R17 Forensic Tool
Quick Start Guide

Warning

r17 was created to assist forensic investigators in quickly collecting forensic artifacts from a live Windows host for offline analysis. It was written using high-quality software development practices and testing procedures. The suitability, quality, behavior, and use of this software for any specific purpose is not guaranteed.

USE THIS TOOL AT YOUR OWN RISK:

```
THIS SOFTWARE IS PROVIDED BY THE PROJECT 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 PROJECT 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.
```

Use of this tool for illegal, malicious, or immoral purposes is prohibited.

Introduction

r17.exe is a forensic tool that collects forensic evidence from a live Windows system. This is useful for investigators who prefer to do offline analysis of forensic artifacts. Written in C++ and statically linked, the tool is lightning fast and does not require any external shared libraries or other dependencies. It runs on all Windows platforms from Windows XP forward.

Features

The below table lists the artifacts r17 acquires.

Application Features - User defined working directory where artifacts are stored prior to archiving. - User defined archive directory where the artifact archive is placed. - Ability to encrypt archive with a password (hunt_4_malware). - Ability to delete working folder upon program exit. - Ability to delete r17.exe binary upon program exit. - Performs MD5 and SHA1 hash calculation on all extracted files. - Creates a chain-of-custody manifest log file.
Filesystem Artifacts - Extracts $MFT from all local NTFS filesystems ($MFT extracted at the sector level). - Extracts $MFTMirror from all local NTFS filesystems. - Extracts $LogFile from all local NTFS filesystems. - Extracts $USNJRNL from all local NTFS filesystems. - Extracts $MFTMirror from all local NTFS filesystems. - Extracts contents of Prefetch folder (Prefetch & superfetch files).

- Extracts NTUser.dat files from all user profiles.
- Extracts UsrClass files from all user profiles.

Internet Artifacts
- Extracts Internet Explorer and Edge history artifacts.
- Extracts Internet FireFox history artifacts.
- Extracts Chrome and Chromium history artifacts.

Log Artifacts
- Extracts the most useful event logs. User configurable switch to collect all event logs.
- Extracts IIS web server logs including error logs.

Network Artifacts
- Executes ‘netstat’ and saves the output to a text file.
  (‘netstat -anob’, ‘netstat -an’ for OS’s that don’t support ‘o b’)
- Executes ‘ipconfig /displaydns’ and saves the output to a text file.
- Extracts the system ‘hosts’ file.
- Executes ‘ipconfig’ and saves the output to a text file.
- Executes ‘arp -a’ and saves the output to a text file.
- Executes ‘nbtstat -crs’ and saves the output to a text file.
- Executes ‘net sessions’ and saves the output to a text file.
- Executes ‘net users’ and saves the output to a text file.
- Executes ‘net share’ and saves the output to a text file.
- Uses WMI to acquire mapped drive information.
- Uses WMI to acquire detailed network adapter configuration information.
- Identifies any network adapter that is in promiscuous mode.

Processes
- Executes ‘tasklist -fo csv -v’ and saves output to a text file.
- Executes ‘tasklist -fo csv /svc’ and saves output to a text file.
- Executes ‘tasklist -fo csv /m’ and saves output to a text file.
- Uses WMI to acquire detailed process information including command line parameters.

Registry
- Extracts SYSTEM, SOFTWARE, APPLICATION, and SECURITY registry hives.
- Extracts backup registry files if present.

Scheduler
- Executes ‘at’ and saves output to a text file.
- Executes ‘schtasks’ and saves output to a text file.
- Acquires SCHEDLGU.txt file from Windows tasks folder.
- Acquires SA.DAT file from Windows tasks folder.

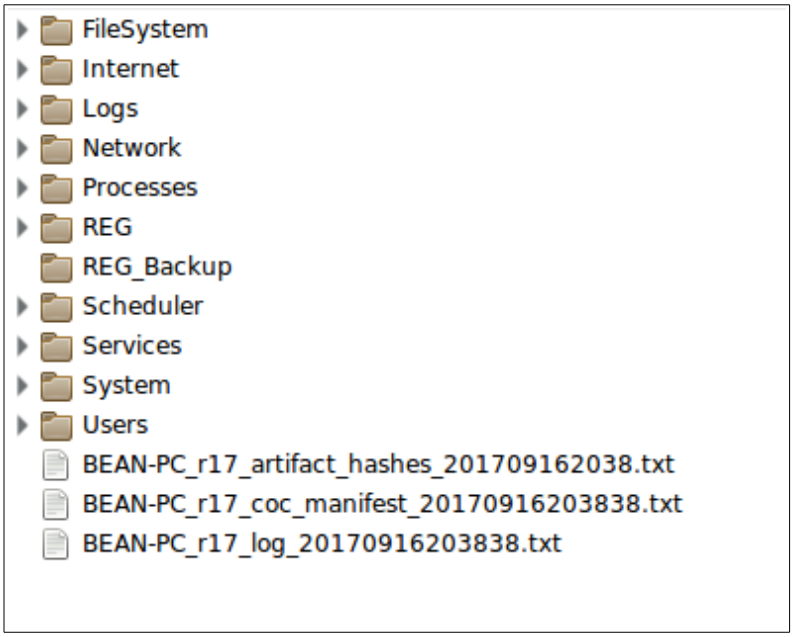
Services
- Uses WMI to acquire detailed services information.

