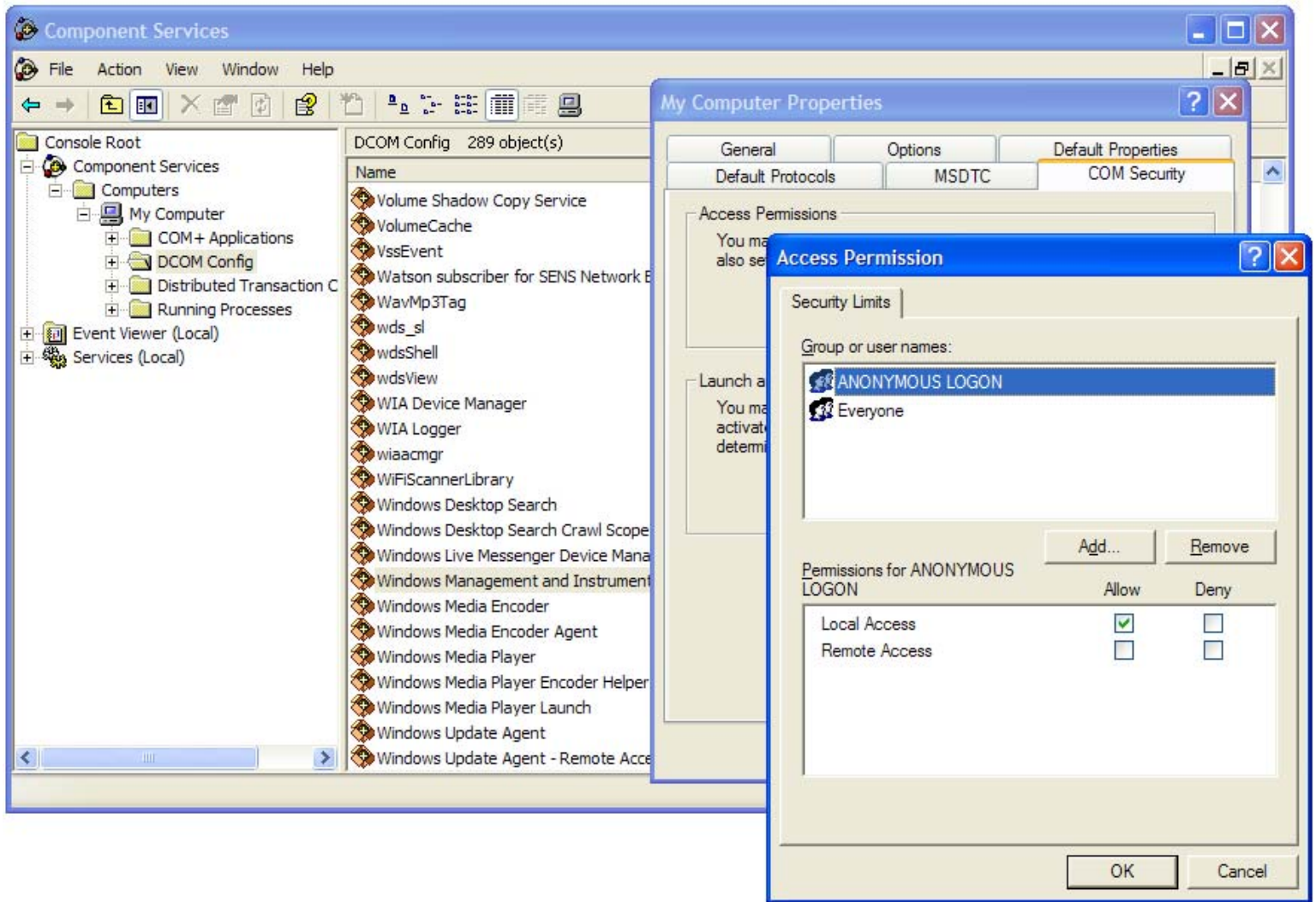
(0) ** Retrieving Run-time environment information.
(3)
(3) (WMIDdiag 2.0) Help
(3)
(3) WMIDdiag [NoEcho] [NoWinZIP] [Silent] [RunOnce] [Force] [SMS] [OldestLogHistory=1-365] [Depth=1|2|3|4|...]
(3) [OldestEVENTLogHistory] [RequestAllInstances[=Static|Dynamic|StaticAndDynamic]]
(3) [WriteInRepository[=Root\XYZ]] [CheckConsistency] [ShowLoadedProviders]
(3) [BaseNameSpace=Root\XYZ] [LoggingLevel=n] [ShowMOFErrors] [CorrelateClassAndProvider]
(3) [LogFilePath=<FolderPath>] [LogNTEvents] [LogNTEventErrors] [LogWMIState] [ErrorPopup]
(3) [SmtpServer=x.y.z] [SmtpPort=nn] [SmtpAuthenticate=<AuthLevel>] [SmtpSSL]
(3) [SmtpUsername=<UserName>] [SmtpPassword=<Password>]
(3) [SmtpFrom=<source@domain.com>] [SmtpTo=<target@domain.com>] [SmtpWMIInvalidState] [SmtpTest]
(3)
(3) Where:
(3)
(3) NoEcho Disables local console echo of WMIDdiag activities (default=False)
(3) NoWinZIP Disables automatic compression of the WMI generated files in a ZIP file (default=False)
(3) Silent Prevents any Window prompts to appear except help (default=False)
(3) RunOnce Prevents WMIDdiag to run more than once a day (default=False)
(3) SMS Turns off the ERRORLEVEL return code and active NoEcho and Silent (default=False)
(3) OldestLogHistory Deletes WMIDdiag files older than the specified # of days (default=False)
(3) OldestEventLogHistory Retrieves Event Log events older than the specified # of days (default=20)
(3) Force Forces WMIDdiag to run even if you are not a FULL administrator (default=False)
(3) Depth Defines the class hierarchy depth to examine (default=1)
(3) RequestAllInstances Requests WMIDdiag to retrieve all instances from STATIC and/or DYNAMIC classes (default=False)
(3) WriteInRepository Requests WMIDdiag to write non-destructive information temporarily in the WMI repository for testing purposes (default=False)
(3) CheckConsistency Executes Windows XP SP2 RUNDLL32 command to validate the WMI repository consistency (default=False)
```

(3)	BaseNamespace	Defines the WMI namespace to start the test from (default=Root)
(3)	LoggingLevel	Defines the verbose of the logging level (default=0)
(3)	CorrelateClassAndProvider	Creates a CSV listing the data correlation between classes, WMI providers and MOF files (default=False)
(3)	ShowMOFErrors	Displays detected MOF errors (default=False)
(3)	ShowLoadedProviders	Displays information about the WMI providers currently loaded by the system (default=False)
(3)		
(3)	LogFilePath	Defines the path where the WMIDdiag files must be created (default=D:\TEMP\USER\)
(3)	LogNTEvents	Enables logging of WMIDdiag messages in the application event log (default=False)
(3)	LogNTEventErrors	Enables logging of WMIDdiag error messages in the application event log (default=False)
(3)	LogWMIState	Creates an event log entry in the NT Event Log (Success, Warning or Error) to reflect the current WMI status (default=False)
(3)	ErrorPopup	Enables the display of all error messages as popups (default=False)
(3)		
(3)	SmtServer	Defines the SMTP relay server address (no default)
(3)		WARNING: By specifying this address YOU CONSENT to send information externally!
(3)	SmtPort	Defines the SMTP port number (default=25)
(3)	SmtAuthenticate	Defines the authentication mechanism. It could be [Anonymous] [Basic] [Ntlm] (default=Ntlm)
(3)	SmtSSL	Enables SSL to communicate with the SMTP relay (default=False)
(3)	SmtUsername	Specifies the username to connect to the SMTP relay (no default)
(3)	SmtPassword	Specifies the password to connect to the SMTP relay (no default)
(3)	SmtFrom	Specifies the SMTP source address (mandatory if SMTP server is mentioned)
(3)	SmtTo	Specifies the SMTP target address (default=WMIDdiag@microsoft.com)
(3)	SmtWMIInvalidState	Sends an email only when the WMI state is reported as WARNING or ERROR (default=False)
(3)	SmtTest	Enables the test of the SMTP parameters. No WMIDdiag tests are executed with this parameter (default=False)

DCOM and WMI namespace security analysis

The WMI Diagnosis Tool automatically analyzes the security settings in the DCOM space and in the various WMI namespaces to compare the actual security settings with the expected settings. For DCOM, the WMI Diagnosis Tool screens the security settings at the computer level for the Default Access Permission (figure below), the Machine Access Restriction, the Default Launch Permission and the Machine Launch Restriction. The WMI Diagnosis Tool also verifies the Access and Launch/Activate permissions security settings on WMI DCOM components called “Windows Management and Instrumentation”, “Microsoft WMI Provider Subsystem Hosts”, “Microsoft WBEM UnSecured Apartment” and “Microsoft WBEM Active Scripting Event Consumer Provider”.

The security information is deciphered when using Windows XP, Windows 2003 and above. This functionality is not supported under Windows 2000 (Client and Server) due to the missing functionality in the operating system.



The WMI Diagnosis Tool verifies the security set on every namespace present in the machine. However, the WMI Diagnosis Tool makes a distinction between namespaces that are:

1. Created by the operating system installation and use default WMI security settings (i.e. ROOT/CIMV2).
2. Created by the operating system installation and use non-default WMI security settings (i.e. ROOT/RSOP).
3. Created by the usage of the operating system and use security settings containing non-well-known SID (i.e. ROOT\RSOP\User\S_1_5_21_2127521184_1604012920_1887927527_2058479 or ROOT\RSOP\User\S_*).
4. Created by the installation of an external application in the operating system which uses specific WMI security settings (i.e. ROOT/CCM/EVENTS or ROOT/CCM/*).

During the WMI namespace security analysis, WMI Diagnosis Tool will take into account these situations and reports any differences with the expected defaults for each category. However, WMI overall health status as reported by the tool is only impacted by situations 1, 2 and 4 listed above. Therefore:

- a) Namespaces for situation 1 are verified, reported and could impact the WMI overall status based of the difference found.
- b) Namespaces for situation 2 are verified, reported and could impact the WMI overall status based of the difference found.
- c) Namespaces for situation 3 are skipped during the security verification process because there is no way for WMI Diagnosis Tool to determine which non-well-known SID are expected. Note however that very few namespaces are in this category.
- d) Namespaces for situation 4 (known by WMI Diagnosis Tool) are verified and reported but do not impact the WMI overall status in case of differences.

- e) Namespaces out of the four categories above (and thus not known by the WMI Diagnosis Tool) are verified and reported based on the WMI expected default. These do not affect the WMI overall status in case of differences.

When security is validated, the WMI Diagnosis Tool reports warnings and errors based on four situations:

1. The removal or modification of a trustee part of the default setup is an error.
2. The removal or modification of a right in an access mask of a default trustee is considered as an error.
3. The addition of a new trustee with an access denied is returned as a warning
4. The addition of a new ACE for a default trustee with an access denied is returned as a warning.

Any other modifications, such as granting a new trustee or adding new granted rights to a default trustee are not considered as a warning or an error as they cannot impact the overall functionality. You can refer to the section of this paper titled “WMI Diagnosis Tool return codes classification” for more information. In case of differences from an expected default, WMI Diagnosis Tool shows what has been modified and what it is expected to have as illustrated below.

The WMI Diagnosis Tool always verifies the security. No switch is required to get security verified.

```
.5450 15:19:00 (0) ** -----
.5451 15:19:00 (0) ** DCOM security for 'Windows Management Instrumentation' (LaunchPermission):..... MODIFIED.
.5452 15:19:00 (1) !! ERROR: Actual trustee 'EVERYONE' DOES NOT match corresponding expected trustee rights
.5453 15:19:00 (0) **   - ACTUAL ACE:
.5454 15:19:00 (0) **       ACEType:   &h0
.5455 15:19:00 (0) **       ACCESS_ALLOWED_ACE_TYPE
.5456 15:19:00 (0) **       ACEFlags:  &h0
.5457 15:19:00 (0) **       ACEMask:   &hB
.5458 15:19:00 (0) **       DCOM_RIGHT_EXECUTE
.5459 15:19:00 (0) **       DCOM_RIGHT_LAUNCH_LOCAL
.5460 15:19:00 (0) **       DCOM_RIGHT_ACTIVATE_LOCAL
.5461 15:19:00 (0) **   - EXPECTED ACE:
.5462 15:19:00 (0) **       ACEType:   &h0
.5463 15:19:00 (0) **       ACCESS_ALLOWED_ACE_TYPE
.5464 15:19:00 (0) **       ACEFlags:  &h0
.5465 15:19:00 (0) **       ACEMask:   &h1F
.5466 15:19:00 (0) **       DCOM_RIGHT_EXECUTE
.5467 15:19:00 (0) **       DCOM_RIGHT_LAUNCH_LOCAL
.5468 15:19:00 (0) **       DCOM_RIGHT_LAUNCH_REMOTE
.5469 15:19:00 (0) **       DCOM_RIGHT_ACTIVATE_LOCAL
.5470 15:19:00 (0) **       DCOM_RIGHT_ACTIVATE_REMOTE
.5471 15:19:00 (0) **
.5472 15:19:00 (0) ** => The actual ACE has the right(s) '&h14 DCOM_RIGHT_LAUNCH_REMOTE DCOM_RIGHT_ACTIVATE_REMOTE'
.5473 15:19:00 (0) ** removed! This will cause some operations to fail!
.5474 15:19:00 (0) ** It is possible to fix this issue by editing the ACE for and adding the removed right.
.5475 15:19:00 (0) ** For DCOM objects, this can be done with 'DCOMCNFG.EXE'.
.5476 15:19:00 (0) **
.5477 15:19:00 (0) **
.5478 15:19:00 (0) ** DCOM security warning(s) detected: ..... 0.
.5479 15:19:00 (0) ** DCOM security error(s) detected: ..... 1.
.5480 15:19:00 (0) ** WMI security warning(s) detected: ..... 0.
.5481 15:19:00 (0) ** WMI security error(s) detected: ..... 0.
```

WinZIP support

When WinZIP (<http://www.winzip.com>) is installed in a Windows system, the WMI Diagnosis Tool takes advantage of it to compress the LOG, TXT and CSV files created. In that case, if the WMI Diagnosis Tool files are to be sent via email only the ZIP file is sent. Note that the actual WMI Diagnosis Tool files are not deleted after compressing them in a ZIP file. (This can be done by using the *OldestLogHistory* command line parameter). If you use an evaluation version of WinZIP, be aware that the WinZIP tool will prompt the user for the license agreement consent; this, in turn, blocks the WMI Diagnosis Tool until the WinZIP license is confirmed. In that case, you should use the *NoWinZIP* command line switch to prevent the WMI Diagnosis Tool from using WinZIP (or manually confirm the license agreement or buy a WinZIP license!). WinZIP usage is only recommended if you have a licensed version installed.

Windows Vista support

This version of WMI Diagnosis Tool 2.0 has been enhanced to support Windows Vista. It also works under Longhorn Server. Note that, under Longhorn Server, it is possible that the WMI Diagnosis Tool 2.0 will return false positives because this operating system is still under development.

User Account Control (UAC) sensing

WMI Diagnosis Tool supports Windows Vista. In this context, it also verifies if UAC and remote token filtering (LocalAccountTokenFilterPolicy) are active. In such a case, WMI Diagnosis Tool warns the user about the impact of UAC and provides advice on how to work with this security feature. Note that WMI Diagnosis Tool requires to be executed from an elevated security context.

```
73271 19:20:37 (0) ** -----
73272 19:20:37 (0) ** INFO: User Account Control (UAC): ..... ENABLED.
73273 19:20:37 (0) ** => WMI tasks requiring Administrative privileges MUST run in an elevated context.
73274 19:20:37 (0) **     i.e. You can start your scripts or WMIC commands from an elevated command
73275 19:20:37 (0) **           prompt by right clicking on the 'Command Prompt' icon in the Start Menu and
73276 19:20:37 (0) **           selecting 'Run as Administrator'.
73277 19:20:37 (0) **     i.e. You can also execute the WMI scripts or WMIC commands as a task
73278 19:20:37 (0) **           in the Task Scheduler within the right security context.
73279 19:20:37 (0) **
73280 19:20:37 (0) ** INFO: Local Account Filtering: ..... ENABLED.
73281 19:20:37 (0) ** => WMI tasks remotely accessing WMI information on this computer and requiring Administrative
73282 19:20:37 (0) **     privileges MUST use a DOMAIN account part of the Local Administrators group of this computer
73283 19:20:37 (0) **     to ensure that administrative privileges are granted.
73284 19:20:37 (0) **     If a Local User account is used for remote accesses, it will be reduced to a plain user
73285 19:20:37 (0) **     (filtered token), even if it is part of the Local Administrators group.
```

Vista Firewall

The WMI Diagnosis Tool analyzes the Vista Firewall settings and returns information about the configuration state of the firewall that may impact the WMI connectivity. The reported information is as follows when analyzing a default Windows Vista setup:

```
37885 16:48:19 (0) ** -----
37886 16:48:19 (0) ** INFO: Windows Firewall status: ..... ENABLED.
37887 16:48:19 (0) ** Windows Firewall Profile: ..... DOMAIN.
37888 16:48:19 (0) ** Inbound connections that do not match a rule BLOCKED: ..... ENABLED.
37889 16:48:19 (0) ** => This will prevent any WMI remote connectivity to this computer except
37890 16:48:19 (0) **     if the following three inbound rules are ENABLED and non-BLOCKING:
37891 16:48:19 (0) **     - 'Windows Management Instrumentation (DCOM-In)'
37892 16:48:19 (0) **     - 'Windows Management Instrumentation (WMI-In)'
37893 16:48:19 (0) **     - 'Windows Management Instrumentation (ASync-In)'
37894 16:48:19 (0) **     Verify the reported status for each of these three inbound rules below.
37895 16:48:19 (0) **
37896 16:48:19 (0) ** Windows Firewall 'Windows Management Instrumentation (WMI)' group rule: ..... DISABLED.
37897 16:48:19 (0) ** => This will prevent any WMI remote connectivity to/from this machine.
37898 16:48:19 (0) **     - You can adjust the configuration by executing the following command:
37899 16:48:19 (0) **     i.e. 'NETSH.EXE ADVFIREWALL FIREWALL
37900 16:48:19 (0) **           SET RULE GROUP="Windows Management Instrumentation (WMI)" NEW ENABLE=YES'
37901 16:48:19 (0) ** Note: With this command all inbound and outbound WMI rules are activated at once!
37902 16:48:19 (0) **     You can also enable each individual rule instead of activating the group rule.
37903 16:48:19 (0) ** Windows Firewall 'Windows Management Instrumentation (ASync-In)' rule: ..... DISABLED.
37904 16:48:19 (0) ** => This will prevent any WMI asynchronous inbound connectivity to this machine.
37905 16:48:19 (0) **     - You can adjust the configuration of this rule by executing the following command:
37906 16:48:19 (0) **     i.e. 'NETSH.EXE ADVFIREWALL FIREWALL
37907 16:48:19 (0) **           SET RULE NAME="Windows Management Instrumentation (ASync-In)" NEW ENABLE=YES'
37908 16:48:19 (0) **
37909 16:48:19 (0) ** Windows Firewall 'Windows Management Instrumentation (WMI-In)' rule: ..... DISABLED.
37910 16:48:19 (0) ** => This will prevent any WMI inbound connectivity to this machine.
37911 16:48:19 (0) ** Note: The rule 'Windows Management Instrumentation (DCOM-In)' rule must be ENABLED to
37912 16:48:19 (0) **     allow incoming WMI connectivity.
37913 16:48:19 (0) **     - You can adjust the configuration of this rule by executing the following command:
37914 16:48:19 (0) **     i.e. 'NETSH.EXE ADVFIREWALL FIREWALL
37915 16:48:19 (0) **           SET RULE NAME="Windows Management Instrumentation (WMI-In)" NEW ENABLE=YES'
37916 16:48:19 (0) **
37917 16:48:19 (0) ** Windows Firewall 'Windows Management Instrumentation (DCOM-In)' rule: ..... DISABLED.
37918 16:48:19 (0) ** => This will prevent any DCOM WMI inbound connectivity to this machine.
37919 16:48:19 (0) **     - You can adjust the configuration of this rule by executing the following command:
37920 16:48:19 (0) **     i.e. 'NETSH.EXE ADVFIREWALL FIREWALL
```

```
SET RULE NAME="Windows Management Instrumentation (DCOM-In)" NEW ENABLE=YES'
```

New WMI firewall rules have been added in Vista. Please refer to the firewall documentation on the Microsoft [website](#) for more information.

Event Log query to locate DCOM, WMI and WMIADAPTER events for the last 20 days

The WMI Diagnosis Tool always queries the event log, not only to validate WMI functionality, but also to retrieve some relevant information regarding DCOM, WMI and WMIAdapter (ADAP). By default, WMI Diagnosis Tool queries for event log entries for the last 20 days. This default period can be overwritten with the *OldestEventLogHistory* command line parameter. WMI Diagnosis Tool distinguishes events created before the execution of WMI Diagnosis Tool from those created during the execution of WMI Diagnosis Tool:

```
44682 17:07:40 (0) ** - NT Event Log information
44683 17:07:40 (3)
44684 17:07:40 (3)   Querying event log for events older than 20 day(s) since Wednesday, August 16, 2006 at 16:54.
44685 17:07:40 (3)
44686 17:07:45 (3)     001 DCOM event(s):
44687 17:07:45 (4)
44688 17:07:45 (3)       #038874: DCOM (10010) - Error - 26 August 2006 11:11:02 (GMT+7)
44689 17:07:45 (3)         The server {ED081F25-6A77-4C89-B689-C6E15C582EC1} did not register with
44690 17:07:45 (3)         DCOM within the required timeout.
44691 17:07:45 (3)
44692 17:07:52 (3)     ZERO WINMGMT event.
44693 17:07:52 (4)
44694 17:07:58 (3)     ZERO WMIADAPTER event.
44695 17:07:58 (4)
44696 17:07:58 (3)   Querying event log for events since Tuesday, September 05, 2006 at 16:54.
44697 17:07:58 (3)
44698 17:07:59 (3)     ZERO DCOM event.
44699 17:07:59 (4)
44700 17:07:59 (3)     ZERO WINMGMT event.
44701 17:07:59 (4)
44702 17:07:59 (3)     ZERO WMIADAPTER event.
44703 17:07:59 (4)
```

Run-time environment verification

Analyze EventLog to detect suspicious improper system shutdown

When querying the event log, the WMI Diagnosis Tool also locates the event log service startup/shutdown events to determine if the Windows system was not properly shutdown over time.

```
44706 17:12:12 (3)     153 Startup/Shutdown event(s) found!
44707 17:12:12 (2) !! WARNING: Incorrect system Shutdown on 20 November 2004 18:54:11 (GMT+8).
44708 17:12:12 (2) !! WARNING: Incorrect system Shutdown on 10 December 2004 12:27:40 (GMT+8).
44709 17:12:12 (2) !! WARNING: Incorrect system Shutdown on 08 March 2005 14:42:45 (GMT+8).
44710 17:12:12 (2) !! WARNING: Incorrect system Shutdown on 26 March 2005 14:08:34 (GMT+8).
44711 17:12:12 (2) !! WARNING: Incorrect system Shutdown on 06 May 2005 16:34:49 (GMT+7).
44712 17:12:12 (2) !! WARNING: Incorrect system Shutdown on 28 November 2005 18:59:51 (GMT+8).
44713 17:12:12 (2) !! WARNING: Incorrect system Shutdown on 24 December 2005 10:25:42 (GMT+8).
44714 17:12:12 (2) !! WARNING: Incorrect system Shutdown on 11 January 2006 14:20:16 (GMT+8).
44715 17:12:12 (2) !! WARNING: Incorrect system Shutdown on 27 June 2006 20:00:12 (GMT+7).
44716 17:12:12 (4)
44717 17:12:12 (2) !! WARNING: 9 incorrect system shutdown(s) found!
```

This information is purely informational and is part of the environment verification as reported in the WMI Diagnosis Tool report:

```
44734 17:12:12 (0) ** INFO: Environment: ..... 1 ITEM(S)!
44735 17:12:12 (0) ** INFO: => 9 incorrect shutdown(s) detected on:
44736 17:12:12 (0) **   - Shutdown on 20 November 2004 18:54:11 (GMT+8).
44737 17:12:12 (0) **   - Shutdown on 10 December 2004 12:27:40 (GMT+8).
44738 17:12:12 (0) **   - Shutdown on 08 March 2005 14:42:45 (GMT+8).
44739 17:12:12 (0) **   - Shutdown on 26 March 2005 14:08:34 (GMT+8).
44740 17:12:12 (0) **   - Shutdown on 06 May 2005 16:34:49 (GMT+7).
44741 17:12:12 (0) **   - Shutdown on 28 November 2005 18:59:51 (GMT+8).
44742 17:12:12 (0) **   - Shutdown on 24 December 2005 10:25:42 (GMT+8).
44743 17:12:12 (0) **   - Shutdown on 11 January 2006 14:20:16 (GMT+8).
```

If the event log has been cleared, this information cannot be reported.

Enhanced registry checking

The WMI Diagnosis Tool has always verified the registry. With version 2.0, the WMI Diagnosis Tool can raise an informational, warning or error message when an incorrect registry key is detected. The level of criticality depends on the registry key. In the same way, WMI Diagnosis Tool can raise a warning when a registry key is present when it should not be (typical with DCOM when the configuration is modified).

Cscript timeout

Some customers configure the WSH subsystem with a script execution timeout. This forces a WSH script to terminate its execution within the specified time period. As the WMI Diagnosis Tool may take between 6 minutes and 2 hours to execute, depending on the command line parameters specified and the system type, this timeout might not give the WMI Diagnosis Tool time to complete execution.

