Windows Registry knowledge base (winreg-kb)

Release 20220806

unknown

Aug 06, 2022

CONTENTS

winreg-kb is a project to build a Windows Registry Knowledge Base.

winregrc is a Python module part of winreg-kb to allow reuse of Windows Registry Resources.

The source code is available from the project page.

CHAPTER

ONE

WINDOWS REGISTRY

1.1 Windows Registry hives

1.2 Windows Registry files

1.2.1 Windows Registry files - Windows 3.1

On Windows 3.1 the SHCC file format is used to store Windows Registry data.

Paths of known Windows Registry files:

1.2.2 Windows Registry files - Windows 9x/Me

On Windows 9x/Me the CREG file format is used to store Windows Registry data.

Paths of known Windows Registry files:

Root keys

The root key of both SYSTEM.DAT and USER.DAT contains an empty name string.

1.2.3 Windows Registry files - Windows NT

On Windows NT and later the REGF file format is used to store Windows Registry data.

Paths of known Windows Registry files:

TODO Windows NT 3.1 user specific file under %SystemRoot%\System32\config TODO BCD check Windows 8 and 10 TODO userdiff no longer present in Windows 10 ? TODO what about \Windows\profiles\user profile\user.dat ? TODO what about \Windows\System32\SMI\Store\Machine\SCHEMA.DAT (Windows 7)

Root keys

The root key names of the different Windows Registry files differ per version of Windows.

Root key - default

TODO

Root key - NTUSER.DAT

Root key - SAM

Root key - SECURITY

Root key - SOFTWARE

Root key - Syscache.hve

Where {%GUID%} is a placeholder for a random GUID in the form: {00000000-0000-0000-0000-000000000000}

Note how consistent are the GUIDs icw CreateHive ?

Root key - SYSTEM

Root key - userdiff

TODO

Root key - UsrClass.dat

1.2.4 Notes

TODO what about earlier versions of Windows?

1.3 Multilingual User Interface (MUI) form

The Multilingual User Interface (MUI) form is used to store strings in multiple languages. This is also known as Registry string redirection.

The actual strings are stored in the resource sections of PE/COFF files (also referred to as resource file) e.g .exe, .dll.

In the Registry the MUI strings are referenced using the following form:

```
@path,-stringID[;comment]
```

Where:

- path; is the full path or filename of the resource file that contains the string

- stringID; the identifier of the string within the string resource of the resource file
- comment; a comment

For example the MUI form string:

```
@%SystemRoot%\system32\SHELL32.dll,-9227
```

This MUI form string refers to the string with identifier 9227, stored in the string resource of SHELL32.dll. Which, for an English version of SHELL32.dll, corresponds to: "My Documents".

An example of a MUI form with comment:

```
@wfplwfs.inf,%WfpLwfs_Name%;WFP LightWeight Filters
```

1.3.1 Notes

What about the form (seen in the Shell Folder value LocalizedString):

```
@%SystemRoot%\System32\fvecpl.dll,-1#immutable1
```

What does # represent here?

```
@wd.inf,%WdServiceDisplayName%;Microsoft Watchdog Timer Driver
```

Yet another variation seen in Windows 2000.

```
@C:\WINNT\system32\shell32.dll,-9227@1033,My Documents
```

1.3.2 External Links

- MSDN: Multilingual User Interface
- MSDN: Using Registry String Redirection

CHAPTER

TWO

WINDOWS SYSTEM KEYS

2.1 Application compatibility cache

The Application compatibility cache can be found in the following Windows Registry keys.

In Windows 2000 and XP:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatibility
```

In Windows 2003 and later:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatCache
```

Note that several sources claim that the Application Compatibility Cache is part of the Application Compatibility Database. However unfortunately these claims are not backed by sources or facts. Since the previous article does not mention the relationship between the cache and the database, this document the Application Compatibility Cache to part of the Windows Application Compatibility subsystem instead.

Note that the actual difference between the Application Compatibility Cache and Shim (Database) Cache is currently unknown. Be aware that in other sources the terms can be used interchangeable. Since MSDN explicitly defines BaseFlushAppcompatCache and ShimFlushCache, there is likely a subtle difference to what data is cached. Also see: Understanding Shims.

2.1.1 Windows 2000

Windows 2000 stores Application Compatibility related data in subkeys in:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatibility
```

At this time it is unclear if these subkeys serve the same purpose as the AppCompatCache value in later versions of Windows.

The subkeys are named as the executable files e.g. `Uninstall.exe` and have been seen to contain the following values:

Also seen values named like `00008 WindowsNT4.0`.

Windows 2000 unknown value

The Windows 2000 unknown value is variable of size and consists of:

Contains additional data if "Unknown 4 > 0"

```
Empty?
00000000  0c 00 00 00 00 00 00 00  06 00 00 00 00 00 00 00  |................|

With data:
00000000  0c 00 00 00 00 00 00 00  06 00 00 00 04 00 00 00  |................|

00000010  10 00 00 00 00 00 00 00  00 00 15 00 ff ff ff ff  |................|
00000020  ff ff ff ff 0f 00 00 00                           |........(...A.u.|

Sting byte size followed by string:
00000020                           28 00 00 00 41 00 75 00  |........(...A.u.|
00000030  74 00 6f 00 43 00 41 00  44 00 20 00 41 00 70 00  |t.o.C.A.D. .A.p.|
00000040  70 00 6c 00 69 00 63 00  61 00 74 00 69 00 6f 00  |p.l.i.c.a.t.i.o.|
00000050  6e 00 00 00                                       |n.......|

00000050              00 00 00 00                           |n.......|
```

Windows 2000 DllPatch value

The Windows 2000 DllPatch value is variable of size and contains an UTF-16 little-endian formatted string with end-of-string character e.g. 'shcmn.dll 7'.

It is currently unclear what the trailing number represents.

2.1.2 Windows XP

Windows XP stores the application compatibility cache in the value: AppCompatCache.

The value data consists of:

- header
  - array of LRU cache entry index values
- array of cache entries (suggested that the maximum is 92)

Note that 64-bit versions of Windows XP will use the Windows 2003 64-bit format.

Windows XP application compat cache header

The Windows XP application compat cache header is 400 bytes of size and consists of:

Windows XP 32-bit application compat cache entry

The Windows XP 32-bit application compat cache entry is 552 bytes of size and consists of:

Note that the last modification time applies to that of the file e.g. for NTFS this is the last modified time of the file as stored in the $STANDARD_INFORMATION attribute.

2.1.3 Windows 2003

Windows 2003 stores the application compatibility cache in the value: AppCompatCache

The value data consists of:

- header
- array of cache entries (suggested that the maximum is 512)
- string data

Windows 2003 application compat cache header

The Windows 2003 application compat cache header is 8 bytes of size and consists of:

Windows 2003 32-bit application compat cache entry

The Windows 2003 32-bit application compat cache entry is 24 bytes of size and consists of:

Note that the last modification time applies to that of the file e.g. for NTFS this is the last modified time of the file as stored in the $STANDARD_INFORMATION attribute.

Windows 2003 64-bit application compat cache entry

The Windows 2003 64-bit application compat cache entry is 32 bytes of size and consists of:

Note that the last modification time applies to that of the file e.g. for NTFS this is the last modified time of the file as stored in the $STANDARD_INFORMATION attribute.

2.1.4 Windows Vista and 2008

Windows Vista and 2008 store the application compatibility cache in the value: AppCompatCache

The value data consists of:

- header
- array of cache entries (suggested that the maximum is 1024)
- string data

[NOTE] If the cache is empty it will only consists of a header.

Windows Vista application compat cache header

The Windows Vista application compat cache header is 8 bytes of size and consists of:

Windows Vista 32-bit application compat cache entry

The Windows Vista 32-bit application compat cache entry is 24 bytes of size and consists of:

Note that the last modification time applies to that of the file e.g. for NTFS this is the last modified time of the file as stored in the $STANDARD_INFORMATION attribute.

Windows Vista 64-bit application compat cache entry

The Windows Vista 64-bit application compat cache entry is 32 bytes of size and consists of:

Note that the last modification time applies to that of the file e.g. for NTFS this is the last modified time of the file as stored in the $STANDARD_INFORMATION attribute.

2.1.5 Windows 7 and 2008 R2

Windows 7 and 2008 R2 store the application compatibility cache in the value: AppCompatCache

The value data consists of:

- header
- array of cache entries (suggested that the maximum is 1024)
- data
- string data

Windows 7 application compat cache header

The Windows 7 application compat cache header is 128 bytes of size and consists of:

Windows 7 32-bit application compat cache entry

The Windows 7 32-bit application compat cache entry is 32 bytes of size and consists of:

Note that the last modification time applies to that of the file e.g. for NTFS this is the last modified time of the file as stored in the $STANDARD_INFORMATION attribute.

Windows 7 64-bit application compat cache entry

The Windows 7 64-bit application compat cache entry is 48 bytes of size and consists of:

Note that the last modification time applies to that of the file e.g. for NTFS this is the last modified time of the file as stored in the $STANDARD_INFORMATION attribute.

2.1.6 Windows 8

Windows 8 store the application compatibility cache in the value: AppCompatCache

The value data consists of:

- header
- array of cache entries

Windows 8 application compat cache header

The Windows 8 application compat cache header is 128 bytes of size and consists of:

Windows 8.0 application compat cache entry

The Windows 8.0 application compat cache entry is variable bytes of size and consists of:

Note that the last modification time applies to that of the file e.g. for NTFS this is the last modified time of the file as stored in the $STANDARD_INFORMATION attribute.

Windows 8.1 application compat cache entry

The Windows 8.1 application compat cache entry is variable bytes of size and consists of:

Note that the last modification time applies to that of the file e.g. for NTFS this is the last modified time of the file as stored in the $STANDARD_INFORMATION attribute.

2.1.7 Windows 10

Windows 10 store the application compatibility cache in the value: AppCompatCache

The value data consists of:

- header
- array of cache entries

Windows 10 application compat cache header

The Windows 10 application compat cache header is 48 bytes of size and consists of:

The Windows 10 Creator update application compat cache header is 52 bytes of size and consists of:

Windows 10 application compat cache entry

The Windows 10 application compat cache entry is variable bytes of size and consists of:

Note that the last modification time applies to that of the file e.g. for NTFS this is the last modified time of the file as stored in the $STANDARD_INFORMATION attribute.

2.1.8 Insertion flags

TODO describe

2.1.9 Shim flags

TODO describe

2.1.10 Data

TODO describe

2.1.11 Notes

```
https://technet.microsoft.com/en-us/library/cc787360(v=ws.10).aspx

Are these related?
0x00000001 MS-DOS-based program
0x00000002 OS/2-based program
0x00000004 Windows-based 16-bit program
0x00000008 Windows-based 32-bit program
0x0000000C Windows-based 16-bit and 32-bit program
0x0000000F Any version of a program
0x00000010 Return user name instead of computer name for GetComputerName.
0x00000020 Return Terminal Server build number instead of Windows 2000 build number for␣
↪GetVersion.
0x00000040 Synchronize user .ini file to system version.*
0x00000080 Do not substitute user \Windows directory.**
0x00000100 Disable registry mapping for program or registry key.
0x00000200 Per-object user/system global mapping
0x00000400 Return system \Windows directory instead of user \Windows directory for␣
↪GetWindowsDir.
0x00000800 Limit the reported physical memory for GlobalMemoryStatus.
0x00001000 Log object creation to file.
0x20000000 Do not put program to sleep on unsuccessful keyboard polling (Windows-based␣
↪16-bit programs only).
```

Related DLLs:

- apphelp.dll; related to "AppHelp" functionality and Application Compatibility database
- kernel32.dll; base cache management functionality

Is the Application compatibility cache in Windows also referred to as AppHelpCache?

AppHelp: https://msdn.microsoft.com/en-us/library/bb432181(v=vs.85).aspx

Different shim types? MSIE and RPC shim types?

Related Registry keys:

```
HKLM\Sofware\Microsoft\Windows NT\CurrentVersion\AppCompatFlags
```

2.1.12 External links

- Leveraging the Application Compatibility Cache in Forensic Investigations, by Andrew Davis, 2012

2.2 Background activity moderator (BAM)

The Background Activity Moderator (BAM) key seems to have been introduced in Windows 10 after version 1709.

The BAM keys can be found in the following Registry paths:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\bam\UserSettings\
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\bam\State\UserSettings\
```

Within the UserSettings key, there is a key for each user SID containing a value for each tracked executable.

2.2.1 Example Entry

Registry Key:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-
↪321011808-3761883066-353627080-1000
```

Value Name:

```
\Device\HarddiskVolume1\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
```

Value Data:

```
00000000  15 3e ae 36 57 de d4 01 00 00 00 00 00 00 00 00  |.>®6WÞÔ.........|
00000010  00 00 00 00 02 00 00 00                          |........|
```

2.2.2 Value Data Format

2.3 Boot Verification key

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\BootVerification
```

The BootVerification key stores configuration for Bootvrfy.exe, a program included in Windows Server 2003 that notifies the system that startup was successful. Bootvrfy.exe can be run on a local or remote computer.

Known values of the BootVerification key:

To run a custom startup verification program the standard startup verification functions in Winlogon need to be disabled. This can be done by setting the Winlogon ReportBootOk value to 0.

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
```

2.3.1 Boot Verification Program key

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\BootVerificationProgram
```

The BootVerificationProgram key stores configuration for a custom startup verification program.

Known values of the BootVerificationProgram key:

According Windows server 2003 documentation Bootvrfy.exe and a custom startup verification program cannot be used in parallel.

