User Guide

MANDIANT Memoryze™

Version 3.0.0

MANDIANT®

INTELLIGENT INFORMATION SECURITY

MANDIANT MEMORYZE™

MANDIANT Memoryze, formerly known as MANDIANT Free Agent, is a memory analysis tool. Memoryze can not only acquire the physical memory from a Windows system but it can also perform advanced analysis of live memory while the computer is running. All analysis can be done either against an acquired image or a live system.

MANDIANT Memoryze Features

MANDIANT Memoryze can:

- image the full range of system memory (not reliant on API calls).
- image a process' entire address space to disk. This includes a process' loaded DLLs, EXEs, heaps, and stacks.
- image a specified driver or all drivers loaded in memory to disk.
- enumerate all running processes (including those hidden by rootkits). For each process, Memoryze can:
    - report all open handles in a process (for example, all files, registry keys, etc.).
    - list the virtual address space of a given process including:
        - displaying all loaded DLLs.
        - displaying all allocated portions of the heap and execution stack.
    - list all network sockets that the process has open, including any hidden by rootkits.
    - specify the functions imported by the EXE and DLLs.
    - specify the functions exported by the EXE and DLLs.
    - hash the EXE and DLL in the process address space. (This is a MemD5 of the binary in memory.)
    - hash the EXE and DLLs in the process address space. (MD5, SHA1, SHA256. This is disk based.)
    - verify the digital signatures of the EXE and DLLs. (This is disk based.)
    - output all strings in memory on a per process base.
- identify all drivers loaded in memory, including those hidden by rootkits. For each driver, Memoryze can:
    - specify the functions the driver imports.
    - specify the functions the driver exports.
    - hash the driver. (MD5, SHA1, SHA256. This is disk based.)
    - verify the digital signature of the driver. (This is disk based.)
    - output all strings in memory on a per driver base.
- report device and driver layering, which can be used to intercept network packets, keystrokes and file activity.
- identify all loaded kernel modules by walking a linked list.
- identify hooks - often used by rootkits - in the System Call Table, the Interrupt Descriptor Tables (IDTs), and driver function tables (IRP tables).

MANDIANT Memoryze can perform all these functions on live system memory or memory image files – whether they were acquired by Memoryze or other memory acquisition tools. However, not all data will be available when working with memory images such as digital signatures and hashes.

Supported Operating Systems

Memoryze officially supports:

- Windows 2000 Service Pack 4 (32-bit)
- Windows XP Service Pack 2 and Service Pack 3 (32-bit)
- Windows Vista Service Pack 1 and Service Pack 2 (32-bit)
- *Windows Vista Service Pack 2 (64-bit)
- Windows 2003 Service Pack 2 (32-bit and 64-bit)
- Windows 7 Service Pack 0 and 1 (32-bit and 64-bit)
- Windows 2008 Service Pack 1 and Service Pack 2 (32-bit)
- Windows 2008 R2 Service Pack 0 (64-bit)
- *Windows 8 Service Pack 0 (32-bit and 64-bit)
- *Windows Server 2012 Service Pack 0 (64-bit)

*means Beta support

Most service packs within a major version of the operating system will work, but the focus was on these.

Installation

Memoryze can run on

1. a forensic workstation when analyzing memory images.
2. the host being analyzed when acquiring memory or analyzing live memory.
3. a USB key for a more forensically friendly acquisition or analysis of a host.

Use the Memoryze MSI to install. If you are running on Vista or later operating system, you will be prompted to elevate privileges during the installation.

When you are installing Memoryze to be used portably (USB key, etc.), you must use special commandline options. At the command prompt type:

msiexec /a MemoryzeSetup.msi /qb TARGETDIR=portable_drive_and_folder

The portable_drive_and_folder should be the drive letter of the USB key and the folder you want to install Memoryze into such as H:\Memoryze.

Now, the first time you run portable Memoryze it will create several files; therefore, you cannot make the media read-only yet.