```
(0) ** Checking Auto-Recovery MOF files presence.
(0) ** Checking MOF files in WBEM folder.
(0) ** Checking '#PRAGMA AUTORECOVER' statement in MOF files.
Script execution time was exceeded on script "C:\WMIDdiag\WMIDdiag.vbs",
Script execution was terminated.
```

Therefore, the WMI Diagnosis Tool verifies if WSH is configured with a timeout and returns an information message for this potential problem (in both the LOG and in the report at the end of the execution). Note that this information is captured immediately after the startup of the tool and is available in the LOG file just in case if WMI Diagnosis Tool does not get the chance to complete its execution due to the WSH timeout):

```
.5486 12:15:29 (0) ** INFO: Environment: ..... 1 ITEM(S)!
.5487 12:15:29 (0) ** INFO: => WSH is configured with a maximum time a script is permitted to run:
.5488 12:15:29 (0) ** - WSH USER settings has a timeout set to 100 seconds. With REGEDIT,
.5489 12:15:29 (0) ** set the 'Timeout' registry key at 'HKCU\Software\Microsoft\Windows Script
.5490 12:15:29 (0) ** Host\Settings' to 0.
.5491 12:15:29 (0) ** => You can also overwrite these settings with the command:
.5492 12:15:29 (0) ** i.e. 'Cscript.Exe //T:5000 WMIDdiag.vbs'
.5493 12:15:29 (0) ** i.e. 'Cscript.Exe //T:0 //S'
```

WBEM presence in path and maximum PATH length

Some applications modify the PATH variable, which moves the %SystemRoot%\System32\WBEM path to an undesired position in the PATH environment string (for example, WBEM path is removed, or it is located beyond the maximum PATH length).

```
.5484 12:24:58 (1) !! ERROR: Environment: ..... 1 ITEM(S)!
.5485 12:24:58 (1) !! ERROR: => The following path(s) is/are missing from the PATH environment variable:
.5486 12:24:58 (0) ** - C:\WINDOWS\SYSTEM32\WBEM
.5487 12:24:58 (0) ** Failing to have the listed path(s) in the PATH environment variable
.5488 12:24:58 (0) ** could prevent the system to work properly.
```

The WMI Diagnosis Tool report is available in a separate file

Many people were confused by the size of the WMI Diagnosis Tool LOG and the actual information it contains. Because of this, people did not realize that the bottom of the LOG contains a summary report guiding the support personnel about what to do regarding issues found. To prevent this, and in addition of the report already in the LOG, WMI Diagnosis Tool now creates a TXT file only containing the report. This report is automatically loaded in Notepad if the global WMI state is at the warning or error level (unless the *Silent* command line parameter is specified).

SMS WMI information

In addition to validating core WMI classes by their presence and by performing some typical enumerations and WQL queries, the WMI Diagnosis Tool also executes validation against the SMS classes in SMS namespaces such as Root\CCM. The WMI Diagnosis Tool supports SMS Advanced Client from SMS 2003 RTM, SMS 2003 SP1 and SMS

2003 SP2. However, the WMI Diagnosis Tool does not validate the SMS agent setup; it only validates the SMS Agent ability to work with the WMI infrastructure.

```
(0) ** Verifying WMI providers loaded BEFORE WMIDiag execution.
(0) ** Verifying WMI namespace 'ROOT/DEFAULT' (L=1).
(0) ** Verifying WMI ADAP status.
(0) ** Verifying WMI features.
(0) ** Verifying SMS Agent v2.50.4160 WMI features.
(0) ** Collecting system information.
(0) ** - Disk information
(0) ** - Network information
(0) ** - DMA resource usage
(0) ** - Memory information
(0) ** - Processor information
(0) ** - Operating System information
(0) ** - Services information
(0) ** - WMI Binary files information
(0) ** - NT Event Log information
(0) ** Verifying WMI providers loaded AFTER WMIDiag execution.
(0) ** Verifying WMI Repository files presence.
```

Permanent subscriptions and timers

The WMI Diagnosis Tool analyzes the Permanent Consumer registrations for WMI Events. By screening the repository content, the tool identifies any permanent subscription and timer instructions registered in the WMI repository for every screened namespace. Detailed information is provided in the WMI Diagnosis Tool LOG and in the WMI Diagnosis Tool report.

```
21543 22:46:45 (0) ** Verifying WMI namespace 'ROOT/CIMV2' (L=2).
21544 22:46:45 (3) Retrieving WMI system class(es) static information.
21545 22:46:45 (3) 45/45 system class(es) found.
21546 22:46:45 (3) Verifying Permanent subscription(s) for 'ROOT/CIMV2'.
21547 22:46:45 (3) 0 permanent subscription(s) in 'ROOT/CIMV2' namespace.
21548 22:46:45 (3) 1 Timer instruction(s) in 'ROOT/CIMV2' namespace.
21549 22:46:45 (4)
21550 22:46:45 (4) #1 'MyIntervalTimerEvent' Timer instruction.
21551 22:46:45 (4) - __IntervalTimerInstruction #1 -----
21552 22:46:45 (4) IntervalBetweenEvents: ... 5000
21553 22:46:45 (4) SkipIfPassed: ..... FALSE
21554 22:46:45 (4) *TimerId: ..... MyIntervalTimerEvent
.
.
30466 22:49:16 (3) Verifying Permanent subscription(s) for 'ROOT/SUBSCRIPTION'.
30467 22:49:16 (3) 3 permanent subscription(s) in 'ROOT/SUBSCRIPTION' namespace.
30468 22:49:16 (4)
30469 22:49:16 (4) #1 'LogFileEventConsumer.Name="LOGFileForSvc"' permanent subscription.
30470 22:49:16 (4) - LogFileEventConsumer #1 -----
30471 22:49:16 (4) CreatorSID: ..... 1,5,0,0,0 ...
30472 22:49:16 (4) Filename: ..... C:\WMIServiceWatcher.LOG
30473 22:49:16 (4) IsUnicode: ..... FALSE
30474 22:49:16 (4) MaximumFileSize: ..... 1024
30475 22:49:16 (4) *Name: ..... LOGFileForSvc
30476 22:49:16 (4) Text: ..... Service %TargetInstance.DisplayName% is %TargetInstance.State%.
30477 22:49:16 (4) - __FilterToConsumerBinding #1 -----
30478 22:49:16 (4) *Consumer: .....
                \\PC-ALAINL-02\ROOT\subscription:LogFileEventConsumer.Name="LOGFileForSvc"
30479 22:49:16 (4) CreatorSID: ..... 1,5,0,0,0,0 ...
30480 22:49:16 (4) DeliverSynchronously: .. FALSE
30481 22:49:16 (4) *Filter: .....
                \\PC-ALAINL-02\ROOT\subscription:__EventFilter.Name="FilterForWIN32_Services"
30482 22:49:16 (4) MaintainSecurityContext: FALSE
30483 22:49:16 (4) SlowDownProviders: ..... FALSE
30484 22:49:16 (4) - __EventFilter #1 -----
30485 22:49:16 (4) CreatorSID: ..... 1,5,0,0,0,0,5,21,0,0,0 ...
30486 22:49:16 (4) *Name: ..... FilterForWIN32_Services
30487 22:49:16 (4) Query: ..... SELECT * FROM __InstanceModificationEvent WITHIN 10
                Where TargetInstance ISA 'Win32_Service'
30488 22:49:16 (4) QueryLanguage: ..... WQL
30489 22:49:16 (4)
30490 22:49:16 (4) #2 'NTEventLogEventConsumer.Name="SCM Event Log Consumer"' permanent subscription.
30491 22:49:16 (4) - NTEventLogEventConsumer #2 -----
30492 22:49:16 (4) Category: ..... 0
```

```

30493 22:49:16 (4) CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30494 22:49:16 (4) EventID: ..... 0
30495 22:49:16 (4) EventType: ..... 1
30496 22:49:16 (4) InsertionStringTemplates:
30497 22:49:16 (4) *Name: ..... SCM Event Log Consumer
30498 22:49:16 (4) NameOfUserSIDProperty: ... sid
30499 22:49:16 (4) NumberOfInsertionStrings: 0
30500 22:49:16 (4) SourceName: ..... Service Control Manager
30501 22:49:16 (4) - __FilterToConsumerBinding #1 -----
30502 22:49:16 (4) *Consumer: ..... NTEventLogEventConsumer.Name="SCM Event Log Consumer"
30503 22:49:16 (4) CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30504 22:49:16 (4) DeliverSynchronously: .. FALSE
30505 22:49:16 (4) *Filter: ..... __EventFilter.Name="SCM Event Log Filter"
30506 22:49:16 (4) MaintainSecurityContext: FALSE
30507 22:49:16 (4) SlowDownProviders: ..... FALSE
30508 22:49:16 (4) - __EventFilter #1 -----
30509 22:49:16 (4) CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30510 22:49:16 (4) EventNamespace: ..... root\cimv2
30511 22:49:16 (4) *Name: ..... SCM Event Log Filter
30512 22:49:16 (4) Query: ..... Select * from MSFT_SCMEEventLogEvent
30513 22:49:16 (4) QueryLanguage: ..... WQL
30514 22:49:16 (4)
30515 22:49:16 (4) #3 'MSFT_UCScenarioControl.Name="Microsoft WMI Updating Consumer Scenario Control"'
    permanent subscription.
30516 22:49:16 (4) - MSFT_UCScenarioControl #3 -----
30517 22:49:16 (4) CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30518 22:49:16 (4) *Name: ..... Microsoft WMI Updating Consumer Scenario Control
30519 22:49:16 (4) - __FilterToConsumerBinding #1 -----
30520 22:49:16 (4) *Consumer: .....
30521 22:49:16 (4) \\.\root\subscription:MSFT_UCScenarioControl.Name="Microsoft WMI Updating Consumer Scenario Control"
30522 22:49:16 (4) CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30523 22:49:16 (4) DeliverSynchronously: ... TRUE
30524 22:49:16 (4) *Filter: .....
30525 22:49:16 (4) \\.\root\subscription:__EventFilter.Name="Microsoft WMI Updating Consumer Scenario Control"
30526 22:49:16 (4) MaintainSecurityContext: FALSE
30527 22:49:16 (4) SlowDownProviders: ..... FALSE
30528 22:49:16 (4) - __EventFilter #1 -----
30529 22:49:16 (4) CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30530 22:49:16 (4) *Name: ..... Microsoft WMI Updating Consumer Scenario Control
30531 22:49:16 (4) Query: ..... SELECT * FROM __InstanceOperationEvent WHERE TargetInstance
    ISA 'MSFT_UCScenario'
30532 22:49:16 (4) QueryLanguage: ..... WQL
30533 22:49:16 (3) 0 Timer instruction(s) in 'ROOT/SUBSCRIPTION' namespace.
.
.
.
35647 22:50:25 (0) ** -----
35648 22:50:25 (0) ** Overall DCOM security status: ..... OK.
35649 22:50:25 (0) ** Overall WMI security status: ..... OK.
35650 22:50:25 (0) ** - Started at 'Root' -----
35651 22:50:25 (0) ** INFO: WMI permanent SUBSCRIPTION(S): ..... 7.
35652 22:50:25 (0) ** - ROOT/CCM/POLICY, CCM_PolicyReplicationConsumer.Id="{9099D177-1AD6-46e6-BBC0-70F460786953}".
35653 22:50:25 (0) ** 'SELECT * FROM __ClassOperationEvent WHERE TargetClass ISA "CCM_Policy_Config"'
35654 22:50:25 (0) ** - ROOT/CCM/POLICY, CCM_PolicyReplicationConsumer.Id="{9099D177-1AD6-46e6-BBC0-70F460786953}".
35655 22:50:25 (0) ** 'SELECT * FROM __NamespaceCreationEvent'
35656 22:50:25 (0) ** - ROOT/CCM/POLICY, CCM_PolicyReplicationConsumer.Id="{9099D177-1AD6-46e6-BBC0-70F460786953}".
35657 22:50:25 (0) ** 'SELECT * FROM __ClassOperationEvent WHERE TargetClass ISA "CCM_Policy"'
35658 22:50:25 (0) ** - ROOT/CCM/POLICY, CCM_PolicyReplicationConsumer.Id="{9099D177-1AD6-46e6-BBC0-70F460786953}".
35659 22:50:25 (0) ** 'SELECT * FROM __ClassOperationEvent WHERE TargetClass ISA "CCM_Policy_EmbeddedObject"'
35660 22:50:25 (0) ** - ROOT/SUBSCRIPTION, LogFileEventConsumer.Name="LOGFileForSvc".
35661 22:50:25 (0) ** 'SELECT * FROM __InstanceModificationEvent WITHIN 10 Where TargetInstance ISA 'Win32_Service''
35662 22:50:25 (0) ** - ROOT/SUBSCRIPTION, NTEventLogEventConsumer.Name="SCM Event Log Consumer".
35663 22:50:25 (0) ** 'select * from MSFT_SCMEEventLogEvent'
35664 22:50:25 (0) ** - ROOT/SUBSCRIPTION, MSFT_UCScenarioControl.Name="Microsoft WMI Updating Consumer Scenario".
35665 22:50:25 (0) ** 'SELECT * FROM __InstanceOperationEvent WHERE TargetInstance ISA 'MSFT_UCScenario''
35666 22:50:25 (0) **
35667 22:50:25 (0) ** INFO: WMI TIMER instruction(s):..... 1.
35668 22:50:25 (0) ** - Interval: ROOT/CIMV2, 'MyIntervalTimerEvent', 5000'.
35669 22:50:25 (0) **
35670 22:50:25 (0) ** WMI ADAP status: ..... OK.
35671 22:50:25 (0) ** WMI MONIKER CONNECTIONS: ..... OK.
35672 22:50:25 (0) ** WMI CONNECTIONS: ..... OK.
35673 22:50:25 (0) ** WMI GET operations: ..... OK.
35674 22:50:25 (0) ** WMI MOF representations: ..... OK.

```

MOF file verification

WMIDdiag, with the *ShowMOFErrors* command line parameter, performs a new test in the MOF files to validate the presence of the ['#PRAGMA AUTORECOVER'](#) statement. This statement is required in a MOF file to ensure that, at compilation time (MOFCOMP), the file appears on the list of MOF file to be recompiled in the event that the repository is reconstructed.

```
.5717 14:10:51 (1) !! ERROR: MOF file(s) not containing the '#PRAGMA AUTORECOVER' statement: ..... 24 FILE(S)!
.5718 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\FCONPROV.MFL
.5719 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\FCONPROV.MOF
.5720 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\NCPROV.MFL
.5721 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\NCPROV.MOF
.5722 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\SCRCONS.MFL
.5723 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\SCRCONS.MOF
.5724 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\SMTPCONS.MFL
.5725 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\SMTPCONS.MOF
.5726 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\TMPLPROV.MFL
.5727 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\TMPLPROV.MOF
.5728 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\TRNSPROV.MFL
.5729 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\TRNSPROV.MOF
.5730 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\UPDPROV.MFL
.5731 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\UPDPROV.MOF
.5732 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\WBEMCONS.MFL
.5733 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\WBEMCONS.MOF
.5734 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\SCM.MOF (*)
.5735 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\FEVPROV.MOF (*)
.5736 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\FEVPROV.MFL (*)
.5737 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\WMITIMEP.MOF (*)
.5738 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\WMITIMEP.MFL (*)
.5739 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\EVNTRPRV.MOF (*)
.5740 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\CMDEVTGPROV.MOF (*)
.5741 14:10:51 (0) ** - C:\WINDOWS\SYSTEM32\WBEM\WHQLPROV.MOF (*)
.5742 14:10:51 (0) ** => MOF file(s) marked with a (*) are ALWAYS included AT SETUP in the auto-recovery process
.5743 14:10:51 (0) ** even if they do NOT contain the '#PRAGMA AUTORECOVER' statement.
.5744 14:10:51 (0) ** If the WMI repository is rebuilt, the listed MOF files not included AT SETUP in the
.5745 14:10:51 (0) ** auto-recovery list and missing the '#PRAGMA AUTORECOVERY' statement when they are compiled
.5746 14:10:51 (0) ** the first time will NOT be recompiled during reconstruction.
.5747 14:10:51 (0) ** If you want the MOF file not marked with a (*) to be part of the Auto-Recovery, make sure the
.5748 14:10:51 (0) ** statement '#PRAGMA AUTORECOVER' is included.
.5749 14:10:51 (0) ** Note: It is also possible that the application implemented its own recovery mechanism.
.5750 14:10:51 (0) ** In that case, no action is required.
.5751 14:10:51 (0) ** Note: You must verify with the application vendor if the application has this
.5751 14:10:51 (0) ** capability (i.e. Microsoft SMS)
```

MOF files marked with (DELETE) contain class and instance DELETE statements. Usually, MOF with DELETE statements are used to uninstall components and should not be listed in the AUTORECOVERY LIST. In addition, they should not specify the ['#PRAGMA AUTORECOVER'](#) statement.

Note: It happens that some MOF files do contain the DELETE statements for installation purposes; this enables them to delete information before it gets re-created.

Loaded WMI providers

The WMI Diagnosis Tool, with the *ShowLoadedProviders* command line parameter, returns the list of WMI providers currently loaded in memory. This feature uses WMI and leverages the *MSFT_Providers* class. For each provider currently running, the WMI Diagnosis Tool returns the display name, the associated binary (i.e., DLL file) and the corresponding MOF. The WMI Diagnosis Tool returns this additional information only for the providers shipped with the operating system and SMS 2003 Advanced Client. When the *ShowLoadedProviders* command line parameter is specified, the WMI Diagnosis Tool only returns the loaded provider list and does not execute any of the verification tests. In such a case, the output is as follows:

```
D:\WMIDdiag>WMIDdiag.vbs ShowLoadedProviders
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.

(0) ** Retrieving Run-time environment information.
(0) ** LOG file "D:\TEMP\USER\WMIDIAG-V2.0_XP___.CLI.SP2.32_PC-ALAINL-02_2006.09.23_04.54.23.LOG" created.
(0) ** Initializing WMI System Information.
```

```
(3) 18 WMI providers are currently loaded:
(4) Provider: 'SMSCLIENTMETHODPROVIDER' (PID: 6040, WMIPRVSE.EXE)
(4)   Provider Name: 'SMS Client Method WMI Provider'
(4)   Namespace:    'ROOT\CCM' (i.e. 'SMS_Client')
(4)   Hosting Model: 'LOCALSYSTEMHOST:SMS'
(4)   Hosting Group: 'SMS'
(4)   Provider Binary: 'C:\WINDOWS\SYSTEM32\CCM\SMSCLIENT.DLL'
(4)   MOF Registration: 'No located MOF file (exception)'
(4)
(4) Provider: 'POLICYAGENTINSTANCEPROVIDER' (PID: 176, WMIPRVSE.EXE)
(4)   Provider Name: 'SMS Policy Agent Event Instance WMI Provider'
(4)   Namespace:    'ROOT\CCM\POLICY\MACHINE' (i.e. 'CCM_Policy')
(4)   Hosting Model: 'LOCALSYSTEMHOST:CCM'
(4)   Hosting Group: 'CCM'
(4)   Provider Binary: 'C:\WINDOWS\SYSTEM32\CCM\POLICYAGENTPROVIDER.DLL'
(4)   MOF Registration: 'No located MOF file (exception)'
(4)
(4) Provider: 'CIMWIN32' (PID: 1484, WMIPRVSE.EXE)
(4)   Provider Name: 'CIMv2 Core OS Method Instance WMI Provider'
(4)   Namespace:    'ROOT\CIMV2' (i.e. 'Win32_ShareToDirectory')
(4)   Hosting Model: 'NETWORKSERVICEHOST'
(4)   Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
(4)   Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\CIMWIN32.DLL'
(4)   MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\CIMWIN32.MOF / C:\WINDOWS\SYSTEM32\WBEM\SECRW32.MOF'
(4)
(4) Provider: 'CIMWIN32A' (PID: 1484, WMIPRVSE.EXE)
(4)   Provider Name: 'CIMv2 Core OS Method Instance WMI Provider'
(4)   Namespace:    'ROOT\CIMV2' (i.e. 'Win32_PhysicalMedia')
(4)   Hosting Model: 'NETWORKSERVICEHOST'
(4)   Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
(4)   Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\WMIPCIMA.DLL'
(4)   MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\CIMWIN32.MOF'
(4)
(4) Provider: 'HIPERFCOOKER_V1' (PID: 1484, WMIPRVSE.EXE)
(4)   Provider Name: 'High Performance Cooker Counter Instance WMI Provider'
(4)   Namespace:    'ROOT\CIMV2' (i.e. '*')
(4)   Hosting Model: 'LOCALSYSTEMHOST'
(4)   Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
(4)   Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\WMICOOKR.DLL'
(4)   MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\WMI.MOF'
(4)
(4) Provider: 'MS_NT_EVENTLOG_PROVIDER' (PID: 1484, WMIPRVSE.EXE)
(4)   Provider Name: 'EventLog Instance Method WMI Provider'
(4)   Namespace:    'ROOT\CIMV2' (i.e. 'Win32_NTEventLogFile')
(4)   Hosting Model: 'NETWORKSERVICEHOST'
(4)   Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
(4)   Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\NTEVT.DLL'
(4)   MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\NTEVT.MOF'
(4)
(4) Provider: 'MS_NT_EVENTLOG_PROVIDER' (PID: 1484, WMIPRVSE.EXE)
(4)   Provider Name: 'EventLog Instance Method WMI Provider'
(4)   Namespace:    'ROOT\CIMV2' (i.e. 'Win32_NTEventLogFile')
(4)   Hosting Model: 'NETWORKSERVICEHOST'
(4)   Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
(4)   Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\NTEVT.DLL'
(4)   MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\NTEVT.MOF'
(4)
(4) Provider: 'MSFT_PROVIDERSUBSYSTEM' (PID: 1396, C:\WINDOWS\SYSTEM32\SVCHOST.EXE)
(4)   Provider Name: 'Microsoft Provider Sub-system Instance Method WMI Provider'
(4)   Namespace:    'ROOT\CIMV2' (i.e. 'Msft_WmiProvider_Counters')
(4)   Hosting Model: 'WMICORE'
(4)   Hosting Group: ''
(4)   Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\WMIPRVSD.DLL'
(4)   MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\SYSTEM.MOF'
(4)
(4) Provider: 'NT5_GENERICPERFPROVIDER_V1' (PID: 1484, WMIPRVSE.EXE)
(4)   Provider Name: 'Performance Counter Instance WMI Provider'
(4)   Namespace:    'ROOT\CIMV2' (i.e. 'Win32_PerfRawData_NETFramework_NETCLRRemoting')
(4)   Hosting Model: 'NETWORKSERVICEHOST'
(4)   Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
(4)   Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\WBEMPERF.DLL'
(4)   MOF Registration: 'No located MOF file (exception)'
(4)
(4) Provider: 'REGPROPPROV' (PID: 3596, WMIPRVSE.EXE)
(4)   Provider Name: 'Registry Property WMI Provider'
(4)   Namespace:    'ROOT\CIMV2' (i.e. '*')
(4)   Hosting Model: 'LOCALSYSTEMHOST'
(4)   Hosting Group: 'DEFAULTLOCALSYSTEMHOST'
(4)   Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\STDPROV.DLL'
```