2.3.2 External links

- BootVerification
- BootVerificationProgram
- BootVerificationProgram\ImagePath
- ReportBootOk

2.4 Cached credentials

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Winlogon
```

Values:

2.4.1 Credentials cache

```
HKEY_LOCAL_MACHINE\Security\Cache
```

Values:

Where %NUMBER% contains the number of the cached credential.

NL$Control value

```
00000000  04 00 01 00 0a 00 00 00                           |........|
```

NL$%NUMBER% value

```
metadata
* username size
* domain size
* Length of the full domain name

0x00000000  0e 00 14 00 0e 00 1c 00  00 00 00 00 38 00 04 00  ............8...
0x00000010  53 04 00 00 01 02 00 00  02 00 00 00 14 00 18 00  S...............
```

(continues on next page)

(continued from previous page)

```
0x00000020  72 0f 92 b3 b1 f8 cc 01                           r...............
FILETIME

0x00000020                           04 00 01 00 01 00 00 00  r...............
0x00000030  01 00 00 00 20 00 00 00  10 00 00 00 20 00 00 00  .... ....... ...

CH: random 16 byte key that is used to generate the decryption key for the encrypted data
0x00000040  e6 ad 1f 22 b9 d1 d3 48  22 f6 d6 61 33 d7 32 74  ..."...H"..a3.2t

T
0x00000050  29 4c 83 1b af bc ca c9  fc 27 9c be 1e 44 2b 69  )L.......'...D+i

Encrypted data
0x00000060  52 46 67 5f f6 85 b0 0f  7a a3 69 03 cc 72 4b 8b  RFg_....z.i..rK.
0x00000070  8b 51 e9 9c 4a 65 92 2d  19 7d 6f 94 d2 81 93 0d  .Q..Je.-.}o.....
0x00000080  f2 9e 7d 2e 11 17 46 a0  31 ac 2c 65 49 89 c2 c0  ..}...F.1.,eI...
0x00000090  92 7a 63 6c ca b2 74 ba  5f 73 c0 d3 6c 0c 58 51  .zcl..t._s..l.XQ
0x000000a0  46 e9 45 48 9b ce 86 a1  68 ae f7 12 f8 d2 c7 7e  F.EH....h......~
0x000000b0  4d 39 a9 bd d4 ad fc e8  b0 b1 94 36 c5 4d 1f 3b  M9.........6.M.;
0x000000c0  3c ce b8 dc a9 50 41 54  f4 5a 31 61 57 66 66 7a  <....PAT.Z1aWffz
0x000000d0  0d 54 9a c0 7e d4 1a a8  e6 af 83 fb cd 61 a1 fe  .T..~........a..
0x000000e0  85 31 ce c9 24 fa f3 a5  7e 71 c9 a4 81 11 e3 b7  .1..$...~q......
0x000000f0  7c ce fb 38 b0 81 b9 75  cc 78 7e 66 9c 7b 4d a7  |..8...u.x~f.{M.
0x00000100  7d 6e 55 d6 8d 22 2d e9  8d 48 0c 22 f1 bc 6b 58  }nU.."-..H."..kX
0x00000110  17 84 db 5b ba 91 8a 39  70 a1 d8 b5 16 df 99 cf  ...[...9p.......
0x00000120  ea f1 af dc 75 27 ea 83  22 ff 8a 5e 63 b2 a9 f9  ....u'.."..^c...
0x00000130  b4 05 47 26 b8 e7 e4 b7  06 bc d9 4b 0f 20 92 25  ..G&.......K. .%
0x00000140  07 7a a5 6b 4e 54 4a 19  19 51 bf 5f c2 09 8b 5e  .z.kNTJ..Q._...^
0x00000150  f1 a3 be aa 1f c3 66 c3  cd 09 7b 85 45 02 0d 28  ......f...{.E..(
0x00000160  02 a5 f8 8a f2 b1 52 a3  a3 dc a4 c7 ed f5 ca 6c  ......R........l
0x00000170  13 3c e5 18 3d fe b3 fc  28 3f be 9b 62 d0 1a 5a  .<..=...(?..b..Z
0x00000180  90 ce e2 a6 c2 aa 2d 40  78 d8 cc db a4 a7 44 e8  ......-@x.....D.
0x00000190  0d ff c8 08 49 19 5b 21  67 f2 62 be 7b f2 be d3  ....I.[!g.b.{...
0x000001a0  37 18 53 33 61 3e 21 7a  e6 08 e3 f2 d5 1c 81 ce  7.S3a>!z........
0x000001b0  9a 45 71 85 bf a6 e9 fd  ea 7e b7 2f 01 0d 7d c7  .Eq......~./..}.
0x000001c0  46 9f e5 73                                       F..s
```

```
Decrypted data:
0x00000000  6e 37 5e e6 a7 99 6c 5c  55 85 74 67 09 af a0 65  n7^...l\U.tg...e
0x00000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0x00000020  01 00 00 00 00 00 00 00  c4 01 00 00 02 00 00 00  ................
0x00000030  14 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0x00000040  00 00 00 00 00 00 00 00                           ........t.d.u.n.

Sizes from metadata
0e 00 14 00 0e 00 1c 00

0x00000040                           74 00 64 00 75 00 6e 00  ........t.d.u.n.
0x00000050  67 00 61 00 6e 00                                 g.a.n...S.H.I.E.

0x00000050                     00 00                          g.a.n...S.H.I.E.
```

(continues on next page)

(continued from previous page)

```
0x00000050                                        53 00 48 00 49 00 45 00  g.a.n...S.H.I.E.
0x00000060  4c 00 44 00 42 00 41 00  53 00 45 00                          L.D.B.A.S.E.S.H.

0x00000060                                                    53 00 48 00  L.D.B.A.S.E.S.H.
0x00000070  49 00 45 00 4c 00 44 00  42 00 41 00 53 00 45 00              I.E.L.D.B.A.S.E.

0x00000080  2e 00 4c 00 4f 00 43 00  41 00 4c 00                          ..L.O.C.A.L.t.d.

0x00000080                                                    74 00 64 00  ..L.O.C.A.L.t.d.
0x00000090  75 00 6e 00 67 00 61 00  6e 00 00 00                          u.n.g.a.n...T.i.

0x00000090                                                    54 00 69 00  u.n.g.a.n...T.i.
0x000000a0  6d 00 6f 00 74 00 68 00  79 00 20 00 44 00 75 00              m.o.t.h.y. .D.u.
0x000000b0  6e 00 67 00 61 00 6e 00                                       n.g.a.n.\.\.c.o.

0x000000b0                                        5c 00 5c 00 63 00 6f 00  n.g.a.n.\.\.c.o.
0x000000c0  6e 00 74 00 72 00 6f 00  6c 00 6c 00 65 00 72 00              n.t.r.o.l.l.e.r.
0x000000d0  5c 00 68 00 6f 00 6d 00  65 00 5c 00 25 00 75 00              \.h.o.m.e.\.%.u.
0x000000e0  73 00 65 00 72 00 6e 00  61 00 6d 00 65 00 25 00              s.e.r.n.a.m.e.%.

0x000000f0  48 00 3a 00 01 02 00 00  07 00 00 00 07 02 00 00  H.:.............
0x00000100  07 00 00 00 53 00 48 00  49 00 45 00 4c 00 44 00  ....S.H.I.E.L.D.
0x00000110  42 00 41 00 53 00 45 00  07 00 00 20 01 05 00 00  B.A.S.E.... ....
0x00000120  00 00 00 05 15 00 00 00  97 2a 67 79 a0 54 4a b6  .........*gy.TJ.
0x00000130  19 87 28 7e 3c 02 00 00  01 04 00 00 00 00 00 05  ..(~<...........
0x00000140  15 00 00 00 97 2a 67 79  a0 54 4a b6 19 87 28 7e  .....*gy.TJ...(~
0x00000150  43 00 4f 00 4e 00 54 00  52 00 4f 00 4c 00 4c 00  C.O.N.T.R.O.L.L.
0x00000160  45 00 52 00                                       E.R.
```

NL$7

```
00000000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000020  00 00 00 00 00 00 00 00  04 00 01 00 00 00 00 00  |................|
00000030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000a0  00 00 00 00 00 00 00 00                           |........|
```

2.4.2 External Links

- Cached domain logon information

2.5 Certificates

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\SystemCertificates\AuthRoot\Certificates
```

2.5.1 External Links

- Certificates Tools and Settings

2.6 Codepage

The codepage settings are stored in the key:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Nls\CodePage
```

Values:

2.7 Component object model (COM) class identifiers (CLSIDs)

The component object model (COM) class Identifier (CLSID) key can be found in:

```
HKEY_CLASSES_ROOT\CLSID\{%GUID%}
HKEY_CLASSES_ROOT\Wow6432Node\CLSID\{%GUID%}
```

Sub keys:

MSDN defines DefaultIcon as a REG_SZ value but in Windows XP it seems to be a key where the icon resource identifier is stored in the default value of the key.

Values:

2.7.1 Type libraries key

The type libraries (typelib or tlb) key can be found in:

```
HKEY_CLASSES_ROOT\TypeLib\{%GUID%}
HKEY_CLASSES_ROOT\Wow6432Node\TypeLib\{%GUID%}
```

Sub keys:

2.7.2 Type library identifier subkey

Sub keys:

Values:

Type library version subkey

Sub keys:

TODO: Determine what MSDN means with the LCID may have a neutral sublanguage ID. Is 0 the neutral sublanguage ID?

Type library locale subkey

Sub keys:

Type library platform subkey

Values:

Type library help directory subkey

Values:

2.7.3 External Links

- CLSID Key
- ProxyStubClsid
- Registering a Type Library

2.8 Current control set

The Windows Registry contains the Current control set key:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet
```

2.8.1 Current Control Set - Windows 9x/Me

In Windows 9x/Me the Current Control Set key is stored in the SYSTEM.DAT Registry file.

2.8.2 Current Control Set - Windows NT

On Windows NT the Current Control Set key is only present at run-time. The contents of the key is stored in the SYSTEM Registry file and can be determined by reading the Current value from the key:

```
<RootKey>\Select
```

The Current value contains number of the current control set. Normally 1 or 2 but other values like 3 or 47 are known to be used. For example a value of 1 maps to the Control Set key:

```
<RootKey>\ControlSet001
```

Normally there are multiple Control Set keys the role each of the Control Set keys can be different:

> ControlSet001 may be the last control set you booted with, while ControlSet002 could be what is known as the last known good control set, or the control set that last successfully booted Windows.

These roles are defined by the other values in the Select key:

Notes

- Determine if a value of 0 indicates not set
- Confirm if speculations that 9 is the largest value ControlSet00#

2.9 Environment variables

The environment variables are stored in multiple keys.

2.9.1 Session Manager\Environment key

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager\Environment
```

Values:

For example the "windir" value that contains "%SystemRoot%".

2.9.2 Windows\CurrentVersion key

Values of environment variables such as %%ProgramFiles% can be derived from values in the Windows\CurrentVersion key:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion
```

Values:

For example the "ProgramFilesDir (x86)" value that contains "C:\Program Files (x86)".

2.9.3 Windows NT\CurrentVersion key

The %SystemRoot% environment variable can be derived from values in the Windows NT\CurrentVersion key:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion
```

Values:

For example the "SystemRoot" value that contains "C:\Windows"

2.9.4 CurrentVersion\ProfileList key

Values of environment variables such as %ProgramData% can be derived from values in the CurrentVersion\ProfileList key:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\ProfileList
```

Values:

If the AllUsersProfile value does not start with an environment variable or an absolute path, but a relative path, it is currently assumed that the value should be prefixed with the value in ProfilesDirectory.

2.9.5 User specific environment variables

```
HKEY_CURRENT_USER\Environment
```

Values:

2.10 File system

Windows file system settings are stored in the File system key.

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\FileSystem
```

2.11 File System key

Values:

2.11.1 NTFS Disable Short (8.3) Filename Creation value

2.11.2 NTFS Disable Last Access Time value

TODO the explanation of the values differs between versions of Windows

The meaning of the value 0 for Windows 2000 according to NtfsDisableLastAccessUpdate:

```
When listing directories, NTFS updates the last-access timestamp on each
directory it detects, and it records each time change in the NTFS log.
```

In contrast to the meaning of the value 0 for Windows 2003 according to NtfsDisableLastAccessUpdate:

```
NTFS updates the last-accessed timestamp of a file whenever that file is opened.
```

TODO value does not exist by default until Windows XP SP3/Vista

2.11.3 External Links

Windows 2000

- NtfsDisable8dot3NameCreation
- NtfsDisableLastAccessUpdate
- Win31FileSystem
- NtfsEncryptionService
- NtfsAllowExtendedCharacterIn8dot3Name

Windows 2003

- NtfsAllowExtendedCharacterIn8dot3Name
- NtfsDisable8dot3NameCreation
- NtfsDisableLastAccessUpdate
- NtfsEncryptionService
- Win95TruncatedExtensions

2.12 Language

The language settings are stored in the key:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Nls\Language
```

Values:

2.12.1 Language groups

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Nls\Language Groups
```

2.13 Local Security Authority (LSA)

Windows 2000 and later.

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa
```

2.13.1 Boot key

The boot key can be determined as following.

