Scanning for “low hanging fruit” using open source tools

H. Carvey
Chief Forensics Scientist, ASI
**Who am I?**

Chief Forensics Scientist at ASI. Forensic Nerd. Published Author.

**Why are we here?**

Topic *du jour* – using open source tools in forensic analysis
What are we trying to do, or what problem are we trying to solve?

- Automate tedious, repetitive tasks
- What are those tasks we perform often/all the time, but are tedious, time consuming, prone to errors, or we simply forget to do?
• Determine what we’re looking at – OS, version, etc.
• Determine (active) users on the system
• Malware detection
  – Sometimes, AV “misses” stuff
• USB Device Analysis
• Installed applications
• Generate data for timeline analysis
“Low hanging fruit” is anything you’ve already found, or are aware of...

- Contents of hosts file (malware detection)
- Check files in Temp directories
- Check for PDF/XLS(X) files in user’s email attachments directory
Automation is the key! Let the computer do the work!

Write a plugin once, use it over and over again...run the same check every time.

Automate:
• Malware detection checklists
• Collection of configuration info that can affect your exam

This **DOES NOT REPLACE ANALYSIS**; it leaves analysts to analyze!
Benefits

• Automation
  – Write once, use many times
• Reproducible results
• Knowledge retention
  – Plugin exists & can be run, even when analyst who wrote it is on vacation, sick, or has left the organization
• Force multiplier
  – Write once, many analysts use
  – One analyst spends 10 hrs “finding something new”; share with 10 other analysts, save 100+ hrs across the team
• Competitive Advantage
  – Create your own unique plugins
• Career progression
  – Retained knowledge is a good “text book” for learning; start at lab tech level, work up to examiner, writing plugins, etc.
Design

- Use Nessus and RegRipper as the model(s)
- Run against a mounted volume
  - Mount with tool or method of choice (FTK Imager, VHD, VMDK, etc.)
  - Allows access to VSCs, as well as access via F-Response
  - Focus is on logical files
  - Be sure to run scanner as an Admin user!!
  - Consider use of RunAsSystem from Joachim Metz (reboot.pro)
  - Again, does not replace analysis; just gets you there quicker
- Started with Windows
  - Currently available platform & target
  - Can be ported to other platforms
  - Can be written to support other target platforms, as long as they can be accessed as a volume or mount point
• Perl, at the moment
  – That’s just how I implemented it for the moment
  – Originally due to available modules
• As the target is a volume, no proprietary APIs
  – To open a directory, use `opendir()/closedir()`
  – To open a file, use `open()/binmode()/close()`
Mounted volume provides greatest overall flexibility

- Mount with FTK Imager, VHDTTool, etc.
- Export logical files to FAT volume (Windows permissions)
- Access VSCs
- Access via F-Response
- Model easily mapped to Linux
- Opens the door for other platforms
  - iDevices via MacDrive (Windows) or Linux
  - Export logical file structure from
Scenario

- Image acquired from XP system
- Image located in D:\cases\XP directory
- Mount image as “F:\”
- Need to scan it...
- What does the process look like?
Mount image via FTK Imager

Add Image
- Image File: D:\cases\XP\image.001
- Mount Type: Physical & Logical
- Drive Letter: Next Available (G:)
- Mount Method: Block Device / Read Only

Write Cache Folder:
- D:\cases\XP

Mapped Image List

<table>
<thead>
<tr>
<th>Drive</th>
<th>Method</th>
<th>Partition</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>PhysicalDrive2</td>
<td>Block Device/Read Only</td>
<td>Image</td>
<td>D:\cases\XP\image.001</td>
</tr>
<tr>
<td>F:</td>
<td>Block Device/Read Only</td>
<td>Partition 1 [20471...</td>
<td>D:\cases\XP\image.001</td>
</tr>
</tbody>
</table>

Mount
Unmount
Close
Scanner UI – step 1

- Path: f:\windows\system32
- Report Dir: d:\cases\XP

[Image of the Forensic Scanner, v.1.0 window with highlighted areas for Path and Report Dir fields, and the Init button.]
• Analyst enters/selects path to system32 folder
• Analyst enters path for reports (must be writeable)
• Analyst clicks the "Init" button

• Scanner:
  – Gets info about version of the "system"
  – Gets info about available user profiles
  – Collects available plugins based on version of the target platform
  – Plugins are split into classes; those for the system, and those for user profiles
• Analyst selects user profiles to be scanned
• Analyst clicks the “Scan” button

• Scanner:
  – Runs plugins based on class (system or user)
  – Also separates plugins based on “category” (thanks to Corey Harrell for pointing this out)
  – Runs plugins for the system first, then runs the plugins for users against each profile, in succession...again, grouped together by category
  – Each user gets their own report file
  – Generates log of activity (plugins run)
• Output report
  • One for the system
  • Each user gets their own report file (automagically)
• Activity Log
  • Includes info about system scanned
Beneficial Uses

• Make part of in-processing of images
  – Lab tech receives image; verifies, scans, uploads image and report for analyst

• Analysts can seek assistance without exposing sensitive information
  – Archive/secure the text report, send to another analyst for review
  – Much smaller than the full image, much easier to secure and send
  – Can be used by on-site analysts seeking assistance with an on-going engagement
Future Direction(s)

• Add plugins
  – Including ability to run external, third-party tools (CLI)
  – Add support for plugin categories
• Make it easier to run on other platforms
• Add support for other target platforms
• Release it...wait...already done!!

http://code.google.com/p/forensicsscanner
Questions??

H. Carvey
harlanc@appliedsec.com
keydet89@yahoo.com

http://windowsir.blogspot.com