System
- Executes “date /t & time /t” and saves output to a text file.
- Executes ‘systeminfo’ and saves output to a text file.
- Uses WMI to acquire detailed OS information.
- Acquires MBR’s of all local disk drives.
- Acquires VBR’s of all volumes on local disk drives.

Users
- Extracts MAC times on all user profile folders.
- Extracts temporal information for all user profiles.

- Acquires logged on user information using the Win32 API.
- Acquires logged on user information using WMI.
- Uses WMI to acquired detailed account information.
- Uses WMI to acquired detailed user account information.
- Uses WMI to acquired detailed group information.
- Uses WMI to acquired detailed user/group membership information.
- Uses WMI to acquired detailed user logon profile information.

The below figure shows the folder hierarchy of the extracted artifact 7zip file.



Configuration

To customize the behavior of r17, you can specify settings in an Ini file. A reference r17.ini file is included with the r17 distribution package. The Ini file is documented in detail making it easy to change settings. The default behavior of each setting is described in the comment sections.

If you make configuration choices in a r17.ini file, this file MUST be placed in the same folder where the r17.exe executes from. If you do not use an Ini file, then r17.exe will use the default settings.

Execution

To run r17.exe on a Windows host, simply copy r17.exe and an optional r17.ini to the host and execute it ***with Administrator privileges***. Most user place r17.exe in a \Temp folder (\Windows\Temp is a common location too). It is encouraged that you run r17 from removable media such as a USB thumb device.

The tool does not accept any command line arguments and it does not create a console window or any other visible window. It is designed to be stealthy.

r17's default behavior is to create a working directory in the directory of execution (cwd). A sub-folder named r17 is created and a sub-folder for each artifact type is created under the r17 parent folder.

Once all the artifacts have been collected, the entire working folder structure is archived in a 7zip file and optionally encrypted with the password '*hunt_4_malware*'. Depending on Ini settings, the working folder, log files, and the r17.exe binary itself is deleted. The default behavior is to delete them, leaving only the artifact archive file on the filesystem.

The name of the artifact archive will be formatted as 'Hostname_r17_yyyymmddhhmmss.7z'. As an example, r17 was executed on a Windows 10 workstation named MAGOO and it was run on August 30, 2017 at 11:45:13 PDT. The archive file is named: MAGOO_r17_20170830114513.7z.
The archive contents contains a log file, hashes file, chain-of-custody manifest file, and a folder structure shown in the below figure.

Each relevant artifact is located in the appropriate sub-folder.

Troubleshooting

The most common problem is that r17.exe is not run with Administrator permissions. You must execute r17 with Admin credentials or it will fail to collect all the artifacts.

When r17 first starts, it looks for a file named r17.ini. If the file is found, the configuration setting are read. Next, a folder named r17 is created in the working_directory folder setting in the Ini file. If no r17.ini file is found, the working directory will be the current directory. If a working folder r17 cannot be created, the program will silently fail. This is due to the fact the log file has not been created yet so you will not know why it failed..

If r17 starts and then exits without creating an archive file, make sure you are running it with Admin credentials and the working_directory in the r17.ini file is correct.

Development Details

r17 is written in C++ using Microsoft Visual Studio. Below is a list of components used in the software.

- The free/open-source cross-platform wxWidgets library for useful functions. (https://www.wxwidgets.org).
- The invaluable SleuthKit library built and maintained by Brian Carrier.
- The SafeArray macro created by Wictor Wilen and placed in the public domain. (https://www.codeproject.com/Articles/1279/Simplify-your-Safearray-loops-using-macros)
- The wxFormatStringAsNumber class provided by Manuel Martin and licensed under the wxWidgets license.
- The implementation of MD1 and SHA1 hash functions by Dominik Reichl released in the public domain. (http://www.dominik-reichl.de).

All other code was written by me, Michael G. Spohn.

Feedback

Comments, suggestions, bug reports, and enhancement requests should be sent to: mspohn@malware-hunters.net.