Verification of Digital Forensic Tools

Jim Lyle

Project Leader: Computer Forensic Tool Testing (CFTT)

Information Technology Laboratory

National Institute of Standards and Technology

Disclaimer

Certain trade names and company products are mentioned in the text or identified. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products are necessarily the best available for the purpose.

Introduction

- The computer is ubiquitous in both civil and criminal cases.
- BTK was solved by digital clues on a floppy disk that pointed to Dennis Rader: Police found metadata embedded in a deleted Microsoft Word document that was, unbeknownst to Rader, on the disk. The metadata, recovered using the forensic software EnCase, contained "Christ Lutheran Church", and the document was marked as last modified by "Dennis". A search of the church website turned up Dennis Rader as president of the congregation council.
- What are the components used to extract digital evidence?
- How reliable is digital evidence?

Outline

- Overview of CFTT
- Digital Forensic Tools
- Test results
  - Data acquisition tools
  - Write Block Tools
- Error rates
- Summary

Goals of NIST Computer Forensics Projects

- Support use of automated processes into the computer forensics investigations
- Provide stable foundation built on scientific rigor to support the introduction of evidence and expert testimony in court

Project Sponsors (aka Steering Committee)

- National Institute of Justice (Major funding)
- FBI (Additional funding)
- Department of Defense, DCCI (Equipment and support)
- Homeland Security (Major funding, Technical input)
- State & Local agencies (Technical input)
- Internal Revenue, IRS (Technical input)
- NIST/OLES (Additional funding & Program management)

Current NIST Activities

- Provide international standard reference data to support investigations and research (NSRL)
- Establish computer and mobile device forensic tool testing methodology (CFTT)
- Provide test material for proficiency testing and lab-based tool testing (CFReDs)

CFTT Products

- Forensic Tool Requirements
- Forensic Tool Test Plan
  - List of test cases
  - Test data sets (via CFReDS)
  - Test support & analysis software
- Forensic Tool Test Reports (submitted to NIJ for publication)

Test Reports Published

- Data acquisition: EnCase, FTK, SafeBack, MFL, dd, Macquisition, IxImager, …
- Software write block: HDL, PDBLOCK & ACES
- Hardware write block: MyKey, Tableau, WiebeTech, DiskJocky, DriveLock, & FastBlock
- Mobile Device (cell phone): Paraben, BitPim, MOBILedit, Neutrino, GSM XRY, …
- Drive wipe: Boot & Nuke, Voom, Drive eRazer

What’s Next for CFTT

- Additional tools such as . . .
  - Deleted file recovery (searching trash can)
  - File carving (searching the dumpster)
  - String search
  - Volatile acquisition of memory & disk
  - etc
- Test methodology and report sharing

Four Main Sources of DE

- Hard drive
  - Static: easy to reacquire
- Live memory
  - Dynamic: frequent change
- Mobile device: cell phone, PDA, iphone
  - Almost static: examination introduces changes
- Network tools
  - Dynamic: like a flowing stream

Tool Testing is Analogous to a Court Trial

Statutes	Tool requirements
Detective/DA	Test operator
Defendant	Tool under test
Guilty verdict	Tool violates at least one requirement
Not guilty verdict	Tool not observed to violate any requirement

Tool Testing Strategy

- Digital forensic tools are often multi-function
- Testing is organized by function
- Develop requirements for a single function
- Test tools for a single function at a time

Good News & Disappointing News

- Good News: Forensic tools as tested work with some minor problems
  - Usually something is omitted
  - Nothing extra (incriminating or not) is created
- Disappointing News: Error rates are hard to define & quantify

Tool Functions

- Data acquisition
- Data protection (write blocking to protect original)
- Data erasing (disk wiping to ensure against cross contamination between cases)
- Data extraction (recovering infromation from mobile devices)
- File reconstruction (under development)
- String searching (under development)

Data Acquisition Requirements

- Entire drive or partition is acquired
- All data is acquired matches original
- Any omitted (e.g., bad sector) data is:
  1. Identified
  2. Replaced with benign replacement
- Tool log is accurate

Testing Data Acquisition

- Tool acquires either
  - entire drive (physical drive)
  - partition (logical drive)
- Evaluate the acquisition by either …
  - Hash of data acquired
  - Compare source to a restore

Data Acquisition Test Results

- Sectors at end of drive omitted
  - Tool dd, using Linux kernel 2.4, with a drive with an odd number of sectors, omits the last sector (512 bytes). The last sector is not used.
  - Tool EnCase version 3, using BIOS access, on hard drives with certain geometry, using a computer with a certain BIOS, omits the last 5,040 sectors.
  - Tool SafeBack version 2, with same setup omits the last 1,008 sectors.
  - Both SafeBack & EnCase, using DIRECT access, no sectors omitted.