How to use MANDIANT Memoryze

XML Scripts

Memoryze takes XML documents that define what to do, and Memoryze then outputs the result in XML format. The user can configure the individual parameters within each execution script in order to perform the desired actions.

Several default execution scripts are provided with Memoryze's installation. These scripts include:

- AcquireDriver.Batch.xml
- AcquireMemory.Batch.xml
- AcquireProcessMemory.Batch.xml

- DriverAuditModuleList.Batch.xml
- DriverAuditSignature.Batch.xml
- ProcessAuditMemory.Batch.xml
- HookAudit.Batch.xml

Each script's options will be discussed in depth, with examples.

Batch Files

To make Memoryze easier to use, each XML script has been wrapped by a corresponding batch file. All the parameters in the XML execution script can be modified from the command line using arguments to the batch file. The batch files include:

- MemoryDD.bat to acquire an image of physical memory.
- ProcessDD.bat to acquire an image of the process' address space.
- DriverDD.bat to acquire an image of a driver.
- Process.bat to enumerate everything about a process including handles, virtual memory, network ports, and strings.
- HookDetection.bat to look for hooks within the operating system.
- DriverSearch.bat to find drivers.
- DriverWalkList.bat to enumerate all modules and drivers in a linked list.

Viewing the Results

Memoryze creates XML documents containing the analysis results. Currently, Memoryze does not provide a built-in viewer for its results. However, result files can be displayed in any XML viewer – such as Windows Internet Explorer, Mozilla Firefox, or even Microsoft Excel 2007. Be careful! Some XML viewers can be sluggish when loading large XML documents.

Memoryze also comes embedded in Redline – MANDIANT's UI that accelerates the process of triaging hosts suspected of being compromised or infected while supporting in-depth live memory analysis. Click here to find the latest version - https://www.mandiant.com/resources/download/redline

Executing Memoryze

There are three ways to use Memoryze.

One way is to use the XML command files native to Memoryze.exe. This requires editing the *.Batch.xml files to configure Memoryze to perform the desired tasks.

The other option is to use the command-line batch scripts provided. These batch scripts generate the XML command files for the desired audit using the options specified on the batch file command line. Using the batch scripts eliminates the need to edit an XML file. These batch scripts are convenient for interactive use.

The final and preferred way to launch Memoryze is to use the user interfaces built for Memoryze called Redline.

Using Memoryze with the XML Execution Scripts

Memoryze.exe is the executable that takes the command line parameters and executes the XML audit or script. Memoryze command line parameters are as follows:

- -o [directory]
  - The optional directory argument specifies the location to store the results. If this location is not specified, the results are stored by default in <the current working directory>/Audits/<machine>/<date>. <machine> is the name of the system on which Memoryze is executing, and <date> is a date/time stamp in the format of YYYYMMDDHHMMSS.
- -script <XML script to execute>
  - Executes the specified audit (*.Batch.xml)
- -encoding [none|aff|gzip]
  - none – no encoding of the output
  - aff – compresses the output in an AFF evidence container
  - gzip – compresses the output in GZIP

Using the Batch Scripts

MemoryDD.bat executes AcquireMemory.Batch.xml. It creates a memory image.

- –offset – offset into physical memory. Omit the –offset option to acquire all memory.
- –size – size of physical memory to acquire. Omit the –size option to acquire all memory.
- –output – directory in which to write results. Defaults to ./Audits

ProcessDD.bat executes AcquireProcessMemory.Batch.xml. It acquires a specified process' address space including the stack, the heap, DLLs, EXEs, and NLSs files.

- –input – name of image to parse (omit –input for live memory).
- –pid – PID of the process to acquire. Required without process name.
- –process – name of the process to acquire. Required without PID.
- –content - only acquire processes that contain a particular regex content. (Default: NULL)
- –output – directory in which to write results. Defaults to ./Audits

DriverDD.bat executes AcquireDriver.Batch.xml. It acquires either a specified driver in memory, or all drivers.