```

(4) MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\REGEVENT.MOF'
(4)
(4) Provider: 'SCM EVENT PROVIDER' (PID: 1396, C:\WINDOWS\SYSTEM32\SVCHOST.EXE)
(4) Provider Name: 'Service Control Manager Event WMI Provider'
(4) Namespace: 'ROOT\CIMV2' (i.e. 'MSFT_SCMEvent')
(4) Hosting Model: 'DECOUPLED:NONCOM'
(4) Hosting Group: ''
(4) Provider Binary: 'No located BIN file'
(4) MOF Registration: 'No located MOF file (exception)'
(4)
(4) Provider: 'WIN32_WIN32_TERMINALSERVICE_PROV' (PID: 1484, WMIPRVSE.EXE)
(4) Provider Name: 'Terminal Service Method Instance WMI Provider'
(4) Namespace: 'ROOT\CIMV2' (i.e. 'Win32_TerminalService')
(4) Hosting Model: 'NETWORKSERVICEHOST'
(4) Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
(4) Provider Binary: 'C:\WINDOWS\SYSTEM32\TSCFGWMI.DLL'
(4) MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\TSCFGWMI.MOF'
(4)
(4) Provider: 'SMS_CIMLD' (PID: 3596, WMIPRVSE.EXE)
(4) Provider Name: 'SMS Logical Disk Instance WMI Provider'
(4) Namespace: 'ROOT\CIMV2\SMS' (i.e. 'SMS_LogicalDisk')
(4) Hosting Model: 'LOCALSYSTEMHOSTORSELFHOST (*)'
(4) Hosting Group: 'DEFAULTLOCALSYSTEMHOST'
(4) Provider Binary: 'C:\WINDOWS\SYSTEM32\CCM\SMSCCMLD.DLL'
(4) MOF Registration: 'No located MOF file (exception)'
(4)
(4) Provider: 'SMS_CIMV2' (PID: 3596, WMIPRVSE.EXE)
(4) Provider Name: 'SMS Processor Instance WMI Provider'
(4) Namespace: 'ROOT\CIMV2\SMS' (i.e. 'SMS_Processor')
(4) Hosting Model: 'LOCALSYSTEMHOSTORSELFHOST (*)'
(4) Hosting Group: 'DEFAULTLOCALSYSTEMHOST'
(4) Provider Binary: 'C:\WINDOWS\SYSTEM32\CCM\SMSPROC.DLL'
(4) MOF Registration: 'No located MOF file (exception)'
(4)
(4) Provider: 'REGPROPPROV' (PID: 3596, WMIPRVSE.EXE)
(4) Provider Name: 'Registry Property WMI Provider'
(4) Namespace: 'ROOT\DEFAULT' (i.e. '*')
(4) Hosting Model: 'LOCALSYSTEMHOST'
(4) Hosting Group: 'DEFAULTLOCALSYSTEMHOST'
(4) Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\STDPROV.DLL'
(4) MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\REGEVENT.MOF'
(4)
(4) Provider: 'SYSTEMRESTOREPROV' (PID: 1484, WMIPRVSE.EXE)
(4) Provider Name: 'System Restore Method Instance WMI Provider'
(4) Namespace: 'ROOT\DEFAULT' (i.e. 'SystemRestore')
(4) Hosting Model: 'NETWORKSERVICEHOST'
(4) Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
(4) Provider Binary: 'C:\WINDOWS\SYSTEM32\SRCLIENT.DLL'
(4) MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\SR.MOF'
(4)
(4) Provider: 'MICROSOFT|DSLDPCLASSPROVIDER|V1.0' (PID: 1484, WMIPRVSE.EXE)
(4) Provider Name: 'Active Directory LDAP Pull Class WMI Provider'
(4) Namespace: 'ROOT\DIRECTORY\LDAP' (i.e. '*')
(4) Hosting Model: 'NETWORKSERVICEHOST'
(4) Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
(4) Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\DSPROV.DLL'
(4) MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\DSPROV.MOF'
(4)
(4) Provider: 'WMIPROV' (PID: 3596, WMIPRVSE.EXE)
(4) Provider Name: 'Windows Driver Model (WDM) Method Push Class Instance WMI Provider'
(4) Namespace: 'ROOT\WMI' (i.e. 'MSNdis_NotifyVcRemoval')
(4) Hosting Model: 'LOCALSYSTEMHOST'
(4) Hosting Group: 'DEFAULTLOCALSYSTEMHOST'
(4) Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\WMIPROV.DLL'
(4) MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\WMI.MOF'
(4)
(0) ** LOG file "D:\TEMP\USER\WMIDIAG-V2.0_XP__.CLI.SP2.32_PC-ALAINL-02_2006.09.23_04.54.23.LOG" closed.

```

However, when the *ShowLoadedProviders* command line parameter is not specified, the WMI Diagnosis Tool executes the usual suite of tests and returns this information before and after the tool has executed all WMI tests. This information is available in the WMI Diagnosis Tool LOG.

MOFCOMP and REGSVR32

The WMI Diagnosis Tool suggests registering a WMI provider and/or recompiling a MOF file only if the related WMI provider and/or related MOF file are part of the Windows operating system. The WMI Diagnosis Tool can inform the user which binary file must be used to register the WMI provider (REGSVR32), or which MOF file must be used to recompile the CIM-relevant information (MOFCOMP). This feature leverages the WMI Diagnosis Tool database embedded in the tool described above. Again, this only applies to operating system components.

```
.9451 17:39:05 (2) !! WARNING: WMI provider DCOM registrations missing for the following provider(s): ... 1 WARNING(S)!
.9452 17:39:05 (0) ** - ROOT/DEFAULT, SystemRestoreProv ({A47401F6-A8A6-40EA-9C29-B8F6026C98B8})
    (i.e. WMI Class 'SystemRestore')
.9453 17:39:05 (0) **   Provider DLL: 'C:\WINDOWS\SYSTEM32\SRCLIENT.DLL'
.9454 17:39:05 (0) ** => This is an issue because there are still some WMI classes referencing this list of providers
.9455 17:39:05 (0) **   while the DCOM registration is wrong or missing. This can be due to:
.9456 17:39:05 (0) **     - a de-installation of the software.
.9457 17:39:05 (0) **     - a deletion of some registry key data.
.9458 17:39:05 (0) **     - a registry corruption.
.9459 17:39:05 (0) ** => You can correct the DCOM configuration by:
.9460 17:39:05 (0) **     - Executing the 'REGSVR32.EXE <Provider.DLL>' command.
.9461 17:39:05 (0) **   Note: You can build a list of classes in relation with their WMI provider and
    MOF file with WMIDiag.
.9462 17:39:05 (0) **     (This list can be built on a similar and working WMI Windows installation)
.9463 17:39:05 (0) **     The following command line must be used:
.9464 17:39:05 (0) **     i.e. 'WMIDiag CorrelateClassAndProvider'
```

WMI Diagnosis Tool known limitations

- The WMI Diagnosis Tool cannot be run against a remote Windows computer. You can run it only on the local computer. However, you can execute it remotely by leveraging external technologies and tools as described in section “WMI Diagnosis Tool usage questions”, questions 4, 5, 6 and 7.
- The WMI Diagnosis Tool is not localized. This means that false positives can be returned on non-US Windows versions.
- The WMI Diagnosis Tool has no knowledge of third-party applications that create new classes and add new WMI providers. The WMI information it returns is based entirely on the information registered in WMI and the Windows system by the third-party application. Therefore, if important information is missing, causing this third-party application to fail, the WMI Diagnosis Tool cannot guarantee any specific solutions regarding what should be done to fix the problems.
- The WMI Diagnosis Tool does not decipher DCOM and WMI security on Windows 2000 platforms. This validation is only available only on Windows XP, Windows Server 2003 and later versions of Windows.

The sequence of operations

When the WMI Diagnosis Tool executes, it performs a series of data gathering and validations around WMI and its dependencies (such as DCOM) without involving any WMI operations. Once the ground work of validating the environment is completed, then the WMI Diagnosis Tool starts to execute some specific WMI operations to validate the infrastructure. During any of these phases, the WMI Diagnosis Tool analyses the collected information and results of operations to drive to the conclusions presented in the WMI Diagnosis Tool report.

Non-WMI operations of the WMI Diagnosis Tool (Step 1)

1. Creation of the WMI Diagnosis Tool files.
2. Verification of the last execution of the WMI Diagnosis Tool.
3. Collecting information about the run-time environment information.
4. Determination of the operating system version and service pack (upgrade, slipstream installation).
5. Initialization of the WMI System Information (data structures).
6. Verification of the computer environment (PATH, WSH timeout, UAC, token filtering)
7. Verification for some specific file presences (i.e. trojan)
8. Verification of WMI System files presence in '%SystemRoot%\SYSTEM32\WBEM'.
9. Verification of WMI Repository files presence (date, time, size) and consistency *before* WMI Diagnosis Tool execution.

10. Verification of additional binaries in WBEM folder.
11. Verification of Auto-Recovery MOF files presence (screening of WBEM Auto-Recovery registry key)
12. Verification of MOF files in WBEM folder.
13. Verification of '[#PRAGMA AUTORECOVER](#)' statement in MOF files.
14. Verification of DCOM configuration (enabled, authentication level, impersonation level)
15. Verification of WMI DCOM component registrations (based on the *arrayWMIRegistryDCOMSetup[]* data structure).
16. Verification of WMI DCOM component security (access, launch and activate).
17. Verification of WMI ProgID registrations (winmgmts: CreateObject)
18. Verification Windows Firewall setup (WMI firewall rules).
19. Verification of the WMI Core registry settings (of the CIMOM hive registry based on the *arrayWMIRegistrySettings[]* data structure).
20. Verification of the SVCHOST and WINMGMT WMI Service registry settings (Startup, Shared/Standalone).
21. Verification of the Windows Service known to be dependent on the WMI service (based on the *arrayWMIServiceKnownDependent[]* data structure).
22. Verification of the RPCSS and WINMGMT service status (running, started)

WMI operations of the WMI Diagnosis Tool (Step 2)

1. Collection of information about the WMI providers loaded *before* WMIDiag execution.
2. Recursive WMI namespace browsing.
3. Verification of the WMI system settings set in the Root namespace.
4. Verification of the WMI ADAP status.
5. Verification of the WMI features (Get, Enumerate, ExecQuery, Put)
6. Verification of the WMI SMS features if the SMS Agent is installed (Get, Enumerate, ExecQuery, Put)
7. Collecting system information about disk, network, IRQ, DMA, memory, processor, operating system, Windows services, WMI binary files, NT event log events for DCOM, WMI and WMIADAPTER.
8. Collection of information about the WMI providers loaded AFTER WMI Diagnosis Tool execution.

Non-WMI operations of the WMI Diagnosis Tool (Step 3)

1. Verification of WMI Repository files presence (date, time, size) and consistency *after* WMI Diagnosis Tool execution.
2. WMI report generation.
3. General WMI status report.
4. Closure of the WMI Diagnosis Tool files.

WMI Diagnosis Tool LOG content

In general, information returned by the WMI Diagnosis Tool TXT report is targeted for advanced IT professionals performing checks of the WMI infrastructure. Most of the time the TXT report is sufficient to understand how to fix WMI issues reported. However, there are situations that require more detailed information about an issue. This information is contained in the WMI Diagnosis Tool LOG created during the WMI Diagnosis Tool execution. Both the TXT report and the WMI Diagnosis Tool LOG are created in the %TEMP% folder by default (this location can be changed with the *LogFilePath* command line parameter).

When started, the WMI Diagnosis Tool traces all the activities executed; this information is then recorded in the LOG file. The information in the LOG file is very advanced and primarily relevant to WMI engineers or to PSS specialists troubleshooting problems. This information can also be collected by large enterprises and advanced customers developing their own solutions to report this information centrally. The following sections briefly display information contained in the LOG.

Collecting run-time environment information

```
....1 16:11:38 (0) ** LOG file "D:\WMIDIAG-V2.0_XP__CLI.SP2.32_PC02_2006.09.28_16.11.38.LOG" created.
....2 16:11:38 (0) ** CSV file "D:\WMIDIAG-V2.0_XP__CLI.SP2.32_PC02_2006.09.28_16.11.38-STATISTICS.CSV" created.
....3 16:11:38 (0) ** TXT file "D:\WMIDIAG-V2.0_XP__CLI.SP2.32_PC02_2006.09.28_16.11.38-REPORT.TXT" created.
....4 16:11:38 (0) ** WMIDIag v2.0 started on Thursday, September 28, 2006 at 16:11.
....5 16:11:38 (0) **
....6 16:11:38 (0) ** Copyright (c) Microsoft Corporation. All rights reserved - October 2006.
....7 16:11:38 (0) **
....8 16:11:38 (0) ** This script is not supported under any Microsoft standard support program or service.
....9 16:11:38 (0) ** The script is provided AS IS without warranty of any kind. Microsoft further disclaims all
...10 16:11:38 (0) ** implied warranties including, without limitation, any implied warranties of merchantability
...11 16:11:38 (0) ** or of fitness for a particular purpose. The entire risk arising out of the use or performance
...12 16:11:38 (0) ** of the scripts and documentation remains with you. In no event shall Microsoft, its authors,
...13 16:11:38 (0) ** or anyone else involved in the creation, production, or delivery of the script be liable for
...14 16:11:38 (0) ** any damages whatsoever (including, without limitation, damages for loss of business profits,
...15 16:11:38 (0) ** business interruption, loss of business information, or other pecuniary loss) arising out of
...16 16:11:38 (0) ** the use of or inability to use the script or documentation, even if Microsoft has been advised
...17 16:11:38 (0) ** of the possibility of such damages.
...18 16:11:38 (0) **
...19 16:11:38 (3) NOECHO=False
...20 16:11:38 (3) SILENT=False
...21 16:11:38 (3) SMS=False
...22 16:11:38 (3) FORCE=False
...23 16:11:38 (3) NOWINZIP=False
...24 16:11:38 (3) DEBUG=False
...25 16:11:38 (3) RUNONCE=False
...26 16:11:38 (3) DEPTH LEVEL=1
...27 16:11:38 (3) LOGGINGLEVEL=0
...28 16:11:38 (3) EVENTLOG=False
...29 16:11:38 (3) EVENTLOGERRORS=False
...30 16:11:38 (3) LOGWMISTATE=False
...31 16:11:38 (3) ERRORPOPUP=False
...32 16:11:38 (3) REQUESTALLINSTANCES=1
...33 16:11:38 (3) WRITEINREPOSITORY=False
...34 16:11:38 (3) CHECKCONSISTENCY=False
...35 16:11:38 (3) SHOWMOFERRORS=False
...36 16:11:38 (3) SHOWHIGHPRIVPROVIDERS=False
...37 16:11:38 (3) CORRELATECLASSANDPROVIDER=False
...38 16:11:38 (3) STRICT=False
...39 16:11:38 (3) OLDESTLOGHISTORY=0
...40 16:11:38 (3) OLDESTEVENTLOGHISTORY=20
...41 16:11:38 (3) BASENAMESPACE=Root
...42 16:11:38 (0) ** Verifying last run of WMIDIag.
...43 16:11:38 (4) Reading registry (REG_SZ) 'HKLM\Software\Microsoft\WMIDIag\LastRun'.
...44 16:11:38 (0) ** WMIDIag last run is 9/28/2006 4:03:58 PM.
...45 16:11:38 (2) !! WARNING: WMIDIag already started today.
...46 16:11:38 (0) ** Logging Run-time environment information.
...47 16:11:38 (3) StartMenu=C:\DOCUMENTS AND SETTINGS\ALAINL\START MENU\
...48 16:11:38 (3) Desktop=C:\DOCUMENTS AND SETTINGS\ALAINL\DESKTOP\
...49 16:11:38 (3) Programs=C:\DOCUMENTS AND SETTINGS\ALAINL\START MENU\PROGRAMS\
...50 16:11:38 (3) SystemDrive=C:\
...51 16:11:38 (3) SystemRoot=C:\WINDOWS\
...52 16:11:38 (3) Path=C:\WINDOWS\SYSTEM32;C:\WINDOWS\C:\WINDOWS\SYSTEM32\WBEM;
D:\APPS\MICROSOFT DEBUGGING TOOLS FOR WINDOWS;
D:\APPS\WINDOWS RESOURCE KIT TOOLS;D:\APPS\SUPPORT TOOLS;D:\APPS\SYSINTERNALS;
C:\PROGRAM FILES\ATI TECHNOLOGIES\ATI CONTROL PANEL;
C:\PROGRAM FILES\CA\SHAREDCOMPONENTS\SCANENGINE;
C:\PROGRAM FILES\CA\ETRUST ANTIVIRUS;D:\NTSPECS\TOOLS;D:\NTSPECS\TOOLS\X86;
...53 16:11:38 (3) UserTemp=D:\TEMP\USER\
...54 16:11:38 (3) SystemTemp=D:\TEMP\SYSTEM\
...55 16:11:38 (3) System32=C:\WINDOWS\SYSTEM32\
...56 16:11:38 (3) System=C:\WINDOWS\SYSTEM\
...57 16:11:38 (3) Wbem=C:\WINDOWS\SYSTEM32\WBEM\
...58 16:11:38 (3) WbemLogs=C:\WINDOWS\SYSTEM32\WBEM\LOGS\
...59 16:11:38 (3) CurrentDirectory=D:\DATA\ALAIN.LISSOIR\MICROSOFT DATA\WMIDIAG
...60 16:11:38 (3) AllUsersStartMenu=C:\DOCUMENTS AND SETTINGS\ALL USERS\START MENU\
...61 16:11:38 (3) AllUsersDesktop=C:\DOCUMENTS AND SETTINGS\ALL USERS\DESKTOP\
...62 16:11:38 (3) AllUsersPrograms=C:\DOCUMENTS AND SETTINGS\ALL USERS\START MENU\PROGRAMS\
...63 16:11:38 (3) ScriptName=WMIDIAG.VBS
...64 16:11:38 (3) ScriptFullName=D:\DATA\ALAIN.LISSOIR\MICROSOFT DATA\WMIDIAG\WMIDIAG.VBS
...65 16:11:38 (3) ScriptingName=WINDOWS SCRIPT HOST
...66 16:11:38 (3) EngineFullName=C:\WINDOWS\SYSTEM32\CSCRIPT.EXE
...67 16:11:38 (3) CommandLine=D:\DATA\ALAIN.LISSOIR\MICROSOFT DATA\WMIDIAG\WMIDIAG.VBS
...68 16:11:38 (3) EnginePath=C:\WINDOWS\SYSTEM32
...69 16:11:38 (3) EngineVersion=5.6
...70 16:11:38 (3) DomainName=PC-ALAINL-02
```

```

...71 16:11:38 (3)   UserName=ALAINL
...72 16:11:38 (3)   UserDNSDomain=
...73 16:11:38 (3)   LogonServerName=
...74 16:11:38 (3)   LocalComputerName=PC-ALAINL-02
...75 16:11:38 (3)   ProductName=Microsoft Windows XP
...76 16:11:38 (3)   Locale=00000409
...77 16:11:38 (3)   IsServerOS=False
...78 16:11:38 (3)   Is64=False
...79 16:11:38 (3)   IsWow64=False
...80 16:11:38 (3)   ProcessorArchitecture=32-bit
...81 16:11:38 (3)   ProcessorIdentifier=x86 Family 15 Model 2 Stepping 9, GenuineIntel
...82 16:11:38 (3)   NTVersion=5.1
...83 16:11:38 (3)   NTBuild=2600
...84 16:11:38 (3)   NTServicePack=Service Pack 2
...85 16:11:38 (3)   FullName=Windows XP - Service pack 2 - 32-bit
...86 16:11:38 (3)   ShortName=XP__CLI.SP2.32
...87 16:11:38 (3)   SMSAgent=2.50.4160
...88 16:11:38 (3)   WinZIP=D:\APPS\WINZIP\winzip32.exe

```

Verifying run-time environment (Upgrade, Privileges, WSH timeout, Files presence)

```

...89 16:11:38 (0) ** Initializing WMI System Information.
...90 16:11:38 (0) ** Windows XP - Service pack 2 - 32-bit (XP__CLI.SP2.32).
...91 16:11:38 (0) ** INFO: Windows system upgraded to Windows XP Service Pack 2 on Sunday, June 12, 2005.
...92 16:11:38 (4)   Reading registry (REG_BINARY)
                        'HKLM\SYSTEM\CurrentControlSet\Services\winmgmt\Security\Security'.
...95 16:11:38 (3)
...96 16:11:38 (3)   21 privilege(s):
...97 16:11:38 (3)   (X) SeChangeNotifyPrivilege = Bypass traverse checking
...98 16:11:38 (3)   (O) SeSecurityPrivilege = Manage auditing and security log
...99 16:11:38 (3)   (O) SeBackupPrivilege = Back up files and directories
...100 16:11:38 (3)   (O) SeRestorePrivilege = Restore files and directories
...101 16:11:38 (3)   (O) SeSystemtimePrivilege = Change the system time
...102 16:11:38 (3)   (O) SeShutdownPrivilege = Shut down the system
...103 16:11:38 (3)   (O) SeRemoteShutdownPrivilege = Force shutdown from a remote system
...104 16:11:38 (3)   (O) SeTakeOwnershipPrivilege = Take ownership of files or other objects
...105 16:11:38 (3)   (O) SeDebugPrivilege Debug programs
...106 16:11:38 (3)   (O) SeSystemEnvironmentPrivilege = Modify firmware environment values
...107 16:11:38 (3)   (O) SeSystemProfilePrivilege = Profile system performance
...108 16:11:38 (3)   (O) SeProfileSingleProcessPrivilege = Profile single process
...109 16:11:38 (3)   (O) SeIncreaseBasePriorityPrivilege = Increase scheduling priority
...110 16:11:38 (3)   (X) SeLoadDriverPrivilege Load and unload device drivers
...111 16:11:38 (3)   (O) SeCreatePagefilePrivilege = Create a pagefile
...112 16:11:38 (3)   (O) SeIncreaseQuotaPrivilege = Adjust memory quotas for a process
...113 16:11:38 (3)   (X) SeUndockPrivilege Remove computer from docking station
...114 16:11:38 (3)   (O) SeManageVolumePrivilege = Perform volume maintenance tasks
...115 16:11:38 (3)   (X) SeCreateGlobalPrivilege = Create global objects
...116 16:11:38 (3)   (X) SeImpersonatePrivilege = Impersonate a client after authentication
...117 16:11:38 (0) ** Verifying computer environment.
...118 16:11:38 (3)   The SYSTEM32 folder IS in the PATH.
...119 16:11:38 (3)   The WBEM folder IS in the PATH.
...120 16:11:38 (3)   The PATH environment variable has a maximum length of 2047 characters.
                        Current PATH length is 356 characters.
...121 16:11:38 (4)   Reading registry (REG_DWORD) 'HKCU\Software\Microsoft\Windows Script Host\Settings\Timeout'.
...122 16:11:38 (4)   Reading registry (REG_DWORD) 'HKLM\SOFTWARE\Microsoft\Windows Script Host\Settings\Timeout'.
...123 16:11:38 (0) ** Verifying specific files presence.
...124 16:11:38 (3)   File 'C:\WINDOWS\WMIPRVSE.EXE' is MISSING, which is FINE.
...125 16:11:38 (3)   File 'C:\WINDOWS\SYSTEM32\WMIPRVSE.EXE' is MISSING, which is FINE.
...126 16:11:38 (3)   File 'C:\WINDOWS\WINMGNT.EXE' is MISSING, which is FINE.
...127 16:11:38 (3)   File 'C:\WINDOWS\SYSTEM32\WINMGNT.EXE' is MISSING, which is FINE.
.
.
.
...127 16:11:38 (0) ** Verifying WMI System files presence at 'C:\WINDOWS\SYSTEM32\WBEM\'
...128 16:11:38 (2) !! WARNING: WMI System file 'C:\WINDOWS\SYSTEM32\WBEM\SMI2SMIR.EXE' is MISSING
                        or is access DENIED but it is an OPTIONAL component.
...129 16:11:38 (3)   Found 'C:\WINDOWS\SYSTEM32\WBEM\WMIPRVSE.EXE'.
...130 16:11:38 (3)   Found 'C:\WINDOWS\SYSTEM32\WBEM\MOFCOMP.EXE'.
...131 16:11:38 (3)   Found 'C:\WINDOWS\SYSTEM32\WBEM\SCRCONS.EXE'.
...132 16:11:38 (3)   Found 'C:\WINDOWS\SYSTEM32\WBEM\UNSECAPP.EXE'.
...133 16:11:38 (3)   Found 'C:\WINDOWS\SYSTEM32\WBEM\WBEMTEST.EXE'.
...134 16:11:38 (3)   Found 'C:\WINDOWS\SYSTEM32\WBEM\WINMGMT.EXE'.
.
.
.

```

Verifying repository file presence

```
..203 16:11:38 (0) ** Verifying WMI Repository files presence.
..204 16:11:38 (3) 'C:\WINDOWS\SYSTEM32\WBEM\Repository\FS' has a size of 30291660 bytes.
..205 16:11:38 (3) 'INDEX.BTR' has a size of 3268608 bytes
(Created: 7/15/2003 1:05:50 PM, Last Accessed: 9/28/2006 4:11:38 PM,
Last Modified: 9/28/2006 4:11:30 PM).
..206 16:11:38 (3) 'INDEX.MAP' has a size of 1852 bytes
(Created: 7/15/2003 1:05:52 PM, Last Accessed: 9/28/2006 4:11:38 PM,
Last Modified: 9/28/2006 4:11:30 PM).
..207 16:11:38 (3) 'MAPPING.VER' has a size of 4 bytes
(Created: 6/11/2005 5:22:14 PM, Last Accessed: 9/28/2006 4:11:38 PM,
Last Modified: 9/28/2006 4:11:30 PM).
..208 16:11:38 (3) 'MAPPING1.MAP' has a size of 15592 bytes
(Created: 6/11/2005 5:22:14 PM, Last Accessed: 9/28/2006 4:11:38 PM,
Last Modified: 9/28/2006 4:11:30 PM).
..209 16:11:38 (3) 'MAPPING2.MAP' has a size of 15596 bytes
(Created: 6/11/2005 5:22:14 PM, Last Accessed: 9/28/2006 4:11:38 PM,
Last Modified: 9/28/2006 4:11:05 PM).
..210 16:11:38 (3) 'OBJECTS.DATA' has a size of 26976256 bytes
(Created: 7/15/2003 1:05:50 PM, Last Accessed: 9/28/2006 4:11:38 PM,
Last Modified: 9/28/2006 4:11:30 PM).
..211 16:11:38 (3) 'OBJECTS.MAP' has a size of 13752 bytes
(Created: 7/15/2003 1:05:52 PM, Last Accessed: 9/28/2006 4:11:38 PM,
Last Modified: 9/28/2006 4:11:30 PM).
```

Verifying additional (unexpected) files in WBEM Folder