More Data Acquisition Results

Acquiring an image of an NTFS partition

- FTK omits the last 8 sectors
- EnCase:
  1. Omits the last sector
  2. Replaces the 7 sectors just before the last sector with 7 sectors acquired earlier.
- However, those last 8 sectors are not used to store user data.

Acquiring Bad Sectors

Disk sectors do fail and become unreadable

- Tool dd running in Linux, omits 7 sectors around a bad sector acquired over the ATA interface.
- Tool dd running in Linux omits multiple of 8 sectors around a bad sector acquired over a non-ATA interface.
- Omitted sectors are replaced with zeros.
- Tool dd running in FreeBSD acquires all readable sectors but replaces bad sectors with non-zero data of unknown (to me) origin.

Write Block Requirements

- All commands that change drive content are blocked
- Data can be read off the drive
- Huh? Why not just say all READ commands are allowed?

Write Block Results

- New WRITE command not blocked
- Some READ commands blocked
- A certain READ command was replaced with a different READ command
- ERASE command allowed

As to General Observations from Daubert

- … known or potential error rate, and the existence and maintenance of standards controlling its operation …
- Usually does not apply to tools used to acquire and examine digital evidence.

Sources of Error

- An algorithm may have a theoretical error rate
- An implementation of an algorithm may have errors
- The execution of a procedure may have a blunder that affects the result

Error Example

- Hashes or checksums (with useful attributes) can be computed for a file.
  - Same files have the same hash
  - Different hash means files are different
  - However, same hash is possible for different files
- This can be used to determine if:
  - A file has changed, or
  - If two files might be the same with some error rate.

An Algorithm To Compare A Pair Of Files With Only One File

- A hash or checksum can be used to determine if any file in a set of files match a given file.

1. Let c be the hash of the given file
2. For each file, f, in the set …
   1. Compute, h, the hash of f
   2. Compare c to h
   3. If c matches h, then declare c equals h

- Hashes can collide (two different files with same hash)
- The error rate of the algorithm is related to the size of the hash (number of bits)

Error Rates for Hash Algorithms

- Hash algorithms are designed to essentially randomize the file content.
- This allows us to assume that different files behave like random data.

Hash Algorithm	Chance of Collision
CRC-16	1 in 32,768
CRC-32	1 in 2,147,483,648
MD5 (128 bits)	1 in 170141183460469231731687303715884105728
SHA-1	1 in 2^{159}
SHA-256	1 in 2^{255}

Implementation Errors

- A variety of implementation errors are possible, some are quite subtle.
  - One common error occurs as follows:
    - Hash algorithm is implemented in a UNIX environment. It works for any file.
    - Same program is moved to MS Windows environment. It works fine for any *binary* file, but computes a different (wrong) value for any *text* file (Windows adds a character to the end of each line of text).

What is the error rate?

- In the science of measurement error analysis this is called a *systematic error.*
- The distribution of text and binary files varies from computer to computer.
- There is no random distribution to the manifestation of the error.
- The implementation error is triggered only under some set of conditions.
- Errors, but no error rate.

Human Errors

- Human errors (blunders) occur
- Difficult to quantify
- Good processes have built in checks to detect blunders

Other Tool Testing Projects

- RCMP
- CART – FBI internal
- DCCC – Available on request

Summary & Observations

- Tools that have been tested so far don’t report data that isn’t there.
- Tools tend to have minor problems, usually omitting data, sometimes duplicating existing data.
- Digital forensic tools are being independently tested by several organizations.
- Conclusions of a test report only apply to the tool version tested.
- Any change to a tool or run environment requires retesting.
- Error rates can often be stated for algorithms, but not for implementations.
- Most digital forensic tool functions are simple collection, extraction or searching operations with a zero error rate for the algorithm.
- An implementation may have systematic errors that can be revealed by tool testing programs.

Resources

- www.cftt.nist.gov
- www.cfreds.nist.gov
- http://www.dc3.mil/dcci/dcciCyberFiles.php
- www.swgde.org
- John Robert Taylor (1999). *An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements*. University Science Books ISBN 093570275X.

Contact Information

Jim Lyle
jlyle@nist.gov

Sue Ballou, Office of Law Enforcement Standards
Steering Committee representative for State/Local Law Enforcement
Susan.ballou@nist.gov