

Digital Forensics and Media Exploitation: Technology, Policy and Countermeasures

Simson L. Garfinkel, Ph.D.

<http://www.simson.net/>

Tutorial M1

ACSAC 2009

Monday, December 7th, Full Day



A bit about me

Tech Journalist: 1985—2002

Entrepreneur: 1988—2002

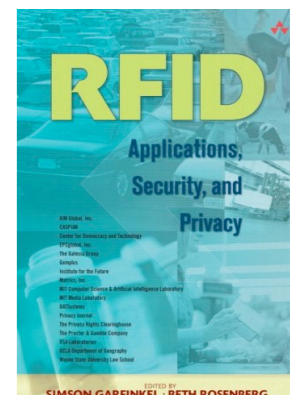
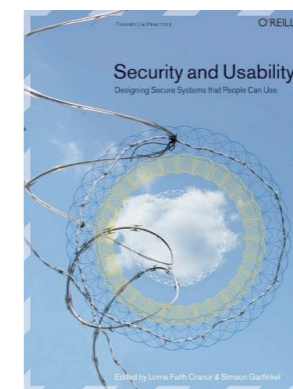
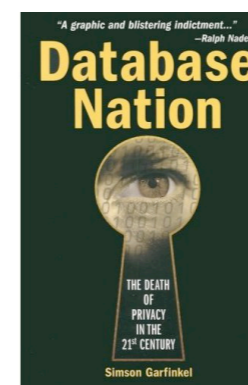
- Vineyard.NET, Broadband2Wireless,
- Sandstorm Enterprises, Inc.

MIT EECS 2002—2005

Harvard 2005—2008

Naval Postgraduate School 2006—

- Associate Professor



“The views expressed in this presentation do not necessarily reflect those of the Department of Defense or the US Government.”

NPS is the Navy's Premiere Research University



Located in: Monterey, CA

627 acres; 1500 students

- US Military (All 5 services)
- US Civilian (Scholarship for Service & SMART)
- Foreign Military (30 countries)

Schools:

- Business & Public Policy
- Engineering & Applied Sciences
- Operational & Information Sciences
- International Graduate Studies



"Ask me about our civilian MS and PhD programs!"

My current research: Automated Document & Media Exploitation

Spring 2009 publications:

- XML and Python for automated Forensics
- Corpora Development
- File Fragment Identification
- AFF4: Evidence file format

<http://simson.net/page/Research>



<http://www.simson.net/clips/academic/2007.ACM.Domex.pdf>

The DOMEX challenge is to turn digital bits into actionable intelligence.

I maintain the Forensics Wiki:

<http://www.forensicswiki.org/>



[page](#) [discussion](#) [view source](#) [history](#)

[Log in / create account](#)

Main Page

This is the **Forensics Wiki**, a [Creative Commons](#)-licensed [wiki](#) devoted to information about [digital forensics](#) (also known as computer forensics). We currently list a total of **498** pages.

Much of [computer forensics](#) is focused on the [tools](#) and [techniques](#) used by [investigators](#), but there are also a number of important [papers](#), [people](#), and [organizations](#) involved. Many of those organizations sponsor [conferences](#) throughout the year and around the world. You may also wish to examine the popular [journals](#) and some special [reports](#).

Selected Forensics Research

2008-Aug-13

[Lest We Remember: Cold Boot Attacks on Encryption Keys](#)

J. Alex Halderman, Princeton University; Seth D. Schoen, Electronic Frontier Foundation; Nadia Heninger and William Clarkson, Princeton University; William Paul, Wind River Systems; Joseph A. Calandrino and Ariel J. Feldman, Princeton University; Jacob Appelbaum; Edward W. Felten, Princeton University

[USENIX Security '08 Refereed Paper](#)

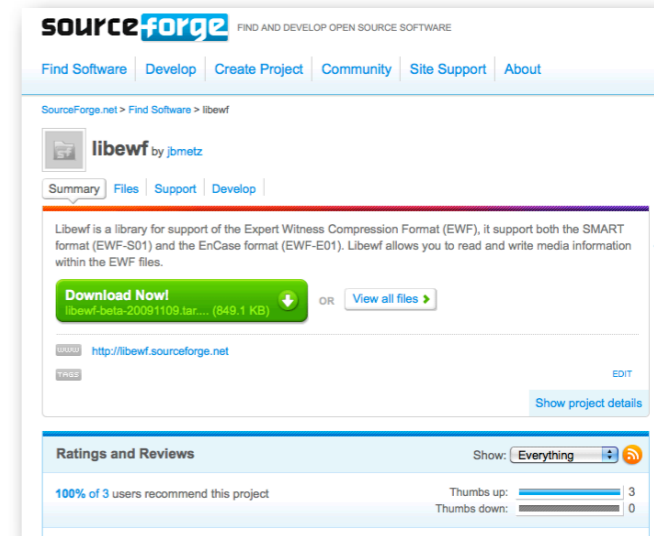
Awarded Best Student Paper

Increasingly memory analysis is of interest in forensic research---both because new malware only resides in memory, and because memory analysis is frequently the only way for analysts to get the keys that are used to protect cryptographic file systems. In this paper the authors show that cryptographic keys in memory are vulnerable to exploitation *after the computer is turned off*. The authors show that the contents of dynamic RAM are retained seconds, and sometimes minutes, after power is turned off. By skillfully timing the memory dump, the data can be obtained as long as necessary. And while most

Download and install open source software from...

<http://sourceforge.net/projects/libewf/>

- For reading EWF files on Unix/MacOS



<http://afflib.org/>

- AFFLIB Disk Image Tools
- Bloom Filter Tools
- Bulk Extractor



<http://sleuthkit.org/>

- Forensic File Systems
FAT, NTFS, HFS, etc.



This is an introductory tutorial!

Theory, Science and Tools

8:30 - 10:00 Introduction

- Introduction to Digital Forensics & The Law

10:00 - 10:30 Coffee

10:30 - 12:00 Data Analysis

- Unicode, File Formats & File Identification

12:00 - 1:30 Lunch

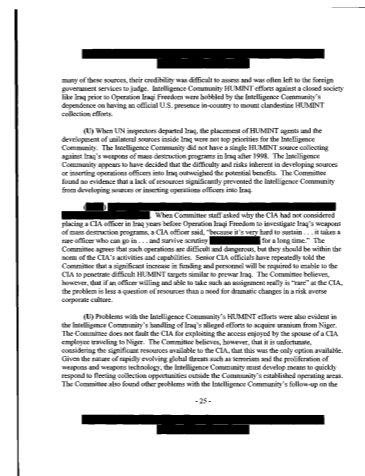
1:30 - 3:00 Disk Forensics

- Disk Imaging
- File Carving
- Sleuth Kit

3:00 - 3:30 Coffee

3:30 - 5:00 Big Finish

- Documents & Metadata
- Memory Forensics
- Anti-Forensics



What's on the disk?

/corp — "corpus" of freely redistributable forensic files

/linux — RPM for testdisk & photorec

/macos — compiled testdisk & photorec

/slides — this presentation

/src — source code for forensic tools

- afflib
- NPS bloom package & frag_find
- bulk_extractor
- fiwalk
- libewf
- sleuthkit
- tcpflow

/papers — forensic papers

- Legal aspects, Fake Photos, Memory Analysis, File Fingerprinting, Time, and more.



/corp

```
$ ls -l
```

```
total 334272
```

```
      83731 Dec  3 21:56 diversity-p5.pdf*
    3994096 Dec  3 21:56 diversityanalysis.pdf*
  13476769 Dec  4 15:13 honeynet-2001-scan15.raw.zip*
   68340753 Dec  3 21:56 nitroba-norm.pcap*
  31129600 Apr 13  2009 nps-2009-canon2-gen6.raw*
   35551648 Jan  6  2009 ntfs1-gen2.aff*
 189702777 Dec  3 20:25 xp-laptop-2005-07-04-1430.zip*
```

```
$
```



Forensics & Digital Investigations

Forensic Definitions
The “Magic Camera”
Hypothesis-based investigation

“Forensics” has two meanings.

fo·ren·sics n. (used with a sing. verb)

1.The art or study of formal debate; argumentation.

2.The use of science and technology to investigate and establish facts in criminal or civil courts of law.

(American Heritage Dictionary, 4th Edition)



Courts settle disputes, redress grievances, and mete out punishment

Deciding some disputes requires the use of physical evidence:

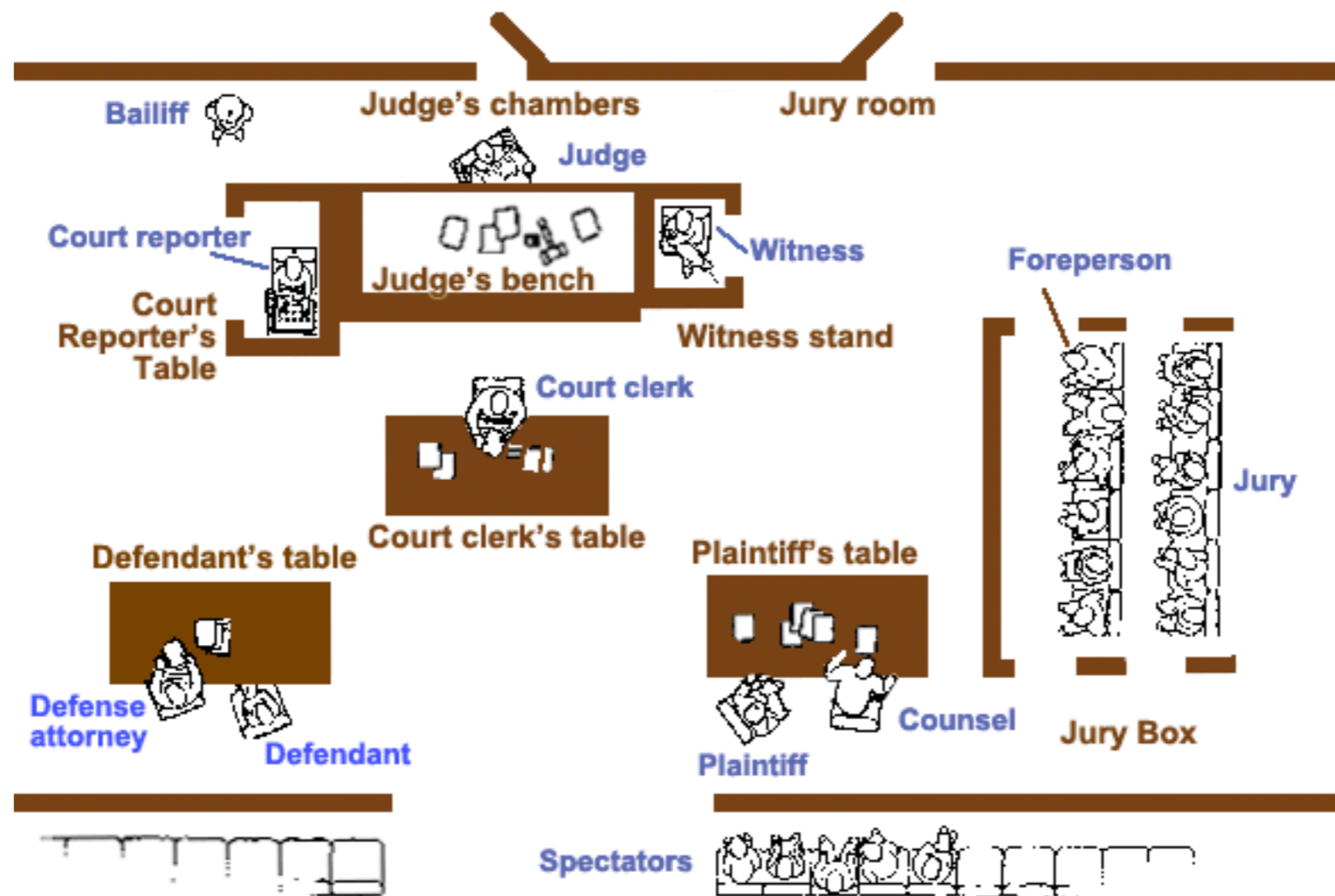
- Fingerprints
- DNA
- Handwriting
- Polygraph



Judges and Juries can't examine physical evidence

- They don't have the expertise.
- Evidence may be open to interpretation.

Forensic experts interpret scientific evidence.



US Courts employ an *adversarial process*.

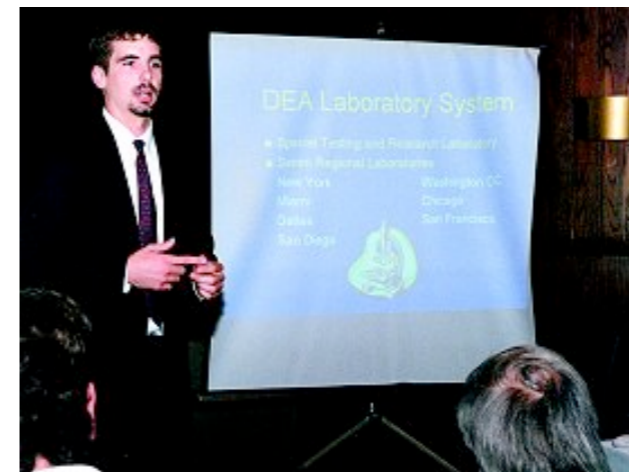
Each side hires its own experts.

In some cases, the court may hire a third expert for the judge.

Investigators for the **prosecution**: conduct the investigation and build the case.

Criminal Digital Investigators:

- Sworn Law Enforcement Officer
- Writes search warrants
- Receives computers, cameras, and other evidence
- Acquires & Analyzes data
- Presents findings
- Prepares report
- Testifies in court



Investigators for the defense: **rebut the evidence and create doubt.**

Defense Experts:

- Employed by the Defense
- Works with defense attorney
- Receives evidence from law enforcement
- May conduct independent investigation, but usually funds do not permit
- May work with other experts.
- May testify in court.



Even photographs may require interpretation

When were these photographs taken? Were they faked?



Stalin's Soviet Union tampered with the past.

After Abel Yenukidze was shot during the purges of 1936-1938, his image was removed from official photographs.

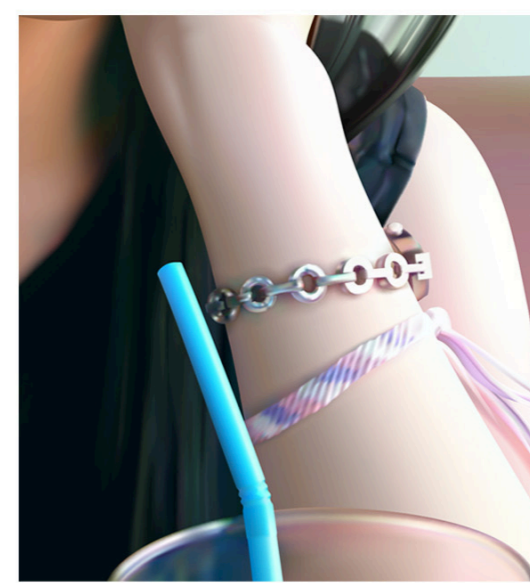
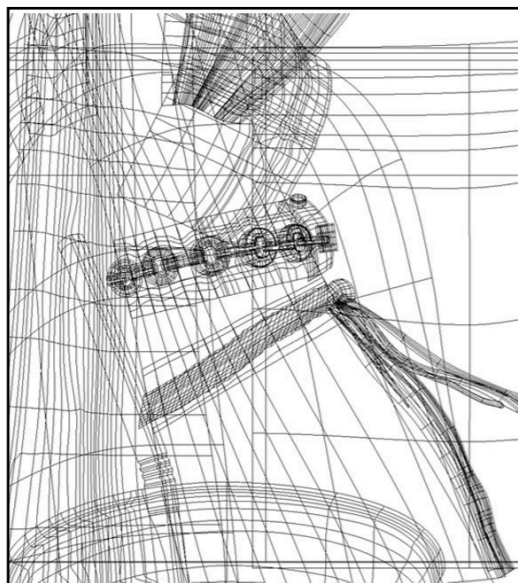
- The Commissar Vanishes
- <http://www.hoover.org/publications/digest/3531641.html>
- http://www.newseum.org/berlinwall/commissar_vanishes/



Computer graphics are so good that it is easy to mistake a simulated photo for reality.

Pisan Kaewma 2006

<http://www.illustratorworld.com/artwork/1336/>



Digital media makes it *easy* to create forgeries.

Most photos are not "doctored" —
but most photographs are not taken into court.

If someone has an *interest* in the interpretation
of a photo, there is a higher chance of it being
modified.

This is true of all evidence.

- “Digital Doctoring: can we trust photographs?”
Hany Farid,
In *Deception: Methods, Motives, Contexts and
Consequences*, 2007



Figure 3. The published (top) and original LA Times photographs showing a British soldier and Iraqi civilians.

Digital forensics applies this process to computers.

Here are some definitions for computer forensics:

- “Involves the preservation, identification, extraction, documentation, and interpretation of computer data.”
(Computer Forensics: Incident Response Essentials, Warren Kruse and Jay Heiser.)
- “The scientific examination, analysis, and/or evaluation of digital evidence in legal matters.”
(Scientific Working Group on Digital Evidence, <http://www.swgde.org>)



Digital Evidence requires interpretation by experts. But what's *Digital Evidence*?

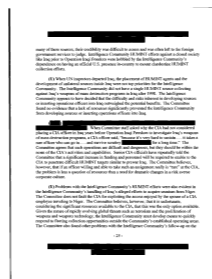
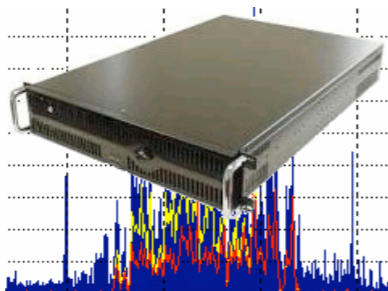
“Information stored or transmitted **in binary form** that may be relied upon in court.” [Int02]

“Information of probative value that is stored or transmitted **in binary form.**” [Sci05]

“Information and data of investigative value that is **stored on or transmitted by a computer.**” [Ass05]

“Any data **stored or transmitted using a computer** that support or refute a theory of how an offense occurred or that address critical elements of the offense such as intent or alibi.” [Cas04]

If it involves computers, it's probably digital evidence.



"Digital evidence" can be:

1) evidence of a crime; 2) the crime itself.

Evidence of a crime:

- Financial Records.
- Emails documenting a conspiracy.
- Photographs of a murder.

The crime itself:

- Computer break-ins.
- Denial-of-service attacks.
- Distribution of child pornography.
- Emailed threats.



Digital evidence may be collected before a crime is known to have taken place!

Computer forensics allows investigators to:

- Discover how a crime was committed
- Determine extent of damage
- Gather evidence of illegal activity
- Confirm/disprove an alibi

We can prime systems to record evidence in advance:

- Log files — Recording events.
- Network Forensics — Packet Capture
- EnCase Enterprise — Remote Disk Forensics

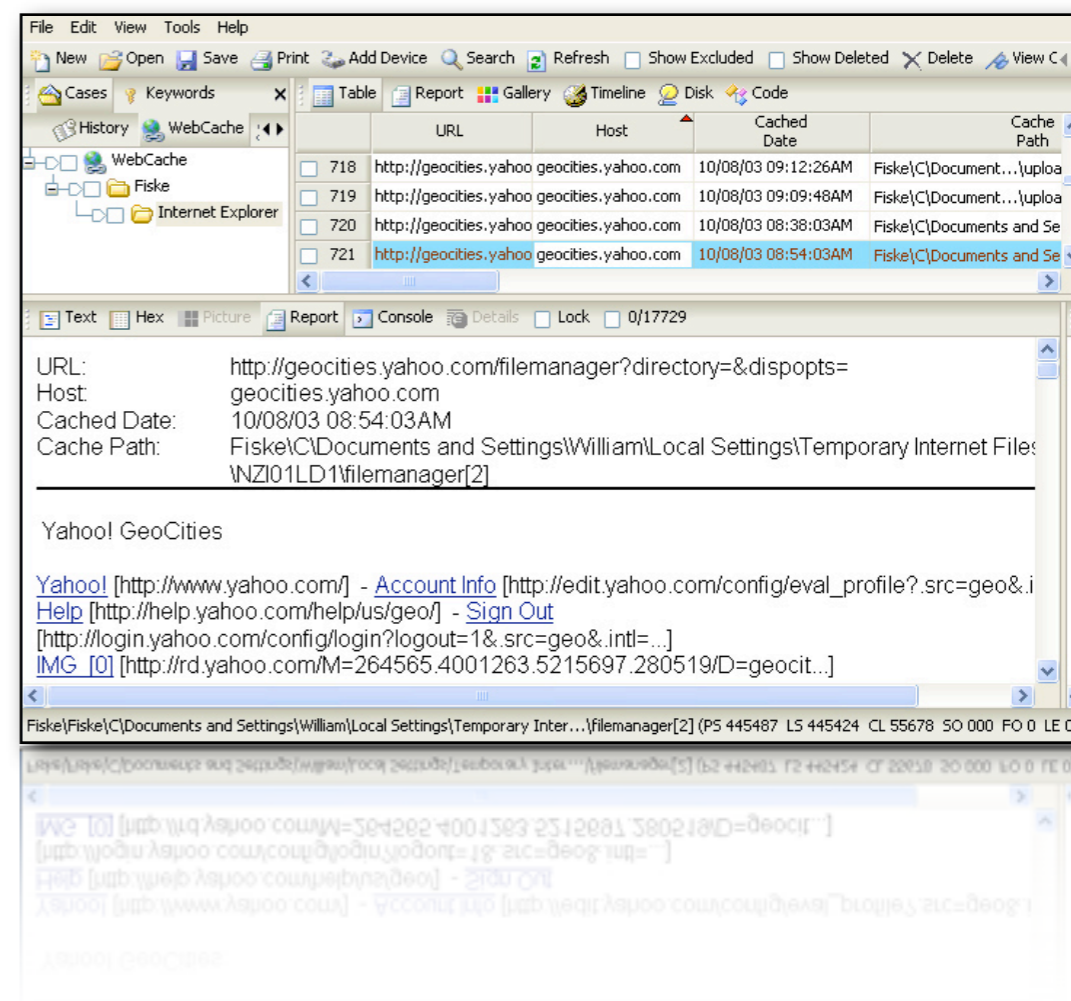
Digital Forensics is like a magic camera

Tools can go “back in time...”

- ✓ View previous versions of files
- ✓ Recover “deleted” files
- ✓ Find out what was typed
- ✓ Discover visited websites

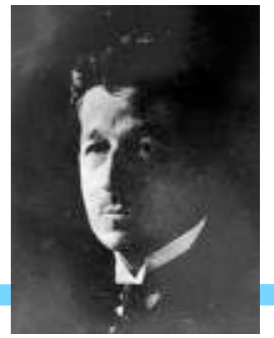
Why does this work?

- ✓ free() doesn't erase memory
- ✓ DELETE doesn't erase files
- ✓ newfs and FORMAT* don't clear disks
- ✓ Computers keep extensive logs
- ✓ Most data is not encrypted



This is very different from “traditional” (blood & bullet) forensics.

Traditional forensics is dominated by the Locard Exchange Principle



Dr. Edmund Locard (1877-1966) - "Every contact leaves a trace."

Wherever he steps, whatever he touches, whatever he leaves, even unconsciously, will serve as a silent witness against him.

Not only his fingerprints or his footprints, but his hair, the fibers from his clothes, the glass he breaks, the tool mark he leaves, the paint he scratches, the blood or semen he deposits or collects.

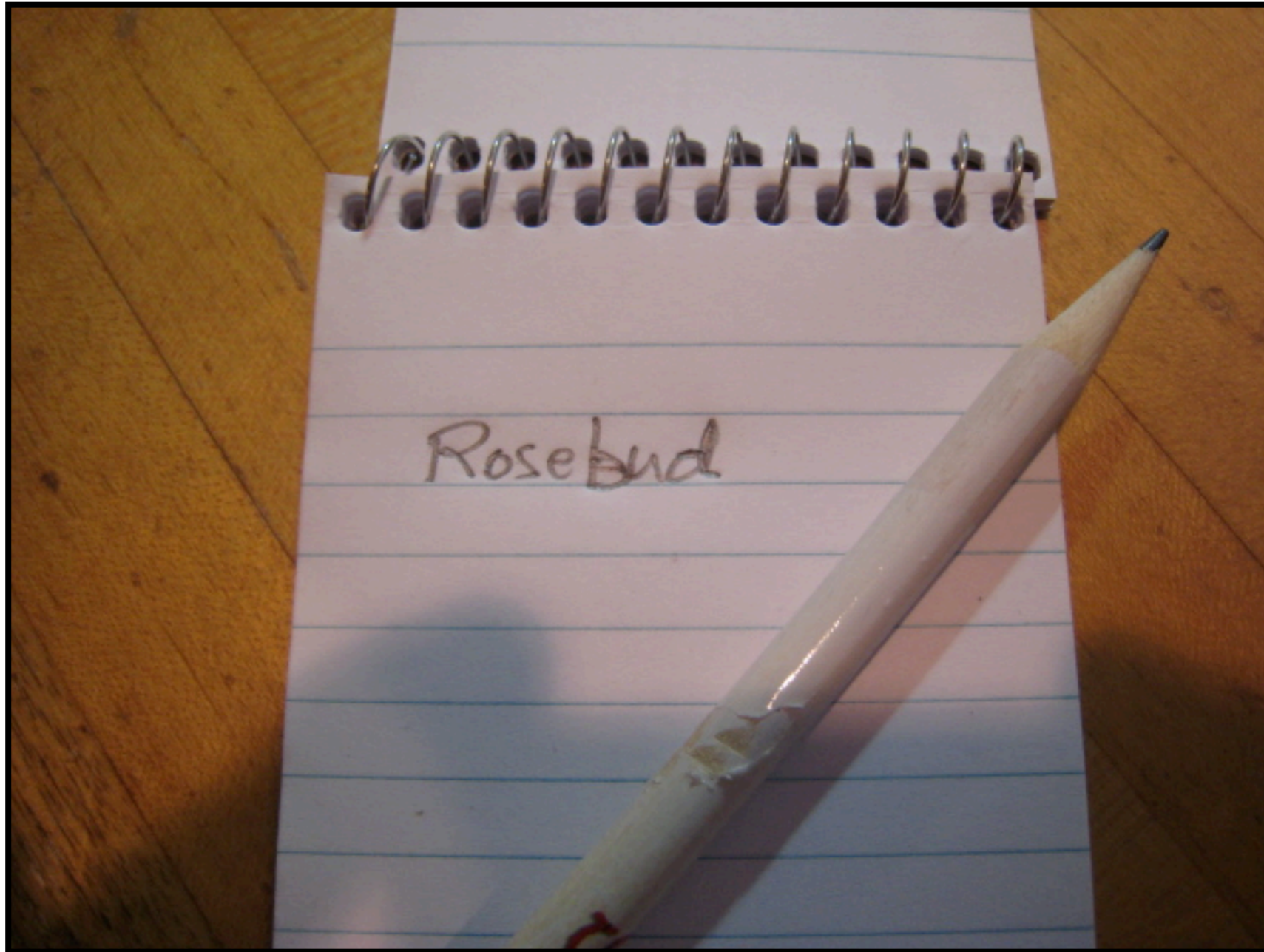
All of these and more, bear mute witness against him. This is evidence that does not forget. It is not confused by the excitement of the moment. It is not absent because human witnesses are. It is factual evidence.

Physical evidence cannot be wrong, it cannot perjure itself, it cannot be wholly absent. Only human failure to find it, study and understand it, can diminish its value.

"Exchange Principle:"
You can't erase pencil writing on paper...

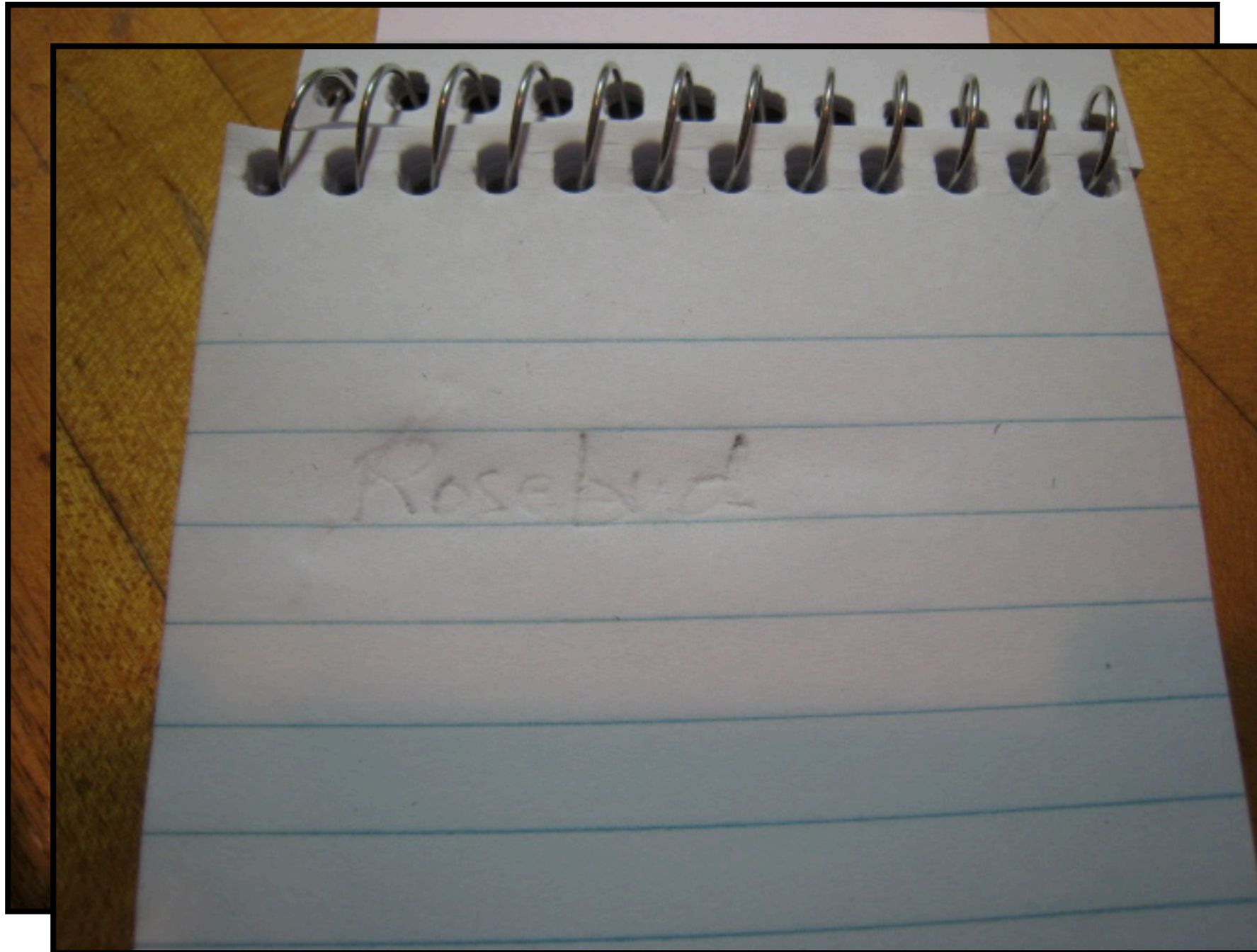
"Exchange Principle:"

You can't erase pencil writing on paper...

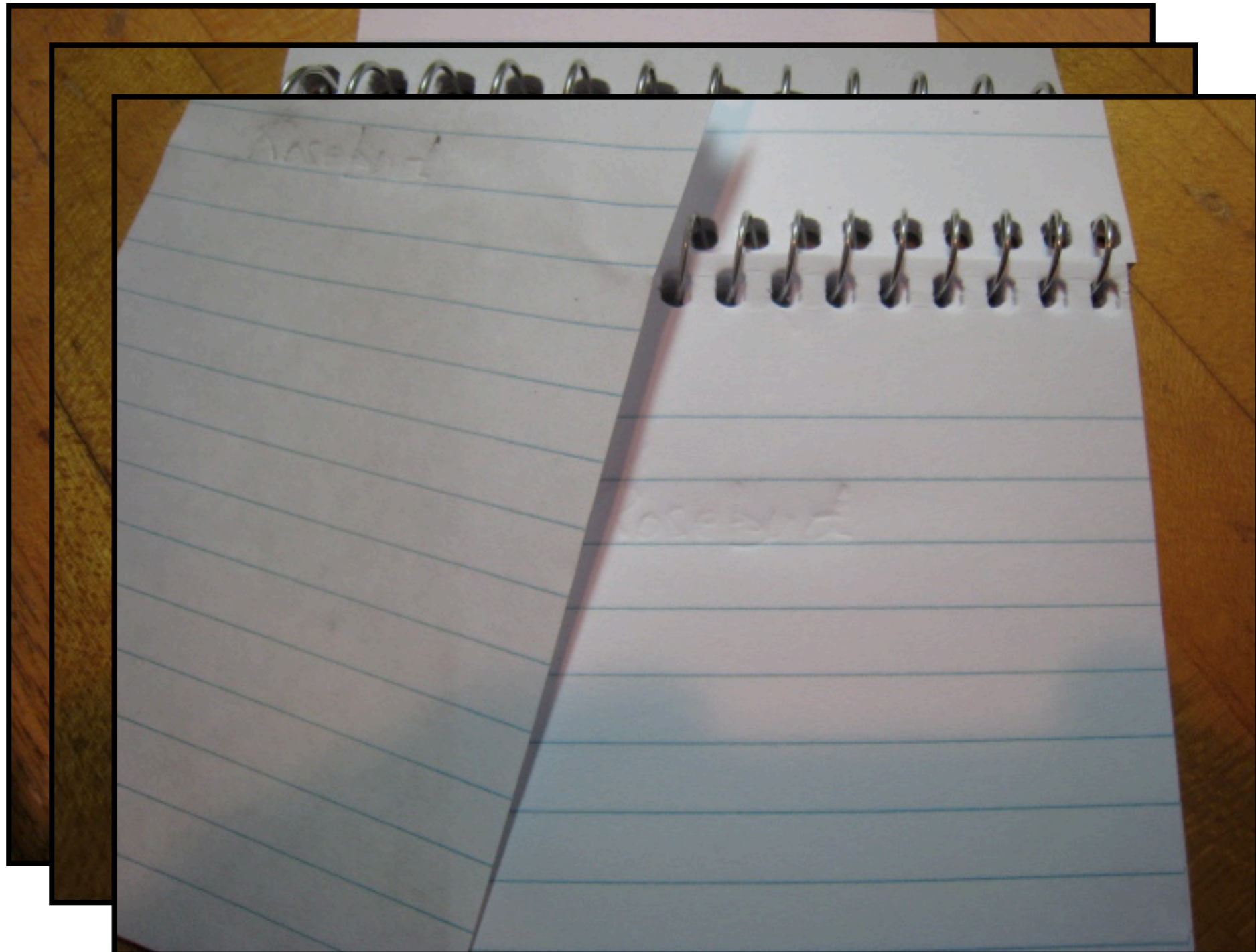


"Exchange Principle:"

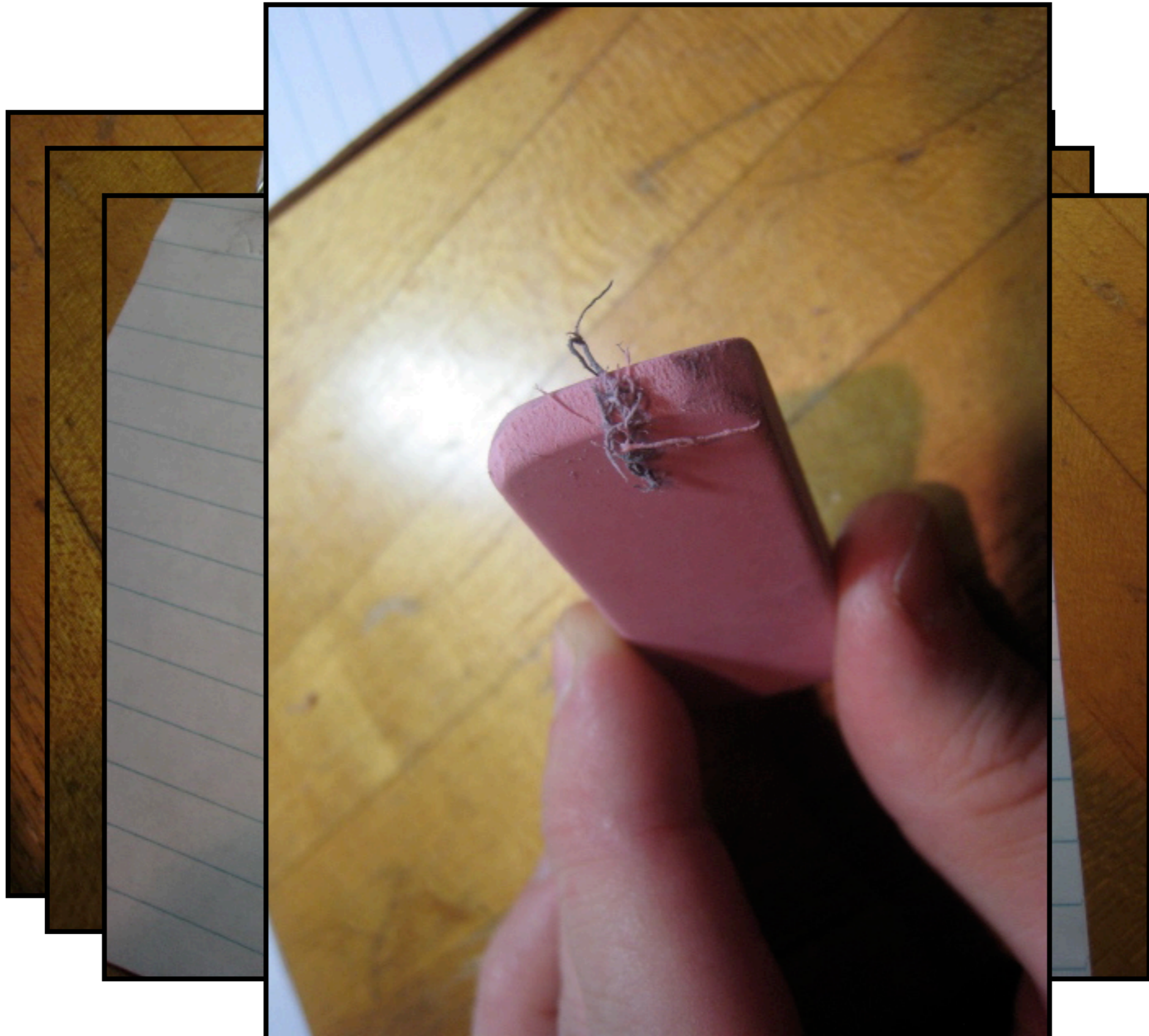
You can't erase pencil writing on paper...



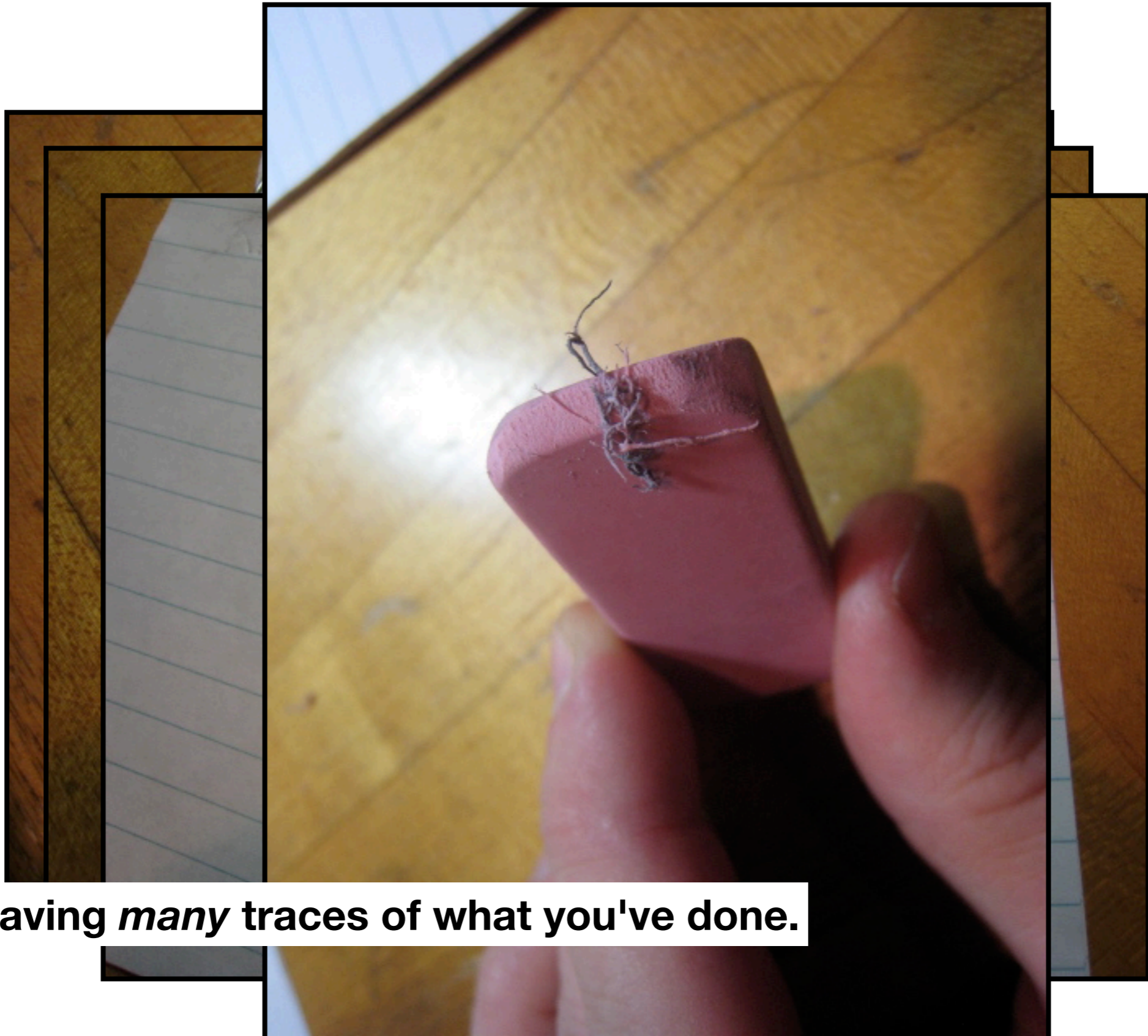
"Exchange Principle:"
You can't erase pencil writing on paper...



"Exchange Principle:"
You can't erase pencil writing on paper...



"Exchange Principle:"
You can't erase pencil writing on paper...



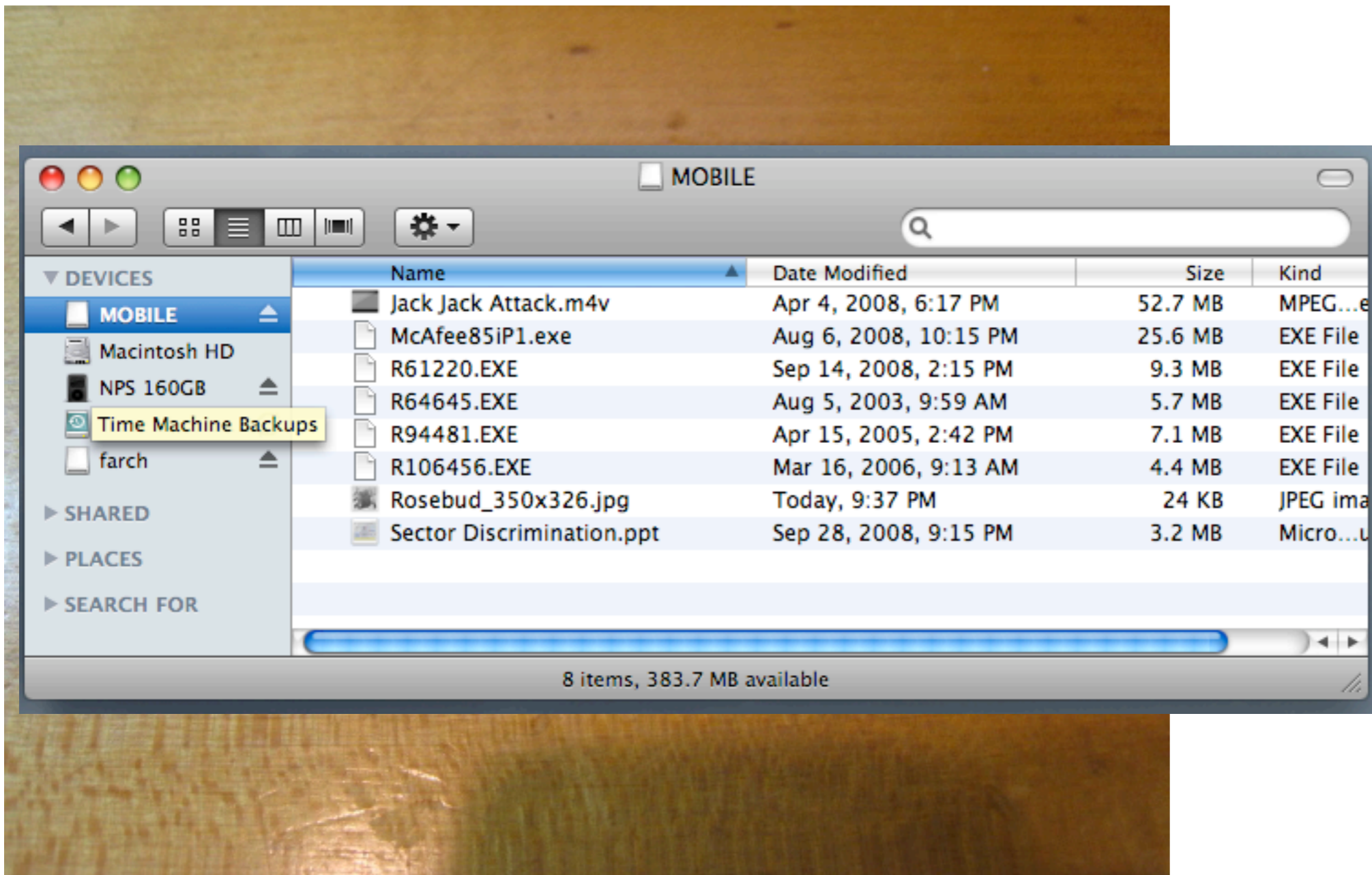
... without leaving *many* traces of what you've done.

The Exchange Principle doesn't apply to bits.

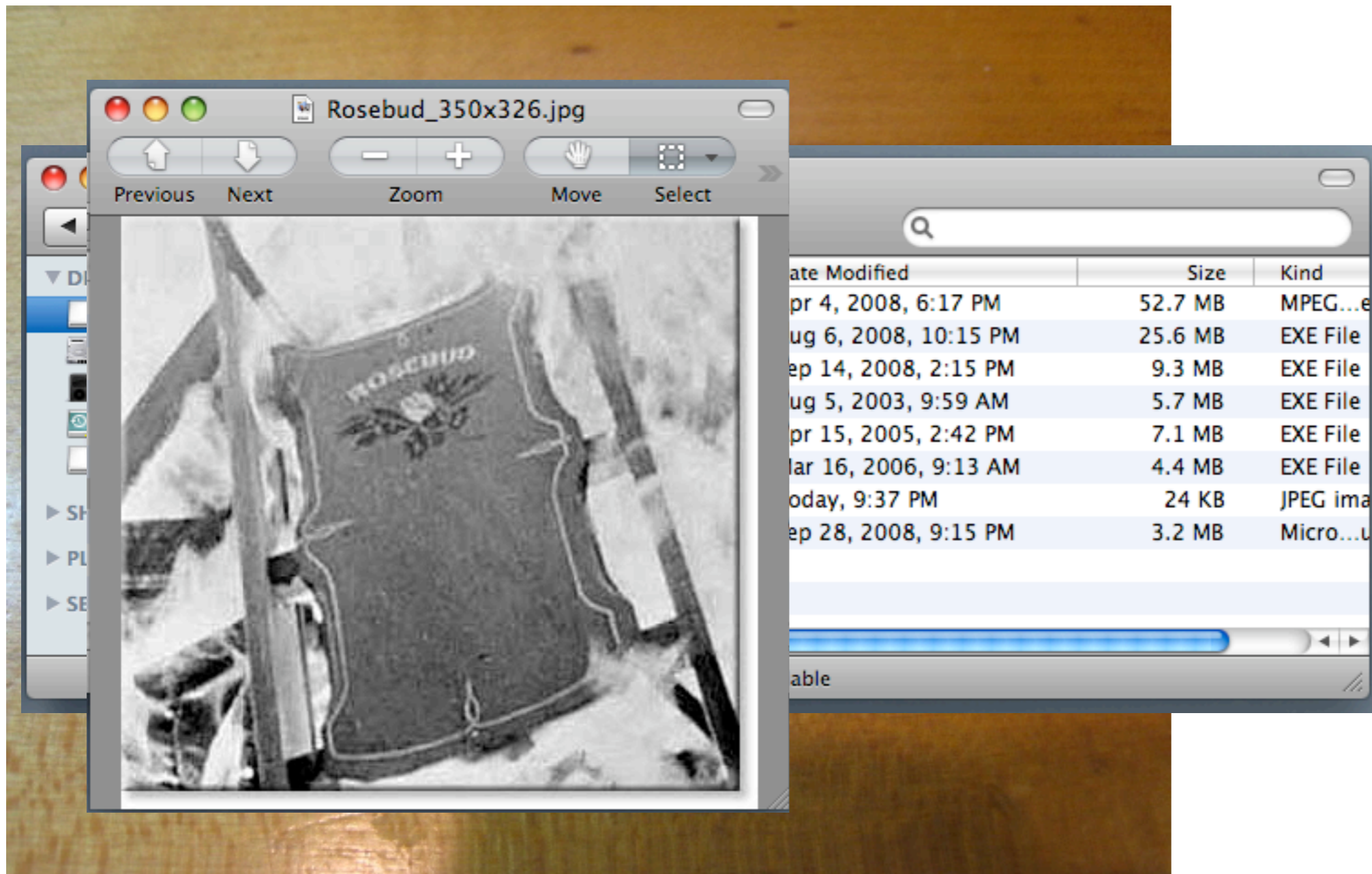
The Exchange Principle doesn't apply to bits.



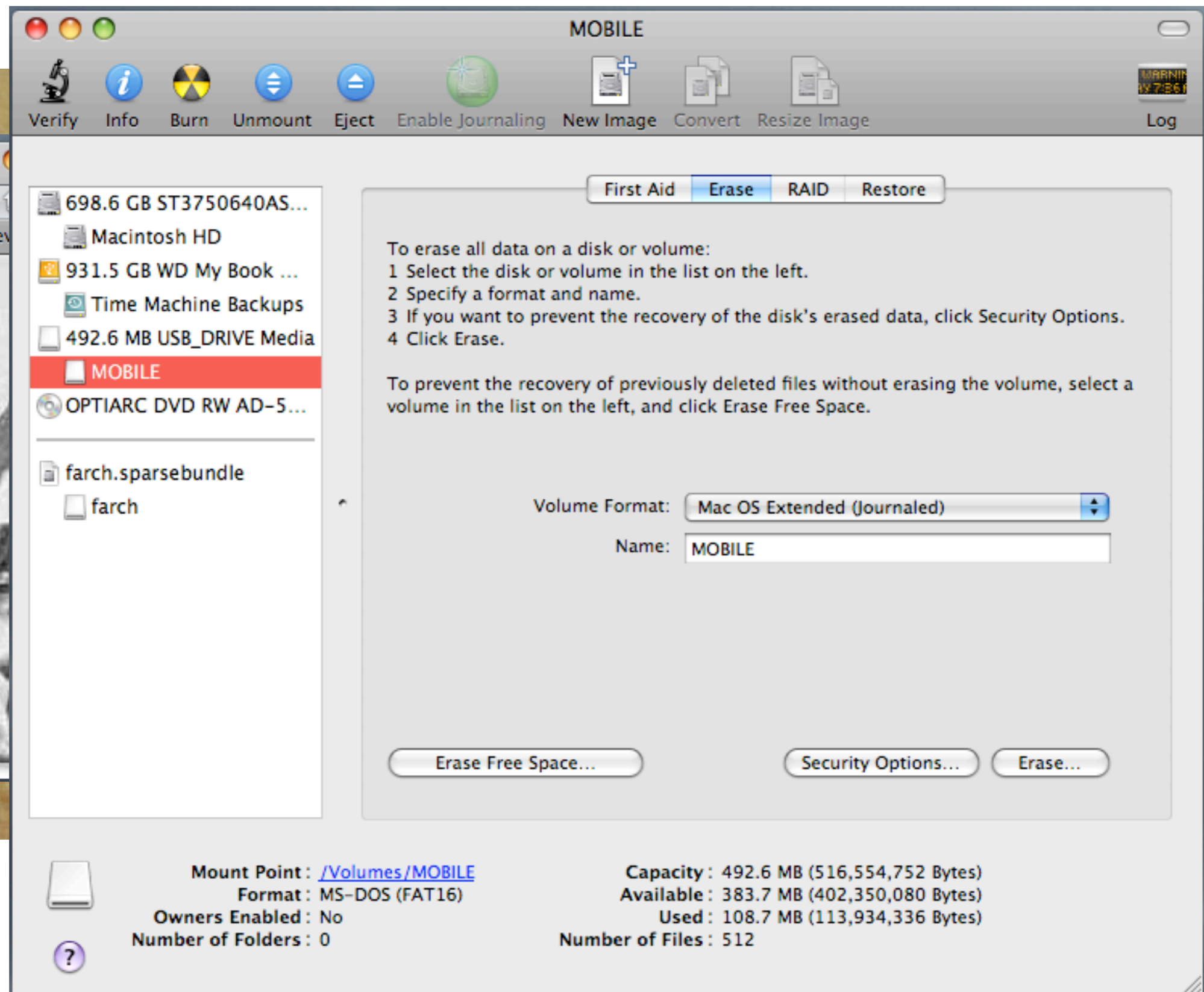
The Exchange Principle doesn't apply to bits.



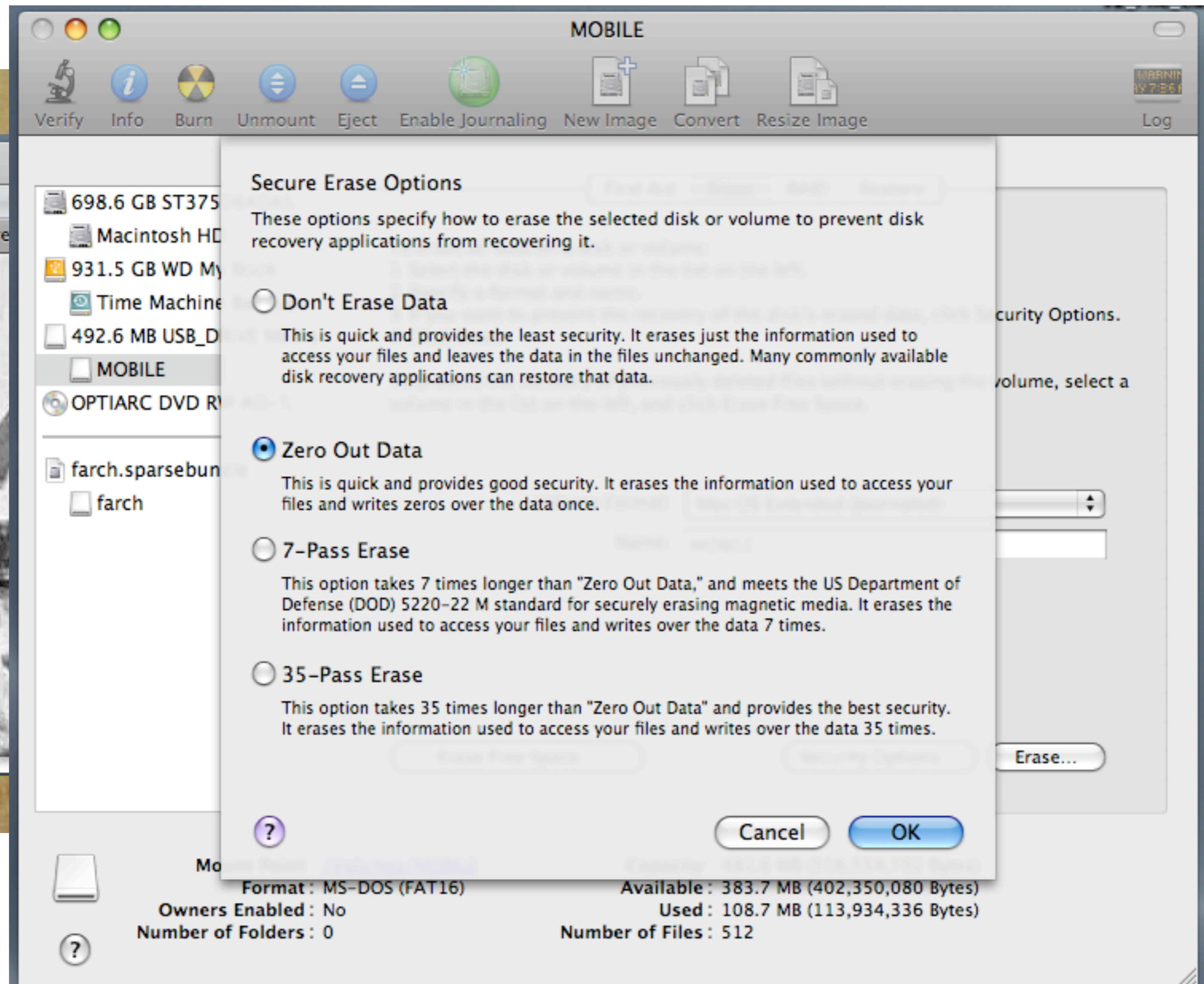
The Exchange Principle doesn't apply to bits.



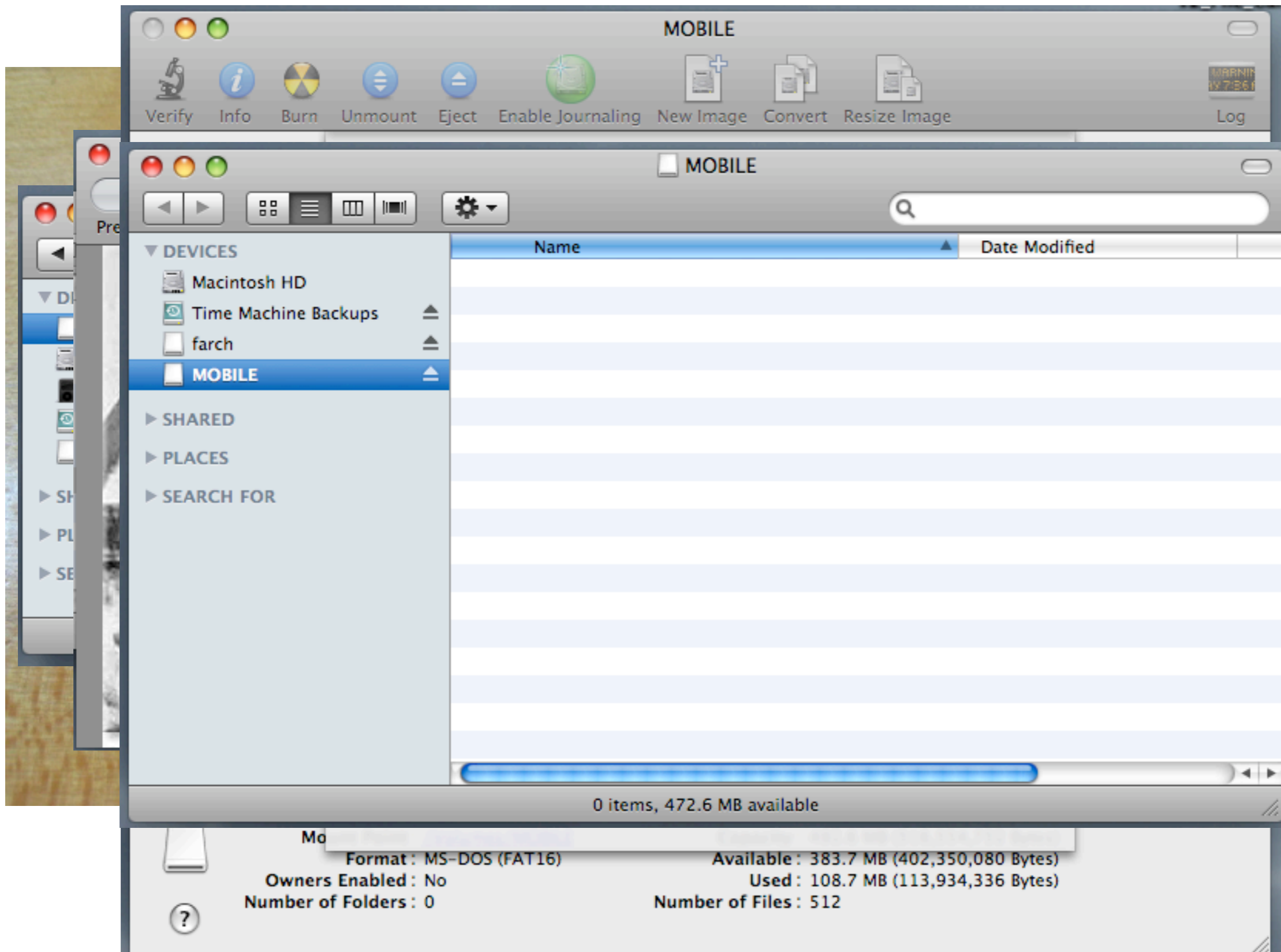
The Exchange Principle doesn't apply to bits.



The Exchange Principle doesn't apply to bits.



The Exchange Principle doesn't apply to bits.



The fundamental problem with digital evidence:
It can't be trusted.

The fundamental problem with digital evidence:
It can't be trusted.

Consider this printout:

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
```

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
```

```
07:17 AM Black:~/Downloads$ ls -l
```

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
07:17 AM Black:~/Downloads$ ls -l
total 74
```

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
07:17 AM Black:~/Downloads$ ls -l
total 74
-rw-r--r--  1 simsong  simsong  73625 Jun 16 06:30 afyi.pdf
```

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
07:17 AM Black:~/Downloads$ ls -l
total 74
-rw-r--r--  1 simsong  simsong  73625 Jun 16 06:30 afyi.pdf
07:18 AM Black:~/afyi$
```

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
07:17 AM Black:~/Downloads$ ls -l
total 74
-rw-r--r--  1 simsong  simsong  73625 Jun 16 06:30 afyi.pdf
07:18 AM Black:~/afyi$
```

Question: **When was afyi.pdf** downloaded?

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
07:17 AM Black:~/Downloads$ ls -l
total 74
-rw-r--r--  1 simsong  simsong  73625 Jun 16 06:30 afyi.pdf
07:18 AM Black:~/afyi$
```

Question: **When was afyi.pdf** downloaded?

Possible explanations:

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
07:17 AM Black:~/Downloads$ ls -l
total 74
-rw-r--r--  1 simsong  simsong  73625 Jun 16 06:30 afyi.pdf
07:18 AM Black:~/afyi$
```

Question: **When was afyi.pdf** downloaded?

Possible explanations:

- The file was downloaded at 7:17, but Safari set the timestamp to be the time on the server.

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
07:17 AM Black:~/Downloads$ ls -l
total 74
-rw-r--r--  1 simsong  simsong  73625 Jun 16 06:30 afyi.pdf
07:18 AM Black:~/afyi$
```

Question: **When was afyi.pdf** downloaded?

Possible explanations:

- The file was downloaded at 7:17, but Safari set the timestamp to be the time on the server.
- The file was downloaded on a different day and moved into the directory.

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
07:17 AM Black:~/Downloads$ ls -l
total 74
-rw-r--r--  1 simsong  simsong  73625 Jun 16 06:30 afyi.pdf
07:18 AM Black:~/afyi$
```

Question: **When was afyi.pdf** downloaded?

Possible explanations:

- The file was downloaded at 7:17, but Safari set the timestamp to be the time on the server.
- The file was downloaded on a different day and moved into the directory.
- The computer's clock was changed before the file was downloaded.

The fundamental problem with digital evidence: It can't be trusted.

Consider this printout:

```
07:16 AM Black:~/Downloads$ ls -l
07:17 AM Black:~/Downloads$ ls -l
total 74
-rw-r--r--  1 simsong  simsong  73625 Jun 16 06:30 afyi.pdf
07:18 AM Black:~/afyi$
```

Question: **When was afyi.pdf** downloaded?

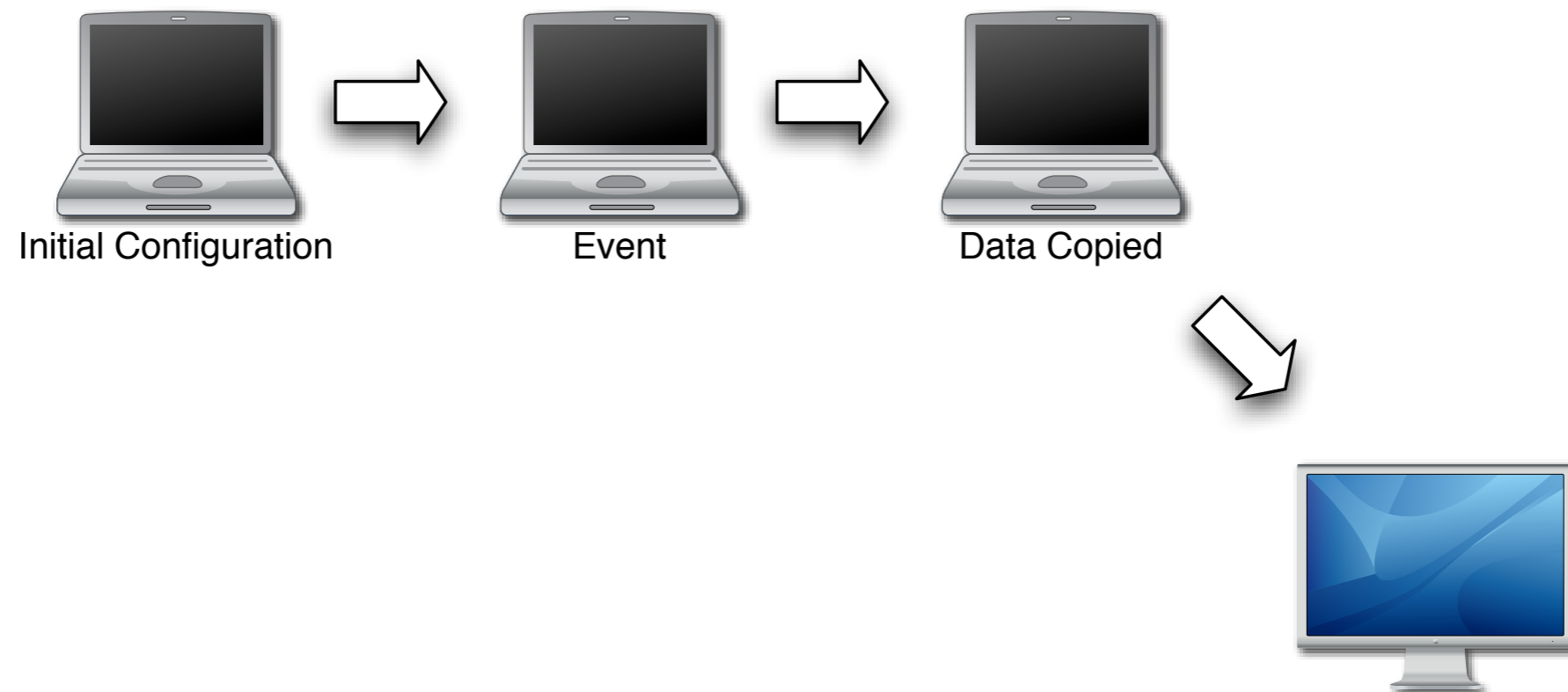
Possible explanations:

- The file was downloaded at 7:17, but Safari set the timestamp to be the time on the server.
- The file was downloaded on a different day and moved into the directory.
- The computer's clock was changed before the file was downloaded.
- The whole example was faked.

When we look at a computer system, we build a *hypothesis* about the computer's past.

The hypothesis makes assumptions about:

- The hardware under investigation.
- The software under investigation.
- The flow of time.
- The movement of the evidence
- The system being used to investigate the data



Usually the assumptions are accurate.
Sometimes they are not.



Other assumptions:

- Event didn't fake the initial configuration.
Attacker creates a new vulnerability to hide one actually used.
- All attacker's code & data was copied.
Program might be hidden in the graphics co-processor.
- Analysis system is faithful and accurate.
Attacker's tools might be invisible due to a bug in the forensic tool.



A Hypothesis-Based Approach to Digital Forensic Investigations,
Brian D. Carrier, PhD. Thesis, Purdue University, 2006

The “Daubert Standard” is designed to keep “junk science” out of the courts.

Daubert turns federal judges “gatekeepers.”

Daubert v. Merrell Dow Pharmaceuticals, 509 US 57

- Birth defects caused by Bendectin

Evidence must be “relevant”

- So as not to waste the court’s time or confuse matters)

Evidence must be “reliable” (ie, scientific)

- Subject to peer review (has been published)
- Generally accepted by the relevant professional community
- Standards for the technique’s operation
- Known error rate



The “Daubert Standard” is designed to keep “junk science” out of the courts.

Daubert turns federal judges “gatekeepers.”

Daubert v. Merrell Dow Pharmaceuticals, 509 US 57

- Birth defects caused by Bendectin

Evidence must be “relevant”

- So as not to waste the court’s time or confuse matters)

Evidence must be “reliable” (ie, scientific)

- Subject to peer review (has been published)
- Generally accepted by the relevant professional community
- Standards for the technique’s operation
- Known error rate



Most digital evidence today does not meet this standard.

In court, testimony is governed by the Federal Rules of Evidence

Article I. General Provisions

Article II. Judicial Notice

Article III. Presumptions In Civil Actions And Proceedings

Article IV. Relevancy And Its Limits

Article V. Privileges

Article VI. Witnesses

Article VII. Opinions and Expert Testimony

Article VIII. Hearsay

Article IX. Authentication and Identification

Article X. Contents of Writings, Records and Photographs

Article XI. Miscellaneous Rules



US Federal Rules of Evidence

Article VII regulates the testimony of “experts”

Rule 702. Testimony by Experts

- Qualified experts are allowed to testify

Rule 703. Bases of Opinion Testimony by Experts

- Experts can use *any* information they wish, even hearsay

Rule 704. Opinion on Ultimate Issue

- Experts are allowed to give an opinion on the "ultimate issue."

Rule 705. Disclosure of Facts or Data Underlying Expert Opinion

- Experts can give their opinion without presenting the facts.

Rule 706. Court Appointed Experts

- The court is allowed to appoint its own experts (but they rarely do)

These rules apply in the Federal Court; many states follow the rules as well

- <http://www.law.cornell.edu/rules/fre/>

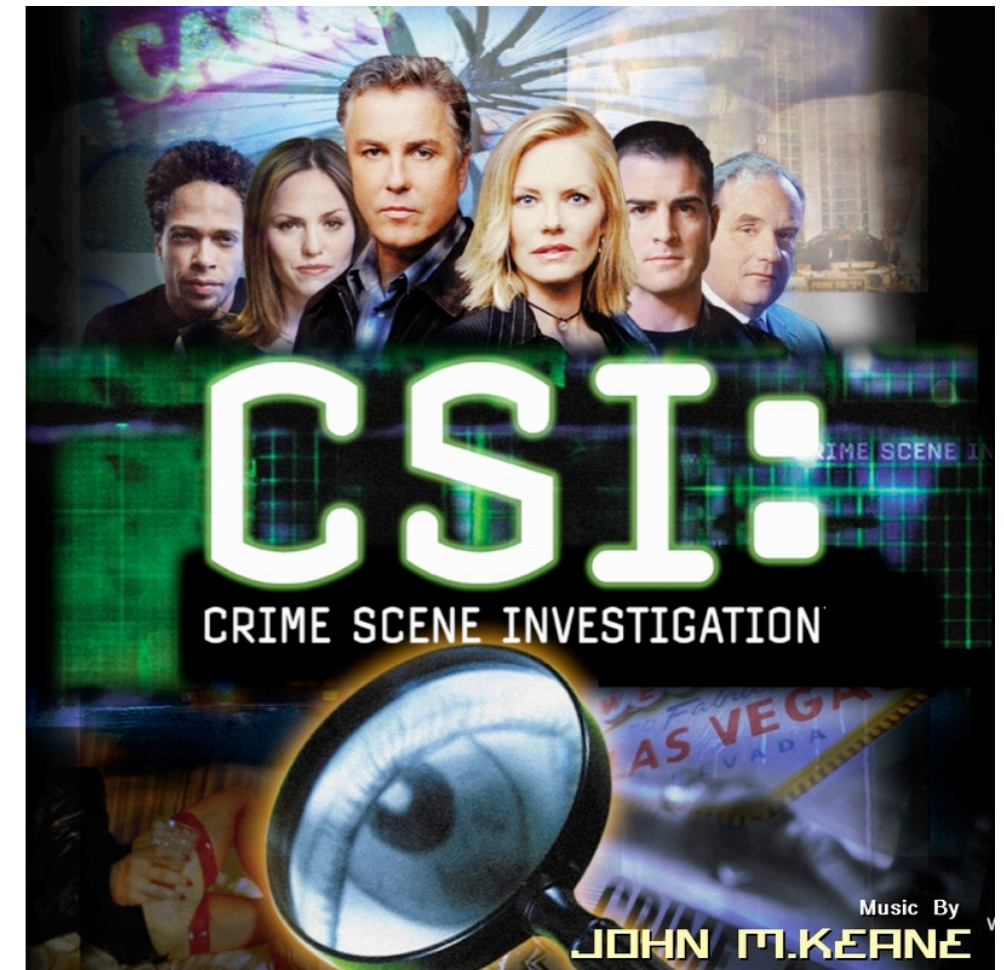
The “CSI Effect” causes victims and juries to have unrealistic expectations.