```
..212 16:11:38 (0) ** Verifying additional binaries in WBEM folder.
..213 16:11:39 (2) !! WARNING: 'WBEMDUMP.EXE' is an unexpected binary file located in C:\WINDOWS\SYSTEM32\WBEM\.
```

Verifying MOF files in auto-recovery registry key presence

```
..214 16:11:40 (0) ** Verifying Auto-Recovery MOF files presence.
..215 16:11:40 (4) Reading registry (REG_MULTI_SZ)
'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\Autorecover MOFs'.
..216 16:11:40 (0) ** Verifying MOF files in WBEM folder.
..217 16:11:40 (3) 'C:\WINDOWS\SYSTEM32\WBEM\FCONPROV.MFL' does exist and is
NOT LISTED BY DEFAULT in the 'Autorecover MOFs' registry key.
..218 16:11:40 (3) 'C:\WINDOWS\SYSTEM32\WBEM\FCONPROV.MOF' does exist and is
NOT LISTED BY DEFAULT in the 'Autorecover MOFs' registry key.
.
.
..236 16:11:41 (3) Found 'C:\WINDOWS\SYSTEM32\WBEM\CIMWIN32.MOF'.
..237 16:11:41 (3) Found 'C:\WINDOWS\SYSTEM32\WBEM\CIMWIN32.MFL'.
..238 16:11:41 (3) Found 'C:\WINDOWS\SYSTEM32\WBEM\SYSTEM.MOF'.
..239 16:11:41 (3) Found 'C:\WINDOWS\SYSTEM32\WBEM\WMIPCI.MOF'.
..240 16:11:41 (3) Found 'C:\WINDOWS\SYSTEM32\WBEM\WMIPCI.MFL'.
```

Verifying '#PRAGMA AUTORECOVER' statement presence in MOF files

```
..298 16:11:41 (0) ** Verifying '#PRAGMA AUTORECOVER' statement in MOF files.
..299 16:11:41 (2) !! WARNING: MOF file 'C:\WINDOWS\SYSTEM32\WBEM\FCONPROV.MFL' does NOT
contain the '#PRAGMA AUTORECOVER' statement.
..300 16:11:41 (2) !! WARNING: MOF file 'C:\WINDOWS\SYSTEM32\WBEM\FCONPROV.MOF' does NOT
contain the '#PRAGMA AUTORECOVER' statement.
..301 16:11:41 (2) !! WARNING: MOF file 'C:\WINDOWS\SYSTEM32\WBEM\NCPROV.MFL' does NOT
contain the '#PRAGMA AUTORECOVER' statement.
.
.
..320 16:11:47 (2) !! WARNING: MOF file 'C:\WINDOWS\SYSTEM32\WBEM\EVNTRPRV.MOF' does NOT contain
the '#PRAGMA AUTORECOVER' statement but it is included in the AUTORECOVERY LIST.
```

```

..321 16:11:47 (2) !! WARNING: MOF file 'C:\WINDOWS\SYSTEM32\WBEM\CMDEVTGPROV.MOF' does NOT contain
the '#PRAGMA AUTORECOVER' statement but it is included in the AUTORECOVERY LIST.
..322 16:11:47 (2) !! WARNING: MOF file 'C:\WINDOWS\SYSTEM32\WBEM\WHQLPROV.MOF' does NOT contain
the '#PRAGMA AUTORECOVER' statement but it is included in the AUTORECOVERY LIST.
.
.
.

```

Verifying DCOM configuration

```

..323 16:11:50 (0) ** Verifying DCOM configuration.
..324 16:11:50 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\Ole\EnabledDCOM'.
..325 16:11:50 (3)   DCOM is ENABLED.
..326 16:11:50 (4)   Reading registry (REG_DWORD) 'HKLM\SOFTWARE\Microsoft\Ole\LegacyAuthenticationLevel'.
..327 16:11:50 (3)   DCOM IS set at the CONNECT authentication level.
..328 16:11:50 (4)   Reading registry (REG_DWORD) 'HKLM\SOFTWARE\Microsoft\Ole\LegacyImpersonationLevel'.
..329 16:11:50 (3)   DCOM IS set at the IDENTIFY impersonation level.
.
.
.

```

Verifying WMI DCOM component registrations

```

..330 16:11:50 (0) ** Verifying WMI DCOM component registrations.
..331 16:11:50 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Classes\AppID\
{8BC3F05E-D86B-11D0-A075-00C04FB68820}'.
..332 16:11:50 (3)   WMI registry key (REG_SZ) '(Default)' is correct.
..333 16:11:50 (4)   Reading registry (REG_DWORD) 'HKLM\SOFTWARE\Classes\AppID\
{8BC3F05E-D86B-11D0-A075-00C04FB68820}\AuthenticationLevel'.
..334 16:11:50 (3)   WMI registry key (REG_DWORD) 'AuthenticationLevel' is missing, which is FINE.
..335 16:11:50 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Classes\AppID\
{8BC3F05E-D86B-11D0-A075-00C04FB68820}\LocalService'.
..336 16:11:50 (3)   WMI registry key (REG_SZ) 'LocalService' is correct.
..337 16:11:50 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Classes\CLSID\
{8BC3F05E-D86B-11D0-A075-00C04FB68820}'.
..338 16:11:50 (3)   WMI registry key (REG_SZ) '(Default)' is correct.
..339 16:11:50 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Classes\CLSID\
{8BC3F05E-D86B-11D0-A075-00C04FB68820}\AppID'.
..340 16:11:50 (3)   WMI registry key (REG_SZ) 'AppID' is correct.
..341 16:11:50 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Classes\CLSID\
{8BC3F05E-D86B-11D0-A075-00C04FB68820}\LocalService'.
.
.
.

```

Verifying DCOM security

```

..776 16:11:50 (0) ** Verifying WMI DCOM component security.
..777 16:11:50 (3)   Deciphering DCOM security for 'My Computer' (DefaultAccessPermission)
..778 16:11:50 (4)   Reading registry (REG_BINARY) 'HKLM\SOFTWARE\Microsoft\Ole\DefaultAccessPermission'.
..779 16:11:50 (4)   +- Security Descriptor -----
..780 16:11:50 (4)   | Owner: ..... BUILTIN\Administrators
..781 16:11:50 (4)   | Group: ..... BUILTIN\Administrators
..782 16:11:50 (4)   | Revision: ..... 1
..783 16:11:50 (4)   | Control: ..... &h8004
..784 16:11:50 (4)   | SE_DACL_PRESENT
..785 16:11:50 (4)   | SE_SELF_RELATIVE
..786 16:11:50 (4)   |+- DiscretionaryAcl -----
..787 16:11:50 (4)   ||+- ACE #01 -----
..788 16:11:50 (4)   ||| Trustee: ..... NT AUTHORITY\SELF
..789 16:11:50 (4)   ||| AceType: ..... &h0
..790 16:11:50 (4)   | ACCESS_ALLOWED_ACE_TYPE
..791 16:11:50 (4)   ||| AceFlags: ..... &h0
..792 16:11:50 (4)   ||| AccessMask: ..... &h7
..793 16:11:50 (4)   | DCOM_RIGHT_EXECUTE
..794 16:11:50 (4)   | DCOM_RIGHT_ACCESS_LOCAL
..795 16:11:50 (4)   | DCOM_RIGHT_ACCESS_REMOTE
..796 16:11:50 (4)   ||+-----
..797 16:11:50 (4)   ||+- ACE #02 -----
..798 16:11:50 (4)   ||| Trustee: ..... NT AUTHORITY\SYSTEM
..799 16:11:50 (4)   ||| AceType: ..... &h0
..800 16:11:50 (4)   | ACCESS_ALLOWED_ACE_TYPE
..801 16:11:50 (4)   ||| AceFlags: ..... &h0

```

```

..802 16:11:50 (4)    ||| AccessMask: ..... &h3
..803 16:11:50 (4)                                DCOM_RIGHT_EXECUTE
..804 16:11:50 (4)                                DCOM_RIGHT_ACCESS_LOCAL
..805 16:11:50 (4)    ||+-----+
..806 16:11:50 (4)    |+-----+
..807 16:11:50 (4)    +-----+
..808 16:11:50 (3)    Verifying actual trustees in ACEs against the default trustees in
                        ACEs to locate actual trustee additions.
..809 16:11:50 (3)    Verifying default trustee in ACEs against the actual trustees in
                        ACEs to locate default trustee removals.
..810 16:11:50 (3)
..811 16:11:50 (3)    Deciphering DCOM security for 'My Computer' (MachineAccessRestriction)
..812 16:11:50 (4)    Reading registry (REG_BINARY) 'HKLM\SOFTWARE\Microsoft\Ole\MachineAccessRestriction'.
.
.
..845 16:11:50 (3)    Deciphering DCOM security for 'My Computer' (DefaultLaunchPermission)
..846 16:11:50 (4)    Reading registry (REG_BINARY) 'HKLM\SOFTWARE\Microsoft\Ole\DefaultLaunchPermission'.
.
.
..896 16:11:50 (3)    Deciphering DCOM security for 'My Computer' (MachineLaunchRestriction)
..897 16:11:50 (4)    Reading registry (REG_BINARY) 'HKLM\SOFTWARE\Microsoft\Ole\MachineLaunchRestriction'.
.
.
..946 16:11:50 (3)    Deciphering DCOM security for 'Windows Management Instrumentation' (AccessPermission)
..947 16:11:50 (4)    Reading registry (REG_BINARY) 'HKLM\SOFTWARE\Classes\AppID\
                        {8BC3F05E-D86B-11D0-A075-00C04FB68820}\AccessPermission'.
.
.
..980 16:11:50 (3)    Deciphering DCOM security for 'Windows Management Instrumentation' (LaunchPermission)
..981 16:11:50 (4)    Reading registry (REG_BINARY) 'HKLM\SOFTWARE\Classes\AppID\
                        {8BC3F05E-D86B-11D0-A075-00C04FB68820}\LaunchPermission'.
.
.

```

Verifying WMI ProgID registrations

```

.1258 16:11:51 (0) ** Verifying WMI ProgID registrations.
.1259 16:11:51 (3)    WMI object ProgID 'WBemscripting.SWBemlocator' instantiated'.
.1260 16:11:51 (3)    WMI object ProgID 'WbemScripting.SWbemDateTime' instantiated'.
.1261 16:11:51 (3)    WMI object ProgID 'WbemScripting.SWbemObjectPath' instantiated'.
.1262 16:11:51 (3)    WMI object ProgID 'WbemScripting.SWbemSink' instantiated'.
.1263 16:11:51 (3)    WMI object ProgID 'WbemScripting.SWbemLocator' instantiated'.
.1264 16:11:51 (3)    WMI object ProgID 'WbemScripting.SWbemNamedValueSet' instantiated'.
.1265 16:11:51 (3)    WMI object ProgID 'WbemScripting.SWbemRefresher' instantiated'.
.
.

```

Verifying Windows Firewall settings

```

.1266 16:11:51 (0) ** Verifying Windows Firewall setup.
.1267 16:11:51 (4)    Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\SharedAccess\DisplayName'.
.1268 16:11:51 (3)    FW/ICS service is running.
.1269 16:11:51 (3)    FW/ICS DOMAIN profile is the current active one.
.1270 16:11:51 (3)    FW/ICS status is ENABLED.
.1271 16:11:51 (2) !! WARNING: FW/ICS 'RemoteAdmin' is DISABLED, preventing remote WMI connections to this machine.
.1272 16:11:51 (2) !! WARNING: 'UNSECAPP.EXE' is not authorized, preventing WMI asynchronous callbacks
                        to this computer for scripts and MMC applications.
.
.

```

Verifying Windows Vista Firewall settings

```

.3036 13:32:00 (0) ** Verifying Windows Firewall setup.
.3037 13:32:00 (4)    Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\MpsSvc\DisplayName'.
.3038 13:32:00 (4)    (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\MpsSvc\DisplayName' ->
                        '@%SystemRoot%\system32\FirewallAPI.dll,-23090'
.3039 13:32:00 (3)    Windows Firewall service is running.

```

```

.3040 13:32:00 (3) Current firewall profile is DOMAIN.
.3041 13:32:00 (3) - Inbound connections that do not match a rule are BLOCKED.
.3042 13:32:00 (3) - Outbound connections that do not match a rule are ALLOWED.
.3043 13:32:00 (3)
.3044 13:32:00 (3) Group rule name: Windows Management Instrumentation (WMI)
.3045 13:32:00 (3) Enabled: False
.3345 13:32:00 (3)
.3346 13:32:00 (3) Rule Name: Windows Management Instrumentation (ASync-In)
.3347 13:32:00 (3) Description: Inbound rule to allow Asynchronous WMI traffic for remote
Windows Management Instrumentation. [TCP]
.3348 13:32:00 (3) Enabled: False
.3349 13:32:00 (3) Application Name: C:\WINDOWS\SYSTEM32\WBEM\UNSECAPP.EXE
.3350 13:32:00 (3) Rule profile: DOMAIN
.3351 13:32:00 (3) IP Protocol: TCP
.3352 13:32:00 (3) Local Ports: *
.3353 13:32:00 (3) Remote Ports: *
.3354 13:32:00 (3) LocalAddresses: *
.3355 13:32:00 (3) RemoteAddresses: *
.3356 13:32:00 (3) Direction: In
.3357 13:32:00 (3) Action: Allow
.3358 13:32:00 (3) Edge: False
.3359 13:32:00 (3) Interface Types: All
.3360 13:32:00 (3) There are no excluded interfaces
.3366 13:32:00 (3)
.3367 13:32:00 (3) Rule Name: Windows Management Instrumentation (WMI-Out)
.3368 13:32:00 (3) Description: Outbound rule to allow WMI traffic for remote
Windows Management Instrumentation. [TCP]
.3369 13:32:00 (3) Enabled: False
.3370 13:32:00 (3) Application Name: C:\WINDOWS\SYSTEM32\SVCHOST.EXE
.3371 13:32:00 (3) Rule profile: DOMAIN
.3372 13:32:00 (3) IP Protocol: TCP
.3373 13:32:00 (3) Local Ports: *
.3374 13:32:00 (3) Remote Ports: *
.3375 13:32:00 (3) LocalAddresses: *
.3376 13:32:00 (3) RemoteAddresses: *
.3377 13:32:00 (3) Direction: Out
.3378 13:32:00 (3) Action: Allow
.3379 13:32:00 (3) Edge: False
.3380 13:32:00 (3) Interface Types: All
.3381 13:32:00 (3) There are no excluded interfaces
.3384 13:32:00 (3)
.3385 13:32:00 (3) Rule Name: Windows Management Instrumentation (WMI-In)
.3386 13:32:00 (3) Description: Inbound rule to allow WMI traffic for remote
Windows Management Instrumentation. [TCP]
.3387 13:32:00 (3) Enabled: False
.3388 13:32:00 (3) Application Name: C:\WINDOWS\SYSTEM32\SVCHOST.EXE
.3389 13:32:00 (3) Rule profile: DOMAIN
.3390 13:32:00 (3) IP Protocol: TCP
.3391 13:32:00 (3) Local Ports: *
.3392 13:32:00 (3) Remote Ports: *
.3393 13:32:00 (3) LocalAddresses: *
.3394 13:32:00 (3) RemoteAddresses: *
.3395 13:32:00 (3) Direction: In
.3396 13:32:00 (3) Action: Allow
.3397 13:32:00 (3) Edge: False
.3398 13:32:00 (3) Interface Types: All
.3399 13:32:00 (3) There are no excluded interfaces
.3403 13:32:00 (3)
.3404 13:32:00 (3) Rule Name: Windows Management Instrumentation (DCOM-In)
.3405 13:32:00 (3) Description: Inbound rule to allow DCOM traffic for remote
Windows Management Instrumentation. [TCP 135]
.3406 13:32:00 (3) Enabled: False
.3407 13:32:00 (3) Application Name: C:\WINDOWS\SYSTEM32\SVCHOST.EXE
.3408 13:32:00 (3) Rule profile: DOMAIN
.3409 13:32:00 (3) IP Protocol: TCP
.3410 13:32:00 (3) Local Ports: 135
.3411 13:32:00 (3) Remote Ports: *
.3412 13:32:00 (3) LocalAddresses: *
.3413 13:32:00 (3) RemoteAddresses: *
.3414 13:32:00 (3) Direction: In
.3415 13:32:00 (3) Action: Allow
.3416 13:32:00 (3) Edge: False
.3417 13:32:00 (3) Interface Types: All
.3418 13:32:00 (3) There are no excluded interfaces
.4329 13:32:00 (3)
.4330 13:32:00 (3) Changing or adding a firewall rule (or group) to the current profile will take effect.

```

Verifying WMI Core registry settings

```
.1273 16:11:51 (0) ** Verifying WMI Core registry settings (WBEM).
.1274 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\Scripting\Default Namespace'.
.1275 16:11:51 (3)   WMI registry key (REG_SZ) 'Default Namespace' is correct.
.1276 16:11:51 (4)   Reading registry (REG_DWORD) 'HKLM\SOFTWARE\Microsoft\WBEM\Scripting\
                        Default Impersonation Level'.
.1277 16:11:51 (3)   WMI registry key (REG_DWORD) 'Default Impersonation Level' is correct.
.1278 16:11:51 (4)   Reading registry (REG_DWORD) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\ADAPdelay'.
.1279 16:11:51 (3)   WMI registry key (REG_DWORD) 'ADAPdelay' is correct.
.1280 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\Default Repository Driver'.
.1281 16:11:51 (3)   WMI registry key (REG_SZ) 'Default Repository Driver' is correct.
.1282 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\EnableEvents'.
.1283 16:11:51 (3)   WMI registry key (REG_SZ) 'EnableEvents' is correct.
.1284 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\Logging'.
.1285 16:11:51 (3)   WMI registry key (REG_SZ) 'Logging' is correct.
.1286 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\Logging Directory'.
.1287 16:11:51 (3)   WMI registry key (REG_EXPAND_SZ) 'Logging Directory' is correct.
.1288 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\Repository Directory'.
.1289 16:11:51 (3)   WMI registry key (REG_EXPAND_SZ) 'Repository Directory' is correct.
.1290 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\TimeOutMs'.
.1291 16:11:51 (3)   WMI registry key (REG_SZ) 'TimeOutMs' is correct.
.1292 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\WMISetup'.
.1293 16:11:51 (3)   WMI registry key (REG_SZ) 'WMISetup' is correct.
.1294 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\Working Directory'.
.1295 16:11:51 (3)   WMI registry key (REG_EXPAND_SZ) 'Working Directory' is correct.
.1296 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\Autorecover MOFs'.
.1297 16:11:51 (3)   WMI registry hive (REG_SZ) 'Autorecover MOFs' is present.
.1298 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\Log File Max Size'.
.1299 16:11:51 (3)   WMI registry key (REG_SZ) 'Log File Max Size' is correct.
.
.
.
```

Verifying WMI Service registry settings

```
.1300 16:11:51 (0) ** Verifying WMI Service registry settings (SVCHOST, WINMGMT).
.1301 16:11:51 (4)   Reading registry (REG_DWORD) 'HKLM\SYSTEM\CurrentControlSet\Services\Winmgmt\Start'.
.1302 16:11:51 (4)   Reading registry (REG_MULTI_SZ) 'HKLM\SOFTWARE\Microsoft\
                        Windows NT\CurrentVersion\SvcHost\netsvcs'.
.1303 16:11:51 (4)   Reading registry (REG_MULTI_SZ) 'HKLM\SOFTWARE\Microsoft\
                        Windows NT\CurrentVersion\SvcHost\winmgmt'.
.1304 16:11:51 (3)   'Windows Management Instrumentation' (WINMGMT) is running as a SHARED HOST SERVICE.
.1305 16:11:51 (4)   Reading registry (REG_DWORD) 'HKLM\SYSTEM\CurrentControlSet\Services\winmgmt\Type'.
.1306 16:11:51 (3)   WMI registry key (REG_DWORD) 'Type' is correct.
.1307 16:11:51 (4)   Reading registry (REG_DWORD) 'HKLM\SYSTEM\CurrentControlSet\Services\winmgmt\Start'.
.1308 16:11:51 (3)   WMI registry key (REG_DWORD) 'Start' is correct.
.
.
.
.1338 16:11:51 (0) ** Verifying WMI Service known dependents.
.1339 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\wscsvc\DisplayName'.
.1340 16:11:51 (4)   Reading registry (REG_MULTI_SZ)
                        'HKLM\SYSTEM\CurrentControlSet\Services\wscsvc\DependOnService'.
.1341 16:11:51 (4)   Reading registry (REG_DWORD) 'HKLM\SYSTEM\CurrentControlSet\Services\wscsvc\Start'.
.1342 16:11:51 (2) !! WARNING: 'Security Center' (WCSVC, StartMode='Automatic') is installed depends on WMI Service.
.1343 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\SharedAccess\DisplayName'.
.1344 16:11:51 (4)   Reading registry (REG_MULTI_SZ)
                        'HKLM\SYSTEM\CurrentControlSet\Services\SharedAccess\DependOnService'.
.1345 16:11:51 (4)   Reading registry (REG_DWORD) 'HKLM\SYSTEM\CurrentControlSet\Services\SharedAccess\Start'.
.1346 16:11:51 (2) !! WARNING: 'Windows Firewall/Internet Connection Sharing (ICS)'
                        (SHAREDACCESS, StartMode='Automatic') is installed depends on WMI Service.
.1347 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\CcmExec\DisplayName'.
.1348 16:11:51 (4)   Reading registry (REG_MULTI_SZ)
                        'HKLM\SYSTEM\CurrentControlSet\Services\CcmExec\DependOnService'.
.1349 16:11:51 (4)   Reading registry (REG_DWORD) 'HKLM\SYSTEM\CurrentControlSet\Services\CcmExec\Start'.
.1350 16:11:51 (2) !! WARNING: 'SMS Agent Host' (CCMEXEC, StartMode='Automatic') is installed depends on WMI Service.
.1351 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\6to4\DisplayName'.
.1352 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\MSExchangeMGMT\DisplayName'.
.1353 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\IIMFilter\DisplayName'.
.1354 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\RtcSrv\DisplayName'.
.1355 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\MngAgent\DisplayName'.
.1356 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\ADSBuilder\DisplayName'.
.1357 16:11:51 (4)   Reading registry (REG_SZ) 'HKLM\SYSTEM\CurrentControlSet\Services\ADSPXE\DisplayName'.
.1358 16:11:51 (0) ** Verifying 'RPCSS' service status.
.1359 16:11:51 (3)   Verifying 'RPCSS' service started state.
.1360 16:11:51 (3)   'RPCSS' service IS started.
```

```
.1361 16:11:51 (0) ** Verifying 'WINMGMT' service status.
.1362 16:11:51 (3)   Verifying 'WINMGMT' service started state.
.1363 16:11:51 (3)   'WINMGMT' service IS started.
```

Verifying loaded providers before/after WMI Diagnosis Tool execution

```
.4507 13:32:17 (0) ** Verifying WMI providers loaded BEFORE WMIDiag execution.
.4508 13:32:17 (3)   8 WMI providers are currently loaded:
.4509 13:32:18 (4)     Provider: 'POLICYAGENTINSTANCEPROVIDER' (PID: 4092, C:\WINDOWS\SYSTEM32\WBEM\WMIIPRVSE.EXE)
.4510 13:32:18 (4)       Provider Name: 'SMS Policy Agent Event/Instance WMI Provider'
.4511 13:32:18 (4)       Namespace: 'ROOT\CCM\POLICY\MACHINE' (i.e. 'CCM_Policy')
.4512 13:32:18 (4)       Hosting Model: 'LOCALSYSTEMHOST:CCM' (HostingSpecification=5)
.4513 13:32:18 (4)       Hosting Group: 'CCM'
.4514 13:32:18 (4)       User: ''
.4515 13:32:18 (4)       Provider Binary: 'C:\WINDOWS\SYSTEM32\CCM\POLICYAGENTPROVIDER.DLL'
.4516 13:32:18 (4)       MOF Registration: 'No located MOF file (exception)'
.4517 13:32:18 (4)
.4518 13:32:18 (4)     Provider: 'POLICYAGENTINSTANCEPROVIDER' (PID: 4092, C:\WINDOWS\SYSTEM32\WBEM\WMIIPRVSE.EXE)
.4519 13:32:18 (4)       Provider Name: 'SMS Policy Agent Event/Instance WMI Provider'
.4520 13:32:18 (4)       Namespace: 'ROOT\CCM\POLICY\S_1_5_21_2127521184_1604012920_1887927527_2058479'
.4521 13:32:18 (4)       (i.e. 'CCM_Policy')
.4522 13:32:18 (4)       Hosting Model: 'LOCALSYSTEMHOST:CCM' (HostingSpecification=5)
.4523 13:32:18 (4)       Hosting Group: 'CCM'
.4524 13:32:18 (4)       User: ''
.4525 13:32:18 (4)       Provider Binary: 'C:\WINDOWS\SYSTEM32\CCM\POLICYAGENTPROVIDER.DLL'
.4526 13:32:18 (4)       MOF Registration: 'No located MOF file (exception)'
```