- –input – name of image to parse (omit –input for live memory).
- –driver – name of driver to acquire (if not specified all drivers are acquired).
- –output – directory in which to write results. Defaults to ./Audits

Process.bat executes ProcessAuditMemory.Batch.xml. It gathers information, such as open ports, files, keys, memory sections, and strings, on a given process or all processes.

- –input – name of image to parse (omit –input for live memory).
- –pid – PID of the process to inspect. Default: 4294967295 which is equivalent to all PIDs.
- –process – optional name of the process to inspect. (Default: excluded)
- –handles – true | false inspect all the process handles. (Default: false)
- –sections – true | false inspect all process memory ranges. (Default: false)
- –ports – true | false inspect all the ports of a process. (Default: false)
- –imports – true | false enumerate the EXE' and DLLs' imports. (Default: false)
- –exports – true | false enumerate the EXE' and DLLs' exports. (Default: false)
- –MenD5 – true | false hash the EXE and DLLs in memory. (Default: false)
  For a more detailed description of how MemD5 works, please see https://media.blackhat.com/bh-us-11/Butler/BH_US_11_ButlerMurdock_Physical_Memory_Forensics-WP.pdf
- –MD5 – true | false hash the EXE and DLLs on disk. (Default: false)
- –SHA1 – true | false hash the EXE and DLLs on disk. (Default: false)
- –SHA256 – true | false hash the EXE and DLLs on disk. (Default: false)
- –digsig – true | false verify if the EXE and DLLs are signed on disk. (Default: false)
- –strings – true | false inspect all the strings of a process. (Default: false)
- – content - only return processes with a particular regex content. (Default: NULL)
- –output – directory in which to write results. Defaults to ./Audits

DriverWalkList.bat executes DriverAuditModuleList.Batch.xml. It enumerates a linked list in the kernel called PsLoadedModuleList.

- –input – name of image to parse (omit –input for live memory).
- –output – directory in which to write results. Defaults to ./Audits

DriverSearch.bat executes DriverAuditSignature.Batch.xml. It finds all loaded drivers using a signature.

- –input – name of image to parse (omit –input for live memory).
- –imports – true | false enumerate the driver’s imports. (Default: false)
- –exports – true | false enumerate the driver’s exports. (Default: false)
- –MD5 – true | false hash the driver on disk. (Default: false)
- –SHA1 – true | false hash the driver on disk. (Default: false)
- –SHA256 – true | false hash the driver on disk. (Default: false)
- –digsig – true | false verify if the driver is signed on disk. (Default: false)
- –strings – true | false inspect all the strings of a process. (Default: false)
- –output – directory in which to write results. Defaults to ./Audits

HookDetection.bat executes HookAudit.Batch.xml. It identifies hooks in kernel memory often used to subvert the integrity of the system.

- –input – name of image to parse (omit –input for live memory).
- –idt – true | false verify certain Interrupt Descriptor Table entries. (Default: true)
- –ssdt – true | false verify the System Call Table. (Default: true)
- –functions – true | false verify System Call Table functions. (Default: true)
- –drivers – true | false verify all drivers’ IRP tables. (Default: true)
- –output – directory in which to write results. Defaults to ./Audits

MemoryDD.bat and AcquireMemory.Batch.xml

Description: Acquires a copy of physical memory from the target system.