On TV:

- Forensics is swift.
- Forensics is certain.
- Human memory is reliable.
- Presentations are highly produced.

TV digital forensics:

- Every investigator is trained on every tool.
- Correlation is easy and instantaneous.
- There are no false positives.
- Overwritten data can be recovered.
- Encrypted data can usually be cracked.
- It is impossible to delete anything.



The reality of digital forensics is less exciting.

There are lots of problems:

- Data that is overwritten cannot be recovered.
- Encrypted data usually can't be decrypted.
- Forensics rarely answers questions or establishes guilt.
- Forensics rarely provides specific information about a specific subject
- Tools crash a lot.

But that doesn't really matter, because:

- Most digital forensics is used to find child pornography.
- When the pornography is found, most suspects plead guilty.



Forensics has many uses beyond the courtroom.

Data Recovery.

Testing and Evaluating:

- System Performance
- Privacy Properties & Tools
- Security Policies

Spot-check regulatory compliance:

- Internal information flows
- Data flow across network boundaries
- Disposal policies

Performance Evaluation

Information Exploitation & Data mining



Conclusion:

Forensics and Digital Investigations

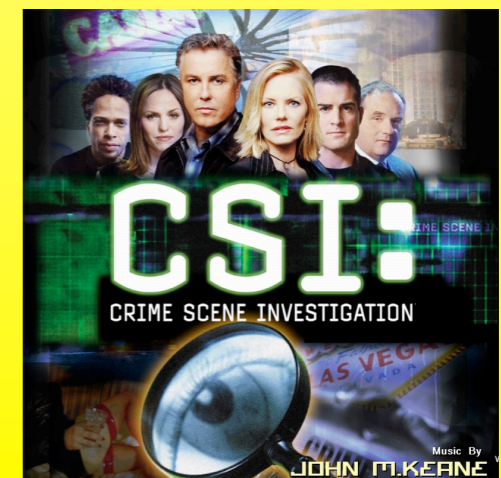
Scientific evidence requires *interpretation* to get it into a court room.



Digital evidence is easy to fake.



The main use of digital forensics today is



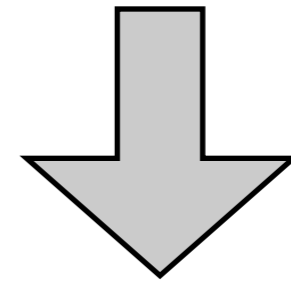


The Forensic Process

From computer to courtroom.
The Investigation.
The “Hacker Defense.”

There are five basic steps to computer forensics.

1. Preparation (you, not the data)
2. Collection (the data)
3. Examination
4. Analysis
5. Reporting



Source:

Electronic Crime Scene Investigation Guide
National Institute of Justice



Step 1: Preparation

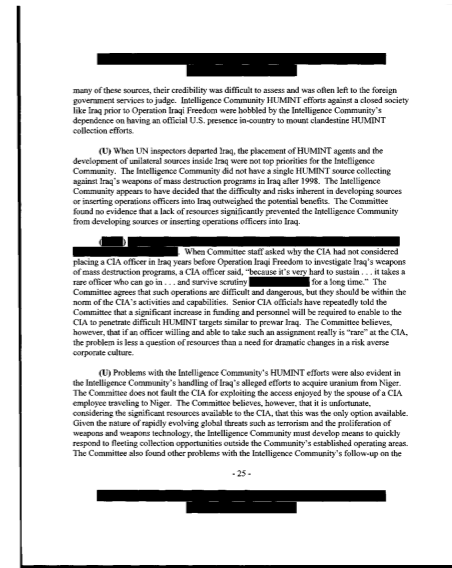
Identify potential sources of evidence

Computer system components:

- Hard drives
- Memory / flash / configuration
- Physical configuration

Other data sources:

- Web Pages
- Files
- Communication networks



Each source may need its own personnel, tools, training procedures.

One of the most difficult tasks is determining what to include & exclude.

Step 2:

Collect and Preserve the evidence

If the activity is ongoing, your choices include:

- Passive Monitoring
- Experimental Probing

If the activity is over, choices include:

- Make an **exact** copy
- Seize the equipment, then make a copy.

Issues to consider:

- Do you legally have access to the data?
- What tools are used? Are they validated?
- Is the copy accurate? Is it complete?
- How can you prove that the copy wasn't modified at a later time?



Investigators need access to the digital evidence.

Consent Searches — The owner gives consent.

- No warrant or probable cause required.
- Officers not required to warn people of their right to withhold consent (Schneckloth v. Bustamonte).
- Employers can give consent for an employee.
- Spouse may give consent to marital property.
- Roommates can give consent for each other.
- Parents can give consent for children under 18, and sometimes over 18.
- System administrators can give consent, but are regulated under the Electronic Communications Privacy Act.



Investigators need access to the digital evidence.

✓ **Consent Searches** — The owner gives consent.

Warrant Searches

- Police swears an oath that proves probable cause or hearsay information.
- Warrant defines the terms of what may be searched and seized.



Investigators need access to the digital evidence.

- ✓ **Consent Searches** — The owner gives consent.
- ✓ **Warrant Searches** — Searches with legal authority

Warrantless Searches

- Everything else.
- May be legal in some cases (Customs, etc).
- Adherence to local laws may not matter.

Step 3: Examination.

Make evidence “visible” and eliminate excess.

Disk Analysis:

- Examine partitions and file systems
- Resident & delete files
- “Slack space” at end of files
- Unallocated space between files

File based evidence:

- Document text
- Deleted text
- Metadata (creation date; author fields; etc.)

Network Evidence:

- Device configuration
- Categorize packets; discard what isn't needed

Forensic tools should be validated.

Validation:

- A series of tests to show the tool can produce **consistent** and **accurate** results.
- Can discover errors in tools or procedures.

NIST's Information Technology Laboratory
Computer Forensics Tool Testing Program has
validated *some* tools.

- Record version number & SHA1 of tools.
- Use of unvalidated tools can get a case dismissed.

You can perform the study with unvalidated
tools, then show that the data is present using
validated tools.



<http://www.cftt.nist.gov/>

Step 4: Analyze to determine “significance and probative value”

Build a hypothesis about what happened.

Look for evidence to prove or disprove hypothesis.

Examples:

- Hypothesis: Suspect broke into a telephone company computer and stole confidential documents.
- Evidence: Hacker tools; confidential information from telco.

- Hypothesis: Suspect is arrested on suspicion of child pornography
- Evidence: Known child pornography on suspect's hard drive

BUT:

Investigators rarely look for counter-evidence.

Build a hypothesis about what happened.

Look for evidence to prove or disprove hypothesis.

Examples:

- Hypothesis: Suspect broke into a telephone company computer and stole confidential documents.
 - Evidence: Hacker tools; confidential information from telco.
 - **Counter Evidence: Documents publicly available**
-
- Hypothesis: Suspect is arrested on suspicion of child pornography
 - Evidence: Known child pornography on suspect's hard drive
 - **Counter Evidence: Hacker software allowing remote access**

Counter Evidence:
Trojan allowed remote access

Counter Evidence: Trojan allowed remote access

Aaron Caffrey, 19, charged w/ crashing systems at Port of Houston

- Caffrey claimed that hackers had broken into his computer and used it as a launch pad.
- Jury acquits, October 2003.

Counter Evidence:

Trojan allowed remote access

Aaron Caffrey, 19, charged w/ crashing systems at Port of Houston

- Caffrey claimed that hackers had broken into his computer and used it as a launch pad.
- Jury acquits, October 2003.

United States v. Michael McCourt, US Court of Appeals Case 061018P 11/24/06

- Defendant claimed hacker put hundreds of child pornography videos and stills on his computer.
- Appellate court ruled that defendant knew files were there, no matter how they got there.
- Hacker defense failed.
- <http://www.ca8.uscourts.gov/opndir/06/11/061018P.pdf>

Counter Evidence:

Trojan allowed remote access

Aaron Caffrey, 19, charged w/ crashing systems at Port of Houston

- Caffrey claimed that hackers had broken into his computer and used it as a launch pad.
- Jury acquits, October 2003.

United States v. Michael McCourt, US Court of Appeals Case 061018P 11/24/06

- Defendant claimed hacker put hundreds of child pornography videos and stills on his computer.
- Appellate court ruled that defendant knew files were there, no matter how they got there.
- Hacker defense failed.
- <http://www.ca8.uscourts.gov/opndir/06/11/061018P.pdf>

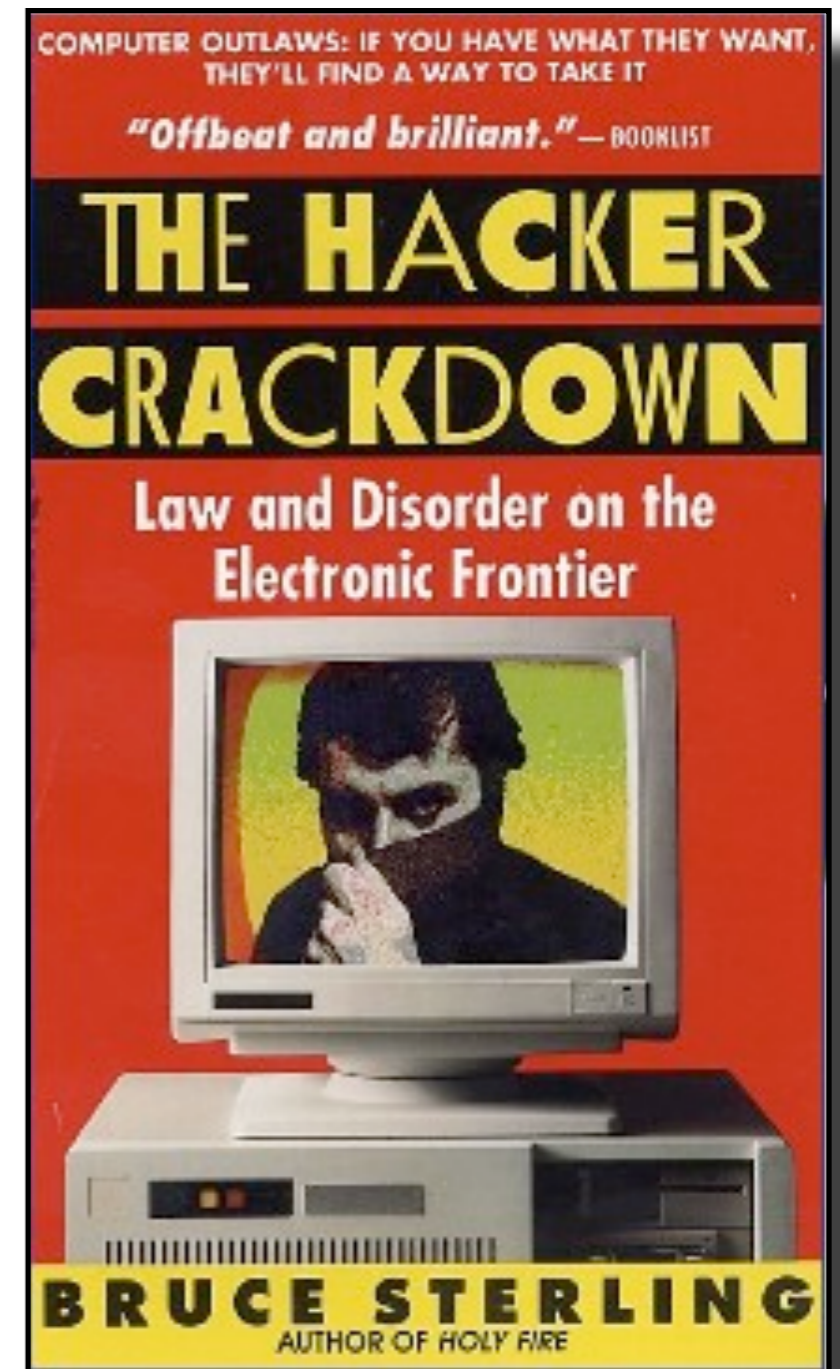
The Hacker/SODDI Defense: Indications & Contradictions

Try the hacker defense when:

- The system has a Trojan on it.
- The suspect has an alibi (e.g., lunch with a friend at a restaurant.)

Avoid the hacker defense when:

- The child porn was copied to DVD-Rs and stored under the suspect's bed.
- The suspect is a hacker or sysadmin (already has hacker tools; has knowledge to secure her own system.)



Step 5: Reporting and Testimony

Many kinds of testimony:

- Written reports
- Depositions
- Courtroom testimony



Testimony needs to include several key points:

- The tools used and procedures that were followed.
- The decision making process.
- What was found.
- Examiner's interpretation of what it means.

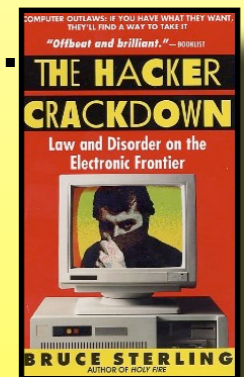
Conclusion: The Forensic Process

You must be prepared to do an investigation.

- No one person can do everything.
- If you aren't trained, the results can be thrown out.



Collect data *before* you try to explain what you have found.



Be sure to use validated tools and procedures (if possible).





Understanding “residual”
and “remnant” data.

REFERENCE

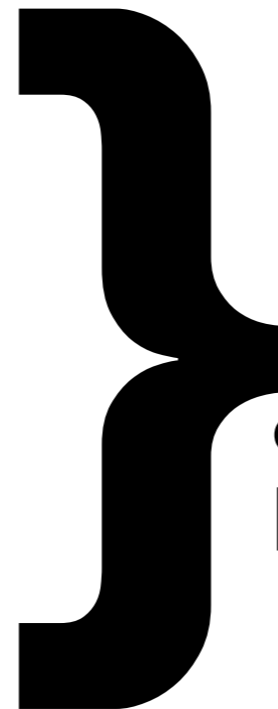
Only show before coffee #1...

Sectors on hard drives can be divided into three categories:

Resident Data

Deleted Data

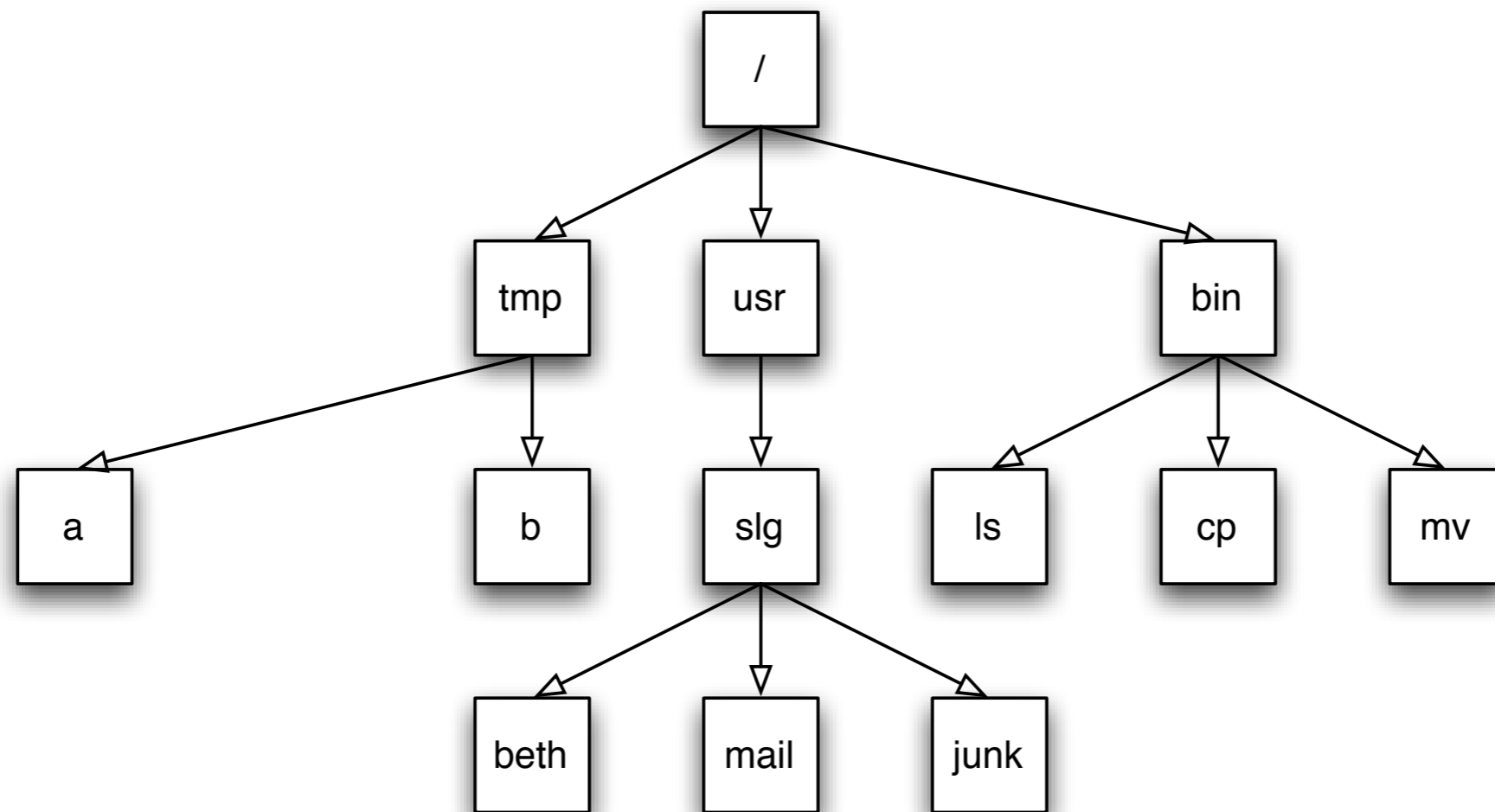
Uninteresting Data



user files
email messages
[temporary files]

blank sectors [OS files]

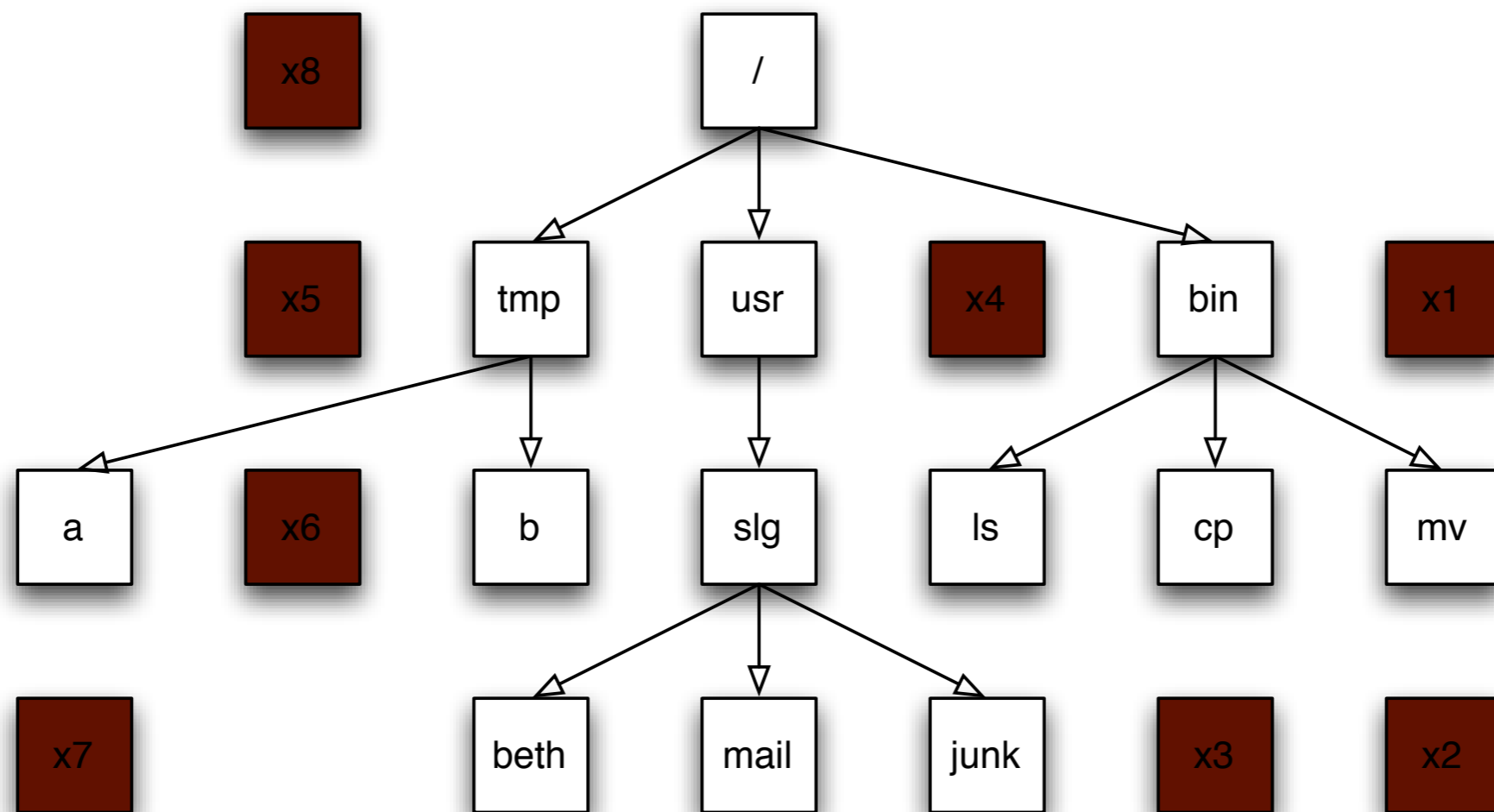
Data on a hard drive is arranged in sectors



Resident Data

= **data visible to the user**

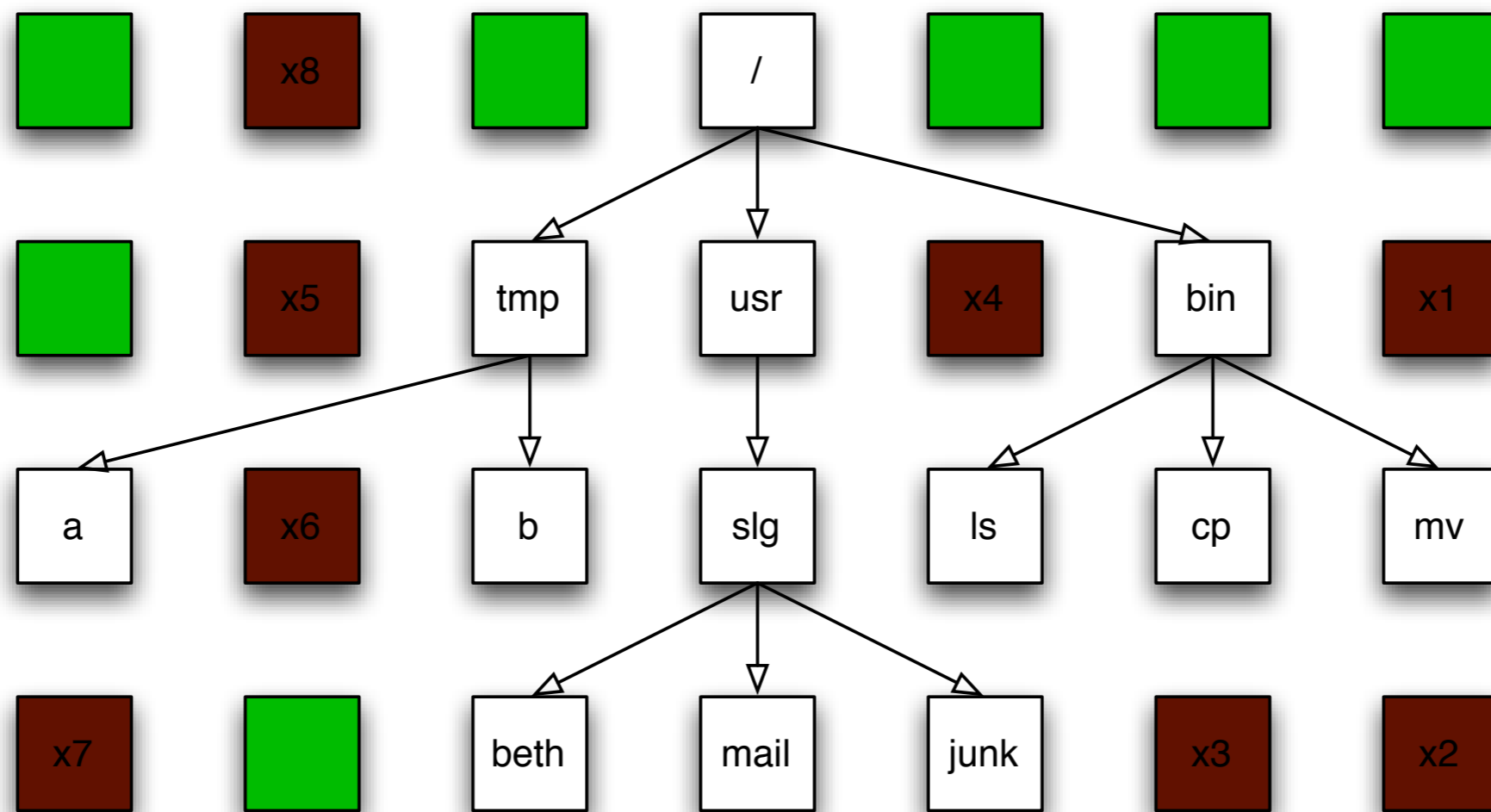
Data on a hard drive is arranged in sectors



Deleted Data

= files that were deleted.

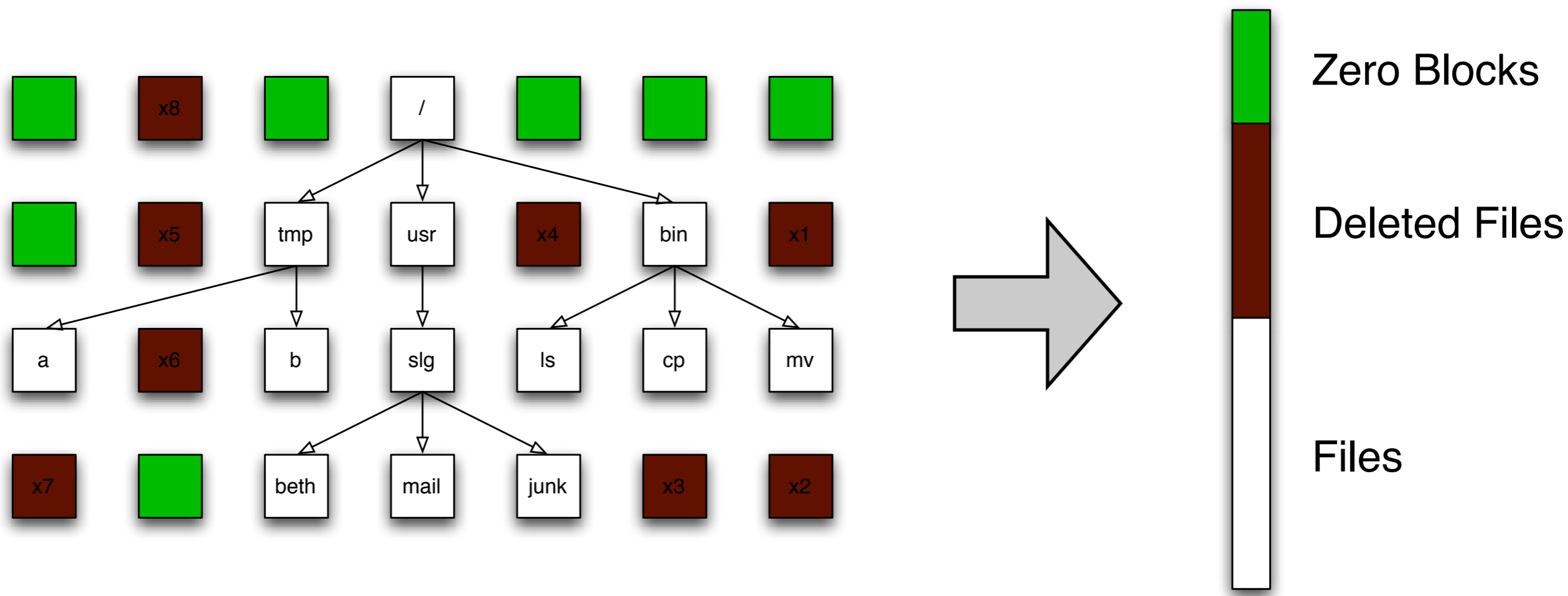
Data on a hard drive is arranged in sectors



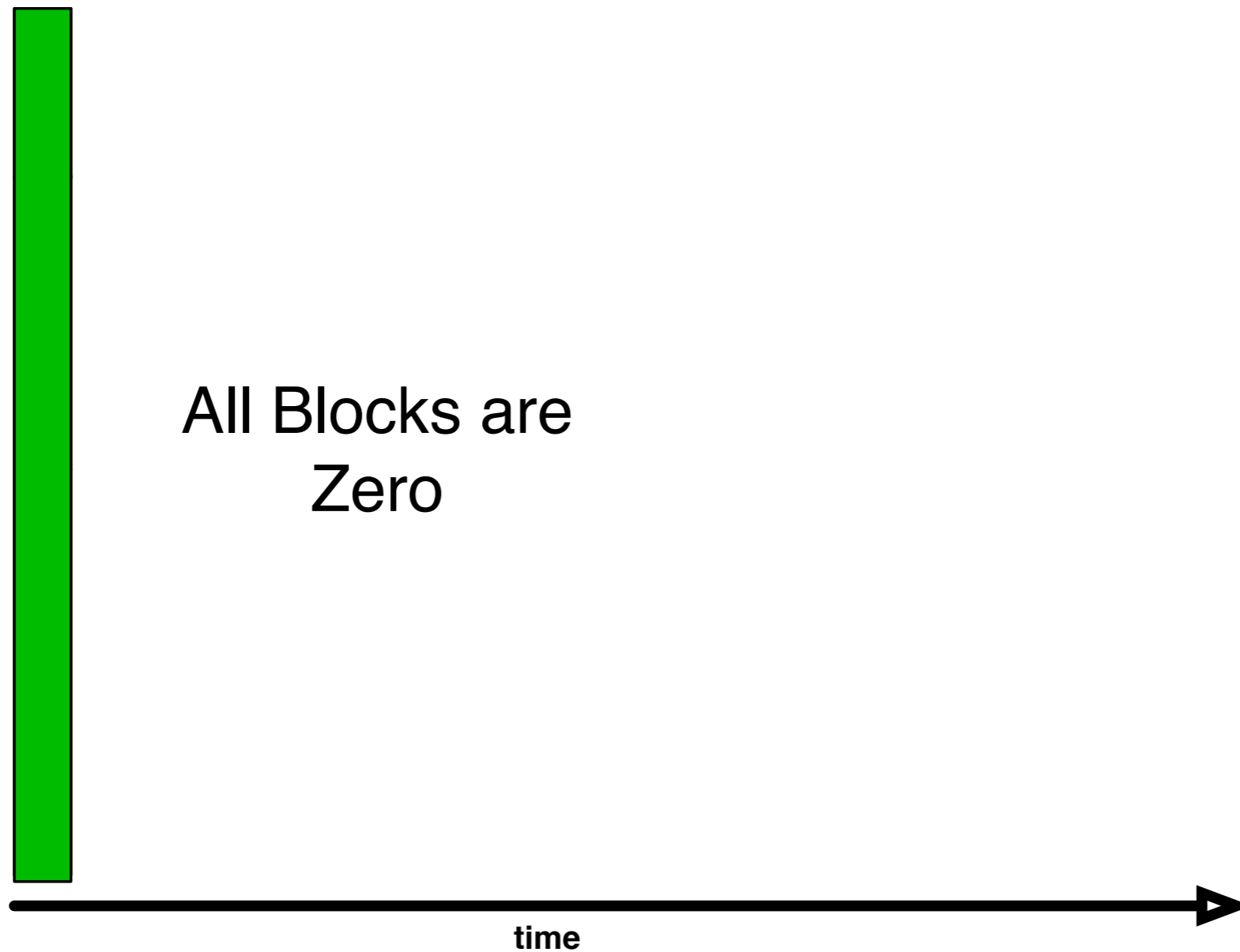
Uninteresting Data

= never written (or wiped clean)

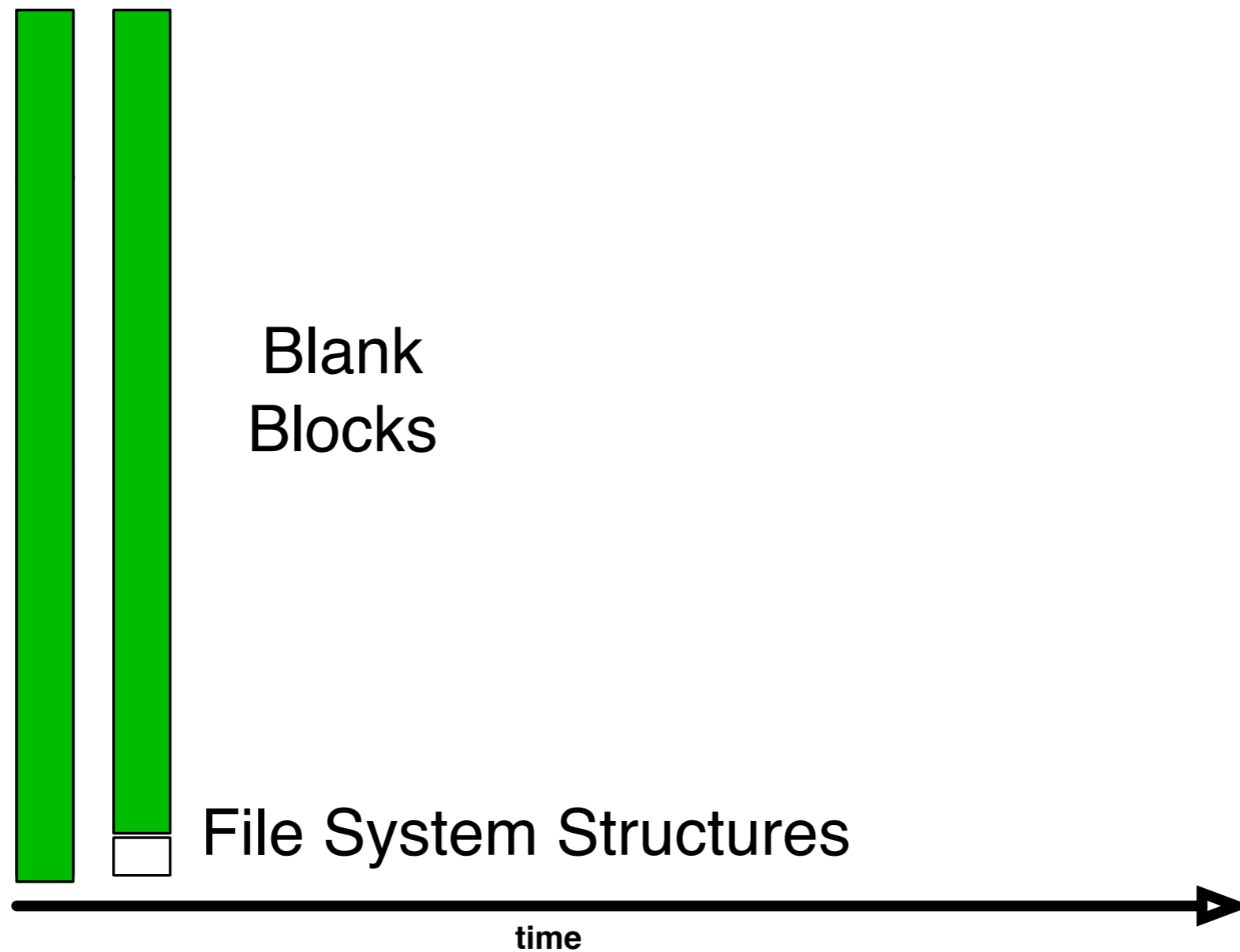
Stack the sectors:



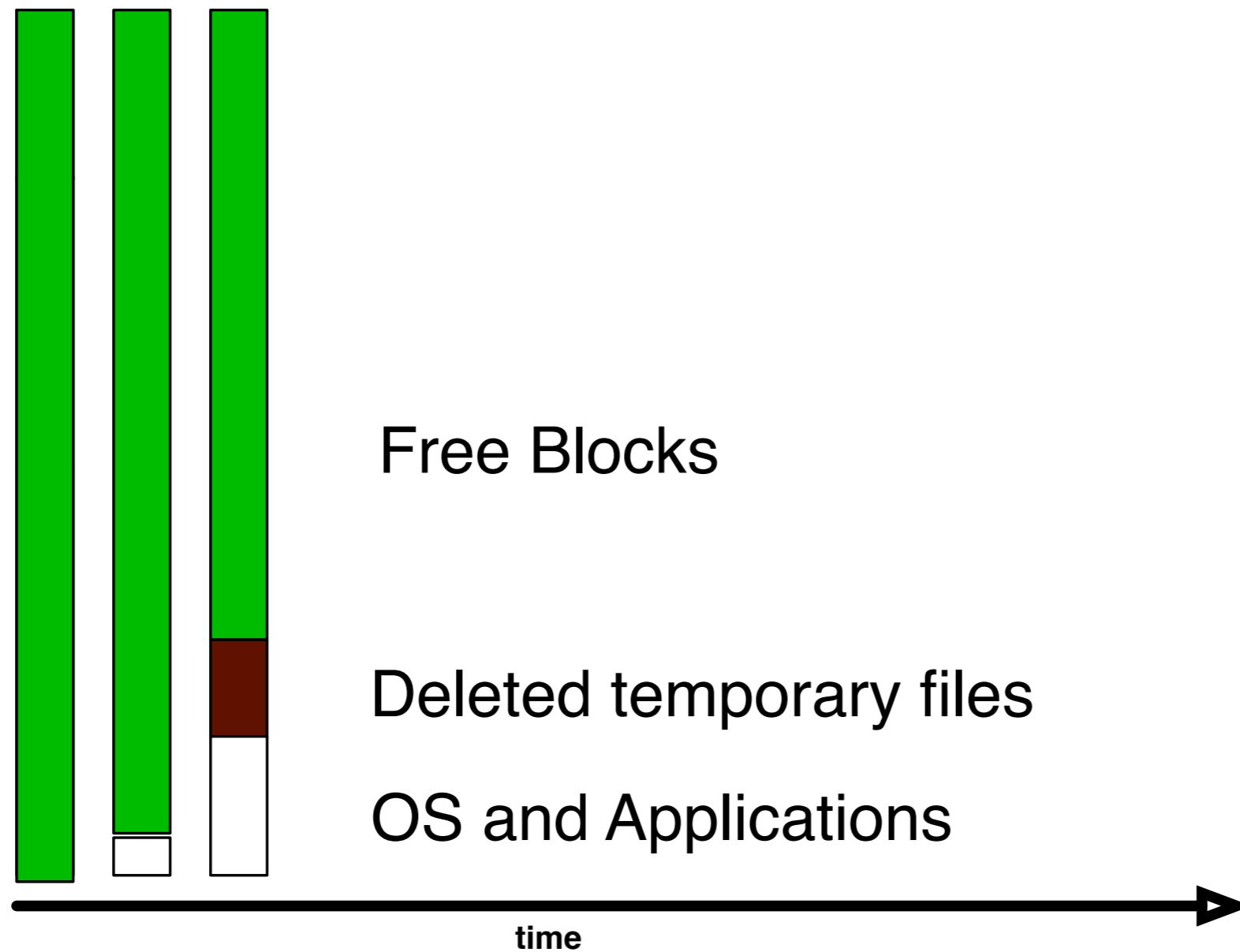
No data: The disk is factory fresh



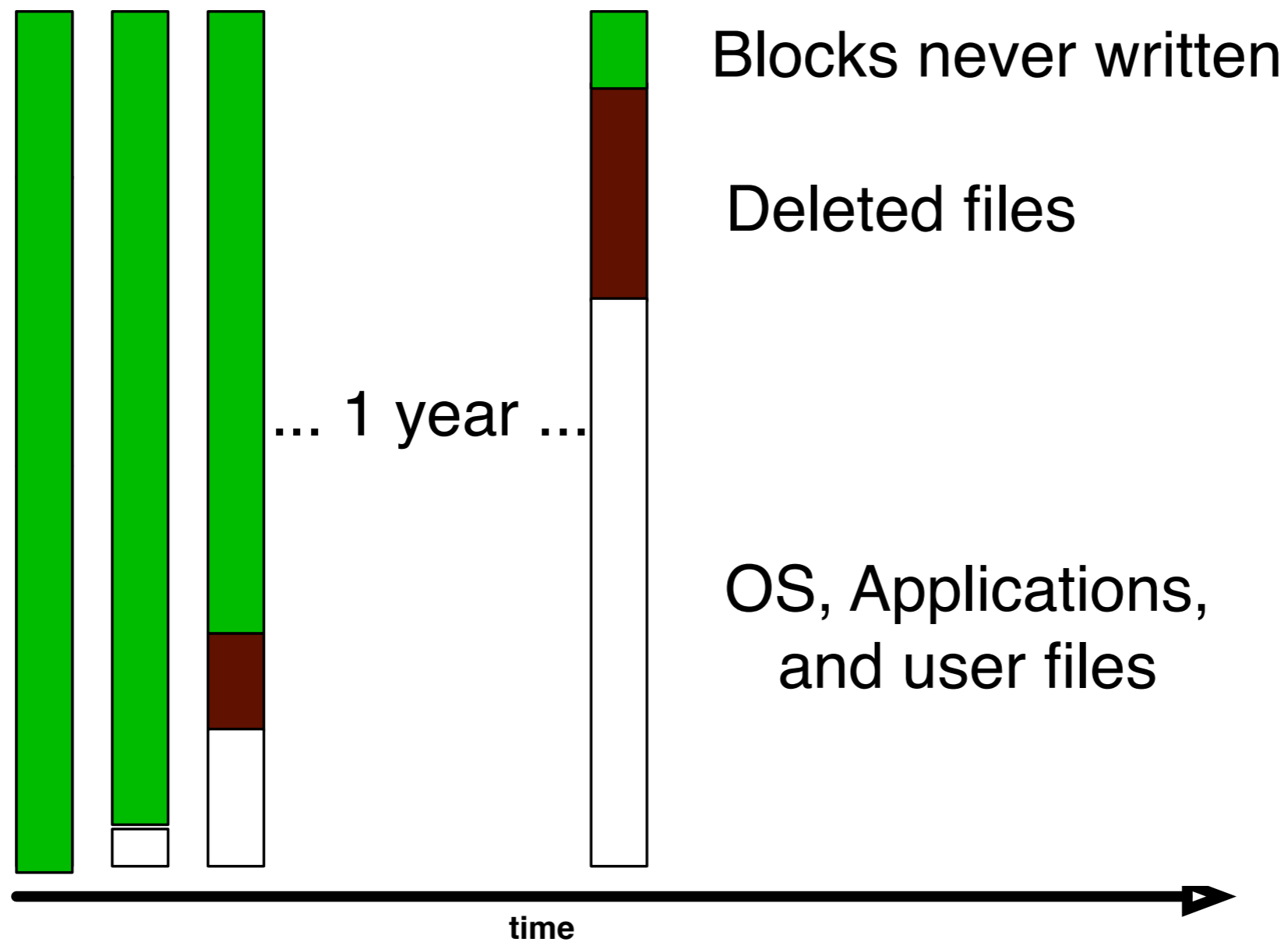
Formatted: the disk has an empty file system



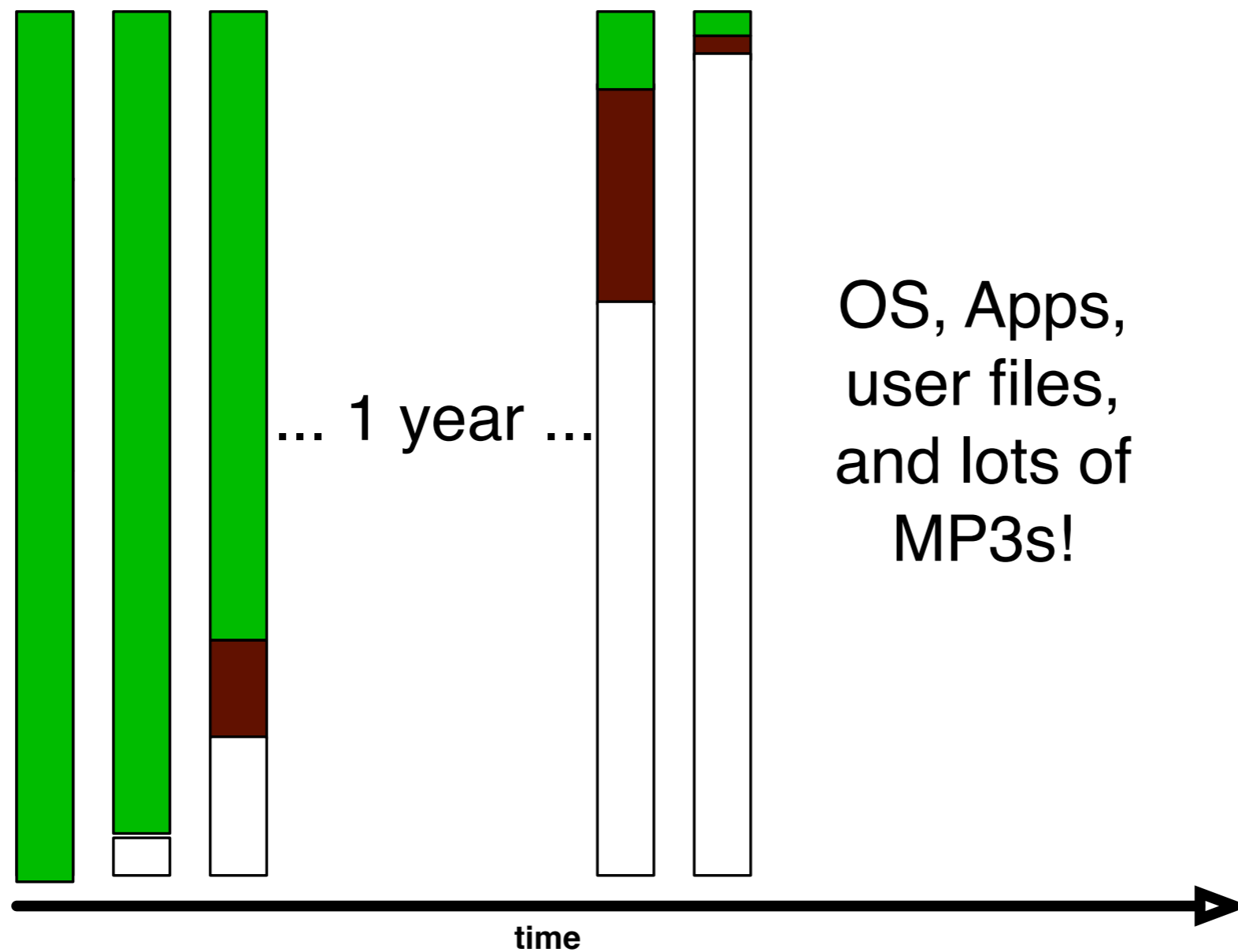
AFTER OS INSTALL: Temp. files have been deleted



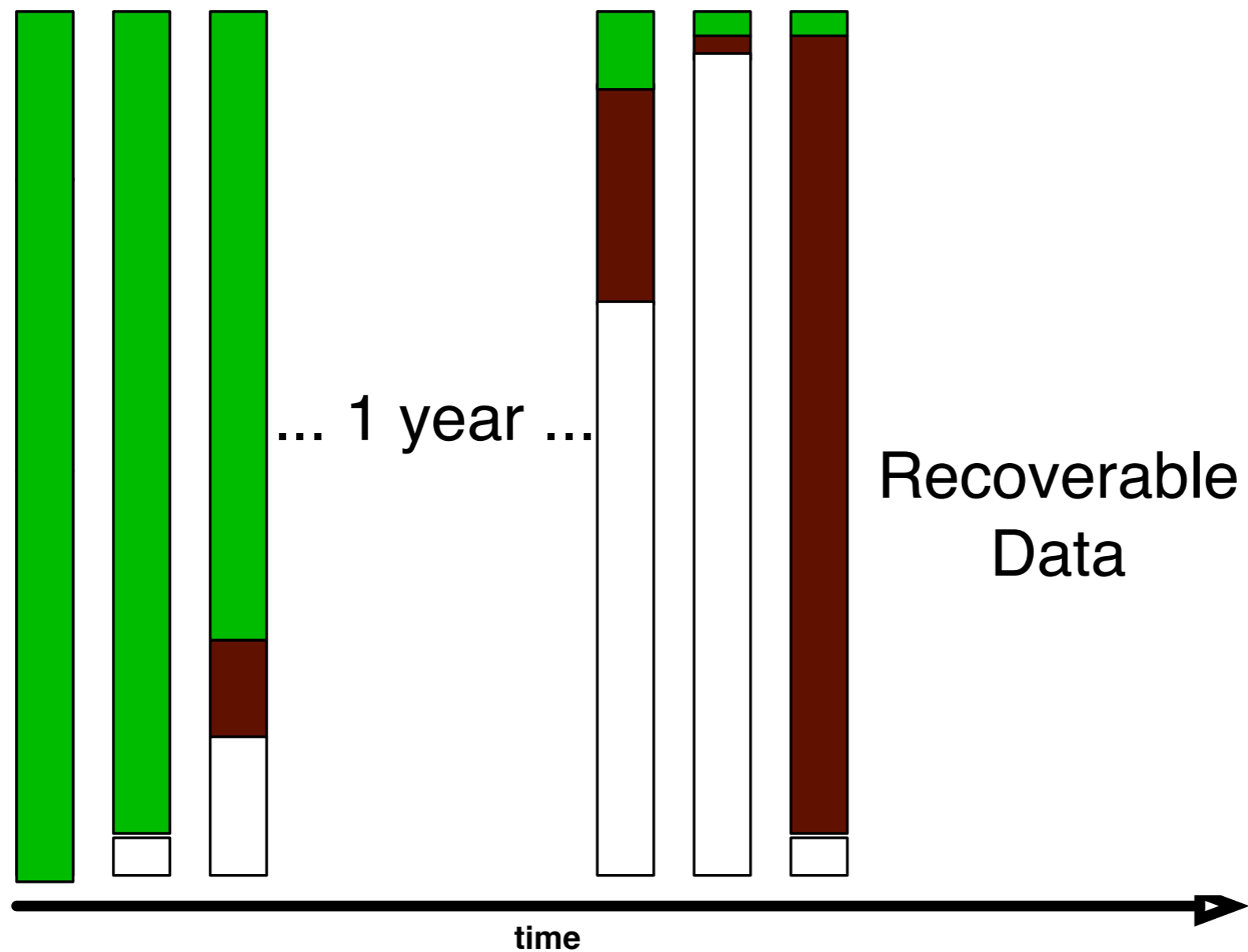
After a year of service



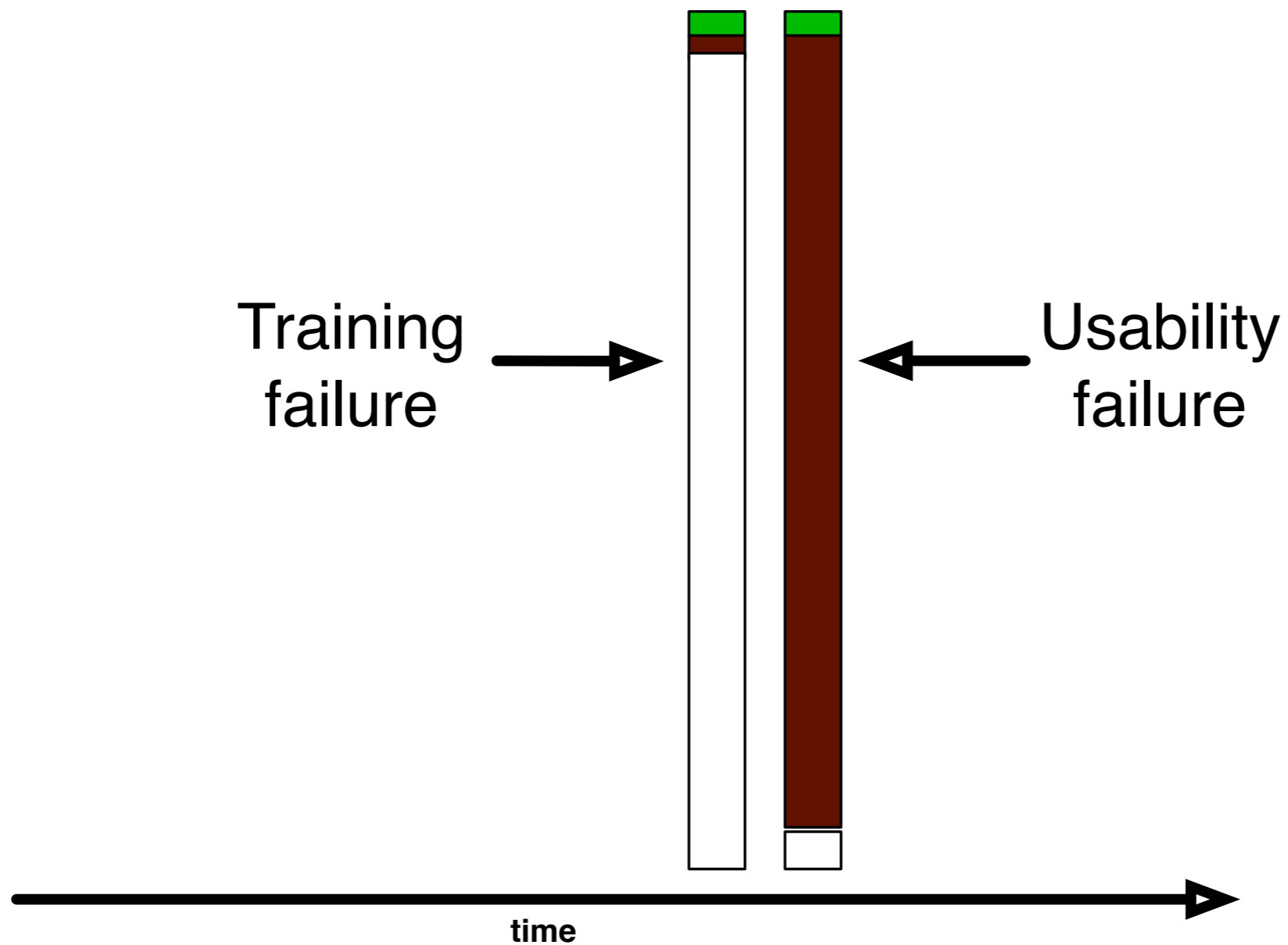
Disk nearly full!



Let's sell the hard drive!
Format c:\



We can use forensics to reconstruct motivations:



Hard drives are frequently sold on the secondary market.

Re-used within an organization

Given to charities

Sold on eBay



All Categories

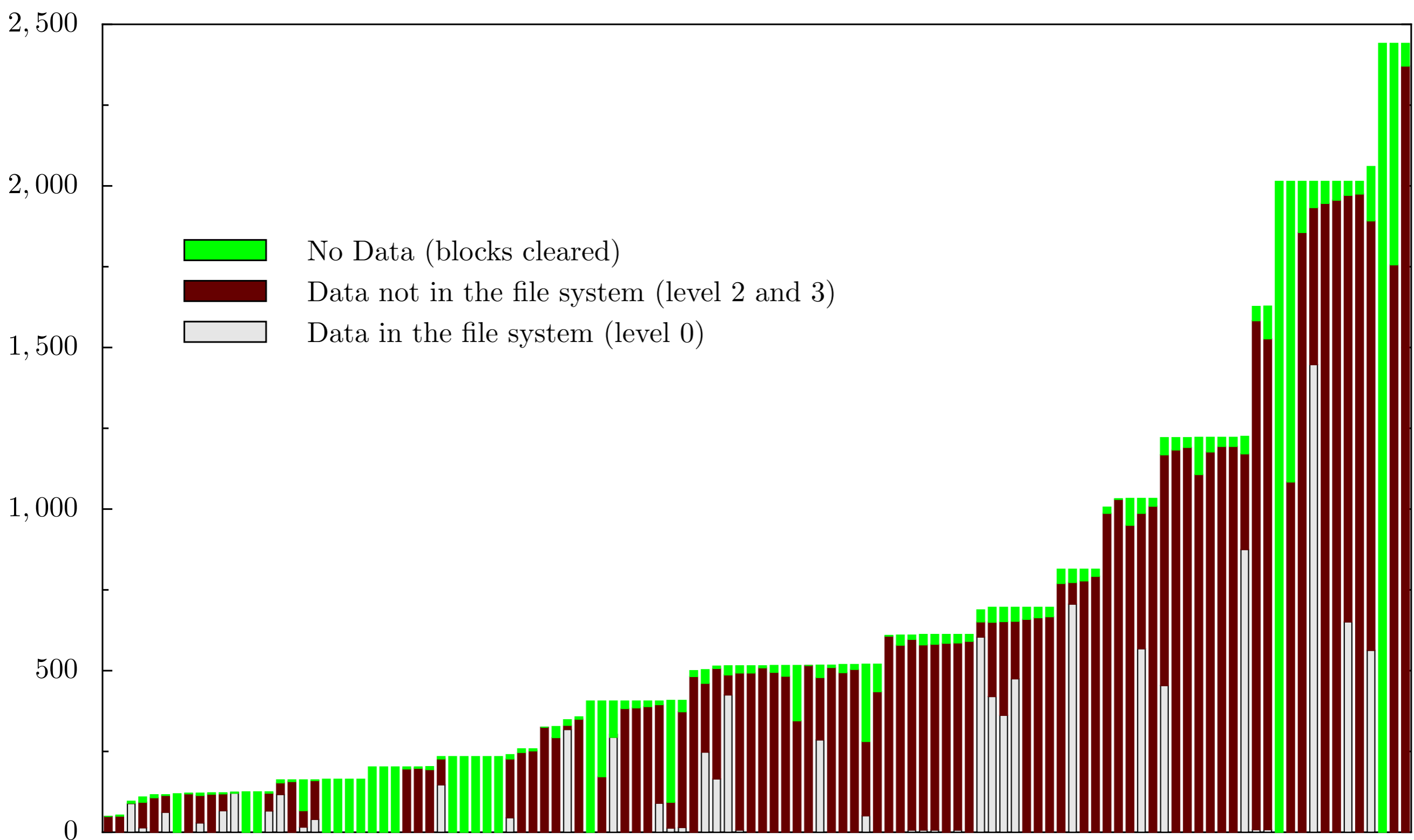
350 items found for **hard drives**

Sort by items: **ending first** | [newly listed](#) | [lowest priced](#) | [highest priced](#)

[Save this search](#)

Picture hide	Item Title	Price	Bids	Time Left
	Lot of hard and floppy drives	\$5.50	2	14n
	Lot of hard and floppy drives	\$5.50	2	22n
	Lot of hard and floppy drives	\$5.50	2	25n
	Lot of 2 hard drives IDE	\$8.00	12	29n
	3.2 gig Hard Drives	\$180.00	-	59n
	(5) 1.2 hard drives & (15) 10/100 network	\$25.00	1	1h 00n
	Lot of 3 Quantum 9.1 gig SCSI Hard Drives	\$26.00	6	1h 25n
	IDE HARD DRIVES (3)	\$6.50	6	1h 46n
	LOT OF 5 Hard Drives! 3.2 Gig Western Digital	\$120.00 \$124.95 Buy It Now	-	1h 50n
	QTY 3...IDE Hard Drives 2.5 Gig	\$20.50	5	2h 02n
	5 WESTERN DIGITAL 2.5 GIG HARD DRIVES	\$30.00	4	2h 03n
	QTY 3...IDE Hard Drives 1.0 Gig	\$9.99	1	2h 04n
	Western Digital 850 meg IDE Hard Drives dutch	\$6.00	1	2h 57n
	WINDOWS	\$6.00	-	3h 18n

236 drives purchased between 1998 and 2003:



Roughly 1/3 of the discarded hard drives have significant amounts of confidential data.

From sampling 150 hard drives collected between 1998 and 2002, we found:

- Thousands of credit cards
- Financial records
- Medical information
- Trade secrets
- Highly personal information



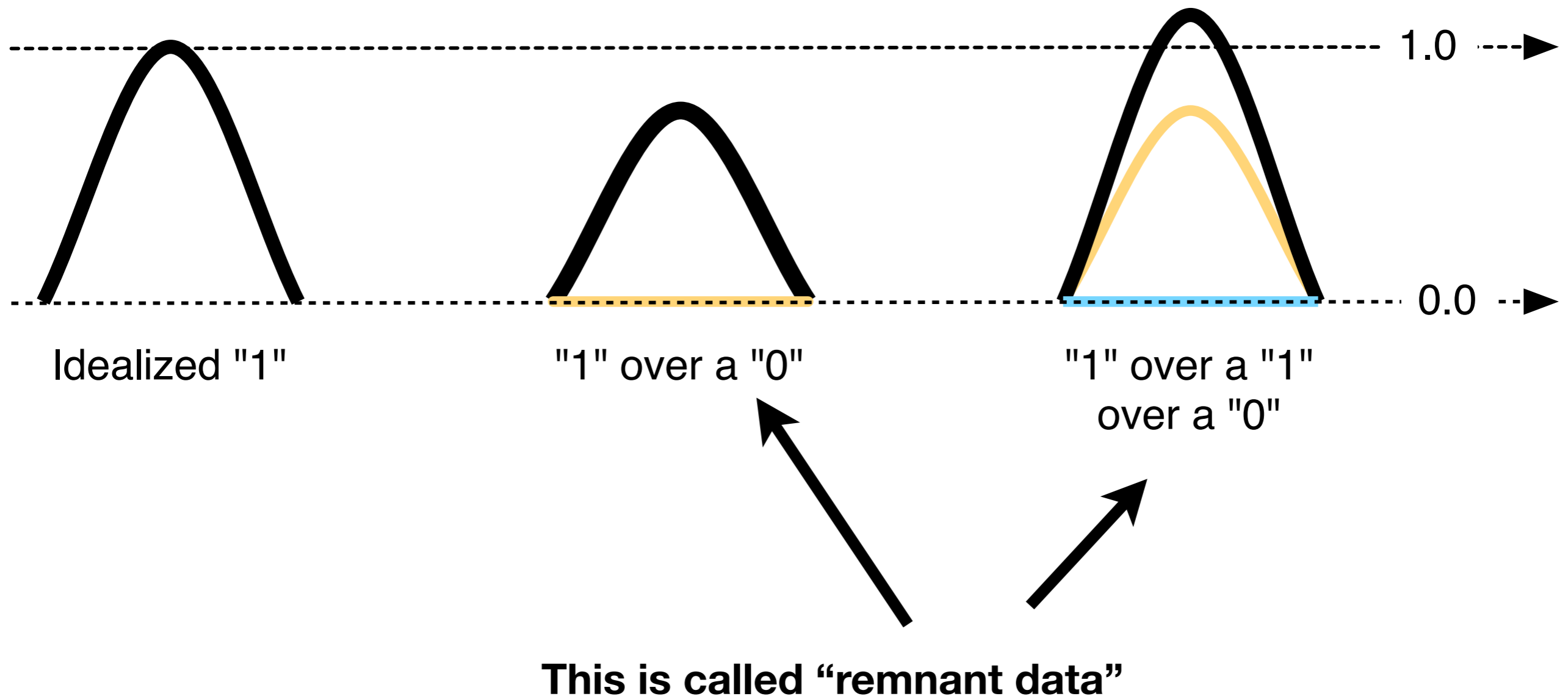
Can data be recovered after it has been overwritten?

?



Remnant data is data that *remains* after writing.

Writing “1” over a “0” is different than writing a “0” over a “0”



DoD 5220.22-M Specifies a “sanitization” procedure for unclassified data

- Write a character
- Write its complement
- Write random data

In 1996 Peter Gutmann published a paper with 35 sanitization patterns.

srm uses a 7-pass pattern (F6, 00, FF, random, 00, FF, random)

What is the sufficient amount of overwriting to make drives forget?



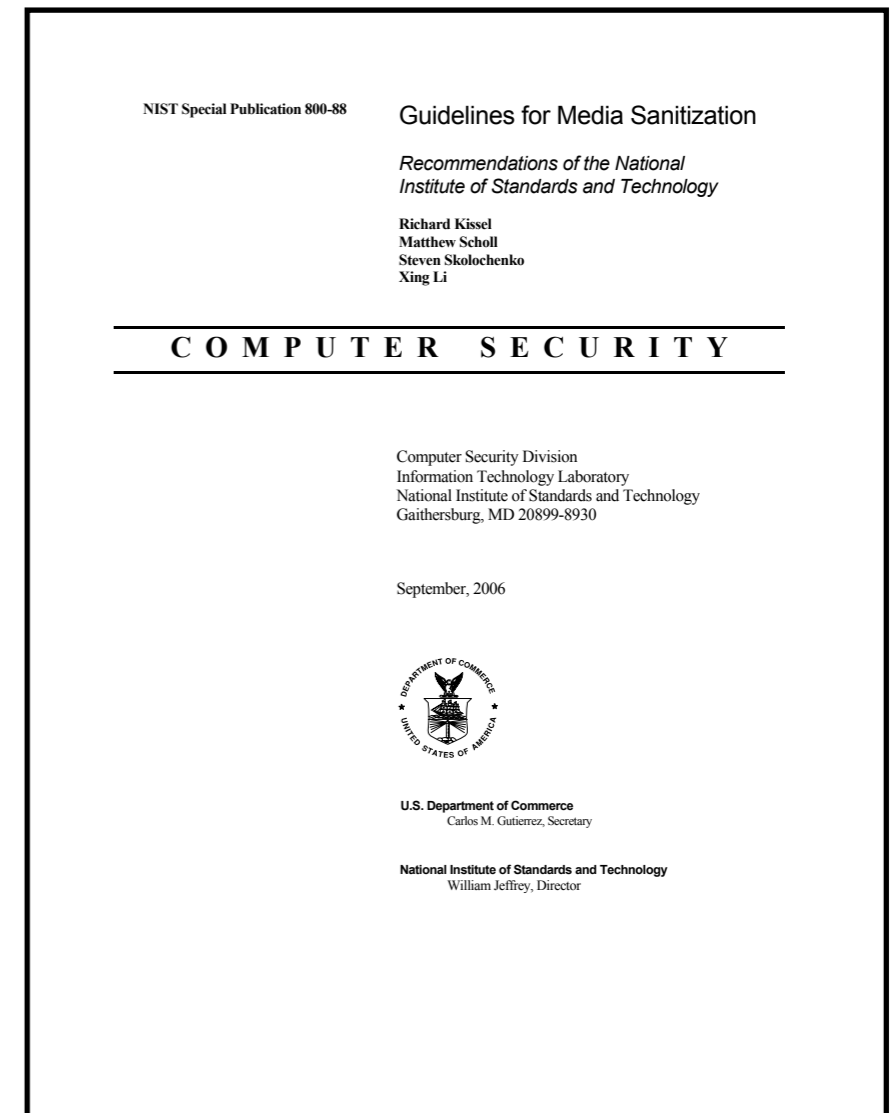
NIST 800-88 says overwritten data cannot be recovered in practice.

Modern disk drives are too complicated:

- Recording densities too high
- Use complex codes, not 0s & 1s
- No space between tracks
- Perpendicular recording will make remnant data even harder to recover.

Recovering overwritten data has *never* been demonstrated.

NIST 800-88, “Guidelines for Media Sanitization,” says a single pass with "secure overwrite" command is good enough for ATA disks manufactured after 2001 (over 15GB).

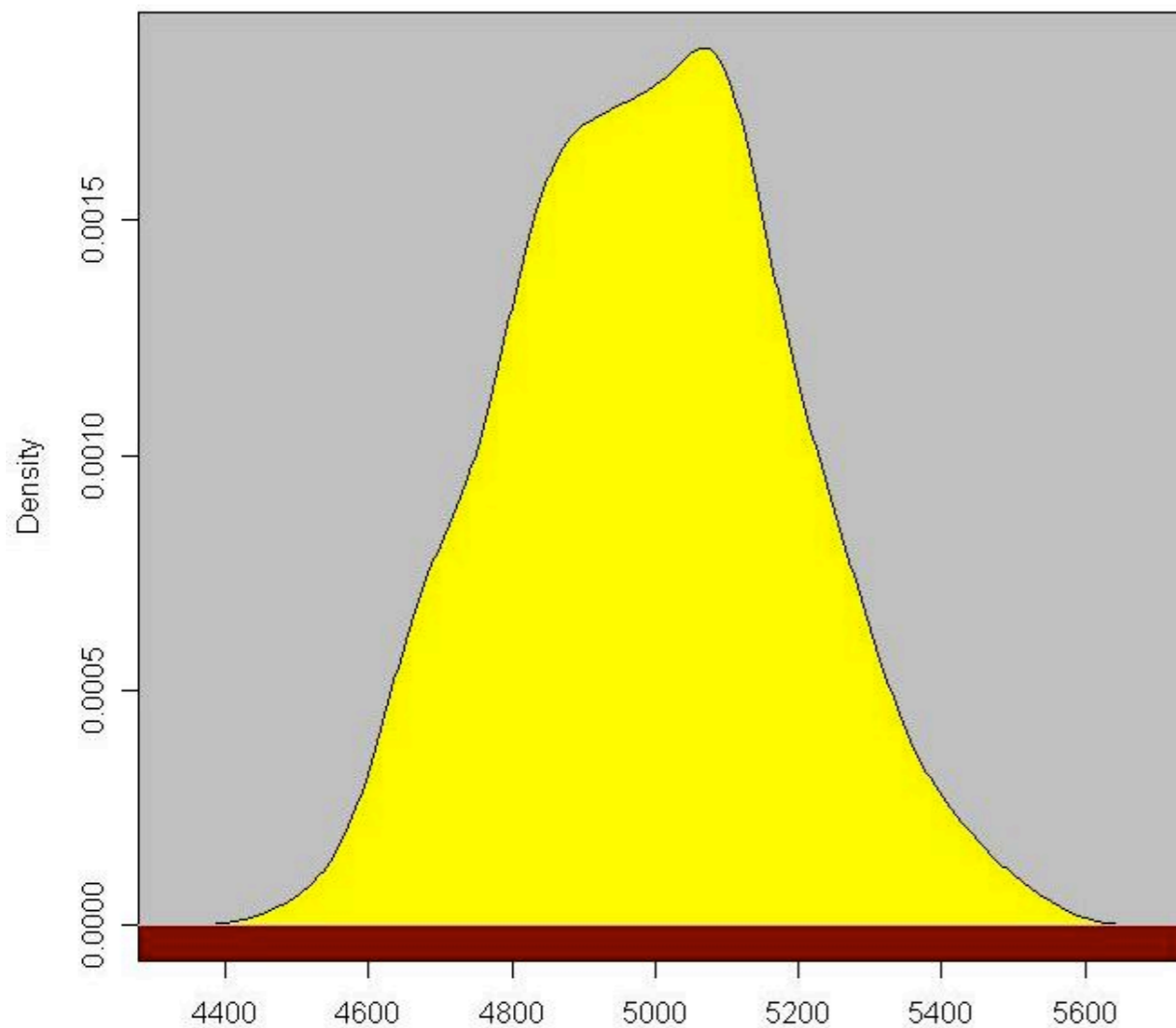


Experiments on Data Remnance - Hard Drives

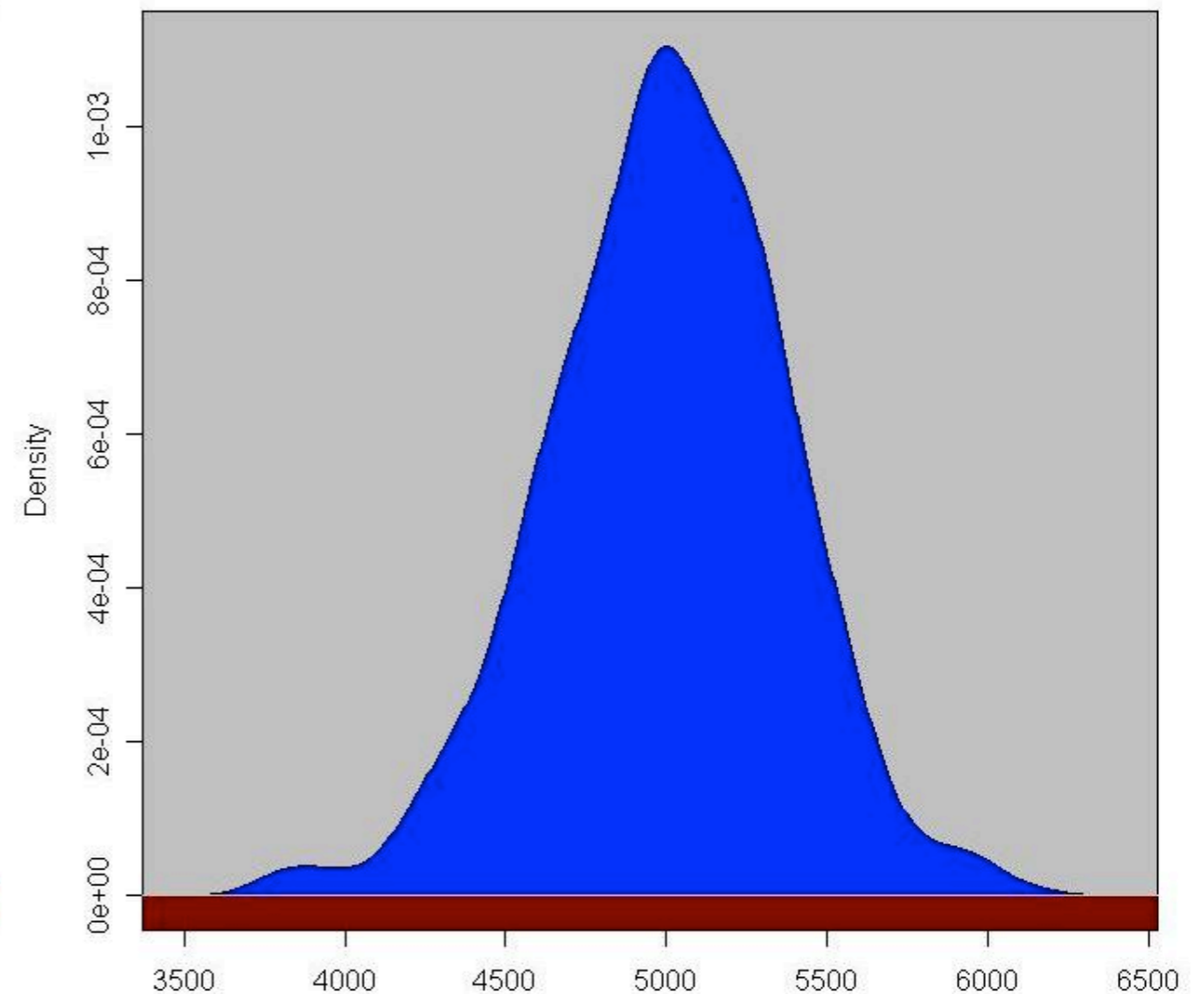
Craig Wright (AU) has conducted extensive remnance experiments.