Determine a 32-character string by combining the classnames of the following subkeys:

- JD
- Skew1
- GBG
- Data

The string contains a base16 encoded 16-byte binary data that contains the scrambled key data. To unscramble the key data:

```
scrambled_key = codecs.decode(class_name_string, 'hex')

key = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
for index, scrambled_index in enumerate([
    8, 5, 4, 2, 11, 9, 13, 3, 0, 6, 1, 12, 14, 10, 15, 7]):
  key[index] = scrambled_key[scrambled_index]

key = codecs.encode(b''.join(key), 'hex')
```

2.13.2 LSA key

The Local Security Authority (LSA) (or Syskey) is a 128-bit RC4 encryption key used to protect credentials stored in the Windows Registry.

```
Key: HKEY_LOCAL_MACHINE\Security\Policy\PolSecretEncryptionKey
Default value
```

```
Windows XP

00000000  01 00 00 00 01 00 00 00  00 00 00 00              |............?.|

RC4 encrypted data
00000000                                       fd d4 3f b3  |............?.|
00000010  ee 4f cd 45 2d 02 e8 1e  f2 ac bd 4f fc 15 12 09  |.O.E-......O....|
00000020  0a b5 48 17 33 8f 42 79  8b 89 11 d8 ec 6e 1c ec  |..H.3.By.....n..|
00000030  38 5f 27 df 72 ca 57 96  8d 16 d9 37              |8_'.r.W....7..d.|

RC4 key material
00000030                                       c4 14 64 d1  |8_'.r.W....7..d.|
00000040  a8 47 7a d4 4b a3 62 d8  e7 2b ef 76              |.Gz.K.b..+.v|
```

```
md5 = MD5.new()
md5.update(boot_key)

iteration = 0
while iteration < 1000:
  md5.update(value_data[60:76])
  iteration += 1

rc4_key = md5.digest()

rc4 = ARC4.new(rc4_key)
decrypted_data = rc4.decrypt(value_data[12:60])

lsa_key = decrypted_data[16:32]
```

```
0x00000000  80 3a ce f0 5f 15 d3 11  b7 e6 00 80 5f 48 ca eb  .:.._........_H..

0x00000010  01 d6 5d f4 43 aa 0a 86  d9 42 d1 17 34 ce 66 7c  ..].C....B..4.f|
0x00000020  24 9a 83 44 c6 a7 57 30  44 dc 27 06 26 94 77 8a  $..D..W0D.'.&.w.
```

2.13.3 NL$KM

```
Key: HKEY_LOCAL_MACHINE\Security\Policy\Secrets\NL$KM\CurrVal
Default value
```

```
Windows XP
00000000  48 00 00 00 48 00 00 20  9c c3 0c 00              |H...H.. ........|

DES encrypted data
00000000                                        c2 0d 08 10  |H...H.. ........|
00000010  9a 04 04 bf 14 8b c7 d0  0b e2 9c 40 52 a7 8e aa  |...........@R...|
00000020  01 49 25 70 71 dc a0 69  8e 6c 03 1c b7 db 19 5c  |.I%pq..i.l.....\|
00000030  8f f4 11 d1 8d 73 07 b0  6f 1a db 0b ee cb 69 7f  |.....s..o.....i.|
00000040  73 50 24 82 f8 e1 a6 27  97 a9 cc 04 8e e4 ca bb  |sP$....'........|
00000050  33 68 00 7c                                       |3h.||
```

```
decrypted data (_LSA_BLOB)

0x00000000  40 00 00 00 01 00 00 00  09 fe 44 48 1b 35 73 b7  @.........DH.5s.
0x00000010  3b 1d fc f7 48 9f c9 60  3b 60 7d cf 62 35 50 fd  ;...H..`;`}.b5P.
0x00000020  b5 d8 8f 21 75 ec 01 e9  85 25 96 6c 68 52 c9 30  ...!u....%.lhR.0
0x00000030  fb 1d b6 9d cd 8c 14 90  91 de f1 dd 5d d7 64 2a  ............].d*
0x00000040  ce 40 97 5a f1 59 71 20                           .@.Z.Yq
```

```
Windows 7
00000000  00 00 00 01                                       |....a.!v.......N|

00000000              61 d8 21 76  d9 02 af de bd aa ba 4e  |....a.!v.......N|
00000010  f3 3f de 78 03 00 00 00  00 00 00 00 1a 7a 20 be  |.?.x.........z .|
00000020  73 10 0b 57 34 88 16 81  00 42 50 a1 8f 5e 78 46  |s..W4....BP..^xF|
```

(continues on next page)

(continued from previous page)

```
00000030  bb f3 5e 61 9b 59 fa de  ff 14 7c c1 70 97 66 8e  |..^a.Y....|.p.f.|
00000040  c8 98 54 5c 8e 0e 13 7d  e7 ba 9a 98 8b cf a4 6f  |..T\...}.......o|
00000050  6d 84 5f 84 9c 9f d9 08  c3 5d 5c bd e9 1a 78 c6  |m._......]\...x.|
00000060  63 de 80 2d ec 3c 75 1f  1b e0 10 f5 24 1c 5d 41  |c..-.<u.....$.]A|
00000070  dd fa 85 7c 6e 20 cd 5e  a4 ac c0 53 7e c3 d6 ef  |...|n .^...S~...|
00000080  23 e2 2c b0 bd 74 52 19  cd a0 4e b2 00 00 00 00  |#.,..tR...N.....|
00000090  00 00 00 00 00 00 00 00  00 00 00 00              |............|
```

2.14 Mounted devices

The mounted devices settings are stored in the key:

```
HKEY_LOCAL_MACHINE\SYSTEM\MountedDevices
```

Seen on:

- Windows 2000
- Windows XP
- Windows 2003
- Windows Vista
- Windows 2008
- Windows 7
- Windows 8.0
- Windows 8.1
- Windows 10
- Windows 11

Values:

Where the following variants of %IDENTIFIER% have been observed:

- "\DosDevices\C:"
- "??\Volume{01234567-89ab-cdef-0123-456789abcdef}"

Where the value data consist of either:

- Device string value data
- GPT partition value data
- MBR partition value data

2.14.1 Device string value data

The device string value data is variable of size and consists of:

For example:

```
\??\FDC#GENERIC_FLOPPY_DRIVE#6&12345678&0&0#{01234567-89ab-cdef-0123-456789abcdef}
\??\IDE#CdRomQEMU_QEMU_DVD-ROM_____________________1.6._____#5&12345678&0&0.1.0#
↪{01234567-89ab-cdef-0123-456789abcdef}
\??\SCSI#CdRom&Ven_VBOX&Prod_CD-ROM#4&0123456&0&010000#{01234567-89ab-cdef-0123-
↪456789abcdef}
_??_USBSTOR#Disk&Ven_Generic&Prod_Flash_Disk&Rev_8.07#01234567&0#{01234567-89ab-cdef-
↪0123-456789abcdef}
```

2.14.2 GPT partition value data

The GPT partition value data is 24 bytes of size and consists of:

2.14.3 MBR fixed-disk value data

The MBR partition value data is 12 bytes of size and consists of:

2.14.4 Notes

The mountvol.exe Windows CLI tool can show information about mounted devices.

2.15 Prefetch

2.15.1 Prefetch Parameters key

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\
↪PrefetchParameters
```

Values:

Enable Prefetcher value

Values:

2.15.2 External Links

- Disabling Prefetch
- Disable Prefetch (Standard 7 SP1)

2.16 Run and RunOnce

Run and RunOnce keys cause programs to run each time a user logs on. There are system and per-user Run and RunOnce keys.

2.16.1 Run and RunOnce keys

System Run and RunOnce keys:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnceEx
```

Per-user Run and RunOnce keys:

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce
```

Values:

RunOnce\Setup sub key

Contains first-boot activities after setup or when the Add/Remove Programs Wizard was used.

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce\Setup
```

2.16.2 RunServices and RunServicesOnce

Only on Windows 9x/Me.

Run in the background when the logon dialog box first appears, or at the boot process stage if there is no logon.

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce
```

Values:

2.16.3 Notes

```
description-string=commandline
```

According to Microsoft:

```
By default, the value of a RunOnce key is deleted before the command line is
run. You can prefix a RunOnce value name with an exclamation point (!) to defer
deletion of the value until after the command runs. Without the exclamation
point prefix, if the RunOnce operation fails the associated program will not be
asked to run the next time you start the computer.

By default, these keys are ignored when the computer is started in Safe Mode.
The value name of RunOnce keys can be prefixed with an asterisk (*) to force
the program to run even in Safe mode.
```

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\policies\Explorer\Run
HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows\Run
```

Installed "Programs and Features"

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Installer
```

2.16.4 External Links

- Run and RunOnce Registry Keys

2.17 Services and drivers

TODO this page currently contains rough notes, fine tune these

Settings to load/run drivers and services are stored in the Services key:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services
```

Sub keys:

2.17.1 Name sub key

Sub keys:

Values:

Parameters sub key

Sub keys:

Values:

ErrorControl value

ObjectName value

The ObjectName value has a different meaning for different types of Driver or Service Name sub keys.

- For a driver type the ObjectName value contains the Windows NT driver object name that the I/O Manager uses to load the device driver.
- For a service type the ObjectName value contains the account name under which the service will log on to run.

Windows Services shows this value as "LogOn As".

Start value

Windows Services shows this value as "Startup Type".

Type value

2.18 Session manager

The session manager settings are stored in the key:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager
```

Values:

2.18.1 External links

- MSDN: BootExecute

2.19 Shell folder identifiers

Shell folder identifiers are class identifiers (CLSID) with ShellFolder sub key of some COM Class Identifier (CLSID) keys.

```
HKEY_CLASSES_ROOT\CLSID\{%GUID%}\ShellFolder
```

2.19.1 Shell Folder class identifier (CLSID) sub key

Sub keys specific to shell folder identifiers:

Values:

Shell Folder sub key

Values:

Attributes values

Localized String value data

The Localize String value contains a localized version of the folder name, e.g. on Windows XP the folder identifier key:

```
HKEY_CLASSES_ROOT\CLSID\{450d8fba-ad25-11d0-98a8-0800361b1103}
```

Has a LocalizedString value with the following data:

```
@%SystemRoot%\system32\SHELL32.dll,-9227
```

Which is the MUI Form for "My Documents".

2.19.2 External Links

- Implementing the Basic Folder Object Interfaces
- libfwsi: Shell Folder identifiers

2.20 System restore

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\SystemRestore
```

2.21 Task scheduler

2.21.1 SchedulingAgent key

In Windows XP:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\SchedulingAgent
```

Values:

2.21.2 Schedule key

In Windows Vista and later:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Schedule
```

Sub keys:

Values:

TaskCache sub key

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache
```

Sub keys:

TaskCache\Tree sub key

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree
```

Values:

TaskCache\Tree\%GUID% sub key

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\
↪%GUID%
```

Values:

TaskCache\Tree\%GUID%\DynamicInfo sub key

Seen in Windows Vista, Windows 2008 and Windows 7:

The dynamic information entry is 28 bytes of size and consists of:

```
0x00000000  03 00 00 00 1c ec 45 16  3f 04 ca 01 00 00 00 00  ......E.?.......
0x00000010  00 00 00 00 00 00 00 00  00 00 00 00              ............

0x00000000  03 00 00 00 16 6f 4a 0f  7f fe c6 01 66 b7 6c 0d  .....oJ.....f.l.
0x00000010  6b 4c c9 01 2b 04 07 80  00 00 00 00              kL..+.......
```

Seen in Windows 8 and Windows 10:

TODO: check Windows 2012

The dynamic information entry is 36 bytes of size and consists of:

```
0x00000000  03 00 00 00 4b 5a 0b 60  ff 6a cd 01 5c 32 e7 45  ....KZ.`.j..\2.E
0x00000010  1b b6 ce 01 20 04 07 80  00 00 00 00 a2 b1 86 4f  .... ..........O
0x00000020  1b b6 ce 01                                       ....
```

Path value

The path value is relative from:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree
```

For example the path:

```
\Microsoft\Windows\Media Center\ehDRMInit
```

Corresponds to:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\
↪Microsoft\Windows\Media Center\ehDRMInit
```

Triggers value

Note that the FILETIME value appear to be stored in local time.

2.22 Time zones

2.22.1 Time zone information key

The time zone information is stored in the following key:

```
HKEY_LOCAL_MACHINE\CurrentControlSet\Control\TimeZoneInformation
```

Values:

Notes:

- the relation between the local time and bias is as following:

```
UTC = local time + bias
```

- the names are stored outside the Windows Registry and are references using MUI Form for example:

```
@tzres.dll,-931
```

- the TimeZoneKeyName is not always present
- The RealTimeIsUniversal value is not installed in the system by default and is not officially supported by Windows

The values in this key corresponds with the TIME_ZONE_INFORMATION and DYNAMIC_TIME_ZONE_INFORMATION structures.

2.22.2 Time Zones key

The time zones definitions are stored in the following key:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Time Zones
```

Sub keys:

%TIMEZONENAME% represents the name of the Windows name of the time zone. The Unicode organization maintains windowsZones.xml to map the Windows names to corresponding IANA names.

Values:

Time Zones %TIMEZONENAME% sub key

Sub keys:

Note that not every Time Zones name sub key contains a Dynamic Daylight Saving Time sub key (Dynamic DST).

Values:

Dynamic Daylight Saving Time sub key

The Dynamic Daylight Saving Time sub key contains time zone information for time zones that apply different daylight saving per year.

Values:

Where %YEAR% represents the year the dynamic daylight saving time zone information applies to, e.g. 2006.

Data Structures

SYSTEMTIME structure

The SYSTEMTIME structure is 16 bytes of size and consists of:

Registry Time Zone information structure

The Registry Time Zone information (`TIME_ZONE_INFORMATION` or `_REG_TZI_FORMAT`) structure is 44 bytes of size and consists of:

The wDayOfWeek member of a SYSTEMTIME structure represents the appropriate weekday, and the wDay member represents the occurrence of the day of the week within the month, where 5 indicates the final occurrence during the month if that day of the week only occurs 4 times in the month.

If the wYear member is 0, the date is relative, meaning the daylight savings occurs yearly. Otherwise the date is abosule, meaning daylight savings only changes once.

Note that DaylightBias can be set when DaylightDate is not set.