Parameter	Required	Description	Data Type
offset	No	Specifies the offset in bytes from the beginning of physical memory. Note: This will be rounded to a page boundary.	64-bit Integer
size	No	Specifies the size of memory to return. Note: This will be rounded to the next page boundary.	64-bit Integer

Acquire all of the system's memory

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32memory-acquisition" version="1.4.0.0" />
    </command>
  </commands>
</script>
```

Acquire a portion of the system's memory starting from offset

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32memory-acquisition" version="1.4.0.0" />
<!-- If the parameter section is left off, all memory is acquired.
     Memory is read in 4k pages, beginning and ending at page boundaries. -->
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
          <param name="offset">
            <value xsi:type="xsd:unsignedLong">204800</value>
          </param>
          <param name="size">
            <value xsi:type="xsd:unsignedLong">16384</value>
          </param>
        </parameters>
      </config>
    </command>
  </commands>
</script>
```

PROCESSDD.BAT AND ACQUIREPROCESSMEMORY.BATCH.XML

Description: Acquires the virtual address space of a process from memory.

Parameter	Required	Description	Data Type	Example
pid	Special	Process ID of the process to acquire. Either **pid** or **process name** must be specified.	Integer	582
process name (Called process in ProcessDD.bat)	Special	Specifies the name of the process to acquire.	String	smss.exe
Content Regex (Called content in ProcessDD.bat)	No	Only acquire processes that contain a particular regex content.	ArrayOfString	@mandiant.com
memory file (Called input in ProcessDD.bat)	No	Specifies the full path and filename containing an image of physical memory. Leave this parameter blank to scan live memory.	String	C:\vmware\Windows\xpsp2.vmem

Live Memory Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32processes-memoryacquire" version="1.4.0.0" />
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
          <param name="process name">
            <value xsi:type="xsd:string">malware.exe</value>
          </param>
        </parameters>
      </config>
    </command>
  </commands>
</script>
```

Memory Image Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32processes-memoryacquire" version="1.4.0.0" />
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
          <param name="process name">
            <value xsi:type="xsd:string">malware.exe</value>
          </param>
          <param name="memory file">
            <value xsi:type="xsd:string">c:\vmware\XPSP2\xpsp2.vmx</value>
          </param>
        </parameters>
      </config>
    </command>
  </commands>
</script>
```

Understanding the output

The output is the specified process or all processes as they occur in memory. Every process acquired can be placed in IDA Pro for further analysis. The output will be files named in the following convention:

- pid_path.ext

  OR
- pid_path_memorystart_memoryend.VAD

Files with .VAD extensions are the process' heap/stack and use the memory address range to name the section.

DriverDD.bat and AcquireDriver.Batch.xml

Description: Acquire a specified driver or all drivers loaded in memory.

Parameter	Required	Description	Data Type	Examples
driver name (Called driver in DriverDD.bat)	No	The name of the driver to acquire. If this parameter is left blank, then all the drivers on the system or in the image file are acquired and written to disk.	String	srv.sys
memory file (Called input in DriverDD.bat)	No	Specifies the full path and filename containing an image of physical memory. Leave this parameter blank to scan live memory.	String	C:\vmware\Windows\win2k3sp2.vmem

Live Memory Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32driver-memoryacquire" version="1.4.0.0" />
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
          <param name="driver name">
            <value xsi:type="xsd:string">srv.sys</value>
          </param>
        </parameters>
      </config>
    </command>
  </commands>
</script>
```

Memory Image Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32driver-memoryacquire" version="1.4.0.0" />
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
          <param name="driver name">
            <value xsi:type="xsd:string">srv.sys</value>
          </param>
          <param name="memory file">
            <value xsi:type="xsd:string">c:\MemoryImage\rootkit_xpsp1.img</value>
          </param>
        </parameters>
      </config>
    </command>
  </commands>
</script>
```

Understanding the Output

The output is the specified driver or all drivers as they occur in memory. Every driver acquired can be placed in IDA Pro for further analysis. It is important to remember that drivers with sections marked INIT will have had those sections freed from memory after the driver is loaded and before the acquisition begins.

Process.bat and ProcessAuditMemory.Batch.xml

Description: Collects a listing of all processes running on the target system by directly parsing structures contained in memory.