- There are some differences, but actually recovering data seems to be impossible.

Distribution of "0" values from overwrite with a "1"

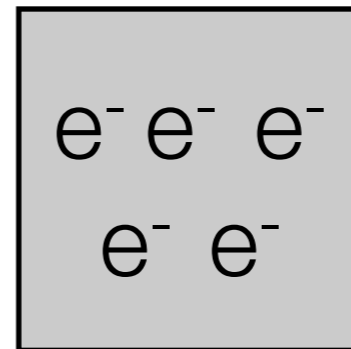


Distribution of "1" values from overwrite with a "1"



Experiments on Data Remnance - Flash

Flash works by storing electrons in “cells.”



Flash cells wear out.

- Each cell can only be cycled 1000-10,000 times
- Wear leveling algorithms spread out the wear evenly.
- This is done at the *physical layer*, invisible to the operating system.
- (Flash file systems do it at the operating system level.)

Therefore, there is a data remnance effect with Flash.

- No credible reports of using this for forensic analysis.

There are many ways to “clear” or “sanitize.”

Built-in Programs:

- cipher.exe (Windows)
- “Secure Empty Trash” (MacOS)

Third-Party Programs:

- BCWipe
- CCleaner
- DBAN (Darik’s Boot and Nuke)
- Eraser



http://www.forensicswiki.org/wiki/Category:Secure_deletion

Beware: Some courts now view "secure erase" as evidence of a crime...

US v. Krause (In re Krause), 2007 WL 1297937, 2007 Bankr. LEXIS 1937 (Bankr. D. Kan. June 4, 2007)

- Court rules that a lawyer who owed 3 million in back taxes is guilty of spoliation of evidence. Lawyer used "super-delete" feature.
<http://ralphlosey.wordpress.com/2007/07/07/ghostsurfer-wipe-out-leads-to-jail-order-sanction-in-bankruptcy-court/>

Kucala Enterprises Ltd. v. Auto Wax Co., 2003 WL 21230605 (N.D. ILL)

- Kucala obtained and ran "Evidence Eliminator" prior to producing discovery documents in a patent litigation case.
- Court noted that parties have a duty to preserve all relevant evidence.
- Kucala's suit dismissed and Auto Wax awarded attorney fees and costs.
http://www.uslfg.com/oldnews.cfm?FuseAction=details&Users_ID=39

“Residual” and “Remnant” data: Conclusions

Residual Data is data left behind — the “residue”

- Deleted files that have not been overwritten
- Files after a disk has been “formatted” with Windows XP (but not Vista)
- Data that has been free()’ed
- *Can be recovered with forensic tools.*

Remnant Data can be recovered with magnetic “remnance”

- Magnetic remnance is real.
- Experiments have shown that remnance data cannot be recovered on modern hard drives.
- Remnance may become an issue for flash.



Coffee

This is an introductory tutorial!

Theory, Science and Tools

8:30 - 10:00 Introduction

- Introduction to Digital Forensics & The Law

10:00 - 10:30 Coffee

10:30 - 12:00 Data Analysis

- Unicode, File Formats & File Identification

12:00 - 1:30 Lunch

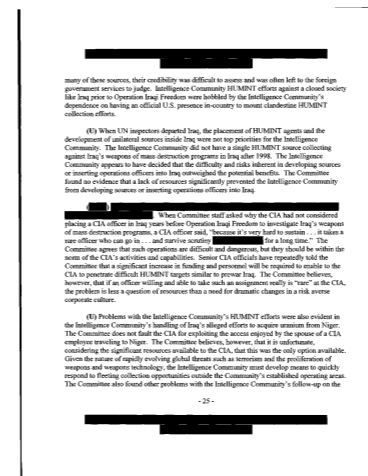
1:30 - 3:00 Disk Forensics

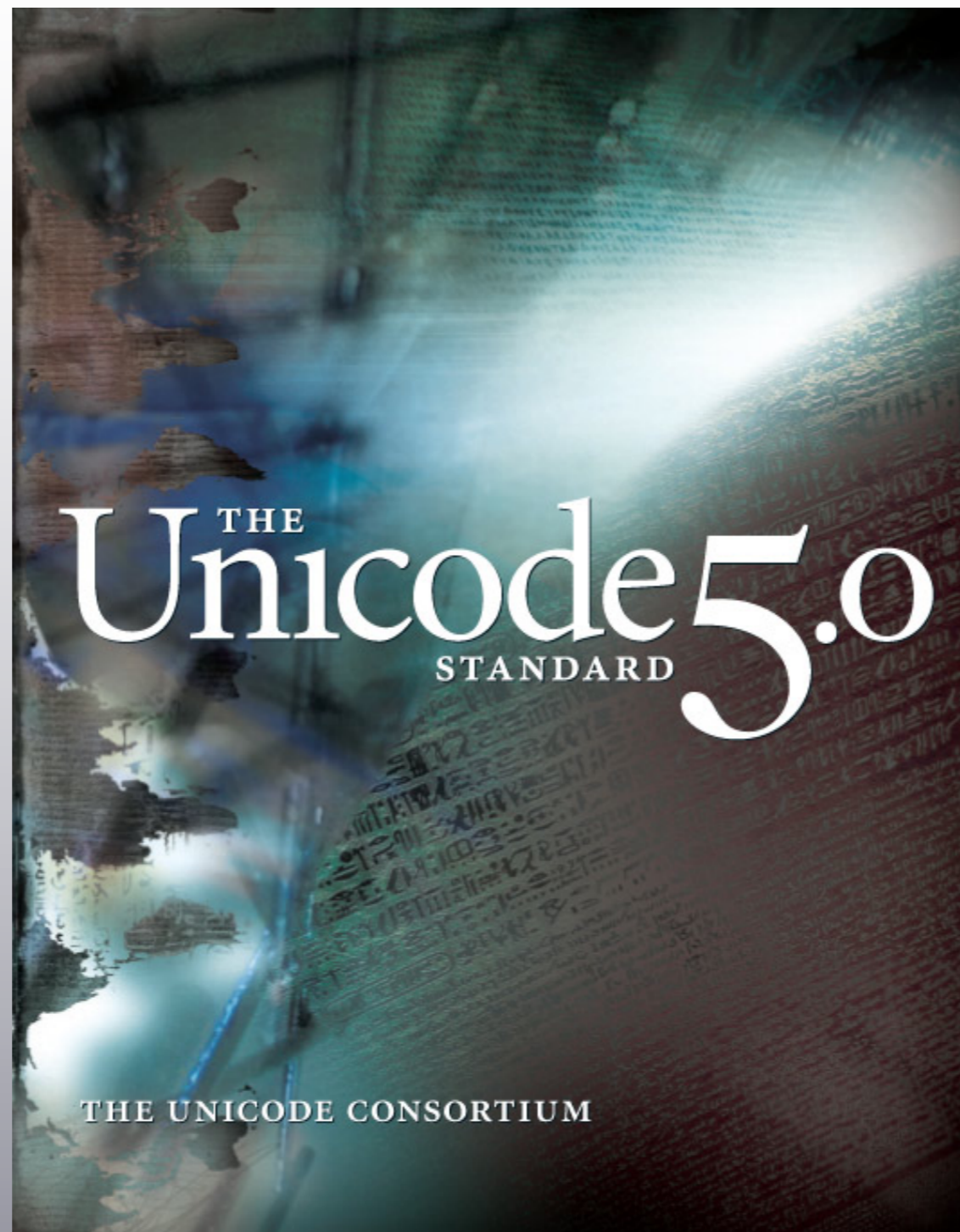
- Disk Imaging
- File Carving
- Sleuth Kit

3:00 - 3:30 Coffee

3:30 - 5:00 Big Finish

- Documents & Metadata
- Memory Forensics
- Anti-Forensics





ASCII, Code Pages,
and Unicode

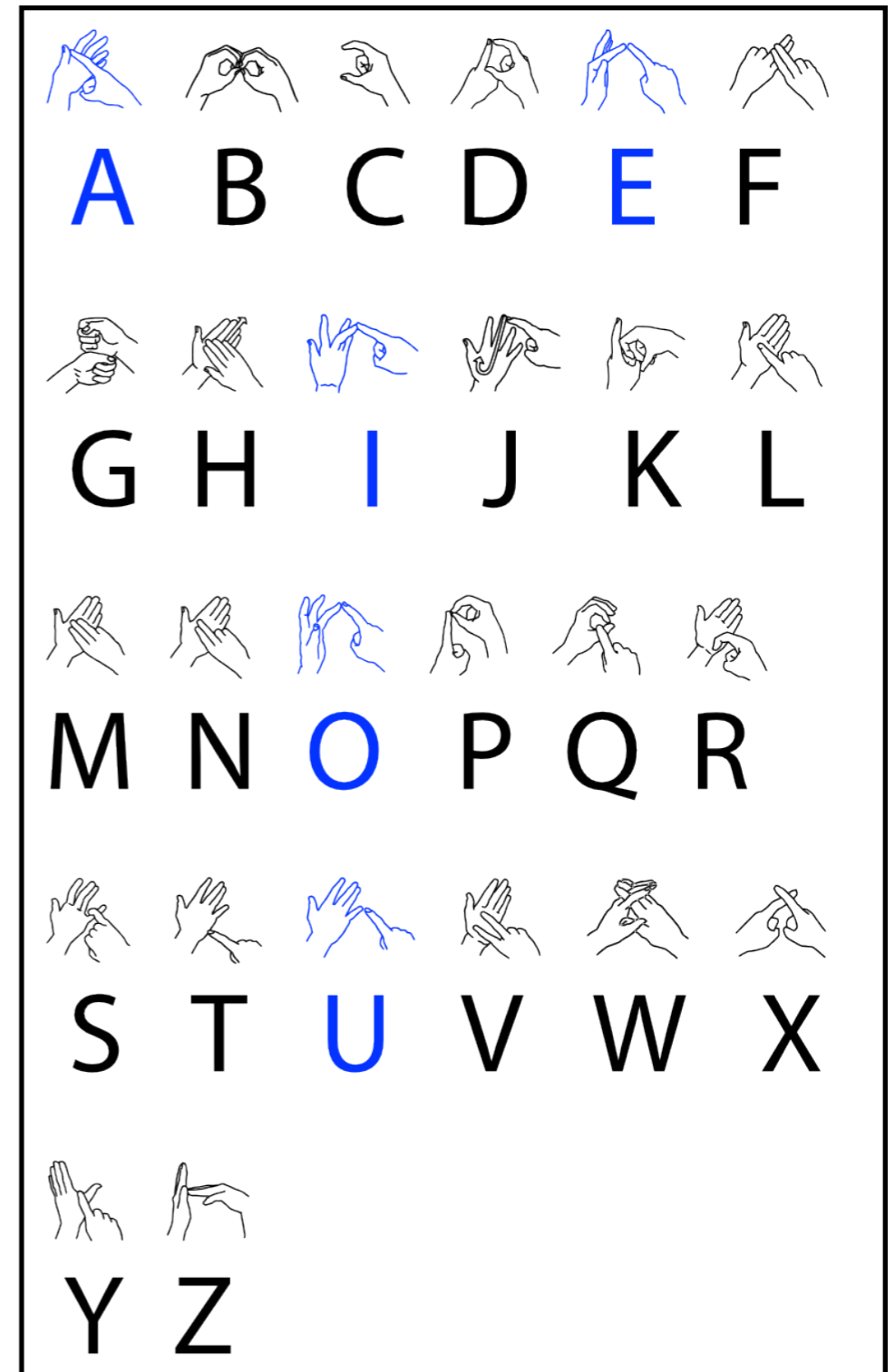
A "Code" is a system for converting one piece of information to another.

There are many codes:

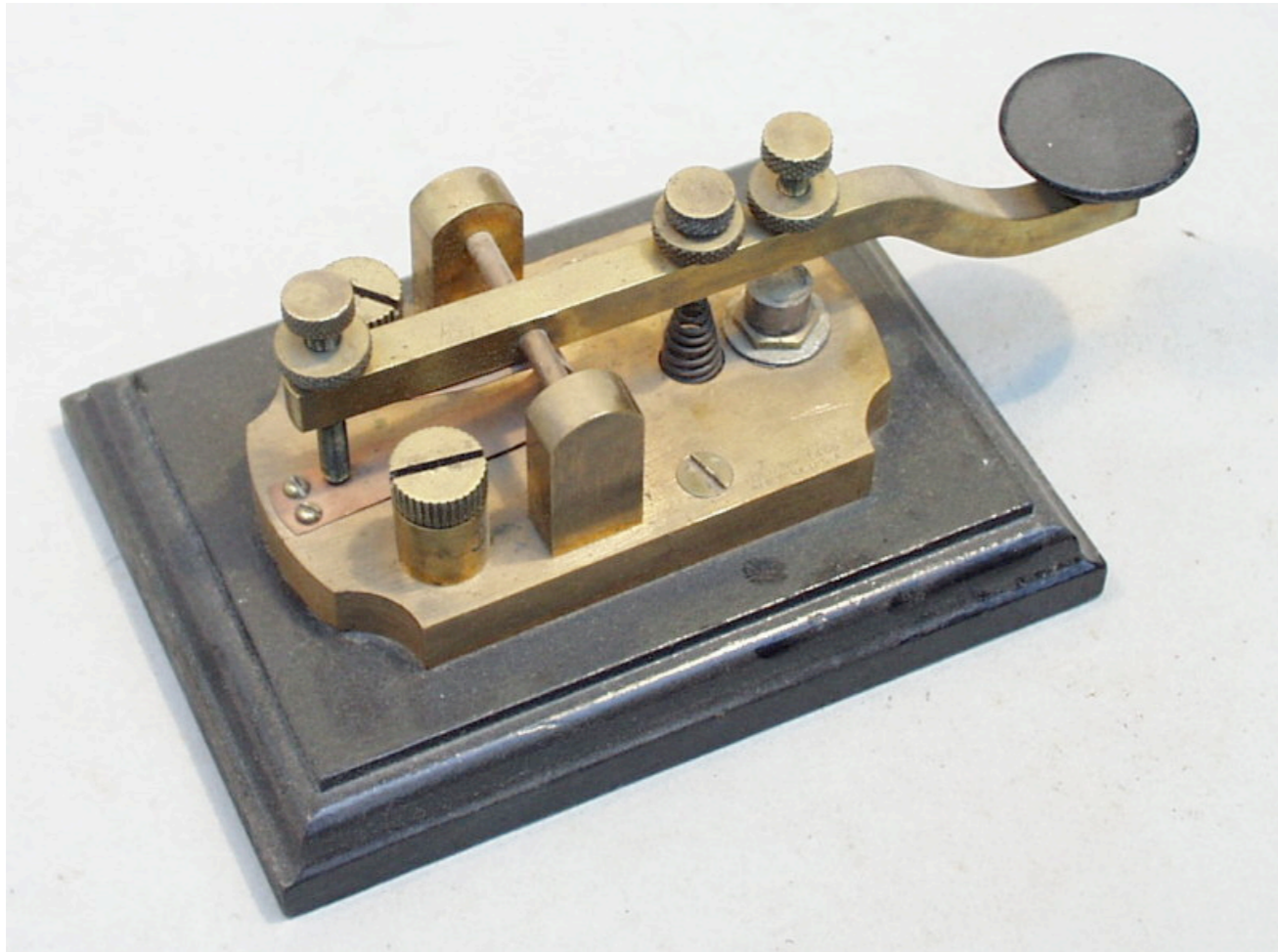
- "Modem Codes" — Values that a modem returns to identify itself.
- ASCII — American Standard Code for Information Interchange
- Unicode — Modern interchange code.
 - ✓ Note: These days codes are *rarely* used for security because they are easily broken.

A *code book* is a list of codes and their meanings.

In computing, a *code point* is a particular number and its graphical representation.



One of the earliest standard codes was Morse Code.



International Morse Code

- 1 dash = 3 dots.
- The space between parts of the same letter = 1 dot.
- The space between letters = 3 dots.
- The space between words = 7 dots.

A	• —	V	• • • —
B	— • • •	W	• — —
C	— • — •	X	— • • —
D	— • •	Y	— • — —
E	•	Z	— — • •
F	• • — •	.	• — • — • —
G	— — •	,	— — • • — —
H	• • • •	?	• • — — • •
I	• •	/	— • • — •
J	• — — —	@	• — — • — •
K	— • —	1	• — — — —
L	• — • •	2	• • — — —
M	— —	3	• • • — —
N	— •	4	• • • • —
O	— — —	5	• • • • •
P	• — — •	6	— • • • •
Q	— — • —	7	— — • • •
R	• — •	8	— — — • •
S	• • •	9	— — — — •
T	—	0	— — — — —
U	• • —		

Morse code has 3 symbols (1½ bits)

Replace the letters with the code to code a message.

Message: "Meet me at 5."

Code:

— — • • • —
— — • •
• — —
• • • • •
• — • — • —

International Morse Code

- 1 dash = 3 dots.
- The space between parts of the same letter = 1 dot.
- The space between letters = 3 dots.
- The space between words = 7 dots.

A	• —	V	• • • —
B	— • • •	W	• — —
C	— • — •	X	— • • —
D	— • •	Y	— • — —
E	•	Z	— — • •
F	• • — •	.	• — • — • —
G	— — •	,	— — • • — —
H	• • • •	?	• • — — • •
I	• •	/	— • • — •
J	• — — —	@	• — — • — •
K	— • —	1	• — — — —
L	• — • •	2	• • — — —
M	— —	3	• • • — —
N	— •	4	• • • • —
O	— — —	5	• • • • •
P	• — — •	6	— • • • •
Q	— — • —	7	— — • • •
R	• — •	8	— — — • •
S	• • •	9	— — — — •
T	—	0	— — — — —
U	• • —		

Early computer systems used a variety of codes.

Baudot was a 5-bit ("level") code designed for teletypes

- Two character sets: "letters" and "figures"
 - ✓ 0x1B shifted to Figures
 - ✓ 0x1F shifted to letters
- Shift codes made processing difficult!

ASCII: 7-bit code created in 1960 for teleprinters. (Bell System Standard.)

EBCDIC: 8-bit code for System/360. (IBM Standard.)

- Designed for compatibility with punch cards.



00	01	02	03	04	05	06	07
NUL	E 3	LF	A -	SP	S ' I 8	U 7	
08	09	0A	0B	0C	0D	0E	0F
CR	D ENQ	R 4	J BEL	N , F !	C :	K <	
10	11	12	13	14	15	16	17
T 5	Z +	L >	W 2	H £	Y 6	P 0	Q 1
18	19	1A	1B	1C	1D	1E	1F
0 9	B ?	G &	FIGS	M .	X /	U ;	LTRS
Letters			Figures			Control Chars.	

ASCII was the dominant code from 1960 through 2000. It doesn't work well for non-US languages.

Each letter can be described by a number.

"A" = $0010\ 0001_2 = 65_{10} = 0101_8 = 0x41_{16}$

"B" = $0010\ 0010_2 = 66_{10} = 0101_8 = 0x42_{16}$

The decimal set:

0	nul	1	soh	2	stx	3	etx	4	eot	5	enq	6	ack	7	bel
8	bs	9	ht	10	nl	11	vt	12	np	13	cr	14	so	15	si
16	dle	17	dc1	18	dc2	19	dc3	20	dc4	21	nak	22	syn	23	etb
24	can	25	em	26	sub	27	esc	28	fs	29	gs	30	rs	31	us
32	sp	33	!	34	"	35	#	36	\$	37	%	38	&	39	'
40	(41)	42	*	43	+	44	,	45	-	46	.	47	/
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;	60	<	61	=	62	>	63	?
64	@	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[92	\	93]	94	^	95	_
96	`	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	{	124		125	}	126	~	127	del

ASCII doesn't represent European or Asian languages well.

ASCII splits the code space into distinct regions.

The decimal set:

0	nul	1	soh	2	stx	3	etx	4	eot	5	enq	6	ack	7	bel
8	bs	9	ht	10	nl	11	vt	12	np	13	cr	14	so	15	si
16	dle	17	dc1	18	dc2	19	dc3	20	dc4	21	nak	22	syn	23	etb
24	can	25	em	26	sub	27	esc	28	fs	29	gs	30	rs	31	us
32	sp	33	!	34	"	35	#	36	\$	37	%	38	&	39	'
40	(41)	42	*	43	+	44	,	45	-	46	.	47	/
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;	60	<	61	=	62	>	63	?
64	@	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[92	\	93]	94	^	95	_
96	`	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	{	124		125	}	126	~	127	del

- 0-31 Control Characters
- 48-57 Numbers
- 65-90 Uppercase Letters
- 97-122 Lowercase Letters

But ASCII was used on 8-bit systems!

The IBM PC used codes 128-255 to represent special symbols and accented characters.

8-	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
	00C7	00FC	00E9	00E2	00E4	00E0	00E5	00E7	00EA	00EB	00E8	00EF	00EE	00EC	00C4	00C5
	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9-	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	ø	£	Ø	×	ƒ
	00C9	00E6	00C6	00F4	00F6	00F2	00FB	00F9	00FF	00D6	00DC	00F8	00A3	00D8	00D7	0192
	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
A-	á	í	ó	ú	ñ	Ñ	ä	ö	¿	®	¬	½	¼	¡	«	»
	00E1	00ED	00F3	00FA	00F1	00D1	00AA	00BA	00BF	00AE	00AC	00BD	00BC	00A1	00AB	00BB
	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
B-	⌘	⌘	⌘		¡	Á	Â	À	©	¶	¶	¶	¶	¢	¥	¬
	2591	2592	2593	2502	2524	00C1	00C2	00C0	00A9	2563	2551	2557	255D	00A2	00A5	2510
	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
C-	Ł	Ł	Ł	ł	—	+	ã	Ã	ℓ	ℓ	ℓ	ℓ	ℓ	ℓ	ℓ	ℓ
	2514	2534	252C	251C	2500	253C	00E3	00C3	255A	2554	2569	2566	2560	2550	256C	00A4
	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
D-	Š	Đ	Ê	Ë	È	ı	Í	Î	İ	Ј	Г	■	■	ı	İ	■
	00F0	00D0	00CA	00CB	00C8	0131	00CD	00CE	00CF	2518	250C	2588	2584	00A6	00CC	2580
	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
E-	Ó	ß	Ô	Õ	õ	Ö	μ	þ	þ	Ú	Û	Ü	ý	Ý	ˆ	ˆ
	00D3	00DF	00D4	00D2	00F5	00D5	00B5	00FE	00DE	00DA	00DB	00D9	00FD	00DD	00AF	00B4
	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
F-	SHY	±	=	¾	Œ	§	÷	,	°	ˆ	ˆ	ˆ	ˆ	ˆ	ˆ	NBSP
	00AD	00B1	2017	00BE	00B6	00A7	00F7	00B8	00B0	00A8	00B7	00B9	00B3	00B2	25A0	00A0
	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Codes 0-127:

Codes

Code Page 850: Latin 1

236	237	238	239	—3	—4	—5	—6	—7	—8	—9	—A	—B	—C	—D	—E	—F
3	2	■	NBSP	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο	Π
00B3	00B2	25A0	00A0	394	395	396	397	398	399	39A	39B	39C	39D	39E	39F	3A0
252	253	254	255													
9—	Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω	α	β	γ	δ	ε	ζ	η	θ
	3A1	3A3	3A4	3A5	3A6	3A7	3A8	3A9	3B1	3B2	3B3	3B4	3B5	3B6	3B7	3B8
A—	ι	κ	λ	μ	ν	ξ	ο	π	ρ	σ	ς	τ	υ	φ	χ	ψ
	3B9	3BA	3BB	3BC	3BD	3BE	3BF	3C0	3C1	3C3	3C2	3C4	3C5	3C6	3C7	3C8
B—	⋯	⋈	▀		†	≡	‖	π	¶	‖	‖	¶	‖	‖	‖	⌞
	2591	2592	2593	2502	2524	2561	2562	2556	2555	2563	2551	2557	255D	255C	255B	2510
C—	Ł	⊥	⌞	ł	—	+	ℓ	‖	ℓ	ℓ	⌞	⌞	‖	=	‖	±
	2514	2534	252C	251C	2500	253C	255E	255F	255A	2554	2569	2566	2560	2550	256C	2567
D—	⌞	⌞	π	ℓ	ℓ	ℓ	π	‖	≠	⌞	ℓ	■	■	■	■	■
	2568	2564	2565	2559	2558	2552	2553	256B	256A	2518	250C	2588	2584	258C	2590	2580
E—	ω	ά	έ	ή	ϊ	ί	ό	ύ	ϋ	ώ	Α	Ε	Η	Ι	Ο	Υ
	3C9	3AC	3AD	3AE	3CA	3AF	3CC	3CD	3CB	3CE	386	388	389	38A	38C	38E
F—	Ω	±	≥	≤	İ	ÿ	÷	≈	°	.	.	√	ⁿ	²	■	
	38F	B1	2265	2264	3AA	3AB	F7	2248	B0	2219	B7	221A	207F	B2	25A0	A0

Code Page 737: Greek

Code pages complicate processing because different code pages show the same text differently!

There are many code pages:

- 437 — Original IBM PC code page
- 737 — Greek
- 775 — Estonian, Lithuanian and Latvian
- 850 — "Multilingual (Latin-1)" (Western European languages)
- 852 — "Slavic (Latin-2)" (Central and Eastern European languages)
- ...

This text in code page 437: "naïve"

Becomes this text in code page 737: "naMve"

- Note: that "M" is character code 8B; it is not an "M" (code 4D)

Problems with code pages:

- No intrinsic coding of current code page.
- Lack of standardization
- Hard to get symbols from multiple code pages.
- Some vendors implemented "shift."
- No obvious way to handle Chinese, Japanese, Korean, or Vietnamese (CJKV)

Unicode was developed as a single coding standard for all of the world's languages

Project started in 1987 at Xerox and Apple.

- Originally called for 16-bit characters (limit of 65,535 symbols)
- Expanded to handle code points 0 through 10FFFF (1,114,112 total) to cover ancient languages.

Goals:

- Compatibility with existing systems.
- Clean "round trip" to legacy codings.
- Stability.
- No "shift" characters.
- Code graphemes, not glyphs (e.g., 'à' and 'a' code the same)
- "Han unification" — A single set of characters for identical kanji in Chinese, Japanese, Korean, and Cantonese

Today Unicode 5.0 is widely used.

Unicode has 1,114,112 *code points* ranging from 0 to 10FFFF.

Most Unicode characters are 16-bit characters.

- U+0041 is "A" "LATIN CAPITAL LETTER A." *Just like ASCII*
- U+0042 is "B" *Just like ASCII*
- U+0495 is "ક" *Gujarati letter KA*
- U+20AC is "€" *Euro*
- U+FE4A is "ⷐ" *Centerline Overline*

Unicode 4.0 has characters for *every living human language*.

- *Arabic* العربية *left-to-right*
- *Hebrew* עברית
- *Japanese* 日本語

Unicode 5.0 added support for dead languages.

- Excellent demo online at <http://www.fileformat.info/info/unicode/>

Principles of the Unicode Standard

Universal Repertoire	Logical order
Efficiency	Unification
Characters, not glyphs	Dynamic Composition
Semantics	Stability
Plain Text	Convertibility

<http://www.unicode.org/standard/principles.html>

Unicode is divided into 17 planes,
each with 65,536 code points.

Only a few code points are actually used:

Plane	Range	Name
0	U+0000 to U+FFFF	Basic multilingual Plane (BMP)
1	U+10000 to U+1FFFF	Supplementary Multilingual Plane (SMP)
2	U+20000 to U+2FFFF	Supplementary Ideographic Plane (SIP)
3 - 13	<i>Unassigned</i>	
14	U+E0000 to U+EFFFF	Supplementary Special-purpose Plane (SSP)
15	U+F0000 to U+FFFFFF	Private Use Area (PUA)
16	U+100000 to U+10FFFF	Private Use Area (PUA)

Unicode code points can be coded as 1, 2, 3 or 4 characters

Most Unicode text is encoded as UTF-8

- Variable-length code; ASCII characters code as ASCII
- Arabic, Armenian, Cyrillic, Coptic, Greek, Syriac & Tāna: 2 characters
- Chinese, Japanese, Korean & Vietnamese: 3 characters
- Other: 4 (or more)

Unicode	Byte1	Byte2	Byte3	Byte4	example
U+0000–U+007F	0xxxxx xx				'\$' U+0024 → 00 <u>100100</u> → 0x24
U+0080–U+07FF	110yyy xx	10xxxx xx			'¢' U+00A2 → 110 <u>00010</u> , 10 <u>100010</u> → 0xC2, 0xA2
U+0800–U+FFFF	1110yy yy	10yyyy xx	10xxxx xx		'€' U+20AC → 1110 <u>0010</u> , 10 <u>000010</u> , 10 <u>101100</u> → 0xE2, 0x82, 0xAC
U+10000–U+10FFFF	11110z zz	10zzyy yy	10yyyy xx	10xxxx xx	U+10ABCD → 11110 <u>100</u> , 10 <u>001010</u> , 10101 <u>111</u> , 10 <u>001101</u> → 0xF4, 0x8A, 0xAF, 0x8D

<http://en.wikipedia.org/wiki/UTF-8>

UTF-16 codes most characters as 2 bytes.

UTF-16 is the original Unicode representation

Widely used by:

- Microsoft filenames
- Text in *some* Microsoft documents.
- Web pages authored in Chinese and Japanese.

Code plans 1 through 16 are encoded with U+D800 to U+DBFF

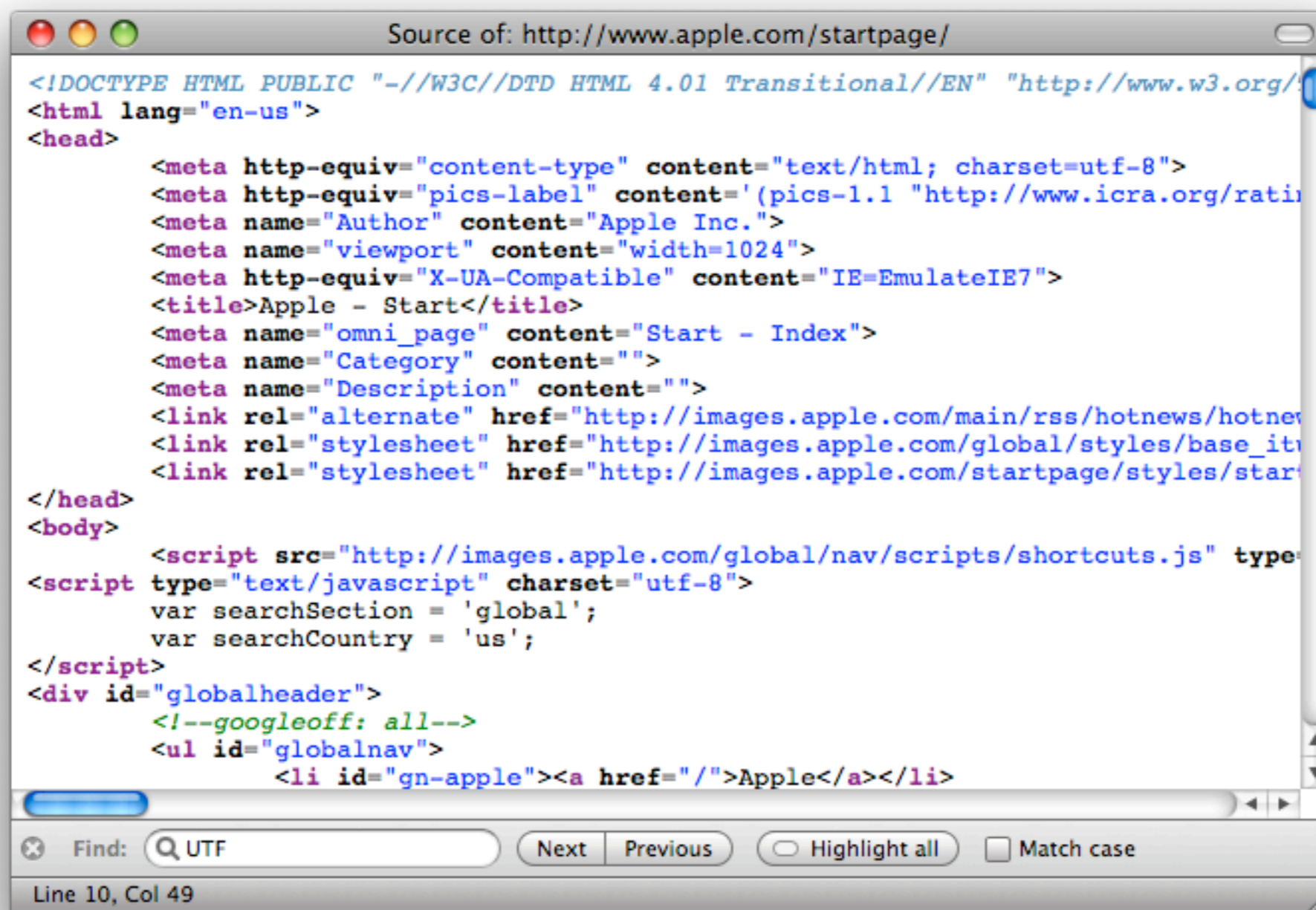
- Character U+10000 becomes 0xD800 0xDC00

Beware:

- UTF-16 can be coded two ways.
- Byte Order Mark (Zero-Width No-Break Space) U+FEFF at the beginning of the file specifies byte order:
 - ✓ *big-endian* — FE FF
 - ✓ *little-endian* — FF FE
- If text is accompanied with encoding of UTF-16BE or UTF-16LE, BOM is ignored.

Most "modern" web pages use UTF-8

www.apple.com:



```
Source of: http://www.apple.com/startpage/

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/
<html lang="en-us">
<head>
  <meta http-equiv="content-type" content="text/html; charset=utf-8">
  <meta http-equiv="pics-label" content='(pics-1.1 "http://www.icra.org/rati
  <meta name="Author" content="Apple Inc.">
  <meta name="viewport" content="width=1024">
  <meta http-equiv="X-UA-Compatible" content="IE=EmulateIE7">
  <title>Apple - Start</title>
  <meta name="omni_page" content="Start - Index">
  <meta name="Category" content="">
  <meta name="Description" content="">
  <link rel="alternate" href="http://images.apple.com/main/rss/hotnews/hotne
  <link rel="stylesheet" href="http://images.apple.com/global/styles/base_it
  <link rel="stylesheet" href="http://images.apple.com/startpage/styles/star
</head>
<body>
  <script src="http://images.apple.com/global/nav/scripts/shortcuts.js" type=
<script type="text/javascript" charset="utf-8">
  var searchSection = 'global';
  var searchCountry = 'us';
</script>
<div id="globalheader">
  <!--googleoff: all-->
  <ul id="globalnav">
    <li id="gn-apple"><a href="/">Apple</a></li>
```

Find: UTF Next Previous Highlight all Match case

Line 10, Col 49

There are many technical problems with Unicode.

Legacy problems:

- Implementations are incomplete
- Not all programmers have implemented all the rules.
- Multiple codings (UTF-8, UTF-16) mean that code that works sometimes with some codings doesn't work other times with other codings.

Ongoing problems

- Behavior of strings becomes complex and may depend on the locale.
- Complex rules for:
 - ✓ case conversion (toUpper(), toLower(), toTitle())
 - ✓ String comparison (isUpper(), isLower(), isTitle())

Complex rules for:

- bidi
- coalition
- line and paragraph breaks (U+2028 LINE SEPARATOR and U+2029 PARAGRAPH SEPARATOR)
- search/string matching

Consider Arabic:

There are multiple unicode glyphs for the same letter.

Different versions are used in different applications.

- For editing, the *general* form is used.
- For printing, the *isolated*, *final*, *medial* or *initial* forms might be used.
- For searching, *any form* needs to match

Each form has a different character code.

General: م

Isolated: م

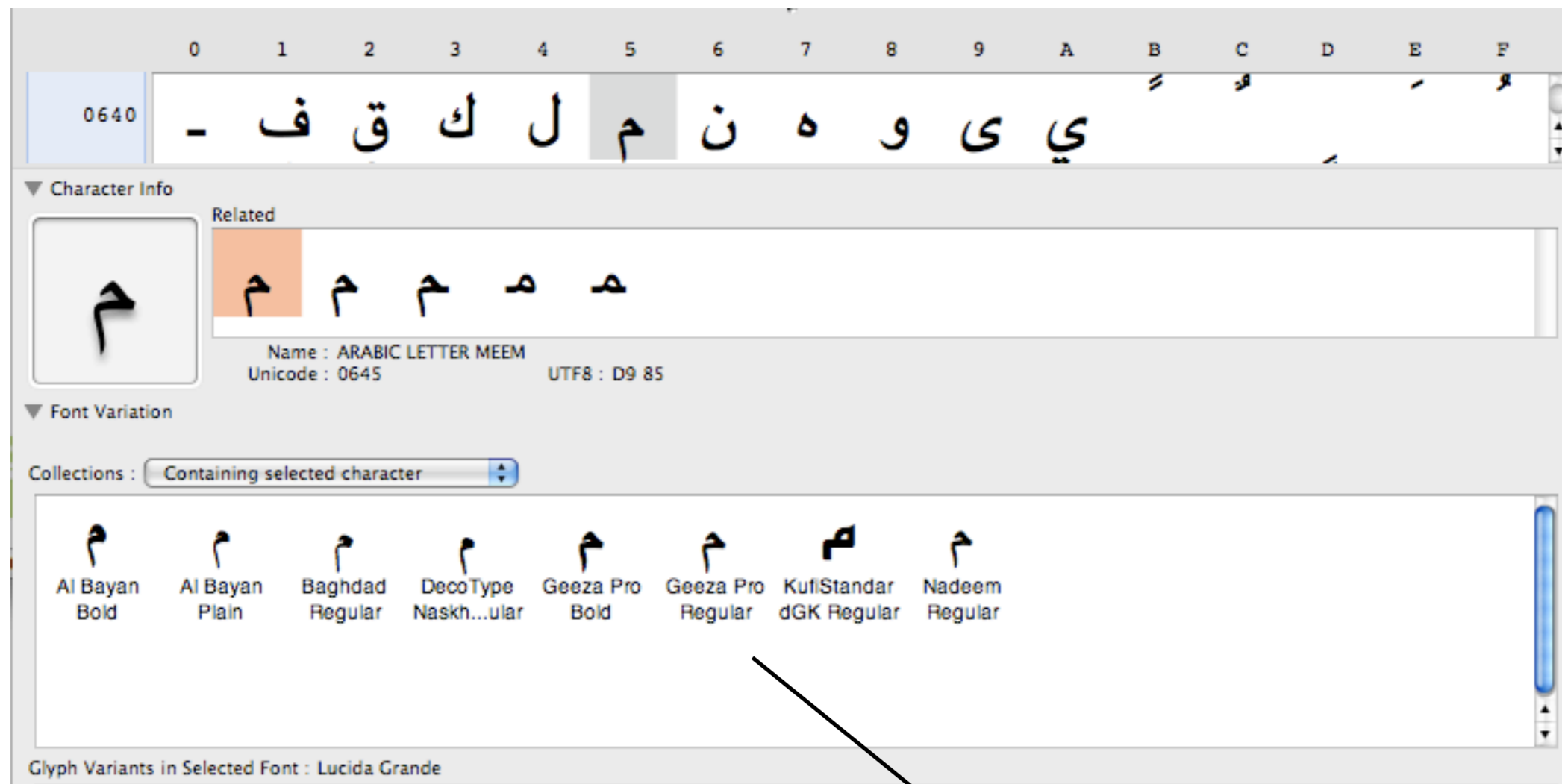
Final: م

Medial: م

Initial: م

Arabic MEEM

U+0645: ARABIC LETTER MEEM



Different fonts render the MEEM differently!

U+FEE1: MEEM ISOLATED FORM

The screenshot displays a font character map interface. At the top, a horizontal row of characters is shown, indexed from 0 to F. The character at index 1, U+FEE1, is the isolated form of the Arabic letter Meem (م), which is highlighted with a grey background. Below this row, the 'Character Info' section shows a larger view of the selected character (م) and a 'Related' section with five other variations of the letter. The 'Font Variation' section shows a 'Collections' dropdown set to 'Containing selected character', followed by a row of 12 font samples. Each sample shows the character 'م' in a different font style, including Arial Bold, Arial Regular, Arial Unico...gular, Courier New Bold, Courier New Regular, Geeza Pro Bold, Geeza Pro Regular, Microsoft Sans ...gular, Tahoma Bold, Tahoma Regular, Times New Roman Bold, and Times New Roman...ular. The character 'م' is also shown in a square box, likely representing a missing glyph or a specific font feature.

0 1 2 3 4 5 6 7 8 9 A B C D E F

FEE0 1 م م م م ن ن ن ن ه ه ه ه و و ي

▼ Character Info

Related

م م م م م

Name : ARABIC LETTER MEEM ISOLATED FORM
Unicode : FEE1
UTF8 : EF B8 A1

▼ Font Variation

Collections : Containing selected character

م م م م م م م م م م م م

Arial Bold Arial Regular Arial Unico...gular Courier New Bold Courier New Regular Geeza Pro Bold Geeza Pro Regular Microsoft Sans ...gular Tahoma Bold Tahoma Regular Times New Roman Bold Times New Roman...ular

U+FEE2: MEEM FINAL FORM

The screenshot displays a character map interface for the Arabic letter Meem Final Form (U+FEE2). At the top, a row of characters is shown, with the Meem Final Form highlighted in the second column. Below this, the 'Character Info' section shows the character's name, 'ARABIC LETTER MEEM FINAL FORM', and its Unicode (FEE2) and UTF8 (EF BB A2) values. The 'Font Variation' section shows a grid of the character rendered in various fonts, including Al Bayan Bold, Al Bayan Plain, Arial Bold, Arial Regular, Arial Unicode, Baghdad Regular, Courier New Bold, Courier New Regular, DecoType Naskh...ular, Geeza Pro Bold, Geeza Pro Regular, KufiStandar dGK Regular, Microsoft Sans...ular, Nadeem Regular, Tahoma Bold, Tahoma Regular, Times New Roman Bold, and Times New Roman...ular.

0 1 2 3 4 5 6 7 8 9 A B C D E F

FEE0 1 م م م م ن ن ن ن ه ه ه ه و و ي

▼ Character Info

Related

م م م م م

Name : ARABIC LETTER MEEM FINAL FORM
Unicode : FEE2 UTF8 : EF BB A2

▼ Font Variation

Collections : Containing selected character

Al Bayan Bold Al Bayan Plain Arial Bold Arial Regular Arial Unicode Baghdad Regular Courier New Bold Courier New Regular DecoType Naskh...ular Geeza Pro Bold Geeza Pro Regular KufiStandar dGK Regular

Microsoft Sans...ular Nadeem Regular Tahoma Bold Tahoma Regular Times New Roman Bold Times New Roman...ular

U+FEE3: MEEM INITIAL FORM

The screenshot displays a font viewer interface for the character U+FEE3, the Arabic letter Meem initial form. The top section shows a grid of characters from U+FEE0 to U+FEEF, with U+FEE3 highlighted. Below this, the 'Character Info' section shows the character 'م' and its related forms. The 'Font Variation' section shows the character 'م' in various fonts and styles.

Character Info

Related

Name : ARABIC LETTER MEEM INITIAL FORM
Unicode : FEE3
UTF8 : EF BB A3

Font Variation

Collections : Containing selected character

Font	Style	Glyph
Al Bayan	Bold	م
Al Bayan	Plain	م
Arial	Bold	م
Arial	Regular	م
Arial	Unico...gular	م
Baghdad	Regular	م
Courier	New Bold	م
Courier	New Regular	م
DecoType	Naskh...ular	م
Geeza Pro	Bold	م
Geeza Pro	Regular	م
KufiStandar	dGK Regular	م
Microsoft	Sans...ular	م
Nadeem	Regular	م
Tahoma	Bold	م
Tahoma	Regular	م
Times New	Roman Bold	م
Times New	Roman...ular	م

Glyph Variants in Selected Font : Lucida Grande

U+FEE4: MEEM MEDIAL FORM

0 1 2 3 4 5 6 7 8 9 A B C D E F

FEE0 ل م م م م ن ن ن ن ه ه ه و و ي

▼ Character Info

Related

م م م م م

Name : ARABIC LETTER MEEM MEDIAL FORM
Unicode : FEE4
UTF8 : EF BB A4

▼ Font Variation

Collections : Containing selected character

Al Bayan Bold	Al Bayan Plain	Arial Bold	Arial Regular	Arial Unicode	Baghdad Regular	Courier New Bold	Courier New Regular	DecoType Naskh...ular	Geeza Pro Bold	Geeza Pro Regular	KufiStandar dGK Regular
Microsoft Sans...ular	Nadeem Regular	Tahoma Bold	Tahoma Regular	Times New Roman Bold	Times New Roman...ular						

Glyph Variants in Selected Font : Lucida Grande

Unicode characters with diacritical marks can be constructed with 1 code or two.

The ñ character can be coded two ways:

U+00F1: ñ (LATIN SMALL LETTER N WITH TIDLE)

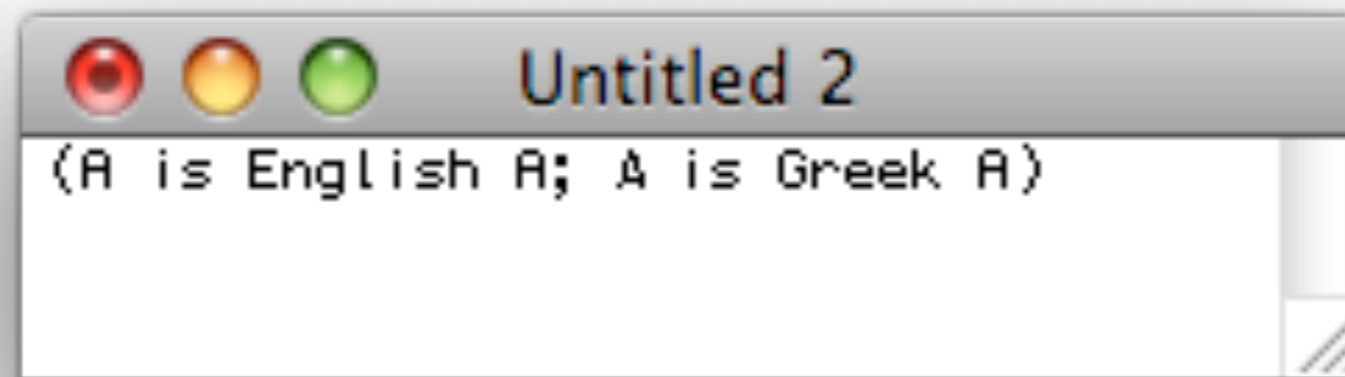
or:

U+0303: ~ (COMBINING TILDE)

U+006E: n (LATIN SMALL LETTER N)

Unicode has Usability problems.

A and A are different characters (A is English A; A is Greek A)



This leads to both database problems and phishing attacks.

In Summary

ASCII and Unicode:

- Display is easier than search
- Information may not display correctly, and you may not know it.

For further information:

- <http://www.unicode.org/standard/principles.html>
- <http://www.unicode.org/versions/Unicode5.1.0/>
- <http://www.unicode.org/notes/tn23/>
- <http://www.unicode.org/faq/>
- <http://macchiato.com/slides/UnicodeMyths.pdf>
- <http://unicode.org/standard/tutorial-info.html>

00001FE023957C19D5E70FA34085B623 | 00000000935899154
0000222286FAC25C704D56A6B3831128 | 00000000885465166
0000224DF8086F79200CADED083A7F8F | 00000000988385431
00002C6A13B13FB588B87C6204E73B0C | 00000000134913247
0000315467D336EB5EF32C584AEA63EC | 00000000160461562

Files

Files vs. Container Files
File Type Classification
File Recognition

File formats define how things are stored in files.

A file is a sequence of bytes:



Files have:

- length
- contents:
 - ✓ header (first N bytes)
 - ✓ footer (last N bytes)

Files do not have:

- names
- file type
- timestamps

These are provided by filesystems and applications

There are many different kinds of file formats.

Common file formats:

html	Web Pages	Hypertext Markup Language
JPEG	Images	Joint Photographic Experts Group
TXT	Text	
DOC	Microsoft Word	“Document” ?
GIF	Images	Graphic Interchange Format
XML	Data	eXtended Markup
TIFF	Images	Tagged Image File Format
PS	Pages	PostScript (adobe)
PDF	Pages	Portable Document Format

File format is a type.

“Integer” is a way to interpret 4 bytes.

✓ 00 00 40 40 = “16448”

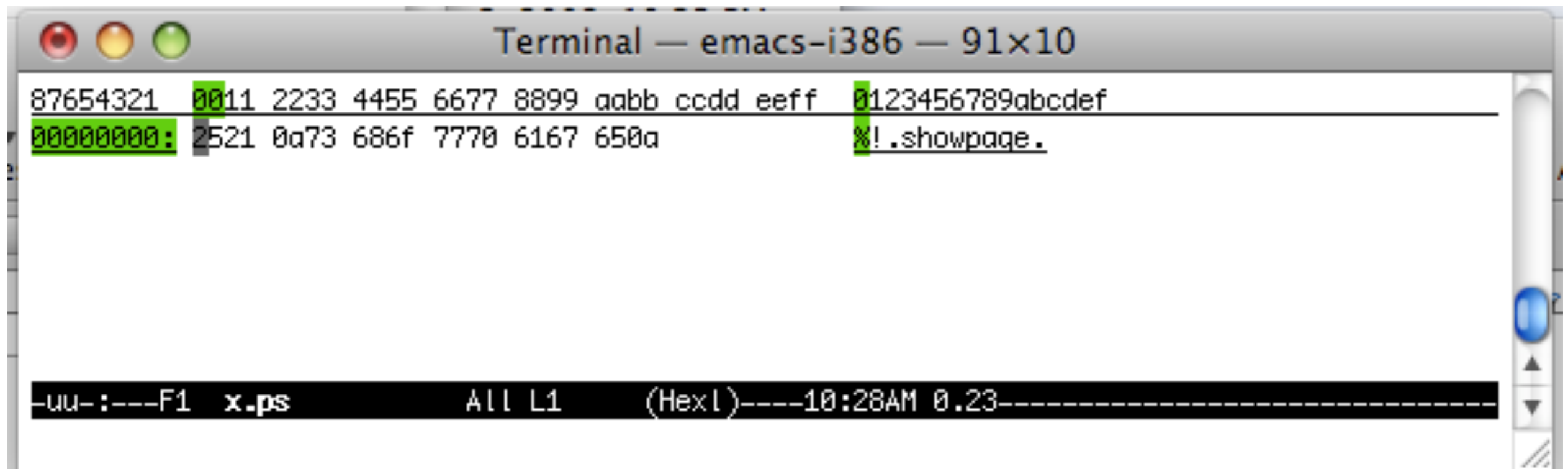
“Float” is a different way to interpret 4 bytes.

✓ 00 00 40 40 = “3.0”

“PostScript” is a way to interpret a string of bytes that begins 25 21:

✓ 25 21 0a 73 68 67 77 70 61 67 65 0a =
show a blank page

EMACS "Hexl" mode is good for looking at hex buffers:



The screenshot shows a terminal window titled "Terminal — emacs-i386 — 91x10". The main content area displays a hex buffer in Hexl mode. The first line shows a sequence of hex values: 87654321, 0011, 2233, 4455, 6677, 8899, aabb, ccdd, eeff, followed by a vertical bar and the ASCII string 0123456789abcdef. The second line shows 00000000:, 2521, 0a73, 686f, 7770, 6167, 650a, followed by a vertical bar and the command %!.showpage.. The status bar at the bottom shows: -uu-:---F1 x.ps All L1 (Hexl)---10:28AM 0.23-----

WinHex is an excellent forensic tool for doing this on Windows.

<http://www.x-ways.net/winhex/>

The same sequence of bytes may be interpreted different ways.

What does this mean:

✓ 25 21 0a 73 68 67 77 70 61 67 65 0a

It could be:

- Integers 622922355, 1751611248, 1634166026
- A blank page (postscript)
- ASCII text: %!/nshowpage/n

There may be a *likely* interpretation.

There may be *multiple correct* interpretations.

There may be no *right* interpretation.

“File type” is a phrase that denotes the *type* of a byte sequence.

There are many ways to determine a file type:

- By extension. (e.g. filename.ps)
- By MIME-type

```
$ wget --quiet --save-headers http://www.simson.net/x.ps
$ cat x.ps
HTTP/1.1 200 OK
Date: Sun, 19 Oct 2008 17:41:19 GMT
Server: Apache/2.0.61 (Unix) PHP/4.4.7 mod_ssl/2.0.61 OpenSSL/0.9.7e
mod_fastcgi/2.4.2 DAV/2 SVN/1.4.2
Last-Modified: Sun, 19 Oct 2008 17:32:15 GMT
ETag: "70629c-c-9452c5c0"
Accept-Ranges: bytes
Content-Length: 12
MS-Author-Via: DAV
Keep-Alive: timeout=2, max=100
Connection: Keep-Alive
Content-Type: application/postscript

%!
showpage
```

By inspection (e.g. “%!” as the first two bytes)

Technical aspects of file formats:

Explicit header? — Does the file start with a specific sequence?

Explicit footer? — Does the file end with a specific sequence?

Explicit metadata? — Are content & metadata separate?

Structured/Free format — Is there a grammar?

Self-validating? — Does the format have internal checks?

Compressed? — Is data expanded when it is used?

Memory dump? — Does the format match memory?

Container? — Can the format contain other objects? What kinds?

- Chunk-based formats (TIFF, PNG, JPEG, etc)
- Directory-based formats (ZIP, tar)

How do these characteristics show up?

XBM format <http://www.simson.net/smile.xbm>

Consider this picture “smile:”

```
#define smile_width 24
#define smile_height 23
static char smile_bits[] = {
    0x00, 0x00, 0x00, 0x00, 0x1F, 0x00, 0xC0, 0xFF, 0x00, 0xE0, 0xC0, 0x03,
    0x78, 0x00, 0x07, 0x18, 0x00, 0x06, 0x1C, 0x00, 0x0C, 0x0C, 0x00, 0x0C,
    0xC4, 0xC1, 0x18, 0xC6, 0xE1, 0x18, 0x06, 0x00, 0x18, 0x06, 0x00, 0x18,
    0x06, 0x00, 0x18, 0x84, 0x40, 0x08, 0x8C, 0xC1, 0x1C, 0x9C, 0x7B, 0x0C,
    0x18, 0x3F, 0x06, 0x70, 0x80, 0x07, 0xF0, 0xC0, 0x03, 0xC0, 0xFF, 0x00,
    0x00, 0x3F, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, };
```



- <http://en.wikipedia.org/wiki/XBM>

XBM format:

Developed in 1980s for representing X bitmaps.

Explicit header	Yes. #defines
Explicit footer	Yes. };\n
Explicit metadata	No.
Structured?	Yes.
Self-validating?	No. Any valid C+.
Compressed?	No. Expanded.
Memory dump?	No.
Container?	No.
Open/Closed?	Open. Unprotected.

How do these characteristics show up?

PPM format <http://www.simson.net/smile.ppm>

Consider this picture “smile:”



```
$ od -h smile.ppm
```

```
 2034 3332 320a 3535 ff0a ffff
0000020 ffff ffff ffff ffff ffff ffff ffff ffff
*
0000140 ffff ffff ffff ffff ffff ffff 00ff 0000
0000160 0000 0000 0000 0000 0000 0000 ffff ffff
0000200 ffff ffff ffff ffff ffff ffff ffff ffff
*
0000240 ffff ffff ffff ffff ffff ffff ffff 00ff
0000260 0000 0000 0000 0000 0000 0000 0000 0000
0000300 0000 0000 0000 0000 0000 0000 ff00 ffff
0000320 ffff ffff ffff ffff ffff ffff ffff ffff
*
0000360 ffff ffff 0000 0000 0000 0000 ff00 ffff
0000400 ffff ffff ffff ffff ffff ffff ffff 00ff
0000420 0000 0000 0000 0000 0000 ff00 ffff ffff
0000440 ffff ffff ffff ffff ffff ffff ffff ffff
0000460 ffff ffff ffff 0000 0000 0000 0000 0000
0000500 0000 ffff ffff ffff ffff ffff ffff ffff
0000520 ffff ffff ffff ffff ffff ffff 00ff 0000
0000540 0000 0000 0000 ffff ffff ffff ffff ffff
```

PPM format:

The Portable Pixmap File Format (part of Netpbm)

Explicit header	P1 / P2 / P3 / P4 / P5 / P6
Explicit footer	No
Explicit metadata	In some formats
Structured?	Not really.
Self-validating?	Not really.
Compressed?	No.
Memory dump?	No.
Container?	No.
Open/Closed?	Open. Unprotected.

How do these characteristics show up?

GIF format <http://www.simson.net/smile.gif>

Consider this picture “smile:”



```
$ cat -v smile.gif
```

```
[xy]$ cat -v smile.gif
```

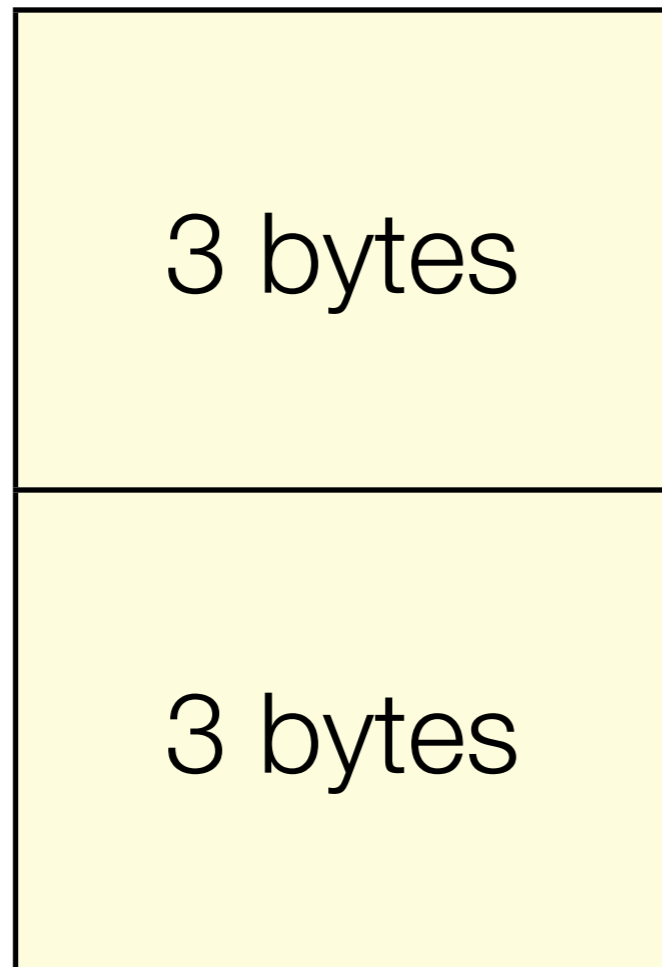
```
GIF89a^X^@^W^@M-p^@^@M-^?M-^?M-^?^@^@^@!M-  
y^D^@^@^@^@^@,^@^@^@^@^@X^@^W^@^@^BEM-^DM-^OM-)M-^KM-aM-NM-bM-^[ /6M-^XM-  
&M-,M-^L+lM-^AW`^]M-"qnM-^_yM-^BM-"M-{M-BWM-^YM-^Q,=M-_8jM-^[M-MM-jM-+M-  
it^^M-^^M--M-"M-a-`I@M-^JM-UM-+=M-^MQ*M-^MBM-YQM-'M-ZM-.M-WP^@^@;[xy]$
```

```
$ od -h smile.gif
```

```
0000000 4947 3846 6139 0018 0017 00f0 ff00 ffff  
0000020 0000 2100 04f9 0000 0000 2c00 0000 0000  
0000040 0018 0017 0200 8445 a98f e18b e2ce 2f9b  
0000060 9836 aca6 2b8c 816c 6057 a21d 6e71 799f  
0000100 a282 c2fb 9957 2c91 df3d 6a38 cd9b abea  
0000120 74e9 9e1e a2ad 2de1 4960 8a40 abd5 8d3d  
0000140 2a51 428d 51d9 daa7 d7ae 0050 3b00  
0000156  
$
```

[http://en.wikipedia.org/wiki/Graphics Interchange Format](http://en.wikipedia.org/wiki/Graphics_Interchange_Format)

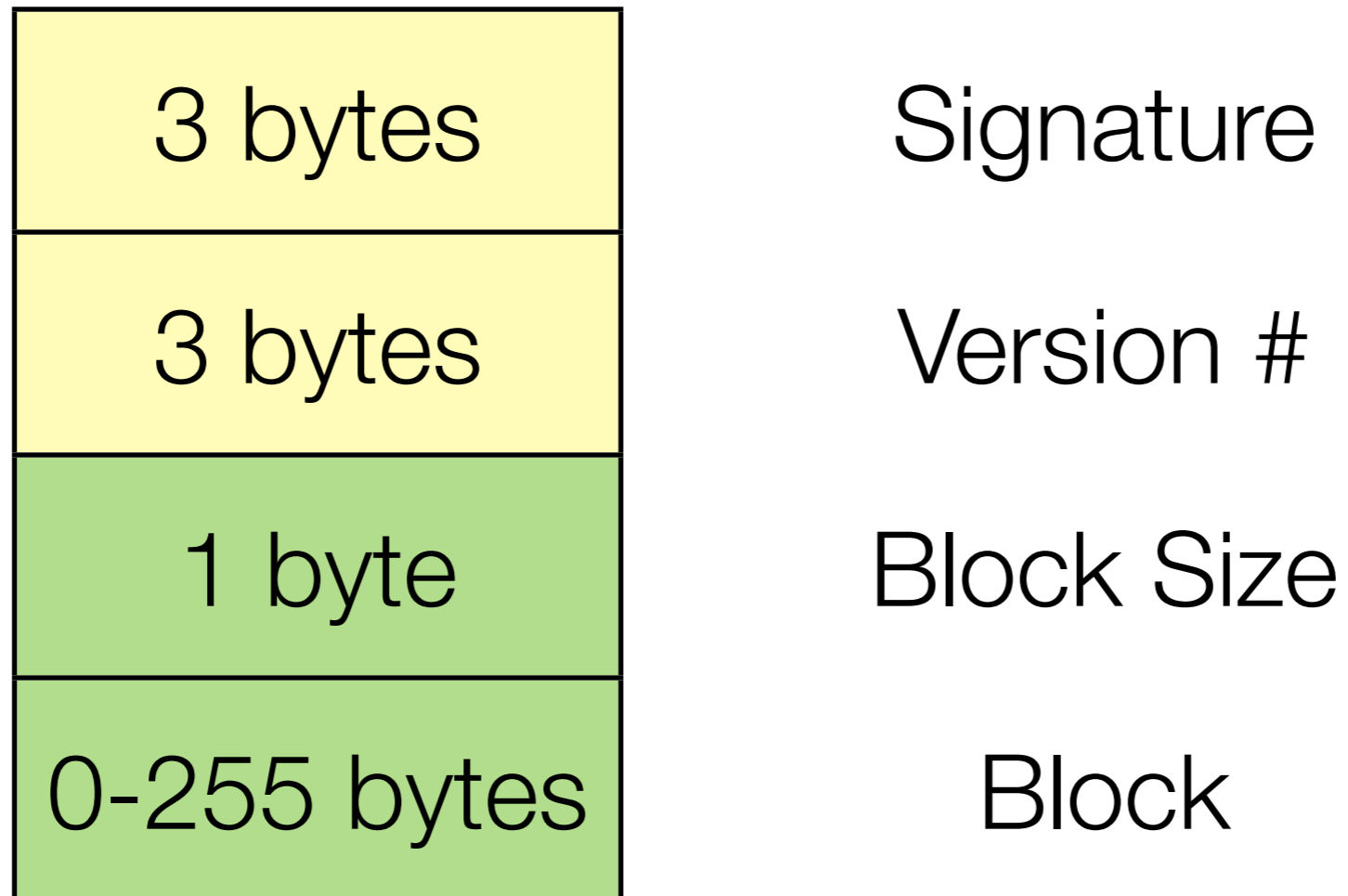
GIF files begin with a Header



Signature

Version #

GIF stores data in variable-sized blocks.



GIF has an OPTIONAL color table...

19. Global Color Table.

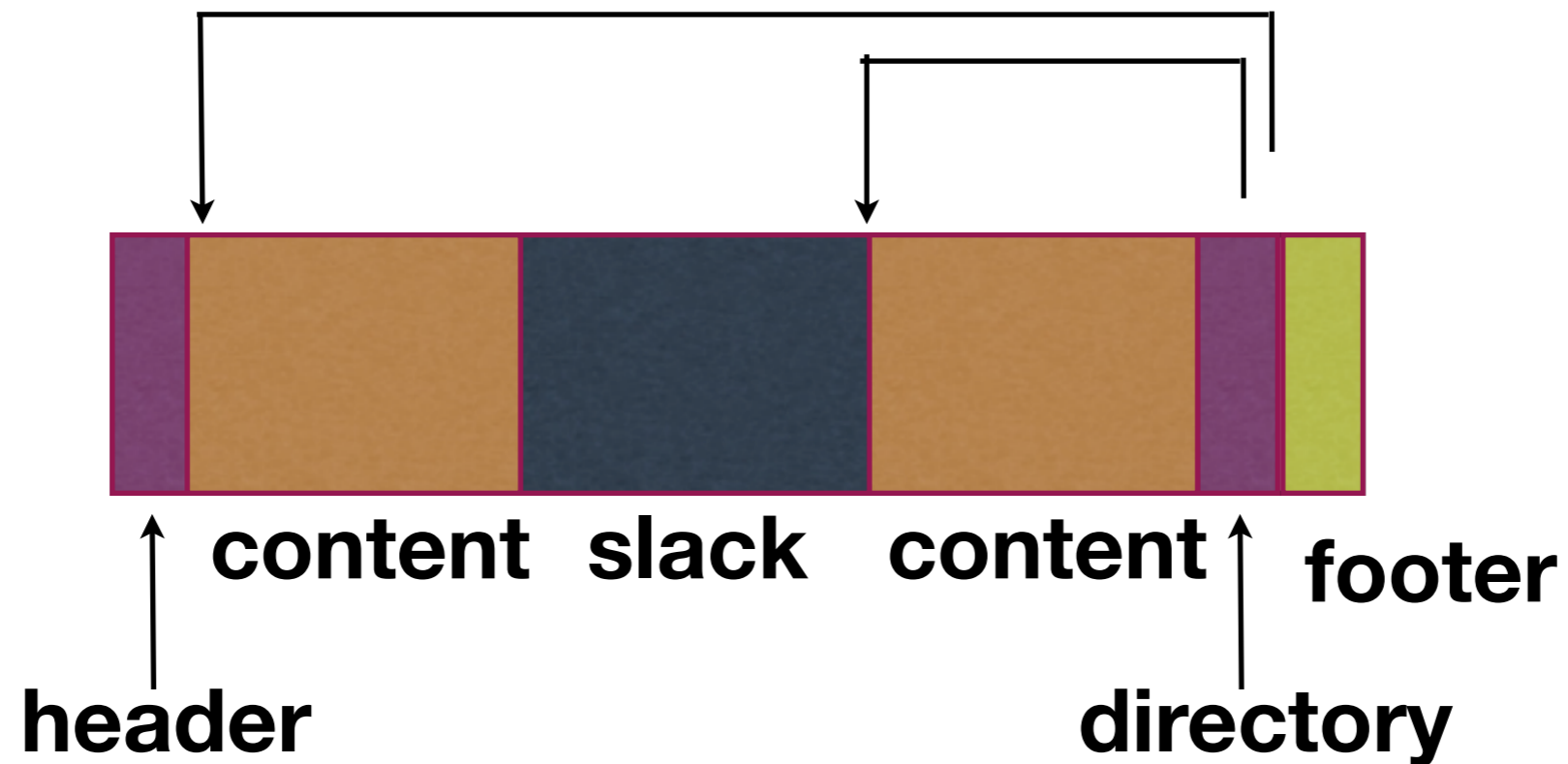
	7	6	5	4	3	2	1	0	Field Name	Type
	+=====+									
0									Red 0	Byte
	+-							-+		
1									Green 0	Byte
	+-							-+		
2									Blue 0	Byte
	+-							-+		
3									Red 1	Byte
	+-							-+		
									Green 1	Byte
	+-							-+		
up										
	+-			-+	...	
to										
	+-							-+		
									Green 255	Byte
	+-							-+		
767									Blue 255	Byte
	+=====+									

GIF format:

Developed in 1980s for representing bitmaps.

Explicit header	Yes. “GIF89a”
Explicit footer	Yes. 3b
Explicit metadata	Yes. (Comments;
Structured?	Yes.
Self-validating?	Yes. (Internal checks.)
Compressed?	Yes.
Memory dump?	No.
Container?	Yes. Chunks.
Open/Closed?	Open today (patent expired)

Anatomy of an idealized file format:



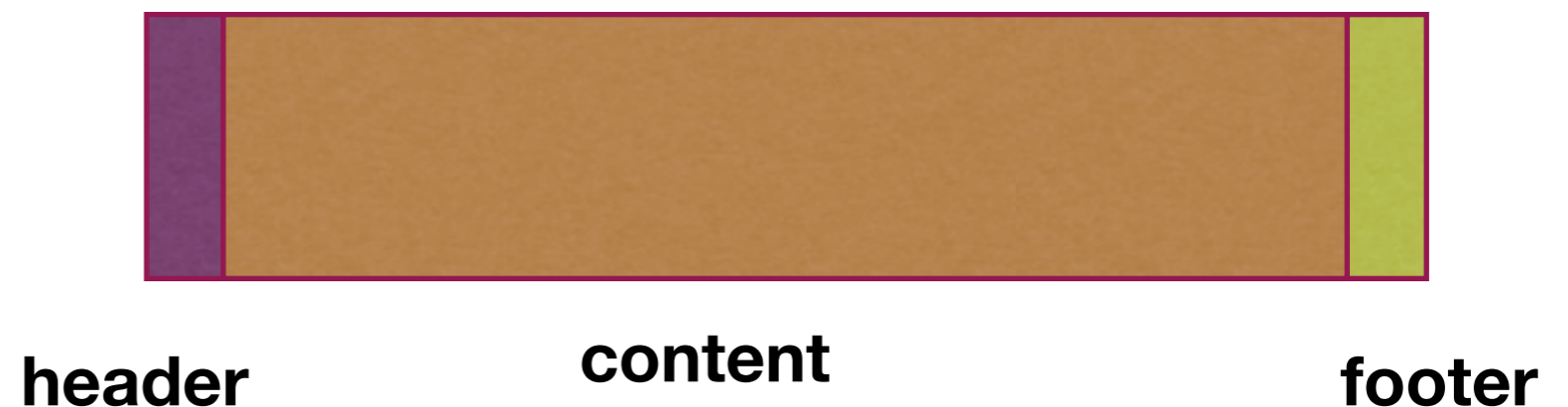
Not every file type has all of these parts.