2.22.3 External Links

- Computer Time Management and Embedded Systems (Standard 7 SP1)
- TIME_ZONE_INFORMATION structure (timezoneapi.h)
- DYNAMIC_TIME_ZONE_INFORMATION structure (timezoneapi.h)
- SYSTEMTIME structure (minwinbase.h)
- windowsZones.xml

2.23 USB storage

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Enum\USBSTOR
```

Sub key level 1: Disk&Ven_&Prod_&Rev_0.00

```
<Device Type>&Ven_<Vendor>&Prod_<Product>&Rev_<Revision Number>
```

Sub key level 2: 1002131402536a&0

Sub keys:

Values:

2.24 Volume shadow copies

2.24.1 FilesNotToSnapshot key

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\BackupRestore\FilesNotToSnapshot
```

2.24.2 External Links

- Excluding Files from Shadow Copies

2.25 Windows Error Reporting (WER) keys

2.25.1 Windows Error Reporting (WER) system key

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\Windows Error Reporting
```

Sub keys:

Values:

BypassDataThrottling value

0 - Disable data bypass throttling 1 - Enable data bypass throttling

ConfigureArchive value

1 - Parameters only (default on Windows 7) 2 - All data (default on Windows Vista)

Consent sub key

Values:

DefaultConsent value

1 - Always ask (default) 2 - Parameters only 3 - Parameters and safe data 4 - All data

DefaultOverrideBehavior value

0 - Vertical consent will override the default consent (default) 1 - Default consent will override the application-specific consent|

Debug sub key

Sub keys:

Values:

UIHandles sub key

Values:

Hangs sub key

Values:

LocalDumps sub key

Per-application setting can be define by an application-specific key under:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\Windows Error Reporting\LocalDumps
```

For example an application-key for MyApplication.exe

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\Windows Error Reporting\LocalDumps\
↪MyApplication.exe
```

Values:

DumpType value

CustomDumpFlags value

The CustomDumpFlags value contains a bitwise combination of the MINIDUMP_TYPE enumeration values.

2.25.2 Windows Error Reporting (WER) user key

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\Windows Error Reporting
```

Sub keys:

Values:

2.25.3 Notes

Kernel faults sub key

Sub keys:

Queue sub key

Values:

```
C:\\Windows\\Minidump\\MMDDYY-#-01.dmp
```

Other

C:\Users%USERNAME%\AppData\Local\Microsoft\Windows\WER

Sub directories:

2.25.4 External Links

- Collecting User-Mode Dumps
- MINIDUMP_TYPE enumeration
- WER Settings

2.26 Windows product information

Windows product information can be found in the key:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion
```

Values:

2.27 Windows system locations

Windows system locations can be found in the key:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion
```

Values:

CHAPTER

THREE

SECURITY ACCOUNTS MANAGER KEYS

3.1 Security Accounts Manager (SAM)

The Security Accounts Manager (SAM) is stored in the key:

```
HKEY_LOCAL_MACHINE\SAM\SAM
```

Sub keys:

Values:

3.1.1 C value data

The C value data is variable of size and consists of:

Format version

3.2 Domains

The Security Accounts Manager (SAM) domains are stored in the key:

```
HKEY_LOCAL_MACHINE\SAM\SAM\Domains
```

Sub keys:

Values:

3.2.1 Account or Builtin sub key

Sub keys:

Values:

F value data

V value data

The V value data consists of:

- 17 x user information descriptors
  - security descriptor
  - username
  - full name
  - comment
  - user comment
  - **Unknown**
  - home directory
  - home directory connect
  - script path
  - profile path
  - workstations
  - hours allowed
  - **Unknown**
  - LM hash (LANMAN)
  - NTLM hash
  - **Unknown**
  - **Unknown**
- user information data

User information descriptor

A user information descriptor is 12 byte of size and consists of:

3.2.2 Aliases sub key

Sub keys:

Where %RID% is the relative identifier (RID) which corresponds to the last sub authority of the SID.

Aliases RID sub key

Values:

C value data

Aliases Members sub key

Sub keys:

Where %SID% is the security identifier (SID) in the form of a string e.g. S-1-5.

Aliases Members SID sub key

Sub keys:

Where %RID% is the relative identifier (RID) which corresponds to the last sub authority of the SID.

3.2.3 Groups sub key

Sub keys:

C value data

3.2.4 Users sub key

Sub keys:

Where %RID% is the relative identifier (RID) which corresponds to the last sub authority of the SID.

Users RID sub key

Values:

F value data

Extended data:

Note that the relative identifier (RID) is sometimes referred to as user number or user identifier.

User account control flags

The user account control flags (or USER_ACCOUNT Codes) are defined in subauth.h

Note that these flags differ from ADS_USER_FLAG_ENUM. Mappings between the two are defined in "MS-SAMR: userAccountControl Mapping Table".

Note that the samba project defines these as flags with the WBC_ACB prefix, where WBC is short for winbind client.

Country code

Unknown. Is this suppose to be the country phone prefix?

V value data

Account types

Predefined RIDs

3.2.5 External Links

- ACCOUNT_TYPE Values
- Built-in and Account Domains
- Predefined RIDs
- SAMPR_USER_ALL_INFORMATION
- Security Account Manager (SAM)
- SysKey and the SAM, by Brendan Dolan-Gavitt, February 21, 2008
- USER_ACCOUNT Codes
- userAccountControl Mapping Table
- USER_ALL_INFORMATION structure
- Well-known SIDs

CHAPTER

FOUR

WINDOWS EXPLORER KEYS

4.1 Bit bucket

The Windows Explorer bit bucket key contains Recycler configuration properties and information about the Recycler of connected volumes.

```
HKEY_CURRENT_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\BitBucket
```

Seen on Windows 2000, XP and 2003.

Sub keys:

Values:

4.1.1 BitBucket\%NAME% sub key

Values:

4.2 Known folder identifier keys

A known folder identifier is a GUID that identifies a system folder. It was introduced in Windows Vista to replace the constant special item identifier list (CSIDL).

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\FolderDescriptions
```

Values:

4.2.1 External links

- libfwsi: Known Folder Identifiers

4.3 Moint points

4.3.1 MountPoints

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints
```

Seen on:

- Windows 2000

Sub keys:

Where the following forms of %NAME% have been observed:

- Drive letter, for example “C”

MountPoints name sub key

Sub keys:

Values:

4.3.2 MountPoints2

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2
```

Seen on:

- Windows XP
- Windows 2003
- Windows Vista
- Windows 2008
- Windows 7
- Windows 8.0
- Windows 8.1
- Windows 10

Sub keys:

Where the following variants of %NAME% have been observed:

- Drive letter, for example “C”
- Volume identifier (GUID), for example “{01234567-89ab-cdef-0123-456789abcdef}”
- UNC path, for example “##1.2.3.4#username”

MountPoints2 name sub key

Sub keys:

Values:

4.4 Most recently used (MRU)

The Windows Registry contains various keys with information about Most Recently files Used (MRU). Windows Explorer (or Windows shell), extensively uses such keys. Several different variants of MRU keys are known to be used, such as:

- Keys with a MRUList value
- Keys with a MRUListEx value
- BagMRU key

4.4.1 Keys with a MRUList value

Values:

String MRUList values

The following keys with a MRUList value contain %ALPHA% values that consists of an UTF-16 little-endian formatted string with an end-of-string character.

Sub keys of: `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\`

Shell Item List MRUList values

The following keys with a MRUList value contain %ALPHA% values that consists of a Shell Item List.

Sub keys of: `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\`

4.4.2 Keys with a MRUListEx value

Values:

The value data of the numeric value depends on the sub key.

String MRUListEx values

The following keys with a MRUListEx value contain %NUMERIC% values that consists of an UTF-16 little-endian formatted string with an end-of-string character.

Shell Item List MRUListEx values

The following keys with a MRUListEx value contain %NUMERIC% values that consists of a Shell Item List.

Sub keys of: `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\`

String and Shell Item MRUListEx values

The following keys with a MRUListEx value contain %NUMERIC% values that consists of a String and Shell Item. The String and Shell Item is variable of size and consists of:

Sub keys of: `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\`

String and Shell Item List MRUListEx values

The following keys with a MRUListEx value contain %NUMERIC% values that consists of a String and Shell Item List. The String and Shell Item List is variable of size and consists of:

Sub keys of: `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\`

4.4.3 BagMRU key

The values in the BagMRU and sub keys are also referred to as "shellbags".

BagMRU keys as of XP (stored in NTUSER.DAT)

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\Shell\BagMRU
HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\BagMRU
```

Additional BagMRU keys as of Vista (stored in USRCLASS.DAT)

```
HKEY_CURRENT_USER\Software\Classes\Local Settings\Software\Microsoft\Windows\Shell\BagMRU
HKEY_CURRENT_USER\Software\Classes\Local Settings\Software\Microsoft\Windows\ShellNoRoam\
↪BagMRU
HKEY_CURRENT_USER\Software\Classes\Wow6432Node\Local Settings\Software\Microsoft\Windows\
↪Shell\BagMRU
HKEY_CURRENT_USER\Software\Classes\Wow6432Node\Local Settings\Software\Microsoft\Windows\
↪ShellNoRoam\BagMRU
```

Seen in Windows 7:

```
HKEY_CURRENT_USER\Local Settings\Software\Microsoft\Windows\Shell\BagMRU
```

The BagMRU sub keys form a hierarchy similar to a folder structure.

Values:

Bag number shell sub key

The numbered sub keys of the Bags key have a Shell sub key e.g.

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\Shell\Bags\1\Shell
```

This key contains various values:

4.4.4 Notes

This section contains some notes on explorer MRU keys that need to be completed.

Wallpaper MRU key MRUListEx value

Sub keys of: HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\

```
00000000  43 00 3a 00 5c 00 57 00  49 00 4e 00 44 00 4f 00  |C.:.\.W.I.N.D.O.|
00000010  57 00 53 00 5c 00 42 00  6c 00 75 00 65 00 20 00  |W.S.\.B.l.u.e. .|
00000020  4c 00 61 00 63 00 65 00  20 00 31 00 36 00 2e 00  |L.a.c.e. .1.6...|
00000030  62 00 6d 00 70 00 00 00  70 00 00 00 00 00 00 00  |b.m.p...p.......|
00000040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000050  00 00 00 00 78 01 08 00  00 00 00 00 00 00 00 00  |....x...........|
00000060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000a0  00 00 00 00 28 f6 0b 00  00 00 00 00 70 4b 0c 00  |....(.......pK..|
000000b0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000c0  00 00 00 00 00 00 00 00  28 f6 0b 00 00 00 00 00  |........(.......|
000000d0  78 5b 0c 00 00 00 00 00  20 f6 0b 00 00 00 00 00  |x[...... .......|
000000e0  78 01 08 00 00 00 00 00  00 00 00 00 00 00 00 00  |x...............|
000000f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000100  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000110  00 00 00 00 00 00 00 00  78 01 08 00 92 02 00 00  |........x.......|
00000120  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000130  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000140  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000150  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000160  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000170  00 00 00 00 00 00 00 00  28 f6 0b 00 00 00 00 00  |........(.......|
00000180  01 02 00 00 00 00 00 00  68 4b 0c 00 08 10 00 00  |........hK......|
00000190  68 4b 0c 00 00 00 00 00  70 4b 0c 00 78 01 08 00  |hK......pK..x...|
000001a0  08 10 00 00 2f 2d f4 77  51 8e e4 77 f8 00 00 00  |..../-.wQ..w....|
000001b0  00 00 00 00 00 00 00 00  00 00 00 00 50 f4 a2 00  |............P...|
000001c0  70 4b 0c 00 00 10 00 00  03 00 00 00 28 8d e4 77  |pK..........(..w|
000001d0  f4 dc 0b 00 36 8e e4 77  04 01 00 00 ab 3d 29 77  |....6..w.....=)w|
000001e0  40 fd a2 00 00 00 00 00  d6 0f 00 00 a8 4e 0c 00  |@............N..|
000001f0  00 d0 fd 7f 00 00 00 00  be 20 08 00 01 00 00 00  |......... ......|
00000200  e0 dc 0b 00 08 00 00 00  30 00 00 00 30 00 00 00  |........0...0...|
00000210  00 60 a9 0f c6 f2 c2 01                           |.`......|
```

Explorer MRUList

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\MRU
```

CIDSizeMRU MRUListEx

Seen on Windows Vista

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\CIDSizeMRU
```

```
00000000  66 00 69 00 72 00 65 00  66 00 6f 00 78 00 2e 00  |f.i.r.e.f.o.x...|
00000010  65 00 78 00 65 00 00 00  00 00 00 00 00 00 00 00  |e.x.e...........|
00000020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
...
00000200  00 00 00 00 00 00 00 00  12 00 00 00 0b 00 00 00  |................|
00000210  22 04 00 00 15 03 00 00  00 00 00 00 00 00 00 00  |"...............|
00000220  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000230  00 00 00 00 00 00 00 00  1a 00 00 00 27 00 00 00  |............'...|
00000240  7c 02 00 00 d6 00 00 00  00 00 00 00 00 00 00 00  |................|
```

MRUListEx

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\FirstFolder
```

Contains an UTF-16 little-endian formatted string.

4.4.5 External Links

- Windows Shell Item format specification

4.5 Multilingual User Interface (MUI) cache

Seen on Windows XP and 2003

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\MUICache
```

Seen on Windows Vista and later

```
HKEY_CURRENT_USER\Software\Classes\Local Settings\Software\Microsoft\Windows\Shell\
↪MuiCache
```

4.6 Program cache

The Windows explorer ProgramsCache Registry values can be stored in the following Windows Registry keys.

- Explorer\StartPage key
- Explorer\StartPage2 key

4.6.1 Explorer\StartPage key

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\StartPage
```

Seen in Windows XP, 2003 and Vista.

Values:

4.6.2 Explorer\StartPage2 key

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\StartPage2
```

Seen in Windows 7.

Values:

Note that the format of the ProgramsCache value data slightly differs from that of the ProgramsCacheSMP and ProgramsCacheTBP value data.

4.6.3 ProgramsCache value data format

ProgramsCacheSMP - Empty list

```
00000000  01 00 00 00                                       |.....|
00000000              00 00 00 00                           |.....|
00000000                          02                        |.....|
```

ProgramsCacheTBP

```
0x00000000  01 00 00 00 0b 00 00 00  01 aa 02 00 00           .............

0x00000000  01 00 00 00 07 00 00 00  01 0e 03 00 00           .............