```
17320 13:45:57 (0) ** Verifying WMI providers loaded AFTER WMIDiag execution.
17321 13:45:57 (3)   10 WMI providers are currently loaded:
17322 13:45:57 (4)     Provider: 'POLICYAGENTINSTANCEPROVIDER' (PID: 4092, C:\WINDOWS\SYSTEM32\WBEM\WMIIPRVSE.EXE)
17323 13:45:57 (4)       Provider Name: 'SMS Policy Agent Event/Instance WMI Provider'
17324 13:45:57 (4)       Namespace: 'ROOT\CCM\POLICY\MACHINE' (i.e. 'CCM_Policy')
17325 13:45:57 (4)       Hosting Model: 'LOCALSYSTEMHOST:CCM' (HostingSpecification=5)
17326 13:45:57 (4)       Hosting Group: 'CCM'
17327 13:45:57 (4)       User: ''
17328 13:45:57 (4)       Provider Binary: 'C:\WINDOWS\SYSTEM32\CCM\POLICYAGENTPROVIDER.DLL'
17329 13:45:57 (4)       MOF Registration: 'No located MOF file (exception)'
17330 13:45:57 (4)
17331 13:45:57 (4)     Provider: 'POLICYAGENTINSTANCEPROVIDER' (PID: 4092, C:\WINDOWS\SYSTEM32\WBEM\WMIIPRVSE.EXE)
17332 13:45:57 (4)       Provider Name: 'SMS Policy Agent Event/Instance WMI Provider'
17333 13:45:57 (4)       Namespace: 'ROOT\CCM\POLICY\S_1_5_21_2127521184_1604012920_1887927527_2058479'
17334 13:45:57 (4)       (i.e. 'CCM_Policy')
17335 13:45:57 (4)       Hosting Model: 'LOCALSYSTEMHOST:CCM' (HostingSpecification=5)
17336 13:45:57 (4)       Hosting Group: 'CCM'
17337 13:45:57 (4)       User: ''
17338 13:45:57 (4)       Provider Binary: 'C:\WINDOWS\SYSTEM32\CCM\POLICYAGENTPROVIDER.DLL'
17339 13:45:57 (4)       MOF Registration: 'No located MOF file (exception)'
17340 13:45:57 (4)
17341 13:45:57 (4)     Provider: 'CIMWIN32' (PID: 3568, C:\WINDOWS\SYSTEM32\WBEM\WMIIPRVSE.EXE)
17342 13:45:57 (4)       Provider Name: 'CIMv2 Core OS Instance/Method WMI Provider'
17343 13:45:57 (4)       Namespace: 'ROOT\CIMV2' (i.e. 'Win32_TimeZone')
17344 13:45:57 (4)       Hosting Model: 'NETWORKSERVICEHOST' (HostingSpecification=8)
17345 13:45:57 (4)       Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
17346 13:45:57 (4)       User: ''
17347 13:45:57 (4)       Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\CIMWIN32.DLL'
17348 13:45:57 (4)       MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\CIMWIN32.MOF /
C:\WINDOWS\SYSTEM32\WBEM\SECRCW32.MOF'
17349 13:45:57 (4)
17350 13:45:57 (4)     Provider: 'MS_NT_EVENTLOG_PROVIDER' (PID: 3568, C:\WINDOWS\SYSTEM32\WBEM\WMIIPRVSE.EXE)
17351 13:45:57 (4)       Provider Name: 'EventLog Method/Instance WMI Provider'
17352 13:45:57 (4)       Namespace: 'ROOT\CIMV2' (i.e. 'Win32_NTLogEvent')
17353 13:45:57 (4)       Hosting Model: 'NETWORKSERVICEHOST' (HostingSpecification=8)
17354 13:45:57 (4)       Hosting Group: 'DEFAULTNETWORKSERVICEHOST'
17355 13:45:57 (4)       User: 'REDMOND\ALAINL'
17356 13:45:57 (4)       Provider Binary: 'C:\WINDOWS\SYSTEM32\WBEM\NTEVT.DLL'
17357 13:45:57 (4)       MOF Registration: 'C:\WINDOWS\SYSTEM32\WBEM\NTEVT.MOF'
17358 13:45:57 (4)
17359 13:45:57 (4)     Provider: 'MSFT_PROVIDERSUBSYSTEM' (PID: 1052, C:\WINDOWS\SYSTEM32\SVCHOST.EXE)
17360 13:45:57 (4)       Provider Name: 'Microsoft Provider Sub-system Instance/Method WMI Provider'
17361 13:45:57 (4)       Namespace: 'ROOT\CIMV2' (i.e. 'Msft_Providers')
```

```

17361 13:45:57 (4)      Hosting Model:      'WMICORE' (HostingSpecification=1)
17362 13:45:57 (4)      Hosting Group:      ''
17363 13:45:57 (4)      User:               ''
17364 13:45:57 (4)      Provider Binary:   'C:\WINDOWS\SYSTEM32\WBEM\WMIPRVSD.DLL'
17365 13:45:57 (4)      MOF Registration:  'C:\WINDOWS\SYSTEM32\WBEM\SYSTEM.MOF'
17366 13:45:57 (4)
17367 13:45:57 (4)      Provider: 'SCM EVENT PROVIDER' (PID: 1052, C:\WINDOWS\SYSTEM32\SVCHOST.EXE)
17368 13:45:57 (4)      Provider Name:     'Service Control Manager Event WMI Provider'
17369 13:45:57 (4)      Namespace:         'ROOT\CIMV2' (i.e. 'MSFT_SCMEvent')
17370 13:45:57 (4)      Hosting Model:     'DECOUPLED:NONCOM' (HostingSpecification=10)
17371 13:45:57 (4)      Hosting Group:     ''
17372 13:45:57 (4)      User:               ''
17373 13:45:57 (4)      Provider Binary:   'No located BIN file'
17374 13:45:57 (4)      MOF Registration:  'No located MOF file (exception)'
17375 13:45:57 (4)
17376 13:45:57 (4)      Provider: 'WMIPERFCLASS' (PID: 5612, C:\WINDOWS\SYSTEM32\WBEM\WMIPRVSE.EXE)
17377 13:45:57 (4)      Provider Name:     'Performance Counter Pull Class WMI Provider'
17378 13:45:57 (4)      Namespace:         'ROOT\CIMV2' (i.e. '*')
17379 13:45:57 (4)      Hosting Model:     'LOCALSYSTEMHOST' (HostingSpecification=5)
17380 13:45:57 (4)      Hosting Group:     'DEFAULTLOCALSYSTEMHOST'
17381 13:45:57 (4)      User:               ''
17382 13:45:57 (4)      Provider Binary:   'C:\WINDOWS\SYSTEM32\WBEM\WMIPERFCLASS.DLL'
17383 13:45:57 (4)      MOF Registration:  'C:\WINDOWS\SYSTEM32\WBEM\WMIPERFCLASS.MOF'
17384 13:45:57 (4)
17385 13:45:57 (4)      Provider: 'NTEVENTLOGEVENTCONSUMER' (PID: 1052, C:\WINDOWS\SYSTEM32\SVCHOST.EXE)
17386 13:45:57 (4)      Provider Name:     'EventLog Event/consumer WMI Provider'
17387 13:45:57 (4)      Namespace:         'ROOT\SUBSCRIPTION' (i.e. 'NTEventLogEventConsumer')
17388 13:45:57 (4)      Hosting Model:     'WMICORE' (HostingSpecification=1)
17389 13:45:57 (4)      Hosting Group:     ''
17390 13:45:57 (4)      User:               ''
17391 13:45:57 (4)      Provider Binary:   'C:\WINDOWS\SYSTEM32\WBEM\WBEMCONS.DLL'
17392 13:45:57 (4)      MOF Registration:  'C:\WINDOWS\SYSTEM32\WBEM\WBEMCONS.MOF'
17393 13:45:57 (4)
17394 13:45:57 (4)      Provider: 'WMIPROV' (PID: 5612, C:\WINDOWS\SYSTEM32\WBEM\WMIPRVSE.EXE)
17395 13:45:57 (4)      Provider Name:     'Windows Driver Model (WDM) Instance/Push Class/Method WMI Provider'
17396 13:45:57 (4)      Namespace:         'ROOT\WMI' (i.e. 'MSiSCSI_Eventlog')
17397 13:45:57 (4)      Hosting Model:     'LOCALSYSTEMHOST' (HostingSpecification=5)
17398 13:45:57 (4)      Hosting Group:     'DEFAULTLOCALSYSTEMHOST'
17399 13:45:57 (4)      User:               'NT AUTHORITY\SYSTEM'
17400 13:45:57 (4)      Provider Binary:   'C:\WINDOWS\SYSTEM32\WBEM\WMIPROV.DLL'
17401 13:45:57 (4)      MOF Registration:  'C:\WINDOWS\SYSTEM32\WBEM\WMI.MOF'
17402 13:45:57 (4)
17403 13:45:57 (4)      Provider: 'WMIPROV' (PID: 5612, C:\WINDOWS\SYSTEM32\WBEM\WMIPRVSE.EXE)
17404 13:45:57 (4)      Provider Name:     'Windows Driver Model (WDM) Instance/Push Class/Method WMI Provider'
17405 13:45:57 (4)      Namespace:         'ROOT\WMI' (i.e. 'MSiSCSI_Eventlog')
17406 13:45:57 (4)      Hosting Model:     'LOCALSYSTEMHOST' (HostingSpecification=5)
17407 13:45:57 (4)      Hosting Group:     'DEFAULTLOCALSYSTEMHOST'
17408 13:45:57 (4)      User:               'REDMOND\ALAINL'
17409 13:45:57 (4)      Provider Binary:   'C:\WINDOWS\SYSTEM32\WBEM\WMIPROV.DLL'
17410 13:45:57 (4)      MOF Registration:  'C:\WINDOWS\SYSTEM32\WBEM\WMI.MOF'
.
.
.

```

Verifying WMI Root namespace settings

```

.1414 16:11:51 (0) ** Verifying WMI namespace 'Root' (L=1).
.1415 16:11:51 (3)   Retrieving WMI system class(es) static information.
.1416 16:11:51 (3)   45/45 system class(es) found.
.1417 16:11:51 (3)   Verifying Permanent subscription(s) for 'Root'.
.1418 16:11:51 (3)   0 permanent subscription(s) in 'Root' namespace.
.1419 16:11:51 (3)   0 Timer instruction(s) in 'Root' namespace.
.1420 16:11:51 (0) ** Verifying WMI system settings.
.1421 16:11:51 (3)   - __CIMOMIdentification #1 -----
.1422 16:11:51 (3)   SetupDate: ..... Thursday, September 07, 2006 GMT
.1423 16:11:51 (3)   SetupTime: ..... 6:39:07 PM GMT
.1424 16:11:51 (3)   VersionCurrentlyRunning: ..... 5.1.2600.2180
.1425 16:11:51 (3)   VersionUsedToCreateDB: ..... 5.1.2600.2180
.1426 16:11:51 (3)   WorkingDirectory: ..... %SystemRoot%\system32\WBEM
.1427 16:11:51 (3)   - __ArbitratorConfiguration #1 -----
.1428 16:11:51 (3)   OutstandingTasksPerUser: ..... 30
.1429 16:11:51 (3)   OutstandingTasksTotal: ..... 3000
.1430 16:11:51 (3)   PermanentSubscriptionsPerUser: ..... 1000
.1431 16:11:51 (3)   PermanentSubscriptionsTotal: ..... 10000
.1432 16:11:51 (3)   PollingInstructionsPerUser: ..... 1000
.1433 16:11:51 (3)   PollingInstructionsTotal: ..... 10000
.1434 16:11:51 (3)   PollingMemoryPerUser: ..... 5000000
.1435 16:11:51 (3)   PollingMemoryTotal: ..... 10000000
.1436 16:11:51 (3)   QuotaRetryCount: ..... 10

```

```

.1437 16:11:51 (3) QuotaRetryWaitInterval: ..... 15000
.1438 16:11:51 (3) TaskThreadsPerUser: ..... 3
.1439 16:11:51 (3) TaskThreadsTotal: ..... 30
.1440 16:11:51 (3) TemporarySubscriptionsPerUser: ..... 1000
.1441 16:11:51 (3) TemporarySubscriptionsTotal: ..... 10000
.1442 16:11:51 (3) TotalCacheDisk: ..... 1048576
.1443 16:11:51 (3) TotalCacheDiskPerTask: ..... 51250
.1444 16:11:51 (3) TotalCacheDiskPerUser: ..... 102500
.1445 16:11:51 (3) TotalCacheMemory: ..... 10240
.1446 16:11:51 (3) TotalCacheMemoryPerTask: ..... 1024
.1447 16:11:51 (3) TotalCacheMemoryPerUser: ..... 2048
.1448 16:11:51 (3) TotalUsers: ..... 50
.1449 16:11:51 (3) - __ProviderHostQuotaConfiguration #1 -----
.1450 16:11:51 (3) HandlesPerHost: ..... 4096
.1451 16:11:51 (3) MemoryAllHosts: ..... 1073741824
.1452 16:11:51 (3) MemoryPerHost: ..... 134217728
.1453 16:11:51 (3) ProcessLimitAllHosts: ..... 32
.1454 16:11:51 (3) ThreadsPerHost: ..... 256
.1455 16:11:51 (3) - __EventConsumerProviderCacheControl #1 -----
.1456 16:11:51 (3) ClearAfter: ..... 00000000000030.000000:000
.1457 16:11:51 (3) - __EventProviderCacheControl #2 -----
.1458 16:11:51 (3) ClearAfter: ..... 00000000000030.000000:000
.1459 16:11:51 (3) - __ObjectProviderCacheControl #3 -----
.1460 16:11:52 (3) ClearAfter: ..... 00000000000030.000000:000
.1461 16:11:52 (3) - __EventSinkCacheControl #4 -----
.1462 16:11:52 (3) ClearAfter: ..... 00000000000015.000000:000
.1463 16:11:52 (3) - __PropertyProviderCacheControl #5 -----
.1464 16:11:52 (3) ClearAfter: ..... 00000000000030.000000:000
.
.
.

```

Verifying existing static instances

```

.1531 16:11:52 (3) 0 WMI Provider CIM registration(s) found.
.1532 16:11:52 (3) 61 class(es) found. (Inheritance level analysis=1).
.1533 16:11:52 (3) Retrieving static information (MOF) of '__SystemClass' (I=1).
.1534 16:11:52 (3) Qualifier information of '__SystemClass': Dynamic=False,
    Provider='<NOT DEFINED>', Association=False.
.1535 16:11:52 (3) Requesting all static instances of '__SystemClass' in 'Root' (9/28/2006 4:11:52 PM) ...
.1536 16:11:52 (3) 0 static instance(s) found for '__SystemClass' in 'Root'.
.1537 16:11:52 (3) Retrieving static information (MOF) of '__thisNAMESPACE' (I=1).
.1538 16:11:52 (3) Qualifier information of '__thisNAMESPACE': Dynamic=False,
    Provider='<NOT DEFINED>', Association=False.
.
.
.
.1547 16:11:52 (3) Requesting all static instances of '__EventConsumerProviderCacheControl' in 'Root'
    (9/28/2006 4:11:52 PM) ...
.1548 16:11:52 (3) 1 static instance(s) found for '__EventConsumerProviderCacheControl' in 'Root' in 0 second(s).
.1549 16:11:52 (3) Retrieving static information (MOF) of '__EventProviderCacheControl' (I=1).
.1550 16:11:52 (3) Qualifier information of '__EventProviderCacheControl': Dynamic=False,
    Provider='<NOT DEFINED>', Association=False.
.
.
.

```

Verifying timer instructions

```

21547 16:14:49 (0) ** Verifying WMI namespace 'ROOT/CIMV2' (L=2).
21548 16:14:49 (3) Retrieving WMI system class(es) static information.
21549 16:14:49 (3) 45/45 system class(es) found.
21550 16:14:49 (3) Verifying Permanent subscription(s) for 'ROOT/CIMV2'.
21551 16:14:49 (3) 0 permanent subscription(s) in 'ROOT/CIMV2' namespace.
21552 16:14:49 (3) 1 Timer instruction(s) in 'ROOT/CIMV2' namespace.
21553 16:14:49 (4)
21554 16:14:49 (4) #1 'MyIntervalTimerEvent' Timer instruction.
21555 16:14:49 (4) - __IntervalTimerInstruction #1 -----
21556 16:14:49 (4) IntervalBetweenEvents: ..... 5000
21557 16:14:49 (4) SkipIfPassed: ..... FALSE
21558 16:14:49 (4) *TimerId: ..... MyIntervalTimerEvent
.
.
.

```

Verifying the namespace security

```
21559 16:14:49 (4)
21560 16:14:49 (3) Deciphering WMI namespace security for 'ROOT/CIMV2'
21561 16:14:50 (4) +- Security Descriptor -----
21562 16:14:50 (4) | Owner: ..... BUILTIN\ADMINISTRATORS
21563 16:14:50 (4) | Group: ..... BUILTIN\ADMINISTRATORS
21564 16:14:50 (4) | Revision: ..... 1
21565 16:14:50 (4) | Control: ..... &h8004
21566 16:14:50 (4) | SE_DACL_PRESENT
21567 16:14:50 (4) | SE_SELF_RELATIVE
21568 16:14:50 (4) +- DiscretionaryAcl -----
21569 16:14:50 (4) ||+- ACE #01 -----
21570 16:14:50 (4) ||| Trustee: ..... REDMOND\SMS_CLIENT_ADMIN
21571 16:14:50 (4) ||| AceType: ..... &h0
21572 16:14:50 (4) | ACCESS_ALLOWED_ACE_TYPE
21573 16:14:50 (4) ||| AceFlags: ..... &h2
21574 16:14:50 (4) | CONTAINER_INHERIT_ACE
21575 16:14:50 (4) ||| AccessMask: ..... &h20020
21576 16:14:50 (4) | WBEM_REMOTE_ACCESS
21577 16:14:50 (4) | WBEM_READ_CONTROL
21578 16:14:50 (4) ||+-----
21579 16:14:50 (4) ||+- ACE #02 -----
21580 16:14:50 (4) ||| Trustee: ..... BUILTIN\ADMINISTRATORS
21581 16:14:50 (4) ||| AceType: ..... &h0
21582 16:14:50 (4) | ACCESS_ALLOWED_ACE_TYPE
21583 16:14:50 (4) ||| AceFlags: ..... &h12
21584 16:14:50 (4) | CONTAINER_INHERIT_ACE
21585 16:14:50 (4) | INHERITED_ACE
21586 16:14:50 (4) ||| AccessMask: ..... &h6003F
21587 16:14:50 (4) | WBEM_ENABLE
21588 16:14:50 (4) | WBEM_METHOD_EXECUTE
21589 16:14:50 (4) | WBEM_FULL_WRITE_REP
21590 16:14:50 (4) | WBEM_PARTIAL_WRITE_REP
21591 16:14:50 (4) | WBEM_WRITE_PROVIDER
21592 16:14:50 (4) | WBEM_REMOTE_ACCESS
21593 16:14:50 (4) | WBEM_WRITE_DAC
21594 16:14:50 (4) | WBEM_READ_CONTROL
21595 16:14:50 (4) ||+-----
21596 16:14:50 (4) ||+- ACE #03 -----
21597 16:14:50 (4) ||| Trustee: ..... NT AUTHORITY\NETWORK SERVICE
21598 16:14:50 (4) ||| AceType: ..... &h0
21599 16:14:50 (4) | ACCESS_ALLOWED_ACE_TYPE
21600 16:14:50 (4) ||| AceFlags: ..... &h12
21601 16:14:50 (4) | CONTAINER_INHERIT_ACE
21602 16:14:50 (4) | INHERITED_ACE
21603 16:14:50 (4) ||| AccessMask: ..... &h13
21604 16:14:50 (4) | WBEM_ENABLE
21605 16:14:50 (4) | WBEM_METHOD_EXECUTE
21606 16:14:50 (4) | WBEM_WRITE_PROVIDER
21607 16:14:50 (4) ||+-----
21608 16:14:50 (4) ||+- ACE #04 -----
21609 16:14:50 (4) ||| Trustee: ..... NT AUTHORITY\LOCAL SERVICE
21610 16:14:50 (4) ||| AceType: ..... &h0
21611 16:14:50 (4) | ACCESS_ALLOWED_ACE_TYPE
21612 16:14:50 (4) ||| AceFlags: ..... &h12
21613 16:14:50 (4) | CONTAINER_INHERIT_ACE
21614 16:14:50 (4) | INHERITED_ACE
21615 16:14:50 (4) ||| AccessMask: ..... &h13
21616 16:14:50 (4) | WBEM_ENABLE
21617 16:14:50 (4) | WBEM_METHOD_EXECUTE
21618 16:14:50 (4) | WBEM_WRITE_PROVIDER
21619 16:14:50 (4) ||+-----
21620 16:14:50 (4) ||+- ACE #05 -----
21621 16:14:50 (4) ||| Trustee: ..... EVERYONE
21622 16:14:50 (4) ||| AceType: ..... &h0
21623 16:14:50 (4) | ACCESS_ALLOWED_ACE_TYPE
21624 16:14:50 (4) ||| AceFlags: ..... &h12
21625 16:14:50 (4) | CONTAINER_INHERIT_ACE
21626 16:14:50 (4) | INHERITED_ACE
21627 16:14:50 (4) ||| AccessMask: ..... &h13
21628 16:14:50 (4) | WBEM_ENABLE
21629 16:14:50 (4) | WBEM_METHOD_EXECUTE
21630 16:14:50 (4) | WBEM_WRITE_PROVIDER
21631 16:14:50 (4) ||+-----
21632 16:14:50 (4) ||+-----
21633 16:14:50 (4) ||+-----
```

Verifying the provider CIM and DCOM registration