HTML just has a header, content and footer:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
<html>  
Hi mom!  
</html>
```

Which might be missing:

```
<html>  
Hi mom!
```



JPEG files are similar to GIF files.

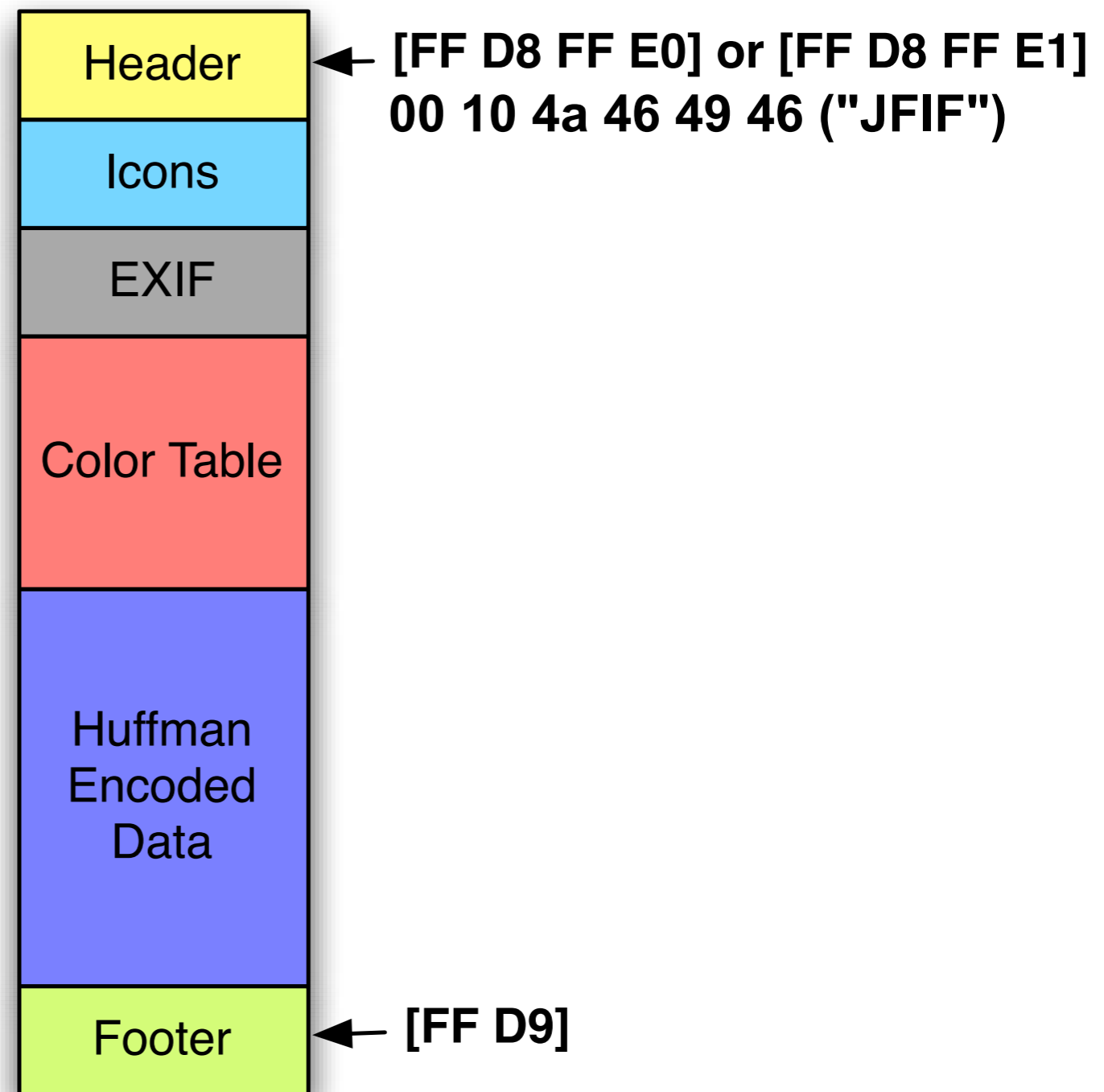
They have a header, sections, and a footer.

Icons may be:

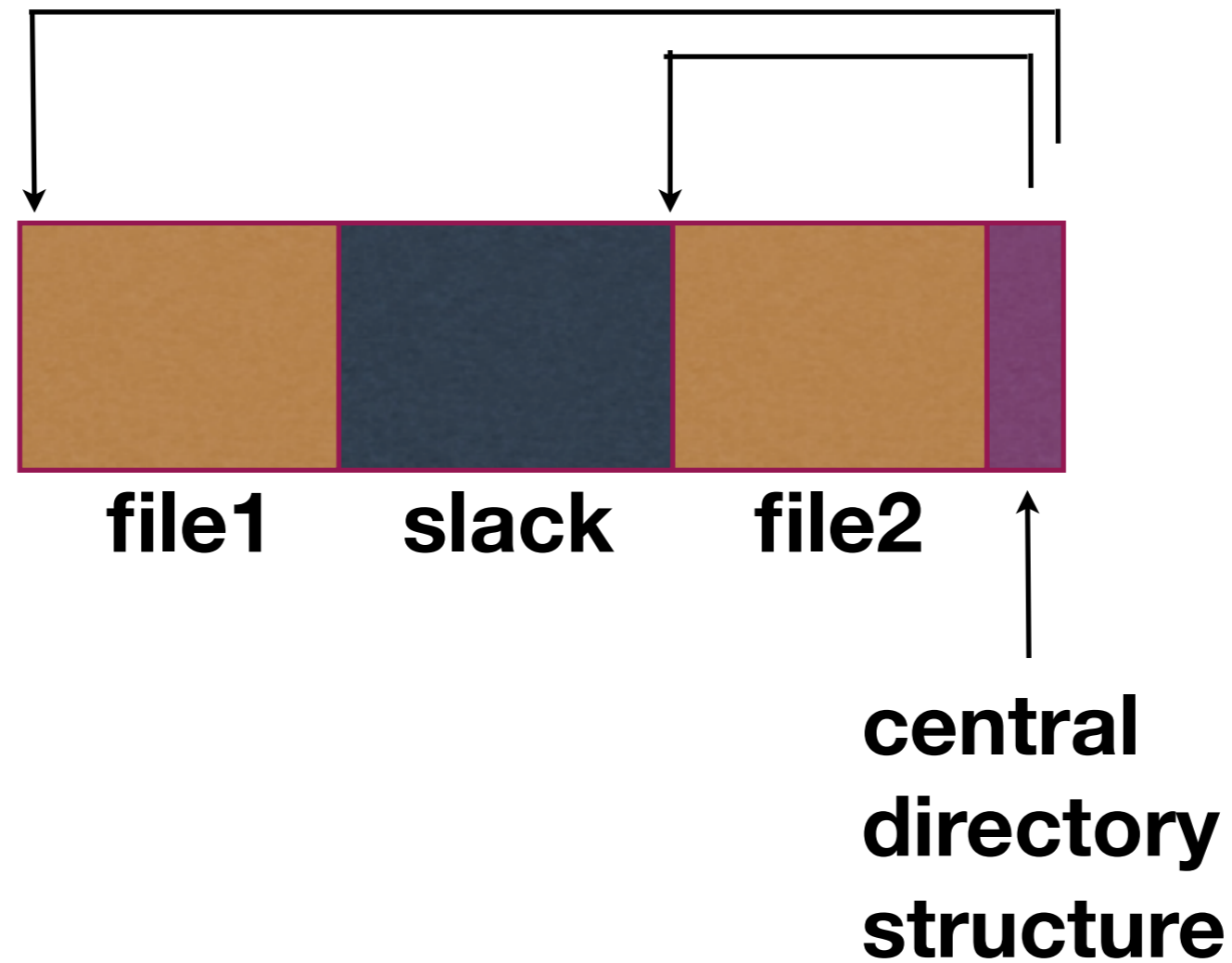
- Bitmaps
- Other JPEG files.
- Missing

The optional EXIF section stores metadata:

- Camera make & model
- Time
- Other info.



ZIP files are read from the back to the front.
There is no header.



ZIP format is described in the PKWare APPNOTE.TXT

<http://www.pkware.com/documents/casestudies/APPNOTE.TXT>

V. General Format of a .ZIP file

Files stored in arbitrary order. Large .ZIP files can span multiple volumes or be split into user-defined segment sizes. All values are stored in little-endian byte order unless otherwise specified.

Overall .ZIP file format:

```
[local file header 1]
[file data 1]
[data descriptor 1]
.
.
.
[local file header n]
[file data n]
[data descriptor n]
[archive decryption header]
[archive extra data record]
[central directory]
[zip64 end of central directory record]
[zip64 end of central directory locator]
[end of central directory record]
```

The Local File Header prefixes each ZIP file:

A. Local file header:

local file header signature	4 bytes	(0x04034b50)
version needed to extract	2 bytes	
general purpose bit flag	2 bytes	
compression method	2 bytes	
last mod file time	2 bytes	
last mod file date	2 bytes	
crc-32	4 bytes	
compressed size	4 bytes	
uncompressed size	4 bytes	
file name length	2 bytes	
extra field length	2 bytes	
file name (variable size)		
extra field (variable size)		

What happens if you concatenate a JPEG and a ZIP file?





something.zip

File Identification

How do you figure out what's in a file?

Option #1: Look at file extension or mime type.

✓ filename.doc

Advantages:

- It's easy.
- It's what the operating system does.

Disadvantages:

- Adversaries can hide data by changing the extension.
- There might not *be* a file extension.

TCP streams

Recovered data (e.g. file carving)

How do you figure out what's in a file?

Option #2: Look at file header

%!

GIF89a

<HTML>

PK

MZ

P1

Advantages:

- It's still pretty easy.

Disadvantages:

- Not all file formats have characteristic headers.
- The same header may imply multiple file formats.
- An adversary may put a bogus header on the file.

How do you figure out what's in a file?

Option #3: Evaluate the file contents

```
300 400 rlineto  
<h3>Better things</h3>  
0x00, 0x07, 0x18, 0x00, 0x06, 0x1C
```

Advantages:

- You might be able to do it without all the document.
- You'll get a paper published! *This is research.*

Disadvantages:

- Not all file formats are documented.
- Contents may code more than one file format.
- How do you evaluate? Open in an application?

State-of-the-art File Identification: Open Source

libmagic

- C;
- Rules in /usr/share/file/magic and compiled at runtime.
- Unix “file” command
- <http://sourceforge.net/projects/libmagic>

DROID

- Java
- Developed by National Archives of UK
- <http://droid.sourceforge.net>

State of the art file identification: Proprietary

TrID

- XML config file
- Closed source; free for non-commercial use
- <http://mark0.net/soft-trid-e.html>

Oracle Outside-In

- Proprietary but free demo.
- http://www.oracle.com/technology/products/content-management/oit/oit_all.html

FileAlyzer

- <http://www.safer-networking.org/en/filealyzer/index.html>

Libmagic configuration file specifies rules for understanding files.

/usr/share/file/magic:

```
# JPEG images
#
0          beshort          0xffd8          JPEG image data
!:mime    image/jpeg
!:apple   8BIMJPEG
!:strength +1
>6        string          JFIF              \b, JFIF standard
# The following added by Erik Rossen <rossen@freesurf.ch> 1999-09-06
# in a vain attempt to add image size reporting for JFIF. Note that these
# tests are not fool-proof since some perfectly valid JPEGs are currently
# impossible to specify in magic(4) format.
# First, a little JFIF version info:
>>11      byte            x                \b %d.
>>12      byte            x                \b%02d
# Next, the resolution or aspect ratio of the image:
#>>13     byte            0                \b, aspect ratio
#>>13     byte            1                \b, resolution (DPI)
#>>13     byte            2                \b, resolution (DPCM)
#>>4      beshort          x                \b, segment length %d
# Next, show thumbnail info, if it exists:
>>18      byte            !0              \b, thumbnail %dx
>>>19     byte            x                \b%d
```

The file command prints identification in human-readable form.

```
$ file simson.net/smile.gif
```

```
simson.net/smile.gif: GIF image data, version 89a, 24 x 23
```

```
$ file simson.net/smile.*
```

```
simson.net/smile.gif: GIF image data, version 89a, 24 x 23
```

```
simson.net/smile.jpg: JPEG image data, JFIF standard 1.01
```

```
simson.net/smile.png: PNG image data, 24 x 23, 1-bit grayscale, non-interlaced
```

```
simson.net/smile.ppm: Netpbm PPM "rawbits" image data
```

```
simson.net/smile.raw: ASCII C program text
```

```
simson.net/smile.xbm: ASCII C program text
```

```
$
```

File has many options

Some interesting ones:

- b, --brief

- c, --checking-printout

- i, --mime

- m, --magic-file list

- z, --uncompress

So what happens with JPEG+ZIP?



Read from the front and it's a JPEG

Read from the back and it's a ZIP

- This is an approach that is currently used to hide data.
- Moral: The same data may be read different ways by different programs.

McDaniel and Heydari proposed three statistical algorithms.

Byte Frequency Analysis (BFA) algorithm

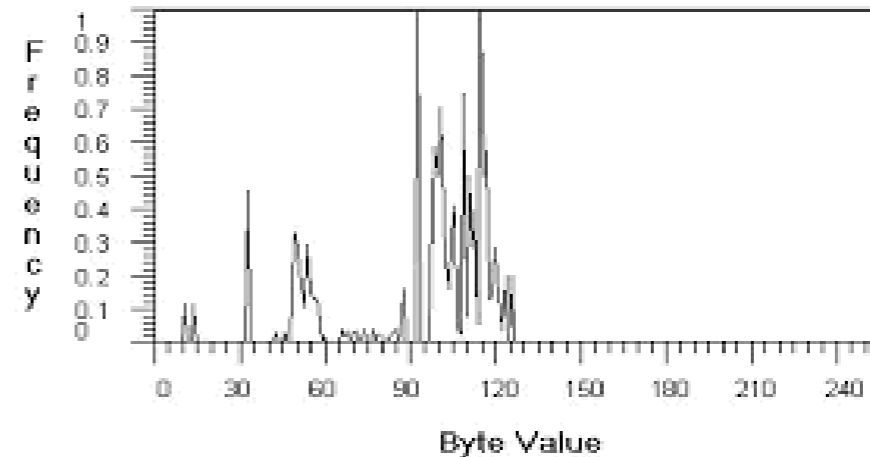


Figure II.1 - Byte frequency distributions for two RTF files.

Byte frequency cross-correlation measures how often two characters are correlated.

Observation: in HTML “<” is highly correlated with “>”

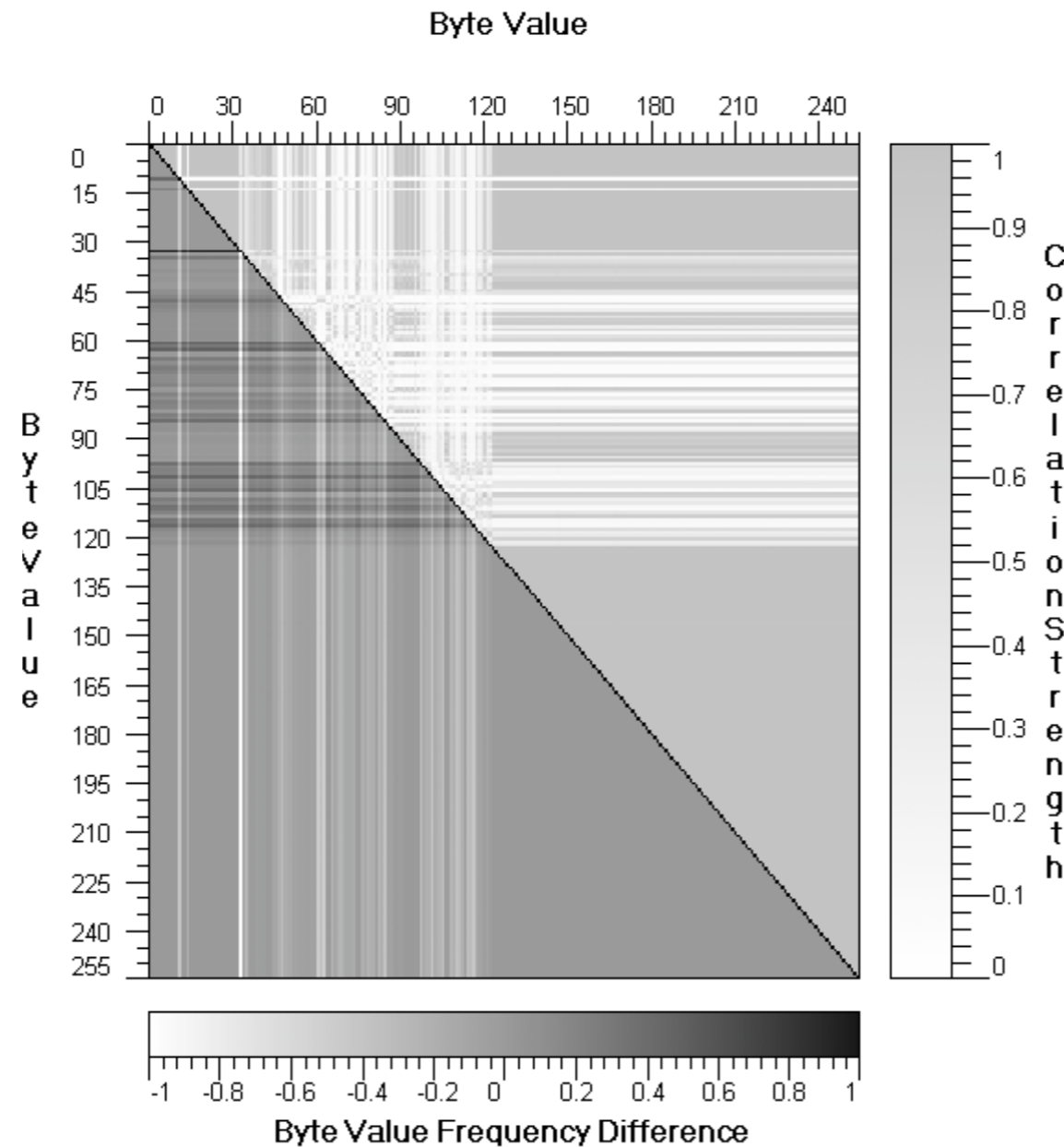


Figure II.9 - Byte frequency cross-correlation plot for the HTML file type

File header/trailer (FHT) algorithm

Analyzes file headers and trailers “to strengthen the recognition of many file types.”

“The file headers and trailers are patterns of bytes that appear in a fixed location at the beginning and end of a file...”

“These can be used to dramatically increase the recognition ability....”

Results in 2003

BFA is accurate to only 27.50%.

- “better than purely random guesses.”

BFC is accurate to 45.83%

- “a significant improvement over BFA, but not accurate enough for practical use.”

FHT's accuracy is 95.83%.

- “may be accurate enough for some fault-tolerant applications.”
- “We should note that using separate fingerprints for ACD, DOC, PPT and XLS files decreases FHT's accuracy to 85%, most of the errors occurred between the ACD, DOC, PPT and XLS file type identification.”

Fileprints: Wei-Jen Li, Ke Wang, Salvatore J. Stolfo, Benjamin Herzog, 2005

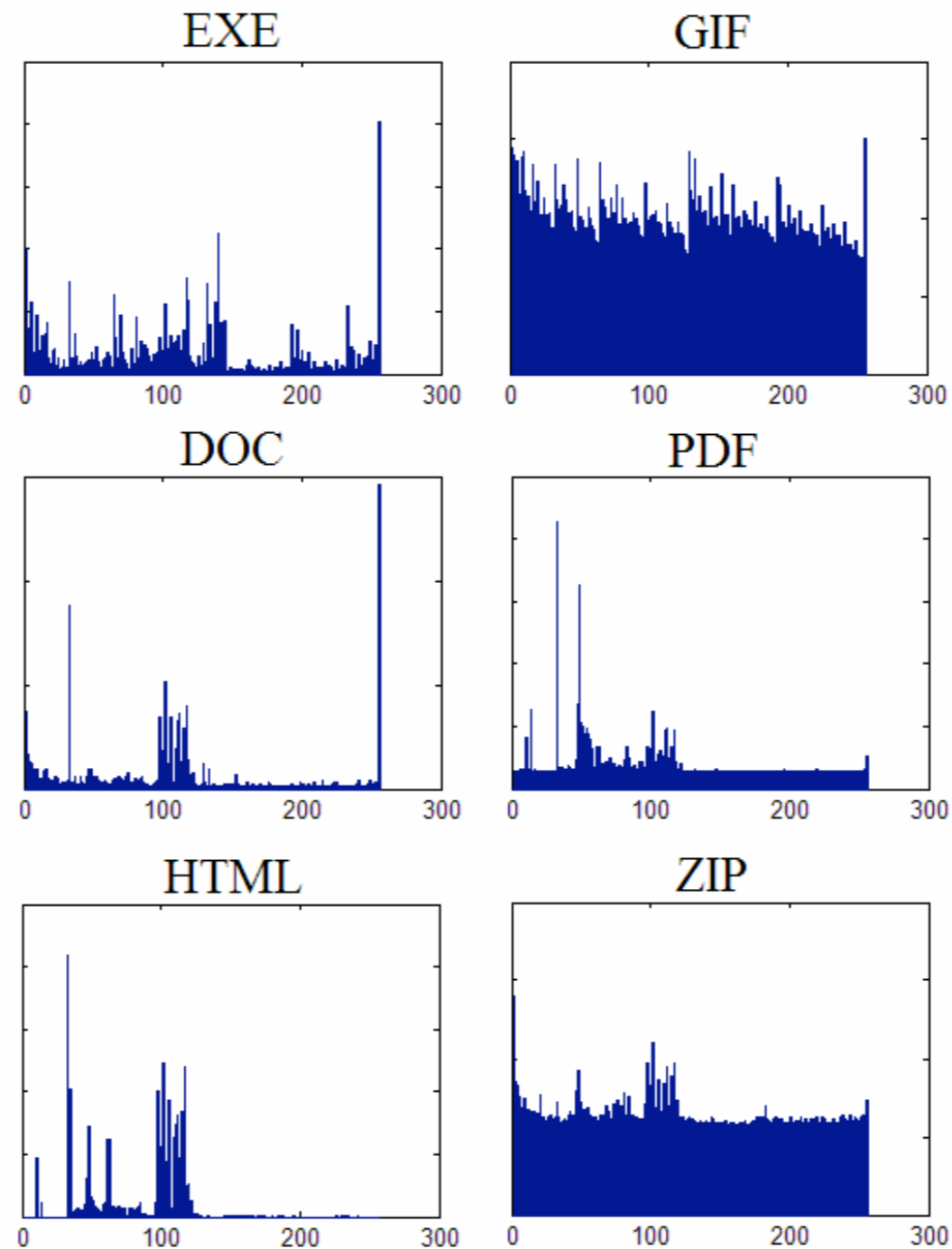


Figure 1: File binary distribution. X axis: bytes from 0 to 255, Y axis: normalized frequency of byte values (as %).

Fileprints uses n-gram analysis.

... it's sort of like correlation, but over a larger area.

“The choice of the window size depends on the application.”

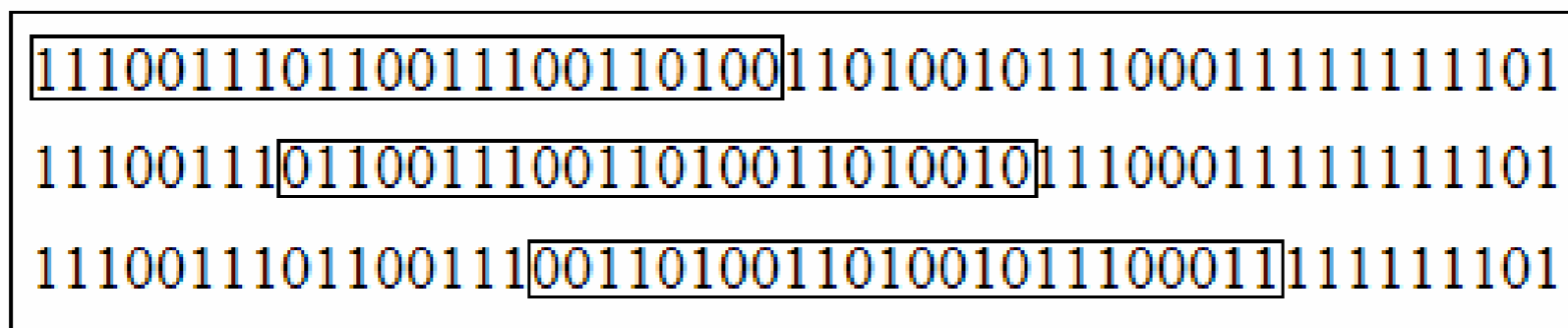


Figure 2: Sliding window (window size = 3)

Problem 1:

Different parts of the file have radically different

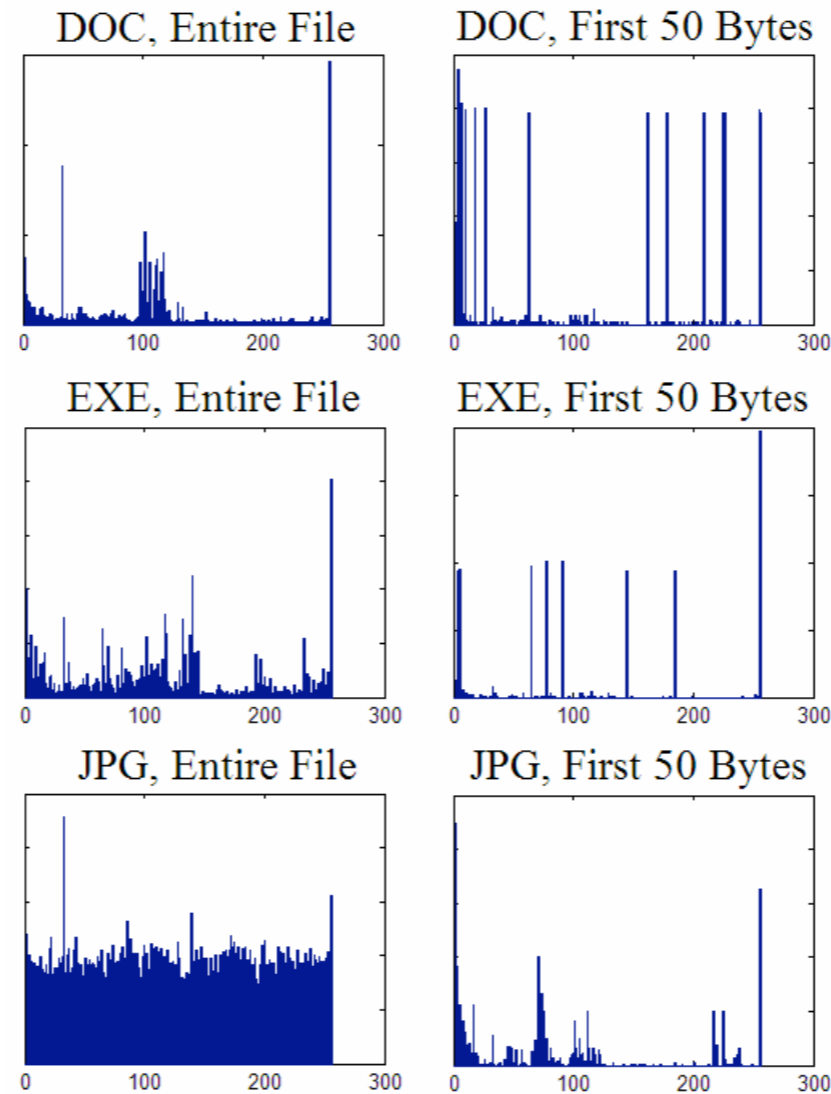


Figure 3: The byte value distributions of entire file (left column) and the first 50 bytes (right column) of the same file types. X-axis: bytes from 0 to 255, Y-axis: normalized frequency of byte values (as a %).

Problem #2: Some files are very similar.

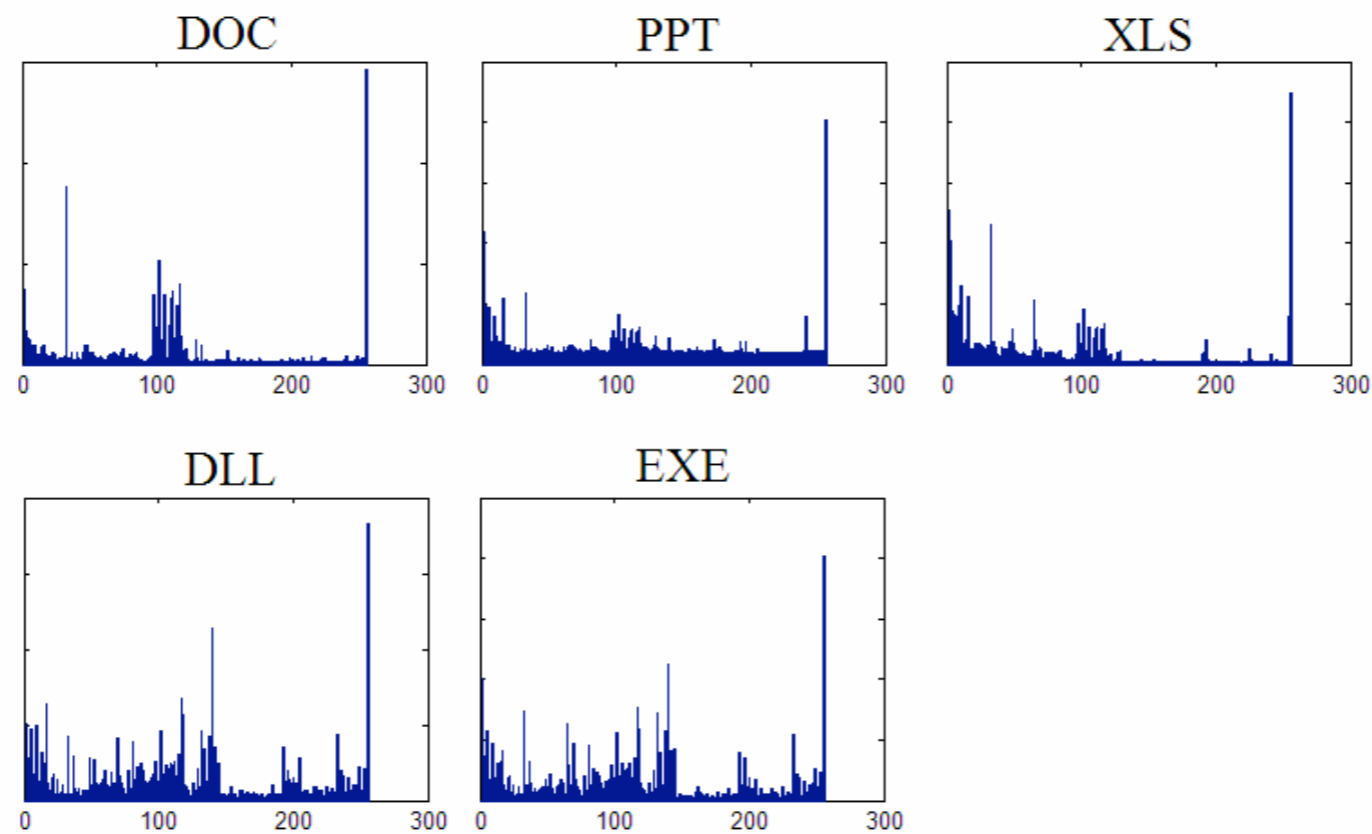


Figure 4: The bytes distribution of DLL and EXE files.
X axis: bytes from 0 to 255, Y axis: normalized frequency
of byte values (as a %).

Fileprints solution: K-means clustering

Obtain a corpus of training data.

Treat each file as a point in 256-dimension space.

Cluster by filetype.

Compute the distance to each centroid to identify.

Multi-centroids file type classifying accuracy						
Truncation Size	EXE	GIF	JPG	PDF	DOC	AVG.
20	99.9%	100%	98.9%	100%	98.8%	99.4%
200	97%	98.3%	96.6%	95%	97.2%	96.9%
500	97.2%	98.4%	94.8%	90%	96.9%	96%
1000	97%	95.1%	93.5%	90.7%	94.5%	94.6%
All	88.9%	76.8%	85.7%	92.3%	94.5%	89.5%
Classifying accuracy using exemplar files as centroids						
Truncation Size	EXE	GIF	JPG	PDF	DOC	AVG.
20	100%	100%	100%	100%	98.9%	99.6%
200	99.4%	91.6%	99.2%	100%	98.7%	98.2%
500	99%	93.6%	96.9%	99.9%	98.5%	98%
1000	98.9%	94.9%	96.1%	86.9%	98.6%	96.4%
All	94.1%	93.9%	77.1%	95.3%	98.9%	93.8%

Table 1: The average accuracy of file type classifying test.
First Column: the truncation size, first 20, 200, 500 1000 byte, and the entire file. Other Columns: “EXE” represents the group which includes .EXE and .DLL. “DOC” represents the group which includes .DOC, .PPT and .XLS. “AVG.” represents the overall performance.

Follow-up work

File type identification of data fragments by their binary structure.,
Karresand Martin, Shahmehri Nahid. Proceedings of the IEEE
workshop on information assurance; 2006

- Largely based on Byte Frequency Distribution
- “detection rates in the range of 45% to 85% with false positive rates approaching 35%
- “The zip file specific implementations gave detection rates that in some cases came close to 13%, and false positive rates between 0.5% and 1.9%”

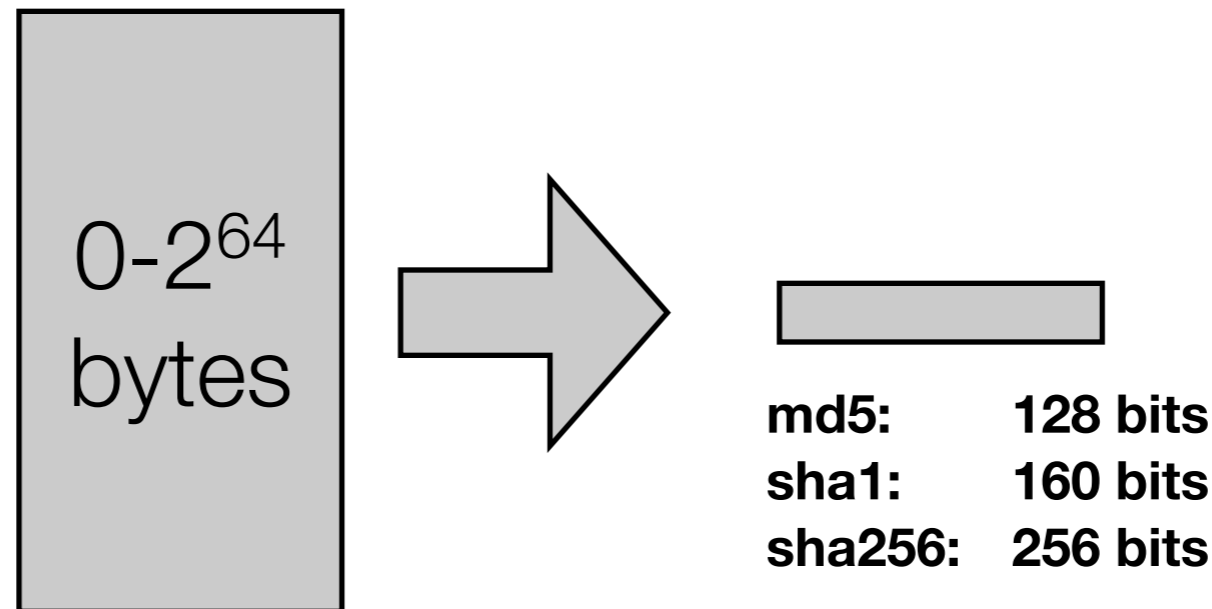
More follow-up work

Predicting the Types of File Fragments, William Calhoun, Drue Coles, DFRWS 2008



File Recognition

Hash functions create a "fingerprint" for a file.



Example:

```
$ md5 Microsoft/windows95.iso
MD5 (Microsoft/windows95.iso) = 089eff8ea0e3a5aa21aa1862bf244957
$
```

Properties of a good hash function:

- Change 1 bit in input, $\approx 50\%$ of bits in output change.
- Can't predict $H(m)$ from m
- Given H , it is hard to find m such that $H(m) = H$ (preimage resistance)
- Given m_1 , it's hard to find m_2 such that $H(m_1) = H(m_2)$ (secondary preimage resistance)
- Hard to find *any* m_1 and m_2 such that $H(m_1) = H(m_2)$ (collision resistance)

A hash set is a collection of hash values.

Here is a hash set of blanks:

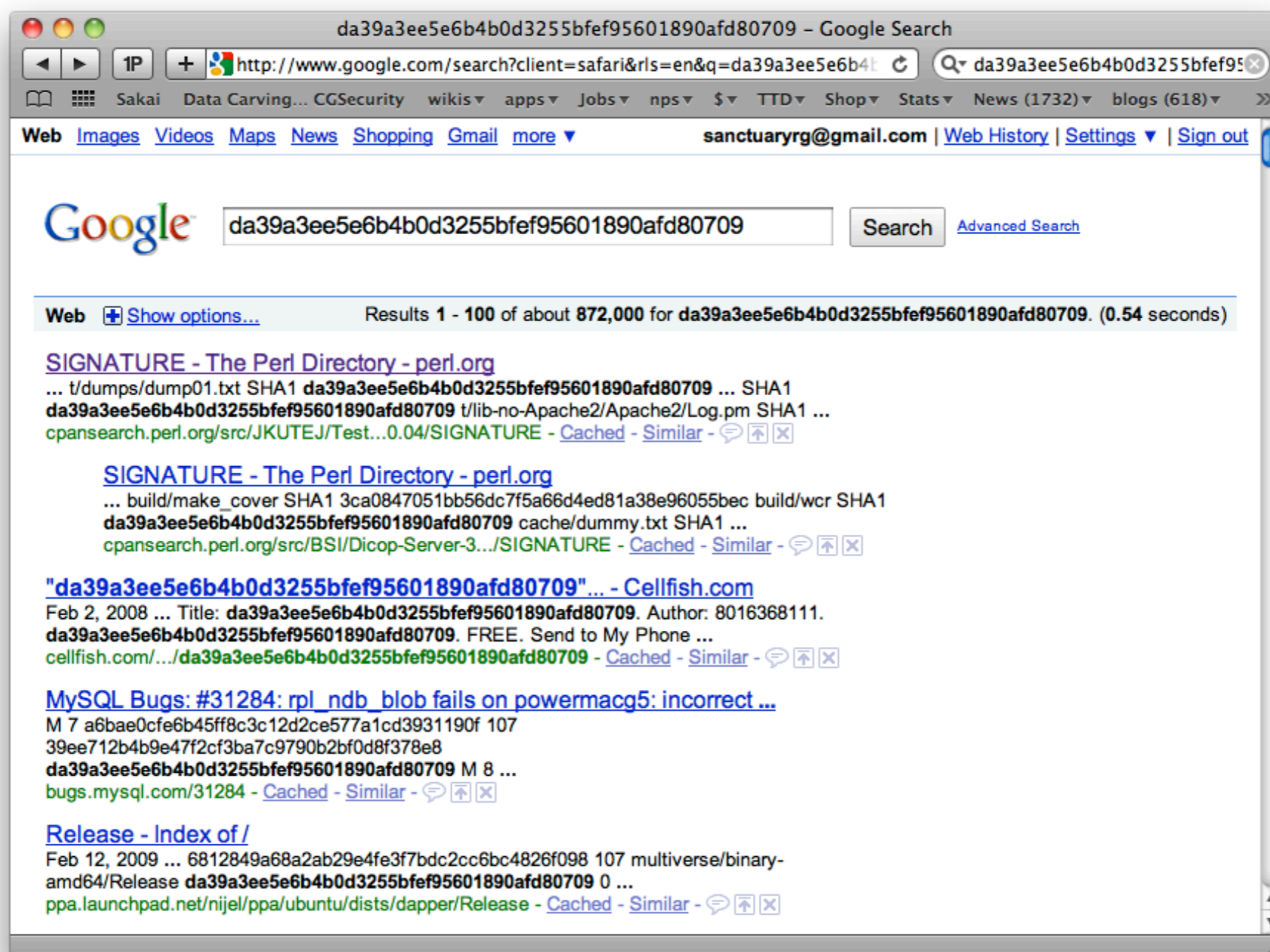
**da39a3ee5e6b4b0d3255bfe95601890afd80709
1ceaf73df40e531df3bfb26b4fb7cd95fb7bfff1d
5c3eb80066420002bc3dcc7ca4ab6efad7ed4ae5**

What are they?

These are all SHA1 codes of "nothing."

You can identify hash codes with Google.

filename	size	SHA1
/dev/null	0	da39a3ee5e6b4b0d3255bfef95601890afd80709
4096	4096	1ceaf73df40e531df3bfb26b4fb7cd95fb7bff1d
512	512	5c3eb80066420002bc3dcc7ca4ab6efad7ed4ae5



CPAN (and others) distribute signed SHA1 codes to insure the validity of the distribution.

This file contains message digests of all files listed in MANIFEST, signed via the Module::Signature module, version 0.55.

To verify the content in this distribution, first make sure you have Module::Signature installed, then type:

```
% cpansign -v
```

It will check each file's integrity, as well as the signature's validity. If "==> Signature verified OK! <==" is not displayed, the distribution may already have been compromised, and you should not run its Makefile.PL or Build.PL.

-----BEGIN PGP SIGNED MESSAGE-----

Hash: SHA1

```
SHA1 946066eb52585118ba9a082c0a1394f612b38b30 Build.PL
SHA1 6af182cc968ce9348d85a0192bec1d27ed8b0d2e Changes
SHA1 531b0f15f11b92a8fed3a194f79e0aa2249c84d4 MANIFEST
SHA1 39bb94d2441c7c561e2d2b423650d3abd2be9a1b META.yml
SHA1 8f18752535441f667b3be68d462b7847a517aab5 Makefile.PL
SHA1 91120916c8c7b038f5e201e8d1268167e91b8aa2 README
SHA1 4511cd2e17c98cafd2ae25638b8899ad012f6f328 debian/changelog
SHA1 5d9474c0309b7ca09a182d888f73b37a8fe1362c debian/compat
```

You can use these SHA1 codes to recognize the file.

The NIST Reference Data Set is a collection of hash codes distributed for file recognition.

The Reference Data Set (RDS):

- Project of the National Institute of Standards and Technology
- Current version: 2.26 (September 2009)
- Distributed as four ISO images
- Can be directly imported into:
 - ✓ EnCase5 & Encase 6
 - ✓ Hashkeeper
 - ✓ Ilook
 - ✓ Vogon
- 53,575,618 files; 16,510,746 unique SHA-1 values.

Other hash sets:

- Child porn
- Malware
- Other "known goods" and "known bads."



**[http://
www.nsrl.nist.gov/](http://www.nsrl.nist.gov/)**

NSRFile.txt:

```
"SHA-1", "MD5", "CRC32", "FileName", "FileSize", "ProductCode", "OpSystemCode", "SpecialCode"

"00000142988AFA836117B1B572FAE4713F200567",
  "9B3702B0E788C6D62996392FE3C9786A", "05E566DF", "J0180794.JPG", 32768, 2322, "WIN", ""

"00000142988AFA836117B1B572FAE4713F200567",
  "9B3702B0E788C6D62996392FE3C9786A", "05E566DF", "J0180794.JPG", 32768, 3271, "WIN", ""

"00000142988AFA836117B1B572FAE4713F200567",
  "9B3702B0E788C6D62996392FE3C9786A", "05E566DF", "J0180794.JPG", 32768, 3290, "WIN", ""

"000005EE5E3F6961B78CE4549270DE5D05CBC0CB",
  "8D025B6AE1994A40FCBB5AEC2EF273F9", "5E8D7D42", "WabIab.bor", 4760, 4616, "WIN", ""

"0000085FC602CD8AD4793A874A47D286DACB0F6A",
  "8BA8BC04896C421A704282E9B87B5520", "8D89A85D", "fpSDtFindLink.gif", 1161, 2988,
"Solaris", ""

"00000FF9D0ED9A6B53BC6A9364C07074DE1565F3",
  "A5D49D6DA9D78FD1E7C32D58BC7A46FB", "2D729A1E", "cmnres.pdb.dll", 76800, 1550, "WIN", ""

"00000FF9D0ED9A6B53BC6A9364C07074DE1565F3",
  "A5D49D6DA9D78FD1E7C32D58BC7A46FB", "2D729A1E", "cmnres.pdb.dll", 76800, 2704, "WIN", ""
```


NSRLOS.txt

```
"AIX", "AIX", "Generic", "Unknown"
"AIX43", "AIX 4.3", "4.3", "IBM"
"AIX432", "AIX 4.3.2", "4.3.2", "IBM"
"AIX433", "AIX 4.3.3", "NA", "Unknown"
"AIX51", "AIX 5.1", "5.1", "IBM"
"AS/400", "AS/400", "N/A", "Unknown"
"AT", "AT", "NA", "Unknown"
"AT&T", "AT&T", "Unknown", "AT&T"
"Amiga", "Amiga", "Unknown", "Unknown"
"Amstrad_6128", "Amstrad 6128", "Unknown", "Unknown"
"Apple_Iic", "Apple IIc", "Unknown", "Apple"
"Apple_II+", "Apple II+", "Unknown", "Apple"
"Apple_IIGS", "Apple IIGS", "Unknown", "Apple"
"Apple_IIE", "Apple IIE", "Unknown", "Apple"
"Atari_ST", "Atari ST", "Unknown", "Unknown"
"CE", "CE", "Unknown", "Microsoft"
"CommodoreAmiga", "Commodore Amiga", "Unknown", "Unknown"
"Commodore_64", "Commodore 64", "Unknown", "Unknown"
```

Many new uses for hash codes...

Profiling previous uses of a hard drive.

Looking for pieces of contraband or confidential documents.

- Hash each 512-byte or 4K sector.

In Summary:

Files, File Type Identification, File Recognition

A **file** is a collection of 0 or more bytes

- Safely $0-2^{64}$ bytes. (18,446,744 terrabytes)
- The file name, timestamp, etc. is not part of the file.

Container Files are files that contain structured components.

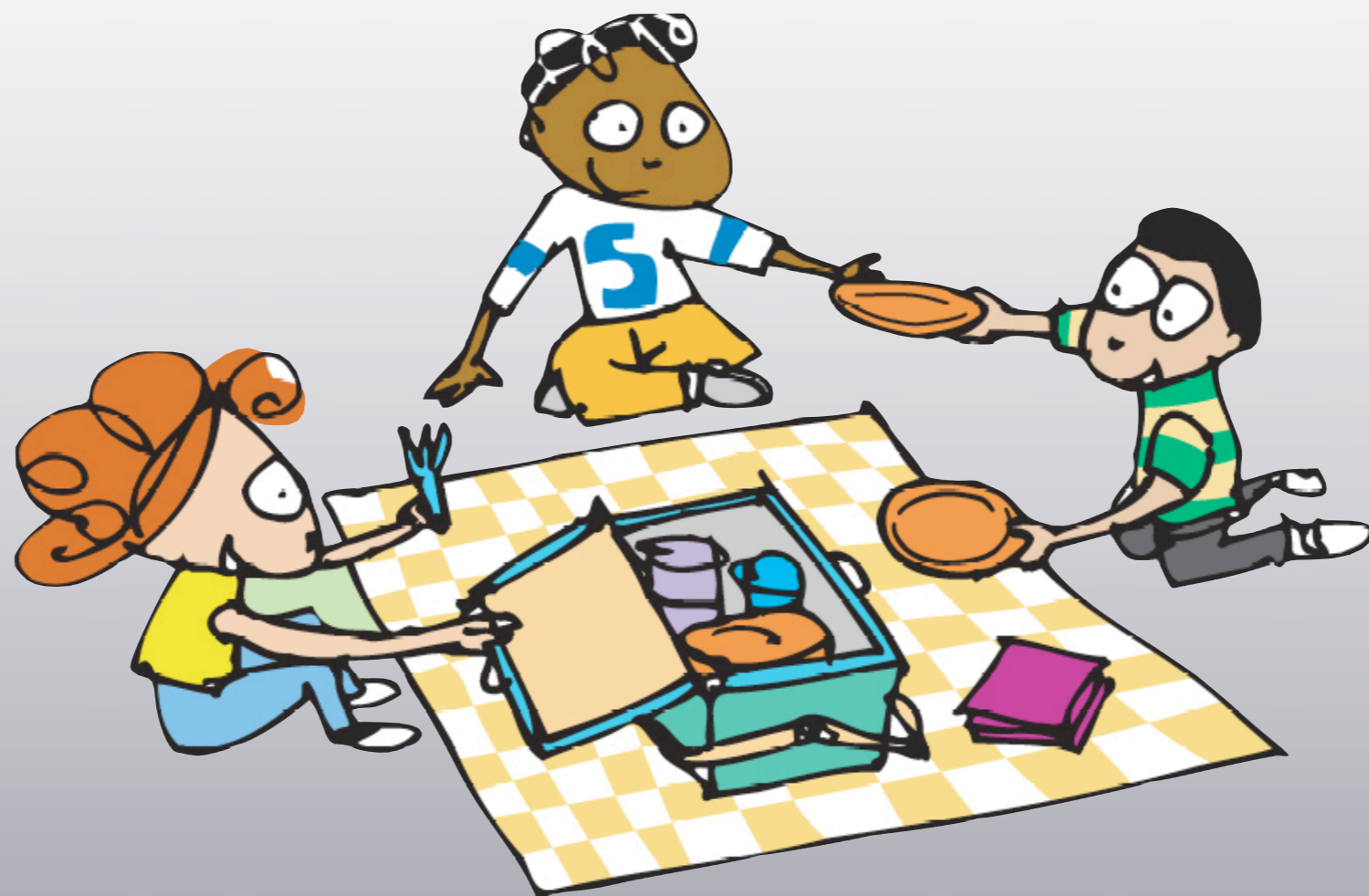
- JPEG, PDF, ZIP, Word, etc. are all container files.

File Type Identification determines the “type” of a file.

- Type is not “extension.”
- Identifying file fragments is a research problem.

File Recognition means matching the file to a corpus.

- Typically done with hash codes.
- Video matching is a research area.



Lunch!

This is an introductory tutorial!

Theory, Science and Tools

8:30 - 10:00 Introduction

- Introduction to Digital Forensics & The Law

10:00 - 10:30 Coffee

10:30 - 12:00 Data Analysis

- Unicode, File Formats & File Identification

12:00 - 1:30 Lunch

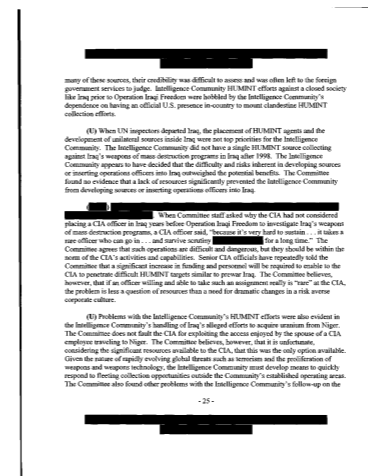
1:30 - 3:00 Disk Forensics

- Disk Imaging
- File Carving
- Sleuth Kit

3:00 - 3:30 Coffee

3:30 - 5:00 Big Finish

- Documents & Metadata
- Memory Forensics
- Anti-Forensics



United States Secret Service

WWW.SECRETSERVICE.GOV

BEST PRACTICES FOR SEIZING ELECTRONIC EVIDENCE



*A Joint Project of the International Association of Chiefs of Police
and the United States Secret Service*
iacp@secretservice.gov



Search

» [Search the website](#)

Powered by [USA.gov](#)

About

» [Home](#)

» [Who We Are](#)

- [Director](#)
- [Deputy Director](#)
- [Special Agents](#)
- [Uniformed Division](#)
- [Support Personnel](#)

» [Our Dual Mission](#)

» [Rowley Training Center](#)

» [History](#)

» [Frequently Asked Questions](#)

» [Student Q&A](#)

Introduction to Investigation

Before you start, consider these issues:

Officer safety — secure the scene and make it safe.



If you reasonably believe that the computer is involved in a ...
you are investigating, **take immediate steps to secure the evidence.**

Do you have a legal basis to seize the computer?

Do special legal considerations apply?

- Doctor, attorney, clergy, psychiatrist, newspapers, publishers, etc.?

http://www.secretservice.gov/electronic_evidence.shtml

Preparing the evidence at the scene of the crime.

Do not access any computer files.

- If the computer is off, leave it off
- If the computer is on, do not start searching through the computer.
- It is okay to move the mouse to activate the screen.

If a camera is available:

- Take pictures of the computer screen (if it is on)
- Photograph computer's front:
Work environment. Notes. Media on the desk.
- Photograph computer's rear:
Cables



Seize all media, notes, cables in the area. Label everything.
Document what you do.

Don't forget the computer's RAM!

The computer's RAM may contain:

- Discoverable evidence (e.g. logfiles, documents)
- Encryption keys

RAM can be captured with:

- Helix 2.0
- ramdd (Mantech)
- EnCase WinEn (300k executable)

Write the RAM to:

- External storage
- Server on the Network



Shutdown the computer

Photograph the computer's screen!

If the computer is networked:

- Unplug power to router or modem.

Desktops and laptops:

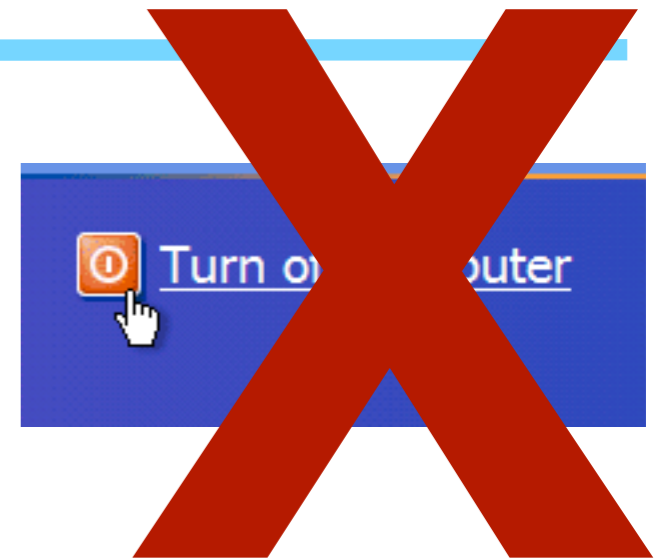
- Unplug the power from the computer.
- Unplug the the laptop battery from the laptop.

Servers:

- Pulling the plug may damage the system and interrupt business.
- Discuss with organization's IT management what to do.

PDAs and telephones:

- Do not turn off (may require a power-on password)
- Keep device charged; seize charger if possible!
- (Put in a metal bag or Faraday cage)



Talk to your suspect.

Ask for consent to search their computer/telephone/PDA.

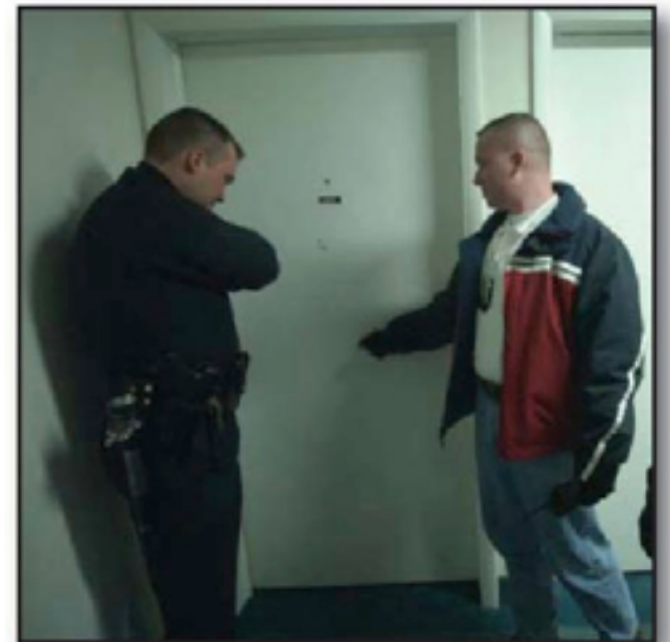
Ask for PINs or passwords.

See: "Knock and Talks," by Jayme W. Holcomb,

FBI Law Enforcement Bulletin

August 2006, volume 75, number 8

<http://www.fbi.gov/publications/leb/2006/august06leb.pdf>



© Mark C. Ide

What's "evidence" depends on the crime.

The computer (or data) is the contraband fruits of the crime:

- Stolen hardware, software, music

The computer is a tool of the offense:

- VISA logos are evidence of fake ID production
- The computer is a tool of the offense.

The computer is incidental to the offense:

- Drug dealer maintains trafficking records on computer

More than one may be true!

- Pedophile may have child porn, use the computer to contact children,



Set time limits.

There isn't enough time to learn *everything* about the
Different forensic examiners are looking for different



Police investigators:

- Most are looking for **proof of crime**.
- Some look for **additional suspects**.
- A few look for **owners of stolen equipment**.

Corporate investigators:

- Evidence of a breakin & methods used.
- Whether or not stolen data was accessed (analysis of MAC times)
- Software evaluation; debugging; data recovery



What's on the disk?

Residual Data
Disk Sanitizing

Disk forensics: Typical tasks

Recover:

- Deleted files
- Child pornography

Recreate:

- Timelines - when did the computer do what?
- Flow of information
- Evidence of Inappropriate use

Gather Intelligence:

- Names of associates
- Meeting places

Typical targets of disk forensics:
anything incriminating.

Email

Pictures

Internet History

Microsoft Word Files

PDF files

Spreadsheets

Chat Logs

Typical targets of disk forensics: anything incriminating.

Suspects may try to hide the data:

- ▶ Deleting Files
- ▶ Wiping Programs
- ▶ Formatting drives.
- ▶ Encryption
- ▶ Storing Data on online services
- ▶ Storing data on iPods, Cameras, and other devices

You have many options once you have the data.

Typical “first steps” include:

- Inventory all files (resident & deleted) on disk
- Determine clock skew (check HTTP files timestamps vs. filemod times)
- Show files modified during a certain time period
- Eliminate “known goods” (operating system files, etc.)
- Search for “known bads” (hacker tools, child porn)
- Scan for key words, email addresses

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.

directory

directory

.

..

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.

directory

directory

.

..

copy file.doc f:file.doc

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.

directory

file.doc

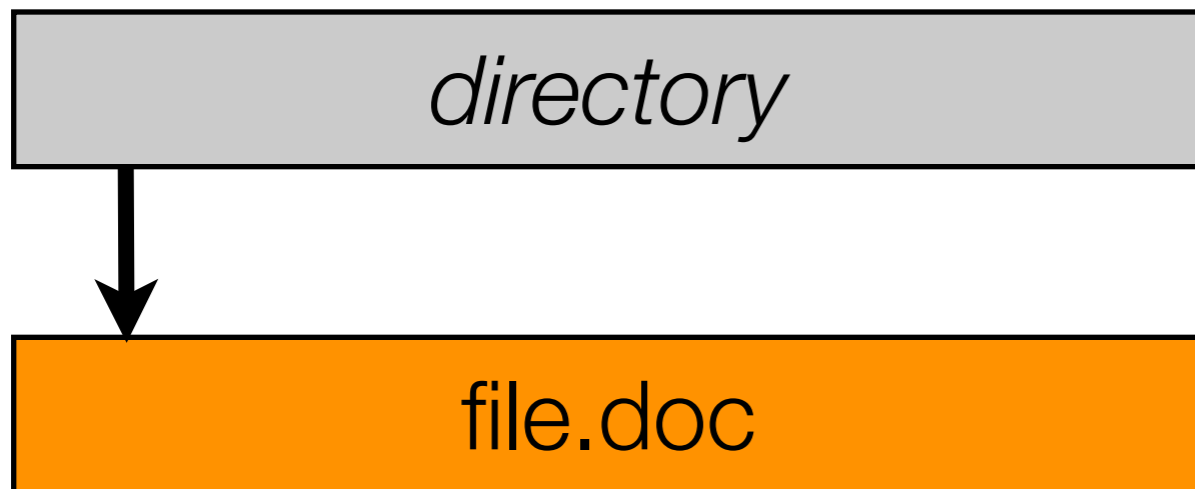
directory

.

..

copy file.doc f:file.doc

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



directory

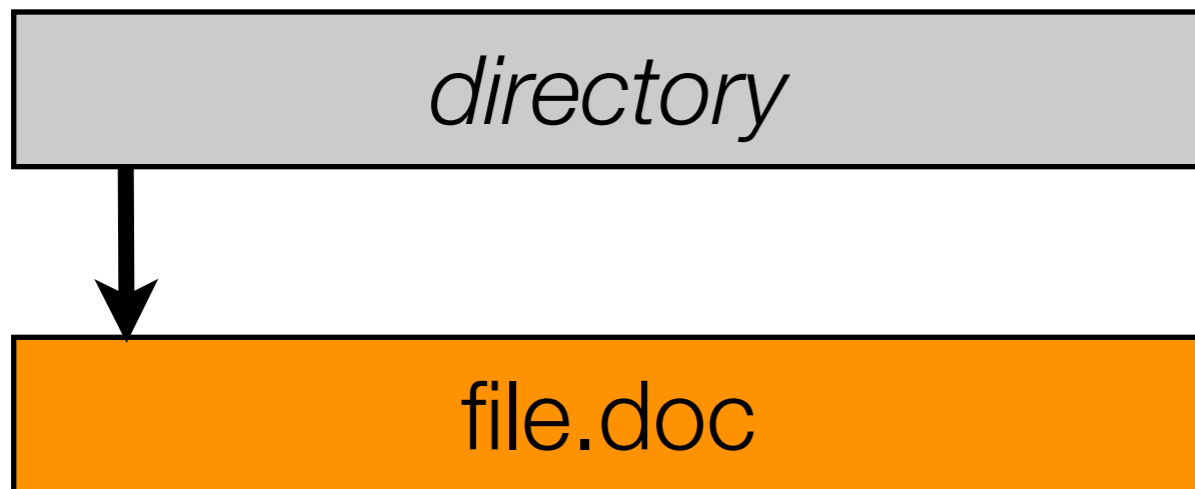
.

..

file.doc

copy file.doc f:file.doc

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



directory

.

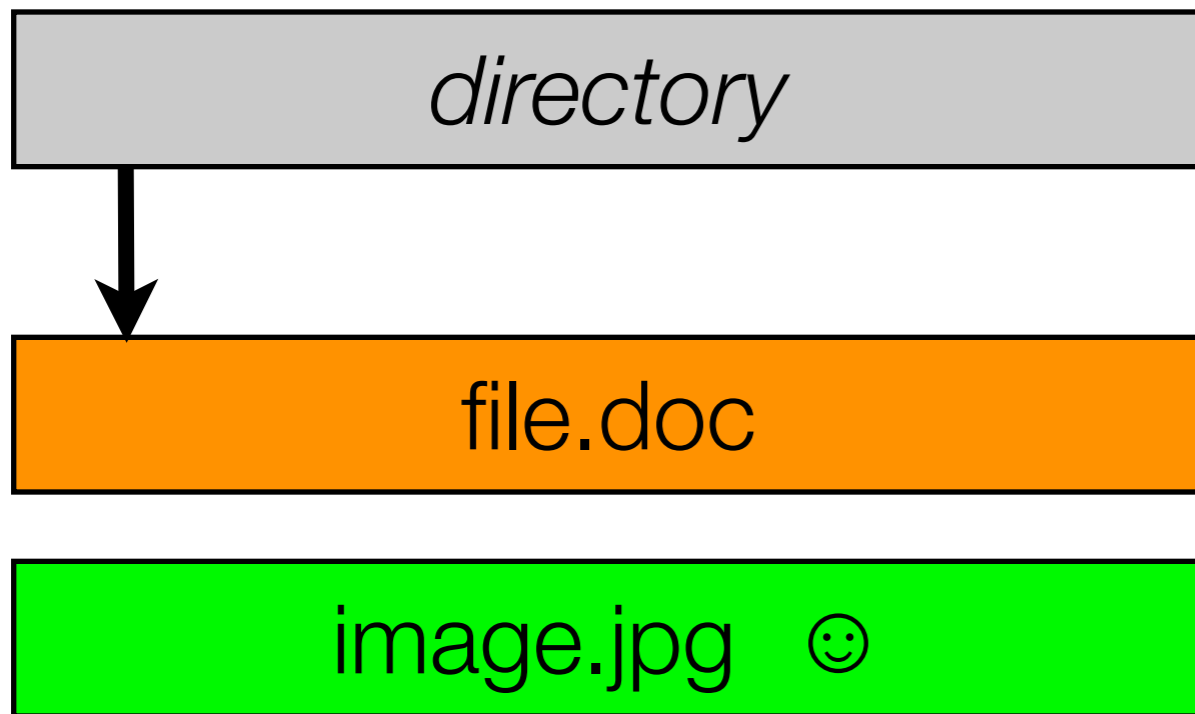
..

file.doc

```
copy file.doc f:file.doc
```

```
copy image.jpg f:image.jpg
```

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



directory

.

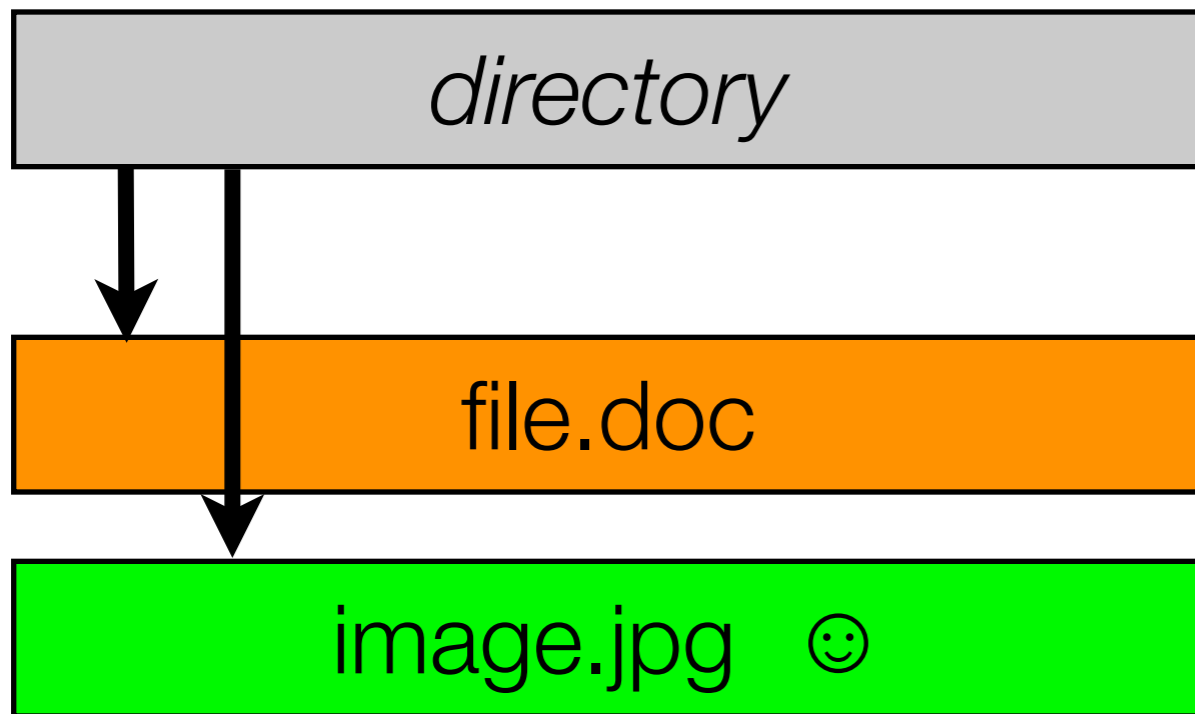
..

file.doc

`copy file.doc f:file.doc`

`copy image.jpg f:image.jpg`

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



directory

.

..

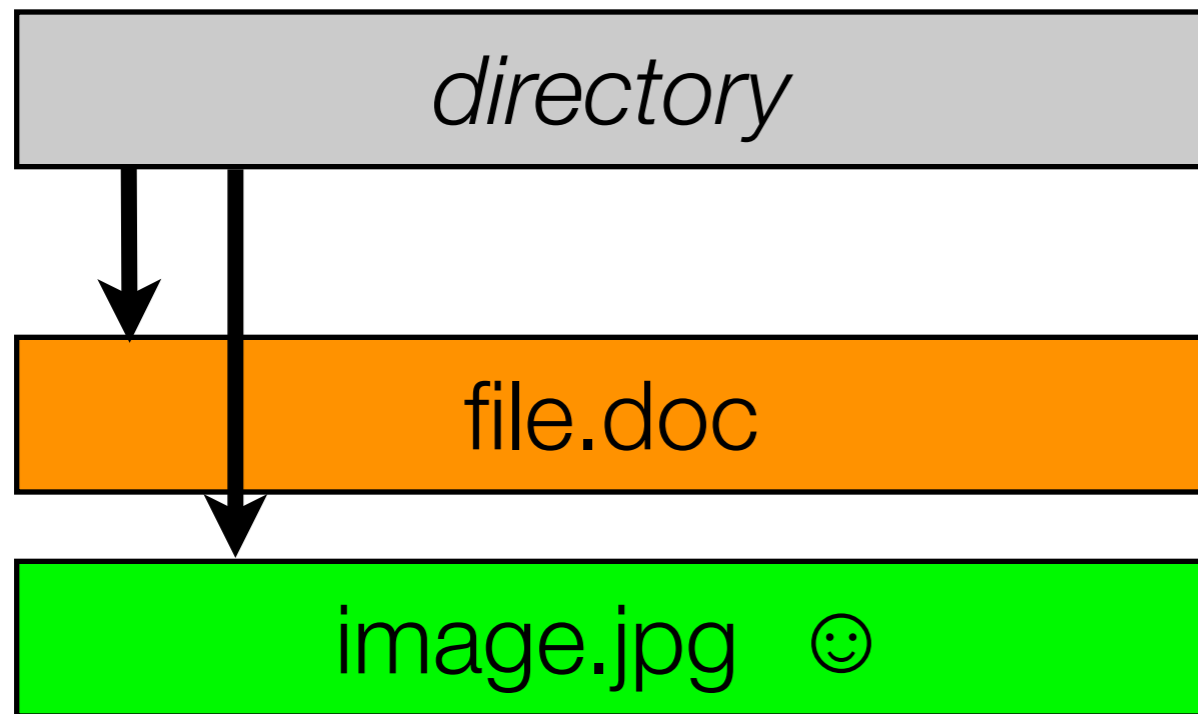
`file.doc`

`image.jpg`

`copy file.doc f:file.doc`

`copy image.jpg f:image.jpg`

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



directory

.

..

file.doc

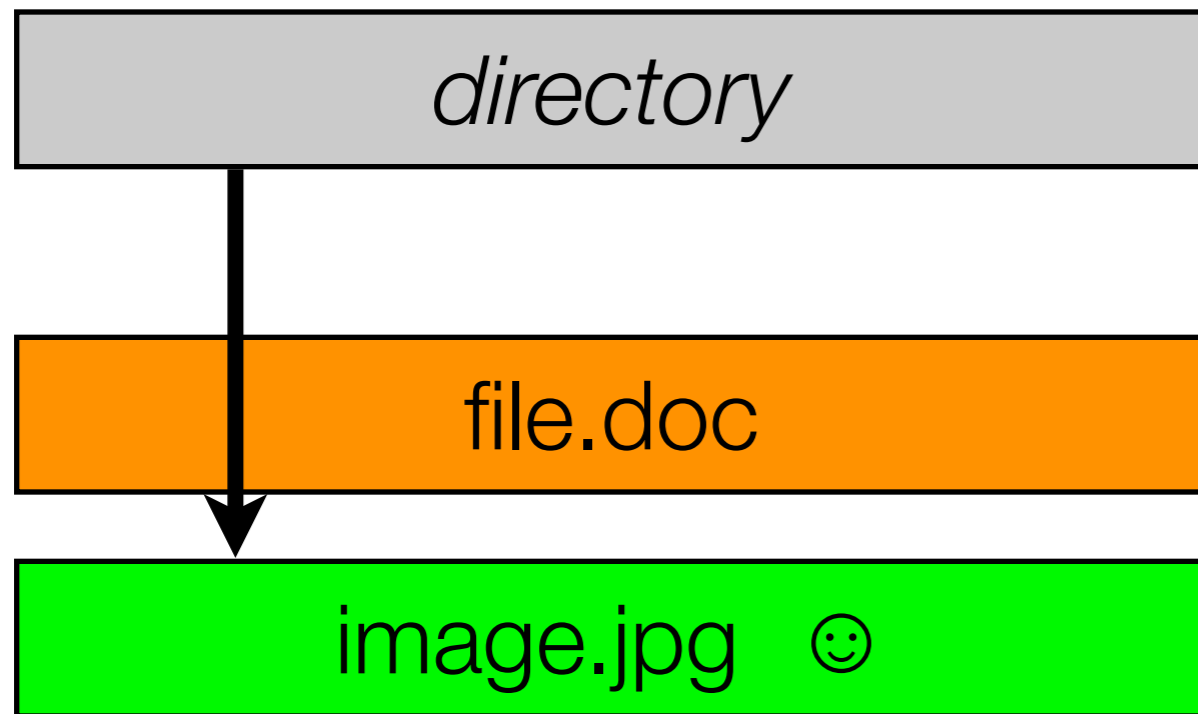
image.jpg

```
copy file.doc f:file.doc
```

```
copy image.jpg f:image.jpg
```

```
del f:file.doc
```

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



```
copy file.doc    f:file.doc
copy image.jpg   f:image.jpg
del f:file.doc
```

directory

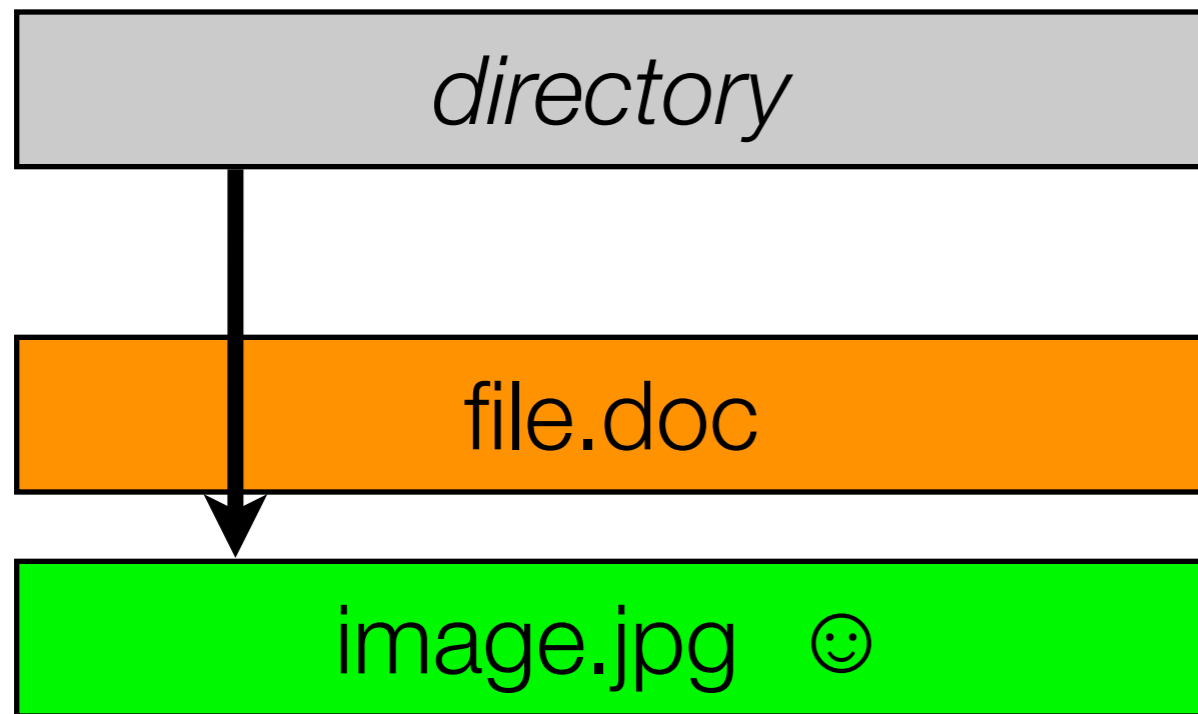
.

..



`image.jpg`

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