00000000  01 00 00 00                                       |.............|
number of entries?
00000000              0e 00 00 00                           |.............|
start of entry marker?
00000000                          01                        |.............|
relative offset to next entry?
00000000                             f2 02 00 00            |.............|
00000000                                         14 00 1f   |.............|

shell item list
00000010  80 c8 27 34 1f 10 5c 10  42 aa 03 2e e4 52 87 d6  |..'4..\.B....R..|
```

(continues on next page)

(continued from previous page)

```
...
000002f0  00 78 00 65 00 00 00 00  00 00 00 1c 00           |.x.e............|
end of list?
000002f0                                          00 00     |.x.e............|
start of entry marker?
000002f0                                                01  |.x.e............|
00000300  3c 02 00 00                                       |<........'4..\.B|

shell item list
00000300              14 00 1f 80  c8 27 34 1f 10 5c 10 42  |<........'4..\.B|
...
00000bb0  4f 00 4b 00 2e 00 45 00  58 00 45 00 00 00 00 00  |O.K...E.X.E.....|
00000bc0  00 00 1c 00                                       |.......|
end of list?
00000bc0              00 00                                 |.......|
00000bc0                    02                              |.......|
```

StartPage2\ProgramsCache

```
Window 7
00000000  13 00 00 00 c3 53 5b 62  48 ab c1 4e ba 1f a1 ef  |.....S[bH..N....|
00000010  41 46 fc 19 00 80 00 00  00                       |AF.......~.1....|

shell item list?
00000010                              7e 00 31 00 00 00 00  |AF.......~.1....|
00000020  00 6a 3d 6c 3e 11 00 50  72 6f 67 72 61 6d 73 00  |.j=l>..Programs.|
00000030  00 66 00 08 00 04 00 ef  be 6a 3d 53 3e 6a 3d 6c  |.f.......j=S>j=l|
00000040  3e 2a 00 00 00 c1 e2 00  00 00 00 01 00 00 00 00  |>*..............|
00000050  00 00 00 00 00 3c 00 00  00 00 00 50 00 72 00 6f  |.....<.....P.r.o|
00000060  00 67 00 72 00 61 00 6d  00 73 00 00 00 40 00 73  |.g.r.a.m.s...@.s|
00000070  00 68 00 65 00 6c 00 6c  00 33 00 32 00 2e 00 64  |.h.e.l.l.3.2...d|
00000080  00 6c 00 6c 00 2c 00 2d  00 32 00 31 00 37 00 38  |.l.l.,.-.2.1.7.8|
00000090  00 32 00 00 00 18 00 00  00                       |.2........<...:.|

00000090                              01 3c 02 00 00        |.2........<...:.|

shell item list?
00000090                                          3a 02     |.2........<...:.|
000000a0  32 00 85 05 00 00 30 3f  97 a9 20 00 49 4e 54 45  |2.....0?.. .INTE|
000000b0  52 4e 7e 31 2e 4c 4e 4b  00 00 b8 00 08 00 04 00  |RN~1.LNK........|
000000c0  ef be 6a 3d 6c 3e 6a 3d  6c 3e 2a 00 00 00 b8 e3  |..j=l>j=l>*.....|

...
00012e80  00 00 00 00 00 00 1c 00  00 00 02                 |...........|


0x00019890  6d 00 2e 00 65 00 78 00  65 00 00 00 00 00 00 00  m...e.x.e.......
0x000198a0  20 00 00 00                                       ......9....O.'H

TODO: edge case or remnant data?
0x000198a0              02 ab 95 39  9e 9c 1f 13 4f b8 27 48   ......9....O.'H
0x000198b0  b2 4b 6c 71 74 00                                .Klqt.T...R.1...
```

(continues on next page)

(continued from previous page)

```
0x000198b0                    54 00  00 00 52 00 31 00 00 00  .Klqt.T...R.1...
0x000198c0  00 00 0c 3d a4 33 11 00  54 61 73 6b 42 61 72 00  ...=.3..TaskBar.
0x000198d0  3c 00 08 00 04 00 ef be  0c 3d a4 33 0c 3d a4 33  <........=.3.=.3
0x000198e0  2a 00 00 00 69 ee 00 00  00 00 04 00 00 00 00 00  *...i...........
```

StartPage\ProgramsCache

```
Windows XP and 2003
00000000  09 00 00 00 0b 00                                 |......V...T.1...|

data size
00000000                    56 00  00 00                    |......V...T.1...|
shell item list
00000000                                 54 00 31 00 00 00  |......V...T.1...|
00000010  00 00 04 3b a3 79 11 00  50 72 6f 67 72 61 6d 73  |...;.y..Programs|
00000020  00 00 3c 00 03 00 04 00  ef be 04 3b 8c 79 04 3b  |..<........;.y.;|
00000030  a3 79 14 00 26 00 50 00  72 00 6f 00 67 00 72 00  |.y..&.P.r.o.g.r.|
00000040  61 00 6d 00 73 00 00 00  40 73 68 65 6c 6c 33 32  |a.m.s...@shell32|
00000050  2e 64 6c 6c 2c 2d 32 31  37 38 32 00 18 00 00 00  |.dll,-21782.....|

00000060  01 d4 00 00 00                                    |.......2.#....;.|

00000060                d2 00 32  00 23 03 00 00 04 3b a3  |.......2.#....;.|
00000070  79 20 00 49 4e 54 45 52  4e 7e 31 2e 4c 4e 4b 00  |y .INTERN~1.LNK.|
00000080  00 42 00 03 00 04 00 ef  be 04 3b a3 79 04 3b a3  |.B........;.y.;.|
...
0x000003e0  1c 00 00 00                                       .........T.1....
sentinel of 0x00 seen before shell item list with more than one shell item?
0x000003e0              00 b0 00 00  00                       .........T.1....
shell item list
0x000003e0                               54 00 31 00 00 00 00  .........T.1....
0x000003f0  00 04 3b a3 79 11 00 50  72 6f 67 72 61 6d 73 00  ..;.y..Programs.
...
0x00001020  00 00 00 00 00 1c 00 00  00                      ................
unknown data 9 bytes (0x02 end marker?)
0x00001020                               02 16 00 02 00 00 00  ................
0x00001030  00 00                                             .........2.....:
data size
0x00001030        01 ea 00 00 00                              .........2.....:
shell item list
0x00001030                      e8  00 32 00 1b 06 00 00 3a  .........2.....:
...
0x00004a40  00 65 00 78 00 65 00 00  00 00 00 1c 00 00 00     .e.x.e..........
unknown data 11 bytes
0x00004a40                                                 02  .e.x.e..........
0x00004a50  10 02 19 00 02 00 00 00  00 00                    ................
0x00004a50                                  01 ca 00 00 00     ................
0x00004a50                                                 c8  ................
0x00004a60  00 32 00 42 06 00 00 04  3b 12 7a 20 00 4d 4f 5a  .2.B....;.z .MOZ
...
00004b10  00 65 00 66 00 6f 00 78  00 2e 00 65 00 78 00 65  |.e.f.o.x...e.x.e|
```

(continues on next page)

(continued from previous page)

```
00004b20  00 00 00 00 00 1c 00 00  00                     |.........|
00004b20                              02                  |.........|
```

```
Windows Vista (c3535b62-48ab-c14e-ba1f-a1ef4146fc19 FOLDERID_StartMenu)

0x00000000  0c 00 00 00 c3 53 5b 62  48 ab c1 4e ba 1f a1 ef  .....S[bH..N....
0x00000010  41 46 fc 19                                       AF...|...z.1....
0x00000010              00 7c 00 00  00                       AF...|...z.1....
...
0x00009fe0  72 00 33 00 32 00 2e 00  65 00 78 00 65 00 00 00  r.3.2...e.x.e...
0x00009ff0  00 00 00 00 1c 00 00 00                           .........a.O..M.

TODO: edge case or remnant data?
0x00009ff0                           02 61 ae 4f 05 d8 4d 87  .........a.O..M.
0x0000a000  47 80 b6 09 02 20 c4 b7  00 02                    G.... ....
```

Value data header Windows XP and 2003.

Value data header Windows Vista.

ProgramsCache value data header Windows 7 and 2008.

ProgramsCacheSMP and ProgramsCacheTBP value data header Windows 7 and 2008.

Value data entry.

if sentinel is 0x02 and there is more data then look for 0x00 which should be followed by 02 00 00 00 00 00 01

4.7 Shell folders

System shell folders:

```
HKEY_CURRENT_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders
HKEY_CURRENT_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell␣
↪Folders
HKEY_CURRENT_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell␣
↪Folders\Backup
```

WoW64 (Windows 32-bit on Windows 64-bit) system shell folders:

```
HKEY_CURRENT_MACHINE\Software\WOW6432Node\Microsoft\Windows\CurrentVersion\Explorer\
↪Shell Folders
HKEY_CURRENT_MACHINE\Software\WOW6432Node\Microsoft\Windows\CurrentVersion\Explorer\User␣
↪Shell Folders
HKEY_CURRENT_MACHINE\Software\WOW6432Node\Microsoft\Windows\CurrentVersion\Explorer\User␣
↪Shell Folders\Backup
```

Per-user shell folders:

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders
```

Values:

4.8 Typed paths

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\TypedPaths
```

4.9 User Assist key

The User Assist key contains settings and data of programs that were launched via Windows Explorer (explorer.exe).

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\Currentversion\Explorer\UserAssist
```

Sub keys:

Note that the Settings sub key does not exist by default.

4.9.1 Known GUIDs

Note that the User Assist key does not seem to be present on NT4, therefore this functionality was likely introduced in Windows 2000.

Sometimes more information about the GUID can be found in the key:

```
HKEY_LOCAL_MACHINE\Software\Classes\CLSID\{%GUID%}\
```

4.9.2 GUID sub key

Sub keys:

Values:

Version value data

Count sub key

Values:

Named value

Note does UEME stand for User Experience Monitoring Element/Extension? Note does CTL stand for client? Note does CUA stand for current user (file) associations?

With the exception of the UEME_CTLSESSION value, these values appear to use a similar data types. The structure of a data type depends on the Version value of the GUID sub key. The following versions have been observed:

- version 3, that is used by Windows 2000, XP, 2003 and Vista.
- version 5, that is used by Windows 2008 (R2?), 7, 8.

UEME_CTLSESSION value data

UEME_CTLSESSION value data - version 3

The UEME_CTLSESSION value data - version 3 is 8 bytes of size and consists of:

UEME_CTLSESSION value data - version 5

The UEME_CTLSESSION value data - version 5 is 1612 bytes of size and consists of:

The UEME_CTLSESSION value data - version 5 record is 532 bytes of size and consists of:

Other value data

Other value data - version 3

The other value data - version 3 is 16 bytes of size and consists of:

Other value data - version 5

The other value data - version 5 is 72 bytes of size and consists of:

4.9.3 Settings sub key

Values:

4.9.4 External links

- UserAssist, by Didier Stevens
- Windows 7 Beta: ROT13 Replaced With Vigenère? Great Joke!
- Windows-userassist-keys
- libfwsi: Known Folder Identifiers

CHAPTER

FIVE

EVENTLOG KEYS

5.1 EventLog providers

Information about EventLog providers is stored across multiple keys:

- the Services\EventLog key, which has been around since at least Windows NT 3.5
- the WINEVT\Publishers key, which was introduced in Windows Vista

Note that the combined information of both keys can be needed, for example the Services\EventLog key:

```
Log type                : System
Log source              : Microsoft-Windows-Time-Service
Identifier              : {06edcfeb-0fd0-4e53-acca-a6f8bbf81bcb}
Event message files     : %SystemRoot%\system32\w32time.dll
```

```
Log type                : System
Log source              : W32Time
Identifier              : {06edcfeb-0fd0-4e53-acca-a6f8bbf81bcb}
Event message files     : %SystemRoot%\system32\w32time.dll
```

In combination with the corresponding WINEVT\Publishers key:

```
Name                       : Microsoft-Windows-Time-Service
Identifier              : {06edcfeb-0fd0-4e53-acca-a6f8bbf81bcb}
Event message files     : %SystemRoot%\system32\w32time.dll
```

Is the following EvenLog provider:

```
Name                       : Microsoft-Windows-Time-Service
Identifier              : {06edcfeb-0fd0-4e53-acca-a6f8bbf81bcb}
Log type                : System
Log source(s)           : Microsoft-Windows-Time-Service
                        : W32Time
Event message files     : %SystemRoot%\system32\w32time.dll
```

Note that an EventLog provider can have multiple log types and log sources. It is not known if a log source that matches the EventLog provider name can be deduplicated.

Or as specified as Event XML:

```
<Provider Name='Microsoft-Windows-Time-Service'
          Guid='{06edcfeb-0fd0-4e53-acca-a6f8bbf81bcb}'
          EventSourceName='W32Time'/>
```

5.2 Services\EventLog key

The event sources are stored in the Services\EventLog key:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\EventLog\
```

On Windows NT it can be found in the SYSTEM Registry file.

The Services\EventLog key contains a per EventLog type sub key, for example for the "System" EventLog type:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\EventLog\System\
```

Common EventLog types are:

- Application
- Security
- System

The EventLog type sub key contains a per EventLog source-per-type sub key, for example for the "Workstation" EventLog source:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\EventLog\System\Workstation\
```

Note that the log source is case insensitive; so "Workstation" and "workstation" are considered equivalent.

5.2.1 Services\EventLog type sub key

Values:

Services\EventLog source-per-type sub key

The Services\EventLog source-per-type sub key contains information about a single event source.

Values:

TypesSupported value data

5.3 WINEVT\Publishers key

The event publishers (or providers) are stored in the WINEVT\Publishers key:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\WINEVT\Publishers
```

On Windows Vista or later it can be found in the SOFTWARE Registry file.