Parameter	**Required**	**Description**	**Data Type**
pid	No	Allows the user to specify a specific process to analyze. If this parameter or **process name** is not specified information on all processes is returned.	Integer
process name (Called process in Process.bat)	No	Allows the user to specify a specific process to analyze by name. If this parameter or **pid** is not specified information on all processes is returned.	String
handles	No	Instructs Memoryze to enumerate all handles in each matching process.	Boolean
sections	No	Instructs Memoryze to parse all memory section information for each matching process.	Boolean
ports	No	Instructs Memoryze to identify all open network ports used by a process.	Boolean
imports	No	Instructs Memoryze to enumerate each loaded binary's import address table.	Boolean
exports	No	Instructs Memoryze to enumerate each loaded binary's export address table	Boolean
injected	No	Instructs Memoryze to report any memory section that contains an injected DLL. ***This only works against certain types of injection attacks.***	Boolean
MemD5	No	Instructs Memoryze to hash the binary from memory. This still requires assess to the filesystem since binaries are memory mapped files. Please see https://media.blackhat.com/bh-us-11/Butler/BH_US_11_ButlerMurdock_Physical_Memory_Forensics-WP.pdf	Boolean
MD5	No	Instructs Memoryze to hash the binary on disk.	Boolean
SHA1	No	Instructs Memoryze to hash the binary on disk.	Boolean
SHA256	No	Instructs Memoryze to hash the binary on disk.	Boolean
digsig	No	Instructs Memoryze to check the digital signer of the binary.	Boolean
strings	No	Outputs all the strings found in memory. This can create very large output files.	Boolean
Content Regex (Called content in Process.bat)	No	Only return processes with a particular regex content.	ArrayOf String
memory file (Called input in Process.bat)	No	Specifies the full path and filename containing an image of physical memory. Leave this parameter blank to scan live memory.	String

Live Memory Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32processes-memory" version="1.4.0.0" />
      <!-- pid = 4294967295 = 0xffffffff, returns all processes -->
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
         <param name="pid">
            <value xsi:type="xsd:unsignedInt">4294967295</value>
          </param>
          <param name="handles">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="sections">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="ports">
            <value xsi:type="xsd:boolean">true</value>
          </param>
         <param name="strings">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="enumerate imports">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="enumerate exports">
            <value xsi:type="xsd:boolean">true</value>
          </param>
         <param name="detect injected dlls">
            <value xsi:type="xsd:boolean">true</value>
          </param>
        </parameters>
      </config>
    </command>
  </commands>
</script>
```

Memory Image Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32processes-memory" version="1.4.0.0" />
      <!-- pid = 4294967295 = 0xffffffff, returns all processes -->
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
         <param name="process name">
            <value xsi:type="xsd:string">calc.exe</value>
          </param>
          <param name="handles">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="sections">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="ports">
            <value xsi:type="xsd:boolean">true</value>
          </param>
         <param name="strings">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="enumerate imports">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="enumerate exports">
            <value xsi:type="xsd:boolean">true</value>
          </param>
         <param name="detect injected dlls">
            <value xsi:type="xsd:boolean">true</value>
          </param>
          <param name="memory file">
            <value xsi:type="xsd:string">c:\MemoryImages\XPSP1.img</value>
          </param>
        </parameters>
      </config>
    </command>
  </commands>
</script>
```

DriverWalkList.bat and DriverAuditModuleList.Batch.xml

Description: Parses operating system maintained lists of loaded drivers and kernel modules. Note that the data generated by this script and that generated by **DriverAuditSignature.Batch.xml or DriverSearch.bat** may be different and that difference is not, in and of itself, an indication of a "hidden" driver.

The script may be used to parse the contents of a memory image. It could also be used to analyze the memory file from a virtual machine. The list of identified drivers is returned, including names, base addresses, sizes, and the path to the executable file on disk.