```
copy file.doc f:file.doc
copy image.jpg f:image.jpg
del f:file.doc

copy mysong.mp3 f:mysong.mp3
```

directory

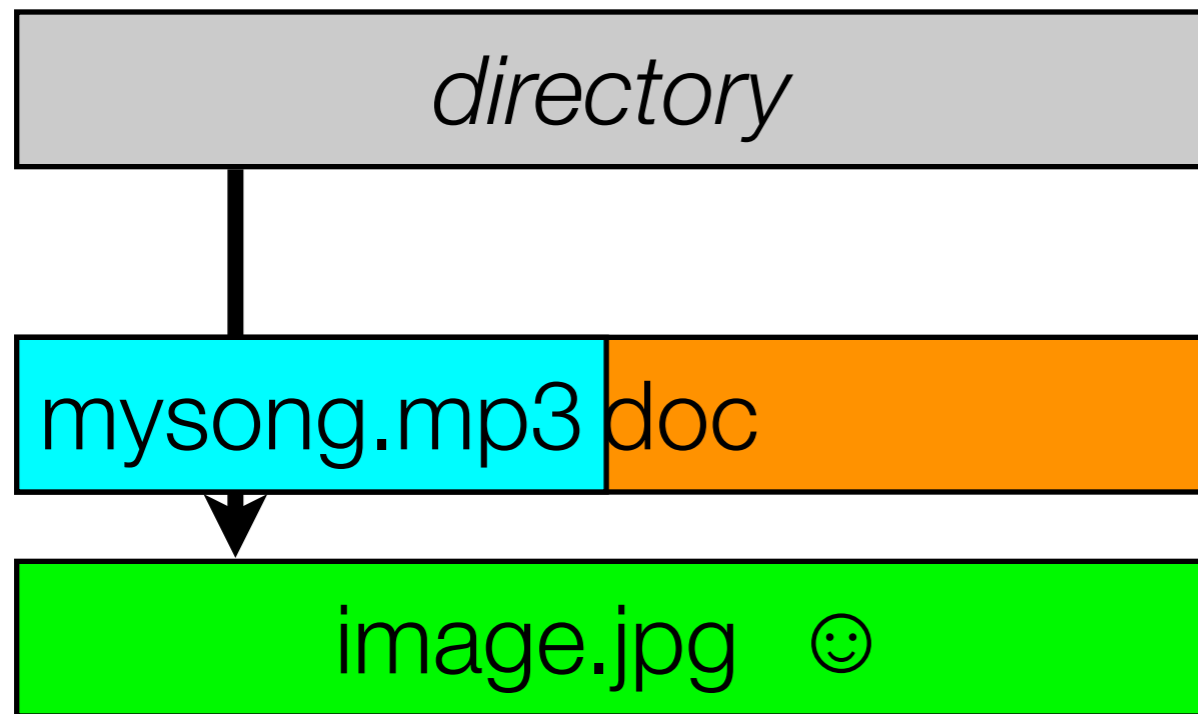
.

..



`image.jpg`

Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



```
copy file.doc f:file.doc
copy image.jpg f:image.jpg
del f:file.doc

copy mysong.mp3 f:mysong.mp3
```

directory

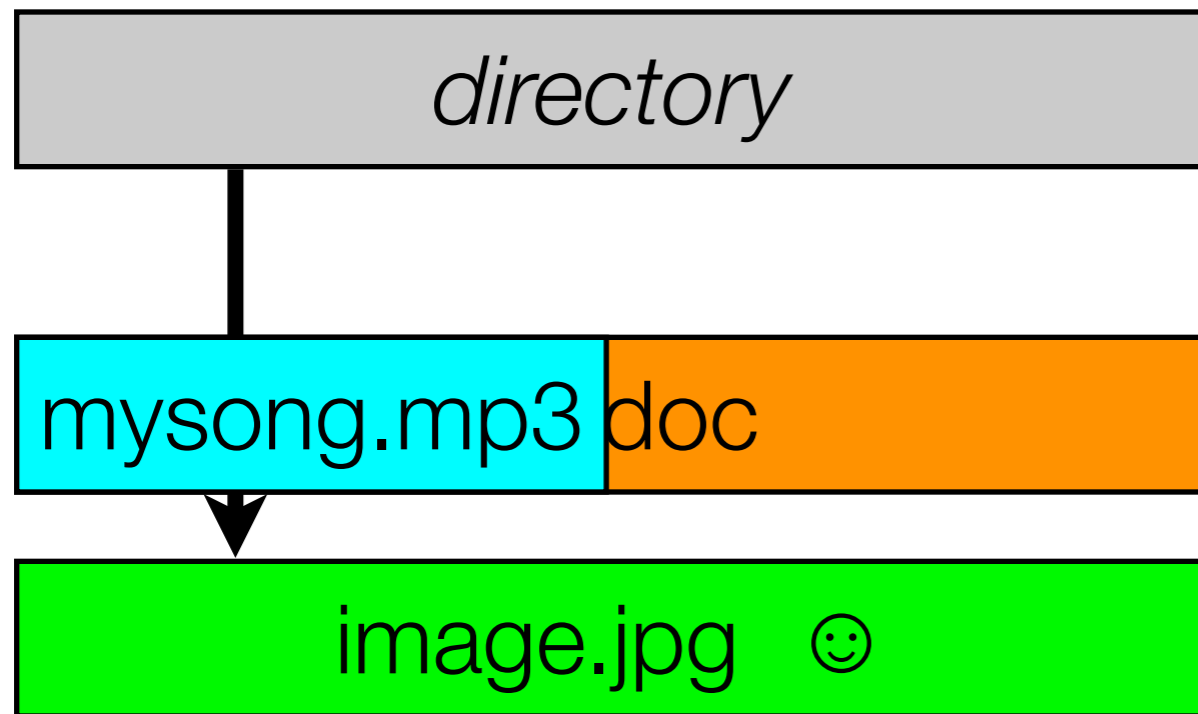
.

..

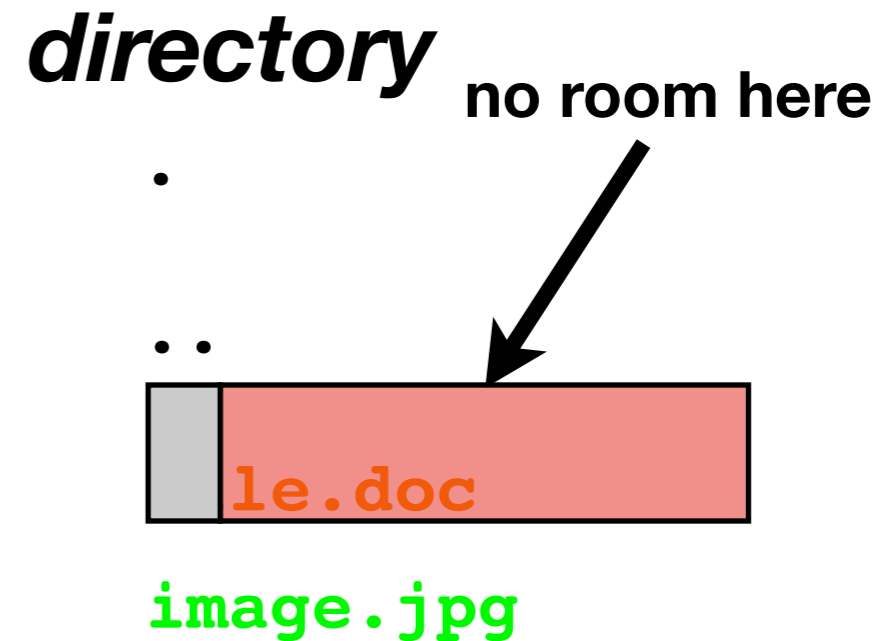


`image.jpg`

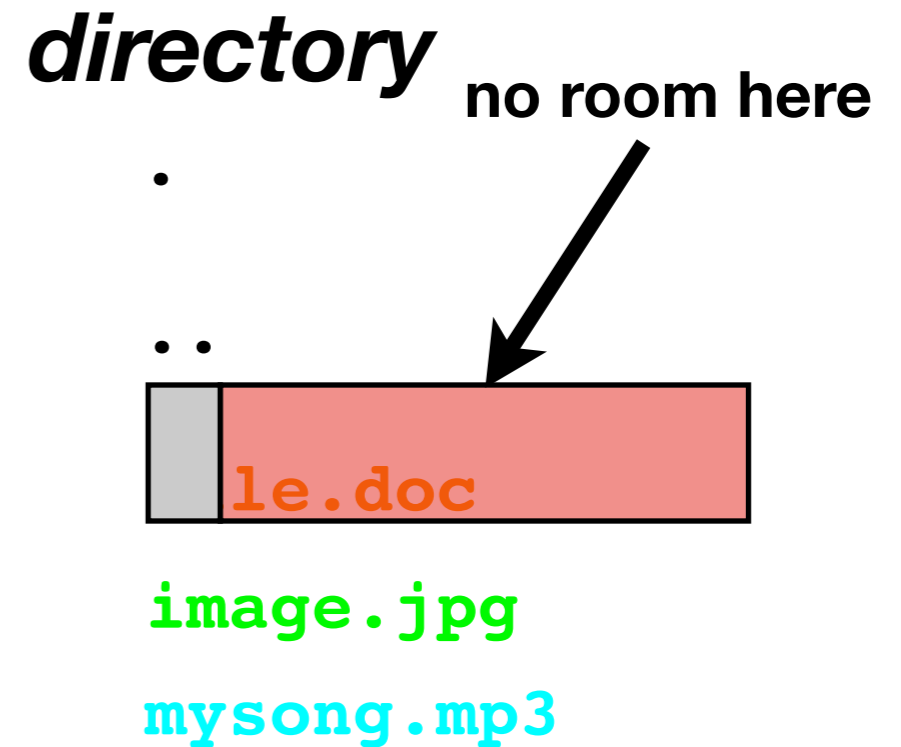
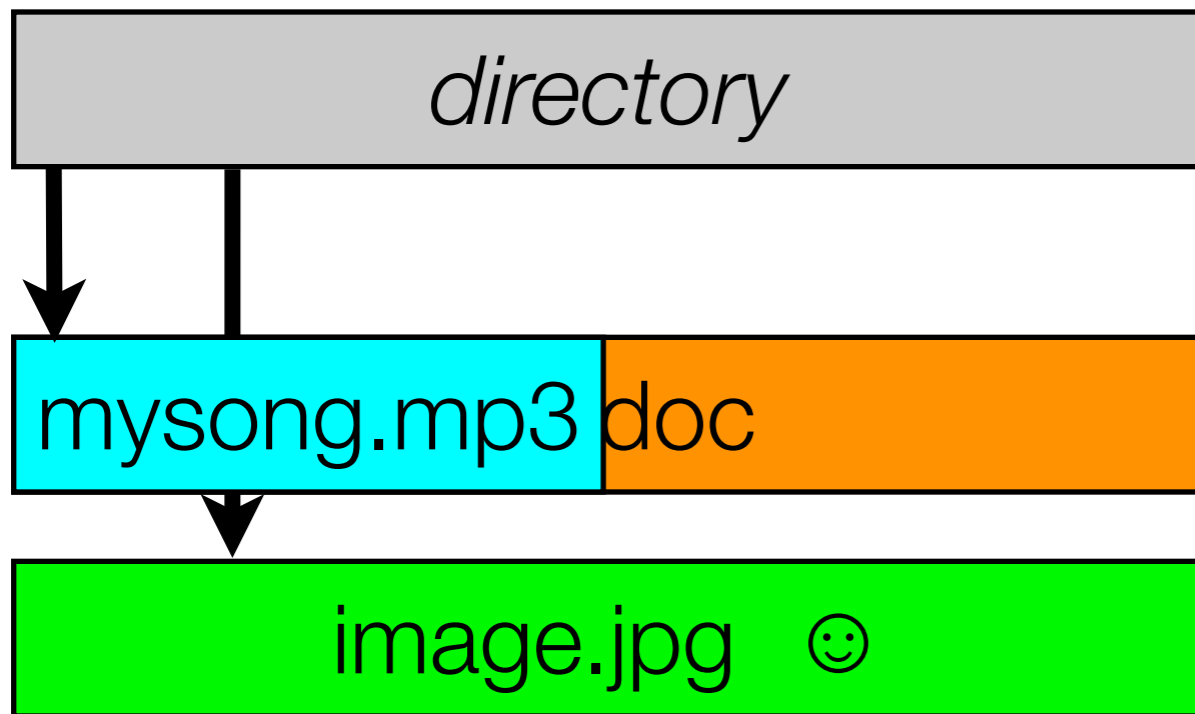
Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



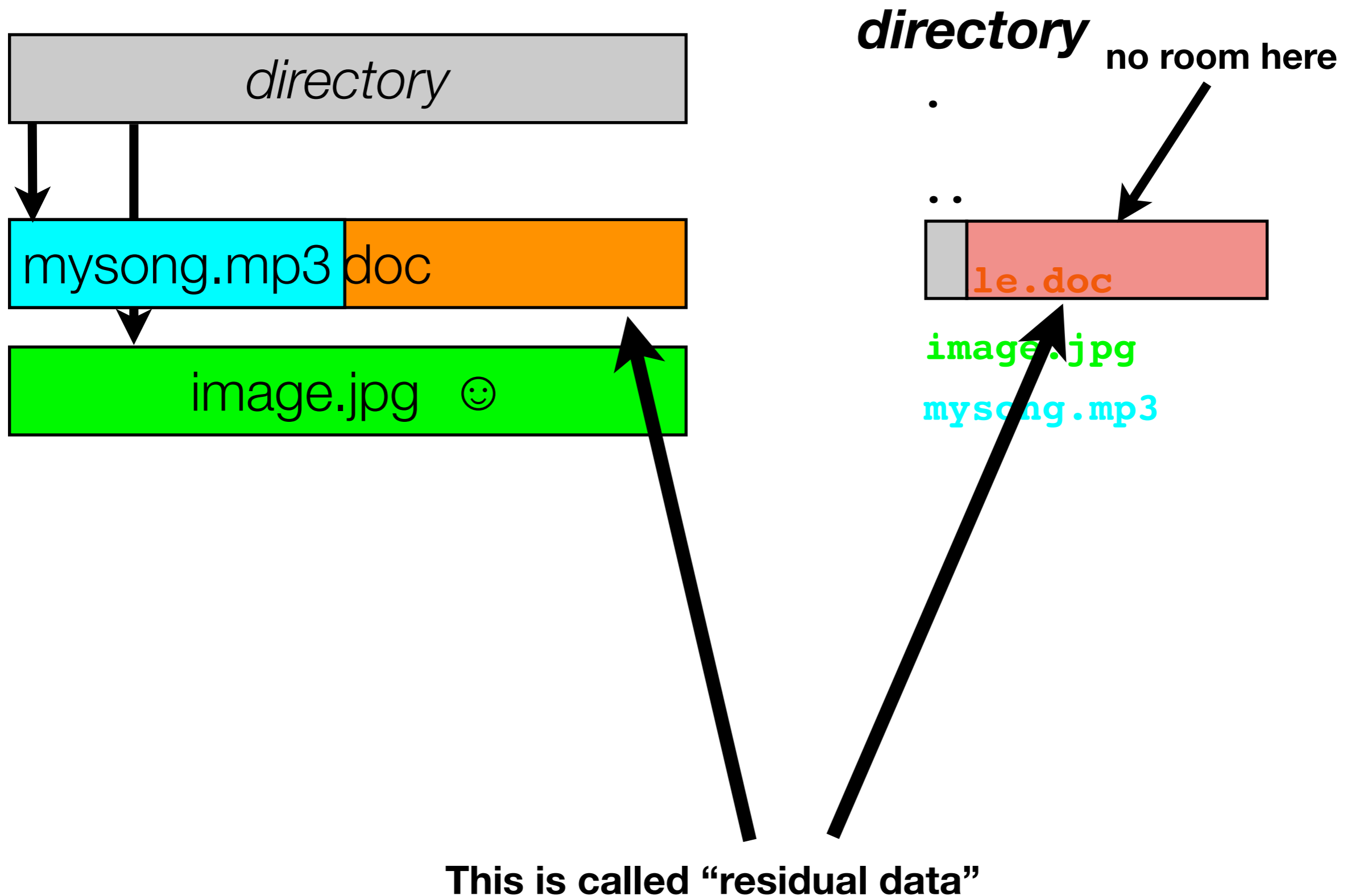
```
copy file.doc      f:file.doc
copy image.jpg     f:image.jpg
del f:file.doc
copy mysong.mp3    f:mysong.mp3
```



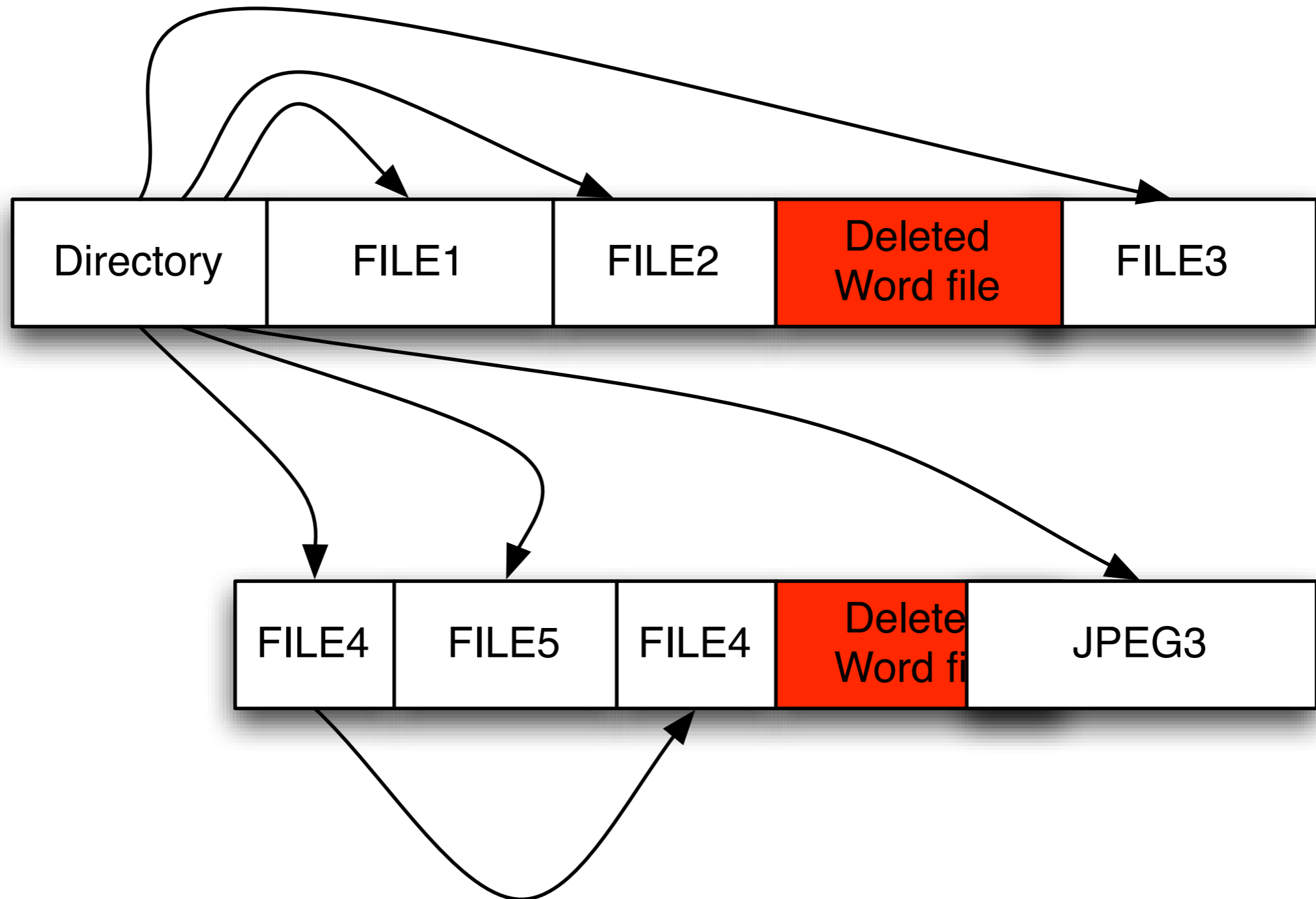
Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



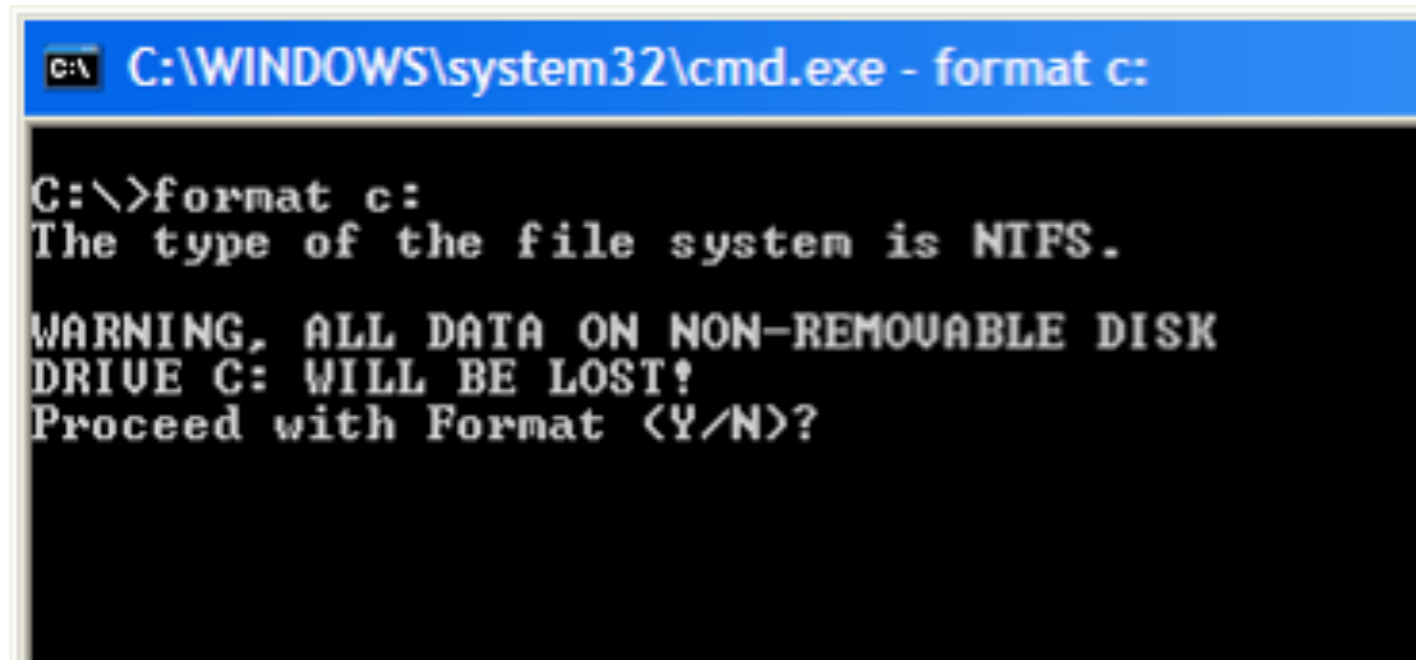
Deleted files can be recovered because “delete” doesn’t really delete, it unlinks.



As a result, a typical disk has many kinds of files and data segments on it:



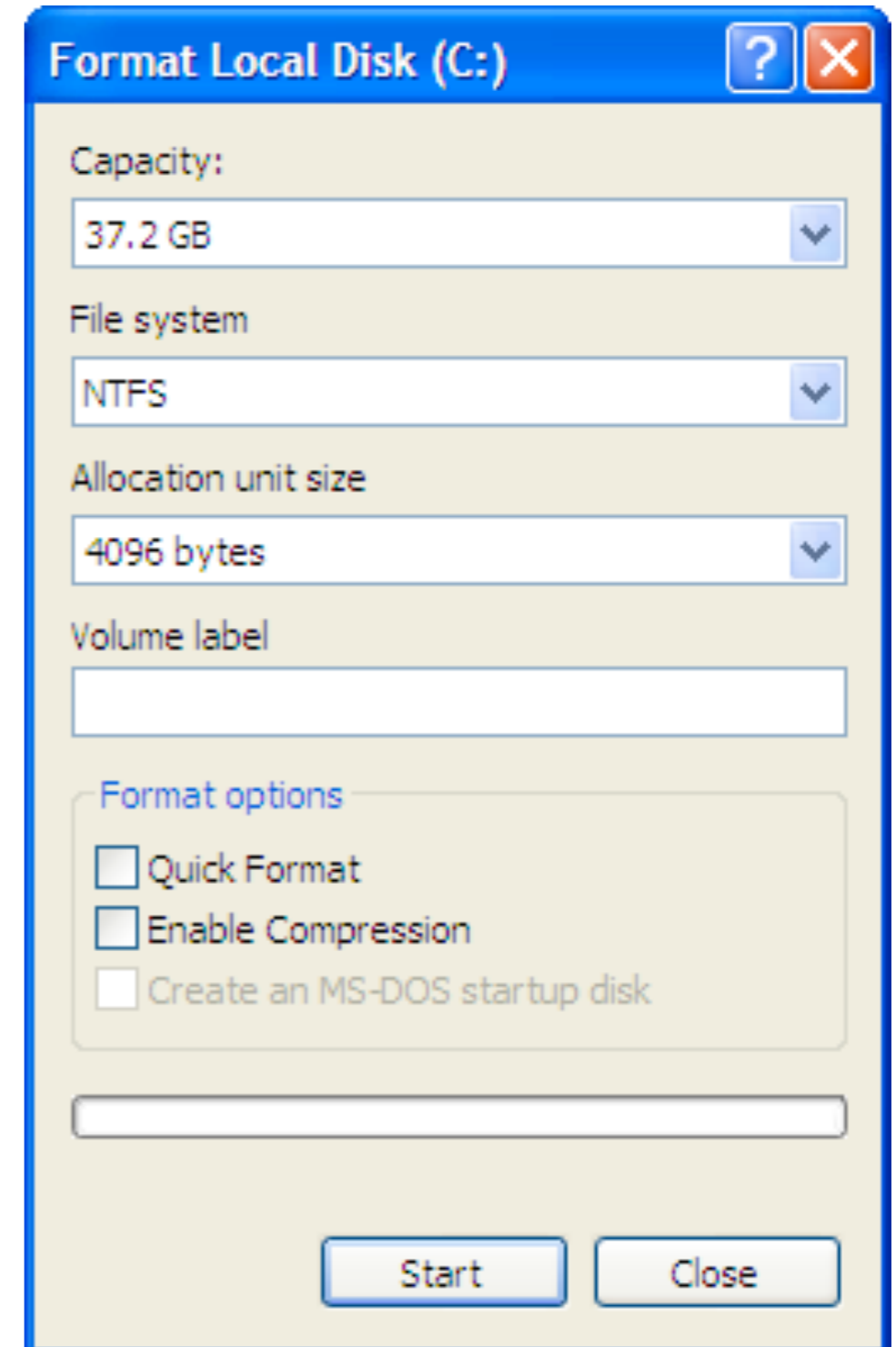
Formatting a disk just writes a new root directory.



```
C:\>format c:
The type of the file system is NTFS.

WARNING, ALL DATA ON NON-REMOVABLE DISK
DRIVE C: WILL BE LOST!
Proceed with Format (Y/N)?
```

A screenshot of a Windows Command Prompt window. The title bar reads 'C:\> C:\WINDOWS\system32\cmd.exe - format c:'. The command prompt shows the command 'format c:' being entered. The output indicates the file system is NTFS and issues a warning that all data on the non-removable disk will be lost, asking for confirmation to proceed with the format.



Format Local Disk (C:)

Capacity: 37.2 GB

File system: NTFS

Allocation unit size: 4096 bytes

Volume label:

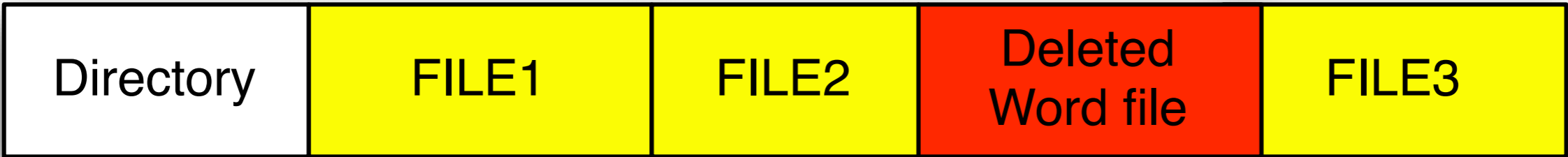
Format options:

- ☐ Quick Format
- ☐ Enable Compression
- ☐ Create an MS-DOS startup disk

Start Close

A screenshot of the 'Format Local Disk (C:)' dialog box. It shows the capacity as 37.2 GB, the file system as NTFS, and the allocation unit size as 4096 bytes. There is a field for the volume label. Under 'Format options', there are three checkboxes: 'Quick Format', 'Enable Compression', and 'Create an MS-DOS startup disk'. At the bottom, there are 'Start' and 'Close' buttons.

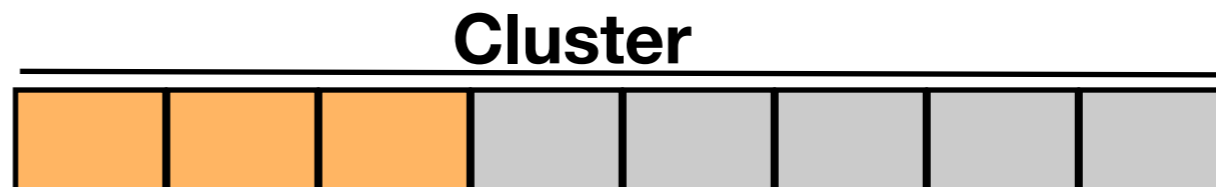
Formatting a disk just writes a new root directory.



There are many places that “deleted” information can hide

Free Space - Sectors on the “free list” (deleted but not overwritten)

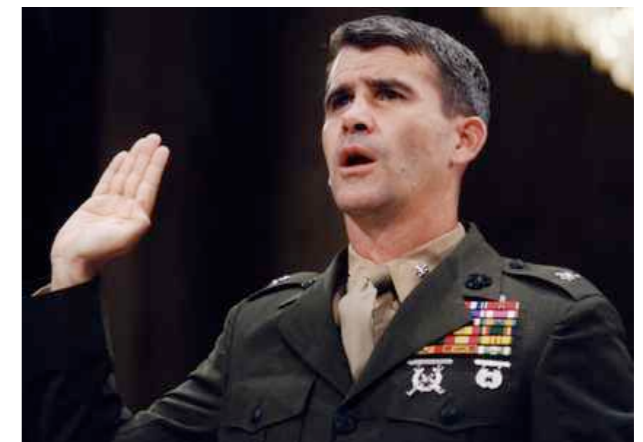
Slack Space - Unused sectors at the end of an allocated cluster



Between partitions

Inside compound document files (MSOffice, etc.)

Backup Tapes



Oliver North

For more information, see:

“One Big File Is Not Enough: A Critical Evaluation of the Dominant Free-Space Sanitization Technique,” Garfinkel & Malan, PET 2006

Let's see what this looks like in practice.

Disk #70: IBM-DALA-3540/81B70E32

Purchased for \$5 from a Mass retail store on eBay

Copied the data off: 541MB

Initial analysis:

- Total disk sectors: 1,057,392
- Total non-zero sectors: 989,514
- Total files: 3

The files:

```
drwxrwxrwx 0 root          0 Dec 31 1979 ./
-r-xr-xr-x 0 root 222390 May 11 1998 IO.SYS
-r-xr-xr-x 0 root          9 May 11 1998 MSDOS.SYS
-rwxrwxrwx 0 root  93880 May 11 1998 COMMAND.COM
```

Image this disk to a file, then use the Unix “strings” command:

```
% strings 70.img | more
```

```
Insert diskette for drive
```

```
and press any key when ready
```

```
Your program caused a divide overflow error.
```

```
If the problem persists, contact your program vendor.
```

```
Windows has disabled direct disk access to protect your lo
```

```
To override this protection, see the LOCK /? command for m
```

```
The system has been halted. Press Ctrl+Alt+Del to restart
```

```
You started your computer with a version of MS-DOS incompatible
```

```
version of Windows. Insert a Startup diskette matching this
```

```
OEMString = "NCR 14 inch Analog Color Display Enhanced SV
```

```
Graphics Mode: 640 x 480 at 72Hz vertical refresh.
```

```
XResolution = 640
```

```
YResolution = 480
```

% strings cont...

ling the Trial Edition

IBM AntiVirus Trial Edition is a full-function but time-limited evaluation version of the IBM AntiVirus Desktop Edition product. It may have been received the Trial Edition on a promotional CD-ROM, a single-file installation program over a network. The Trial Edition is available in seven national languages, and each language is provided on a separate CD-ROM or as a separate

EAS.STCm

EET.STC

ELR.STCq

ELS.STC




% strings 70.img cont...

MAB-DEDUCTIBLE
MAB-MOOP
MAB-MOOP-DED
METHIMAZOLE
INSULIN (HUMAN)
COUMARIN ANTICOAGULANTS
CARBAMATE DERIVATIVES
AMANTADINE
MANNITOL
MAPROTILINE
CARBAMAZEPINE
CHLORPHENESIN CARBAMATE
ETHINAMATE
FORMALDEHYDE
MAFENIDE ACETATE

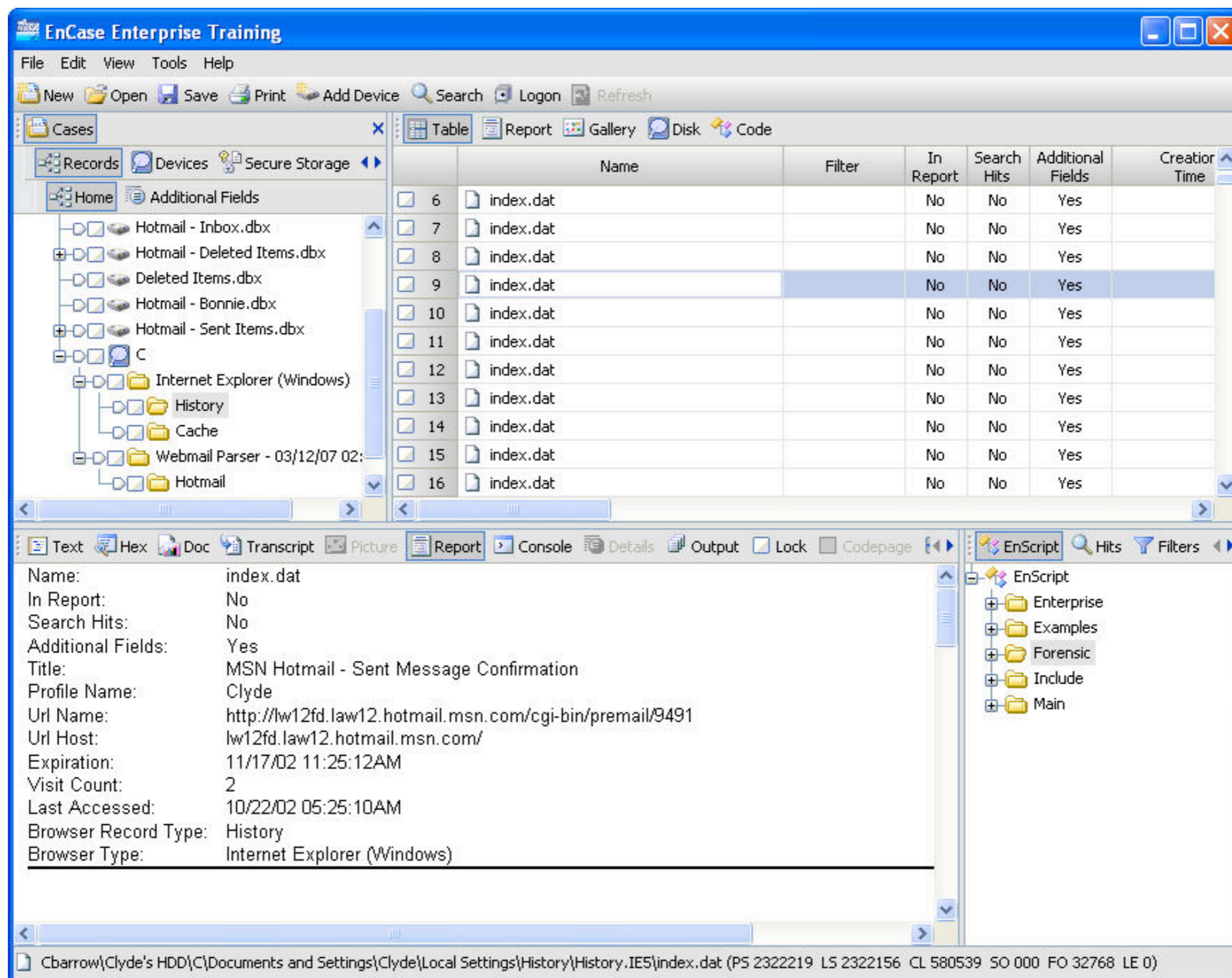
If you want to do more than just look at strings,
you'll need to use a forensic tool.

There are many tools for viewing forensic data.



			
Program	EnCase	FTK	Sleuth Kit
Publisher	Guidance Software	AccessData	Brian Carrier
Cost	≈\$5000	≈\$2500	\$0
Runs On	Windows	Windows	Windows, MacOS, Linux, FreeBSD, OpenBSD, others.
Remote Operation?	Encase Enterprise	no	Yes
Scriptable	EScript	?	bash, perl, python, C/C++, Java, etc.

EnCase Forensic.



AccessData FTK

AccessData FTK 1.61 DEMO VERSION

File Edit View Tools Help

Overview Explore Graphics E-Mail Search Bookmark

Evidence Items: 2 KFF Alert Files: 0 Documents: 16
Total File Items: 378 Bookmarked Items: 0 Spreadsheets: 0
Checked Items: 0 Bad Extension: 4 Databases: 0
Unchecked Items: 378 Encrypted Files: 0 Graphics: 0
Flagged Thumbnails: 0 From E-mail: 0 E-mail Messages: 0
Other Thumbnails: 0 Deleted Files: 223 Executables: 0
Filtered In: 378 From Recycle Bin: 0 Archives: 0
Filtered Out: 0 Duplicate Items: 16 Folders: 29
Unfiltered Filtered OLE Subitems: 0 Slack/Free Space: 131
All Items Actual Files Flagged Ignore: 0 Other Known Type: 1
Data Carved Files: 0 KFF Ignorable: 0 Unknown Type: 201

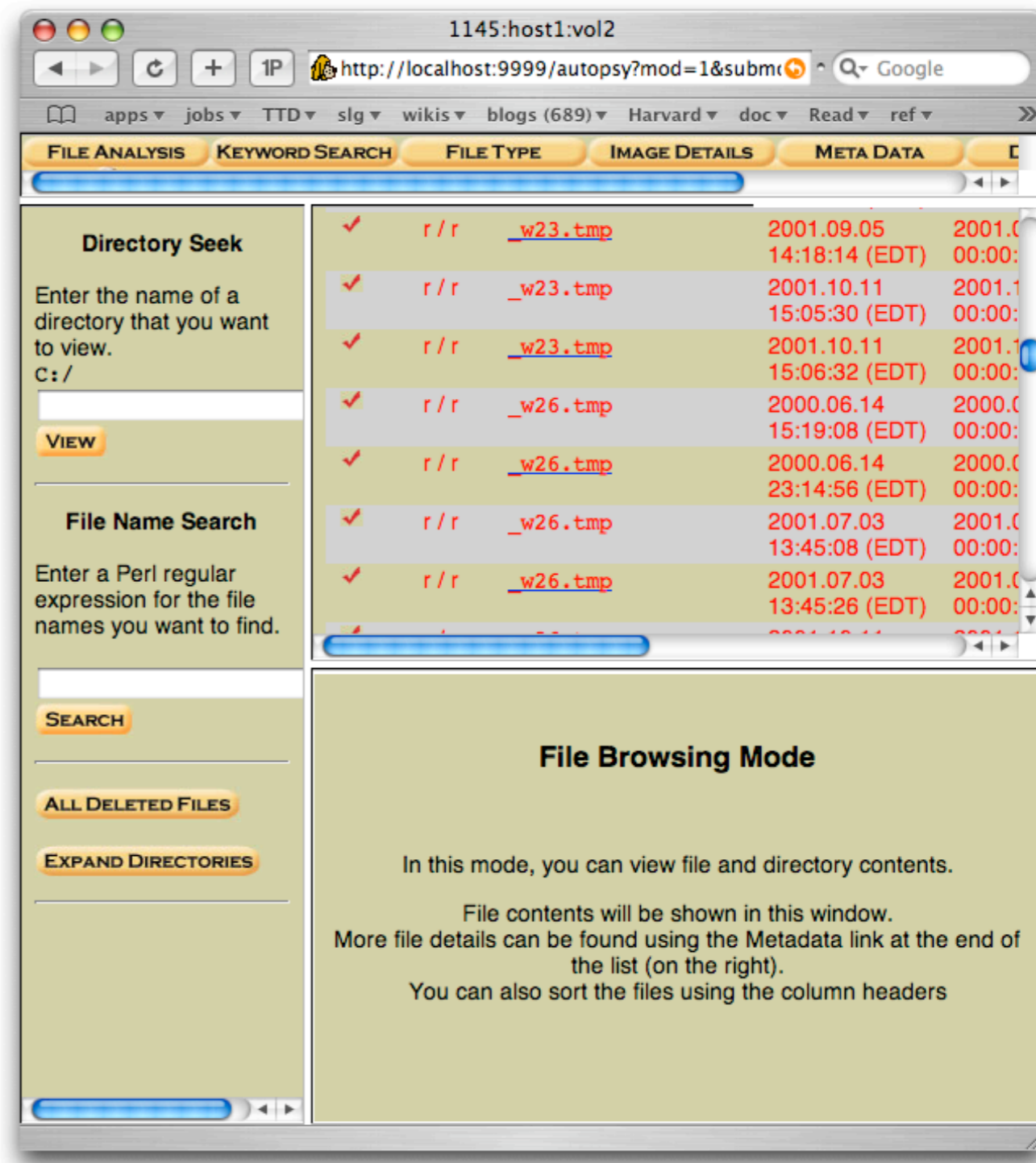
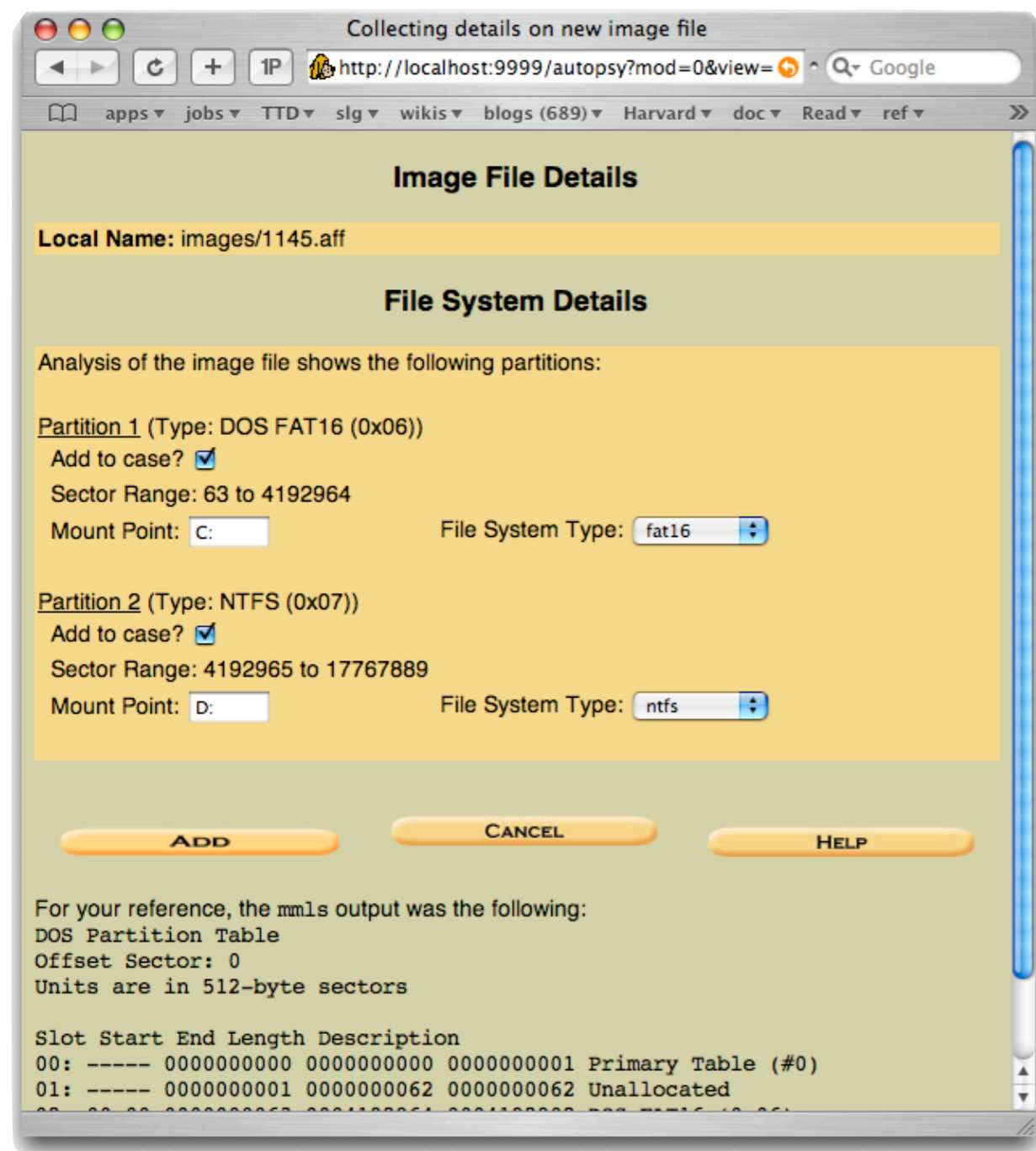
C:\Documents and Settings\natasa\My Documents\VB\Data\WorldWideWeb.htm:WorldWideWeb.htm[1] C:\Documents and Settings\natasa\My Documents\VB\Data\WWWafter10 years.htm:WWWafter10years.htm[2] C:\Documents and Settings\natasa\My Documents\VB\Data\CyberpublishingManifesto.htm:CyberpublishingManifesto.htm [3] C:\Documents and Settings\natasa\My Documents\VB\Data\TheClientSideoftheWeb.htm:ClientSideoftheWeb.htm[7]

Unfiltered All Columns DTZ

File Name	Full Path	Recycl...	E..	File Type	Category
<input checked="" type="checkbox"/> TheClientSideoftheWeb.htm	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\ads\Data\ThreadContentFiles.html\TheClientSide...		htm	Unknown Fil...	Unknown
<input checked="" type="checkbox"/> ThreadContentFiles.html	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\archive\ads\Data\ThreadContentFiles.html		html	Unknown Fil...	Unknown
<input checked="" type="checkbox"/> ThreadContentFiles.html	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\ads\Data\ThreadContentFiles.html		html	Unknown Fil...	Unknown
<input checked="" type="checkbox"/> Threditor.vbw	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\Threditor\SourceCodeFile\Threditor.vbw		vbw	Unknown Fil...	Unknown
<input checked="" type="checkbox"/> Threditor.vbw	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\archive\Threditor\SourceCodeFile\Threditor.vbw		vbw	Unknown Fil...	Unknown
<input checked="" type="checkbox"/> ThreadList.txt	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\Data\ThreadList.txt		txt	Hypertext Do...	Document
<input checked="" type="checkbox"/> ThreadList.txt	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\Data\ThreadList.txt\ThreadList.txt		txt	Plain Text D...	Document
<input checked="" type="checkbox"/> ThreadList.txt	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\archive\Data\ThreadList.txt		txt	Hypertext Do...	Document
<input checked="" type="checkbox"/> ThreadList.txt	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\archive\Data\ThreadList.txt\ThreadList.txt		txt	Plain Text D...	Document
<input checked="" type="checkbox"/> ThreadList.txt	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\archive\ads\Data\ThreadList.txt		txt	Hypertext Do...	Document
<input checked="" type="checkbox"/> ThreadList.txt	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\archive\ads\Data\ThreadList.txt\ThreadList.txt		txt	Plain Text D...	Document
<input checked="" type="checkbox"/> ThreadList.txt	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\ads\Data\ThreadList.txt		txt	Hypertext Do...	Document
<input checked="" type="checkbox"/> ThreadList.txt	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\ads\Data\ThreadList.txt\ThreadList.txt		txt	Plain Text D...	Document
<input checked="" type="checkbox"/> Threditor	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\Threditor			Folder	Folder
<input checked="" type="checkbox"/> Threditor	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\archive\Threditor			Folder	Folder
<input checked="" type="checkbox"/> Threditor.chm	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\Threditor\Executable\Threditor.chm		chm	Unknown Fil...	Unknown
<input checked="" type="checkbox"/> Threditor.chm	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\archive\Threditor\Executable\Threditor.chm		chm	Unknown Fil...	Unknown
<input checked="" type="checkbox"/> Threditor.exe	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\Threditor\Executable\Threditor.exe		exe	Unknown Fil...	Unknown
<input checked="" type="checkbox"/> Threditor.exe	Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\archive\Threditor\Executable\Threditor.exe		exe	Unknown Fil...	Unknown

378 Listed 0 Checked Total Stick2.wiped\Part_1\NNAME-NTFS\wantADS_threditor\archive\ads\Data\ThreadList.txt

SleuthKit with Autopsy



```
Terminal — ssh — 80x24
IMAGING Thu Nov 10 10:53:27 2005
Source device: /dev/ad2 AFF Output: /project/junk.aff
Model #: QUANTUM FIREBALL ST3.2A
firmware: A0F.0000 Sector size: 512 bytes
S/N: 153718340531 Total sectors: 6,306,048

[=====]

Currently reading sector: 97,792 (512 sectors at once)
Sectors read: 98,304 ( 1.56%) # blank: 1,026

Time spent reading: 00:00:05 Estimated total time left: 00:21:34
Total bytes read: 50,331,648

Compressed bytes written: 25,735,396
Time spent compressing: 00:00:09
Overall compression ratio: 48.87% (0% is none; 100% is perfect)
Free space on 192.168.1.1:/project: 68,937 MB (12.44%)
```

Disk Imaging

You can analyze a live system, a disk, or a disk image

Live system:

- You run the program on the system you are analyzing!
- It's changing as you analyze it.
- Not forensically sound, but it may be your only choice.

Disk:

- You can analyze the disk without imaging it.
- Use a write blocker.
- Good for a quick view.
- Bad with fragile disks.



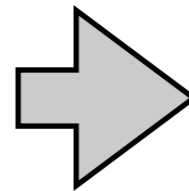
Disk Image:

- Use an approved imaging tool (dcfldd; LinEn; etc.)
- Use a write blocker if possible.

Disk Images store sector-for-sector "images" of a hard disk.

The disk image stores all of the data (ideally)

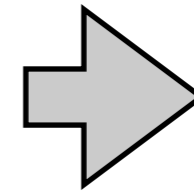
- All of the sectors from the subject drive.
- All of the files
- All of the deleted files
- All of the residual data.



```
Terminal - ssh - 80x24
Thu Nov 10 10:53:27 2005
Source device: /dev/sd2
Model #: QUANTUM FIREBALL ST3.2A
Firmware: AAF-0000
S/N: 353720H0531
AFF Output: /project/junk.coff
Sector size: 512 bytes
Total sectors: 16,384

Currently reading sector: 97,792 (512 sectors at once)
Sectors read: 98,304 ( 1.56%) # blank: 1,026
Time spent reading: 00:00:05 Estimated total time left: 00:21:34
Total bytes read: 50,331,648

Compressed bytes written: 25,735,396
Time spent compressing: 00:00:09
Overall compression ratio: 48.87% (0% is none; 100% is perfect)
Free space on 192.168.1.1/project: 68,937 MB (12.44%)
```



The image may also store metadata:

- Subject drive's serial number.
- Examiner's name.
- Time of imaging
- Checksums
- Hash of the image
- Digital Signatures.

Option #1 for imaging: Use an Imaging Workstation

Remove the drive/media from the subject computer.

Attach to an imaging workstation with
a write-blocker

Use a forensic disk imager



Acquisition Hardware

Acquisition Tools:

- Write-Blockers prevent modification
- Network agents allow capture over a network
- Information stored in an “image file” or on a “mirror disk.”



Acquisition Software: Integrity is paramount

Imaging options:

- `dd if=/dev/hda of=diskfile.img conv=sync,noerror bs=65536`
- `aimage /dev/hda diskfile.img`
- LinEn (Linux EnCase imager)
- FTK Imager
- Hardware Tools

Most tools will:

- Copy the raw device to a file
- Compute MD5 & SHA1
- Properly handle bad blocks

Some tools will:

- Compress image
- Capture metadata (Drive s/n)
- Record investigative notes
- Encrypt, Digitally sign (AFF Only)



Option #2: Use a LiveCD

Bootable CDRoms combine Linux + Forensic tools

- Free:
 - ✓ Lnx 4n6 - <http://www.lnx4n6.be/>
 - ✓ <http://www.caine-live.net/>
- Commercial:
 - ✓ The Farmer's Boot CD - <http://www.forensicbootcd.com/>
 - ✓ Helix - <http://www.e-fense.com/helix>

Advantages: No need to acquire hard drive

Dangers:

- Not all Linux distributions are forensically sound! Be careful!
- Some Linux distributions will swap on the hard drive
- Many Linux distributions are not up-to-date and lack important drivers.

Lists:

- http://www.forensicswiki.org/wiki/Category:Live_CD
- <http://www.livecdlist.com/purpose/forensics>

Helix runs as both a Windows Live Analysis and as a Live CD



Helix Live CD under Windows

<http://www.e-fense.com/>

HELIX™ INCIDENT RESPONSE • ELECTRONIC DISCOVERY • COMPUTER FORENSICS



Preview system information.



Acquire a "live" image of a Windows System using dd.



Incident Response Tools for Windows Systems.



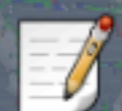
Documents pertaining to Incident Response, Computer Forensics, Computer Security & Computer Crime.



Browse contents of the CD-ROM and Host OS.



Scan for Pictures from a live system.



Investigative Notes.

System Information



Operating System:

Windows XP Service Pack 2

Owner Information:

Owner: Simson Garfinkel
 Organization:
 Admin: No
 Admin Rights: Yes

Network Information:

Host: DELL
 User: simsong
 IP: 192.168.1.104
 NIC: 000d5608e2af
 Domain: WORKGROUP

Drive:	Label:	Type:	Size:
C:\	(Logical drive)	NTFS	76285.2 MB
D:\	(Logical drive)	CDFS	698.3 MB
E:\	(CD/DVD-ROM drive)		
G:\	(CD/DVD-ROM drive)		
H:\	(CD/DVD-ROM drive)		
I:\	(CD/DVD-ROM drive)		
J:\	(CD/DVD-ROM drive)		
V:\	(Logical drive)	NTFS	238916 MB

Live Acquisition

Acquire Physical Memory and/or Disk Drives



Source:

\\.\PhysicalMemory - [1032 MB]

Location Options:



Attached/Share



NetCat

Destination:



\\forensics\images\

Image Name:

image.dd

DD Options:

block size:

default

conv:

noerror



Split Image

Acquire

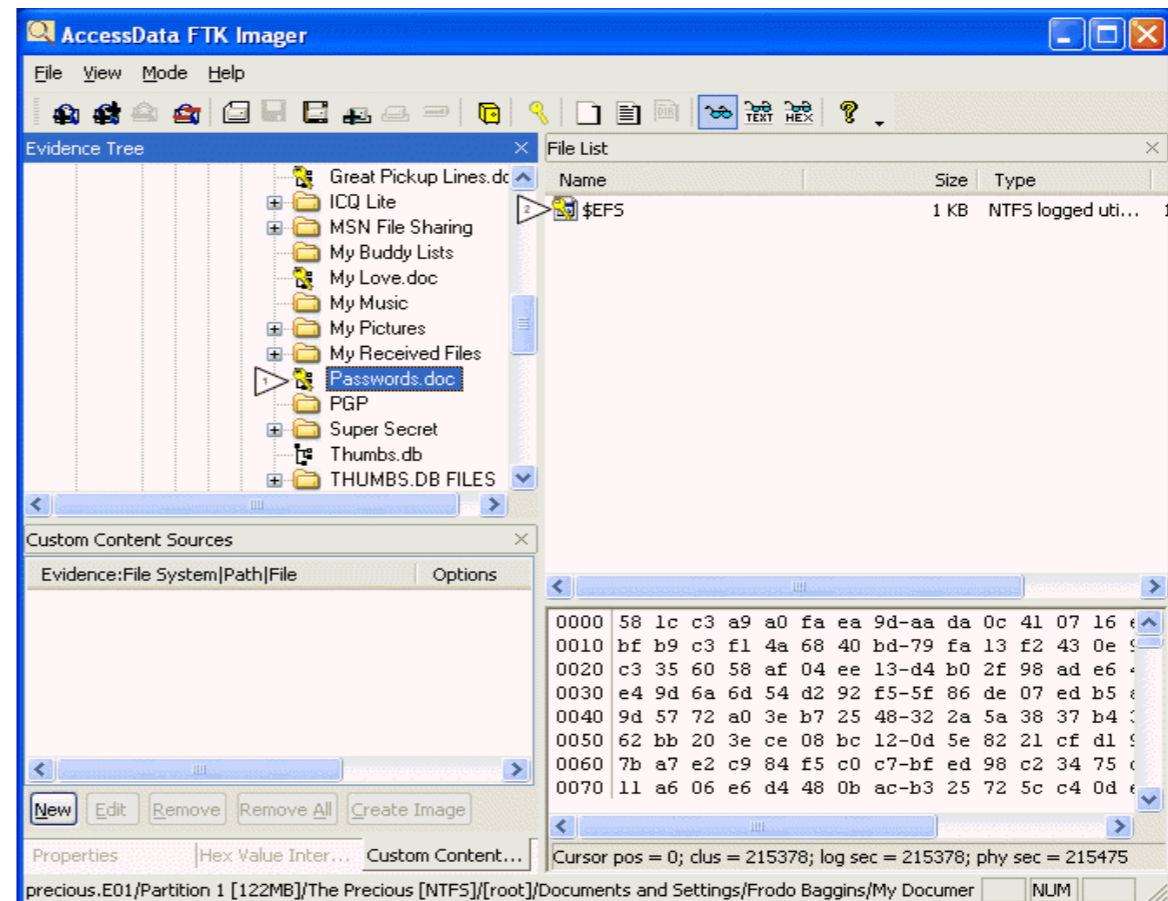
FTK Imager from Access Data

Data preview and imaging tool

- Runs on Windows
- Two versions: Installed and uninstalled
- Produces RAW and E01 files
- GUI or command-line
- Free download from <http://www.accessdata.com/downloads.html>

Limited forensic analysis:

- Reports known files
- Detects EFS encryption



Option #3: "Hardware" disk duplication

Really a tiny computer...

Typically these make sector-for-sector copies



<http://www.deepspare.com/products-ds-disk-imager.html>



<http://www.ics-iq.com/>

Option #4 Get an image from somebody else.

Advantages:

- Somebody else does the work.

Disadvantages:

- Chain of custody may be questionable

Disk images can be *physical* or *logical*.

Physical Evidence File (.E01, .aff, .raw)

- Raw disk image
- Optionally from multiple devices
- Can be used with many systems

Logical Evidence File (.L01)

- EnCase specific file.
- Contains information copied out of the disk image.

Always get a physical disk image.

Always image the raw device

- ***You may also image the partition if it is encrypted***

Disk images can be in many formats.

RAW & Split Raw — All of the sectors.

gzip'ed raw

Seekable GZIP (sgzip)

AFFLIB

EnCase

Raw disk images store the data; nothing else.

Raw: *disk.raw, disk.dd, disk.iso*

- Advantages: fast access, supported by most programs
- Disadvantages: big, lots of wasted space; no encryption; no checksums
- gzip'ed or zip'ed raw is a common distribution for research & challenges.

disk.raw (6GB)

Split raw: *disk.000, disk.001, disk.002*

- Files >4GB not supported by FAT32, ext2, and other file systems
- It's hard to burn big files.
- Disadvantage: easy to lose a piece!

disk.000

disk.001

disk.002

AFF is the Advanced Forensic Format

<http://www AFFlib.org/>

Advantages:

- Better compression than EnCase
- AES-256 encryption (passphrase or PKI) for true data security
- X.509 digital signatures for true chain-of-custody
- Open source implementation (BSD license)

disk.aff (1GB)

Disadvantages:

- No direct support in EnCase or FTK
(but disk can be "mounted" in host OS.)

disk.afd/file001.aff

disk.afd/file002.aff

AFF4:

- Upwards compatible version of AFF.
- Uses ZIP64 to store segments.

disk.raw

disk.afm

EnCase Expert Witness Format (EWF): Compressed data; some metadata

EnCase: disk.E01, disk.E02

- Supported by most forensic software (FTK, TSK, AFF);
- CRC32 and MD5
- Limited investigator notes.

Disadvantage:

- File size limit forces splitting;
- No support for encryption

disk.E01	disk.E02	disk.E03
----------	----------	----------

The EnCase "password" does not encrypt the disk image!

libewf: an open source EnCase implementation

Marketed as an “Expert Witness Format” implementation.

- “Expert Witness” was the original name of EnCase.

Developed by Joachim Metz & Robert-Jan Mora of Hoffman Investigations, NL

Included commands:

- ewfacquire — disk imager
- ewfacquirestream — Image a stream to an E01 file
- ewfalter — Change an E01 file’s permissions
- ewfexport — Turns an E01 file into a raw or another kind of file.
- ewfinfo — Information about an E01 file
- ewfverify — Verifies the CRC32 and MD5

ewfacquire — Turn a raw file into an .E01

```
$ ewfacquire spicel.raw
```

```
ewfacquire 20080820 (libewf 20080820, zlib 1.2.3, libcrypto 0.9.7)
```

```
Acquiry parameters required, please provide the necessary input
```

```
Image path and filename without extension: spicel
```

```
Case number: SLG-0001
```

```
Description: My Special Case
```

```
Evidence number: SLG-0001-E001
```

```
Examiner name: Simson Garfinkel
```

```
Notes: Test image.
```

```
Media type (fixed, removable) [fixed]:
```

```
Volume type (logical, physical) [physical]:
```

```
Use compression (none, fast, best) [none]: best
```

```
Use EWF file format (ewf, smart, ftk, encase1, encase2, encase3, encase4,  
encase5, encase6, linen5, linen6, ewfx) [encase5]:
```

```
Start to acquire at offset (0 >= value >= 32079872) [0]:
```

```
Amount of bytes to acquire (0 >= value >= 32079872) [32079872]:
```

```
Evidence segment file size in bytes (1.0 MiB >= value >= 1.9 GiB) [1.4 GiB]:
```

```
The amount of sectors to read at once (64, 128, 256, 512, 1024, 2048, 4096,  
8192, 16384, 32768) [64]:
```

```
The amount of sectors to be used as error granularity (1 >= value >= 64) 64]:
```

```
The amount of retries when a read error occurs (0 >= value >= 255) [2]:
```

```
Wipe sectors on read error (mimic EnCase like behavior) (yes, no) [no]:
```

ewfacquire verifies parameters before it starts.

The following acquiry parameters were provided:

Image path and filename: spice1.E01
Case number: SLG-0001
Description: My Special Case
Evidence number: SLG-0001-E001
Examiner name: Simson Garfinkel
Notes: Test image.
Media type: fixed
Volume type: physical
Compression used: best
EWF file format: EnCase 5
Acquiry start offset: 0
Amount of bytes to acquire: 30 MiB (32079872 bytes)
Evidence segment file size: 1.4 GiB (1572864000 bytes)
Block size: 64 sectors
Error granularity: 64 sectors
Retries on read error: 2
Wipe sectors on read error: no

Continue acquiry with these values (yes, no) [yes]: **yes**

ewfacquire calculates the CRC32s and MD5

```
Acquiry started at: Mon Nov  3 21:05:42 2008
```

```
This could take a while.
```

```
Status: at 87%.
```

```
    acquired 26 MiB (27983872 bytes) of total 30 MiB (32079872 bytes).  
    completion in 0 second(s) with 30 MiB/s (32079872 bytes/second).
```

```
Acquiry completed at: Mon Nov  3 21:05:43 2008Mon Nov  3 21:05:43 2008
```

```
Written: 30 MiB (32081188 bytes) in 1 second(s) with 30 MiB/s (32113956  
bytes/second).
```

```
MD5 hash calculated over data:      aebfd76cdd9b3eb0f6c1658efc226886
```

You can verify the MD5:

```
$ md5 spice1.raw
```

```
MD5 (spice1.iso) = aebfd76cdd9b3eb0f6c1658efc226886
```

Of course, the .E01 file has a different MD5:

```
$ md5 spice1.E01
```

```
MD5 (spice1.E01) = 62f49c77f75cf83c2e316880e45e1dd0
```

ewfexport: turns an EnCase file into a RAW file

Convert from EnCase to RAW:

```
$ ewfexport spice1.E01
```

```
ewfexport 20080820 (libewf 20080820, zlib 1.2.3, libcrypto 0.9.7)
```

```
Information for export required, please provide the necessary input
```

```
Export to file format (raw, ewf, smart, ftk, encase1, encase2, encase3, encase4,  
encase5, encase6, linen5, linen6, ewfx) [raw]:
```

```
Target path and filename with extension or - for stdout: spice1.iso
```

```
Start export at offset (0 >= value >= 32079872) [0]:
```

```
Amount of bytes to export (0 >= value >= 32079872) [32079872]:
```

```
Export started at: Mon Nov 3 21:07:57 2008
```

```
This could take a while.
```

```
Status: at 65%.
```

```
    exported 20 MiB (21004288 bytes) of total 30 MiB (32079872 bytes).
```

```
    completion in 0 second(s) with 30 MiB/s (32079872 bytes/second).
```

```
Export completed at: Mon Nov 3 21:07:58 2008
```

```
Written: 30 MiB (32079872 bytes) in 1 second(s) with 30 MiB/s (32079872 bytes/  
second).
```

```
$
```

ewfverify — Verifies the E01

```
$ ewfverify spice1.E01
```

```
ewfverify 20080820 (libewf 20080820, zlib 1.2.3, libcrypto 0.9.7)
```

```
Verify started at: Mon Nov 3 21:08:34 2008
```

```
This could take a while.
```

```
Verify completed at: Mon Nov 3 21:08:34 2008
```

```
Read: 30 MiB (32079872 bytes) in 0 second(s).
```

```
MD5 hash stored in file: aebfd76cdd9b3eb0f6c1658efc226886
```

```
MD5 hash calculated over data: aebfd76cdd9b3eb0f6c1658efc226886
```

```
ewfverify: SUCCESS
```

```
$
```



File Recovery with
The Sleuth Kit

The Sleuth Kit (TSK) is a tool for working with disk images.

Command-line tools for working with disk images.

Open source computer forensics toolkit

Originally “The Coroner's Toolkit,”
developed by Dan Farmer & Wietse Venema

Rewritten and maintained by Brian Carrier:

- Carrier created a modular internal design.
- Added image layer, disk tools, FAT recover, 64-bit support, live analysis, UFS2 & EXT3 Journal support.
- Coordinating community development

<http://www.sleuthkit.org/>



<http://www.sleuthkit.org/>

Home

Projects

Informer


sleuthkit.org is the official web site for [The Sleuth Kit](#) (a.k.a. digital forensic tools) that run on Windows and Linux. They can be used to analyze NTFS, FAT, HFS+, Ext2, Ext3, and Ext4.

The Sleuth Kit (TSK) is a C library and a collection of command-line tools integrated into automated forensics systems in many ways. It can be used to create, analyze, and recover data.

2ND ANNUAL CONFERENCE

The Sleuth Kit
& Open Source Digital Forensics

June 13, 2011 TUTORIALS / June 14
HILTON MCLEAN TYSONS CORNER, MCLAREN

Recent Updates 

TSK is *the* open source forensic standard.



Image Formats	raw, split-raw, AFF, EWF, etc.
Partitioning Schemes	DOS MBR, GPT, Apple, BSD, Solaris
File Systems	FAT 12/16/32; NTFS; ext2/3; UFS 1/2; ISO9660
Platforms	Linux, OSX, Windows, *BSD, Cygwin, Solaris

Shortcomings:

- No support for encrypted file systems.
- Poor support for compressed files.

Sleuth Kit work directly with disk images.

Common uses:

- View files & directories in a forensically sound manner
- View deleted files
- Document location of information.

Without forensic tools, viewing data can change it!

- "last viewed" and "last modified" times can be changed.
- Entries can be put into the registry.
- Temp files can be created.

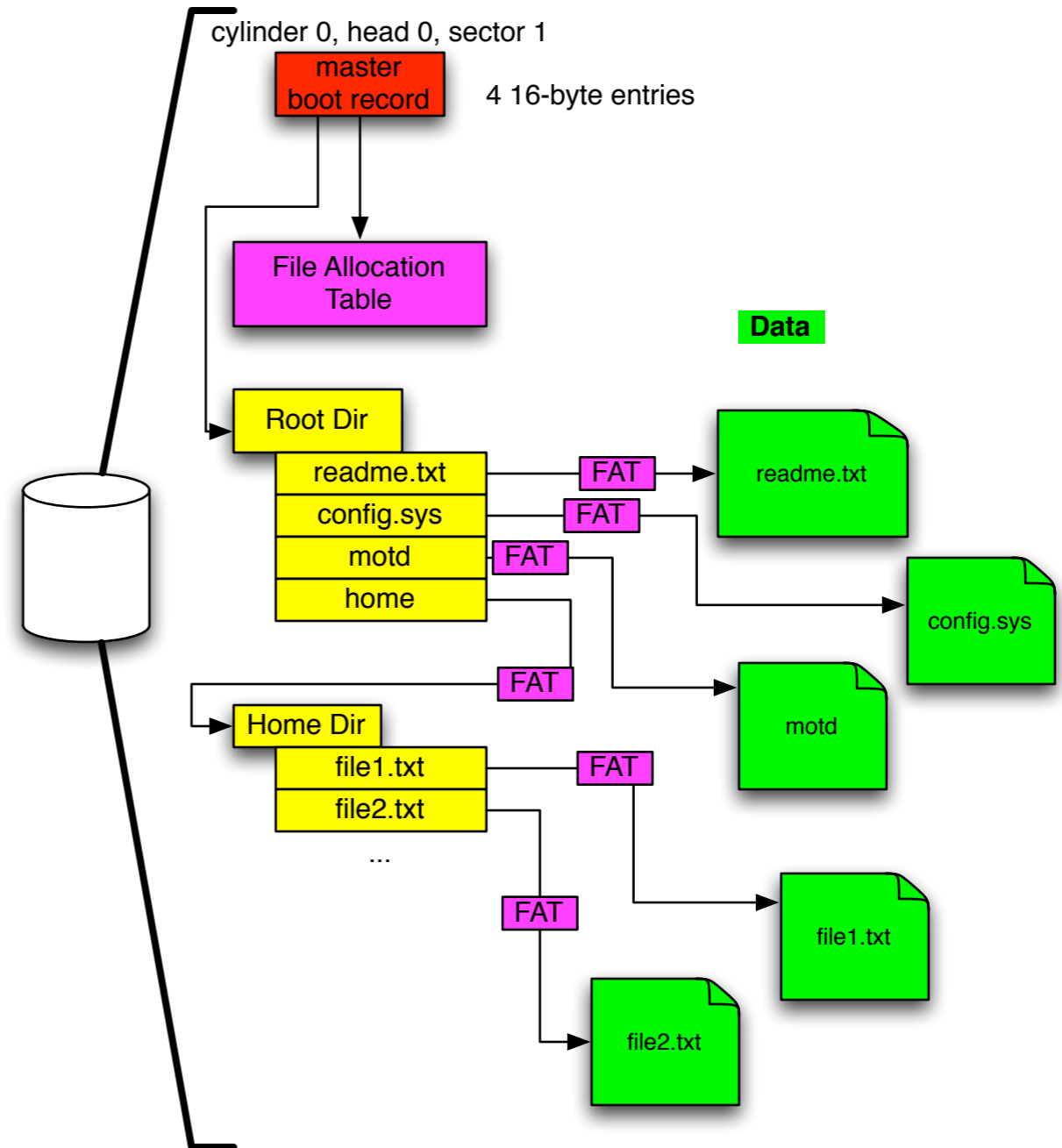
Sleuth Kit works with both *data* and *metadata*.

Data is the content of files.

Metadata tells how to work with the disk and the data.

- Partition table
- List of available sectors
- Directory information

Note: "Metadata" like EXIF and Word "properties" are considered *data* here.

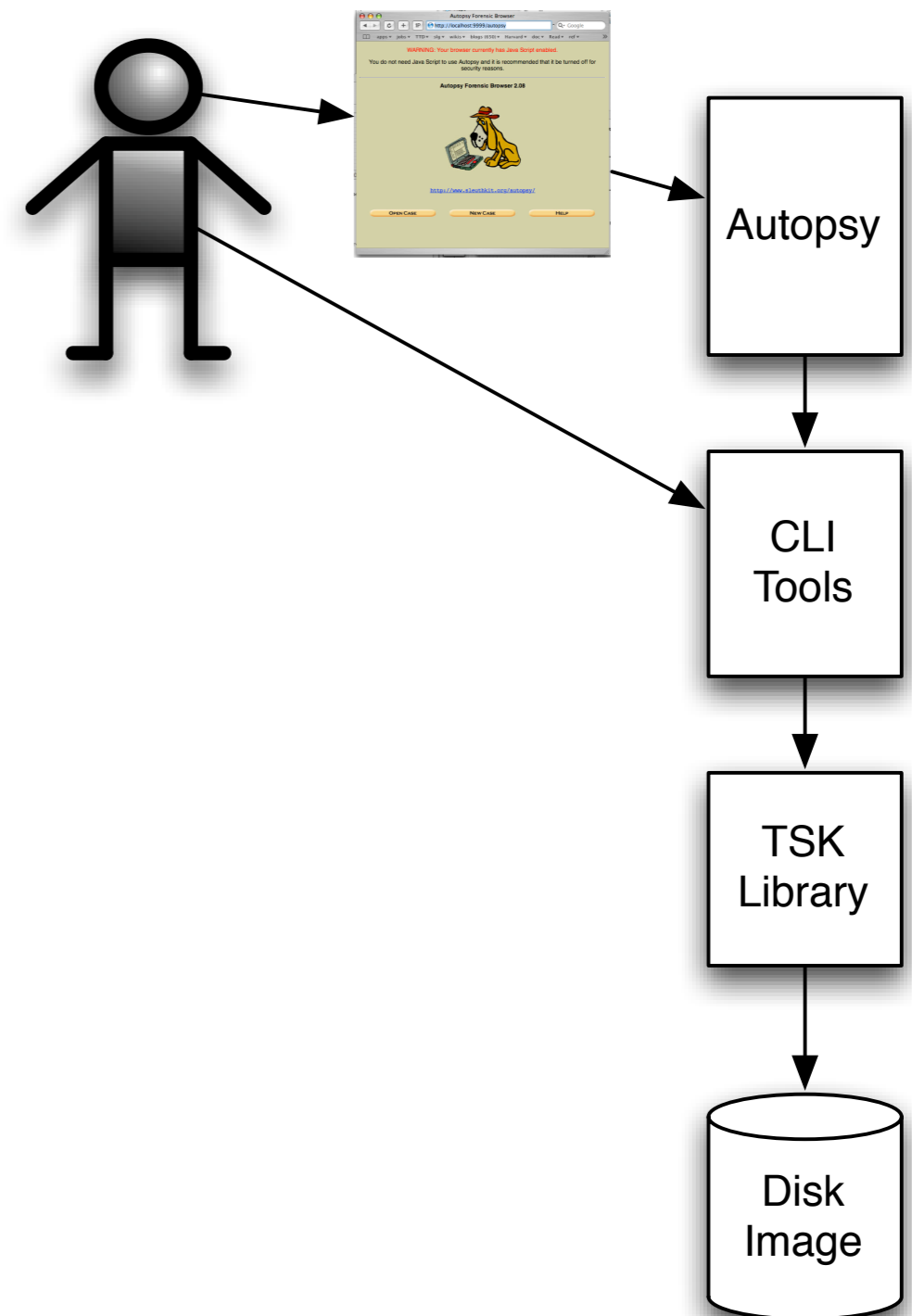


TSK is a modular system

Most TSK commands are run from the command line.

You can also write your own programs that call the library directly.

The Autopsy Forensic Explorer runs the commands and shows you the results in a web browser.



TSK's "f" programs work with file systems.

fls	File List
fsstat	File System Status
ffind	File Find

TSK tools handle many disk image formats:

List the file systems with "-f list":

```
$ fls -i list
```

```
Supported image format types:
```

```
raw (Single raw file (dd))
```

```
aff (Advanced Forensic Format)
```

```
afd (AFF Multiple File)
```

```
afm (AFF with external metadata)
```

```
ewf (Expert Witness format (encase))
```

```
split (Split raw files)
```

```
$
```

To have support for AFF & EWF, you need to separately install them *first!*

TSK routines handle many file systems:

List the file systems with "-f list":

```
$ fls -f list
```

```
Supported file system types:
```

```
ntfs (NTFS)
```

```
fat (FAT (Auto Detection))
```

```
ext (ExtX (Auto Detection))
```

```
iso9660 (ISO9660 CD)
```

```
hfs (HFS+)
```

```
ufs (UFS (Auto Detection))
```

```
raw (Raw Data)
```

```
swap (Swap Space)
```

```
fat12 (FAT12)
```

```
fat16 (FAT16)
```

```
fat32 (FAT32)
```

```
ext2 (Ext2)
```

```
ext3 (Ext3)
```

```
ufs1 (UFS1)
```

```
ufs2 (UFS2)
```

```
$
```

Add support for HFS by editing tsk3/fs/tsk_fs_i.h:

```
// set to 1 to open HFS+ file systems -- which is not fully tested
#ifndef TSK_USE_HFS
#define TSK_USE_HFS 1
#endif
```

Let's look at spice1.raw:

spice1 is an image from 32 MB SD card.

You have three files:

```
$ ls -l spice1*  
-rw-r--r--  1 simsong  staff    263759 Nov  3 21:05 spice1.E01  
-rw-r--r--  1 simsong  staff    215837 Nov  3 22:11 spice1.aff  
-rw-r--r--  1 simsong  staff  32079872 Nov  3 20:54 spice1.raw  
$
```



List files in the disk image with **fls** - File List

```
$ fls spice1.raw
r/r 3:  SPICE          (Volume Label Entry)
d/d * 5: New Folder
d/d 6:  junk
d/d * 8: New Folder
d/d 9:  guns
d/d * 11:   New Folder
d/d * 12:   _rugs
r/r * 13:   _ecret.gif
r/r 14: secret.gif
r/r * 15:   _ecret2.gif
r/r 16: secret2.gif
r/r * 17:   _giastw.jpg
r/r 18: ogiastw.jpg
v/v 994691: $MBR
v/v 994692: $FAT1
v/v 994693: $FAT2
d/d 994694: $OrphanFiles
$
```

Try it!

Options for fls:

```
usage: fls [-adDFlpruvV] [-f fstype] [-i imgtype] [-m dir/] [-o imgoffset] [-z ZONE]
[-s seconds] image [images] [inode]
  If [inode] is not given, the root directory is used
  -a: Display "." and ".." entries
  -d: Display deleted entries only
  -D: Display only directories
  -F: Display only files
  -l: Display long version (like ls -l)
  -i imgtype: Format of image file (use '-i list' for supported types)
  -f fstype: File system type (use '-f list' for supported types)
  -m: Display output in mactime input format with
      dir/ as the actual mount point of the image
  -o imgoffset: Offset into image file (in sectors)
  -p: Display full path for each file
  -r: Recurse on directory entries
  -u: Display undeleted entries only
  -v: verbose output to stderr
  -V: Print version
  -z: Time zone of original machine (i.e. EST5EDT or GMT) (only useful with -l)
  -s seconds: Time skew of original machine (in seconds) (only useful with -l & -m)
```

Show *all* the files with full path names:
fls -rp spice1.raw

Try it!