```
21641 16:14:50 (3) 57 WMI Provider CIM registration(s) found.
21642 16:14:50 (3) (#1) Found WMI Provider 'WIN32_WIN32_TSLOGONSETTING_PROV' CIM registration.
21643 16:14:50 (3) WMI Provider hosting model is 'NETWORKSERVICEHOST'.
21644 16:14:50 (3) WMI Provider 'WIN32_WIN32_TSLOGONSETTING_PROV' is an INSTANCE provider and supports
instance enumeration.
21645 16:14:50 (3) WMI Provider 'WIN32_WIN32_TSLOGONSETTING_PROV' is a METHOD provider.
21646 16:14:50 (3) Verifying if WMI Provider 'WIN32_WIN32_TSLOGONSETTING_PROV' has an In-Proc registration.
21647 16:14:50 (4) Reading registry (REG_EXPAND_SZ) 'HKCR\CLSID\
{C41FF872-07B1-4926-819B-8C94E6B1FBB9}\InProcServer32\''.
21648 16:14:50 (3) Found WMI Provider In-Proc registration 'WIN32_WIN32_TSLOGONSETTING_PROV'
({C41FF872-07B1-4926-819B-8C94E6B1FBB9}).
21649 16:14:50 (3) WMI Provider DLL 'C:\WINDOWS\SYSTEM32\TSCFGWMI.DLL' is available
on the system (WIN32_WIN32_TSLOGONSETTING_PROV).
21650 16:14:51 (3) MOF file(s) 'C:\WINDOWS\SYSTEM32\WBEM\TSCFGWMI.MOF' contain(s) registration data for WMI Provider
'WIN32_WIN32_TSLOGONSETTING_PROV' ({C41FF872-07B1-4926-819B-8C94E6B1FBB9})
of namespace 'ROOT/CIMV2'.
21651 16:14:51 (3) (#2) Found WMI Provider 'REGPROV' CIM registration.
21652 16:14:51 (2) !! WARNING: WMI Provider hosting model is undefined, which implies 'LocalSystemHostOrSelfHost'.
21653 16:14:51 (3) WMI Provider is 'REGPROV' IS a trusted WMI providers.
21654 16:14:51 (3) WMI Provider 'REGPROV' is an INSTANCE provider and supports instance enumeration.
21655 16:14:51 (3) Verifying if WMI Provider 'REGPROV' has an In-Proc registration.
21656 16:14:51 (4) Reading registry (REG_EXPAND_SZ) 'HKCR\CLSID\
{FE9AF5C0-D3B6-11CE-A5B6-00AA00680C3F}\InProcServer32\''.
21657 16:14:51 (3) Found WMI Provider In-Proc registration 'REGPROV' ({FE9AF5C0-D3B6-11CE-A5B6-00AA00680C3F}).
21658 16:14:51 (3) WMI Provider DLL 'C:\WINDOWS\SYSTEM32\WBEM\STDPROV.DLL' is available on the system (REGPROV).
21659 16:14:53 (3) MOF file(s) 'C:\WINDOWS\SYSTEM32\WBEM\REGEVENT.MOF' contain(s) registration data for WMI Provider
'REGPROV' ({FE9AF5C0-D3B6-11CE-A5B6-00AA00680C3F}) of namespace 'ROOT/CIMV2'.
21660 16:14:53 (3) (#3) Found WMI Provider 'MS_NT_EVENTLOG_PROVIDER' CIM registration.
21661 16:14:53 (3) WMI Provider hosting model is 'NETWORKSERVICEHOST'.
21662 16:14:53 (3) WMI Provider 'MS_NT_EVENTLOG_PROVIDER' is an INSTANCE provider and supports instance enumeration.
21663 16:14:53 (3) WMI Provider 'MS_NT_EVENTLOG_PROVIDER' is a METHOD provider.
21664 16:14:53 (3) Verifying if WMI Provider 'MS_NT_EVENTLOG_PROVIDER' has an In-Proc registration.
21665 16:14:53 (4) Reading registry (REG_EXPAND_SZ) 'HKCR\CLSID\
{FD4F53E0-65DC-11D1-AB64-00C04FD9159E}\InProcServer32\''.
21666 16:14:53 (3) Found WMI Provider In-Proc registration 'MS_NT_EVENTLOG_PROVIDER'
({FD4F53E0-65DC-11D1-AB64-00C04FD9159E}).
21667 16:14:53 (3) WMI Provider DLL 'C:\WINDOWS\SYSTEM32\WBEM\NTEVT.DLL' is available on the system
(MS_NT_EVENTLOG_PROVIDER).
21668 16:14:54 (3) MOF file(s) 'C:\WINDOWS\SYSTEM32\WBEM\NTEVT.MOF' contain(s) registration data for WMI Provider
'MS_NT_EVENTLOG_PROVIDER' ({FD4F53E0-65DC-11D1-AB64-00C04FD9159E}) of
namespace 'ROOT/CIMV2'.
21669 16:14:54 (3) (#4) Found WMI Provider 'WIN32_WIN32_TSENVIRONMENTSETTING_PROV' CIM registration.
21670 16:14:54 (3) WMI Provider hosting model is 'NETWORKSERVICEHOST'.
21671 16:14:54 (3) WMI Provider 'WIN32_WIN32_TSENVIRONMENTSETTING_PROV' is an INSTANCE provider and supports
instance enumeration.
21672 16:14:54 (3) WMI Provider 'WIN32_WIN32_TSENVIRONMENTSETTING_PROV' is a METHOD provider.
21673 16:14:54 (3) Verifying if WMI Provider 'WIN32_WIN32_TSENVIRONMENTSETTING_PROV' has an In-Proc registration.
21674 16:14:54 (4) Reading registry (REG_EXPAND_SZ) 'HKCR\CLSID\
{9A17DFD1-34FA-4D61-B9BB-3A1097E7FADF}\InProcServer32\''.
21675 16:14:54 (3) Found WMI Provider In-Proc registration 'WIN32_WIN32_TSENVIRONMENTSETTING_PROV'
({9A17DFD1-34FA-4D61-B9BB-3A1097E7FADF}).
21676 16:14:54 (3) WMI Provider DLL 'C:\WINDOWS\SYSTEM32\TSCFGWMI.DLL' is available on the system
(WIN32_WIN32_TSENVIRONMENTSETTING_PROV).
21677 16:14:56 (3) MOF file(s) 'C:\WINDOWS\SYSTEM32\WBEM\TSCFGWMI.MOF' contain(s) registration data for WMI Provider
'WIN32_WIN32_TSENVIRONMENTSETTING_PROV' ({9A17DFD1-34FA-4D61-B9BB-3A1097E7FADF})
of namespace 'ROOT/CIMV2'.
```

Verifying Permanent subscriptions

```
30466 22:49:16 (3) Verifying Permanent subscription(s) for 'ROOT/SUBSCRIPTION'.
30467 22:49:16 (3) 3 permanent subscription(s) in 'ROOT/SUBSCRIPTION' namespace.
30468 22:49:16 (4)
30469 22:49:16 (4) #1 'LogFileEventConsumer.Name="LogFileForSvc"' permanent subscription.
30470 22:49:16 (4) - LogFileEventConsumer #1 -----
30471 22:49:16 (4) CreatorSID: ..... 1,5,0,0,0 ...
```

```

30472 22:49:16 (4)      Filename: ..... C:\WMIServiceWatcher.LOG
30473 22:49:16 (4)      IsUnicode: ..... FALSE
30474 22:49:16 (4)      MaximumFileSize: ..... 1024
30475 22:49:16 (4)      *Name: ..... LOGFileForSvc
30476 22:49:16 (4)      Text: ..... Service %TargetInstance.DisplayName% is %TargetInstance.State%.
30477 22:49:16 (4)      - __FilterToConsumerBinding #1 -----
30478 22:49:16 (4)      *Consumer: .....
          \\PC-ALAINL-02\ROOT\subscription:LogFileEventConsumer.Name="LOGFileForSvc"
30479 22:49:16 (4)      CreatorSID: ..... 1,5,0,0,0,0 ...
30480 22:49:16 (4)      DeliverSynchronously: .. FALSE
30481 22:49:16 (4)      *Filter: .....
          \\PC-ALAINL-02\ROOT\subscription:__EventFilter.Name="FilterForWIN32_Services"
30482 22:49:16 (4)      MaintainSecurityContext: FALSE
30483 22:49:16 (4)      SlowDownProviders: ..... FALSE
30484 22:49:16 (4)      - __EventFilter #1 -----
30485 22:49:16 (4)      CreatorSID: ..... 1,5,0,0,0,0,0,5,21,0,0,0 ...
30486 22:49:16 (4)      *Name: ..... FilterForWIN32_Services
30487 22:49:16 (4)      Query: ..... SELECT * FROM __InstanceModificationEvent WITHIN 10
          Where TargetInstance ISA 'Win32_Service'
30488 22:49:16 (4)      QueryLanguage: ..... WQL
30489 22:49:16 (4)
30490 22:49:16 (4)      #2 'NTEventLogEventConsumer.Name="SCM Event Log Consumer"' permanent subscription.
30491 22:49:16 (4)      - NTEventLogEventConsumer #2 -----
30492 22:49:16 (4)      Category: ..... 0
30493 22:49:16 (4)      CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30494 22:49:16 (4)      EventID: ..... 0
30495 22:49:16 (4)      EventType: ..... 1
30496 22:49:16 (4)      InsertionStringTemplates:
30497 22:49:16 (4)      *Name: ..... SCM Event Log Consumer
30498 22:49:16 (4)      NameOfUserSIDProperty: ... sid
30499 22:49:16 (4)      NumberOfInsertionStrings: 0
30500 22:49:16 (4)      SourceName: ..... Service Control Manager
30501 22:49:16 (4)      - __FilterToConsumerBinding #1 -----
30502 22:49:16 (4)      *Consumer: ..... NTEventLogEventConsumer.Name="SCM Event Log Consumer"
30503 22:49:16 (4)      CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30504 22:49:16 (4)      DeliverSynchronously: .. FALSE
30505 22:49:16 (4)      *Filter: ..... __EventFilter.Name="SCM Event Log Filter"
30506 22:49:16 (4)      MaintainSecurityContext: FALSE
30507 22:49:16 (4)      SlowDownProviders: ..... FALSE
30508 22:49:16 (4)      - __EventFilter #1 -----
30509 22:49:16 (4)      CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30510 22:49:16 (4)      EventNamespace: ..... root\cimv2
30511 22:49:16 (4)      *Name: ..... SCM Event Log Filter
30512 22:49:16 (4)      Query: ..... Select * from MSFT_SCMEventLogEvent
30513 22:49:16 (4)      QueryLanguage: ..... WQL
30514 22:49:16 (4)
30515 22:49:16 (4)      #3 'MSFT_UCScenarioControl.Name="Microsoft WMI Updating Consumer Scenario Control"'
          permanent subscription.
30516 22:49:16 (4)      - MSFT_UCScenarioControl #3 -----
30517 22:49:16 (4)      CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30518 22:49:16 (4)      *Name: ..... Microsoft WMI Updating Consumer Scenario Control
30519 22:49:16 (4)      - __FilterToConsumerBinding #1 -----
30520 22:49:16 (4)      *Consumer: .....
          \\.\root\subscription:MSFT_UCScenarioControl.Name="Microsoft WMI Updating Consumer Scenario Control"
30521 22:49:16 (4)      CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30522 22:49:16 (4)      DeliverSynchronously: ... TRUE
30523 22:49:16 (4)      *Filter: .....
          \\.\root\subscription:__EventFilter.Name="Microsoft WMI Updating Consumer Scenario Control"
30524 22:49:16 (4)      MaintainSecurityContext: FALSE
30525 22:49:16 (4)      SlowDownProviders: ..... FALSE
30526 22:49:16 (4)      - __EventFilter #1 -----
30527 22:49:16 (4)      CreatorSID: ..... 1,1,0,0,0,0,0,5,18,0,0,0
30528 22:49:16 (4)      *Name: ..... Microsoft WMI Updating Consumer Scenario Control
30529 22:49:16 (4)      Query: ..... SELECT * FROM __InstanceOperationEvent WHERE TargetInstance
          ISA 'MSFT_UCScenario'
30530 22:49:16 (4)      QueryLanguage: ..... WQL
30531 22:49:16 (4)
30532 22:49:16 (3)      0 Timer instruction(s) in 'ROOT/SUBSCRIPTION' namespace..
.

```

Verifying ADAP status

```

31979 16:17:37 (0) ** Verifying WMI ADAP status.
31980 16:17:37 (3) ADAP last start time is: '21 July 2006 15:58:27:603000 (GMT+8)''
31981 16:17:37 (3) ADAP last stop time is: '21 July 2006 16:05:54:313000 (GMT+8)''
31982 16:17:37 (3) ADAP last message is: 'The WMI ADAP process has finished (4).'
```

Get Operations

```
31983 16:17:37 (0) ** Verifying WMI features.
31984 16:17:37 (3) Opening WMI namespace 'Root'.
31985 16:17:37 (3) No WMI ENUMERATIONS operations available for 'Root'.
31986 16:17:37 (3) No WMI EXECQUERY operations available for 'Root'.
31987 16:17:37 (3) WMI GET VALUE operations for 'Root'.
31988 16:17:37 (3) Retrieving instance '__ArbitratorConfiguration=@'.
31989 16:17:37 (3) Retrieving value 'OutstandingTasksPerUser' property value.
31990 16:17:37 (3) 'OutstandingTasksPerUser' property value is CORRECT. ('30')
31991 16:17:37 (3) Retrieving value 'OutstandingTasksTotal' property value.
31992 16:17:37 (3) 'OutstandingTasksTotal' property value is CORRECT. ('3000')
31993 16:17:37 (3) Retrieving value 'PermanentSubscriptionsPerUser' property value.
31994 16:17:37 (3) 'PermanentSubscriptionsPerUser' property value is CORRECT. ('1000')
31995 16:17:37 (3) Retrieving value 'PermanentSubscriptionsTotal' property value.
31996 16:17:37 (3) 'PermanentSubscriptionsTotal' property value is CORRECT. ('10000')
31997 16:17:37 (3) Retrieving value 'PollingInstructionsPerUser' property value.
31998 16:17:37 (3) 'PollingInstructionsPerUser' property value is CORRECT. ('1000')
31999 16:17:37 (3) Retrieving value 'PollingInstructionsTotal' property value.
32000 16:17:37 (3) 'PollingInstructionsTotal' property value is CORRECT. ('10000')
32001 16:17:37 (3) Retrieving value 'PollingMemoryPerUser' property value.
32002 16:17:37 (3) 'PollingMemoryPerUser' property value is CORRECT. ('5000000')
32003 16:17:37 (3) Retrieving value 'PollingMemoryTotal' property value.
32004 16:17:37 (3) 'PollingMemoryTotal' property value is CORRECT. ('10000000')
32005 16:17:37 (3) Retrieving value 'QuotaRetryCount' property value.
32006 16:17:37 (3) 'QuotaRetryCount' property value is CORRECT. ('10')
32007 16:17:37 (3) Retrieving value 'QuotaRetryWaitInterval' property value.
32008 16:17:37 (3) 'QuotaRetryWaitInterval' property value is CORRECT. ('15000')
32009 16:17:37 (3) Retrieving value 'TaskThreadsPerUser' property value.
32010 16:17:37 (3) 'TaskThreadsPerUser' property value is CORRECT. ('3')
32011 16:17:37 (3) Retrieving value 'TaskThreadsTotal' property value.
32012 16:17:37 (3) 'TaskThreadsTotal' property value is CORRECT. ('30')
32013 16:17:37 (3) Retrieving value 'TemporarySubscriptionsPerUser' property value.
32014 16:17:37 (3) 'TemporarySubscriptionsPerUser' property value is CORRECT. ('1000')
32015 16:17:37 (3) Retrieving value 'TemporarySubscriptionsTotal' property value.
32016 16:17:37 (3) 'TemporarySubscriptionsTotal' property value is CORRECT. ('10000')
32017 16:17:37 (3) Retrieving value 'TotalCacheDisk' property value.
32018 16:17:37 (3) 'TotalCacheDisk' property value is CORRECT. ('1048576')
32019 16:17:37 (3) Retrieving value 'TotalCacheDiskPerTask' property value.
32020 16:17:37 (3) 'TotalCacheDiskPerTask' property value is CORRECT. ('51250')
32021 16:17:37 (3) Retrieving value 'TotalCacheDiskPerUser' property value.
32022 16:17:37 (3) 'TotalCacheDiskPerUser' property value is CORRECT. ('102500')
32023 16:17:37 (3) Retrieving value 'TotalCacheMemory' property value.
32024 16:17:37 (3) 'TotalCacheMemory' property value is CORRECT. ('10240')
32025 16:17:37 (3) Retrieving value 'TotalCacheMemoryPerTask' property value.
32026 16:17:37 (3) 'TotalCacheMemoryPerTask' property value is CORRECT. ('1024')
32027 16:17:37 (3) Retrieving value 'TotalCacheMemoryPerUser' property value.
32028 16:17:37 (3) 'TotalCacheMemoryPerUser' property value is CORRECT. ('2048')
32029 16:17:37 (3) Retrieving value 'TotalUsers' property value.
32030 16:17:37 (3) 'TotalUsers' property value is CORRECT. ('50')
32031 16:17:37 (3) Retrieving instance '__ProviderHostQuotaConfiguration=@'.
```

WMI enumeration operations

```
32057 16:17:37 (3) Opening WMI namespace 'Root/Default'.
32058 16:17:37 (3) WMI GET operations for 'Root/Default'.
32059 16:17:37 (3) Retrieving 'SystemRestore' WMI Class static information.
32060 16:17:37 (3) WMI ENUMERATION operations for 'Root/Default'.
32061 16:17:37 (3) Retrieving instance(s) of WMI class 'SystemRestore'.
32062 16:17:37 (3) 13 instance(s) found.
32063 16:17:37 (3) Retrieving 'SystemRestoreConfig' WMI Class static information.
32064 16:17:37 (3) WMI ENUMERATION operations for 'Root/Default'.
32065 16:17:37 (3) Retrieving instance(s) of WMI class 'SystemRestoreConfig'.
32066 16:17:37 (3) 1 instance(s) found.
```

WMI ExecQuery operations

```
32197 16:17:39 (3) WMI EXECQUERY operations for 'Root/CIMv2'.
32198 16:17:39 (3) Executing WQL query 'SELECT * FROM Win32_LogicalDisk WHERE FreeSpace > 10000000
AND DriveType = 3'.
32199 16:17:39 (3) 2 instance(s) found, BUT this is expected.
32200 16:17:39 (3) Executing WQL query 'SELECT * FROM Win32_PageFileUsage'.
32201 16:17:39 (3) 1 instance(s) found, BUT this is expected.
32202 16:17:39 (3) Executing WQL query 'SELECT * FROM Win32_BIOS WHERE Version IS NOT NULL'.
32203 16:17:39 (3) 1 instance(s) found, BUT this is expected.
32204 16:17:39 (3) Executing WQL query 'SELECT * FROM Win32_NetworkAdapter WHERE AdapterType IS NOT NULL
AND AdapterType != "Wide Area Network (WAN)" AND Description != "Packet Scheduler Miniport" '.
32205 16:17:40 (3) 5 instance(s) found, BUT this is expected.
32206 16:17:40 (3) Executing WQL query 'SELECT * FROM Win32_Processor WHERE Name IS NOT NULL'.
32207 16:17:42 (3) 2 instance(s) found, BUT this is expected.
32208 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_DiskDrive'.
32209 16:17:42 (3) 2 instance(s) found, BUT this is expected.
32210 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_ComputerSystem'.
32211 16:17:42 (3) 1 instance(s) found, BUT this is expected.
32212 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_DiskPartition'.
32213 16:17:42 (3) 2 instance(s) found, BUT this is expected.
32214 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_LogicalDisk WHERE Description != "Network Connection"'.
32215 16:17:42 (3) 4 instance(s) found, BUT this is expected.
32216 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_SoundDevice'.
32217 16:17:42 (3) 1 instance(s) found, BUT this is expected.
32218 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_VideoController'.
32219 16:17:42 (3) 1 instance(s) found, BUT this is expected.
32220 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_USBController'.
32221 16:17:42 (3) 5 instance(s) found, BUT this is expected.
32222 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_DesktopMonitor'.
32223 16:17:42 (3) 1 instance(s) found, BUT this is expected.
32224 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_PointingDevice WHERE Status = "OK"'.
32225 16:17:42 (3) 1 instance(s) found, BUT this is expected.
32226 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_Keyboard'.
32227 16:17:42 (3) 1 instance(s) found, BUT this is expected.
32228 16:17:42 (3) Executing WQL query 'SELECT * FROM Win32_SystemDriver WHERE StartMode != "Disabled"'.
32229 16:17:43 (3) 189 instance(s) found, BUT this is expected.
.
.
.
```

Verifying SMS WMI typical operations

```
32237 16:17:43 (0) ** Verifying SMS Agent v2.50.4160 WMI features.
32238 16:17:43 (3) Opening WMI namespace 'root/ccm/Messaging'.
32239 16:17:43 (3) WMI GET operations for 'root/ccm/Messaging'.
32240 16:17:43 (3) Retrieving 'CCM_Message_TransferStatus' WMI Class static information.
32241 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/Messaging' for
WMI class 'CCM_Message_TransferStatus'.
32242 16:17:43 (3) No WMI EXECQUERY operations available for 'root/ccm/Messaging'.
32243 16:17:43 (3) No WMI GET VALUE operations available for 'root/ccm/Messaging'.
32244 16:17:43 (3) Opening WMI namespace 'root/ccm/ContentTransferManager'.
32245 16:17:43 (3) WMI GET operations for 'root/ccm/ContentTransferManager'.
32246 16:17:43 (3) Retrieving 'CCM_CTM_JobStateEx' WMI Class static information.
32247 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/ContentTransferManager'
for WMI class 'CCM_CTM_JobStateEx'.
32248 16:17:43 (3) Retrieving 'CCM_CTM_ContentLocation' WMI Class static information.
32249 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/ContentTransferManager'
for WMI class 'CCM_CTM_ContentLocation'.
32250 16:17:43 (3) No WMI EXECQUERY operations available for 'root/ccm/ContentTransferManager'.
32251 16:17:43 (3) No WMI GET VALUE operations available for 'root/ccm/ContentTransferManager'.
32252 16:17:43 (3) Opening WMI namespace 'root/ccm/DataTransferService'.
32253 16:17:43 (3) WMI GET operations for 'root/ccm/DataTransferService'.
32254 16:17:43 (3) Retrieving 'CCM_DTS_JobEx' WMI Class static information.
32255 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/DataTransferService'
for WMI class 'CCM_DTS_JobEx'.
32256 16:17:43 (3) Retrieving 'CCM_DTS_JobItem' WMI Class static information.
32257 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/DataTransferService'
for WMI class 'CCM_DTS_JobItem'.
32258 16:17:43 (3) Retrieving 'CCM_FileBITS_Job' WMI Class static information.
32259 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/DataTransferService'
for WMI class 'CCM_FileBITS_Job'.
32260 16:17:43 (3) Retrieving 'CCM_FileBITS_JobItem' WMI Class static information.
32261 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/DataTransferService'
for WMI class 'CCM_FileBITS_JobItem'.
```

```

32262 16:17:43 (3) No WMI EXECQUERY operations available for 'root/ccm/DataTransferService'.
32263 16:17:43 (3) No WMI GET VALUE operations available for 'root/ccm/DataTransferService'.
32264 16:17:43 (3) Opening WMI namespace 'root/ccm/LocationServices'.
32265 16:17:43 (3) WMI GET operations for 'root/ccm/LocationServices'.
32266 16:17:43 (3) Retrieving 'LocationRequest' WMI Class static information.
32267 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/LocationServices'
for WMI class 'LocationRequest'.
32268 16:17:43 (3) Retrieving 'SMS_MPLList' WMI Class static information.
32269 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/LocationServices'
for WMI class 'SMS_MPLList'.
32270 16:17:43 (3) Retrieving 'SMS_MPInformation' WMI Class static information.
32271 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/LocationServices'
for WMI class 'SMS_MPInformation'.
32272 16:17:43 (3) Retrieving 'TrustedRootKey' WMI Class static information.
32273 16:17:43 (3) No WMI ENUMERATION operations requested from 'root/ccm/LocationServices'
for WMI class 'TrustedRootKey'.
32274 16:17:43 (3) WMI EXECQUERY operations for 'root/ccm/LocationServices'.
32275 16:17:43 (3) Executing WQL query 'Select * From TrustedRootKey'.
32276 16:17:43 (3) 1 instance(s) found, BUT this is expected.
32277 16:17:43 (3) No WMI GET VALUE operations available for 'root/ccm/LocationServices'.
32278 16:17:43 (3) Opening WMI namespace 'root/ccm/SoftMgmtAgent'.
32279 16:17:43 (3) WMI GET operations for 'root/ccm/SoftMgmtAgent'.
32280 16:17:43 (3) Retrieving 'CCM_RejectedPolicy' WMI Class static information.
.
.
.

```

Collecting system information about the disk sub-system

```

33400 16:17:47 (0) ** Collecting system information.
33401 16:17:47 (0) ** - Disk information
33402 16:17:48 (3) - Win32_DiskDrive #1 -----
33403 16:17:48 (3) BytesPerSector: ..... 512
33404 16:17:48 (3) Capabilities: ..... 3
33405 16:17:48 (3) ..... 4
33406 16:17:48 (3) Caption: ..... IC25N080ATMR04-0
33407 16:17:48 (3) ConfigManagerErrorCode: ..... 0
33408 16:17:48 (3) ConfigManagerUserConfig: ..... FALSE
33409 16:17:48 (3) CreationClassName: ..... Win32_DiskDrive
33410 16:17:48 (3) Description: ..... Disk drive
33411 16:17:48 (3) *DeviceID: ..... \\.\PHYSICALDRIVE0
33412 16:17:48 (3) Index: ..... 0
33413 16:17:48 (3) InterfaceType: ..... IDE
33414 16:17:48 (3) Manufacturer: ..... (Standard disk drives)
33415 16:17:48 (3) MediaLoaded: ..... TRUE
33416 16:17:48 (3) MediaType: ..... Fixed hard disk media
33417 16:17:48 (3) Model: ..... IC25N080ATMR04-0
33418 16:17:48 (3) Name: ..... \\.\PHYSICALDRIVE0
33419 16:17:48 (3) Partitions: ..... 2
33420 16:17:48 (3) PNPDeviceID: ..... IDE\DISKIC25N080ATMR04-
0_____MO40AD5A\5&A63E6F1&0&0.0.0
33421 16:17:48 (3) SCSIbus: ..... 0
33422 16:17:48 (3) SCSILogicalUnit: ..... 0
33423 16:17:48 (3) SCSIPort: ..... 0
33424 16:17:48 (3) SCISITargetId: ..... 0
33425 16:17:48 (3) SectorsPerTrack: ..... 63
33426 16:17:48 (3) Signature: ..... 70255663
33427 16:17:48 (3) Size: ..... 80023749120
33428 16:17:48 (3) Status: ..... OK
33429 16:17:48 (3) SystemCreationClassName: ..... Win32_ComputerSystem
33430 16:17:48 (3) SystemName: ..... PC-ALAINL-02
33431 16:17:48 (3) TotalCylinders: ..... 9729
33432 16:17:48 (3) TotalHeads: ..... 255
33433 16:17:48 (3) TotalSectors: ..... 156296385
33434 16:17:48 (3) TotalTracks: ..... 2480895
33435 16:17:48 (3) TracksPerCylinder: ..... 255
33436 16:17:48 (3) - Win32_DiskPartition #1 -----
33437 16:17:48 (3) BlockSize: ..... 512
33438 16:17:48 (3) Bootable: ..... TRUE
33439 16:17:48 (3) BootPartition: ..... TRUE
33440 16:17:48 (3) Caption: ..... Disk #0, Partition #0
33441 16:17:48 (3) CreationClassName: ..... Win32_DiskPartition
33442 16:17:48 (3) Description: ..... Installable File System
33443 16:17:48 (3) *DeviceID: ..... Disk #0, Partition #0
33444 16:17:48 (3) DiskIndex: ..... 0
33445 16:17:48 (3) Index: ..... 0
33446 16:17:48 (3) Name: ..... Disk #0, Partition #0
33447 16:17:48 (3) NumberOfBlocks: ..... 20980827

```