Live Memory Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32drivers-modulelist" version="1.4.0.0" />
    </command>
  </commands>
</script>
```

Memory Image Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32drivers-modulelist" version="1.4.0.0" />
     <config xsi:type="ParameterListModuleConfig">
      <parameters>
          <param name="memory file">
            <value   xsi:type="xsd:string">C:\MemoryImages\win2ksp4.img</value>
          </param>
          </parameters>
       </config>
    </command>
  </commands>
</script>
```

Understanding the output

- **ModuleAddress -** Address of the driver object or kernel module object in memory.
- **ModuleInit -** Address of the initialization function for the driver.
- **ModuleBase -** Base address of the driver or module.
- **ModuleSize -** Size of the module in bytes.
- **ModuleName -** Name of the driver or module (e.g. "kdcom.dll").
- **ModulePath -** Path to the executable file on disk represented by the driver or module.

DRIVERSEARCH.BAT AND DRIVERAUDITSIGNATURE.BATCH.XML

Description: Scans memory looking for structures representing drivers and reports them. Note that the data generated by this module and that generated by **DriverAuditModuleList.Batch.xml or DriverWalkList.bat** may be different and that difference is not, in and of itself, an indication of a "hidden" driver. The module may also be used to parse the contents of a memory image. It could also be used to analyze the memory file from a virtual machine. The list of detected driver objects is returned, including names, base addresses, sizes, and the memory addresses of various functions within the driver.

Parameter	Required	Description	Data Type
imports	No	Instructs Memoryze to enumerate each loaded binary's import address table.	Boolean
exports	No	Instructs Memoryze to enumerate each loaded binary's export address table	Boolean
MD5	No	Instructs Memoryze to hash the binary on disk.	Boolean
SHA1	No	Instructs Memoryze to hash the binary on disk.	Boolean
SHA256	No	Instructs Memoryze to hash the binary on disk.	Boolean
digsig	No	Instructs Memoryze to check the digital signer of the binary.	Boolean
strings	No	Outputs all the strings found in memory. This can create very large output files.	Boolean
memory file (Called input in DriverSearch.bat)	No	Specifies the full path and filename containing an image of physical memory. Leave this parameter blank to scan live memory.	String

Live Memory Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32drivers-signature" version="1.4.0.0" />
    </command>
  </commands>
</script>
```

Memory Image Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32drivers-signature" version="1.4.0.0" />
     <config xsi:type="ParameterListModuleConfig">
      <parameters>
          <param name="memory file">
            <value xsi:type="xsd:string">C:\MemoryImages\win2ksp2.img</value>
          </param>
          </parameters>
       </config>
    </command>
  </commands>
</script>
```

Understanding the output

- **Driver Object Address -** Address in memory of the driver object
- **Image Size -** Size of the entire driver image in memory.
- **Image Base -** Base address of the driver.
- **DriverName -** Name of the driver (if available).
- **DriverInit -** Address of the driver's initialization function.
- **DriverstartIo -** Address of the driver's DriverStartIo function.
- **DriverUnload -** Address of the driver's DriverUnload function.
- **DeviceObject** – address of the DRIVER_OBJECT's corresponding device is it exists.
- **AttachedToDevice** – if the driver is attached to a device this would be the device's name.
- **AttachedToDriver** – If the driver is attached to a device this would be that device's driver's name.
- **AttachedDriver** – if the current driver has another driver attached to it, this would be the attached driver's name.
- **AttachedDevice** – if the driver has an attached device this would be that attached device's name.
- **Irp* -** Address of functions handling various IRP_MJ messages, including:
    - IRP_MJ_CREATE, IRP_MJ_CREATE_NAMED_PIPE, IRP_MJ_CLOSE, IRP_MJ_WRITE, IRP_MJ_READ, IRP_MJ_QUERY_INFORMATION, IRP_MJ_SET_INFORMATION, IRP_MJ_QUERY_EA, IRP_MJ_FLUSH_BUFFERS, IRP_MJ_QUERY_VOLUME_INFORMATION, IRP_MJ_SET_VOLUME_INFORMATION, IRP_MJ_DIRECTORY_CONTROL, IRP_MJ_FILE_SYSTEM_CONTROL, IRP_MJ_DEVICE_CONTROL, IRP_MJ_SHUTDOWN, IRP_MJ_LOCK_CONTROL, IRP_MJ_CLEANUP, IRP_MJ_CREATE_MAILSLOT, IRP_MJ_QUERY_SECURITY, IRP_MJ_SET_SECURITY, IRP_MJ_POWER, IRP_MJ_SYSTEM_CONTROL, IRP_MJ_DEVICE_CHANGE, IRP_MJ_QUERY_QUOTA, IRP_MJ_SET_QUOTA, IRP_MJ_PNP