```
$ fls -rp spice1.raw
r/r 3:  SPICE          (Volume Label Entry)
d/d * 5: New Folder
d/d 6:  junk
r/r * 517:  junk/_an1.jpg
r/r 518: junk/man1.jpg
r/r * 519:  junk/_an2.jpg
r/r 520: junk/man2.jpg
d/d * 8: New Folder
d/d 9:  guns
r/r * 533:  guns/_unpage1.htm
r/r 534: guns/gunpage1.htm
d/d * 11:   New Folder
r/r * 549:  New Folder/_rugs1.htm
d/d * 551:  New Folder/drugs1_files
r/r * 552:  New Folder/_rugs1.htm
d/d * 12:   _rugs
r/r * 549:  _rugs/_rugs1.htm
d/d * 551:  _rugs/drugs1_files
r/r * 552:  _rugs/_rugs1.htm
r/r * 13:   _ecret.gif
r/r 14: secret.gif
r/r * 15:   _ecret2.gif
r/r 16: secret2.gif
r/r * 17:   _giastw.jpg
r/r 18: ogiastw.jpg
v/v 994691: $MBR
v/v 994692: $FAT1
v/v 994693: $FAT2
```

What do these numbers mean?

`r/r 14: secret.gif`

<code>r/r</code>	Regular file
<code>14</code>	metadata block # ("inode")
<code>secret.gif</code>	file name

Use **icat** and the inode # to get the file contents.

```
r/r 14: secret.gif
```

```
$ icat spicel.raw 14 > 14.jpg
```

```
$ open 14.jpg mac
```

```
$ gnome-open 14.jpg gnome
```

```
> start 14.jpg Windows
```

Use **icat** and the inode # to get the file contents.

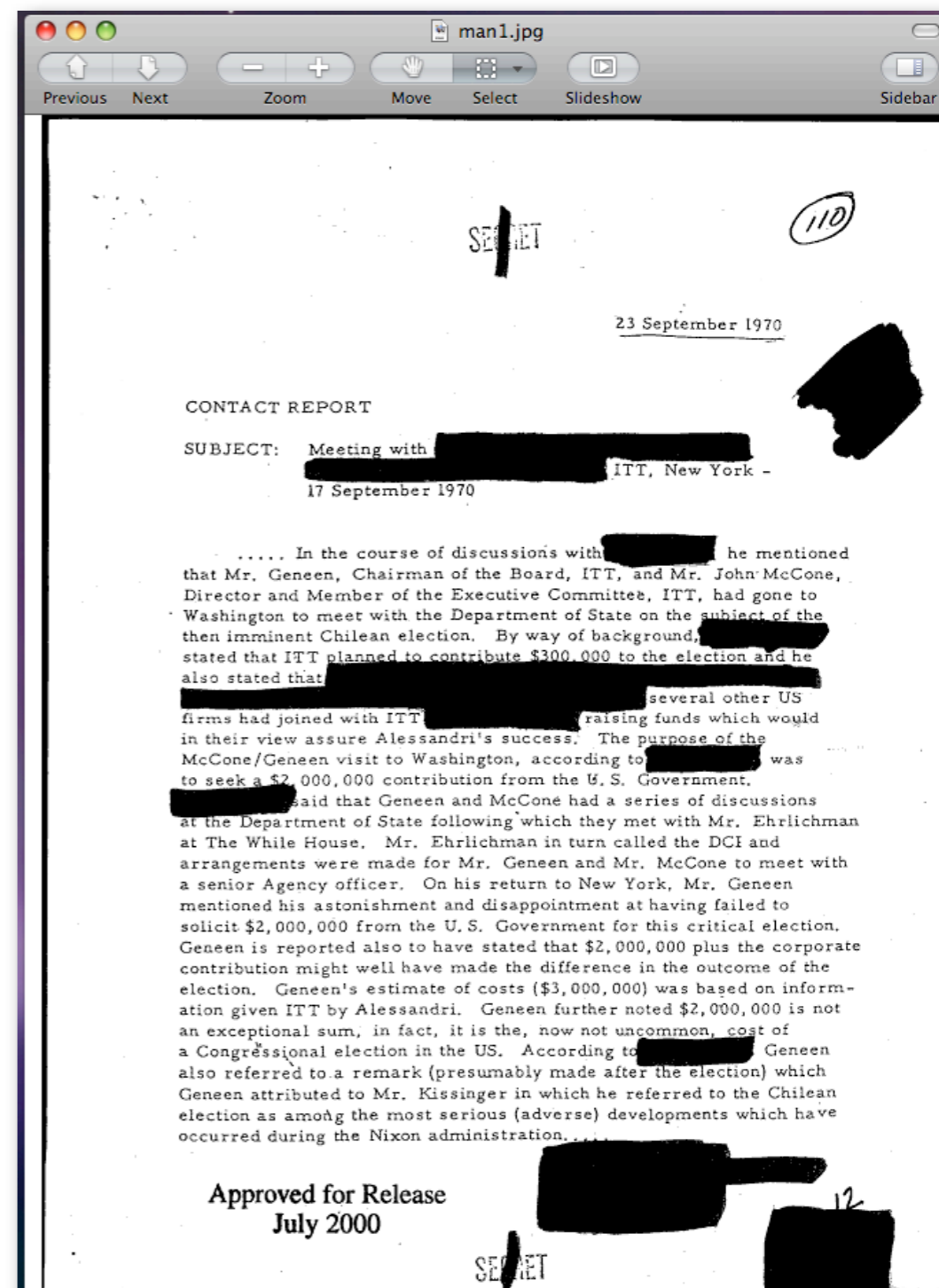
r/r 14: secret.gif

\$ **icat spicel.raw 14 > 14.jpg**

\$ **open 14.jpg** *mac*

\$ **gnome-open 14.jpg** *gnome*

> **start 14.jpg** *Windows*



Look at the contents of ogiastw.jpg!

Try it!

```
$ fls -rp spice1.raw
r/r 3:  SPICE          (Volume Label Entry)
d/d * 5: New Folder
d/d 6:  junk
r/r * 517:  junk/_an1.jpg
r/r 518: junk/man1.jpg
r/r * 519:  junk/_an2.jpg
r/r 520: junk/man2.jpg
d/d * 8: New Folder
d/d 9:  guns
r/r * 533:  guns/_unpage1.htm
r/r 534: guns/gunpage1.htm
d/d * 11:   New Folder
r/r * 549:  New Folder/_rugs1.htm
d/d * 551:  New Folder/drugs1_files
r/r * 552:  New Folder/_rugs1.htm
d/d * 12:   _rugs
r/r * 549:  _rugs/_rugs1.htm
d/d * 551:  _rugs/drugs1_files
r/r * 552:  _rugs/_rugs1.htm
r/r * 13:   _ecret.gif
r/r 14: secret.gif
r/r * 15:   _ecret2.gif
r/r 16: secret2.gif
r/r * 17:   _giastw.jpg
r/r 18: ogiastw.jpg
v/v 994691: $MBR
v/v 994692: $FAT1
v/v 994693: $FAT2
```

Try it!!

Look at the contents of ogiastw.jpg!

Here is another way to do it:

First, use **ifind** to get the inode number of the file:

```
$ ifind -n /ogiastw.jpg spice1.iso
18
$
```

Next, use **icat** to extract the file from the disk image:

```
$ icat spice1.iso 18 > 18.jpg
$ open 18.jpg
```

Try it!!

Look at the contents of ogiastw.jpg!

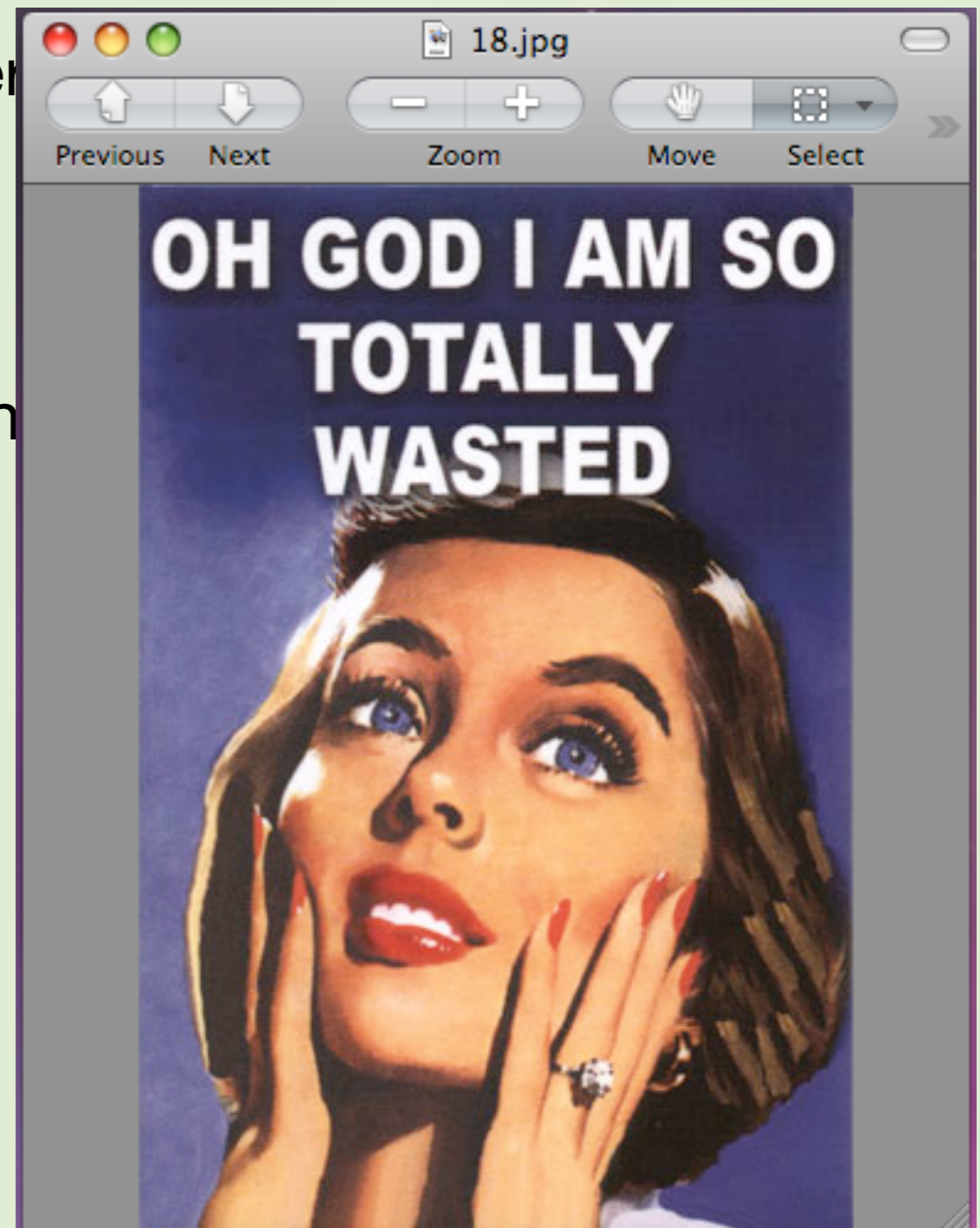
Here is another way to do it:

First, use **ifind** to get the inode number

```
$ ifind -n /ogiastw.jpg spice1.iso  
18  
$
```

Next, use **icat** to extract the file from the

```
$ icat spice1.iso 18 > 18.jpg  
$ open 18.jpg
```



fsstat shows technical details about the file system.

```
$ fsstat spice1.raw
```

FILE SYSTEM INFORMATION

```
-----  
File System Type: FAT16
```

```
OEM Name: MSDOS5.0
```

```
Volume ID: 0x64fb06c6
```

```
Volume Label (Boot Sector): NO NAME
```

```
Volume Label (Root Directory): SPICE
```

```
File System Type Label: FAT16
```

```
Sectors before file system: 64
```

File System Layout (in sectors)

```
Total Range: 0 - 62655
```

```
* Reserved: 0 - 1
```

```
** Boot Sector: 0
```

```
* FAT 0: 2 - 244
```

```
* FAT 1: 245 - 487
```

```
* Data Area: 488 - 62655
```

```
** Root Directory: 488 - 519
```

```
** Cluster Area: 520 - 62655
```

METADATA INFORMATION

```
-----  
Range: 2 - 994694
```

```
Root Directory: 2
```

CONTENT INFORMATION

```
-----  
Sector Size: 512
```

```
Cluster Size: 512
```

```
Total Cluster Range: 2 - 62137
```

FAT CONTENTS (in sectors)

```
-----  
520-520 (1) -> EOF
```

```
521-521 (1) -> EOF
```

```
523-526 (4) -> EOF
```

```
527-539 (13) -> EOF
```

```
540-610 (71) -> EOF
```

```
611-687 (77) -> EOF
```

```
688-795 (108) -> EOF
```

```
796-864 (69) -> EOF
```

```
$
```

img_stat shows information about the disk image.

```
$ img_stat spice1.raw
```

```
IMAGE FILE INFORMATION
```

```
-----
```

```
Image Type: raw
```

```
Size in bytes: 32079872
```

```
$
```

```
$ img_stat spice1.E01
```

```
IMAGE FILE INFORMATION
```

```
-----
```

```
Image Type:      ewf
```

```
Size of data in bytes: 32079872
```

```
MD5 hash of data:  aebfd76cdd9b3eb0f6c1658efc226886
```

```
$
```

TSK command-line programs divided up by layer.

j-	journal layer
f-	file name layer
i-	metadata (inode) layer
blk-	content (data) layer
mm-	volumes/partitions
img_-	Disk images

TSK command-line programs divided by function.

-stat	print status
-ls	list something
-find	find something
-cat	output contents
-calc	compute something

Here are the commands we've used so far.

"f" tools work with file systems:

- `fsstat` — File system stat
- `fls` — list files and their inodes
- `ffind` — translates an inode number back to a file.

"i" tools work with file system metadata (*inodes & MFT*)

- `ifind` — Finds the metadata given a data unit (-d), a file name (-n), or the parent's metadata address (-p)
- `icat` — Outputs the contents of an inode.

"img" tools work with disk images:

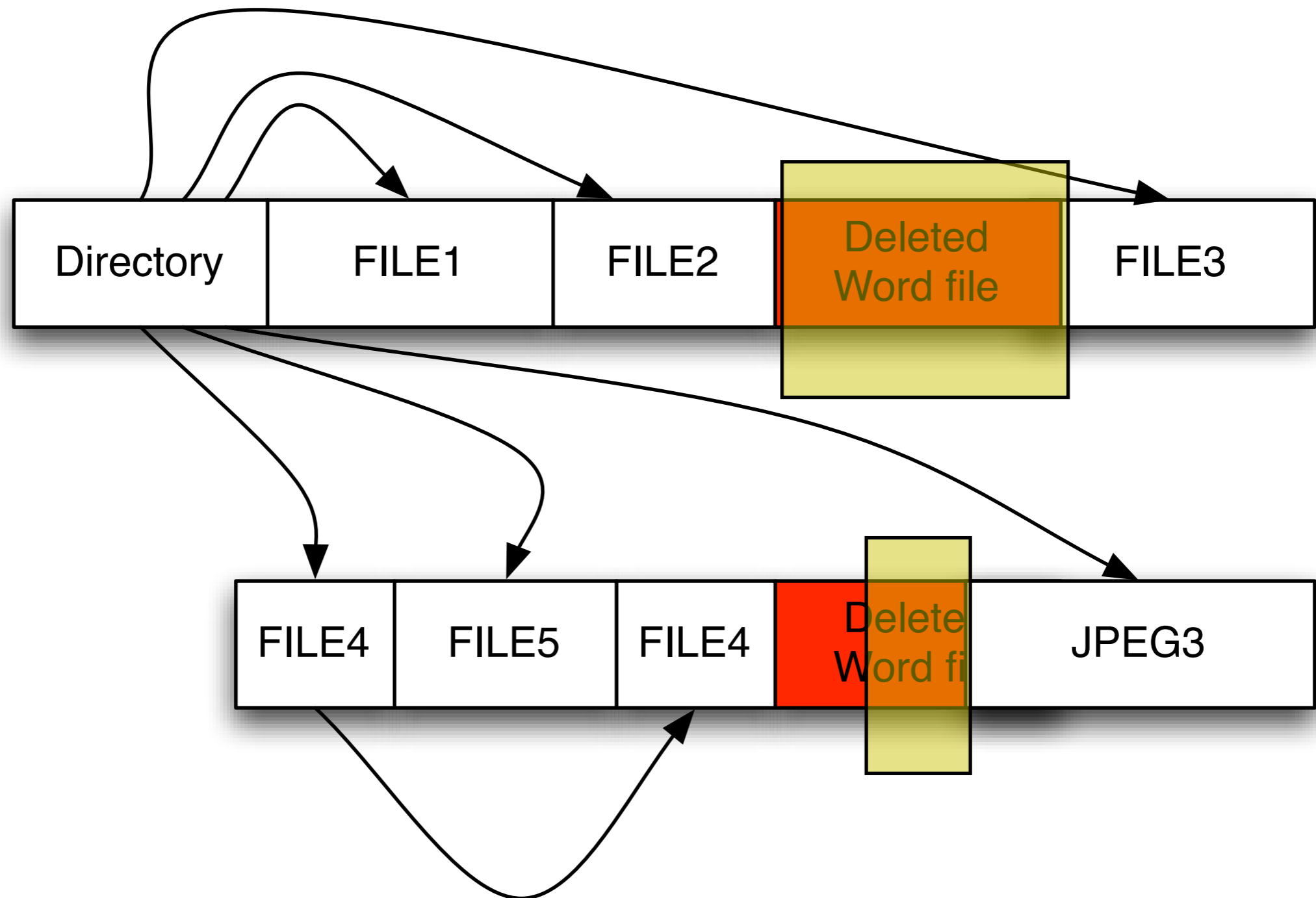
- `img_stat` — Prints statistics about the image
- `img_cat` — Copies the raw sectors to stdout.



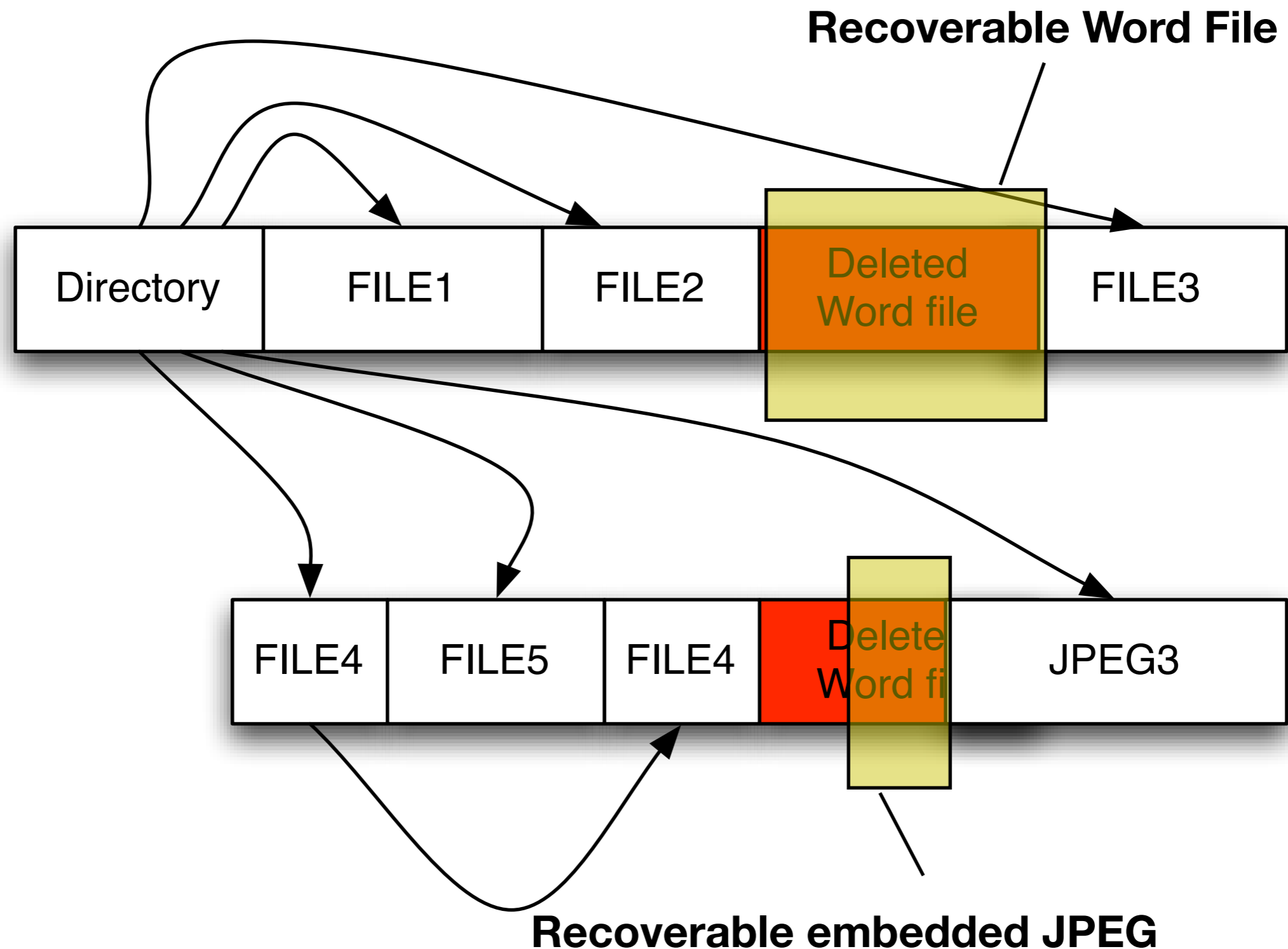


File Carving

“Carving” searches for objects based on content, rather than on metadata.



“Carving” searches for objects based on content, rather than on metadata.



File carving is a powerful tool for finding useful pieces of information.

What can be carved:

- Disks & Disk Images
- Memory
- Files of unknown format (to find embedded objects)

Objects that can be recovered:

- Images
- Text files & documents
- Cryptographic Keys

Why carve?

- Directory entries are overwritten
- Directory entries are damaged
- File formats aren't known

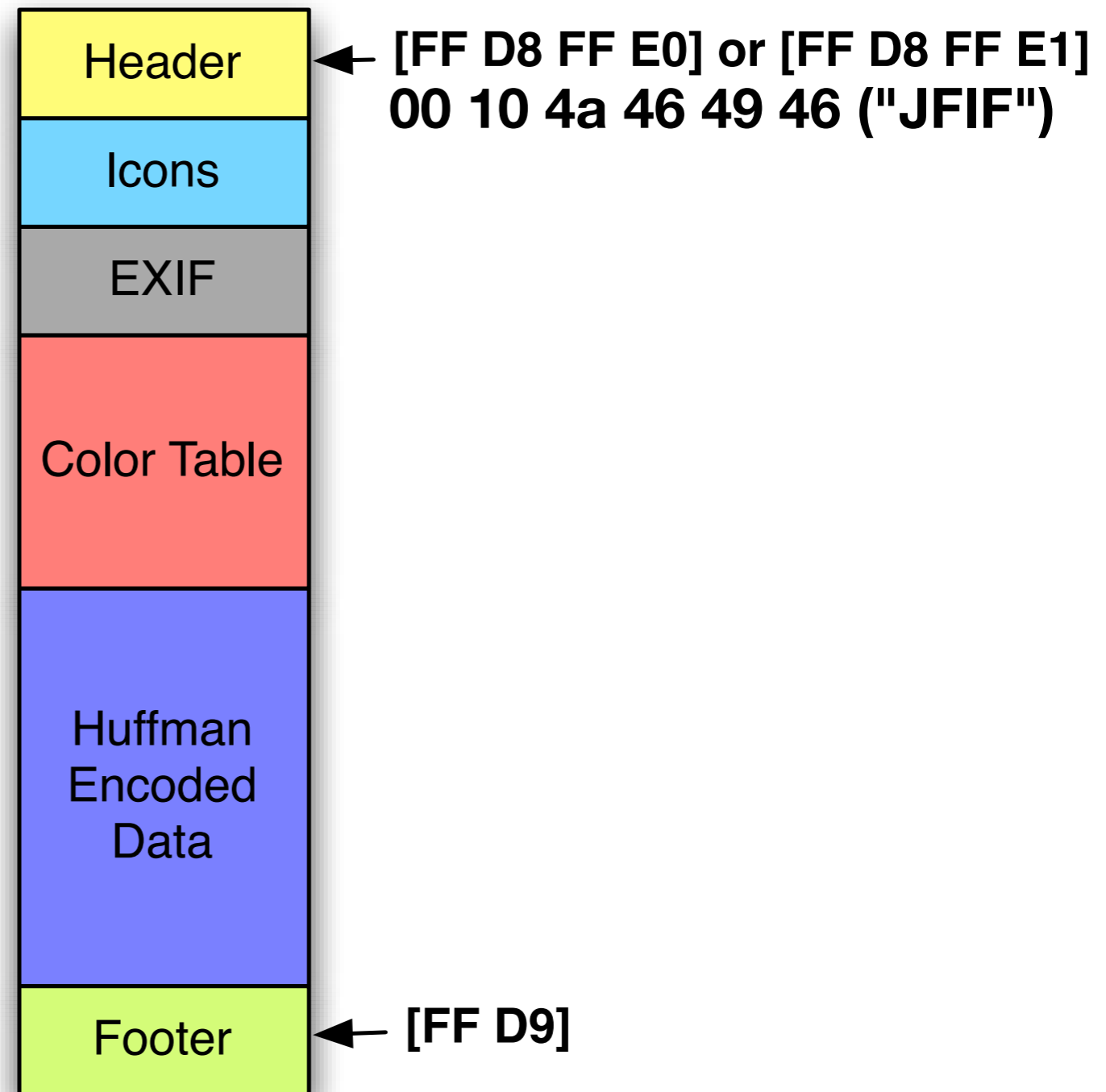
Example: Carving JPEG Files

JPEGs are container files

- Standard Header
- Standard Footer
- Embedded Images

Carving strategy:

- Find all headers
- Find all footers
- Save sectors to files



Header/Footer carving involves saving the data between a known header & known footer.



Disk Sectors ➡

Header/Footer carving involves saving the data between a known header & known footer.



Disk Sectors ➡

Header/Footer carving involves saving the data between a known header & known footer.



Disk Sectors ➔



Header/Footer carving involves saving the data between a known header & known footer.

This strategy is used by foremost and scalpel.

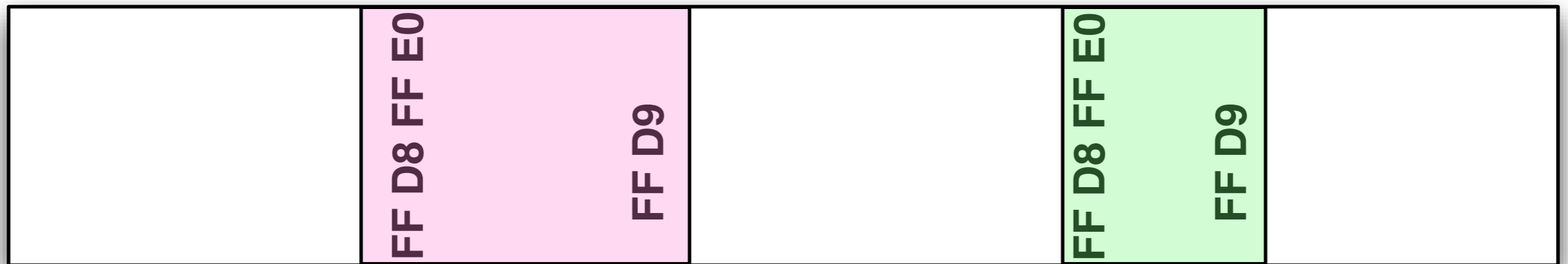


Disk Sectors ➔



Header/Footer carving involves saving the data between a known header & known footer.

This strategy is used by foremost and scalpel.

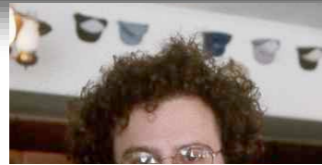
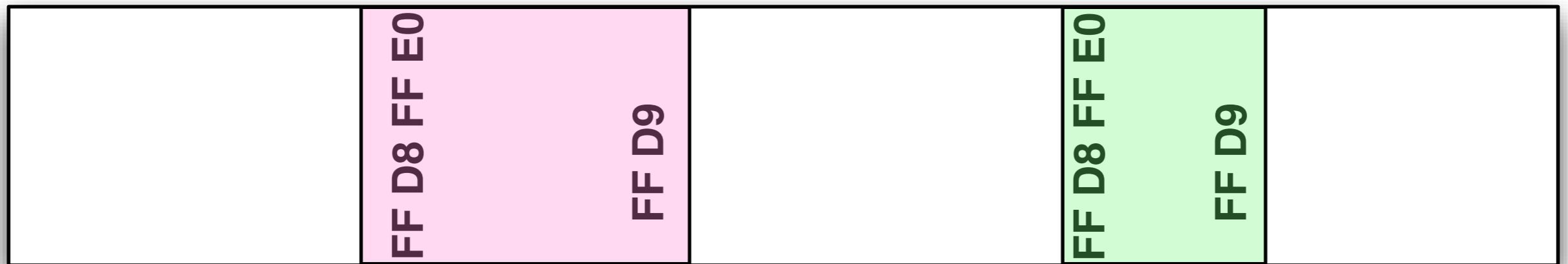


Disk Sectors ➔



Header/Footer carving involves saving the data between a known header & known footer.

This strategy is used by foremost and scalpel.

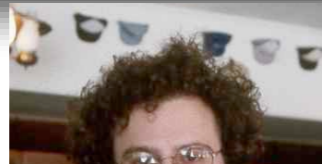


Disk Sectors ➔



Header/Footer carving involves saving the data between a known header & known footer.

This strategy is used by foremost and scalpel.

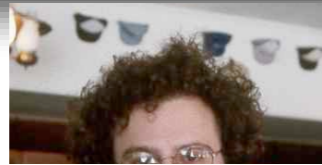
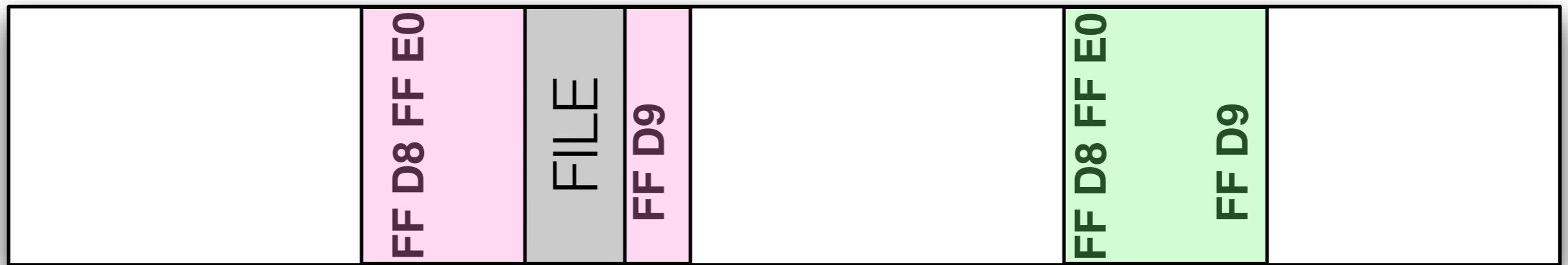


Disk Sectors ➔



Header/Footer carving involves saving the data between a known header & known footer.

This strategy is used by foremost and scalpel.



Disk Sectors ➔



Possible explanations:

1.This file may be fragmented.

2.The file may have been overwritten.

**If the file is fragmented, it can be recovered with
*fragment recovery carving***

Is fragment reassembly carving important?

We analyzed 400 hard drives to find out.

Today's file carvers cannot process fragmented files.

My research group has disk images from used hard drives acquired around the world.

These drives simulate drives taken from production during a search.

≈275 had relevant file systems.



Files can be fragmented into two or more pieces.

	FAT ¹	NTFS	UFS
# File systems:	219	51	5
# Fragments	Number of Files		
(contiguous)	1,285,975	502,050	70,222
2	25,151	20,851	10,932
3	4,929	5,622	1,047
4	2,473	3,176	408
5–10	4,340	11,730	658
11–20	1,591	7,001	94
21–100	1,246	10,912	13
101–1000	185	5,672	0
1001–	2	567	0
Total Files:	1,325,892	567,581	83,374

Forensically important files are more likely to be fragmented than non-important files.

This is result of:

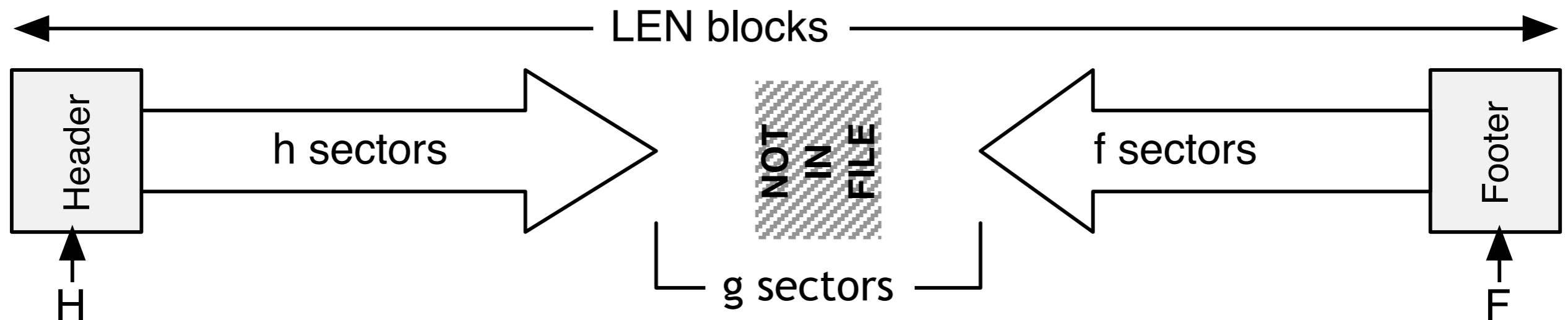
- Incremental writing of log files.
- Writing files to disks that have been in use for an extended period of time.
- Microsoft Word update strategies.

Ext	file count	Size of files with 2 fragments:		
		avg	stddev	max
pnf	7,583	41,583	81,108	1,317,368
dll	7,479	221,409	384,758	9,857,608
html	3,417	28,388	66,694	2,505,490
jpeg	2,963	29,673	178,563	6,601,153
gif	2,566	22,133	99,370	3,973,951
exe	2,348	399,528	4,354,053	206,199,144
1	1,125	57,475	130,630	1,998,576
dat	780	291,407	673,906	7,793,936
z	716	74,353	340,808	6,248,869
h	690	16,444	12,232	110,592
inf	683	79,578	101,448	522,916
wav	575	1,949,459	6,345,280	39,203,180
swf	548	62,582	120,138	1,155,989
ttf	540	163,854	649,919	10,499,104
sys	513	1,276,323	12,446,966	150,994,944
txt	480	33,410	275,641	5,978,896
hlp	475	185,259	375,461	3,580,078
tmp	450	206,908	772,290	8,388,608
so	440	103,939	205,617	1,501,148
wmf	418	48,864	49,869	586,414
...

Table 7: Most common files in corpus consisting of two fragments, by file extension.

Fragment Recovery Carving can reassemble fragmented files using validation.

Fragment Recovery Carving:



$$\text{LEN} = S - F + 1$$

```
for I in range(0, LEN):
```

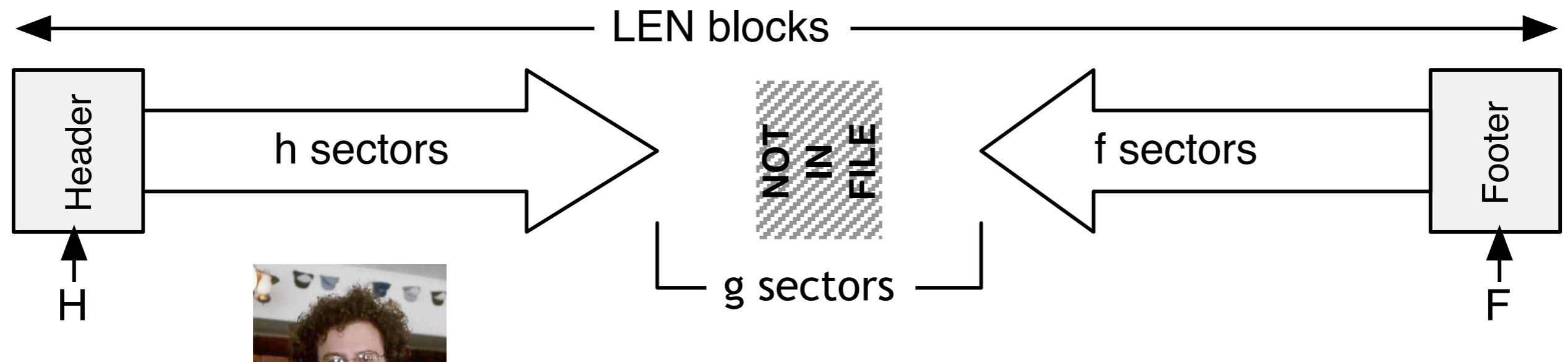
```
    for J in range(0, LEN - I):
```

```
        data = blocks[S:S+I] + blocks[F-J:J]
```

```
        if valid(data) == True: save(data)
```

Fragment Recovery Carving can reassemble fragmented files using validation.

Fragment Recovery Carving:

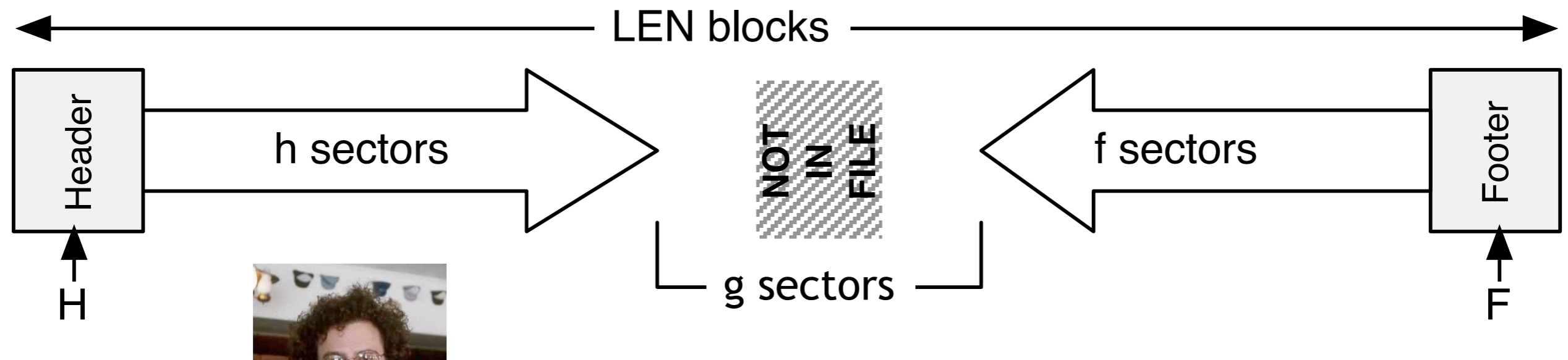


$$LEN = S - F + 1$$

```
for I in range(0,LEN):  
    for J in range(0,LEN-I):  
        data = blocks[S:S+I] + blocks[F-J:J]  
        if valid(data)==True: save(data)
```

Fragment Recovery Carving can reassemble fragmented files using validation.

Fragment Recovery Carving:



$$LEN = S - F + 1$$

```
for I in range(0, LEN):
```

```
    for J in range(0, LEN - I):
```

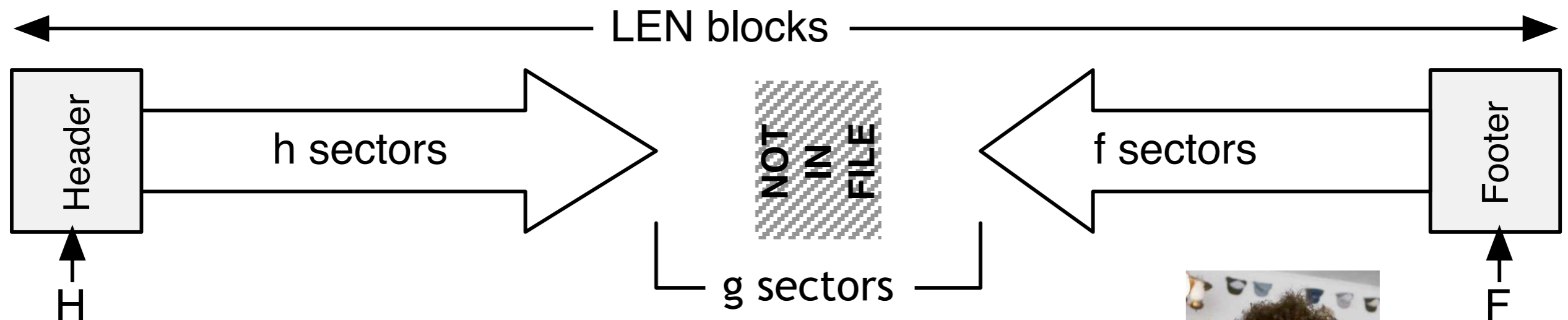
```
        data = blocks[S:S+I] + blocks[F-J:J]
```

```
        if valid(data) == True: save(data)
```

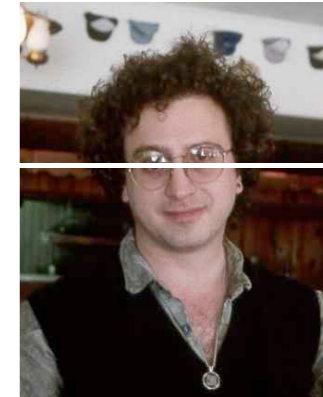


Fragment Recovery Carving can reassemble fragmented files using validation.

Fragment Recovery Carving:

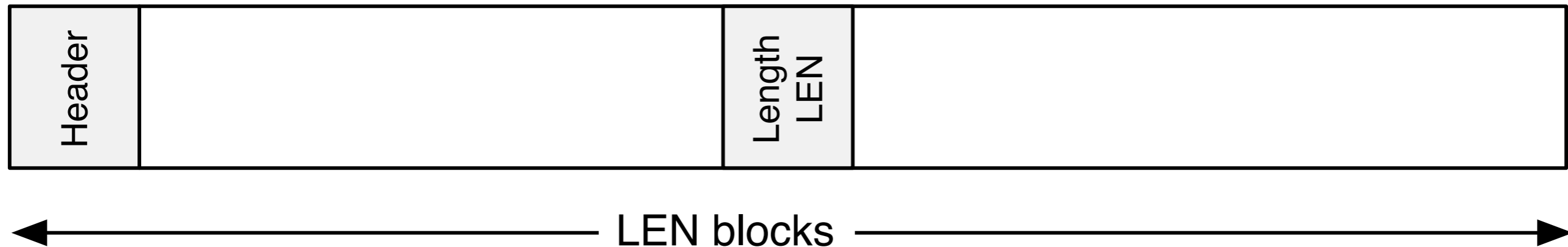


```
LEN = S-F+1
for I in range(0,LEN):
    for J in range(0,LEN-I):
        data = blocks[S:S+I] + blocks[F-J:J]
        if valid(data)==True: save(data)
```



Header/Length Carving takes advantage of blocks that code a file's length.

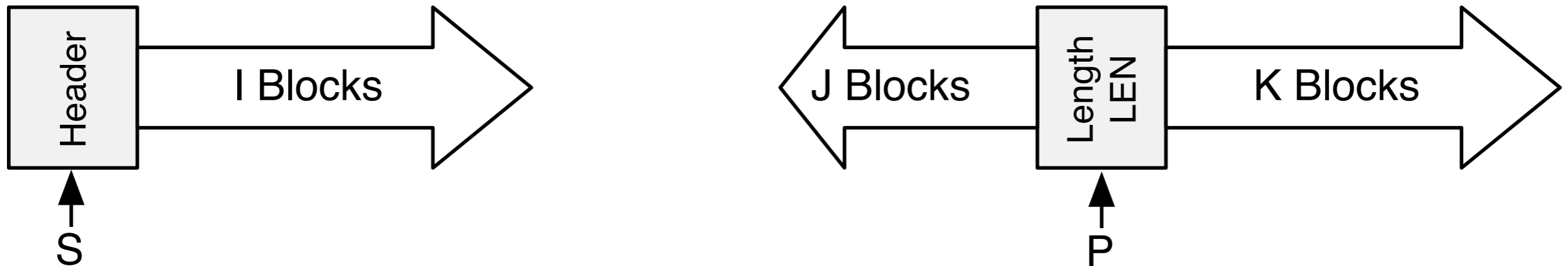
Header/Length sectors: (LEN blocks are found in ZIP & MSOffice)



Header/Embedded Length Carving:

- Looks for structures that code length.
- Works with MS Office and ZIP files

Header/Length Fragment Recovery Carving:



```
for I in range(0,LEN):  
    for J in range(0,LEN-I):  
        K = LEN - (I+J)  
        data = blocks[S:S+I] + blocks[P-J:P+K]  
        if valid(data)==True: save(data)
```

Carving tools available today:

Open Source:

- **Foremost** - Developed by Jesse Kornblum and Kris Kendall at AFOSI
- **Scalpel** - Improved version of Foremost, by Golden G. Richard III
- **CarvFS** - Virtual file system for carving
- **PhotoRec** - Recovers lost photos from hard drives
- **RevIT & S2** - Experimental carvers developed for DFRWS 2006 carving challenge

Commercial:

- **Adroit Photo Recovery** — Amazing, but only works on JPEGs
- **EnCase** - comes with some eScripts that will carve
- **DataLifter** - File Extractor Pro

Let's use scalpel to find those JFIF files...

```
$ tar xfz scalpel-1.60.tar.gz
$ cd scalpel-1.60;make bsd
gcc -Wall -O2      -D__OPENBSD  -c helpers.c
gcc -Wall -O2      -D__OPENBSD  -c scalpel.c
gcc -Wall -O2      -D__OPENBSD  -c files.c
gcc -Wall -O2      -D__OPENBSD  -c dig.c
gcc -Wall -O2      -D__OPENBSD  -c prioque.c
gcc -Wall -O2      -D__OPENBSD  -c base_name.c
gcc -Wall -O2      -D__OPENBSD  -o scalpel
helpers.o scalpel.o files.o dig.o prioque.o
base_name.o -lm
$
```

Edit the Scalpel config file to look for JPEGs...

```
#-----
# GRAPHICS FILES
#-----
#
# GIF and JPG files (very common)
#
# gif      y      5000000      \x47\x49\x46\x38\x37\x61      \x00\x3b
# gif      y      5000000      \x47\x49\x46\x38\x39\x61      \x00\x3b
# jpg      y      200000000     \xff\xd8\xff\xe0\x00\x10      \xff\xd9
#
#
# PNG
#
# png      y      200000000     \x50\x4e\x47?      \xff\xfc\xfd\xfe
#
#
# BMP      (used by MSWindows, use only if you have reason to think there are
# BMP files worth digging for. This often kicks back a lot of false
# positives
#
# bmp      y      100000      BM??\x00\x00\x00
#
# TIFF
#
# tif      y      200000000     \x49\x49\x2a\x00
#
# TIFF
#
# tif      y      200000000     \x4D\x4D\x00\x2A
```

Edit the Scalpel config file to look for JPEGs...

```
#-----  
# GRAPHICS FILES  
#-----  
#  
# GIF and JPG files (very common)  
    gif      y      5000000      \x47\x49\x46\x38\x37\x61      \x00\x3b  
    gif      y      5000000      \x47\x49\x46\x38\x39\x61      \x00\x3b  
    jpg      y      200000000     \xff\xd8\xff\xe0\x00\x10      \xff\xd9  
#  
#  
# PNG  
    png      y      200000000     \x50\x4e\x47?      \xff\xfc\xfd\xfe  
#  
#  
# BMP (used by MSWindows, use only if you have reason to think there are  
# BMP files worth digging for. This often kicks back a lot of false  
# positives  
#  
    bmp      y      100000      BM??\x00\x00\x00  
#  
# TIFF  
    tif      y      200000000     \x49\x49\x2a\x00  
# TIFF  
    tif      y      200000000     \x4D\x4D\x00\x2A
```

Edit the Scalpel config file to look for JPEGs...

```
#-----  
# GRAPHICS FILES  
#-----  
#  
# GIF and JPG files (very common)  
    gif      y      5000000      \x47\x49\x46\x38\x37\x61      \x00\x3b  
    gif      y      5000000      \x47\x49\x46\x38\x39\x61      \x00\x3b  
    jpg      y      200000000     \xff\xd8\xff\xe0\x00\x10      \xff\xd9  
#  
#  
# PNG  
    png      y      200000000     \x50\x4e\x47?      \xff\xfc\xfd\xfe  
#  
#  
# BMP      (used by MSWindows, use only if you have reason to think there are  
# BMP files worth digging for. This often kicks back a lot of false  
# positives  
#  
    bmp      y  
#  
# TIFF  
    tif      y      200000000     \x49\x49\x2a\x00  
# TIFF  
    tif      y      200000000     \x4D\x4D\x00\x2A
```

Case Sensitive Header/Footer

Extension

Edit the Scalpel config file to look for JPEGs...

```
#-----  
# GRAPHICS FILES  
#-----  
#  
# GIF and JPG files (very common)  
    gif      y      5000000      \x47\x49\x46\x38\x37\x61      \x00\x3b  
    gif      y      5000000      \x47\x49\x46\x38\x39\x61      \x00\x3b  
    jpg      y      200000000     \xff\xd8\xff\xe0\x00\x10      \xff\xd9  
#  
#  
# PNG  
    png      y      200000000     \x50\x4e\x47?      \xff\xfc\xfd\xfe  
#  
#  
# BMP (used by MSWindows, use only if you have reason to think there are  
# BMP files worth digging for. This often kicks back a lot of false  
# positives  
#  
    bmp      y  
#  
# TIFF  
    tif      y      200000000     \x49\x49\x2a\x00  
# TIFF  
    tif      y      200000000     \x4D\x4D\x00\x2A
```

Case Sensitive Header/Footer



Extension

Edit the Scalpel config file to look for JPEGs...

```
#-----
# GRAPHICS FILES
#-----
#
# GIF and JPG files (very common)
#
#   gif      y      5000000      \x47\x49\x46\x38\x37\x61      \x00\x3b
#   gif      y      5000000      \x47\x49\x46\x38\x39\x61      \x00\x3b
#   jpg      y      200000000     \xff\xd8\xff\xe0\x00\x10      \xff\xd9
#
#
# PNG
#
#   png      y      200000000     \x50\x4e\x47?      \xff\xfc\xfd\xfe
#
#
# BMP (used by MSWindows, use only if you have reason to think there are
# BMP files worth digging for. This often kicks back a lot of false
# positives
#
#   bmp      y
#
# TIFF
#
#   tif      y      200000000     \x49\x49\x2a\x00
#
# TIFF
#
#   tif      y      200000000     \x4D\x4D\x00\x2A
```

Case Sensitive Header/Footer



Extension

Max Size

Edit the Scalpel config file to look for JPEGs...

```
#-----
# GRAPHICS FILES
#-----
#
# GIF and JPG files (very common)
#
#   gif      y      5000000      \x47\x49\x46\x38\x37\x61      \x00\x3b
#   gif      y      5000000      \x47\x49\x46\x38\x39\x61      \x00\x3b
#   jpg      y      200000000     \xff\xd8\xff\xe0\x00\x10      \xff\xd9
#
#
# PNG
#
#   png      y      200000000     \x50\x4e\x47?      \xff\xfc\xfd\xfe
#
#
# BMP (used by MSWindows, use only if you have reason to think there are
# BMP files worth digging for. This often kicks back a lot of false
# positives
#
#   bmp      y
#
# TIFF
#
#   tif      y      200000000     \x49\x49\x2a\x00
#
# TIFF
#
#   tif      y      200000000     \x4D\x4D\x00\x2A
```

Case Sensitive Header/Footer



Extension

Max Size

Header

Edit the Scalpel config file to look for JPEGs...

```
#-----
# GRAPHICS FILES
#-----
#
# GIF and JPG files (very common)
#
#   gif      y      5000000      \x47\x49\x46\x38\x37\x61      \x00\x3b
#   gif      y      5000000      \x47\x49\x46\x38\x39\x61      \x00\x3b
#   jpg      y      200000000     \xff\xd8\xff\xe0\x00\x10      \xff\xd9
#
#
# PNG
#
#   png      y      200000000     \x50\x4e\x47?      \xff\xfc\xfd\xfe
#
#
# BMP (used by MSWindows, use only if you have reason to think there are
# BMP files worth digging for. This often kicks back a lot of false
# positives
#
#   bmp      y
#
# TIFF
#
#   tif      y      200000000     \x49\x49\x2a\x00
#
# TIFF
#
#   tif      y      200000000     \x4D\x4D\x00\x2A
```

Footer

Case Sensitive Header/Footer



Extension

Max Size

Header

Run scalpel with memory image as input file...

```
$ ./scalpel -c scalpel.conf -o outdir1 ~/image.dd  
Scalpel version 1.60  
Written by Golden G. Richard III, based on Foremost 0.69.
```

```
Opening target "/Users/simsong/image.dd"
```

```
Image file pass 1/2.
```

```
/Users/simsong/image.dd: 19.5% |*****| 100.0 MB  
00:21 ETA
```

Run scalpel with memory image as input file...

Carve lists built. Workload:

```
gif with header "\x47\x49\x46\x38\x37\x61" and footer "\x00\x3b" --> 9 files
gif with header "\x47\x49\x46\x38\x39\x61" and footer "\x00\x3b" --> 103 files
jpg with header "\xff\xd8\xff\xe0\x00\x10" and footer "\xff\xd9" --> 15 files
png with header "\x50\x4e\x47\x3f" and footer "\xff\xfc\xfd\xfe" --> 5 files
bmp with header "\x42\x4d\x3f\x3f\x00\x00\x00" and footer "" --> 32 files
tif with header "\x49\x49\x2a\x00" and footer "" --> 2 files
tif with header "\x4d\x4d\x00\x2a" and footer "" --> 3 files
```

Carving files from image.

Image file pass 2/2.

/Users/simsong/image.dd: 100.0% |

*****| 512.0 MB 00:00

ETAProcessing of image file complete. Cleaning up...

Done.

Scalpel is done, files carved = 169, elapsed = 45 seconds.

\$ ls -l outdir

total 12

-rw-r--r--	1	simsong	simsong	10055	Oct	5	18:36	audit.txt
drwxr-xr-x	34	simsong	simsong	1156	Oct	5	18:35	bmp-4-0/
drwxr-xr-x	11	simsong	simsong	374	Oct	5	18:36	gif-0-0/
drwxr-xr-x	105	simsong	simsong	3570	Oct	5	18:36	gif-1-0/
drwxr-xr-x	17	simsong	simsong	578	Oct	5	18:35	jpg-2-0/
drwxr-xr-x	7	simsong	simsong	238	Oct	5	18:35	png-3-0/
drwxr-xr-x	4	simsong	simsong	136	Oct	5	18:35	tif-5-0/
drwxr-xr-x	5	simsong	simsong	170	Oct	5	18:35	tif-6-0/

\$

Run scalpel with memory image as input file...

Scalpel version 1.60 audit file

Started at Sun Oct 5 18:35:20 2008

Command line:

```
./scalpel -c scalpel.conf -o outdir1 /Users/simsong/image.dd
```

Output directory: /Users/simsong/scalpel-1.60/outdir1

Configuration file: /Users/simsong/scalpel-1.60/scalpel.conf

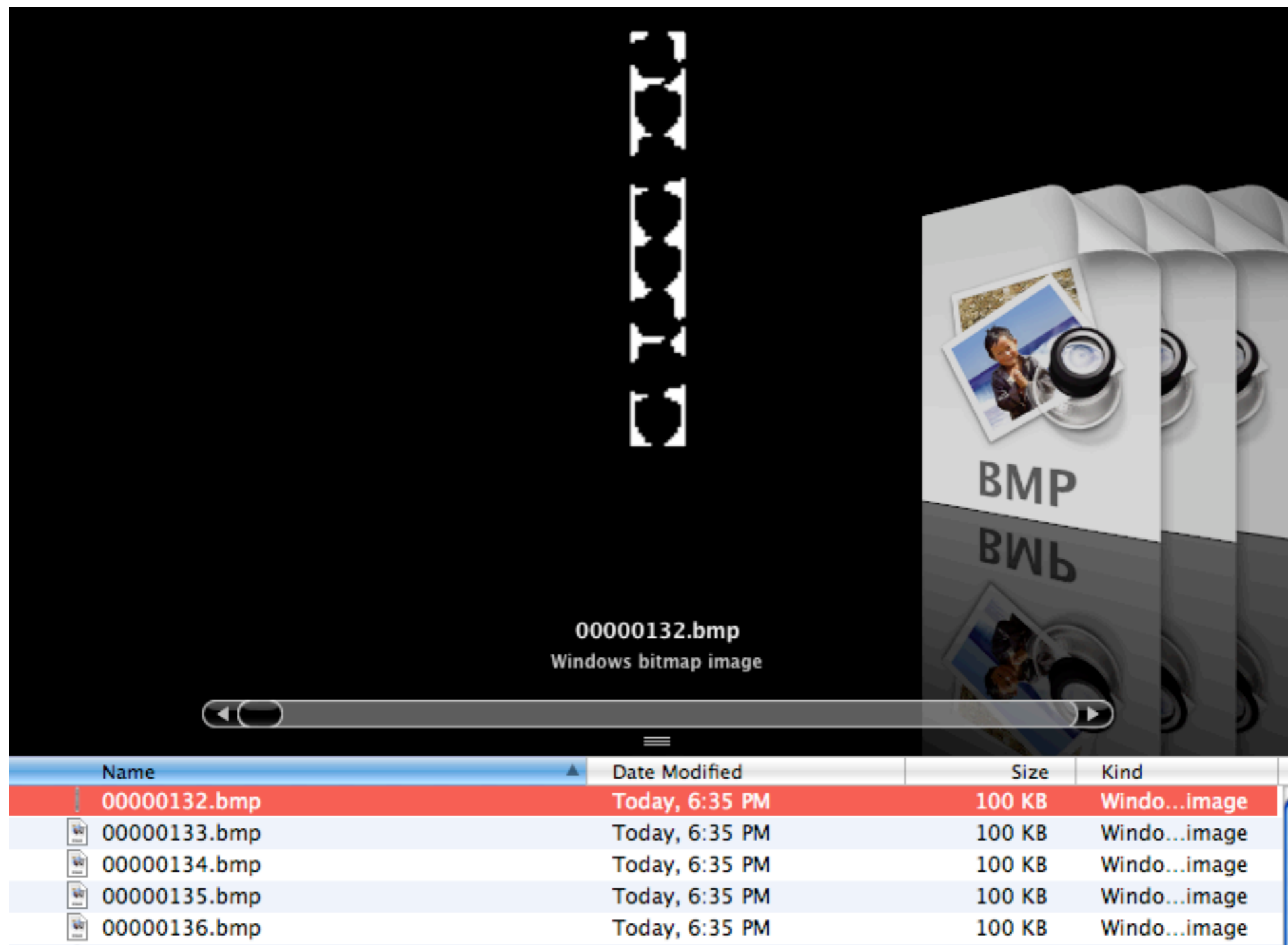
Opening target "/Users/simsong/image.dd"

The following files were carved:

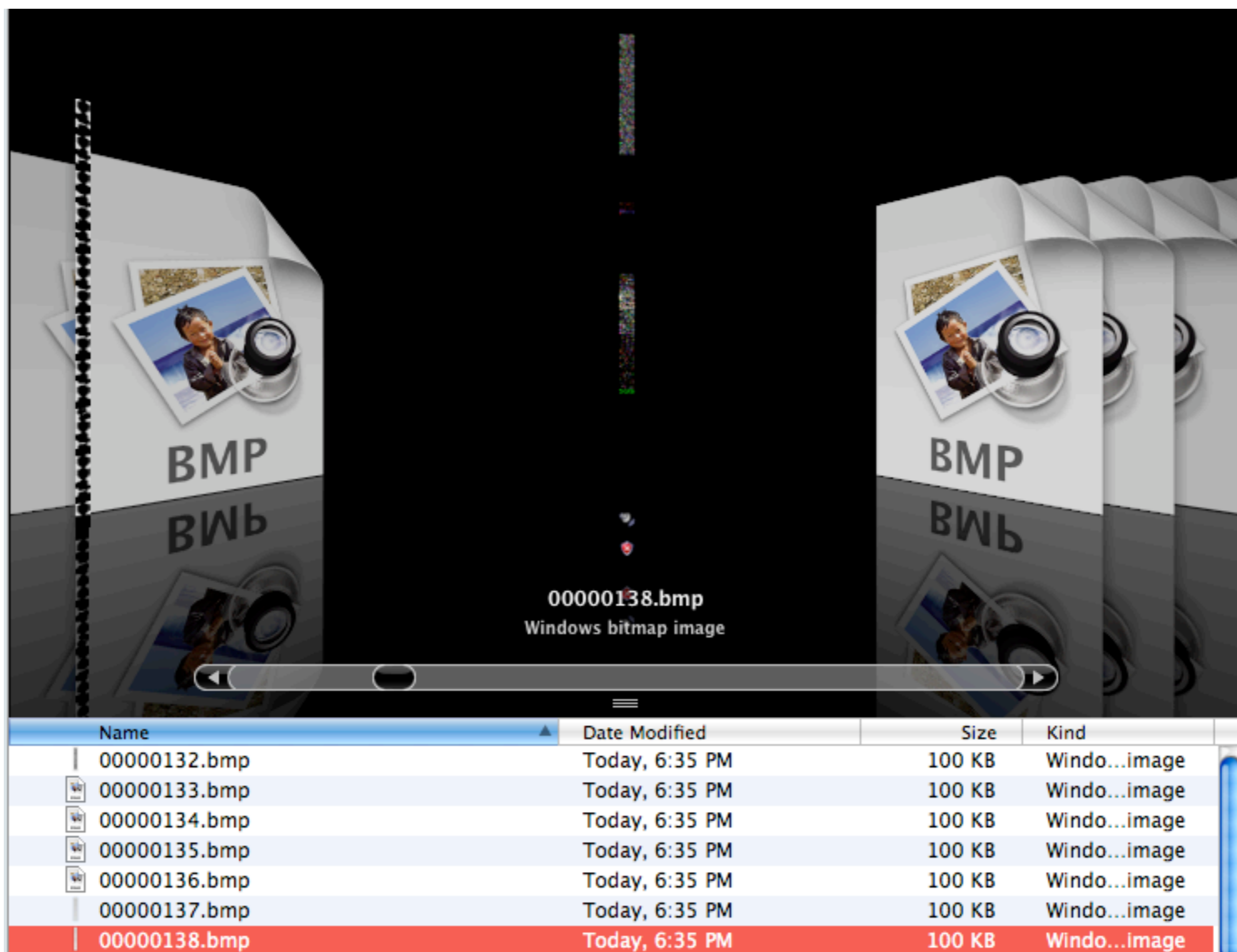
File	Start	Chop	Length	Extracted
From				
00000132.bmp	14597258	YES	100000	image.dd
00000010.gif	11806720	NO	1416	image.dd
00000009.gif	11804672	NO	2004	image.dd
00000012.gif	50000312	NO	44	image.dd
00000011.gif	42857896	NO	332	image.dd
00000015.gif	62047623	NO	53	image.dd
00000014.gif	60672512	NO	371	image.dd
00000013.gif	60672208	NO	61	image.dd
...				
00000017.gif	65827840	NO	477	image.dd
00000016.gif	65826816	NO	451	image.dd
00000000.gif	66591424	NO	3537	image.dd
00000113.jpg	74222592	NO	129055	image.dd
00000112.jpg	74219520	NO	2383	image.dd

And here's some of what we found...

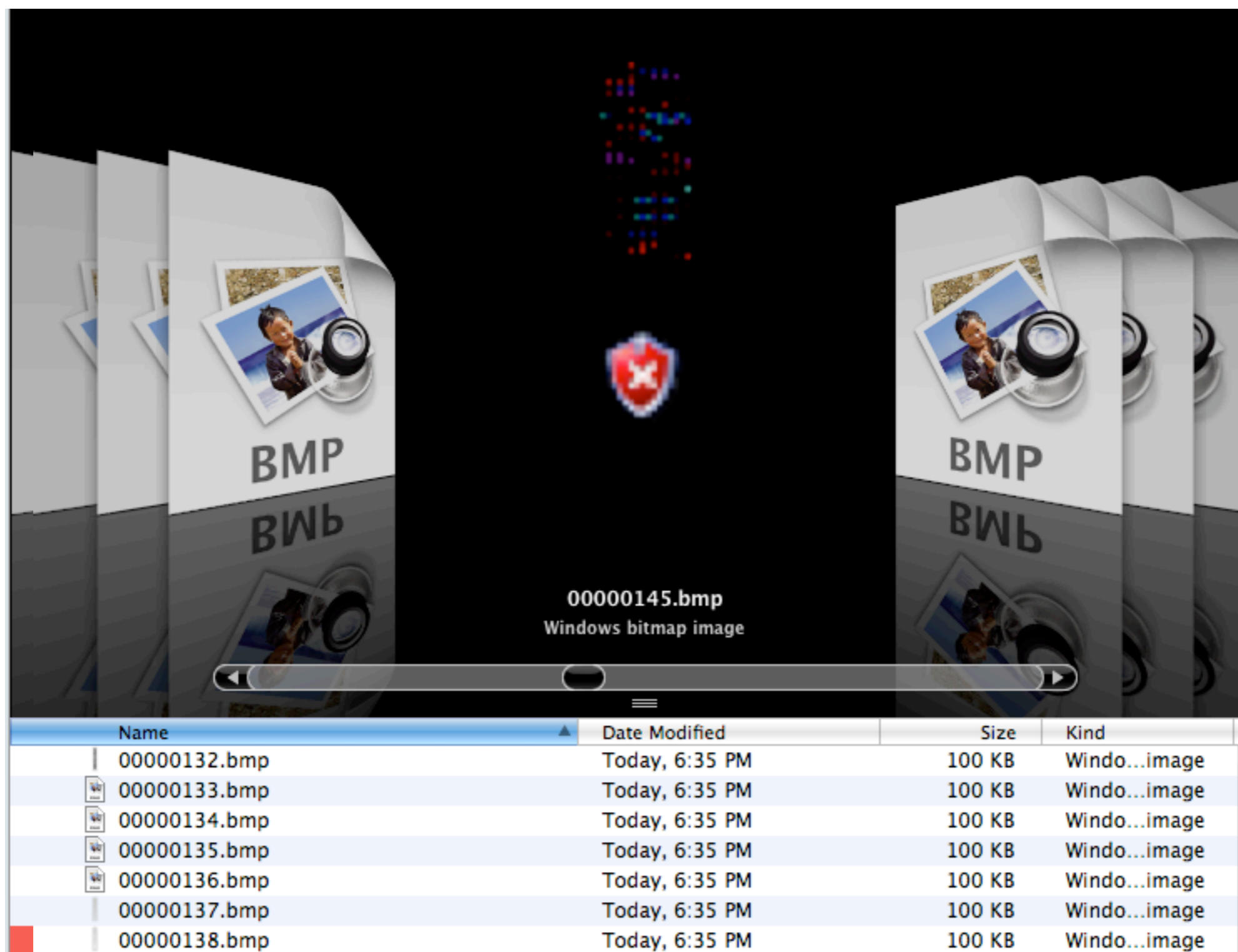
And here's some of what we found...



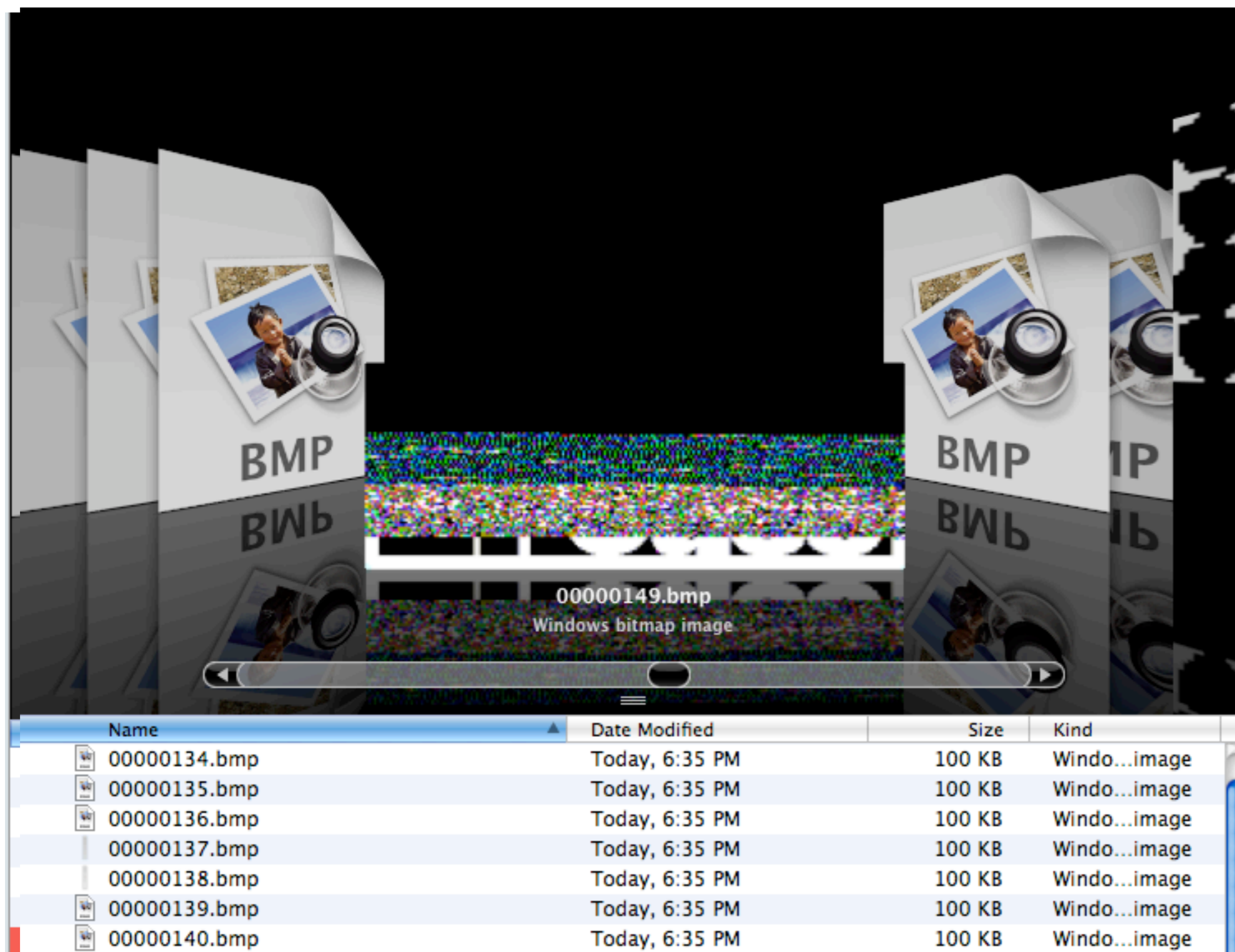
And here's some of what we found...



And here's some of what we found...



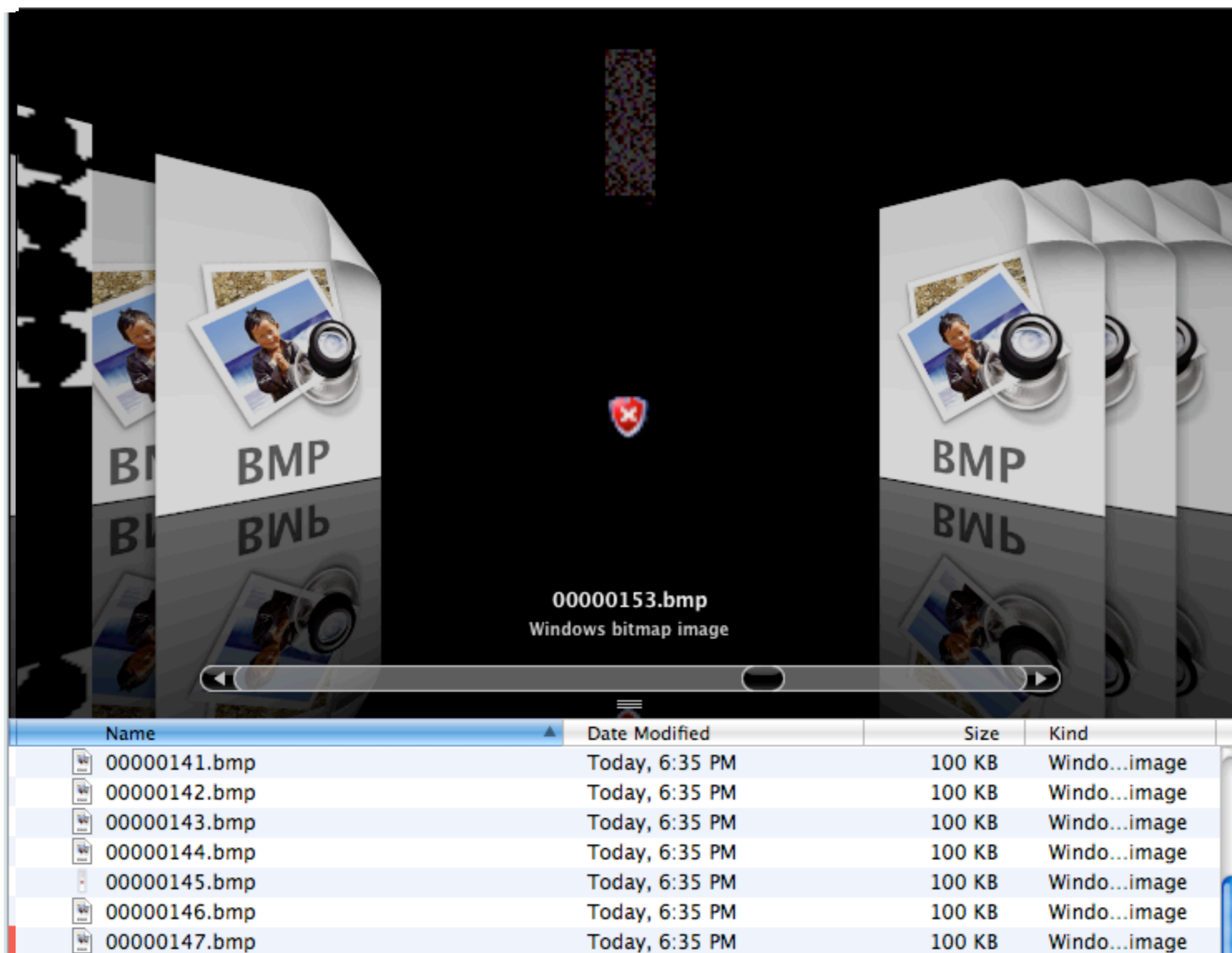
And here's some of what we found...



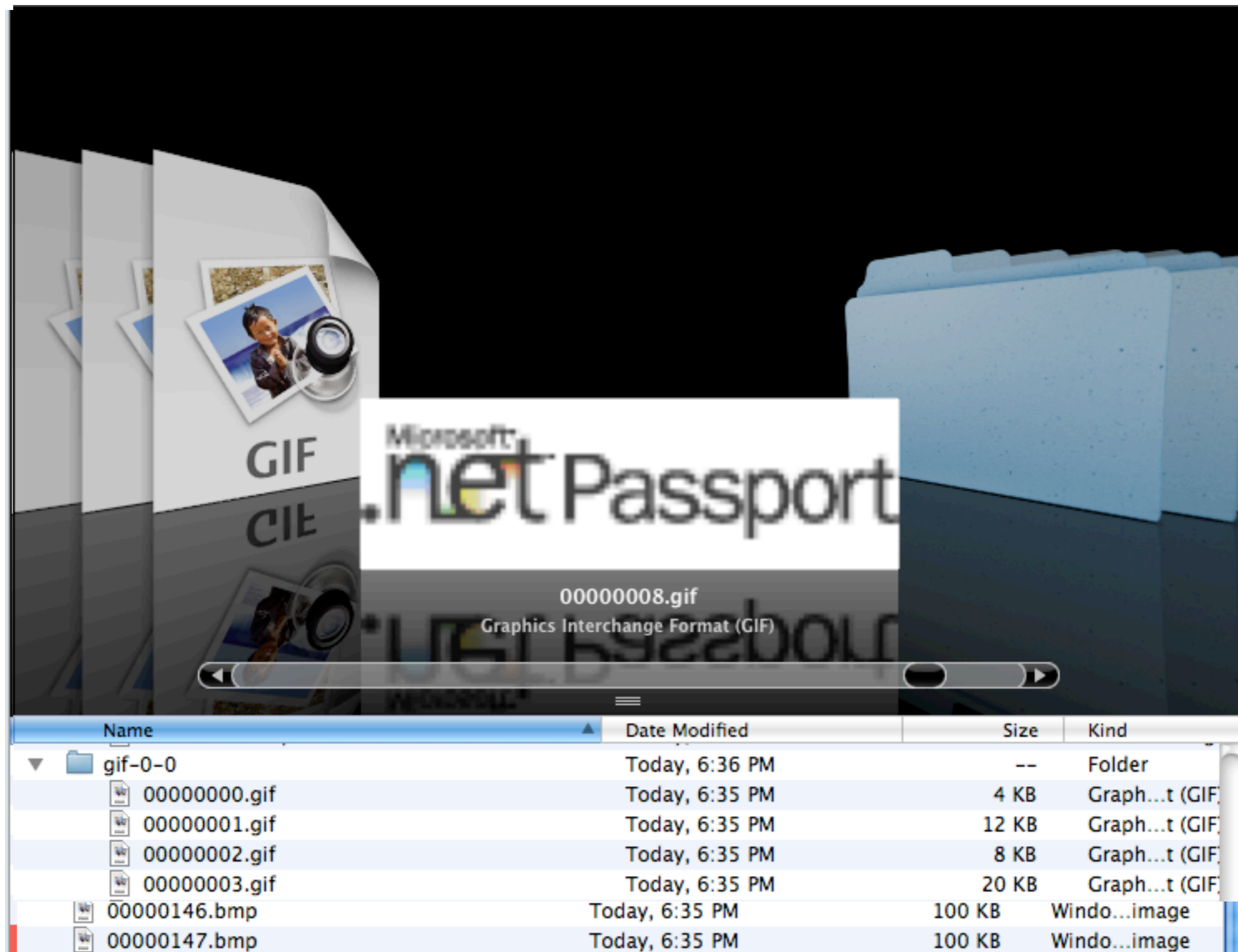
And here's some of what we found...



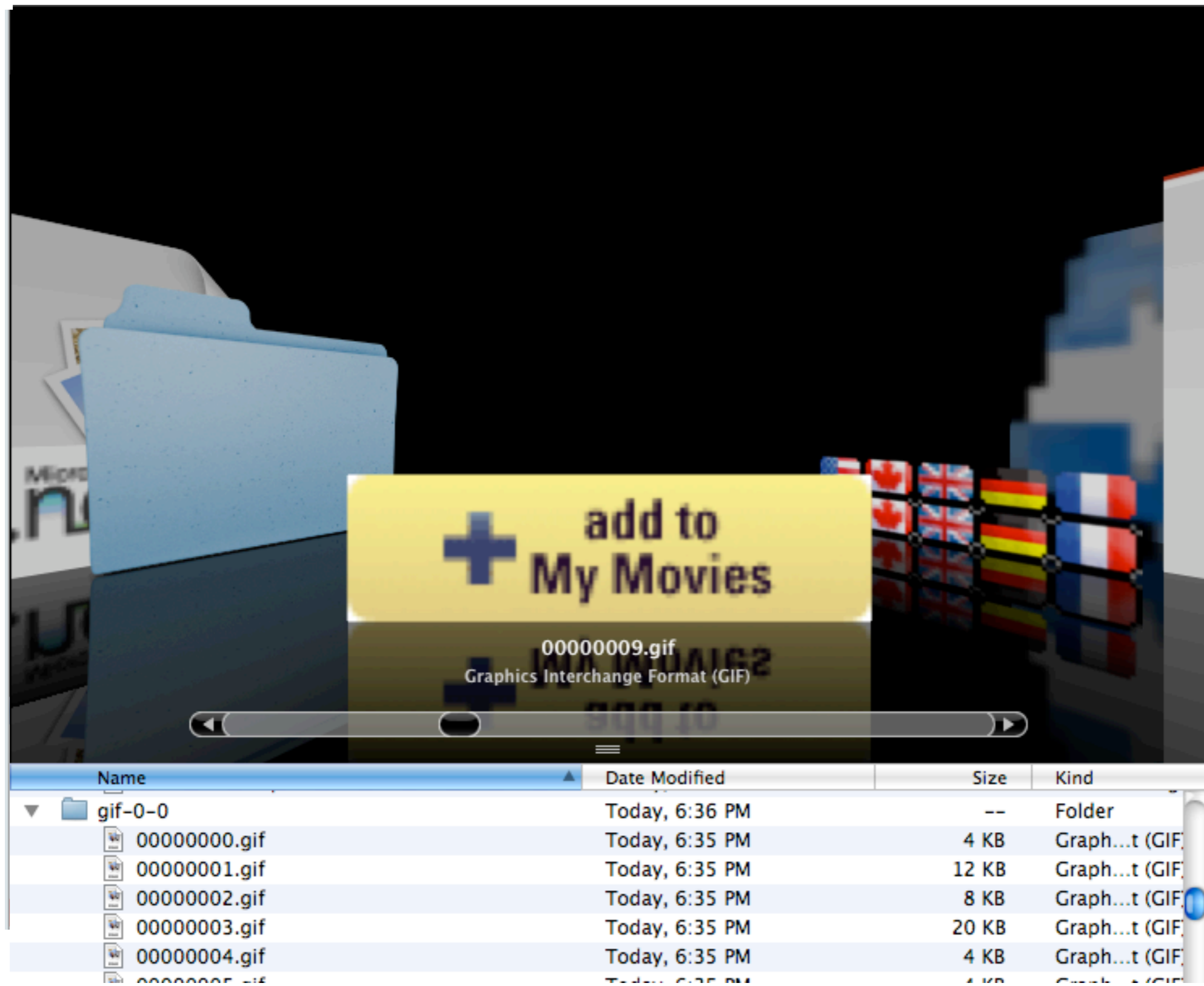
And here's some of what we found...



And here's some of what we found...

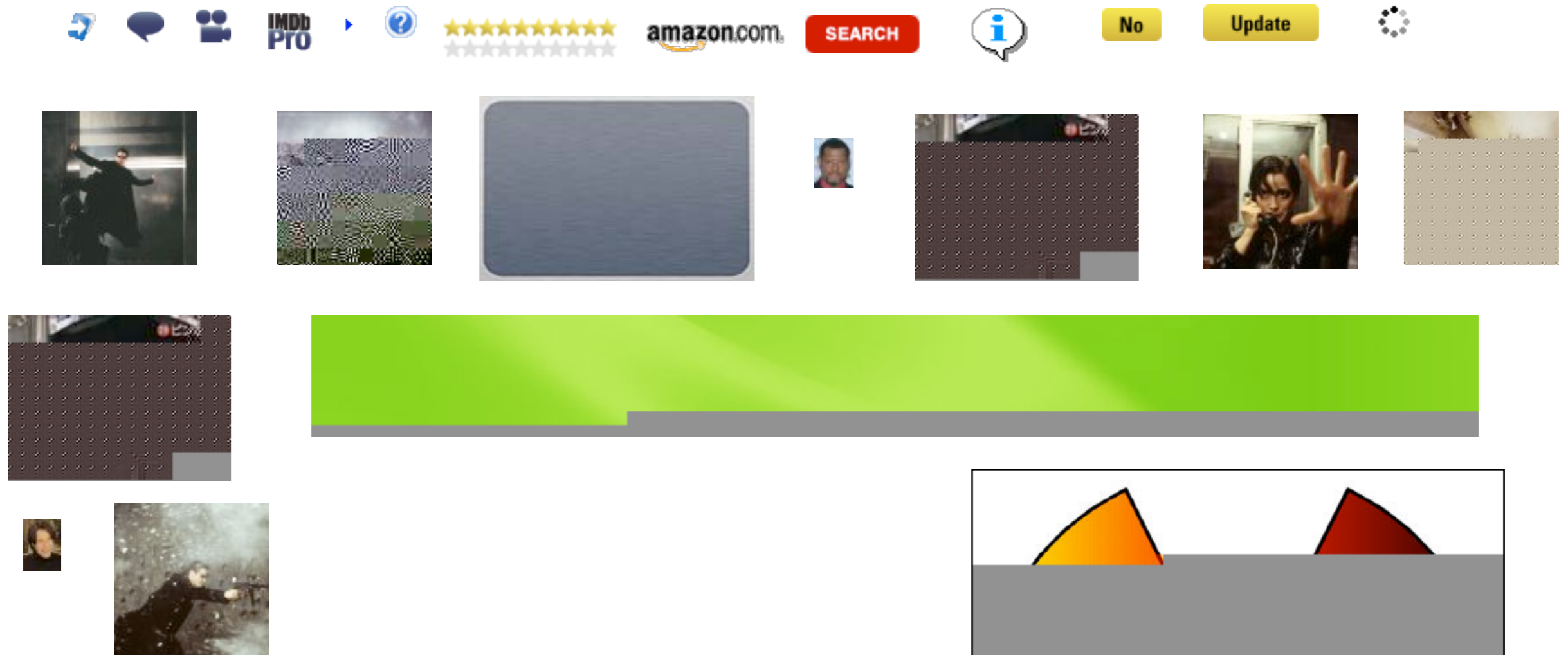


And here's some of what we found...



More photos...(actual size)

Note: It will be unusual to find one larger than 4K.



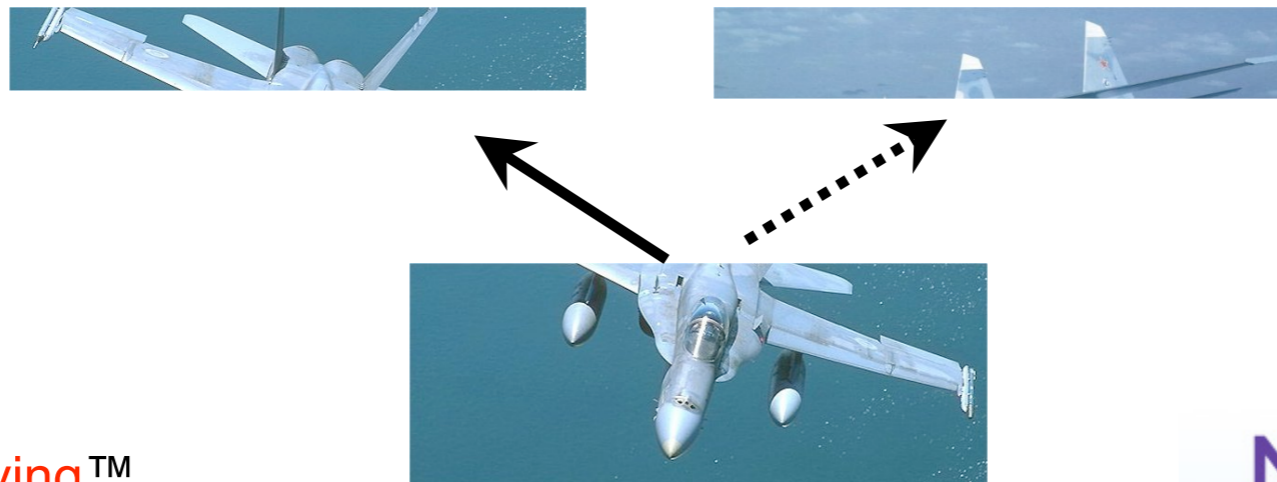
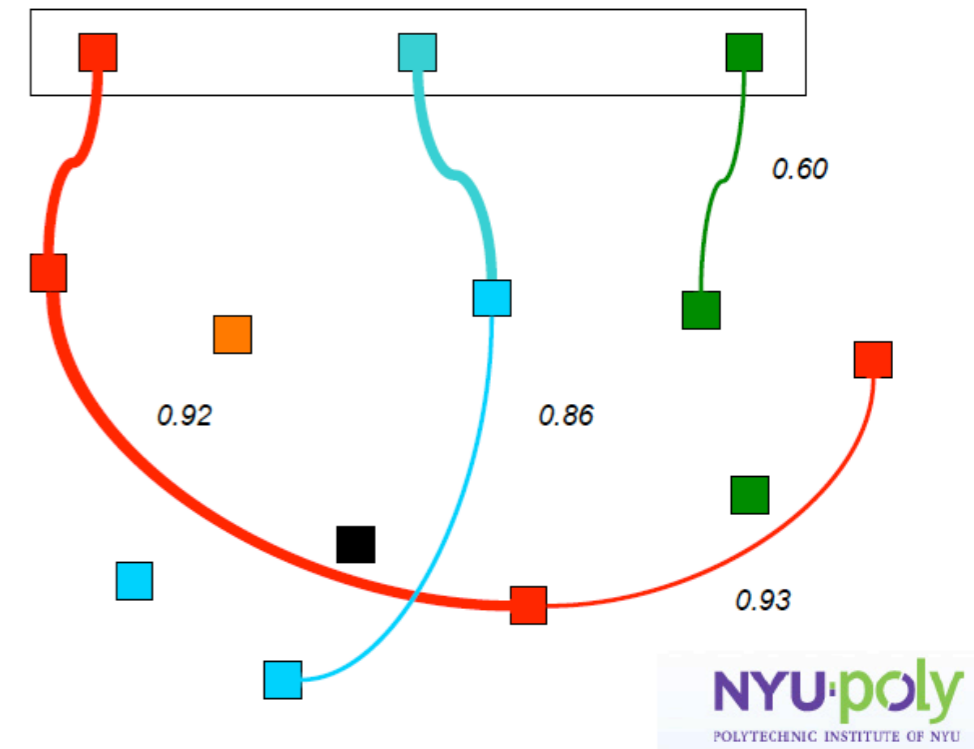
What's The Matrix doing in my computer???

Adroit constructs paths through sequential hypothesis testing (SHT)

Incorrect paths:

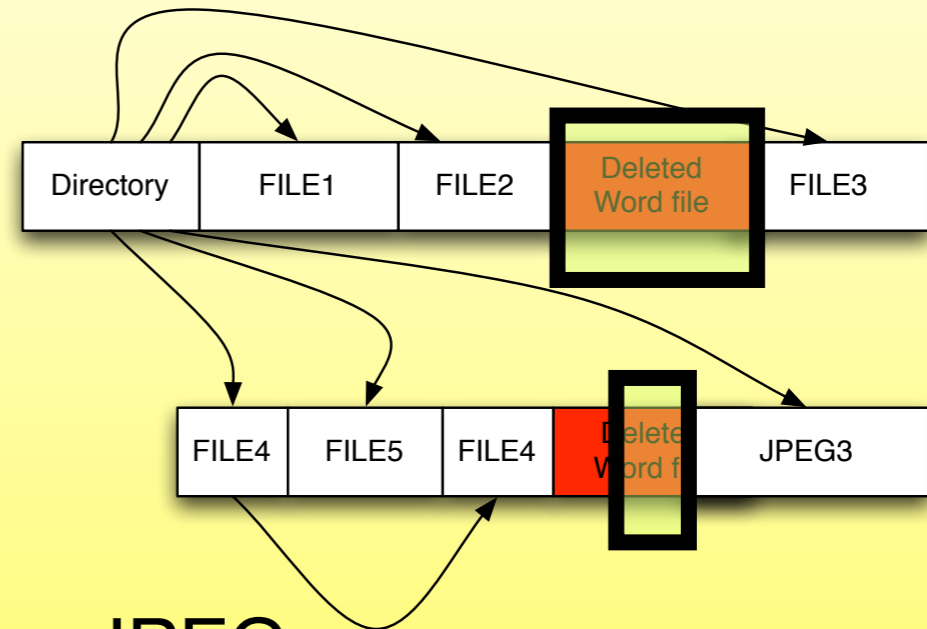
- Do not decompress
- Have sudden changes between scan lines.

<http://digital-assembly.com/>

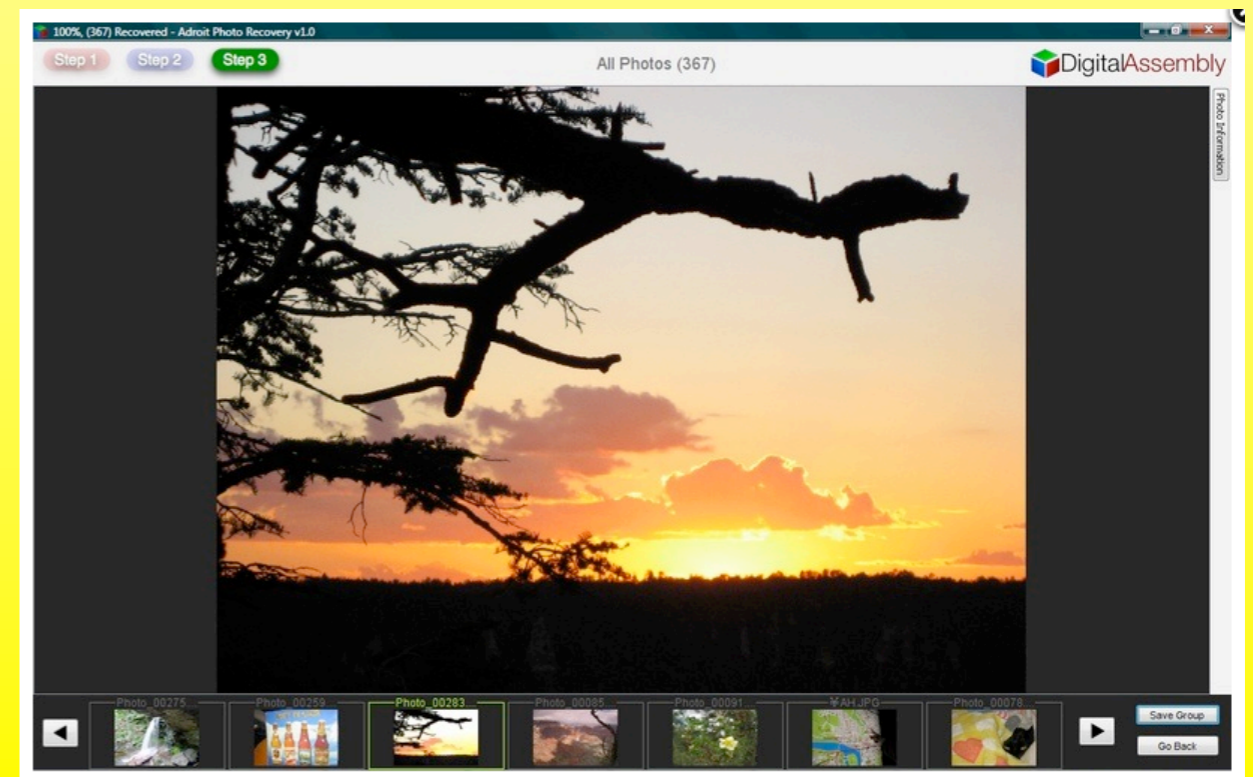


Summary: Carving

Carving is a powerful tool for finding *recognizable data*



Today, the best carving tools will recover JPEGs





Time for more coffee

This is an introductory tutorial!

Theory, Science and Tools

8:30 - 10:00 Introduction

- Introduction to Digital Forensics & The Law

10:00 - 10:30 Coffee

10:30 - 12:00 Data Analysis

- Unicode, File Formats & File Identification

12:00 - 1:30 Lunch

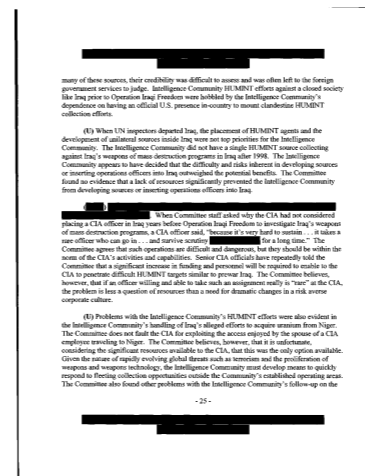
1:30 - 3:00 Disk Forensics

- Disk Imaging
- File Carving
- Sleuth Kit

3:00 - 3:30 Coffee

3:30 - 5:00 Big Finish

- Documents & Metadata
- Memory Forensics
- Anti-Forensics



[REDACTED]

The Department's attorney workforce is **more diverse than the U.S. legal workforce**: 38% female, compared to 30% in the U.S. legal labor pool, and 15% minority, compared to 12% in the labor pool. The Department's attorney workforce is about **as diverse as the federal government legal workforce**, whose attorneys are 38% female and 16% minority.

Hiring is serving to make the Department even more diverse: hires in 2001 were 40% female and 21% minority. [REDACTED]

[REDACTED] Honors Program hires in 2001 were 63% female, compared to 45% of the law school graduating class, and 30% minority, compared to 21% of the class of 2001.

Minorities [REDACTED] They comprise only 7% of (career) SES attorneys and 11% of supervisory Assistant U.S. Attorneys. Women constitute 31% of SESs and 37% of supervisory AUSAs. Among GS-15 attorneys in the Litigating Divisions, minorities comprise 11% of non-supervisors and 6% of supervisors, and women comprise 37% of non-supervisors and 33% of supervisors.

[REDACTED] In 2001, the attrition rate was 49% higher among minorities than whites. There was no difference in recent attrition between men and women.

[REDACTED] For example, the average minority GS attorney is currently 0.4 steps lower than the average white, and the average woman is 0.3 steps lower than the average man, controlling for seniority, grade, and component.

Based on these findings, we recommend that the Department take the following actions:

[REDACTED]

Is this document “authentic?”



Approaches for Data and Document Analysis:

Look for hidden data:

- Deleted information; previous versions
- GIDs embedded in Microsoft Word document

Look for characteristic data:

- Indicates authorship
- Indicates program used to create document.

Look for inconsistent data:

- Indicates possible tampering.

Intentional Metadata can be very important— but it is subject to manipulation

Office programs store "Document Properties" in each file:

The screenshot shows the 'dsl_contract.doc Properties' dialog box with the 'Statistics' tab selected. The 'General' tab is also visible. The 'Statistics' section contains a table with the following data:

Statistic name	Value
Characters (with spaces):	3116
Characters:	2667
Words:	493
Lines:	119
Paragraphs:	87
Pages:	2

Other fields in the 'General' tab include: Created: Tuesday, October 16, 2007 10:15 PM; Modified: Tuesday, October 16, 2007 10:18 PM; Printed: Tuesday, January 25, 2005 1:46 PM; Last saved by: Simson Garfinkel; Revision number: 3; Total editing time: 2 Minutes. The 'OK' button is highlighted.

The screenshot shows the 'dsl_contract.doc Properties' dialog box with the 'Summary' tab selected. The 'General' tab is also visible. The 'Summary' section contains the following fields:

- Title: SDSL Service
- Subject:
- Author: Mr. Joe Author
- Manager:
- Company: My Fancy DSL Company
- Category:
- Keywords:
- Comments:
- Hyperlink base:
- Template: Normal.dotm
- ☐ Save preview picture with this document

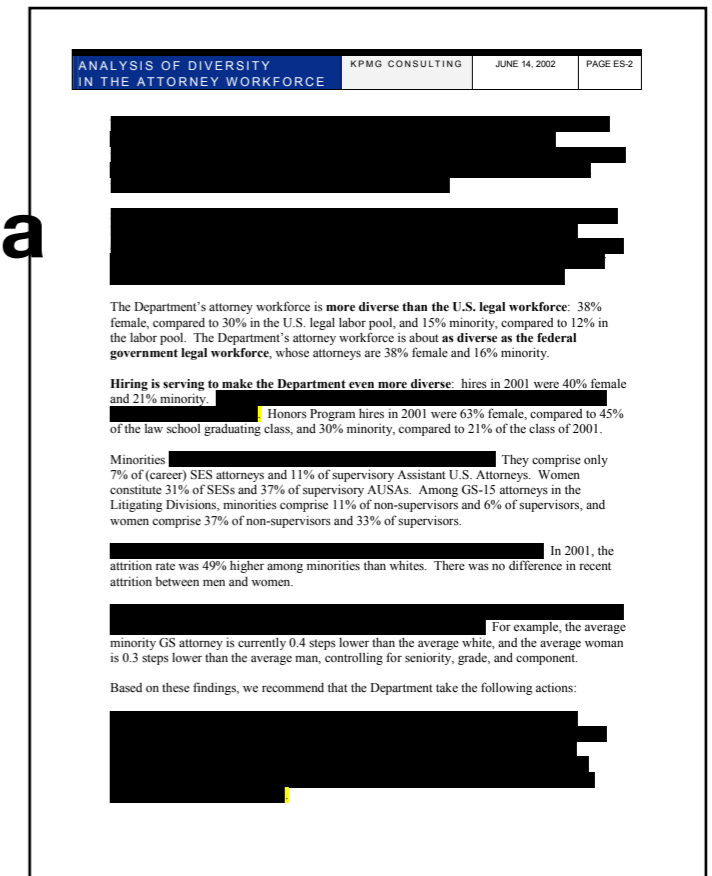
The 'OK' button is highlighted.

Privacy and Security violations result when improperly sanitized documents are released.

Adobe PDF files:

- The New York Times published a PDF file containing the names of Iranians who helped with the 1953 coup. (2000) (<http://cryptome.org/cia-iran.htm>)
- US DoJ published a PDF file “diversity report” with embarrassing redacted information. (2003) (<http://www.thememoryhole.org/feds/doj-attorney-diversity.htm>)
- Multinational Force-Iraq report (2005)

Most privacy violations come from **covered data**



Privacy and Security violations result when improperly sanitized documents are released.

Microsoft Word Files:

- SCO Word file revealed its anti-Linux legal strategy. (2004)
- Intelligence report by Blair Government was found to be plagiarized from a postgraduate student at the Monterey Institute of International Studies based on transaction log (2003)
<http://www.computerbytesman.com/privacy/blair.htm>

Most privacy violations come from **edit history**.

Why is there hidden data in documents?

Information that is written but never read.

Confusion between “covering data” and removing it.

Failure to implement “complete delete.”

The next few slides will look at:

- JPEG Exifs
- Data left in Acrobat Files
- Data left in Microsoft Word files

JPEG Exif: SD800.jpg

```
$ exif SD800.jpg
```

```
EXIF tags in 'sd800.JPG' ('Intel' byte order):
```

Tag	Value
Manufacturer	Canon
Model	Canon PowerShot SD800 IS
Orientation	top - left
x-Resolution	180.00
y-Resolution	180.00
Resolution Unit	Inch
Date and Time	2008:10:27 19:40:19
YCbCr Positioning	centered
Compression	JPEG compression
x-Resolution	180.00
y-Resolution	180.00
Resolution Unit	Inch
Exposure Time	1/60 sec.
FNumber	f/2.8
Exif Version	Exif Version 2.2
Date and Time (origi	2008:10:27 19:40:19
Date and Time (digit	2008:10:27 19:40:19
ComponentsConfigurat	Y Cb Cr -
Compressed Bits per	5.00
Shutter speed	5.91 EV (APEX: 7, 1/59 sec.)
Aperture	2.97 EV (f/2.8)
Exposure Bias	0.00 EV
MaxApertureValue	2.97 EV (f/2.8)
Metering Mode	Pattern
Flash	Flash fired, auto mode, red-eye reduction mode.
Focal Length	4.6 mm
Maker Note	2372 bytes unknown data
User Comment	
FlashPixVersion	FlashPix Version 1.0
Color Space	sRGB
PixelXDimension	640
PixelYDimension	480
Focal Plane x-Resolu	2844.44
Focal Plane y-Resolu	2840.24
Focal Plane Resoluti	Inch



Maker data: Additional stuff not in the standard

What do you think is going on with these?

Exif.Canon.0x0002 **Short** 4 2 4600 230 173

Exif.Canon.0x000d **Long** 148 507 411 0 0 0 293 576 96 0 0 0 0 293 480 195 0 0
4294967290 4294967235 4294967235 4294967290 0 0 0 7 10 4294967204 4294967265 4294967234 293 540
261 0 0 4294967265 4294967234 0 1 2 3072 3072 3072 3072 3072 4294964224 4294964224 4294964224
4294964224 4294964224 0 4294964224 4294967290 0 0 641 4294967174 192 0 96 0 0 0 243 1024 1024
4294967115 273 0 0 0 0 0 0 181 0 4294967115 273 4294967033 300 4 184 0 0 849 1052 1024 1280 0
4294967034 300 24 931 1823 1435 931 1 480 293 4294967204 603 261 0 0 506 5 0 0 0 835 5 0 0 0 0 1 0 890 0 0
0 506 5 4294965720 5 9 846 862 870 851 857 863 851 853 857 396 1536 460 354 105 276 83 0 0 3 3 23 7
971617331

Exif.Canon.0x0026 **Short** 48 96 4 9 9 640 480 1536 230 276 276 276 276 276 276 276 276
276 41 41 41 41 41 41 41 41 41 65260 0 276 65260 0 276 65260 0 276 65495 65495 65495 0 0 0 41 41 41 32 0 0
5

Exif.Canon.0x001d **Short** 16 32 1 0 2 2 2 2 0 0 0 0 0 0 0 0

Exif.Canon.0x0022 **Short** 208 416 2 1 16 8 24 16 640 480 65216 65304 320 232 2 8 384 0 0
0 0 3083 3340 3085 2828 0 1800 0 0 0 0 0 3072 3597 3855 3855 14 2828 10 0 0 0 7 2816 3840 4624 4883 4883
18 3328 2571 0 0 0 0 3339 4367 5651 6680 6170 0 3857 2829 2058 7 0 0 3852 5137 7192 8736 8226 6172 4372
3087 2315 7 0 2826 0 0 9246 10535 10025 7716 4888 3600 2571 1544 0 0 0 6933 10275 11564 11309 9000 5403
3601 2572 1800 0 0 4608 7702 10789 12078 11823 9514 5662 3858 2572 1801 0 0 4608 7702 10789 12078
11823 9514 5662 3858 2572 1801 0 0 4366 6933 10275 11564 11309 9000 5403 3601 12 0 0 0 4110 6163 30
10535 10025 7716 4888 3600 2571 8 0 0 0 5120 7192 8736 8226 6172 4372 3087 2315 7 0 0 0 0 5651 6680

Maker data: Additional stuff not in the standard

Exif.Canon.0x0002	Short	4 2 4600 230 173
• <i>'Focal Plane X Size'</i>		
Exif.Canon.0x000d	Long	148 507 411 0 0 0 293 576 96 0 0 0 0 293 480 195 0 0 4294967290 4294967235 4294967235 4294967290 0 0 0 7 10 ...
• <i>'Canon Lens Info' (an array of bytes)</i>		
Exif.Canon.0x0026	Short	48 96 4 9 9 640 480 1536 230 276 276 276 276 276 ...
• <i>'Canon AF Info2'</i>		
Exif.Canon.0x001d	Short	16 32 1 0 2 2 2 2 0 0 0 0 0 0 0 0
• <i>"My Colors"</i>		
Exif.Canon.0x0022	Short	208 416 2 1 16 8 24 16 640 480 65216 65304 320 ...
• <i>"Canon-1-0x0022"</i>		
Exif.Canon.0x0024	Short	78 156 35 0 640 480 1 1 16 0 0 0 0 0 0 0 0 0 0 0 0...
• <i>"Face Detect1"</i>		
Exif.Canon.0x0025	Byte	14 14 35 0 0 0 0 0 0 0 0 0 0 0 0
• <i>"Face Detect 2"</i>		
Exif.Canon.0x0028	Byte	16 56 108 144 93 46 211 93 33 211 24 138 161 98 113 83 226
• <i>"Canon-1-0x0028" - 16 bytes</i>		

Privacy Sensitive Web Hosting:

Automatically remove EXIF with mod_rewrite:

Put this in .htaccess:

```
# run jpegs through privacy filter
RewriteRule ^(*[.]je?pg)$ strip.cgi/$1^ [L,NS,PT]
```

Create this program ~/strip.cgi

```
#!/bin/sh
JPEG_FILENAME=${DOCUMENT_ROOT}${PATH_INFO/^/}
echo Content-type: image/jpeg
echo ""
convert -quiet $JPEG_FILENAME -strip -
```

This relies on ImageMagick to remove the EXIF with the “-strip” option.

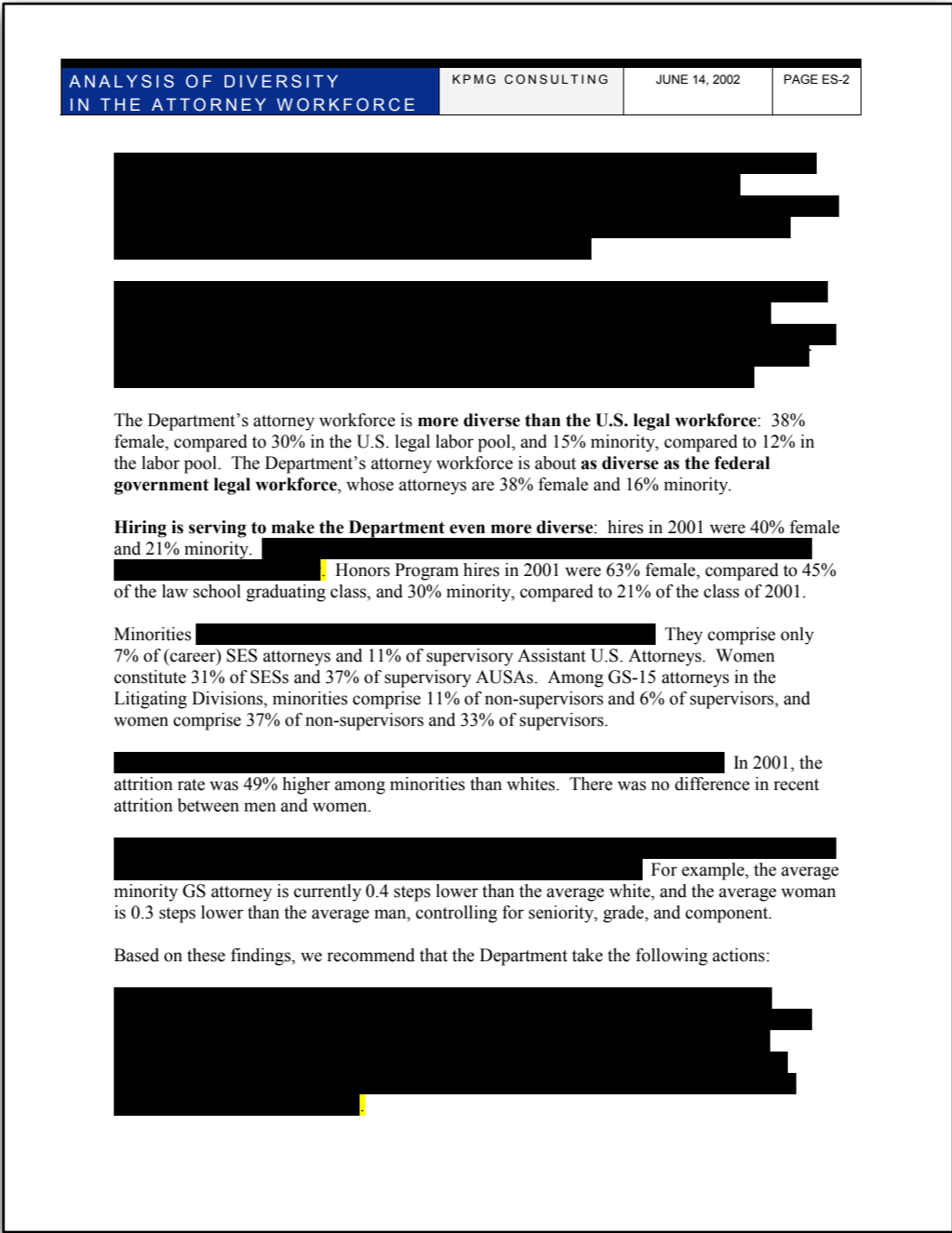
exif tools

<http://search.cpan.org/dist/Image-ExifTool/>

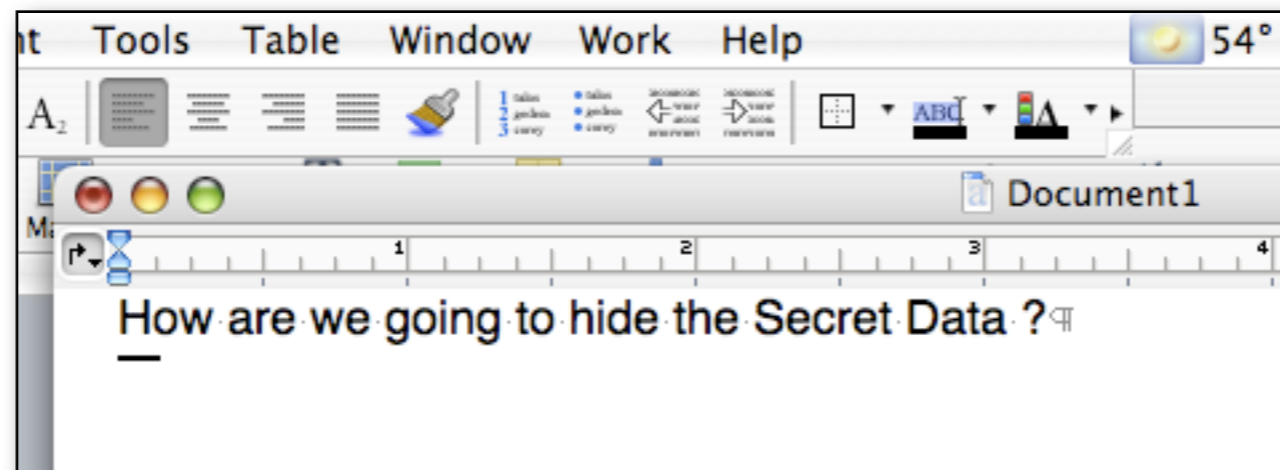
<http://gvsoft.homedns.org/exif/makernote-canon-type1.html>

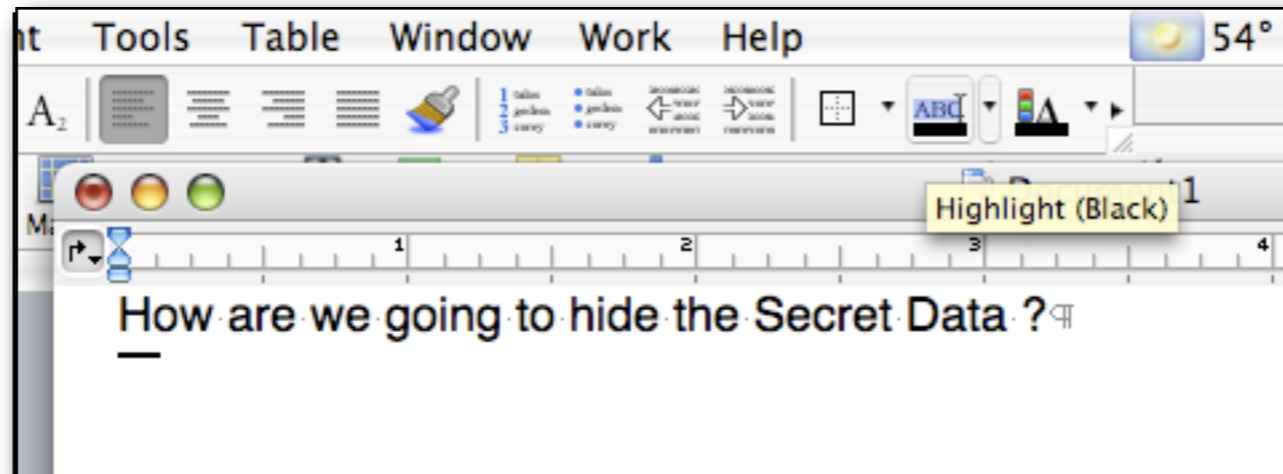
<http://www.exiv2.org/>

Most Acrobat leakage is a result of Microsoft Word.

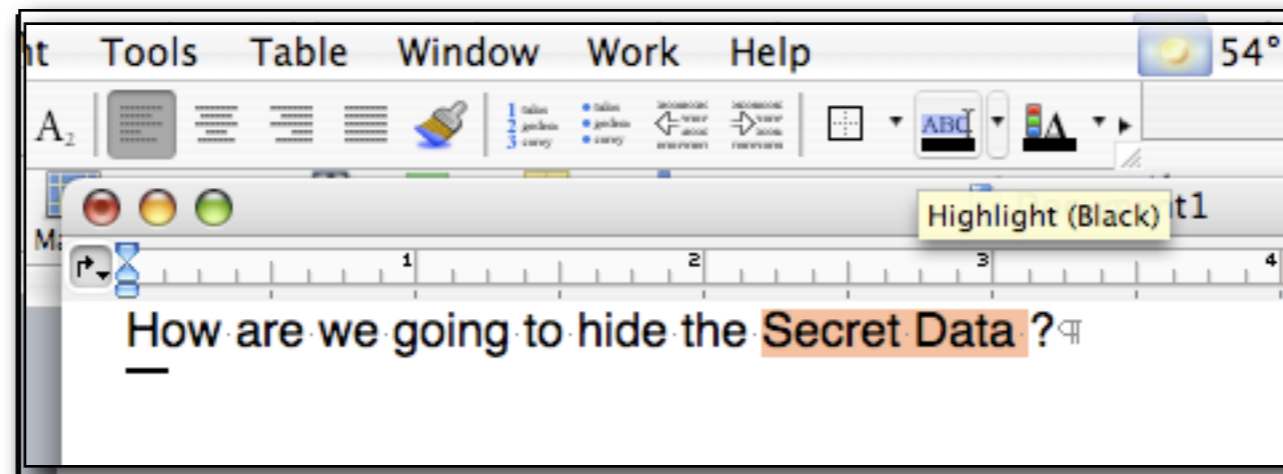


Microsoft Word encourages people to use the highlight feature to eradicate data.

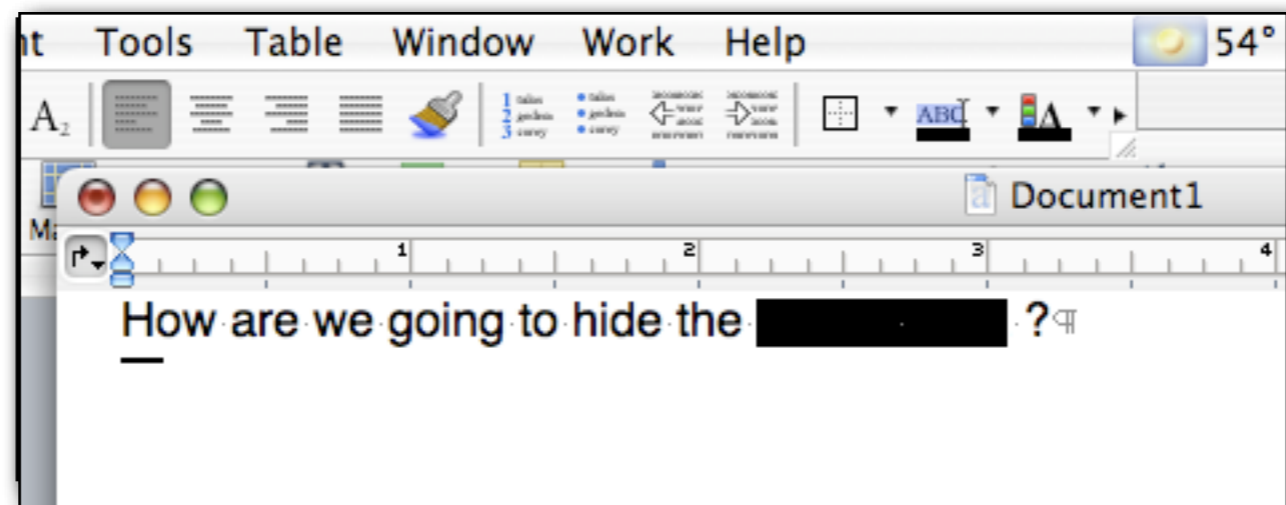




Microsoft Word encourages people to use the highlight feature to eradicate data.



Microsoft Word encourages people to use the highlight feature to eradicate data.



When Microsoft Word generates the PDF file, “Secret Data” is covered with the black box



Tools for recovering hidden data in Acrobat files:

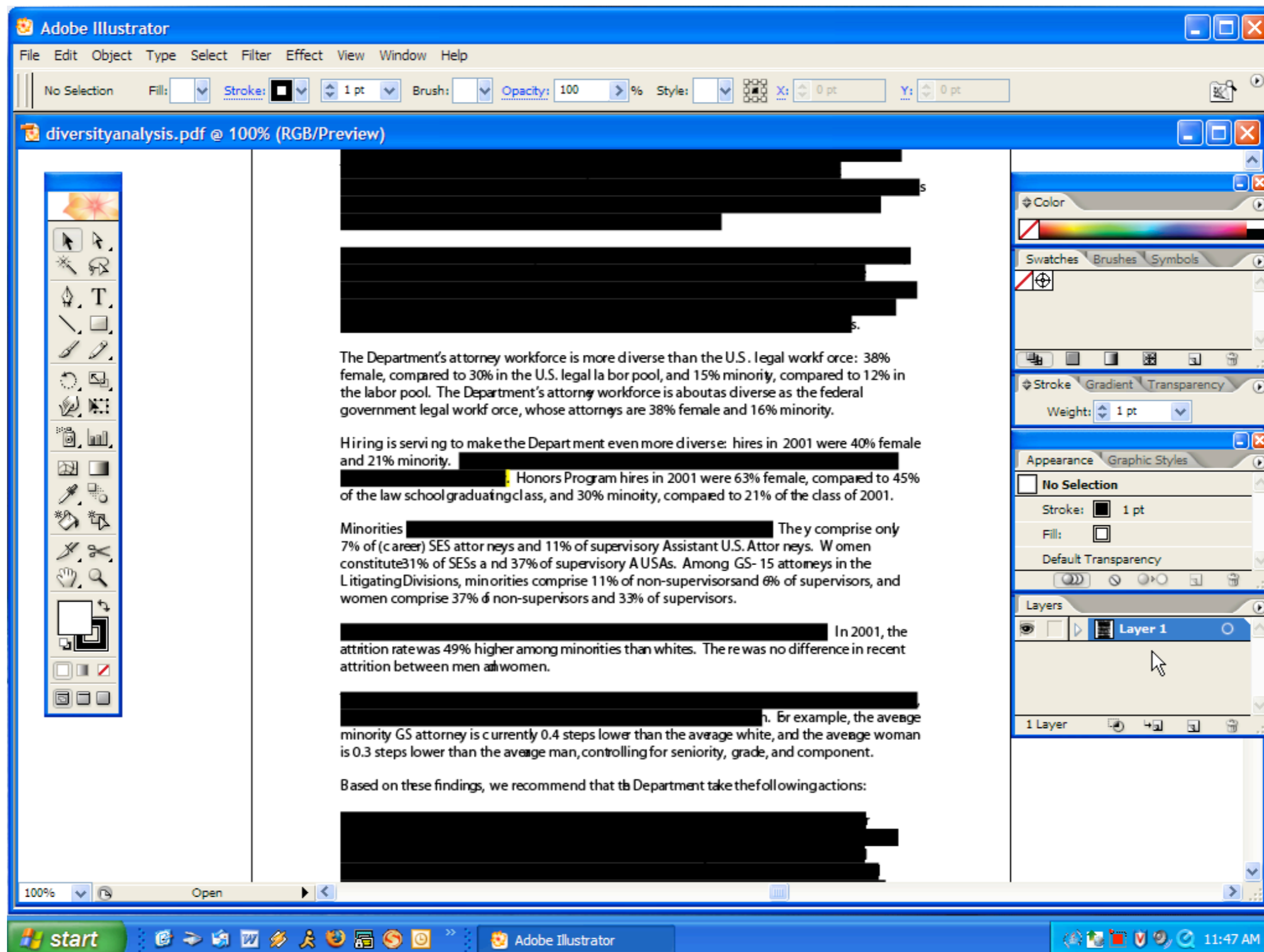
Adobe Illustrator

- Move the boxes
- Turn the boxes yellow

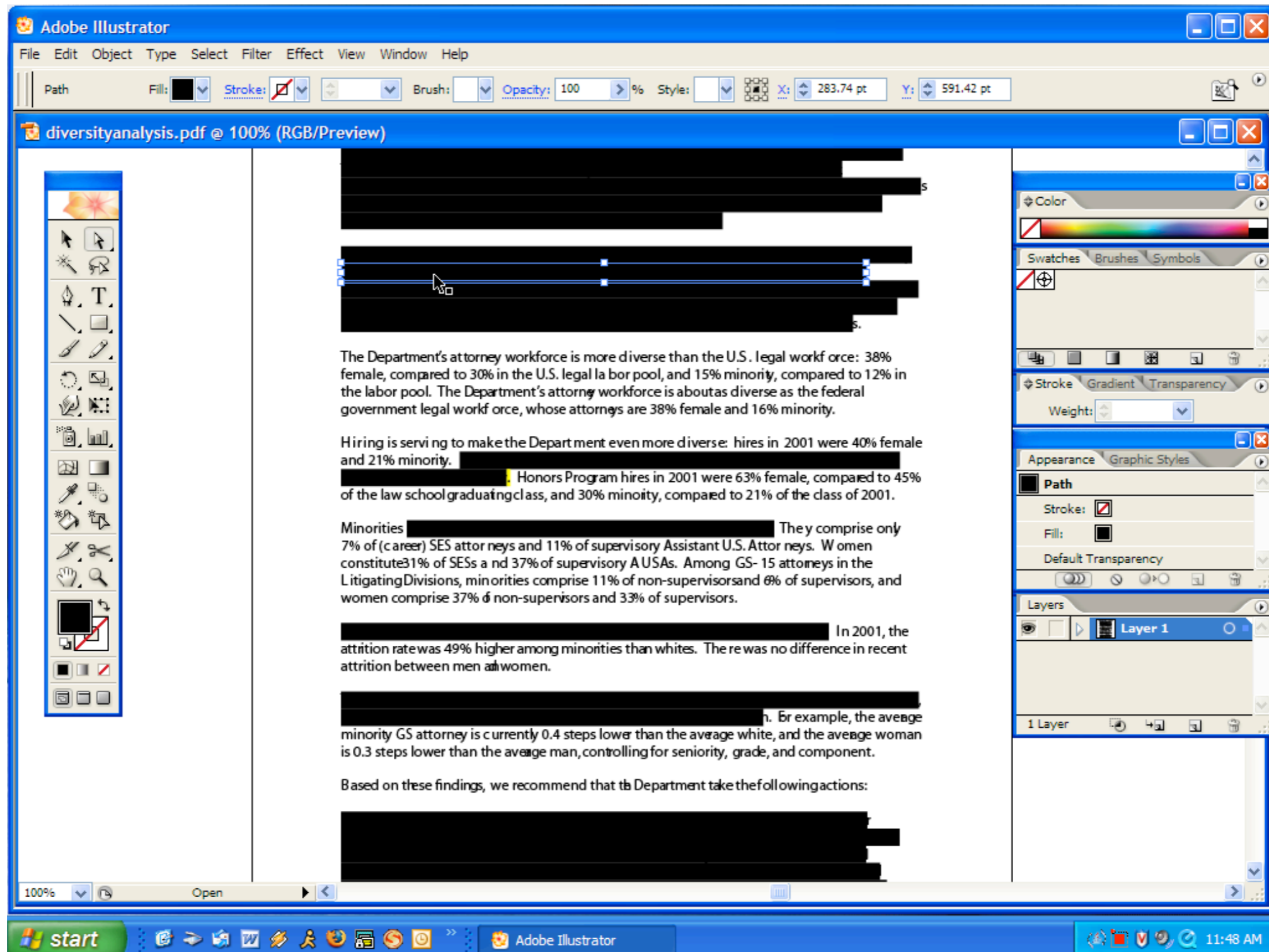
Adobe Acrobat Reader

- Select and copy the text

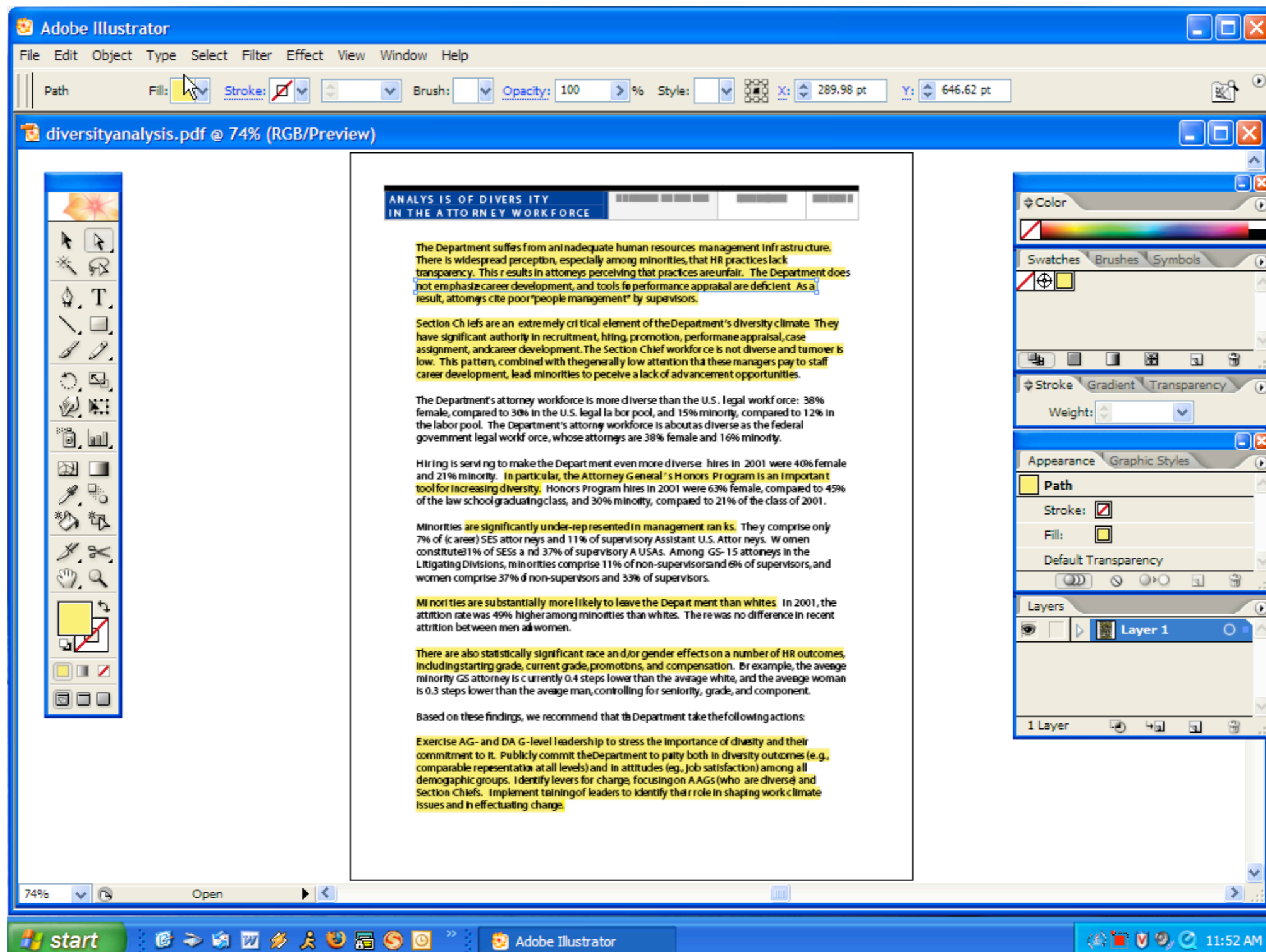
Adobe's Illustrator can read and edit PDF files.



Select each “block box.”



Change the color of the box to yellow.



Behold the “redacted” data.

The Department suffers from an inadequate human resources management infrastructure.

There is widespread perception, especially among minorities, that HR practices lack transparency. This results in attorneys perceiving that practices are unfair. The Department does not emphasize career development, and tools for performance appraisal are deficient. As a result, attorneys cite poor “people management” by supervisors.

Section Chiefs are an extremely critical element of the Department’s diversity climate. They have significant authority in recruitment, hiring, promotion, performance appraisal, case assignment, and career development. The Section Chief workforce is not diverse and turnover is low. This pattern, combined with the generally low attention that these managers pay to staff career development, lead minorities to perceive a lack of advancement opportunities.

The Department’s attorney workforce is more diverse than the U.S. legal workforce: 38% female, compared to 30% in the U.S. legal labor pool, and 15% minority, compared to 12% in the labor pool. The Department’s attorney workforce is about as diverse as the federal government legal workforce, whose attorneys are 38% female and 16% minority.

Hiring is serving to make the Department even more diverse: hires in 2001 were 40% female and 21% minority. In particular, the Attorney General’s Honors Program is an important tool for increasing diversity. Honors Program hires in 2001 were 63% female, compared to 45% of the law school graduating class, and 30% minority, compared to 21% of the class of 2001.

Minorities are significantly under-represented in management ranks. They comprise only 7% of (career) SES attorneys and 11% of supervisory Assistant U.S. Attorneys. Women constitute 31% of SESs and 37% of supervisory AUSAs. Among GS-15 attorneys in the Litigating Divisions, minorities comprise 11% of non-supervisors and 6% of supervisors, and women comprise 37% of non-supervisors and 33% of supervisors.

Minorities are substantially more likely to leave the Department than whites. In 2001, the attrition rate was 49% higher among minorities than whites. There was no difference in recent attrition between men and women.

There are also statistically significant race and/or gender effects on a number of HR outcomes, including starting grade, current grade, promotions, and compensation. For example, the average minority GS attorney is currently 0.4 steps lower than the average white, and the average woman is 0.3 steps lower than the average man, controlling for seniority, grade, and component.

Based on these findings, we recommend that the Department take the following actions:

Exercise AG- and DAG-level leadership to stress the importance of diversity and their commitment to it. Publicly commit the Department to parity both in diversity outcomes (e.g., comparable representation at all levels) and in attitudes (e.g., job satisfaction) among all demographic groups. Identify levers for change, focusing on AAGs (who are diverse) and Section Chiefs. Implement training of leaders to identify their role in shaping work climate issues and in effectuating change.

Other tools for working with PDFs:

pdfinfo — Reports metadata from a PDF file

pdftimages — Extracts all of the images from a PDF file

Both are part of the xpdf package, <http://www.foolabs.com/xpdf/>

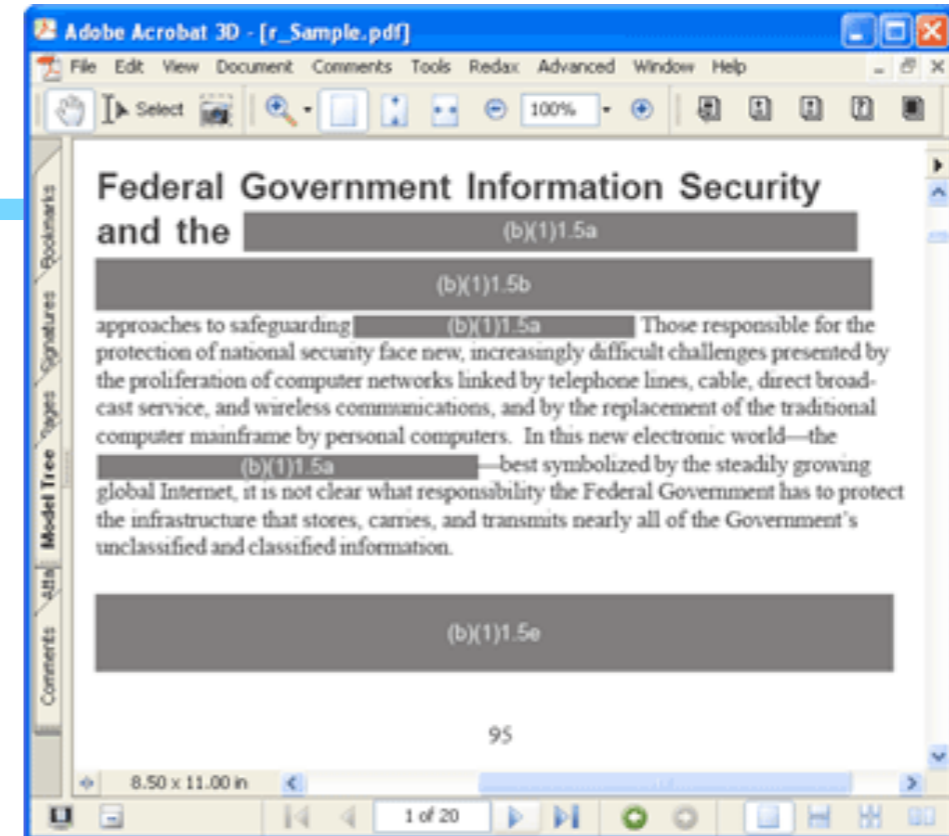
Tools for redacting PDFs:

Redax (Appligent)

ISIToolBox Professional (Image Solutions)

Rapidredact (OnStream Systems)

Acrobat 9 Professional

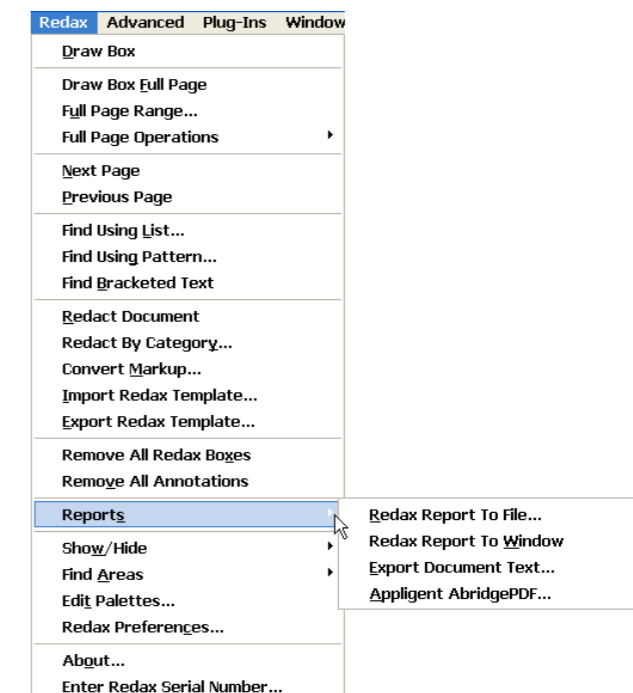


Survey of tools:

- "Redacting PDF files: a survey of tools," by Duff Johnson, http://www.acrobatusers.com/articles/2006/06/redaction_tools/redaction_tools.php

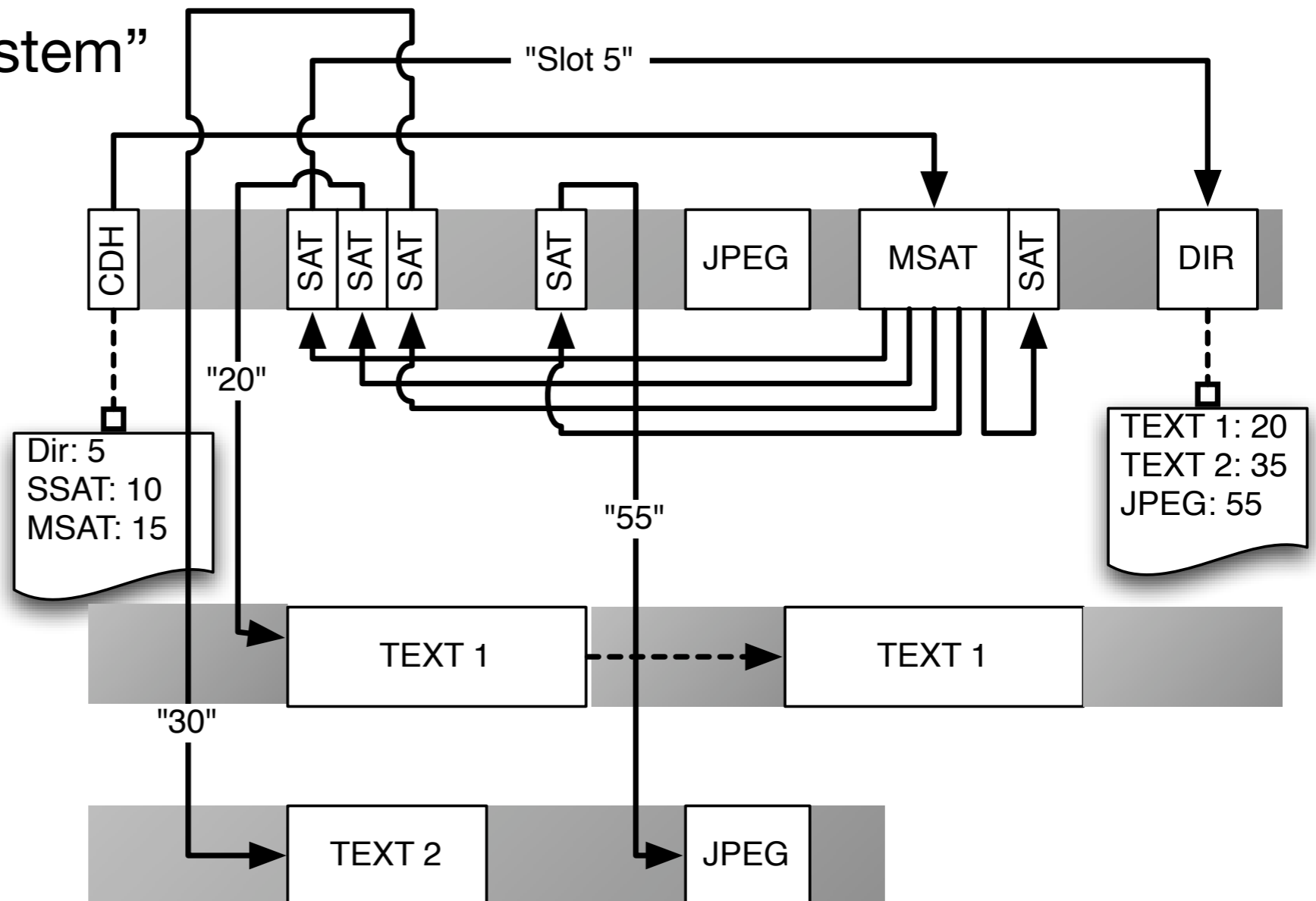
For further reading:

- "Redacting with Confidence," National Security Agency <http://www.fas.org/sgp/othergov/dod/nsa-redact.pdf>



Data can be left in a Word document in unallocated sectors.

Microsoft Word implements a “file system” inside every file.



Data left behind in Microsoft Word Files is a big problem

Byers, Simon, “Scalable Exploitation of, and Responses to,
Information Leakage Through Hidden Data in Published Documents,”
AT&T Research, 2003

http://www.user-agent.org/word_docs.pdf

1000 documents download per hour from cable modem with no parallelization

- 50% documents have 10-50 hidden words
- 10% have more than 500 hidden words



Tools for recovering hidden Word data:

Unix strings(1) command reveals:

- Deleted text
- Names and/or usernames of author and editors
- Paths where document was saved
- GUID of system on which it was saved

Note: Text may be UTF16 (remove NULLs or use more intelligent processing)

Text Extraction Tools:

- Antiword (<http://www.winfield.demon.nl/>)
- catdoc
- wvText

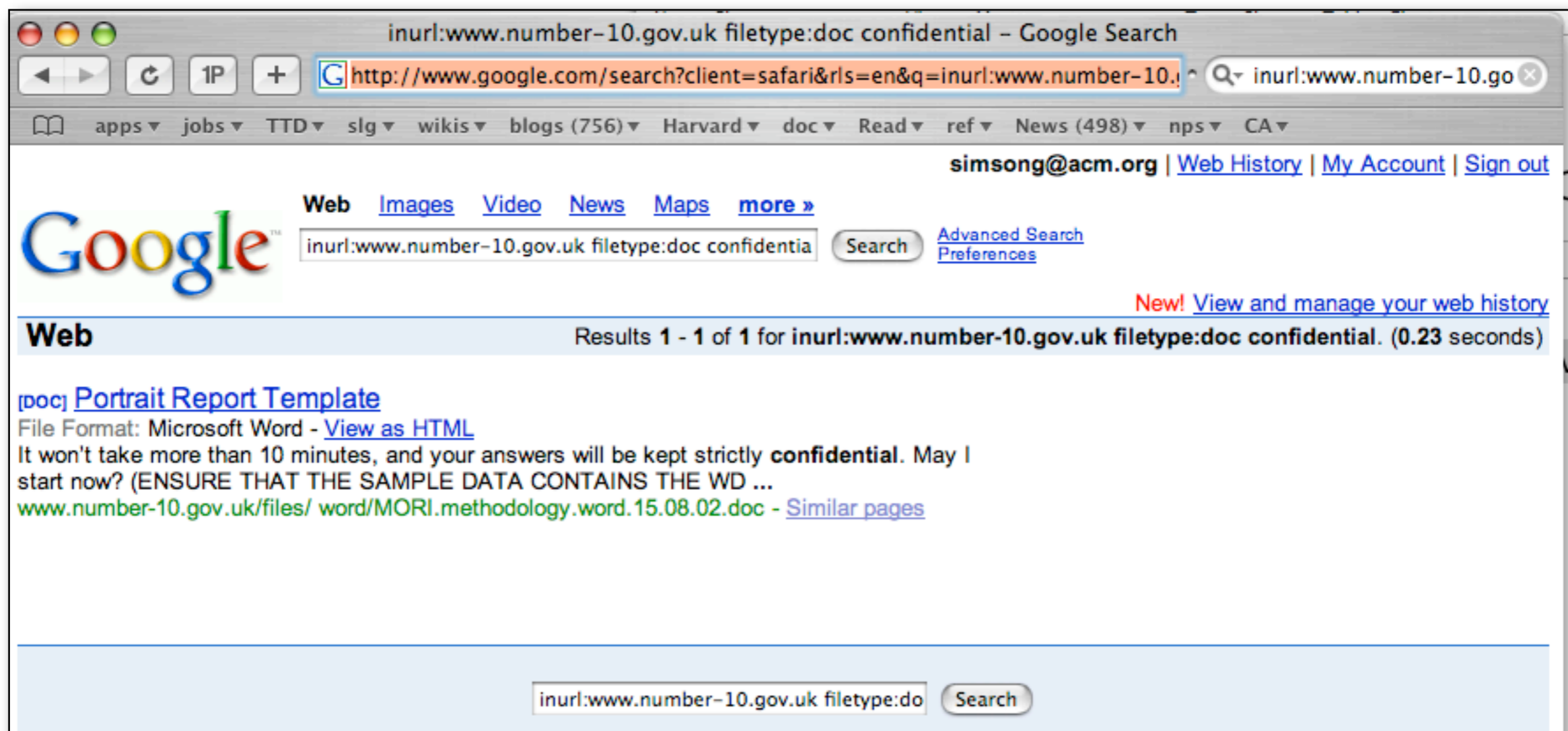
Metadata Extraction Tools:

- NZ National Library: <http://meta-extractor.sourceforge.net/>

Tools for finding Microsoft Word files


Use Google!

- `inurl:www.number-10.gov.uk filetype:doc confidential`



Forensic Wiki page:

http://www.forensicswiki.org/wiki/Tools:Document_Metadata_Extraction



[page](#) [discussion](#) [view source](#) [history](#)

[Log in / create account](#)

Document Metadata Extraction

(Redirected from [Tools:Document Metadata Extraction](#))

Here are tools that will extract metadata from document files.

Contents [\[hide\]](#)

- 1 Office Files
- 2 PDF Files
- 3 Images
- 4 General

Office Files

antiword
<http://www.winfield.demon.nl/> [↗](#)

catdoc
<http://www.45.free.net/~vitus/software/catdoc/> [↗](#)

laola
<http://user.cs.tu-berlin.de/~schwartz/pmh/index.html> [↗](#)

word2x
<http://word2x.sourceforge.net/> [↗](#)

wwWare
<http://wwware.sourceforge.net/> [↗](#)
Extracts metadata from various [Microsoft Word](#) files (**doc**). Can also convert doc files to other formats such as HTML or plain text.

navigation:

- [Main Page](#)
- [Categories](#)

about forensicswiki.org:

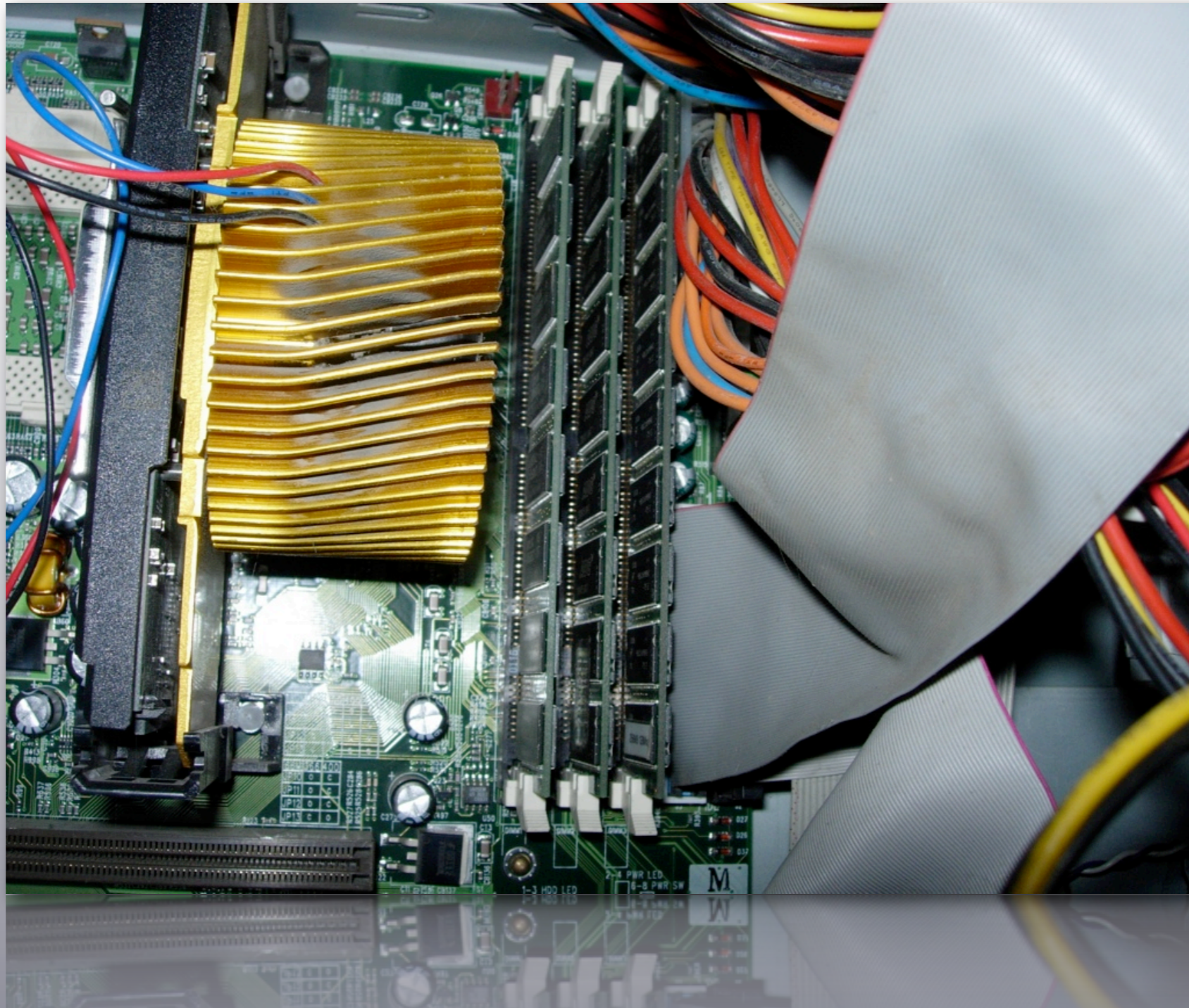
- [Recent changes](#)
- [Random page](#)
- [sitesupport](#)

search

toolbox

- [What links here](#)
- [Related changes](#)
- [Special pages](#)
- [Printable version](#)
- [Permanent link](#)

In Summary...

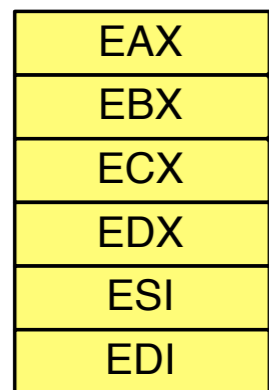


Memory Forensics

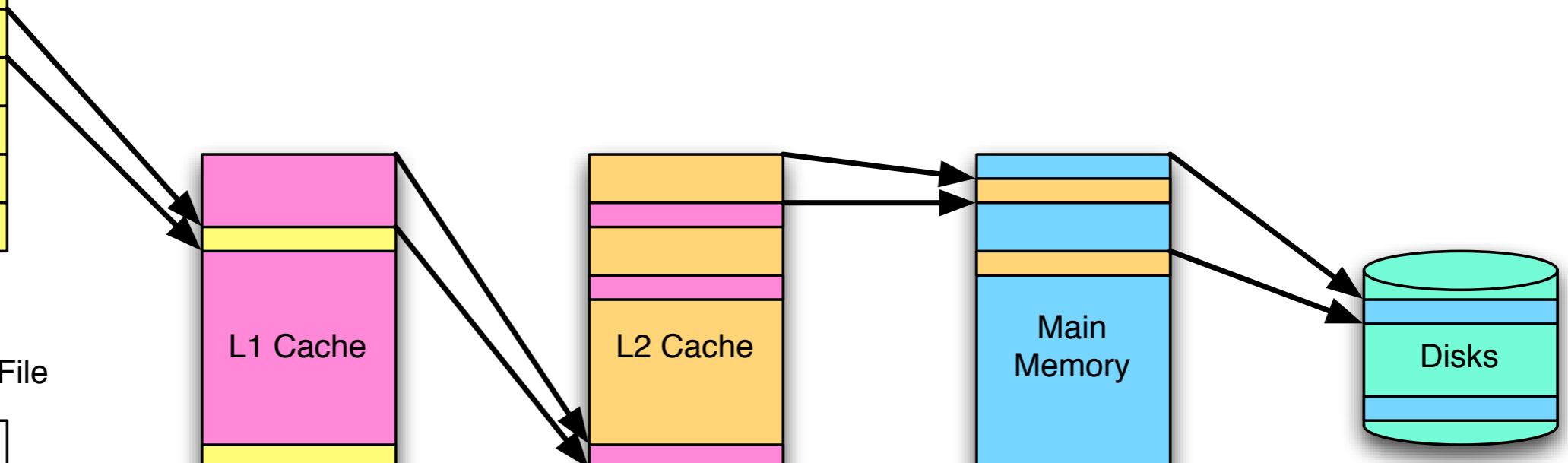
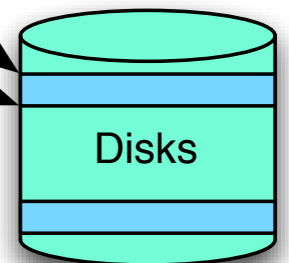
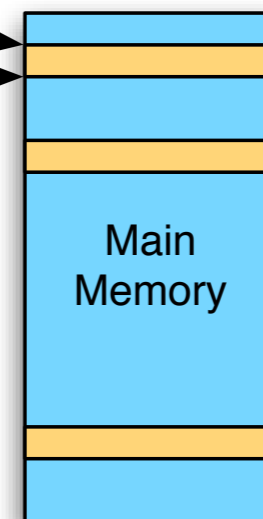
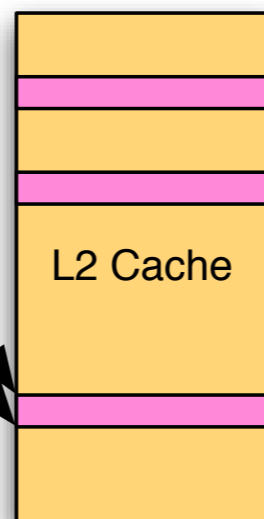
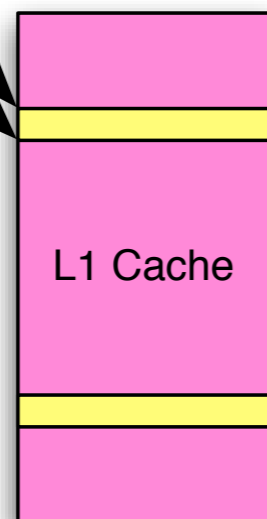
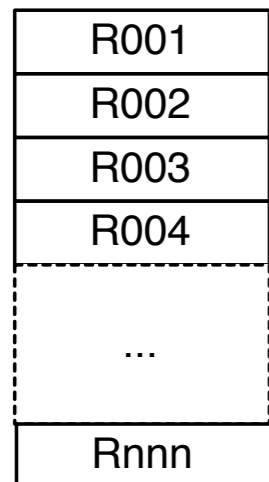
What was *really* happening on the subject's computer?

Computer systems arrange memory in a hierarchy.

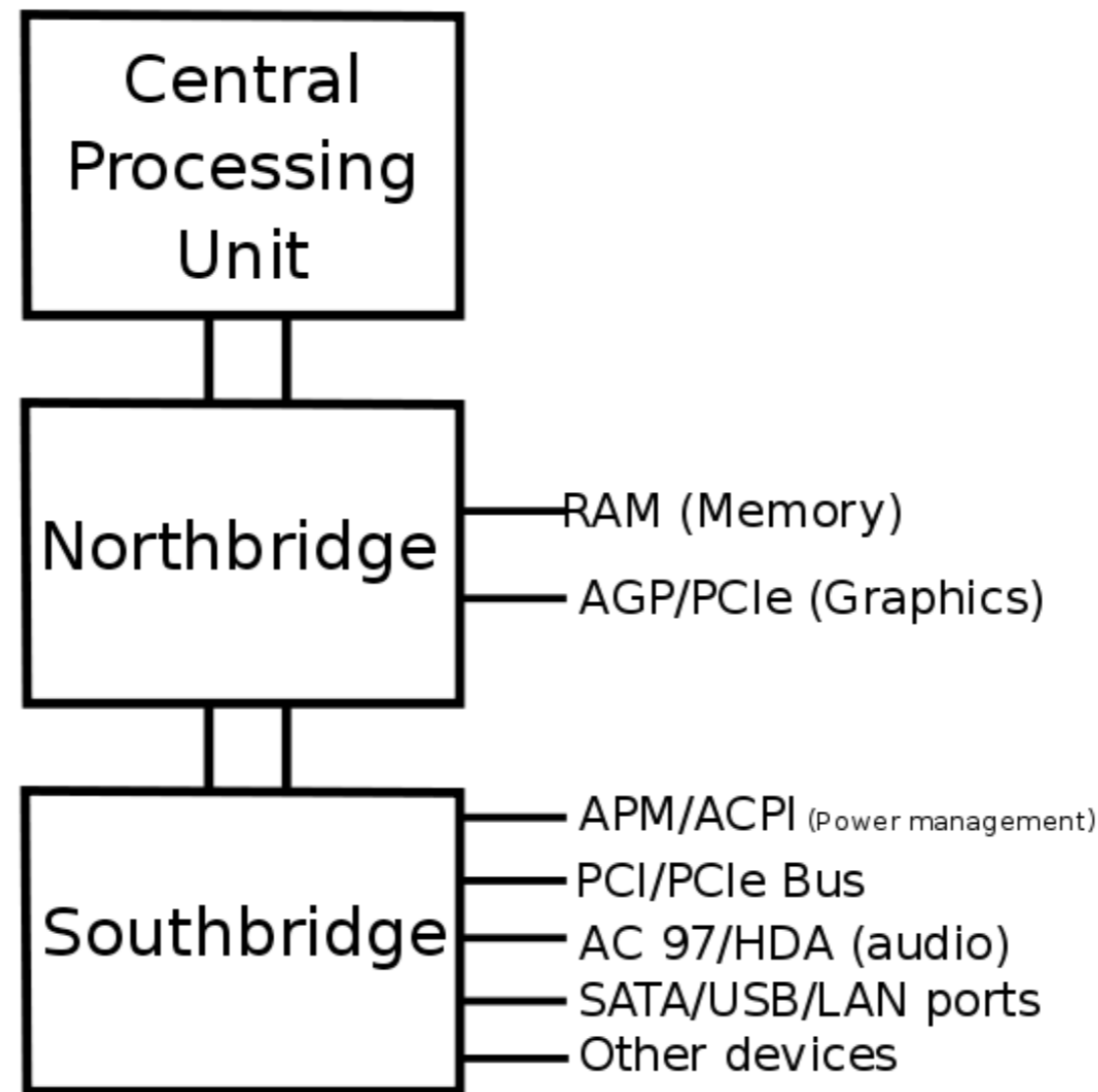
Architectural Registers



Active Register File



PCI-based systems typically employ a “northbridge” and a “southbridge” between CPU, Memory and Devices



[http://en.wikipedia.org/wiki/Northbridge \(computing\)](http://en.wikipedia.org/wiki/Northbridge_(computing))

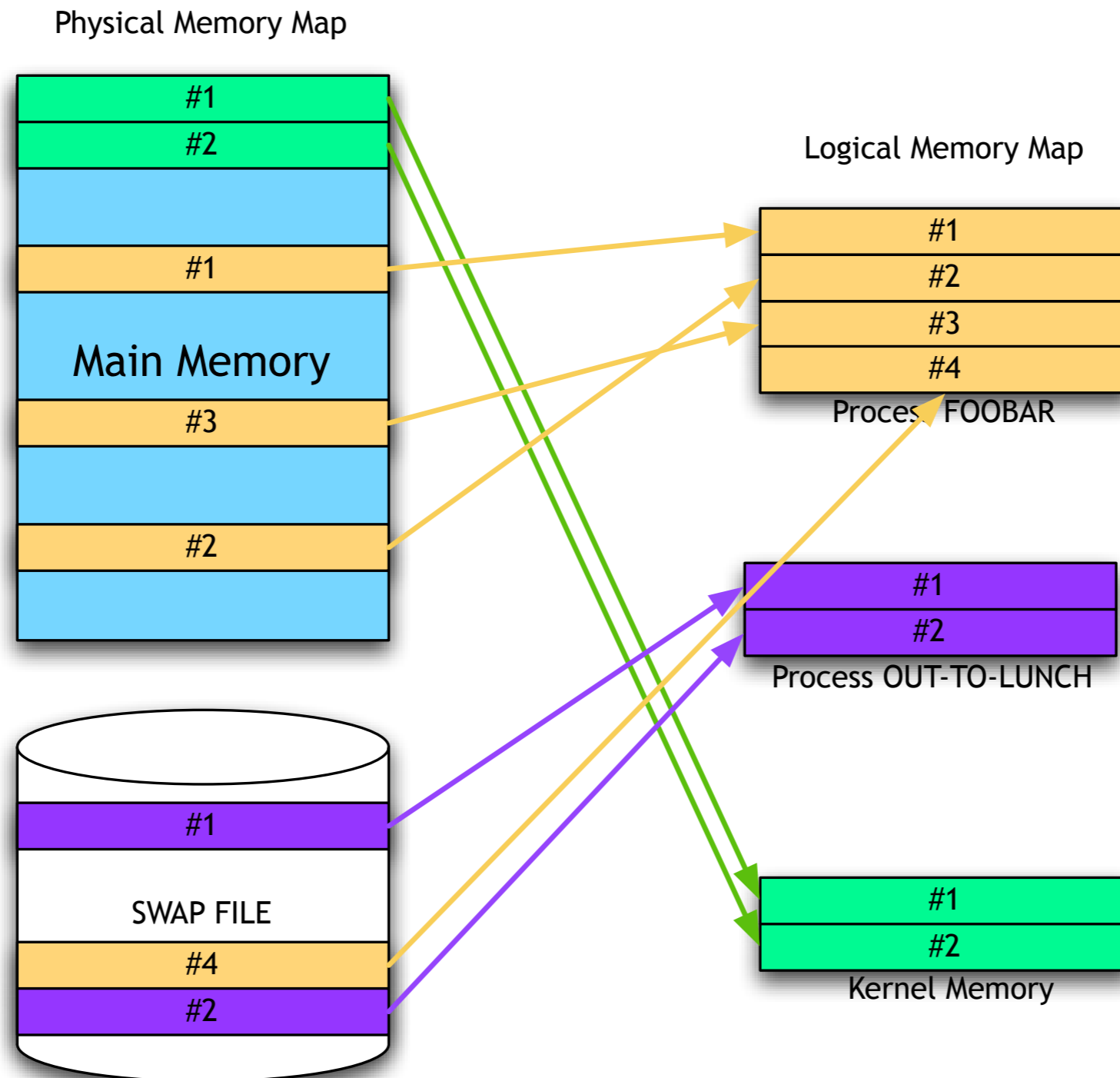
Control of memory *is* control of the computer.

Reading:

- Contents of the screen
- Cryptographic Keys
- Passwords (BIOS & programs)
- Current Running Programs
- Remnants of previously run programs
- Open TCP/UDP ports
- Cached data
- Hidden data

Writing:

- Patch programs on the fly
- Change security levels
- Install malware



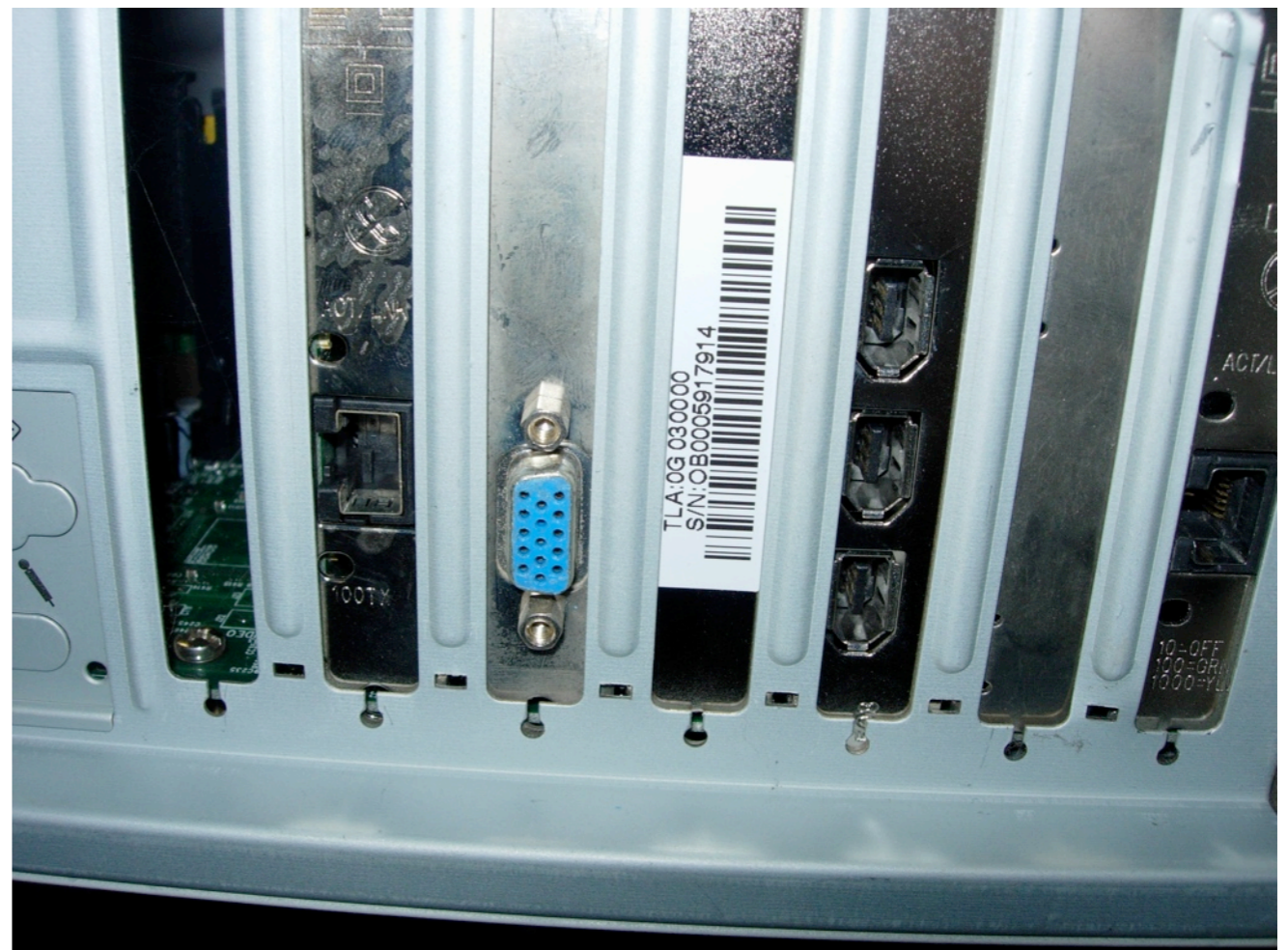
Memory can be acquired "LIVE" or "DEAD"

Swap space on live or dead systems

- PAGEFILE.SYS
- /private/var/vm/swapfile
- Swap Partitions
- *Suspend/Resume*

Live Memory:

- /dev/mem
- /proc/kcore
- \\.\PhysicalMemory
- \\.\DebugMemory
- Device Drivers
- Special programs (WinEn)
- Hardware memory imagers
- Firewire (provides DMA)



Options for RAM acquisition.

Hardware Acquisition

- Special-purpose PCI card
- Firewire / PATA / SATA
- Cold Boot Attack

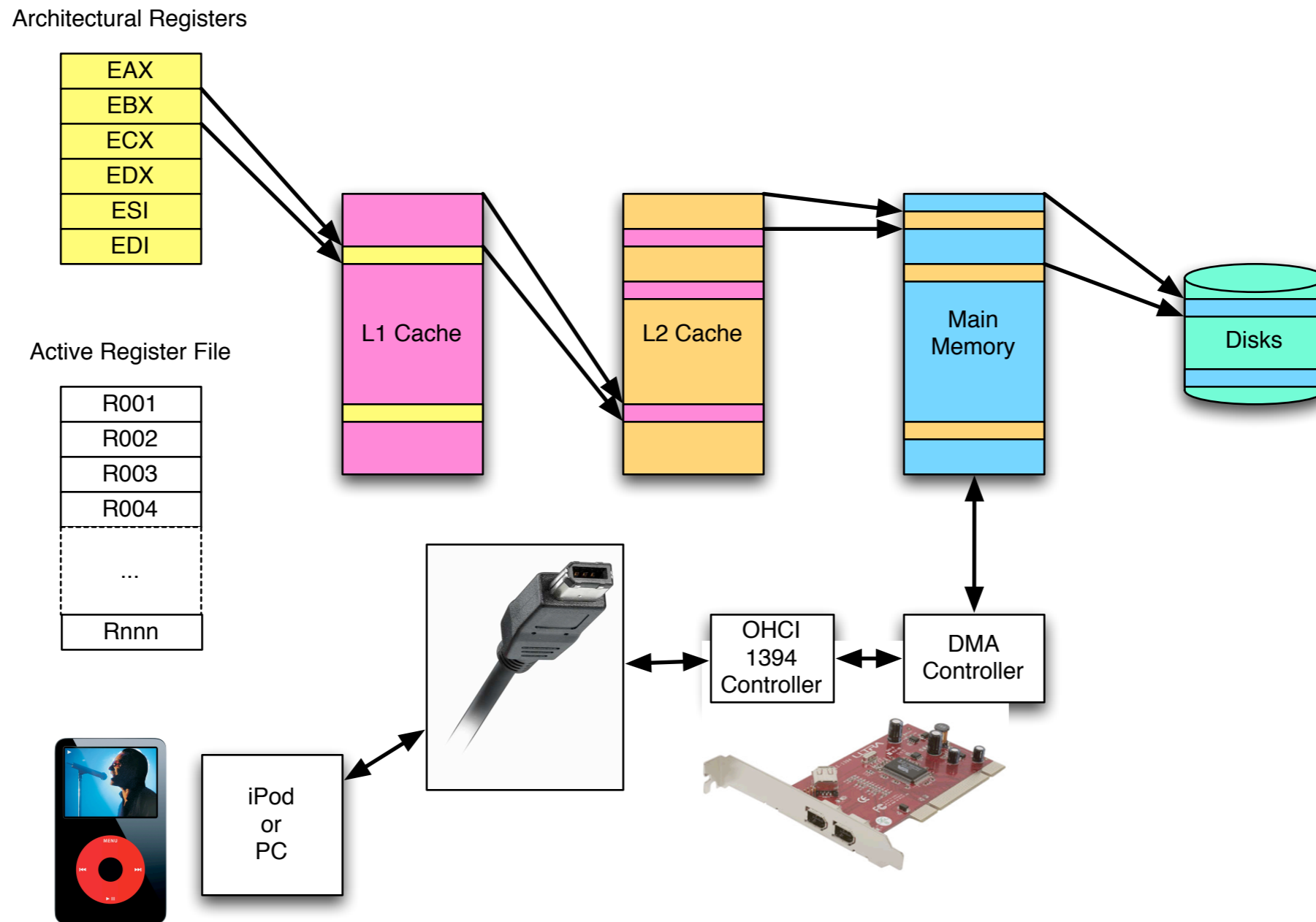
Software Acquisition

- User-level program (Windows XP2)
- User-level program with device driver (Windows XP3, Vista, Windows 7)

Hibernation Files and Virtual Machines

- hiberfil.sys
- VMWare stores "Ram" in ***FILENAME.vmem***

Hardware approach #1: acquisition by DMA (Firewire; PCI card; etc.)



DMA bypasses the operating system and the CPU.

Hardware Approach #2: "Cold Boot Attack"

1. Reboot computer with attack disk.

Some BIOS may wipe memory on boot.

You don't know in advance, so...

2. Chill memory.

3. Remove memory from subject computer and place in a computer with known BIOS

4. Reboot computer with attack disk.

<http://citp.princeton.edu/>



Software approaches for acquiring memory from live system:

Windows:

- Windd (Matthieu Suiche) - <http://www.msuiche.net/>
- dd from windows memory device (pre-XP SP3)
- ManTech's Memory DD - <http://www.mantech.com/msma/mdd.asp>
- Guidance Software's Winen.exe
 - ✓ Many of these are built into Helix 3
 - ✓ <http://www.e-fense.com/helix/>

Unix:

- dd from /dev/mem or /dev/kmem

VMWare:

- Suspend and grab *filename.vmem*

Windows preferred too: win32dd.exe / win64dd.exe
<http://www.msuiche.net/>