The WINEVT\Publishers key contains a GUID type sub key, for example "{de513a55-c345-438b-9a74-e18cac5c5cc5}":

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\WINEVT\Publishers\%GUID%
```

5.3.1 WINEVT\Publishers GUID sub key

A WINEVT\Publishers GUID sub key contains information about a single event publisher.

Values:

5.4 Message file paths

A message file path can be defined in numerous different ways for example:

As an absolute path

```
C:\Windows\System32\mscoree.dll
```

As a relative path:

```
mscoree.dll
```

As a path using environment variables:

```
%SystemDrive%\Windows\System32\mscoree.dll
%SystemRoot%\System32\mscoree.dll
%WinDir%\System32\mscoree.dll
```

As a path using universal OEM runtime macros:

```
$(runtime.system32)\mscoree.dll
```

```
\SystemRoot\system32\mscoree.dll
```

5.5 EventLog provider with multiple provider GUIDs

Seen on Windows 8.0, 8.1, 10, 11 and 2012:

```
Key path: HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\EventLog\Application\
↪Microsoft-Windows-KdsSvc
Name: Microsoft-Windows-KdsSvc
Last written time: Oct 30, 2015 07:25:12.126588100 UTC

Value: 0 providerGuid
Type: string (REG_SZ)
Data size: 78
Data: {d4be7726-dc7a-11df-a6e6-0902dfd72085}
```

```
Key path: HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\WINEVT\Publishers\
↪{89203471-d554-47d4-bde4-7552ec219999}
Name: {89203471-d554-47d4-bde4-7552ec219999}
Last written time: Oct 30, 2015 07:25:53.860831900 UTC

Value: 0 (default)
Type: string (REG_SZ)
Data size: 50
Data: Microsoft-Windows-KdsSvc

Value: 1 ResourceFileName
Type: expandable string (REG_EXPAND_SZ)
Data size: 66
Data: %SystemRoot%\system32\KdsCli.dll

Value: 2 MessageFileName
Type: expandable string (REG_EXPAND_SZ)
Data size: 66
Data: %SystemRoot%\system32\KdsCli.dll
```

5.6 EventLog provider with multiple log types

Seen on Windows 10:

```
Key path: HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\EventLog\Application\
↪Microsoft-Windows-EventCollector
Name: Microsoft-Windows-EventCollector
Last written time: Sep 13, 2014 07:27:56.080450600 UTC

Value: 0 ProviderGuid
Type: string (REG_SZ)
Data size: 78
Data: {b977cf02-76f6-df84-cc1a-6a4b232322b6}

Value: 1 EventMessageFile
Type: expandable string (REG_EXPAND_SZ)
Data size: 66
Data: %SystemRoot%\system32\wecsvc.dll
```

```
Key path: HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\EventLog\System\Microsoft-
↪Windows-EventCollector
Name: Microsoft-Windows-EventCollector
Last written time: Sep 13, 2014 07:27:56.080450600 UTC

Value: 0 ProviderGuid
Type: string (REG_SZ)
Data size: 78
Data: {b977cf02-76f6-df84-cc1a-6a4b232322b6}

Value: 1 EventMessageFile
```

(continues on next page)

(continued from previous page)

```
Type: expandable string (REG_EXPAND_SZ)
Data size: 66
Data: %SystemRoot%\system32\wecsvc.dll
```

```
Key path:  HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\WINEVT\
↪Publishers\{b977cf02-76f6-df84-cc1a-6a4b232322b6}
Name: {b977cf02-76f6-df84-cc1a-6a4b232322b6}
Last written time: Sep 13, 2014 07:27:56.080450600 UTC

Value: 0 (default)
Type: string (REG_SZ)
Data size: 66
Data: Microsoft-Windows-EventCollector

Value: 1 ResourceFileName
Type: expandable string (REG_EXPAND_SZ)
Data size: 66
Data: %SystemRoot%\system32\wecsvc.dll

Value: 2 MessageFileName
Type: expandable string (REG_EXPAND_SZ)
Data size: 66
Data: %SystemRoot%\system32\wecsvc.dll
```

5.7 External Links

- Eventlog Key
- Event Sources
- winevt.h header
- Windows Event Log

CHAPTER

SIX

INTERNET EXPLORER KEYS

6.1 Browser helper objects (BHO)

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper␣
↪Objects
```

6.2 Policies

The Internet Explorer polices are stored in multiple keys.

Order of application:

1. HKEY_LOCAL_MACHINE policy key (Administrative override)
2. HKEY_CURRENT_USER policy key
3. HKEY_CURRENT_USER preference key
4. HKEY_LOCAL_MACHINE preference key (System default settings)

Note that the location of the HKEY_LOCAL_MACHINE policy and preference key is dependent on the usage of WoW64 (Windows 32-bit on Windows 64-bit).

Normal:

1. HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Internet Explorer\Main\FeatureControl
2. HKEY_CURRENT_USER\Software\Policies\Microsoft\Internet Explorer\Main\FeatureControl
3. HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\Main\FeatureControl
4. HKEY_LOCAL_MACHINE\Software\Microsoft\Internet Explorer\Main\FeatureControl

WoW64:

1. HKEY_LOCAL_MACHINE\Wow6432Node\Software\Policies\Microsoft\Internet Explorer\Main\FeatureControl
2. HKEY_CURRENT_USER\Software\Policies\Microsoft\Internet Explorer\Main\FeatureControl
3. HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\Main\FeatureControl
4. HKEY_LOCAL_MACHINE\Wow6432Node\Software\Microsoft\Internet Explorer\Main\FeatureControl

6.2.1 Policies

```
HKEY_CURRENT_USER\Software\Policies\Microsoft\Internet Explorer
```

Values:

Download policies

```
HKEY_CURRENT_USER\Software\Policies\Microsoft\Internet Explorer\Download
```

Values:

6.2.2 Feature controls

Security Zones

Also stored in "Description" Registry value in zone-specific Registry key.

Local Machine Zone Lockdown

Applies the Lockdown Zones instead of the Zones.

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Main\FeatureControl\FEATURE_
↪LOCALMACHINE_LOCKDOWN\

HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Internet Explorer\Main\FeatureControl\
↪FEATURE_LOCALMACHINE_LOCKDOWN\

HKEY_CURRENT_USER\SOFTWARE\Microsoft\Internet Explorer\Main\FeatureControl\FEATURE_
↪LOCALMACHINE_LOCKDOWN\
```

Add a REG_DWORD value to this key named for your application (for example, MyApplication.exe) and set it to 1. Any other setting for this value will disable Local Machine Zone Lockdown for the application.

Network Protocol Lockdown

```
HKEY_LOCAL_MACHINE\Software\(Policies)\Microsoft\Internet Explorer\Main\FeatureControl\
↪FEATURE_PROTOCOL_LOCKDOWN

HKEY_CURRENT_USER\Software\(Policies)\Microsoft\Internet Explorer\Main\FeatureControl\
↪FEATURE_PROTOCOL_LOCKDOWN

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Main\FeatureControl\FEATURE_
↪PROTOCOL_LOCKDOWN
```

HTML from CD

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Main\FeatureControl\FEATURE_
↪LOCALMACHINE_LOCKDOWN\Settings\LOCALMACHINE_CD_UNLOCK
```

6.2.3 Notes

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Low Rights\ElevationPolicy
HKEY_CURRENT_USER\SOFTWARE\Microsoft\Internet Explorer\Low Rights\ElevationPolicy
```

6.2.4 External Links

- About URL Security Zones
- Internet Explorer Local Machine Zone Lockdown
- Internet Explorer Network Protocol Lockdown
- Internet Explorer Protected Mode Elevation Policy and Administrative Templates
- Internet Explorer security zones registry entries for advanced users
- Internet Feature Controls
- Introduction to Feature Controls
- Understanding and Working in Protected Mode Internet Explorer
- Understanding user-agent strings

CHAPTER

SEVEN

APPLICATION KEYS

7.1 7-Zip

TODO this page currently contains rough notes, fine tune these

The 7-Zip application uses the following Windows Registry key to store various user specific information.

```
HKEY_CURRENT_USER\Software\7-Zip
```

Sub keys:

Values:

7.1.1 7-Zip FM sub key

Sub keys:

Values:

7-Zip FM Columns sub key

Values:

7.2 Microsoft Office

TODO this page currently contains rough notes, fine tune these

7.2.1 Microsoft Outlook keys

```
HKEY_CURRENT_USER\Software\Microsoft\Office\%VERSION%\Outlook\
```

Where %VERSION% corresponds to:

Values:

```
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\Outlook\Catalog
```

```
HKEY_CURRENT_USER\Software\Microsoft\Office\14.0\Outlook\Search
HKEY_CURRENT_USER\Software\Microsoft\Office\14.0\Outlook\Search\Catalog
```

```
HKEY_CURRENT_USER\Software\Microsoft\Office\15.0\Outlook\Search
```

Values:

Where %FILENAME% is the full filename of Outlook file.

```
HKEY_CURRENT_USER\Software\Microsoft\Office\15.0\Outlook\Search\Catalog
```

Values:

Where %FILENAME% is the full filename of Outlook file.

For "file/options/search" allow to search through "deleted items"

```
Key: HKCU\Software\Microsoft\Office\14.0\Outlook\Search
Value: IncludeDeletedItems
```

Where the value data is:

```
1 = Yes
```

Outlook generates log files in %temp%\Outlook logging:

```
Key: HKEY_CURRENT_USER\Software\Microsoft\Office\version number\Outlook\Search
Value: EnableLogging = 0xffff0000
```

Protected View mode

```
Key: HKEY_CURRENT_USER\Software\Microsoft\Office\14.0\Outlook\Security
Key: HKEY_CURRENT_USER\Software\Policies\Microsoft\Office\14.0\Outlook\Security
Value: MarkInternalAsUnsafe
```

Where the value data is:

```
1 = Yes
```

Offline Address Book (OAB)

```
HKEY_CURRENT_USER\Software\Microsoft\Office\%VERSION%\Outlook\Cached Mode
```

Values:

Setting the value to zero prevents Offline Address Book (OAB) download and forces Outlook to use the global address list. If the "Cached Mode" key does not exist, create it.

Secure Temp Folder

```
HKEY_CURRENT_USER\Software\Microsoft\Office\%VERSION%\Outlook\Security
```

Values:

7.2.2 Most Recently Used (MRU) keys

File Name MRU keys

Values:

```
HKEY_CURRENT_USER\Software\Microsoft\Office\11.0\Common\Open Find\Microsoft Office Excel\
↪Settings\Open\File Name MRU
HKEY_CURRENT_USER\Software\Microsoft\Office\11.0\Common\Open Find\Microsoft Office Excel\
↪Settings\Save As\File Name MRU
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\Common\Open Find\Microsoft Office Excel\
↪Settings\Open\File Name MRU
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\Common\Open Find\Microsoft Office Excel\
↪Settings\Save As\File Name MRU
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\Common\Open Find\Microsoft Office Word\
↪Settings\Save As\File Name MRU
```

Item MRU keys

Values:

```
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\Excel\File MRU
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\Excel\Options\MRUFuncs
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\PowerPoint\File MRU
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\PowerPoint\RecentAnimationList\
↪EntranceMRU
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\PowerPoint\RecentAnimationList\
↪EmphasisMRU
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\PowerPoint\RecentAnimationList\ExitMRU
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\PowerPoint\RecentAnimationList\
↪MotionPathMRU
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\PowerPoint\Slide Libraries\Taskpane MRU
HKEY_CURRENT_USER\Software\Microsoft\Office\12.0\Word\File MRU
```

Every %ITEM% value contains:

in Office 12

```
[F00000000][T%FILETIME%]*\\%FILENAME%
```

in Office 14

```
[F00000000][T%FILETIME%][O00000000]*%FILENAME%
```

Where T%FILETIME% contains a FILETIME timestamp as a hexadecimal string (base-16), in upper case, e.g. T01CD10EC460129A0

Other MRU keys

Find Contact toolbar button (Outlook 97 – 2003)

```
HKEY_CURRENT_USER\Software\Microsoft\Office\%VERSION%\Outlook\Contact\QuickFindMRU\
↪QuickfindMRU
```

Find Pane's Look in list

```
HKEY_CURRENT_USER\Software\Microsoft\Office\%VERSION%\Outlook\Contact\StripSearchMRU\
↪StripSearchMRU
```

Appointment Locations

```
HKEY_CURRENT_USER\Software\Microsoft\Office\%VERSION%\Outlook\Preferences\LocationMRU
```

Advanced Find's Search for Word(s)

```
HKEY_CURRENT_USER\Software\Microsoft\Office\%VERSION%\Outlook\Office Finder\MRU 1
```

Advanced Find's More Choices, Categories

```
HKEY_CURRENT_USER\Software\Microsoft\Office\%VERSION%\Outlook\Office Finder\MRU 3
```

Choose Form dialog (New Item-> More Items-> Choose Form) (Outlook 2010 – 2013)

```
HKEY_CURRENT_USER\Software\Microsoft\Office\%VERSION%\Outlook\Office
```

7.2.3 External links

- Administering the offline address book in Outlook

7.3 Terminal server client

The most recent used (MRU) connnections of the Terminal server client can be found in the key:

```
HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Default
```

Values:

The contents of MRU# is either an IP address, e.g. 192.168.16.60, or a hostname, e.g. computer.domain.com.

7.3.1 External Links

- How to Remove Entries from the Remote Desktop Connection Computer Box

7.4 WinRAR

TODO this page currently contains rough notes, fine tune these

The WinRAR application uses the following Windows Registry key to store various user specific information.