HookDetection.bat and HookAudit.Batch.xml

Description: Detects potential rootkits on target systems by identifying system calls, interrupts, and drivers on the target system that have been hooked. While hooking is a common rootkit technique, not all hooks are malicious - some legitimate software performs its primary function through hooking system calls and inserting itself in the control flow of the operating system.

Parameter	Required	Description	Data Type
idt	Yes	Check certain Interrupt Descriptor Table entries to determine if they have been hooked.	Boolean
ssdt_index (Called ssdt in HookDetection.bat)	Yes	Check the System Call Table for hooks.	Boolean
ssdt_inline (Called functions in HookDetection.bat)	Yes	Check the System Call Table functions for hooks.	Boolean
drivers	Yes	Check all drivers for IRP hooks.	Boolean
memory file (Called input in HookDetection.bat)	No	Specifies the full path and filename containing an image of physical memory. Leave this parameter blank to scan live memory.	String

Live Memory Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32kernel-hookdetection" version="1.4.0.0" />
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
        <param name="idt">
            <value xsi:type="xsd:boolean">true</value>
        </param>
        <param name="ssdt_index">
            <value xsi:type="xsd:boolean">true</value>
        </param>
        <param name="ssdt_inline">
            <value xsi:type="xsd:boolean">true</value>
        </param>
        <param name="drivers">
            <value xsi:type="xsd:boolean">true</value>
        </param>
        </parameters>
      </config>
    </command>
  </commands>
</script>
```

Memory Image Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" chaining="implicit">
  <commands>
    <command xsi:type="ExecuteModuleCommand">
      <module name="w32kernel-hookdetection" version="1.4.0.0" />
      <config xsi:type="ParameterListModuleConfig">
        <parameters>
        <param name="idt">
            <value xsi:type="xsd:boolean">true</value>
        </param>
        <param name="ssdt_index">
            <value xsi:type="xsd:boolean">true</value>
        </param>
        <param name="ssdt_inline">
            <value xsi:type="xsd:boolean">true</value>
        </param>
        <param name="drivers">
            <value xsi:type="xsd:boolean">true</value>
        </param>
       <param name="memory file">
            <value xsi:type="xsd:string">c:\MemoryImages\w2k3sp2.img</value>
        </param>
        </parameters>
      </config>
    </command>
  </commands>
</script>
```

Understanding the output

- **HookDescription -** Identifies whether the hooked entity is a System Call or a Driver.
- **HookedFunction -** Identifies the function that has been hooked.
- **HookedModule -** Identifies the executable module that has been hooked.
- **HookingModule -** Identifies the executable module performing the system or driver hooking.
- **HookingAddress -** The address in memory of the function performing the system or driver hook.

Component Licensing Notices

MANDIANT Memoryze contains software from the following Open Source Projects:

- **libcurl** MIT License, http://curl.haxx.se
- **libxml2** MIT License, http://xmlsoft.org
- **log4net** Apache License, http://logging.apache.org
- **xerces** Apache License http://xerces.apache.org/xerces-c/
- **xqilla** Apache License http://xqilla.sourceforge.net/
- **OpenSSL** OpenSSL License, http://www.openssl.org
- **pcre** BSD License, http://www.pcre.org
- **zlib** ZLIB License, http://www.zlib.net

Your use of MANDIANT Memoryze is governed solely by the EULA and software documentation, which comply with the notices contained herein.