```

33448 16:17:48 (3) PrimaryPartition: ..... TRUE
33449 16:17:48 (3) Size: ..... 10742183424
33450 16:17:48 (3) StartingOffset: ..... 32256
33451 16:17:48 (3) SystemCreationClassName: ..... Win32_ComputerSystem
33452 16:17:48 (3) SystemName: ..... PC-ALAINL-02
33453 16:17:48 (3) Type: ..... Installable File System
33454 16:17:48 (3) - Win32_LogicalDisk #1 -----
33455 16:17:48 (3) Caption: ..... C:
33456 16:17:48 (3) Compressed: ..... FALSE
33457 16:17:48 (3) CreationClassName: ..... Win32_LogicalDisk
33458 16:17:48 (3) Description: ..... Local Fixed Disk
33459 16:17:48 (3) *DeviceID: ..... C:
33460 16:17:49 (3) DriveType: ..... 3
33461 16:17:49 (3) FileSystem: ..... NTFS
33462 16:17:49 (3) FreeSpace: ..... 962686976
33463 16:17:49 (3) MaximumComponentLength: ..... 255
33464 16:17:49 (3) MediaType: ..... 12
33465 16:17:49 (3) Name: ..... C:
33466 16:17:49 (3) QuotasDisabled: ..... TRUE
33467 16:17:49 (3) QuotasIncomplete: ..... TRUE
33468 16:17:49 (3) QuotasRebuilding: ..... FALSE
33469 16:17:49 (3) Size: ..... 10742181888
33470 16:17:49 (3) SupportsDiskQuotas: ..... TRUE
33471 16:17:49 (3) SupportsFileBasedCompression: ..... TRUE
33472 16:17:49 (3) SystemCreationClassName: ..... Win32_ComputerSystem
33473 16:17:49 (3) SystemName: ..... PC-ALAINL-02
33474 16:17:49 (3) VolumeDirty: ..... FALSE
33475 16:17:49 (3) VolumeName: ..... System
33476 16:17:49 (3) VolumeSerialNumber: ..... 51D076D8
33477 16:17:49 (3) - Win32_DiskPartition #2 -----
33478 16:17:49 (3) BlockSize: ..... 512
33479 16:17:49 (3) BootPartition: ..... FALSE
33480 16:17:49 (3) Caption: ..... Disk #0, Partition #1
33481 16:17:49 (3) CreationClassName: ..... Win32_DiskPartition
33482 16:17:49 (3) Description: ..... Extended w/Extended Int 13
33483 16:17:49 (3) *DeviceID: ..... Disk #0, Partition #1
33484 16:17:49 (3) DiskIndex: ..... 0
33485 16:17:49 (3) Index: ..... 1
33486 16:17:49 (3) Name: ..... Disk #0, Partition #1
33487 16:17:49 (3) NumberOfBlocks: ..... 135315495
33488 16:17:49 (3) PrimaryPartition: ..... FALSE
33489 16:17:49 (3) Size: ..... 69281533440
33490 16:17:49 (3) StartingOffset: ..... 10742215680
33491 16:17:49 (3) SystemCreationClassName: ..... Win32_ComputerSystem
33492 16:17:49 (3) SystemName: ..... PC-ALAINL-02
33493 16:17:49 (3) Type: ..... Extended w/Extended Int 13
33494 16:17:49 (3) - Win32_LogicalDisk #1 -----
33495 16:17:49 (3) Caption: ..... D:
33496 16:17:49 (3) Compressed: ..... FALSE
33497 16:17:49 (3) CreationClassName: ..... Win32_LogicalDisk
33498 16:17:49 (3) Description: ..... Local Fixed Disk
33499 16:17:49 (3) *DeviceID: ..... D:
33500 16:17:49 (3) DriveType: ..... 3
33501 16:17:49 (3) FileSystem: ..... NTFS
33502 16:17:49 (3) FreeSpace: ..... 32950349824
33503 16:17:49 (3) MaximumComponentLength: ..... 255
33504 16:17:49 (3) MediaType: ..... 12
33505 16:17:49 (3) Name: ..... D:
33506 16:17:49 (3) QuotasDisabled: ..... TRUE
33507 16:17:49 (3) QuotasIncomplete: ..... FALSE
33508 16:17:49 (3) QuotasRebuilding: ..... FALSE
33509 16:17:49 (3) Size: ..... 69281497088
33510 16:17:49 (3) SupportsDiskQuotas: ..... TRUE
33511 16:17:49 (3) SupportsFileBasedCompression: ..... TRUE
33512 16:17:49 (3) SystemCreationClassName: ..... Win32_ComputerSystem
33513 16:17:49 (3) SystemName: ..... PC-ALAINL-02
33514 16:17:49 (3) VolumeDirty: ..... FALSE
33515 16:17:49 (3) VolumeName: ..... Data
33516 16:17:49 (3) VolumeSerialNumber: ..... 2CC5E184
.
.
.

```

Collecting system information about the network adapters

```

33538 16:17:49 (0) ** - Network information
33539 16:17:50 (3) - Win32_NetworkAdapter #1 -----
33540 16:17:50 (3) Adapter name: ..... Realtek RTL8139/810x Family Fast Ethernet NIC

```

```

33541 16:17:50 (3) Device availability: ..... 3
33542 16:17:50 (3) Adapter type: ..... Ethernet 802.3
33543 16:17:50 (3) Adapter state: ..... 7
33544 16:17:50 (3) MAC address is : ..... 00:0F:B0:0F:83:78
33545 16:17:50 (3) Adapter service name: ..... RTL8023
33546 16:17:50 (3) Last reset: ..... 20060928032340.084059-420
33547 16:17:50 (3) DHCP enabled: ..... TRUE
33548 16:17:50 (3) DHCP expires: ..... 20061006134638.000000-420
33549 16:17:50 (3) DHCP obtained: ..... 20060928134638.000000-420
33550 16:17:50 (3) DHCP server: ..... 157.56.114.20
33551 16:17:50 (3) IP address(es): ..... 0.0.0.0
33552 16:17:50 (3) IP connection metric: ..... 1
33553 16:17:50 (3) DNS registration enabled: ..... FALSE
33554 16:17:50 (3) DNS FULL registration enabled: ..... TRUE
33555 16:17:50 (3) DNS enabled for WINS resolution: ..... FALSE
33556 16:17:50 (3) Primary WINS Server: ..... 157.54.14.163
33557 16:17:50 (3) Secondary WINS Server: ..... 157.54.14.154
33558 16:17:50 (3) WINS scope ID: .....
33559 16:17:50 (3) Enable LMHOSTS lookup: ..... TRUE
33560 16:17:50 (3) NETBIOS over TCP/IP: ..... by DHCP
33561 16:17:50 (3) - Win32_NetworkAdapter #2 -----
33562 16:17:50 (3) Adapter name: ..... Packet Scheduler Miniport
33563 16:17:50 (3) Device availability: ..... 3
33564 16:17:50 (3) Adapter type: ..... Ethernet 802.3
33565 16:17:50 (3) MAC address is : ..... 00:0F:B0:0F:83:78
33566 16:17:50 (3) Adapter service name: ..... PSched
33567 16:17:50 (3) Last reset: ..... 20060928032340.084059-420
33568 16:17:50 (3) IP status: ..... DISABLED
33569 16:17:50 (3) - Win32_NetworkAdapter #3 -----
33570 16:17:50 (3) Adapter name: ..... HP WLAN 802.11a/b/g W500
33571 16:17:50 (3) Device availability: ..... 3
33572 16:17:50 (3) Adapter type: ..... Ethernet 802.3
33573 16:17:50 (3) Adapter state: ..... 9
33574 16:17:50 (3) MAC address is : ..... 00:11:85:1B:48:EA
33575 16:17:50 (3) Adapter service name: ..... WLAN_400_500_SERVICE
33576 16:17:50 (3) Last reset: ..... 20060928032340.084059-420
33577 16:17:50 (3) DHCP enabled: ..... TRUE
33578 16:17:50 (3) DHCP expires: ..... 20060928180529.000000-420
33579 16:17:50 (3) DHCP obtained: ..... 20060928150529.000000-420
33580 16:17:50 (3) DHCP server: ..... 157.54.54.6
33581 16:17:50 (3) IP address(es): ..... 172.28.18.92 255.255.252.0
33582 16:17:50 (3) IP connection metric: ..... 30
33583 16:17:50 (3) Default Gateway(s) and metric: ..... 172.28.16.1 30
33584 16:17:50 (3) DNS registration enabled: ..... FALSE
33585 16:17:50 (3) DNS FULL registration enabled: ..... TRUE
33586 16:17:50 (3) DNS search order: ..... 157.54.14.178
33587 16:17:50 (3) ..... 157.54.14.146
33588 16:17:50 (3) ..... 157.54.14.162
33589 16:17:50 (3) DNS domain: ..... redmond.corp.microsoft.com
33590 16:17:50 (3) DNS enabled for WINS resolution: ..... FALSE
33591 16:17:50 (3) Primary WINS Server: ..... 157.54.14.163
33592 16:17:50 (3) Secondary WINS Server: ..... 157.54.14.154
33593 16:17:50 (3) WINS scope ID: .....
33594 16:17:50 (3) Enable LMHOSTS lookup: ..... TRUE
33595 16:17:50 (3) NETBIOS over TCP/IP: ..... by DHCP
.
.
.

```

Collecting system information about DMA

```

33710 16:17:50 (0) ** - DMA resource usage
33711 16:17:51 (3) Channel 4
33712 16:17:51 (3) Direct memory access controller.
33713 16:17:51 (3) Channel 2
33714 16:17:51 (3) Standard floppy disk controller.
33715 16:17:51 (3) Service name is 'FDC' and status is OK.
33716 16:17:51 (3) Address range is 0x000003F0-0x000003F5
33717 16:17:51 (3) Address range is 0x000003F7-0x000003F7
33718 16:17:51 (3) Channel 1
33719 16:17:51 (3) ECP Printer Port (LPT1).
33720 16:17:51 (3) Service name is 'PARPORT' and status is OK.
33721 16:17:51 (3) Address range is 0x00000378-0x0000037F
33722 16:17:51 (3) Address range is 0x00000778-0x0000077B
.
.
.

```

Collecting system information about IRQ

```
33713 02:30:59 (0) ** - IRQ resource usage
33714 02:30:59 (3)   IRQ 21
33715 02:30:59 (3)     Microsoft ACPI-Compliant System.
33716 02:30:59 (3)     Service name is 'ACPI' and status is OK.
33717 02:31:00 (3)   IRQ 16
33718 02:31:00 (3)     ATI MOBILITY RADEON 9000/9100 IGP.
33719 02:31:00 (3)     Service name is 'ATI2MTAG' and status is OK.
33720 02:31:00 (3)     Address range is 0x00009000-0x00009FFF
33721 02:31:00 (3)     Address range is 0x000003B0-0x000003BB
33722 02:31:00 (3)     Address range is 0x000003C0-0x000003DF
33723 02:31:00 (3)     Texas Instruments OHCI Compliant IEEE 1394 Host Controller.
33724 02:31:00 (3)     Service name is 'OHCI1394' and status is OK.
33725 02:31:00 (3)     Texas Instruments PCI-1620 CardBus Controller with UltraMedia..
33726 02:31:00 (3)     Service name is 'PCMCIA' and status is OK.
33727 02:31:00 (3)     Address range is 0x0000FA00-0x0000FAFF
33728 02:31:00 (3)     Address range is 0x0000F900-0x0000F9FF
33729 02:31:00 (3)     NEC PCI to USB Open Host Controller.
33730 02:31:00 (3)     Service name is 'USBOHCI' and status is OK.
33731 02:31:01 (3)   IRQ 19
33732 02:31:01 (3)     Standard OpenHCD USB Host Controller.
33733 02:31:01 (3)     Service name is 'USBOHCI' and status is OK.
33734 02:31:01 (3)     Standard OpenHCD USB Host Controller.
33735 02:31:01 (3)     Service name is 'USBOHCI' and status is OK.
33736 02:31:01 (3)     Realtek RTL8139/810x Family Fast Ethernet NIC.
33737 02:31:01 (3)     Service name is 'RTL8023' and status is OK.
33738 02:31:01 (3)     Address range is 0x0000A000-0x0000A0FF
33739 02:31:01 (3)     Standard Enhanced PCI to USB Host Controller.
33740 02:31:01 (3)     Service name is 'USBHCI' and status is OK.
33741 02:31:01 (3)   IRQ 14
33742 02:31:01 (3)     Primary IDE Channel.
33743 02:31:01 (3)     Service name is 'ATAPI' and status is OK.
33744 02:31:01 (3)     Address range is 0x000001F0-0x000001F7
33745 02:31:01 (3)     Address range is 0x000003F6-0x000003F6
33746 02:31:01 (3)   IRQ 15
33747 02:31:01 (3)     Secondary IDE Channel.
33748 02:31:01 (3)     Service name is 'ATAPI' and status is OK.
33749 02:31:01 (3)     Address range is 0x00000170-0x00000177
33750 02:31:02 (3)     Address range is 0x00000376-0x00000376
33751 02:31:02 (3)   IRQ 13
33752 02:31:02 (3)     Numeric data processor.
33753 02:31:02 (3)   IRQ 8
33754 02:31:02 (3)     System CMOS/real time clock.
33755 02:31:02 (3)   IRQ 0
33756 02:31:02 (3)     System timer.
33757 02:31:02 (3)   IRQ 1
33758 02:31:02 (3)     Standard 101/102-Key or Microsoft Natural PS/2 Keyboard.
33759 02:31:02 (3)     Service name is 'I8042PRT' and status is OK.
33760 02:31:03 (3)     Address range is 0x00000060-0x00000060
33761 02:31:03 (3)     Address range is 0x00000064-0x00000064
33762 02:31:03 (3)   IRQ 12
33763 02:31:03 (3)     Alps Pointing-device.
33764 02:31:03 (3)     Service name is 'I8042PRT' and status is OK.
33765 02:31:03 (3)   IRQ 6
33766 02:31:03 (3)     Standard floppy disk controller.
33767 02:31:03 (3)     Service name is 'FDC' and status is OK.
33768 02:31:03 (3)     Address range is 0x000003F0-0x000003F5
33769 02:31:03 (3)     Address range is 0x000003F7-0x000003F7
33770 02:31:04 (3)   IRQ 18
33771 02:31:04 (3)     HP WLAN 802.11a/b/g W500.
33772 02:31:04 (3)     Service name is 'WLAN_400_500_SERVICE' and status is OK.
33773 02:31:04 (3)     NEC PCI to USB Open Host Controller.
33774 02:31:04 (3)     Service name is 'USBOHCI' and status is OK.
33775 02:31:04 (3)   IRQ 17
33776 02:31:04 (3)     Texas Instruments PCI-1620 CardBus Controller with UltraMedia..
33777 02:31:04 (3)     Service name is 'PCMCIA' and status is OK.
33778 02:31:04 (3)     Address range is 0x0000FD00-0x0000FDFF
33779 02:31:04 (3)     Address range is 0x0000FC00-0x0000FCFF
33780 02:31:04 (3)   IRQ 3
33781 02:31:04 (3)     Gemplus GemPC400 PCMCIA Smart Card Reader.
33782 02:31:04 (3)     Service name is 'GEMPPC' and status is OK.
33783 02:31:04 (3)     Address range is 0x0000FFE0-0x0000FFFF
33784 02:31:05 (3)   IRQ 5
33785 02:31:05 (3)     SoundMAX Integrated Digital Audio.
33786 02:31:05 (3)     Service name is 'SMWDM' and status is OK.
33787 02:31:05 (3)     Agere Systems AC'97 Modem.
33788 02:31:05 (3)     Service name is 'MODEM' and status is OK.
33789 02:31:05 (3)
```

Collecting system information about memory

```
.
.
.
33723 16:17:51 (3)
33724 16:17:51 (0) ** - Memory information
33725 16:17:51 (3)   'Bank 0' is 1024 MB chip size (U5).
33726 16:17:51 (3)   'Bank 0' is 1024 MB chip size (U6).
33727 16:17:51 (3)
.
.
.
```

Collecting system information about processor

```
33728 16:17:51 (0) ** - Processor information
33729 16:17:53 (3)   - Win32_Processor #1 -----
33730 16:17:53 (3)   AddressWidth: ..... 32
33731 16:17:53 (3)   Architecture: ..... 0
33732 16:17:53 (3)   Availability: ..... 3
33733 16:17:53 (3)   Caption: ..... x86 Family 15 Model 2 Stepping 9
33734 16:17:53 (3)   CpuStatus: ..... 1
33735 16:17:53 (3)   CreationClassName: ..... Win32_Processor
33736 16:17:53 (3)   CurrentClockSpeed: ..... 3200
33737 16:17:53 (3)   CurrentVoltage: ..... 22
33738 16:17:53 (3)   DataWidth: ..... 32
33739 16:17:53 (3)   Description: ..... x86 Family 15 Model 2 Stepping 9
33740 16:17:53 (3)   *DeviceID: ..... CPU0
33741 16:17:53 (3)   Family: ..... 2
33742 16:17:53 (3)   L2CacheSize: ..... 0
33743 16:17:53 (3)   Level: ..... 15
33744 16:17:53 (3)   LoadPercentage: ..... 8
33745 16:17:53 (3)   Manufacturer: ..... GenuineIntel
33746 16:17:53 (3)   MaxClockSpeed: ..... 3200
33747 16:17:53 (3)   Name: ..... Intel(R) Pentium(R) 4 CPU 3.20GHz
33748 16:17:53 (3)   PowerManagementSupported: ..... FALSE
33749 16:17:53 (3)   ProcessorID: ..... BFEBFBFF00000F29
33750 16:17:53 (3)   ProcessorType: ..... 3
33751 16:17:53 (3)   Revision: ..... 521
33752 16:17:53 (3)   Role: ..... CPU
33753 16:17:53 (3)   SocketDesignation: ..... JP8
33754 16:17:54 (3)   Status: ..... OK
33755 16:17:54 (3)   StatusInfo: ..... 3
33756 16:17:54 (3)   Stepping: ..... 9
33757 16:17:54 (3)   SystemCreationClassName: ..... Win32_ComputerSystem
33758 16:17:54 (3)   SystemName: ..... PC-ALAINL-02
33759 16:17:54 (3)   UpgradeMethod: ..... 4
33760 16:17:54 (3)   Version: ..... Model 2, Stepping 9
.
.
.
```

Collecting system information about the Operating System

```
33793 16:17:54 (3)
33794 16:17:54 (0) ** - Operating System information
33795 16:17:54 (3)   AdminPasswordStatus: ..... 1
33796 16:17:54 (3)   AutomaticResetBootOption: ..... TRUE
33797 16:17:54 (3)   AutomaticResetCapability: ..... TRUE
33798 16:17:54 (3)   BootOptionOnLimit: ..... 3
33799 16:17:54 (3)   BootOptionOnWatchDog: ..... 3
33800 16:17:54 (3)   BootROMSupported: ..... TRUE
33801 16:17:54 (3)   BootupState: ..... Normal boot
33802 16:17:54 (3)   Caption: ..... PC-ALAINL-02
33803 16:17:54 (3)   ChassisBootupState: ..... 3
33804 16:17:54 (3)   CreationClassName: ..... Win32_ComputerSystem
33805 16:17:54 (3)   CurrentTimeZone: ..... -420
33806 16:17:54 (3)   DaylightInEffect: ..... TRUE
33807 16:17:54 (3)   Description: ..... AT/AT COMPATIBLE
33808 16:17:54 (3)   Domain: ..... redmond.corp.microsoft.com
33809 16:17:54 (3)   DomainRole: ..... 1
33810 16:17:54 (3)   EnableDaylightSavingsTime: ..... TRUE
33811 16:17:54 (3)   FrontPanelResetStatus: ..... 3
```

```

33812 16:17:54 (3) InfraredSupported: . . . . . FALSE
33813 16:17:54 (3) KeyboardPasswordStatus: . . . . . 3
33814 16:17:54 (3) Manufacturer: . . . . . Hewlett-Packard
33815 16:17:54 (3) Model: . . . . . Compaq nx9110 (PL612UA#ABA)
33816 16:17:54 (3) *Name: . . . . . PC-ALAINL-02
33817 16:17:54 (3) NetworkServerModeEnabled: . . . . . TRUE
33818 16:17:54 (3) NumberOfProcessors: . . . . . 2
33819 16:17:54 (3) OEMStringArray: . . . . . Warranty:
33820 16:17:54 (3) PartOfDomain: . . . . . TRUE
33821 16:17:54 (3) PauseAfterReset: . . . . . 3932100000
33822 16:17:54 (3) PowerOnPasswordStatus: . . . . . 0
33823 16:17:54 (3) PowerState: . . . . . 0
33824 16:17:54 (3) PowerSupplyState: . . . . . 3
33825 16:17:54 (3) PrimaryOwnerName: . . . . . AlainL
33826 16:17:54 (3) ResetCapability: . . . . . 1
33827 16:17:54 (3) ResetCount: . . . . . -1
33828 16:17:54 (3) ResetLimit: . . . . . -1
33829 16:17:54 (3) Roles: . . . . . LM_Workstation
33830 16:17:54 (3) . . . . . LM_Server
33831 16:17:54 (3) . . . . . NT
33832 16:17:54 (3) Status: . . . . . OK
33833 16:17:54 (3) SupportContactDescription: . . . . . Web Site: http://www.LissWare.Net
33834 16:17:54 (3) SystemStartupDelay: . . . . . 30
33835 16:17:54 (3) SystemStartupOptions: . . . . . "Microsoft Windows XP Professional"
    /fastdetect /NoExecute=OptIn
33836 16:17:54 (3) SystemStartupSetting: . . . . . 0
33837 16:17:54 (3) SystemType: . . . . . X86-based PC
33838 16:17:54 (3) ThermalState: . . . . . 3
33839 16:17:54 (3) TotalPhysicalMemory: . . . . . 2012196864
33840 16:17:54 (3) UserName: . . . . . REDMOND\alainl
33841 16:17:54 (3) WakeUpType: . . . . . 6
33842 16:17:54 (3)
33843 16:17:54 (3) - Win32_OperatingSystem #1 -----
33844 16:17:54 (3) BootDevice: . . . . . \Device\HarddiskVolumel
33845 16:17:54 (3) BuildNumber: . . . . . 2600
33846 16:17:54 (3) BuildType: . . . . . Multiprocessor Free
33847 16:17:54 (3) Caption: . . . . . Microsoft Windows XP Professional
33848 16:17:54 (3) CodeSet: . . . . . 1252
33849 16:17:54 (3) CountryCode: . . . . . 1
33850 16:17:54 (3) CreationClassName: . . . . . Win32_OperatingSystem
33851 16:17:54 (3) CSCreationClassName: . . . . . Win32_ComputerSystem
33852 16:17:54 (3) CSDVersion: . . . . . Service Pack 2
33853 16:17:54 (3) CSName: . . . . . PC-ALAINL-02
33854 16:17:54 (3) CurrentTimeZone: . . . . . -420
33855 16:17:54 (3) DataExecutionPrevention_Available: . . . . . FALSE
33856 16:17:54 (3) DataExecutionPrevention_Drivers: . . . . . FALSE
33857 16:17:54 (3) DataExecutionPrevention_SupportPolicy: . . . . . 2
33858 16:17:54 (3) Debug: . . . . . FALSE
33859 16:17:54 (3) Description: . . . . . My Computer
33860 16:17:54 (3) Distributed: . . . . . FALSE
33861 16:17:54 (3) EncryptionLevel: . . . . . 168
33862 16:17:54 (3) ForegroundApplicationBoost: . . . . . 2
33863 16:17:54 (3) FreePhysicalMemory: . . . . . 1113692
33864 16:17:54 (3) FreeSpaceInPagingFiles: . . . . . 3218404
33865 16:17:54 (3) FreeVirtualMemory: . . . . . 2024592
33866 16:17:54 (3) InstallDate: . . . . . 20050611131842.000000-420
33867 16:17:54 (3) LargeSystemCache: . . . . . 0
33868 16:17:54 (3) LastBootUpTime: . . . . . 20060928032340.084059-420
33869 16:17:54 (3) LocalDateTime: . . . . . 20060928161754.317000-420
33870 16:17:54 (3) Locale: . . . . . 0409
33871 16:17:54 (3) Manufacturer: . . . . . Microsoft Corporation
33872 16:17:54 (3) MaxNumberOfProcesses: . . . . . -1
33873 16:17:54 (3) MaxProcessMemorySize: . . . . . 2097024
33874 16:17:54 (3) *Name: . . . . . Microsoft Windows XP Professional|
    C:\WINDOWS|\Device\Harddisk0\Partition1
33875 16:17:54 (3) NumberOfProcesses: . . . . . 64
33876 16:17:54 (3) NumberOfUsers: . . . . . 3
33877 16:17:54 (3) Organization: . . . . .
33878 16:17:54 (3) OSLanguage: . . . . . 1033
33879 16:17:54 (3) OSType: . . . . . 18
33880 16:17:54 (3) Primary: . . . . . TRUE
33881 16:17:54 (3) ProductType: . . . . . 1
33882 16:17:54 (3) QuantumLength: . . . . . 0
33883 16:17:54 (3) QuantumType: . . . . . 0
33884 16:17:54 (3) RegisteredUser: . . . . . AlainL
33885 16:17:54 (3) SerialNumber: . . . . . 55274-OEM-0011903-00101
33886 16:17:54 (3) ServicePackMajorVersion: . . . . . 2
33887 16:17:54 (3) ServicePackMinorVersion: . . . . . 0
33888 16:17:54 (3) SizeStoredInPagingFiles: . . . . . 3907548

```