```
HKEY_CURRENT_USER\Software\WinRAR\ArcHistory
HKEY_CURRENT_USER\Software\WinRAR\DialogEditHistory\ArcName
HKEY_CURRENT_USER\Software\WinRAR\DialogEditHistory\ExtrPath
```

Values:

CHAPTER

EIGHT

WINREGRC PACKAGE

8.1 Submodules

8.2 winregrc.appcompatcache module

Application Compatibility Cache collector.

class winregrc.appcompatcache.**AppCompatCacheCachedEntry**

Bases: object

Application Compatibility Cache cached entry.

cached_entry_size

size of the cached entry.

Type
int

data

data of the cached entry.

Type
bytes

file_size

size of file corresponding to the cached entry.

Type
int

insertion_flags

insertion flags of the cached entry.

Type
int

last_modification_time

last modification timestamp of the file corresponding to the cached entry.

Type
int

last_update_time

last update timestamp the cached entry.

Type
int

shim_flags

shim flags of the cached entry.

Type
int

path

path of the cached entry.

Type
str

class winregrc.appcompatcache.**AppCompatCacheCollector**(*debug=False*, *output_writer=None*)

Bases: *WindowsRegistryKeyCollector*

Application Compatibility Cache collector.

cached_entries

cached entries.

Type
list[*AppCompatCacheCachedEntry*]

Collect(*registry*, *all_control_sets=False*)

Collects the Application Compatibility Cache.

Parameters

- **registry** (*dfwinreg.WinRegistry*) – Windows Registry.
- **all_control_sets** (*Optional[bool]*) – True if the services should be collected from all control sets instead of only the current control set.

Returns

True if the Application Compatibility Cache key was found,
False if not.

Return type
bool

class winregrc.appcompatcache.**AppCompatCacheDataParser**(*debug=False*, *output_writer=None*)

Bases: *BinaryDataFormat*

Application Compatibility Cache data parser.

CheckSignature(*value_data*)

Parses and validates the signature.

Parameters
value_data (*bytes*) – value data.

Returns
format type or None if format could not be determined.

Return type
int

Raises
ParseError – if the value data could not be parsed.

ParseCachedEntry(*format_type*, *value_data*, *cached_entry_index*, *cached_entry_offset*)

Parses a cached entry.

Parameters

- **format_type** (*int*) – format type.
- **value_data** (*bytes*) – value data.
- **cached_entry_index** (*int*) – cached entry index.
- **cached_entry_offset** (*int*) – offset of the first cached entry data relative to the start of the value data.

Returns
cached entry.

Return type
AppCompatCacheCachedEntry

Raises
ParseError – if the value data could not be parsed.

ParseHeader(*format_type*, *value_data*)

Parses the header.

Parameters

- **format_type** (*int*) – format type.
- **value_data** (*bytes*) – value data.

Returns
header.

Return type
AppCompatCacheHeader

Raises
ParseError – if the value data could not be parsed.

class winregrc.appcompatcache.**AppCompatCacheHeader**

Bases: object

Application Compatibility Cache header.

number_of_cached_entries

number of cached entries.

Type
int

header_size

header size.

Type
int

8.3 winregrc.application_identifiers module

Windows application identifiers (AppID) collector.

class winregrc.application_identifiers.**ApplicationIdentifier**(*guid*, *description*)

Bases: object

Application identifier.

description

description.

Type
str

guid

identifier.

Type
str

class winregrc.application_identifiers.**ApplicationIdentifiersCollector**(*debug=False*)

Bases: *WindowsRegistryKeyCollector*

Windows application identifiers collector.

Collect(*registry*)

Collects Windows application identifiers (AppID).

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Yields
ApplicationIdentifier – an application identifier.

8.4 winregrc.cached_credentials module

Domain cached credentials collector.

class winregrc.cached_credentials.**CachedCredentialsKeyCollector**(*debug=False*, *output_writer=None*)

Bases: *WindowsRegistryKeyCollector*

Domain cached credentials key collector.

Collect(*registry*)

Collects system information.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Returns
True if the system key was found, False if not.

Return type
bool

8.5 winregrc.catalog module

Catalog collector.

class winregrc.catalog.**CatalogCollector**(*group_keys=False*)

Bases: object

Catalog collector.

Collect(*root_key*)

Collects the catalog descriptors from a Windows Registry file.

Parameters
root_key (*dfwinreg.WinRegistryKey*) – root Windows Registry key.

Yields
CatalogKeyDescriptor – catalog key descriptor.

class winregrc.catalog.**CatalogKeyDescriptor**

Bases: object

Catalog key descriptor.

grouped_key_paths

paths of Windows Registry keys with similar values.

Type
list[str]

key_path

path of Windows Registry key.

Type
str

value_descriptors

pairs of value name and data type.

Type
tuple[str,str]

8.6 winregrc.data_format module

Binary data format.

class winregrc.data_format.**BinaryDataFormat**(*debug=False*, *output_writer=None*)

Bases: object

Binary data format.

8.7 winregrc.environment_variables module

Environment variables collector.

class winregrc.environment_variables.**EnvironmentVariable**(*name*, *value*)

Bases: object

Environment variable.

name

name.

Type
str

value

value.

Type
str

class winregrc.environment_variables.**EnvironmentVariablesCollector**(*debug=False*)

Bases: *WindowsRegistryKeyCollector*

Environment variables collector.

Collect(*registry*)

Collects environment variables.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Yields
EnvironmentVariable – an environment variable.

8.8 winregrc.errors module

The error objects.

exception winregrc.errors.**Error**

Bases: Exception

The error interface.

exception winregrc.errors.**ParseError**

Bases: *Error*

Error that is raised when value data cannot be parsed.

8.9 winregrc.eventlog_providers module

Windows Event Log providers collector.

class winregrc.eventlog_providers.**EventLogProvider**

Bases: object

Windows Event Log provider.

additional_identifier

additional identifier of the provider, contains a GUID.

Type
str

category_message_files

paths of the category message files.

Type
set[str]

event_message_files

paths of the event message files.

Type
set[str]

identifier

identifier of the provider, contains a GUID.

Type
str

log_sources

names of the corresponding Event Log sources.

Type
list[str]

log_types

Windows Event Log types.

Type
list[str]

name

name of the provider.

Type
str

parameter_message_files

paths of the parameter message files.

Type
set[str]

class winregrc.eventlog_providers.**EventLogProvidersCollector**(*debug=False*)

Bases: *WindowsRegistryKeyCollector*

Windows Event Log providers collector.

Collect(*registry*)

Collects Windows Event Log providers from a Windows Registry.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Returns
Event Log provider generator.

Return type
generator[*EventLogProvider*]

8.10 winregrc.filters module

The Windows Registry key and value filters.

class winregrc.filters.**BaseWindowsRegistryKeyFilter**

Bases: object

Windows Registry key filter interface.

abstract Match(*registry_key*)

Determines if a Windows Registry key matches the filter.

Parameters
registry_key (*dfwinreg.WinRegistryKey*) – a Windows Registry key.

Returns
True if a match, False otherwise.

Return type
bool

property key_paths

List of key paths defined by the filter.

class winregrc.filters.**WindowsRegistryKeyPathFilter**(*key_path*)

Bases: *BaseWindowsRegistryKeyFilter*

Windows Registry key path filter.

Match(*registry_key*)

Determines if a Windows Registry key matches the filter.

Parameters
registry_key (*dfwinreg.WinRegistryKey*) – a Windows Registry key.

Returns
True if a match, False otherwise.

Return type
bool

property key_paths

key paths defined by the filter.

Type
list[str]

class winregrc.filters.**WindowsRegistryKeyPathPrefixFilter**(*key_path_prefix*)

Bases: *BaseWindowsRegistryKeyFilter*

Windows Registry key path prefix filter.

Match(*registry_key*)

Determines if a Windows Registry key matches the filter.

Parameters
registry_key (*dfwinreg.WinRegistryKey*) – a Windows Registry key.

Returns
True if a match, False otherwise.

Return type
bool

class winregrc.filters.**WindowsRegistryKeyPathSuffixFilter**(*key_path_suffix*)

Bases: *BaseWindowsRegistryKeyFilter*

Windows Registry key path suffix filter.

Match(*registry_key*)

Determines if a Windows Registry key matches the filter.

Parameters
registry_key (*dfwinreg.WinRegistryKey*) – a Windows Registry key.

Returns
True if a match, False otherwise.

Return type
bool

class winregrc.filters.**WindowsRegistryKeyWithValuesFilter**(*value_names*)

Bases: *BaseWindowsRegistryKeyFilter*

Windows Registry key with values filter.

Match(*registry_key*)

Determines if a Windows Registry key matches the filter.

Parameters
registry_key (*dfwinreg.WinRegistryKey*) – a Windows Registry key.

Returns
True if a match, False otherwise.

Return type
bool

8.11 winregrc.hexdump module

Function to provide hexadecimal representation of data.

winregrc.hexdump.**Hexdump**(*data*)

Formats data in a hexadecimal representation.

Parameters
data (*byte*) – data.

Returns
hexadecimal representation of the data.

Return type
str

8.12 winregrc.interface module

Windows Registry key and value collector.

class winregrc.interface.**WindowsRegistryKeyCollector**(*debug=False*)

Bases: object

Windows Registry key and value collector.

8.13 winregrc.knownfolders module

Windows known folders collector.

class winregrc.knownfolders.**KnownFolder**(*guid*, *name*, *localized_name*)

Bases: object

Known folder.

guid

identifier.

Type
str

localized_name

localized name.

Type
str

name

name.

Type
str

class winregrc.knownfolders.**KnownFoldersCollector**(*debug=False*)

Bases: *WindowsRegistryKeyCollector*

Windows known folders collector.

Collect(*registry*)

Collects Windows known folders.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Yields
KnownFolder – a known folder.

8.14 winregrc.mounted_devices module

Windows mounted devices collector.

class winregrc.mounted_devices.**MountedDevice**(*identifier*)

Bases: object

Mounted device.

device

device.

Type
str

disk_identity

MBR disk identity.

Type
int

identifier

identifier.

Type
str

partition_identifier

GPT partition identifier.

Type
str

partition_offset

MBR partition offset.

Type
int

class winregrc.mounted_devices.**MountedDevicesCollector**(*debug=False*, *output_writer=None*)

Bases: *BinaryDataFormat*

Windows mounted devices collector.

Collect(*registry*)

Collects Windows mounted devices.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Yields
MountedDevice – a mounted device.

Raises
ParseError – if a mounted devices value could not be parsed.

8.15 winregrc.mru module

Most Recently Used (MRU) collector.

class winregrc.mru.**MostRecentlyUsedCollector**(*debug=False*, *output_writer=None*)

Bases: *BinaryDataFormat*

Most Recently Used (MRU) collector.

mru_entries

most recently used (MRU) entries.

Type
list[*MostRecentlyUsedEntry*]

Collect(*registry*)

Collects Most Recently Used (MRU) entries.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Returns
True if a Most Recently Used (MRU) key was found, False if not.

Return type
bool

class winregrc.mru.**MostRecentlyUsedEntry**(*key_path=None*, *shell_item_data=None*, *shell_item_list_data=None*, *string=None*, *value_name=None*)

Bases: object

Most Recently Used (MRU) entry.

key_path

path of the Windows Registry key.

Type
str

shell_item_data

Shell Item data.

Type
bytes

shell_item_list_data

Shell Item list data.

Type
bytes

string

string.

Type
str

value_name

name of the Windows Registry value.

Type
str

8.16 winregrc.msie_zone_info module

Microsoft Internet Explorer (MSIE) zone information collector.

class winregrc.msie_zone_info.**MSIEZoneInformation**(*zone*, *zone_name*, *control*, *control_value*)

Bases: object

MSIE zone information.

control

control.

Type
str

control_value

value to which the control is set.

Type
int|str

zone

identifier of the zone to which the control applies.

Type
str

zone_name

name of the zone to which the control applies.

Type
str

class winregrc.msie_zone_info.**MSIEZoneInformationCollector**(*debug=False*)

Bases: *WindowsRegistryKeyCollector*

MSIE zone information collector.

Collect(*registry*)

Collects the MSIE zone information.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Yields
MSIEZoneInformation – MSIE zone information.

8.17 winregrc.output_writers module

Output writer.

class winregrc.output_writers.**OutputWriter**

Bases: object

Output writer interface.

abstract Close()

Closes the output writer.

DebugPrintData(*description*, *data*)

Prints data for debugging.

Parameters

- **description** (*str*) – description.
- **data** (*bytes*) – data.

DebugPrintText(*text*)

Prints text for debugging.

Parameters
text (*str*) – text.

DebugPrintValue(*description*, *value*)

Prints a value for debugging.

Parameters

- **description** (*str*) – description.
- **value** (*object*) – value.

abstract Open()

Opens the output writer.

Returns
True if successful or False if not.

Return type
bool

abstract WriteDebugData(*description*, *data*)

Writes data for debugging.

Parameters

- **description** (*str*) – description to write.
- **data** (*bytes*) – data to write.

abstract WriteFiletimeValue(*description*, *value*)

Writes a FILETIME timestamp value.

Parameters

- **description** (*str*) – description to write.
- **value** (*str*) – value to write.

abstract WriteIntegerValueAsDecimal(*description*, *value*)

Writes an integer value as decimal.

Parameters

- **description** (*str*) – description to write.
- **value** (*int*) – value to write.

abstract WriteText(*text*)

Writes text.

Parameters
text (*str*) – text to write.

abstract WriteValue(*description*, *value*)

Writes a value.

Parameters

- **description** (*str*) – description to write.
- **value** (*str*) – value to write.

class winregrc.output_writers.**StdoutOutputWriter**

Bases: *OutputWriter*

Stdout output writer.

Close()

Closes the output writer.

Open()

Opens the output writer.

Returns
True if successful or False if not.

Return type
bool

WriteDebugData(*description*, *data*)

Writes data for debugging.

Parameters

- **description** (*str*) – description to write.
- **data** (*bytes*) – data to write.

WriteFiletimeValue(*description*, *value*)

Writes a FILETIME timestamp value.

Parameters

- **description** (*str*) – description to write.
- **value** (*str*) – value to write.

WriteIntegerValueAsDecimal(*description*, *value*)

Writes an integer value as decimal.

Parameters

- **description** (*str*) – description to write.

- **value** (*int*) – value to write.

WriteText(*text*)

Writes text.

Parameters
text (*str*) – text to write.

WriteValue(*description*, *value*)

Writes a value.

Parameters

- **description** (*str*) – description to write.
- **value** (*object*) – value to write.

8.18 winregrc.profiles module

Windows user profiles collector.