LIBCURL

PROJECT URL

http://curl.haxx.se

License

LIBXML2

PROJECT URL

http://xmlsoft.org

License

LOG4NET, XERCES, AND XQILLA

PROJECT URL

log4net: http://logging.apache.org/log4net/
xerces: http://xerces.apache.org/xerces-c/
xqilla: http://xqilla.sourceforge.net/

License

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/
TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION
1. Definitions.
"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.
"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.
"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.
"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.
"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.
"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).
"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship.
For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.
"Contribution" shall mean any work of authorship, including the original version of the

Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."
"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory
patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative
Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
You must give any other recipients of the Work or Derivative Works a copy of this License; and You must cause any modified files to carry prominent notices stating that You changed the files; and You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works;

or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.
You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify,

defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.
END OF TERMS AND CONDITIONS

OpenSSL

Project URL

http://www.openssl.org

License

LICENSE ISSUES
==============
The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL please contact openssl-core@openssl.org.

OpenSSL License

/* ==
* Copyright (c) 1998-2007 The OpenSSL Project. All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
*
* 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
*
* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in
* the documentation and/or other materials provided with the
* distribution.
*
* 3. All advertising materials mentioning features or use of this
* software must display the following acknowledgment:
* "This product includes software developed by the OpenSSL Project
* for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
*
* 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
* endorse or promote products derived from this software without
* prior written permission. For written permission, please contact
* openssl-core@openssl.org.
*
* 5. Products derived from this software may not be called "OpenSSL"

Original SSLeay License

```
* Copyright remains Eric Young's, and as such any Copyright notices in
* the code are not to be removed.
* If this package is used in a product, Eric Young should be given attribution
* as the author of the parts of the library used.
* This can be in the form of a textual message at program startup or
* in documentation (online or textual) provided with the package.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the copyright
* notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the
* documentation and/or other materials provided with the distribution.
* 3. All advertising materials mentioning features or use of this software
* must display the following acknowledgement:
* "This product includes cryptographic software written by
* Eric Young (eay@cryptsoft.com)"
* The word 'cryptographic' can be left out if the rouines from the library
* being used are not cryptographic related :-).
* 4. If you include any Windows specific code (or a derivative thereof) from
* the apps directory (application code) you must include an acknowledgement:
* "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
*
* THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*
* The licence and distribution terms for any publically available version or
* derivative of this code cannot be changed. i.e. this code cannot simply be
* copied and put under another distribution licence
* [including the GNU Public Licence.]
*/
```

PCRE

PROJECT URL

http://www.pcre.org

License

PCRE LICENCE

PCRE is a library of functions to support regular expressions whose syntax and semantics are as close as possible to those of the Perl 5 language.

Release 7 of PCRE is distributed under the terms of the "BSD" licence, as specified below. The documentation for PCRE, supplied in the "doc" directory, is distributed under the same terms as the software itself.

The basic library functions are written in C and are freestanding. Also included in the distribution is a set of C++ wrapper functions.

THE BASIC LIBRARY FUNCTIONS

Written by: Philip Hazel
Email local part: ph10
Email domain: cam.ac.uk

University of Cambridge Computing Service,
Cambridge, England.

Copyright (c) 1997-2008 University of Cambridge
All rights reserved.

THE C++ WRAPPER FUNCTIONS

Contributed by: Google Inc.

Copyright (c) 2007-2008, Google Inc.

All rights reserved.

THE "BSD" LICENCE

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

- Neither the name of the University of Cambridge nor the name of Google Inc. nor the names of their contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

End

ZLIB

PROJECT URL

http://www.zlib.net

License

Copyright (C) 1995-2005 Jean-loup Gailly and Mark Adler This software is provided 'asis', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software. 3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly jloup@gzip.org
Mark Adler madler@alumni.caltech.edu