```

33889 16:17:54 (3) Status: ..... OK
33890 16:17:54 (3) SuiteMask: ..... 272
33891 16:17:54 (3) SystemDevice: ..... \Device\HarddiskVolume1
33892 16:17:54 (3) SystemDirectory: ..... C:\WINDOWS\system32
33893 16:17:54 (3) SystemDrive: ..... C:
33894 16:17:54 (3) TotalVirtualMemorySize: ..... 2097024
33895 16:17:54 (3) TotalVisibleMemorySize: ..... 1965036
33896 16:17:54 (3) Version: ..... 5.1.2600
33897 16:17:54 (3) WindowsDirectory: ..... C:\WINDOWS
33898 16:17:54 (3)
.
.
.

```

Collecting information about the QFE

```

33899 16:17:54 (3) - Win32_OSRecoveryConfiguration #1 -----
33900 16:17:54 (3) AutoReboot: ..... TRUE
33901 16:17:54 (3) DebugFilePath: ..... %SystemRoot%\MEMORY.DMP
33902 16:17:54 (3) DebugInfoType: ..... 0
33903 16:17:54 (3) ExpandedDebugFilePath: ..... C:\WINDOWS\MEMORY.DMP
33904 16:17:54 (3) ExpandedMiniDumpDirectory: ..... C:\WINDOWS\Minidump
33905 16:17:54 (3) KernelDumpOnly: ..... FALSE
33906 16:17:54 (3) MiniDumpDirectory: ..... %SystemRoot%\Minidump
33907 16:17:54 (3) *Name: ..... Microsoft Windows XP Professional|
                                     C:\WINDOWS\Device\Harddisk0\Partition1

33908 16:17:54 (3) OverwriteExistingDebugFile: ..... TRUE
33909 16:17:54 (3) SendAdminAlert: ..... TRUE
33910 16:17:54 (3) WriteDebugInfo: ..... FALSE
33911 16:17:54 (3) WriteToSystemLog: ..... TRUE
33912 16:17:54 (3)
33913 16:17:54 (3) - Win32_QuickFixEngineering #1 -----
33914 16:17:54 (3) CSName: ..... PC-ALAINL-02
33915 16:17:54 (3) Description: .....
33916 16:17:54 (3) FixComments: .....
33917 16:17:54 (3) *HotFixID: ..... File 1
33918 16:17:54 (3) InstalledBy: .....
33919 16:17:54 (3) InstalledOn: .....
33920 16:17:54 (3) *ServicePackInEffect: ..... KB873333
33921 16:17:54 (3) - Win32_QuickFixEngineering #2 -----
33922 16:17:54 (3) CSName: ..... PC-ALAINL-02
33923 16:17:54 (3) Description: .....
33924 16:17:54 (3) FixComments: .....
33925 16:17:54 (3) *HotFixID: ..... File 1
33926 16:17:54 (3) InstalledBy: .....
33927 16:17:54 (3) InstalledOn: .....
33928 16:17:54 (3) *ServicePackInEffect: ..... KB873339
33929 16:17:54 (3) - Win32_QuickFixEngineering #3 -----
33930 16:17:54 (3) CSName: ..... PC-ALAINL-02
33931 16:17:54 (3) Description: .....
33932 16:17:54 (3) FixComments: .....
33933 16:17:54 (3) *HotFixID: ..... File 1
33934 16:17:54 (3) InstalledBy: .....
33935 16:17:54 (3) InstalledOn: .....
33936 16:17:54 (3) *ServicePackInEffect: ..... KB885250
33937 16:17:54 (3) - Win32_QuickFixEngineering #4 -----
33938 16:17:54 (3) CSName: ..... PC-ALAINL-02
33939 16:17:54 (3) Description: .....
33940 16:17:54 (3) FixComments: .....
33941 16:17:54 (3) *HotFixID: ..... File 1
33942 16:17:54 (3) InstalledBy: .....
33943 16:17:54 (3) InstalledOn: .....
33944 16:17:54 (3) *ServicePackInEffect: ..... KB885835
33945 16:17:54 (3) - Win32_QuickFixEngineering #5 -----
33946 16:17:54 (3) CSName: ..... PC-ALAINL-02
33947 16:17:54 (3) Description: .....
33948 16:17:54 (3) FixComments: .....
33949 16:17:54 (3) *HotFixID: ..... File 1
33950 16:17:54 (3) InstalledBy: .....
33951 16:17:54 (3) InstalledOn: .....
33952 16:17:54 (3) *ServicePackInEffect: ..... KB885836

.
.
.
35161 16:17:58 (3) - Win32_QuickFixEngineering #157 -----
35162 16:17:58 (3) CSName: ..... PC-ALAINL-02
35163 16:17:58 (3) Description: ..... Security Update for Windows XP (KB922616)
35164 16:17:58 (3) FixComments: ..... Update
35165 16:17:58 (3) *HotFixID: ..... KB922616

```

```

35166 16:17:58 (3) InstalledBy: ..... SYSTEM
35167 16:17:58 (3) InstalledOn: ..... 8/9/2006
35168 16:17:58 (3) *ServicePackInEffect: ..... SP3
35169 16:17:58 (3) - Win32_QuickFixEngineering #158 -----
35170 16:17:58 (3) CSName: ..... PC-ALAINL-02
35171 16:17:58 (3) Description: ..... Security Update for Windows XP (KB925486)
35172 16:17:58 (3) FixComments: ..... Update
35173 16:17:58 (3) *HotFixID: ..... KB925486
35174 16:17:58 (3) InstalledBy: ..... SYSTEM
35175 16:17:58 (3) InstalledOn: ..... 9/27/2006
35176 16:17:58 (3) *ServicePackInEffect: ..... SP3
35177 16:17:58 (3)
.
.
.

```

Collecting system information about Windows Services

```

35178 16:17:58 (0) ** - Services information
35179 16:17:58 (3) AdobeActiveFileMonitor Running Auto Tracks files that are managed by Adobe >...
35180 16:17:58 (3) Alerter Stopped Disabled Notifies selected users and computers of >...
35181 16:17:58 (3) ALG Running Manual Provides support for 3rd party protocol plug>...
35182 16:17:58 (3) AppMgmt Stopped Manual Provides software installation services such>...
.
.
.
35278 16:17:58 (3) wuauserv Running Auto Enables the download and installation of >...
35279 16:17:58 (3) WZCSVC Running Auto Provides automatic configuration for the >...
35280 16:17:58 (3) xmlprov Stopped Manual Manages XML configuration files on a domain>...
35281 16:17:58 (3)
.
.
.

```

Collecting system information about WMI binary files

```

35282 16:17:58 (0) ** - WMI Binary files information
35283 16:17:59 (3) CIMWIN32.DLL 1352192 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35284 16:17:59 (3) CMDEVTEGPROV.DLL 45568 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35285 16:17:59 (3) DSPROV.DLL 120320 - Microsoft Corporation - 5.1.2600.0 (xpclient.010817-1148)
35286 16:17:59 (3) ESSCLI.DLL 247808 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35287 16:17:59 (3) EVNTRPRV.DLL 22016 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35288 16:17:59 (3) FASTPROX.DLL 472064 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35289 16:17:59 (3) FRAMEDYN.DLL 185856 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35290 16:17:59 (3) FWDPROV.DLL 53248 - Microsoft Corporation - 5.1.2600.0 (xpclient.010817-1148)
35291 16:17:59 (3) KRNLPROV.DLL 24576 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35292 16:17:59 (3) MOFCOMP.EXE 16384 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
.
.
.
35343 16:17:59 (3) WMIPRVSE.EXE 218112 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35344 16:17:59 (3) WMIPSESS.DLL 41472 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35345 16:17:59 (3) WMISVC.DLL 144896 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35346 16:17:59 (3) WMITIMEP.DLL 52224 - Microsoft Corporation - 5.1.2600.0 (xpclient.010817-1148)
35347 16:17:59 (3) WMIUTILS.DLL 95232 - Microsoft Corporation - 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
35348 16:17:59 (3)
35349 16:17:59 (0) ** - NT Event Log information
35350 16:17:59 (4) Reading registry (REG_DWORD)
'HKLM\SYSTEM\CurrentControlSet\Control\TimeZoneInformation\ActiveTimeBias'.
35351 16:17:59 (4) Reading registry (REG_DWORD)
'HKLM\SYSTEM\CurrentControlSet\Control\TimeZoneInformation\ActiveTimeBias'.
35352 16:17:59 (3)
.
.
.

```

Collecting system information about DCOM, WMI and WMIADAPTER event log entries

```

35353 16:17:59 (3) Querying event log to locate events older than 20 day(s)
since Friday, September 08, 2006 at 16:11.
35354 16:17:59 (3)
35355 16:17:59 (3) 060 DCOM event(s):
35356 16:17:59 (4)
35357 16:17:59 (3) #000331: DCOM (10016) - Error - 27 September 2006 19:28:30 (GMT+7)
35358 16:17:59 (3) The application-specific permission settings do not grant Local Activation

```

```

35359 16:17:59 (3) permission for the COM Server application with CLSID
{0C0A3666-30C9-11D0-8F20-00805F2CD064}
35360 16:17:59 (3) to the user REDMOND\alainl SID (S-1-5-21-2127521184-1604012920-1887927527-2058479).
35361 16:17:59 (3) This security permission can be modified using the Component Services
35362 16:17:59 (3) administrative tool.
35363 16:17:59 (3) #000332: DCOM (10016) - Error - 27 September 2006 19:29:53 (GMT+7)
35364 16:17:59 (3) The application-specific permission settings do not grant Local Activation
35365 16:17:59 (3) permission for the COM Server application with CLSID
{0C0A3666-30C9-11D0-8F20-00805F2CD064}
35366 16:17:59 (3) to the user REDMOND\alainl SID (S-1-5-21-2127521184-1604012920-1887927527-2058479).
35367 16:17:59 (3) This security permission can be modified using the Component Services
35368 16:17:59 (3) administrative tool.
.
.
35753 16:18:00 (3)
35754 16:18:00 (3) ZERO WMIADAPTER event.
35755 16:18:00 (4)
35756 16:18:00 (3) Querying event log to locate events created during the execution of
WMIdiag since Thursday, September 28, 2006 at 16:11.
35757 16:18:00 (3)
35758 16:18:00 (3) ZERO DCOM event.
35759 16:18:00 (4)
35760 16:18:00 (3) ZERO WINMGMT event.
35761 16:18:00 (4)
35762 16:18:00 (3) ZERO WMIADAPTER event.
35763 16:18:00 (4)
.
.

```

Collecting system information about suspicious shutdowns

```

35764 16:18:00 (3) Querying event log to locate startup/shutdown event(s).
35765 16:18:00 (3)
35766 16:18:00 (3) 6 Startup/Shutdown event(s) found!
35767 16:18:00 (4)
35768 16:18:00 (3) No incorrect system shutdown found.
.
.

```

WMI report

```

35941 16:18:02 (0) **
35942 16:18:02 (0) ** -----
35943 16:18:02 (0) ** WMI REPORT: BEGIN -----
35944 16:18:02 (0) ** -----
35945 16:18:02 (0) ** -----
35946 16:18:02 (0) ** -----
35947 16:18:02 (0) ** Windows XP - Service pack 2 - 32-bit - User 'PC-ALAINL-02\ALAINL' on computer 'PC-ALAINL-02'.
35948 16:18:02 (0) ** -----
35949 16:18:02 (0) ** Environment: ..... OK.
35950 16:18:02 (0) ** System drive: ..... C: (Disk #0 Partition #0).
35951 16:18:02 (0) ** Drive type: ..... IDE (IC25N080ATMR04-0).
35952 16:18:02 (0) ** INFO: The following UNEXPECTED binary files are/is found in the WBEM folder: ..... 1 FILE(S)!
35953 16:18:02 (0) ** - WBEMDUMP.EXE, 122880 bytes, 7/25/2002 10:05:48 PM
35954 16:18:02 (0) ** => This list is provided for information. Unexpected binary file(s) in'C:\WINDOWS\SYSTEM32\WBEM\'
35955 16:18:02 (0) ** do not necessarily represent an error. For instance, the file(s) listed can be added by
35956 16:18:02 (0) ** any applications implementing WMI providers.
35957 16:18:02 (0) ** => NO ACTION is required.
35958 16:18:02 (0) **
35959 16:18:02 (0) ** There are no missing WMI system files: ..... OK.
35960 16:18:02 (0) ** There are no missing WMI repository files: ..... OK.
35961 16:18:02 (0) ** WMI repository state: ..... NOT TESTED.
35962 16:18:02 (0) ** BEFORE running WMIdiag:
35963 16:18:02 (0) ** The WMI repository has a size of: ..... 29 MB.
35964 16:18:02 (0) ** - Disk free space on 'C:': ..... 918 MB.
35965 16:18:02 (0) ** - INDEX.BTR, 3268608 bytes, 9/28/2006 4:11:30 PM
35966 16:18:02 (0) ** - INDEX.MAP, 1852 bytes, 9/28/2006 4:11:30 PM
35967 16:18:02 (0) ** - MAPPING.VER, 4 bytes, 9/28/2006 4:11:30 PM
35968 16:18:02 (0) ** - MAPPING1.MAP, 15592 bytes, 9/28/2006 4:11:30 PM
35969 16:18:02 (0) ** - MAPPING2.MAP, 15596 bytes, 9/28/2006 4:11:05 PM
35970 16:18:02 (0) ** - OBJECTS.DATA, 26976256 bytes, 9/28/2006 4:11:30 PM
35971 16:18:02 (0) ** - OBJECTS.MAP, 13752 bytes, 9/28/2006 4:11:30 PM
35972 16:18:02 (0) ** AFTER running WMIdiag:

```

```

35973 16:18:02 (0) ** The WMI repository has a size of: ..... 29 MB.
35974 16:18:02 (0) ** - Disk free space on 'C:': ..... 918 MB.
35975 16:18:02 (0) ** - INDEX.BTR, 3268608 bytes, 9/28/2006 4:18:00 PM
35976 16:18:02 (0) ** - INDEX.MAP, 1852 bytes, 9/28/2006 4:18:00 PM
35977 16:18:02 (0) ** - MAPPING.VER, 4 bytes, 9/28/2006 4:18:01 PM
35978 16:18:02 (0) ** - MAPPING1.MAP, 15592 bytes, 9/28/2006 4:17:31 PM
35979 16:18:02 (0) ** - MAPPING2.MAP, 15596 bytes, 9/28/2006 4:18:01 PM
35980 16:18:02 (0) ** - OBJECTS.DATA, 26976256 bytes, 9/28/2006 4:18:00 PM
35981 16:18:02 (0) ** - OBJECTS.MAP, 13752 bytes, 9/28/2006 4:18:01 PM
35982 16:18:02 (0) ** -----
35983 16:18:02 (0) ** INFO: Windows Firewall status: ..... ENABLED.
35984 16:18:02 (0) ** Windows Firewall Profile: ..... DOMAIN.
35985 16:18:02 (0) ** Windows Firewall 'RemoteAdmin' status: ..... DISABLED.
35986 16:18:02 (0) ** => This will prevent any WMI remote connectivity to this machine.
35987 16:18:02 (0) ** - You can adjust the configuration by executing the following command:
35988 16:18:02 (0) ** i.e. 'NETSH.EXE FIREWALL SET SERVICE REMOTEADMIN ENABLE SUBNET'
35989 16:18:02 (0) **
35990 16:18:02 (0) ** Windows Firewall application exception for 'UNSECAPP.EXE': ..... MISSING.
35991 16:18:02 (0) ** => This will prevent any script and MMC application asynchronous callbacks to this machine.
35992 16:18:02 (0) ** - You can adjust the configuration by executing the following command:
35993 16:18:02 (0) ** i.e. 'NETSH.EXE FIREWALL SET ALLOWEDPROGRAM
C:\WINDOWS\SYSTEM32\WBEM\UNSECAPP.EXE WMICALLBACKS ENABLE'

35994 16:18:02 (0) ** -----
35995 16:18:02 (0) ** DCOM Status: ..... OK.
35997 16:18:02 (0) ** WMI registry setup: ..... OK.
35998 16:18:02 (0) ** INFO: WMI service has dependents: ..... 3 SERVICE(S)!
35999 16:18:02 (0) ** - Security Center (WSCSVC, StartMode='Automatic')
36000 16:18:02 (0) ** - Windows Firewall/Internet Connection Sharing (ICS) (SHAREDACCESS, StartMode='Automatic')
36001 16:18:02 (0) ** - SMS Agent Host (CCMEEXEC, StartMode='Automatic')
36002 16:18:02 (0) ** => If the WMI service is stopped, the listed service(s) will have to be stopped as well.
36003 16:18:02 (0) ** Note: If the service is marked with (*), it means that the service/application uses WMI but
36004 16:18:02 (0) ** there is no hard dependency on WMI. However, if the WMI service is stopped,
36005 16:18:02 (0) ** this can prevent the service/application to work as expected.
36006 16:18:02 (0) **
36007 16:18:02 (0) ** RPCSS service: ..... OK (Already started).
36008 16:18:02 (0) ** WINMGMT service: ..... OK (Already started).
36009 16:18:02 (0) ** -----
36010 16:18:02 (0) ** WMI service DCOM setup: ..... OK.
36011 16:18:02 (0) ** WMI components DCOM registrations: ..... OK.
36012 16:18:02 (0) ** WMI ProgID registrations: ..... OK.
36013 16:18:02 (0) ** WMI provider DCOM registrations: ..... OK.
36014 16:18:02 (0) ** WMI provider CIM registrations: ..... OK.
36015 16:18:02 (0) ** WMI provider CLSIDs: ..... OK.
36016 16:18:02 (0) ** WMI providers EXE/DLL availability: ..... OK.
36017 16:18:02 (0) ** -----
36018 16:18:02 (0) ** Overall DCOM security status: ..... OK.
36019 16:18:02 (0) ** Overall WMI security status: ..... OK.
36020 16:18:02 (0) ** - Started at 'Root' -----
36021 16:18:02 (0) ** INFO: WMI permanent SUBSCRIPTION(S): ..... 7.
36022 16:18:02 (0) ** - ROOT/CCM/POLICY, CCM_PolicyReplicationConsumer.Id="{9099D177-1AD6-46e6-BBC0-70F460786953}".
36023 16:18:02 (0) ** 'SELECT * FROM __ClassOperationEvent WHERE TargetClass ISA "CCM_Policy_Config"'
36024 16:18:02 (0) ** - ROOT/CCM/POLICY, CCM_PolicyReplicationConsumer.Id="{9099D177-1AD6-46e6-BBC0-70F460786953}".
36025 16:18:02 (0) ** 'SELECT * FROM __NamespaceCreationEvent'
36026 16:18:02 (0) ** - ROOT/CCM/POLICY, CCM_PolicyReplicationConsumer.Id="{9099D177-1AD6-46e6-BBC0-70F460786953}".
36027 16:18:02 (0) ** 'SELECT * FROM __ClassOperationEvent WHERE TargetClass ISA "CCM_Policy"'
36028 16:18:02 (0) ** - ROOT/CCM/POLICY, CCM_PolicyReplicationConsumer.Id="{9099D177-1AD6-46e6-BBC0-70F460786953}".
36029 16:18:02 (0) ** 'SELECT * FROM __ClassOperationEvent WHERE TargetClass ISA "CCM_Policy_EmbeddedObject"'
36030 16:18:02 (0) ** - ROOT/SUBSCRIPTION, LogFileEventConsumer.Name="LogFileForSvc".
36031 16:18:02 (0) ** 'SELECT * FROM __InstanceModificationEvent WITHIN 10 Where TargetInstance ISA 'Win32_Service'
36032 16:18:02 (0) ** - ROOT/SUBSCRIPTION, MSFT_UCScenarioControl.Name="Microsoft WMI Updating
Consumer Scenario Control".
36033 16:18:02 (0) ** 'SELECT * FROM __InstanceOperationEvent WHERE TargetInstance ISA 'MSFT_UCScenario'
36034 16:18:02 (0) ** - ROOT/SUBSCRIPTION, NTEventLogEventConsumer.Name="SCM Event Log Consumer".
36035 16:18:02 (0) ** 'select * from MSFT_SCMEventLogEvent'
36036 16:18:02 (0) **
36037 16:18:02 (0) ** INFO: WMI TIMER instruction(s): ..... 1.
36038 16:18:02 (0) ** - Interval: ROOT/CIMV2, 'MyIntervalTimerEvent', 5000'.
36039 16:18:02 (0) **
36040 16:18:02 (0) ** WMI ADAP status: ..... OK.
36041 16:18:02 (0) ** WMI MONIKER CONNECTIONS: ..... OK.
36042 16:18:02 (0) ** WMI CONNECTIONS: ..... OK.
36043 16:18:02 (0) ** WMI GET operations: ..... OK.
36044 16:18:02 (0) ** WMI MOF representations: ..... OK.
36045 16:18:02 (0) ** WMI QUALIFIER access operations: ..... OK.
36046 16:18:02 (0) ** WMI ENUMERATION operations: ..... OK.
36047 16:18:02 (0) ** WMI EXECQUERY operations: ..... OK.
36048 16:18:02 (0) ** WMI GET VALUE operations: ..... OK.
36049 16:18:02 (0) ** WMI WRITE operations: ..... NOT TESTED.

```

```
36050 16:18:02 (0) ** WMI PUT operations: ..... NOT TESTED.
36051 16:18:02 (0) ** WMI DELETE operations: ..... NOT TESTED.
36052 16:18:02 (0) ** WMI static instances retrieved: ..... 5453.
36053 16:18:02 (0) ** WMI dynamic instances retrieved: ..... 0.
36054 16:18:02 (0) ** WMI instance request cancellations (to limit performance impact): ..... 1.
36055 16:18:02 (0) ** -----
36056 16:18:02 (0) ** # of Event Log events BEFORE WMIdiag execution since the last 20 day(s):
36057 16:18:02 (0) **   DCOM: ..... 60.
36058 16:18:02 (0) **   WINMGMT: ..... 5.
36059 16:18:02 (0) **   WMIADAPTER: ..... 0.
36060 16:18:02 (0) ** => Verify the WMIdiag LOG at line #35353 for more details.
36061 16:18:02 (0) ** -----
36062 16:18:02 (0) ** # of additional Event Log events AFTER WMIdiag execution:
36063 16:18:02 (0) **   DCOM: ..... 0.
36064 16:18:02 (0) **   WINMGMT: ..... 0.
36065 16:18:02 (0) **   WMIADAPTER: ..... 0.
36066 16:18:02 (0) ** -----
36067 16:18:02 (0) ** WMI Registry key setup: ..... OK.
36068 16:18:02 (0) ** -----
36069 16:18:02 (0) ** -----
36070 16:18:02 (0) ** -----
36071 16:18:02 (0) ** -----
36072 16:18:02 (0) ** -----
36073 16:18:02 (0) ** -----
36074 16:18:02 (0) ** -----
36075 16:18:02 (0) ** ----- WMI REPORT: END -----
36076 16:18:02 (0) ** -----
36077 16:18:02 (0) ** -----
36078 16:18:02 (0) ** SUCCESS: WMIdiag determined that WMI works CORRECTLY.
36079 16:18:02 (0) ** -----
36080 16:18:02 (0) ** WMIdiag executed in 7 minutes.
36081 16:18:02 (3)
36083 16:18:02 (3)
36084 16:18:02 (0) ** WMIdiag v2.0 ended on Thursday, September 28, 2006 at 16:18 (W:129 E:3 S:0).
```