class winregrc.profiles.**UserProfile**(*security_identifier*, *profile_path*)

Bases: object

User profile.

profile_path

path of the users profile.

Type
str

security_identifier

security identifier of the user.

Type
str

class winregrc.profiles.**UserProfilesCollector**(*debug=False*)

Bases: *WindowsRegistryKeyCollector*

Windows user profiles collector.

Collect(*registry*)

Collects user profiles.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Yields
UserProfile – an user profile.

8.19 winregrc.programscache module

Windows Programs Cache information collector.

class winregrc.programscache.**ProgramsCacheCollector**(*debug=False*, *output_writer=None*)

Bases: *WindowsRegistryKeyCollector*

Windows program cache collector.

Collect(*registry*)

Collects the Programs Cache information.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Returns
True if the Programs Cache information key was found, False if not.

Return type
bool

class winregrc.programscache.**ProgramsCacheDataParser**(*debug=False*, *output_writer=None*)

Bases: *BinaryDataFormat*

Programs Cache data parser.

Parse(*value_data*)

Parses the value data.

Parameters
value_data (*bytes*) – value data.

Raises
ParseError – if the value data could not be parsed.

8.20 winregrc.sam module

Security Accounts Manager (SAM) collector.

class winregrc.sam.**SecurityAccountManagerCollector**(*debug=False*, *output_writer=None*)

Bases: *WindowsRegistryKeyCollector*

Security Accounts Manager (SAM) collector.

user_accounts

user accounts.

Type
list[*UserAccount*]

Collect(*registry*)

Collects the Security Accounts Manager (SAM) information.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Returns

True if the Security Accounts Manager (SAM) information key was found, False if not.

Return type
bool

class winregrc.sam.**SecurityAccountManagerDataParser**(*debug=False*, *output_writer=None*)

Bases: *BinaryDataFormat*

Security Accounts Manager (SAM) data parser.

ParseCValue(*value_data*)

Parses the C value data.

Parameters
value_data (*bytes*) – F value data.

Raises
ParseError – if the value data could not be parsed.

ParseFValue(*value_data*, *user_account*)

Parses the F value data.

Parameters

- **value_data** (*bytes*) – F value data.
- **user_account** (UserAccount) – user account.

Raises
ParseError – if the value data could not be parsed.

ParseVValue(*value_data*, *user_account*)

Parses the V value data.

Parameters

- **value_data** (*bytes*) – V value data.
- **user_account** (UserAccount) – user account.

Raises
ParseError – if the value data could not be parsed.

class winregrc.sam.**UserAccount**

Bases: object

User account.

account_expiration_time

account expiration date and time.

Type
dfdatetime.DateTimeValues

codepage

code page.

Type
str

comment

comment.

Type
str

full_name

full name.

Type
str

last_login_time

last log-in date and time.

Type
dfdatetime.DateTimeValues

last_password_failure_time

last password failure date and time.

Type
dfdatetime.DateTimeValues

last_password_set_time

last password set date and time.

Type
dfdatetime.DateTimeValues

name

name

Type
str

number_of_logons

number of log-ons.

Type
int

number_of_password_failures

number of password failures.

Type
int

primary_gid

primary group identifier (GID).

Type
int

rid

relative identifier (RID).

Type
str

user_account_control_flags

user account control flags.

Type
int

user_comment

user comment.

Type
str

username

username.

Type
str

8.21 winregrc.services module

Windows services and drivers collector.

class winregrc.services.**WindowsService**(*name*, *service_type*, *display_name*, *description*, *image_path*, *object_name*, *start_value*)

Bases: object

Windows service.

description

service description.

Type
str

display_name

display name.

Type
str

image_path

image path.

Type
str

name

name.

Type
str

object_name

object name

Type
str

service_type

service type.

Type
str

start_value

start value.

Type
str

GetObjectNameDescription()

Retrieves the object name description.

Returns
object name description.

Return type
str

GetServiceTypeDescription()

Retrieves the service type description.

Returns
service type description.

Return type
str

GetStartValueDescription()

Retrieves the start value description.

Returns
start value description.

Return type
str

__eq__(*other*)

Determines the current Windows service is equal to the other.

Returns
True if equal.

Return type
bool

__ne__(*other*)

Determines the current Windows service is not equal to the other.

Returns
True if not equal.

Return type
bool

class winregrc.services.**WindowsServicesCollector**(*debug=False*)

Bases: *WindowsRegistryKeyCollector*

Windows services and drivers collector.

Collect(*registry*, *all_control_sets=False*)

Collects Windows services and drivers.

Parameters

- **registry** (*dfwinreg.WinRegistry*) – Windows Registry.

- **all_control_sets** (*Optional[bool]*) – True if the services should be collected from all control sets instead of only the current control set.

Yields
WindowsService – a Windows service.

Compare(*registry*, *output_writer*)

Compares services in the different control sets.

Parameters

- **registry** (*dfwinreg.WinRegistry*) – Windows Registry.
- **output_writer** (OutputWriter) – output writer.

Returns
True if the services key was found, False if not.

Return type
bool

8.22 winregrc.shellfolders module

Shell Folder collector.

class winregrc.shellfolders.**ShellFolder**(*guid*, *name*, *localized_string*)

Bases: object

Shell folder.

guid

GUID.

Type
str

name

name.

Type
str

localized_string

localized string of the name.

Type
str

class winregrc.shellfolders.**ShellFoldersCollector**(*debug=False*)

Bases: *WindowsRegistryKeyCollector*

Shell folder collector.

Collect(*registry*)

Collects the shell folders.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Yields
ShellFolder – a shell folder.

8.23 winregrc.srum_extensions module

System Resource Usage Monitor (SRUM) extension collector.

class winregrc.srum_extensions.**SRUMExtension**(*guid*, *dll_name*)

Bases: object

System Resource Usage Monitor (SRUM) extension.

dll_name

DLL name.

Type
str

guid

identifier.

Type
str

class winregrc.srum_extensions.**SRUMExtensionsCollector**(*debug=False*)

Bases: *WindowsRegistryKeyCollector*

Windows System Resource Usage Monitor (SRUM) extension collector.

Collect(*registry*, *output_writer*)

Collects the SRUM extensions.

Parameters

- **registry** (*dfwinreg.WinRegistry*) – Windows Registry.
- **output_writer** (OutputWriter) – output writer.

Returns
True if the SRUM extensions key was found, False if not.

Return type
bool

8.24 winregrc.sysinfo module

System information collector.

class winregrc.sysinfo.**SystemInfoCollector**(*debug=False*, *output_writer=None*)

Bases: *WindowsRegistryKeyCollector*

System information collector.

system_information

system information.

Type
SystemInformation

Collect(*registry*)

Collects system information.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Returns
True if the system information key was found, False if not.

Return type
bool

class winregrc.sysinfo.SystemInformation

Bases: object

System information.

csd_version

CSD version.

Type
str

current_build_number

current build number.

Type
str

current_type

current type.

Type
str

current_version

current version.

Type
str

installation_date

installation date and time.

Type
dfdatetime.DateTimeValues

path_name

path name.

Type
str

product_identifier

product identifier.

Type
str

product_name

product name.

Type
str

registered_organization

registered organization.

Type
str

registered_owner

registered owner.

Type
str

system_root

system root path.

Type
str

8.25 winregrc.syskey module

System key (syskey) collector.

class winregrc.syskey.**SystemKey**

Bases: object

System key.

boot_key

boot key.

Type
bytes

class winregrc.syskey.**SystemKeyCollector**(*debug=False*, *output_writer=None*)

Bases: *WindowsRegistryKeyCollector*

System key collector.

system_key

system key.

Type
SystemKey

Collect(*registry*)

Collects system information.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Returns
True if the system key was found, False if not.

Return type
bool

8.26 winregrc.task_cache module

Task Cache collector.

class winregrc.task_cache.**CachedTask**

Bases: object

Cached task.

identifier

identifier.

Type
str

last_registered_time

last registered date and time.

Type
dfdatetime.DateTimeValues

launch_time

launch date and time.

Type
dfdatetime.DateTimeValues

name

name.

Type
str

class winregrc.task_cache.**TaskCacheCollector**(*debug=False*, *output_writer=None*)

Bases: *WindowsRegistryKeyCollector*

Task Cache collector.

cached_tasks

cached tasks.

Type
list[*CachedTask*]

Collect(*registry*)

Collects the Task Cache.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Returns
True if the Task Cache key was found, False if not.

Return type
bool

class winregrc.task_cache.**TaskCacheDataParser**(*debug=False*, *output_writer=None*)

Bases: *BinaryDataFormat*

Task Cache data parser.

ParseDynamicInfo(*value_data*, *cached_task*)

Parses the DynamicInfo value data.

Parameters

- **value_data** (*bytes*) – DynamicInfo value data.
- **cached_task** (CachedTask) – cached task.

Raises

ParseError – if the value data could not be parsed.

8.27 winregrc.time_zones module

Windows time zones collector.

class winregrc.time_zones.**TimeZone**(*name*)

Bases: object

Time zone.

localized_name

localized name.

Type

str

name

name.

Type

str

offset

time zone offset in number of minutes from UTC.

Type

int

class winregrc.time_zones.**TimeZoneInformationDataParser**(*debug=False*, *output_writer=None*)

Bases: *BinaryDataFormat*

Time Zone Information (TZI) data parser.

ParseTZIValue(*value_data*, *time_zone*)

Parses the TZI value data.

Parameters

- **value_data** (*bytes*) – TZI value data.
- **time_zone** (TimeZone) – time zone.

Raises

ParseError – if the value data could not be parsed.

class winregrc.time_zones.**TimeZonesCollector**(*debug=False*)

Bases: *WindowsRegistryKeyCollector*

Windows time zones collector.

Collect(*registry*, *output_writer*)

Collects the time zones.

Parameters

- **registry** (*dfwinreg.WinRegistry*) – Windows Registry.
- **output_writer** (OutputWriter) – output writer.

Returns

True if the time zones key was found, False if not.

Return type

bool

8.28 winregrc.type_libraries module

Windows type libraries collector.

class winregrc.type_libraries.**TypeLibrariesCollector**(*debug=False*, *output_writer=None*)

Bases: *WindowsRegistryKeyCollector*

Windows type libraries collector.

type_libraries

type libraries.

Type

list[*TypeLibrary*]

Collect(*registry*)

Collects the type libraries.

Parameters

registry (*dfwinreg.WinRegistry*) – Windows Registry.

Returns

True if the type libraries key was found, False if not.

Return type

bool

class winregrc.type_libraries.**TypeLibrary**(*guid*, *version*, *description*, *typelib_filename*)

Bases: object

Type library.

description

description.

Type

str

guid

identifier.

Type
str

typelib_filename

typelib_filename.

Type
str

version

version.

Type
str

8.29 winregrc.userassist module

Windows UserAssist information collector.

class winregrc.userassist.**UserAssistCollector**(*debug=False*, *output_writer=None*)

Bases: *WindowsRegistryKeyCollector*

Windows UserAssist information collector.

Returns
UserAssist entries.

Return type
user_assist_entries (list[*UserAssistEntry*])

Collect(*registry*)

Collects the UserAssist information.

Parameters
registry (*dfwinreg.WinRegistry*) – Windows Registry.

Returns
True if the UserAssist key was found, False if not.

Return type
bool

class winregrc.userassist.**UserAssistDataParser**(*debug=False*, *output_writer=None*)

Bases: *BinaryDataFormat*

UserAssist data parser.

ParseEntry(*format_version*, *entry_data*)

Parses an UserAssist entry.

Parameters

- **format_version** (*int*) – format version.
- **entry_data** (*bytes*) – entry data.

Returns
UserAssist entry.

Return type
user_assist_entry_v3|user_assist_entry_v5

Raises
ParseError – if the value data could not be parsed.

class winregrc.userassist.**UserAssistEntry**(*guid=None*, *name=None*, *value_name=None*)

Bases: object

UserAssist entry.

guid

GUID.

Type
str

name

name.

Type
str

value_name

name of the Windows Registry value.

Type
str

8.30 winregrc.volume_scanner module

Windows Registry volume scanner.

class winregrc.volume_scanner.**SingleFileWindowsRegistryFileReader**(**args: Any*, ***kwargs: Any*)

Bases: WinRegistryFileReader

Single file Windows Registry file reader.

Open(*path*, *ascii_codepage='cp1252'*)

Opens the Windows Registry file specified by the path.

Parameters

- **path** (*str*) – path of the Windows Registry file. The path is a Windows path relative to the root of the file system that contains the specific Windows Registry file. E.g. C:WindowsSystem32configSYSTEM
- **ascii_codepage** (*Optional[str]*) – ASCII string codepage.

Returns

Windows Registry file or None if the file cannot
be opened.

Return type
WinRegistryFile

class winregrc.volume_scanner.**WindowsRegistryVolumeScanner**(**args: Any*, ***kwargs: Any*)

Bases: WindowsVolumeScanner

Windows Registry volume scanner.

registry

Windows Registry.

Type
dfwinreg.WinRegistry

IsSingleFileRegistry()

Determines if the Registry consists of a single file.

Returns
True if the Registry consists of a single file.

Return type
bool

OpenFile(*windows_path*)

Opens the file specified by the Windows path.

Parameters
windows_path (*str*) – Windows path to the file.

Returns
file-like object or None if the file does not exist.

Return type
dfvfs.FileIO

Raises
ScannerError – if the scan node is invalid or the scanner does not know how to proceed.

ScanForWindowsVolume(*source_path*, *options=None*)

Scans for a Windows volume.

Parameters

- **source_path** (*str*) – source path.
- **options** (*Optional[VolumeScannerOptions]*) – volume scanner options. If None the default volume scanner options are used, which are defined in the VolumeScannerOptions class.

Returns
True if a Windows volume was found.

Return type
bool

Raises
ScannerError – if the source path does not exists, or if the source path is not a file or directory, or if the format of or within the source file is not supported.

class winregrc.volume_scanner.**WindowsRegistryVolumeScannerMediator**(**args: Any*, ***kwargs: Any*)

Bases: CLIVolumeScannerMediator

Windows Registry volume scanner mediator.

PrintUsersSubDirectoriesOverview(*users_file_entry*)

Prints an overview of the Users sub directories.

Parameters
users_file_entry (*dfvfs.FileEntry*) – file entry of the Users directory.

8.31 Module contents

Windows Registry resources (winregrc).

CHAPTER

NINE

INDICES AND TABLES

- genindex
- modindex

PYTHON MODULE INDEX

W

INDEX

Symbols

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

Z