```
Administrateur : C:\Windows\System32\cmd.exe
C:\>Suiche\amd64>win64dd.exe /d /f toto.dmp
win64dd - v1.3.20091010 (RTM) - Kernel land physical memory acquisition
Copyright (c) 2007 - 2009, Matthieu Suiche <http://www.msuiche.net>
Copyright (c) 2008 - 2009, MoonSols <http://www.moonsols.com>

Name                               Value
----                               -
File type:                          Microsoft memory crash dump file
Acquisition method:                 PFN Mapping
Content:                            Memory manager physical memory block

Destination path:                   toto.dmp

O.S. Version:                       Microsoft Windows 7 Ultimate, 64-bit (build 7600)

Computer name:                      M1330

Physical memory in use:              24%
Physical memory size:               8386596 Kb ( 8190 Mb)
Physical memory available:          6370732 Kb ( 6221 Mb)

Paging file size:                   16771296 Kb ( 16378 Mb)
Paging file available:              14423216 Kb ( 14085 Mb)

Virtual memory size:                8589934464 Kb (8388607 Mb)
Virtual memory available:           8589886504 Kb (8388561 Mb)

Extended memory available:          0 Kb ( 0 Mb)

Physical page size:                 4096 bytes
Minimum physical address:           0x0000000000001000
Maximum physical address:           0x0000000021FFFF000

Address space size:                 9126805504 bytes (8912896 Kb)

--> Are you sure you want to continue? [y/n]
Acquisition started at:             [10/10/2009 (DD/MM/YYYY) 20:41:3 (UTC)]

Processing....Done.

Acquisition finished at: [2009-10-10 (YYYY-MM-DD) 20:45:07 (UTC)]
Time elapsed:                4:04 minutes:seconds (244 secs)

Created file size:                8587882496 bytes ( 8190 Mb)

NtStatus (troubleshooting):        0x00000000
Total of written pages:             2096651
Total of inaccessible pages:        0
Total of accessible pages:          2096651

Physical memory in use:              23%
Physical memory size:               8386596 Kb ( 8190 Mb)
Physical memory available:          6375016 Kb ( 6225 Mb)

Paging file size:                   16771296 Kb ( 16378 Mb)
Paging file available:              14437448 Kb ( 14099 Mb)

Virtual memory size:                8589934464 Kb (8388607 Mb)
Virtual memory available:           8589886504 Kb (8388561 Mb)
```

e-fense: live response CDs

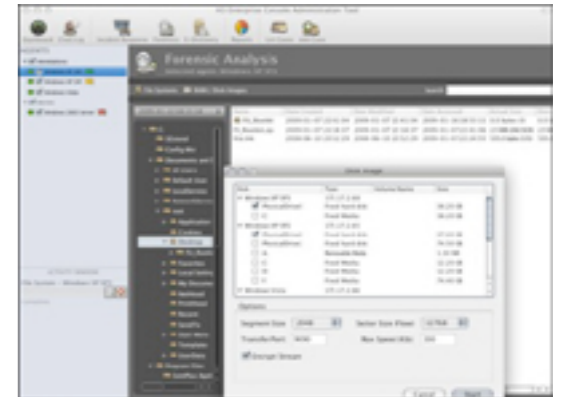
<http://www.e-fense.com/>

Helix: Bootable CD

- Enterprise & open source versions available

Live Response USB Key

- Physical memory
- Network connections, open TCP or UDP ports, NetBIOS
- Currently logged on user / user accounts
- Current executing processes and services
- Scheduled jobs; Installed applications and drives
- Windows registry; Windows SAM files / NTUser.dat files
- Browser auto-completion data, passwords
- Screen capture; Chat logs; System logs; Environment variables; Internet history



Potential problems with acquiring live memory:

Speed:

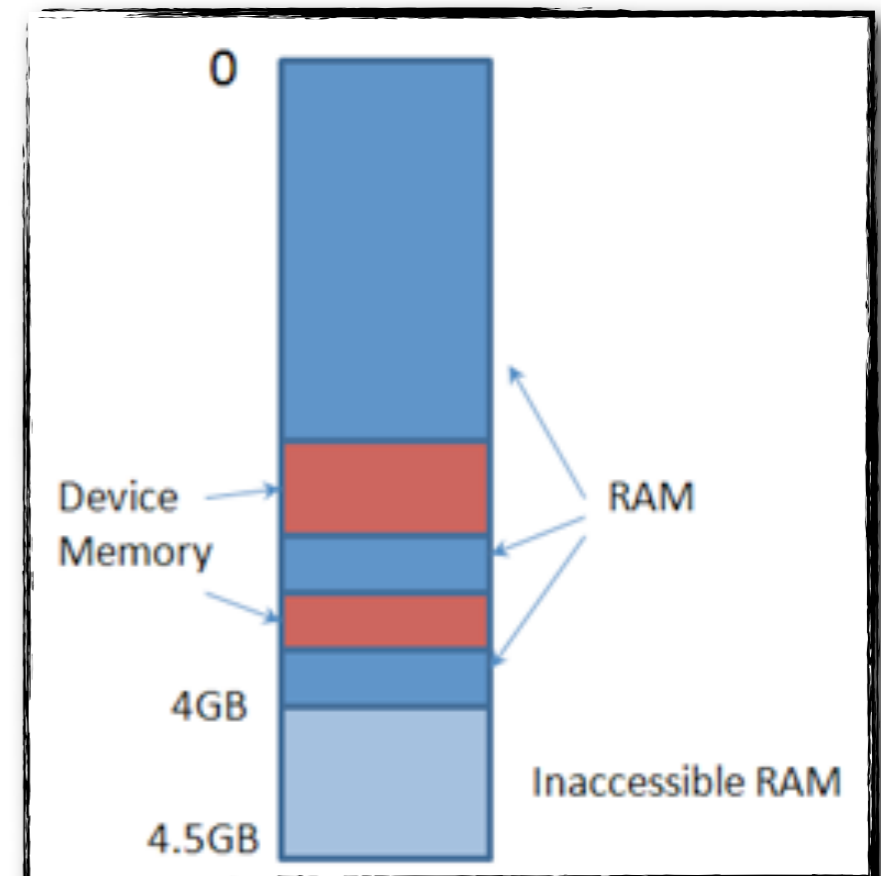
- Memory changes fast; it won't be consistent.

Reliability:

- Software methods can be blocked by attacker.

Integrity:

- Software changes the memory map
- You can't get all the memory



Memory Analysis Techniques

Look for ASCII and UNICODE strings.

- strings(1), grep

"File carving"

- foremost, scalpel
- Princeton key search program

Identify and interpret kernel or program data structures

- Convert Windows memory image to Microsoft crashdump format, then analyze with standard debugging tools (WinDbg):
 - ✓ http://computer.forensikblog.de/en/2006/03/dmp_file_structure.html
 - ✓ <http://www.shakacon.org/talks/NFI-Shakacon-win32dd0.3.pdf>
- KnTTools (George Garner)
- Volatility, by Volatile Systems (<http://www.volatilesystems.com>)
- Idetect

KnTTools (Windows), by George M. Garner, Jr.

KNTDD - Acquires memory

- Acquisition to removable drive or network
- Cryptographic integrity checks, auditing
- Conversion to Microsoft crash dump format
- Remote deployment as a service

KnTList - Lists Kernel Structures

- Reconstructs virtual address space
- Drives, Device Objects, System Tables
- Threads, access tokens, handle table, objects, etc.
- Outputs as text and XML

<http://forensic.seccure.net/>

<http://gmgsystemsinc.com/knttools/>

WMFT - Windows Memory Forensic Toolkit

Enumerates processes, modules, libraries

Finds hidden data (rootkits)

Detailed information:

- Access tokens
- Handles
- Processes
- Modules

<http://forensic.seccure.net/>

Idetect (Linux)

Displays detailed information for each process

Enumerates all process-related structures

Can work on memory image or live system

- <http://forensic.seccure.net/tools/idetect.tar.gz>
- http://forensic.seccure.net/pdf/mburdach_digital_forensics_of_physical_memory.pdf

Lots more information about memory forensics, including 53-page presentation:

- <http://forensic.seccure.net> (2006)



Volatility:

A tool for analyzing windows memory dumps

Created by AAron Walters and Nick L. Petroni

- Open Source (unlike prior systems)
- Written in Python

Extracts:

- Image date & time
- Memory map for each running process
- Network sockets
- DLLs loaded for each process
- Lots more.

<https://www.volatilesystems.com/VolatileWeb/volatility.gsp>

<http://volatility.tumblr.com/>

Memory Lab: Try out Volatility

I have provided you with:

Exemplar data from NIST

- <http://www.cfreds.nist.gov/mem/memory-images.rar>

A copy of Volatility 1.3 Beta:

- https://www.volatilesystems.com/volatility/1.3/Volatility-1.3_Beta.tar.gz

Or Acquire and Analyze your own memory!

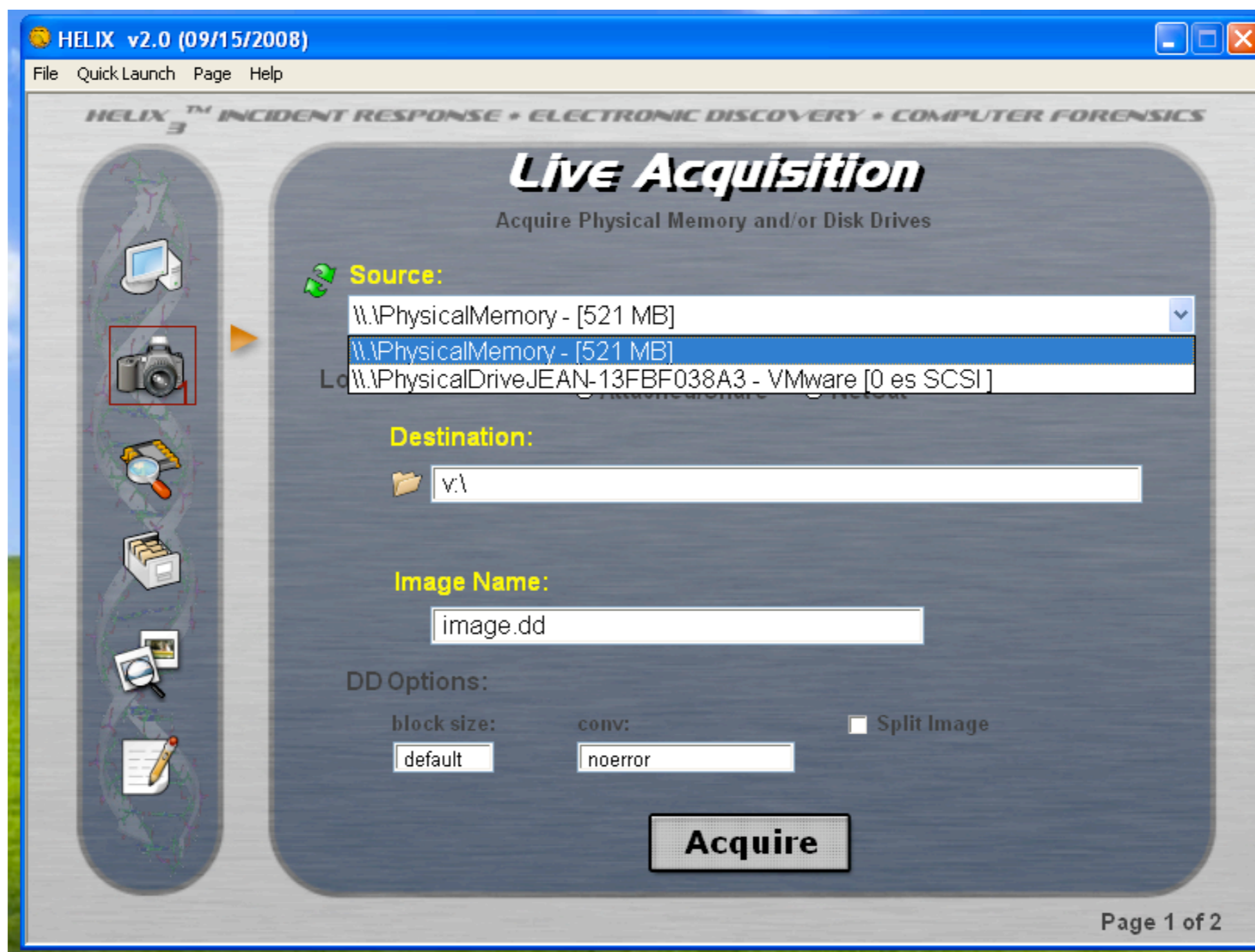
Windows: Run the Helix 3.0, save memory to a disk file (or USB), and analyze.



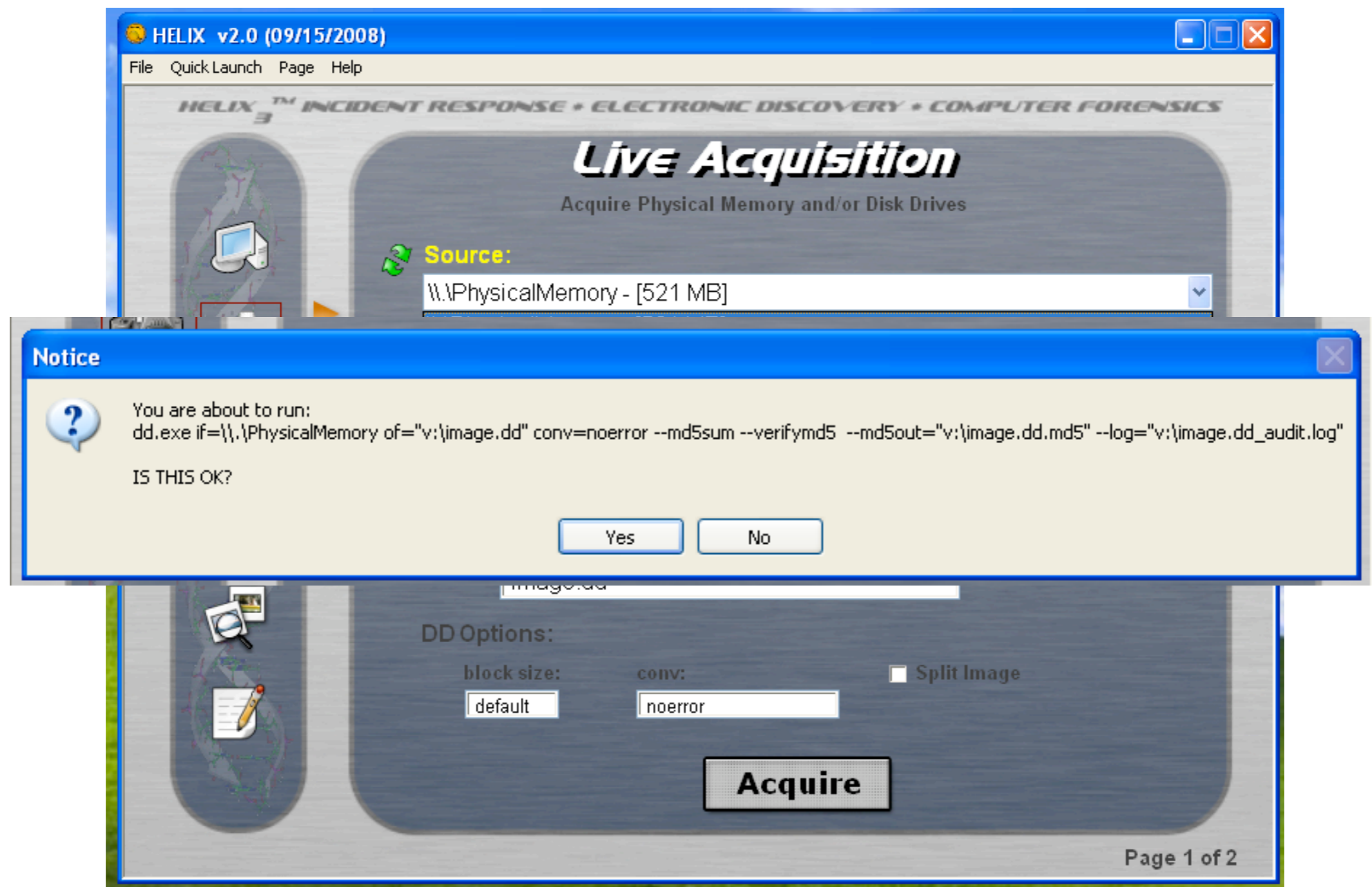
Select Live Acquisition



Select what to acquire and where it goes.



Select what to acquire and where it goes.



Or take it out of a suspended VMWare Machine:

```
drwxr-xr-x  2  simsong  admin           68 Sep 15 22:37 Applications/
-rw-----@ 1  simsong  admin          8684 Aug 12 18:36 Windows XP Clean.nvram
-rw-----@ 1  simsong  admin      6123552768 Aug 12 18:36 Windows XP Clean.vmdk
-rw-----@ 1  simsong  admin      536870912 Aug 12 17:09 Windows XP Clean.vmem
-rw-----@ 1  simsong  admin          1014 Jul 24 10:56 Windows XP Clean.vmsd
-rw-----@ 1  simsong  admin      67521174 Aug 12 18:36 Windows XP Clean.vmss
-rwxr-xr-x@ 1  simsong  admin          2820 Aug 12 18:36 Windows XP Clean.vmx*
-rw-----@ 1  simsong  admin         16655 Jul 24 10:44 Windows XP Clean.vmx*
-rw-r--r--  1  simsong  admin      276990 Nov  7 22:00 quicklook-cache.png
-rw-r--r--  1  simsong  simsong       61315 Jul 29 15:32 vmware-0.log
-rw-r--r--  1  simsong  simsong       74794 Jul 29 12:26 vmware-1.log
-rw-r--r--  1  simsong  simsong       66044 Jul 29 10:29 vmware-2.log
-rw-r--r--  1  simsong  simsong       76096 Aug 12 18:36 vmware.log
```

image.dd_audit.log

Forensic Acquisition Utilities, 1, 0, 0, 1035
dd, 3, 16, 2, 1035
Copyright (C) 2002-2004 George M. Garner Jr.

Command Line: dd.exe if=\\.\PhysicalMemory of="v:\image.dd" conv=noerror --md5sum --verifymd5
--md5out="v:\image.dd.md5" --log="v:\image.dd_audit.log"
Based on original version developed by Paul Rubin, David MacKenzie, and Stuart Kemp
Microsoft Windows: Version 5.1 (Build 2600.Professional Service Pack 3)

05/10/2008 22:21:40 (UTC)
05/10/2008 23:21:40 (local time)

Current User: JEAN-13FBF038A3\Administrator

Total physical memory reported: 523760 KB
Copying physical memory...
D:\IR\FAU\dd.exe:
Stopped reading physical memory:

The parameter is incorrect.
\74eb8e6cdaa43589e0b27449bd7ac03f [\\.\PhysicalMemory] *v:\\image.dd

Verifying output file...
\74eb8e6cdaa43589e0b27449bd7ac03f [v:\\image.dd] *v:\\image.dd
The checksums do match.
The operation completed successfully.

Output v:\image.dd (536866816 bytes)
131071+0 records in
131071+0 records out

My Windows machine has 512MB of RAM:

```
-r-xr-xr-x    1  simsong  simsong          1013 Oct  5 15:22 image.dd_audit.log
-r-xr-xr-x    1  simsong  simsong 536866816 Oct  5 15:22 image.dd
-r-xr-xr-x    1  simsong  simsong          73 Oct  5 15:22 image.dd.md5
```

```
$ cat image.dd.md5
\74eb8e6cdaa43589e0b27449bd7ac03f [\\\.\PhysicalMemory] *v:.\image.dd
$
```

Volatility commands:

```
$ python volatility
```

Supported Commands:

connections	Print list of open connections
connscan	Scan for connection objects
datetime	Get date/time information for image
dlllist	Print list of loaded dlls for each process
files	Print list of open files for each process
ident	Identify image properties such as DTB and VM type
modules	Print list of loaded modules
pslist	Print list of running processes
psscan	Scan for EPROCESS objects
sockets	Print list of open sockets
sockscan	Scan for socket objects
strings	Match physical offsets to virtual addresses
thrdscan	Scan for ETHREAD objects
vaddump	Dump the Vad sections to files
vadinfo	Dump the VAD info
vadwalk	Walk the vad tree

volatility pslist -f *filename*: See the processes

```
$ python volatility pslist -f winxp.mem
```

Name	Pid	PPid	Thds	Hnds	Time
System	4	0	57	187	Thu Jan 01 00:00:00 1970
smss.exe	612	4	3	19	Wed Aug 13 00:09:58 2008
csrss.exe	660	612	12	370	Wed Aug 13 00:10:01 2008
winlogon.exe	684	612	18	519	Wed Aug 13 00:10:02 2008
services.exe	728	684	16	269	Wed Aug 13 00:10:02 2008
lsass.exe	740	684	20	344	Wed Aug 13 00:10:02 2008
vmacthlp.exe	888	728	1	25	Wed Aug 13 00:10:03 2008
svchost.exe	904	728	17	196	Wed Aug 13 00:10:03 2008
svchost.exe	1020	728	10	269	Wed Aug 13 00:10:05 2008
svchost.exe	1056	728	55	1237	Wed Aug 13 00:10:06 2008
svchost.exe	1200	728	4	73	Wed Aug 13 00:10:07 2008
svchost.exe	1364	728	15	212	Wed Aug 13 00:10:13 2008
spoolsv.exe	1496	728	11	117	Wed Aug 13 00:10:15 2008
VMwareService.e	1796	728	4	139	Wed Aug 13 00:10:16 2008
searchindexer.e	1976	728	20	678	Wed Aug 13 00:10:17 2008
wscntfy.exe	276	1056	1	37	Wed Aug 13 00:10:22 2008
explorer.exe	480	456	13	351	Wed Aug 13 00:10:23 2008
VMwareTray.exe	548	480	1	37	Wed Aug 13 00:10:24 2008
VMwareUser.exe	556	480	3	184	Wed Aug 13 00:10:24 2008
Eraser.exe	572	480	3	90	Wed Aug 13 00:10:24 2008
ctfmon.exe	580	480	1	71	Wed Aug 13 00:10:24 2008
WindowsSearch.e	704	480	10	238	Wed Aug 13 00:10:25 2008
alg.exe	1108	728	6	105	Wed Aug 13 00:10:26 2008
imapi.exe	1336	728	5	118	Wed Aug 13 00:10:29 2008

volatility files -f *filename*: See the open files

```
$ python volatility pslist -f winxp.mem
```

```
Pid: 4
```

```
File    \pagefile.sys
```

```
File    \Documents and Settings\NetworkService\NTUSER.DAT
```

```
File    \WINDOWS\system32\config\SECURITY
```

```
File    \WINDOWS\system32\config\software
```

```
File    \WINDOWS\system32\config\SECURITY.LOG
```

```
File    \Documents and Settings\NetworkService\ntuser.dat.LOG
```

```
File    \WINDOWS\system32\config\software.LOG
```

```
File    \WINDOWS\system32\config\system
```

```
File    \WINDOWS\system32\config\system.LOG
```

```
File    \WINDOWS\system32\config\default
```

```
File    \WINDOWS\system32\config\default.LOG
```

```
File    \WINDOWS\system32\config\SAM
```

```
File    \WINDOWS\system32\config\SAM.LOG
```

```
File    \Documents and Settings\NetworkService\Local Settings\Application Data  
        \Microsoft\Windows\UsrClass.dat
```

```
File    \Documents and Settings\NetworkService\Local Settings\Application Data  
        \Microsoft\Windows\UsrClass.dat.LOG
```

```
File    \Documents and Settings\Administrator\NTUSER.DAT
```

```
File    \Documents and Settings\Administrator\Local Settings\Application Data  
        \Microsoft\Windows\UsrClass.dat.LOG
```

```
File    \
```

```
File    \Documents and Settings\Administrator\ntuser.dat.LOG
```

```
File    \Documents and Settings\Administrator\Local Settings\Application Data  
        \Microsoft\Windows\UsrClass.dat
```

Use strings(1) to find the printable strings...

```
$ strings image.dd | grep .....|head -10
Invalid partition ta
r loading operating system
Missing operating system
X509_REQ_add1_attr_by_txt
X509_REQ_add_extensions
X509_REQ_add_extensions_nid
X509_REQ_check_private_key
X509_REQ_delete_attr
X509_REQ_digest
```

Use strings(1) detect JPEG files...

```
$ strings filename.jpg
```

```
JFIF
```

```
ICC_PROFILE
```

```
appl
```

```
mntrRGB XYZ
```

```
acspAPPL
```

```
appl
```

```
-appl
```

```
rXYZ
```

```
gXYZ
```

```
...
```

```
$ strings image.dd | grep -i JFIF | head -10
```

```
JFIF
```

```
JFIF
```

```
.jfif:
```

```
.jfif
```

```
HKLM,"%PATH_ALLOWEDIMGEXTS%",".jfif",0x10001,0x1
```

```
ijjgiiggjffifjjgiijjjjjigjjijgjjiiijijjjiiiffjijjjjjjjijjjijijjjiiijiijjjiigfijjjjjjjijjjjjjjgi  
jjjj0
```

```
JFIF
```

```
JFIF
```

```
.jfif:
```

```
HKCR,".jfif",,, "jpegfile"
```

```
$
```

Other tricks with Windows Memory:

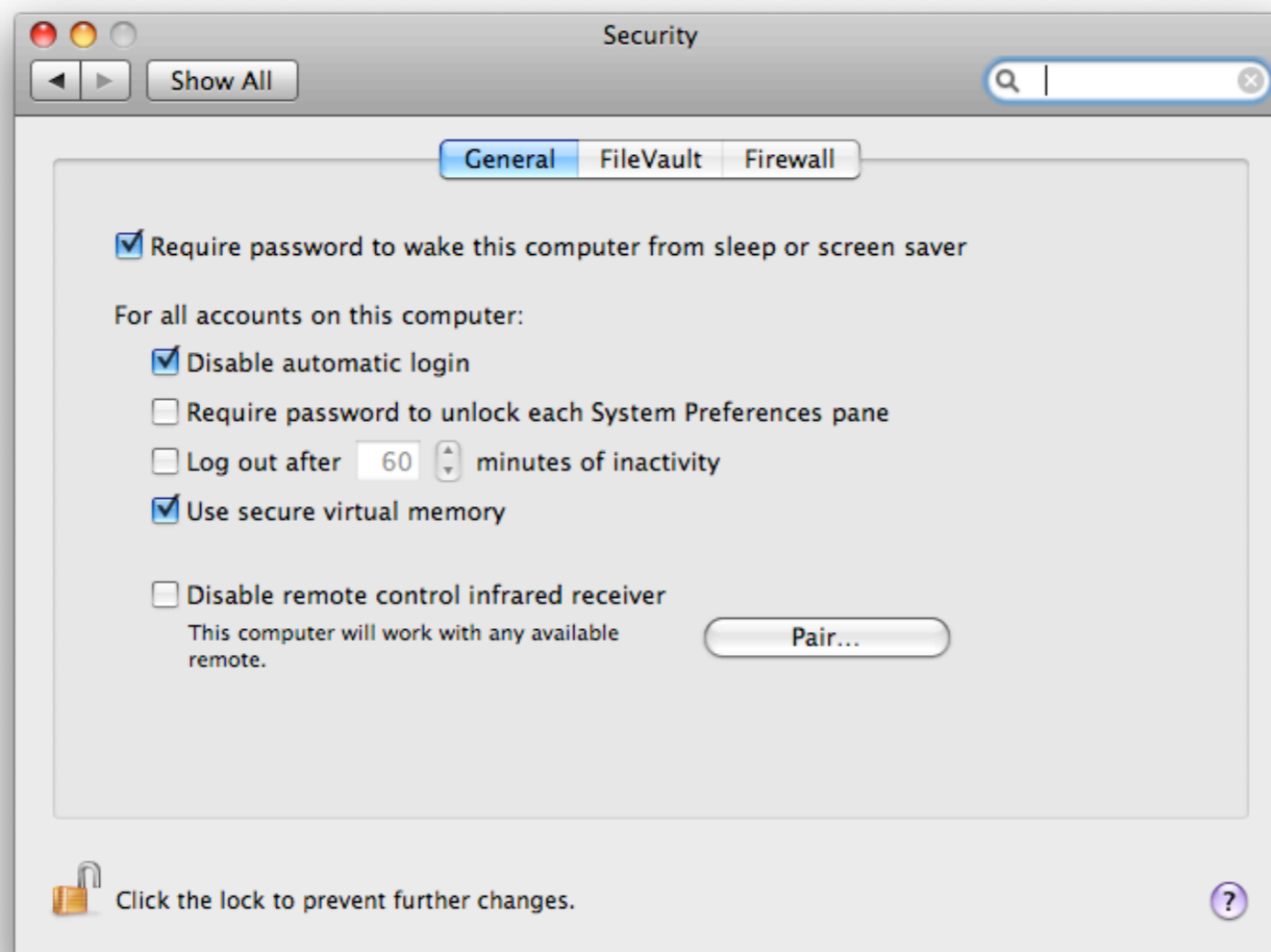
Recover gmail messages from the browser's memory

- **pdgmail** to get the mail messages.

✓ <http://sansforensics.wordpress.com/2008/10/20/pdgmail-new-tool-for-gmail-memory-forensics/>

Use a fragment recovery carver (more about this in a bit.)

Don't believe MacOS "Secure Virtual Memory"



```
$ ls -l /private/var/vm
total 4259840
-rw-----T  1 root  wheel  4294967296 May  3 13:51 sleepimage
-rw-----T  1 root  wheel    67108864 May  4 00:08 swapfile0
```

Summary:

Memory Forensics

Memory forensics analysis:

- Analysis of live memory & suspended memory
- Bulk analysis & high-level analysis

Advantages:

- Gets around disk encryption
- No systems have encrypted memory (yet)

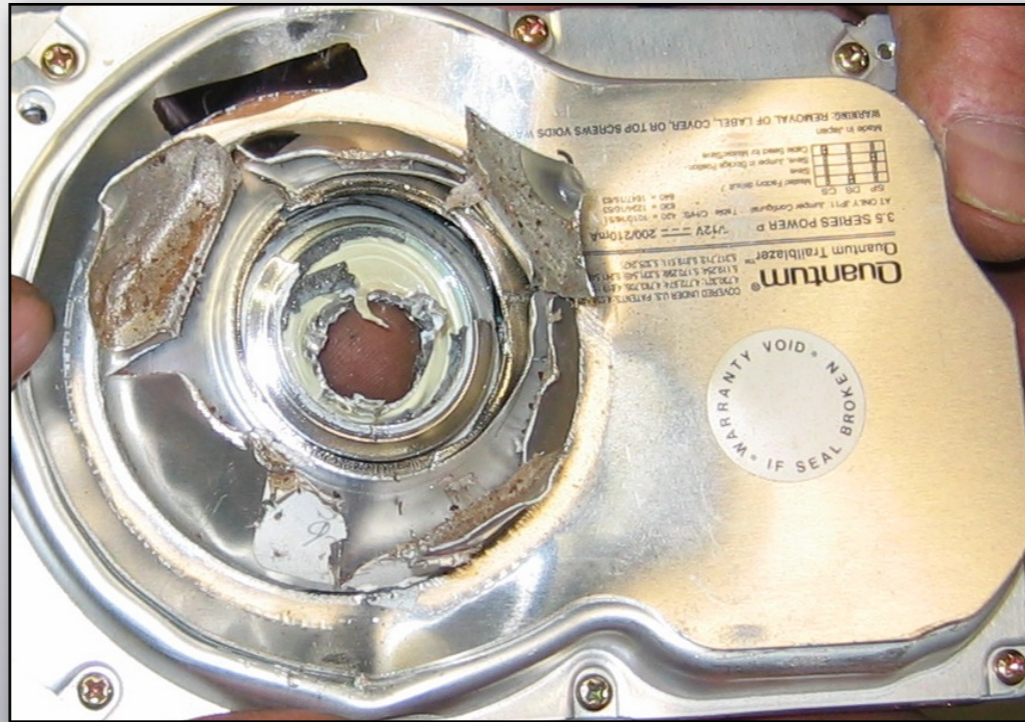
Disadvantages:

- Operating system specific.
- Tools are very primitive, but getting better.

See also:

- http://www.forensicswiki.org/wiki/Windows_Memory_Analysis





Anti-Forensics: Techniques, Detection and Countermeasures

What is Anti-Forensics?

Computer Forensics: *“Scientific Knowledge for collecting, analyzing, and presenting evidence to the courts” (USCERT 2005)*

Anti-Forensics: *tools and techniques that frustrate forensic tools, investigations and investigators*

Goals of Anti-Forensics:

- *Avoiding detection*
- *Disrupting information collection*
- *Increasing the examiner’s time*
- *Casting doubt on a forensic report or testimony (Liu and Brown, 2006)*

- *Forcing a tool to reveal its presence*
- *Subverting the tool — using it to attack the examiner or organization*
- *Leaving no evidence that the AF tool has been run*

Physical destruction makes forensic recovery impossible.



One traditional Anti-Forensic technique is to overwrite or otherwise destroy data.

Overwriting: Eliminate data or metadata (e.g. disk sanitizers, Microsoft Word metadata “washers,” timestamp eliminators.)

Disk Sanitizers; Free Space Sanitizers; File Shredders

- Microsoft **Remove Hidden Data Tool**; **cipher.exe**; **ccleaner**

Metadata Erasers

- Example: **timestomp** - Gives all files the same atime/mtime/ctime

Hard problem: *What should be overwritten?*

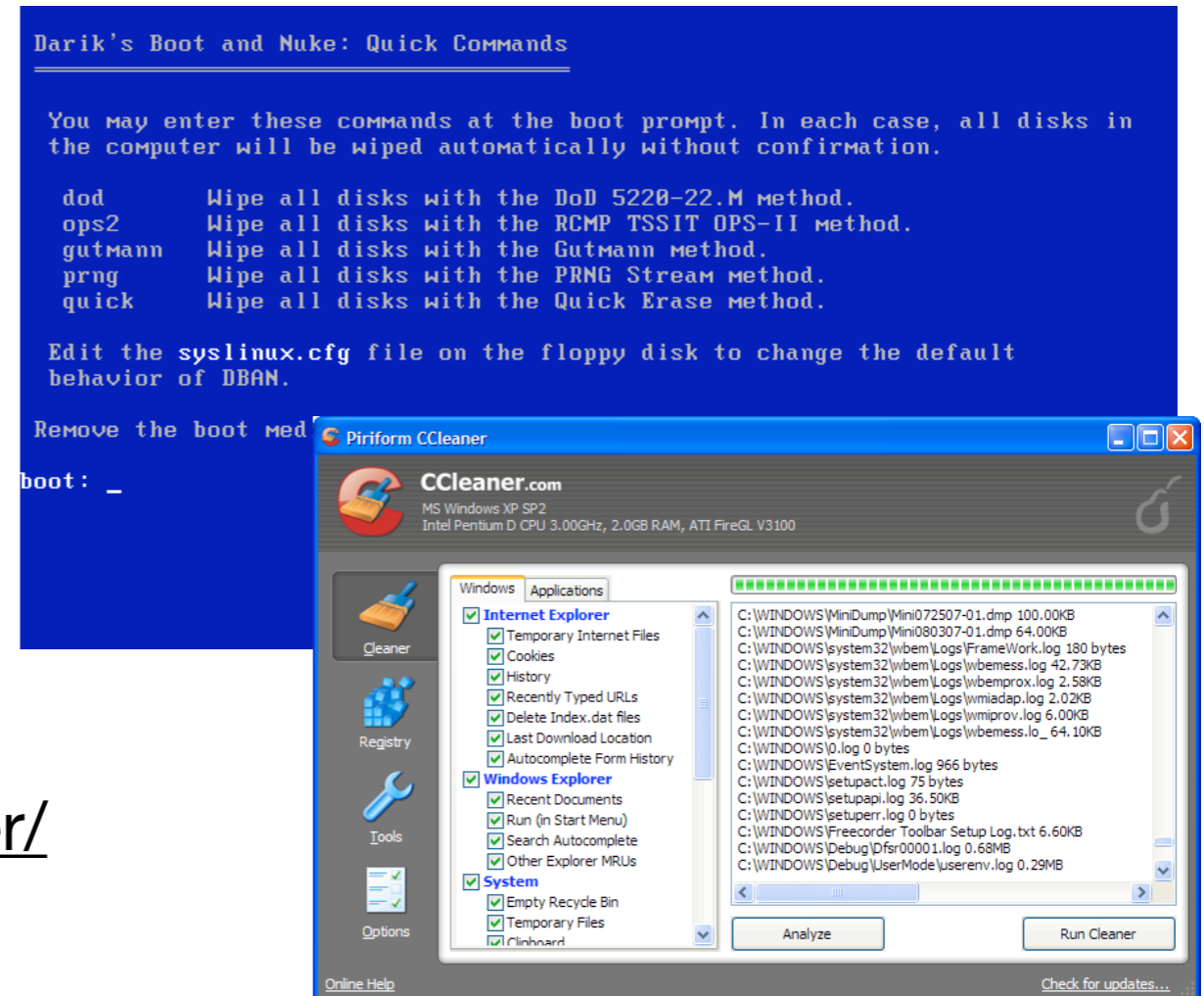
Overwriting: Two Approaches

Overwrite Everything — DBAN — Darik's Boot and Nuke

- <http://www.dban.org/>
- A single pass is sufficient.

Overwrite just...

- Windows temp files?
- Cookies?
- Pornography?
- <http://www.ccleaner.com/>
- <http://sourceforge.net/projects/eraser/>
- <http://heidi.ie/eraser>



See “Evaluating Commercial Counter-Forensic Tools,” Matthew Geiger

- http://www.dfrws.org/2005/proceedings/geiger_couterforensics.pdf
- <http://www.first.org/conference/2006/papers/geiger-matthew-papers.pdf>

Another approach: Hide data where tools won't look for it.

Data Hiding in File System Structures

- Slacker — Hides data in slack space
- FragFS — Hides in NTFS Master File Table
- RuneFS — Stores data in “bad blocks”
- KY FS — Stores data in directories
- Data Mule FS — Stores in inode reserved space

Data Hiding "out of the map"

- Host Protected Areas (HPA) & Device Configuration Overlay (DCO)
- Bad block areas of hard drives
- Graphics RAM

Approach Two: Cryptography or steganography.

Cryptographic File Systems

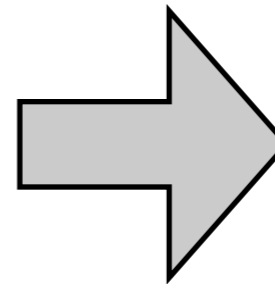
- Built in: FileVault & EFS
- Add-on: BestCrypt, TrueCrypt, FreeOTFE

Encrypted Network Protocols (SSL, SSH, Onion Routing*)

Program Packers (PECompact, Burneye) & Rootkits

Steganography

- OpenStego (Images)
- MP3Stego (Music)



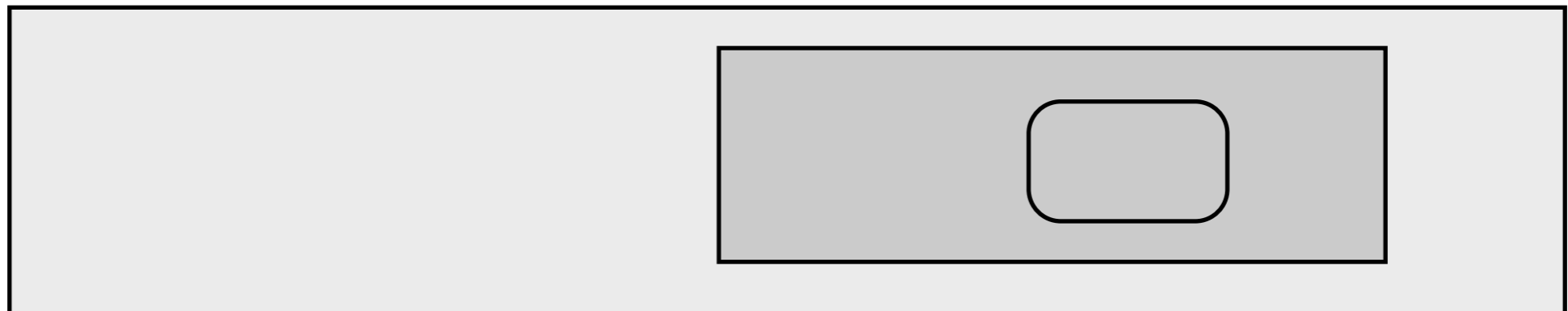
***Onion routing also protects from traffic analysis**

Cryptographic File Systems are increasingly a problem for forensic investigators

TRUECRYPT

FREE OPEN-SOURCE ON-THE-FLY ENCRYPTION

Many allow hiding an encrypted file system inside an encrypted file system:



Law on forcing people to reveal keys is unclear.

Transparency (FileVault, EFS, IronKey) makes it easier for the bad guys.



Anti-Forensics 3: Minimizing the Footprint

Overwriting and Data Hiding are *easy to detect*.

- Tools leave tell-tale signs; examiners know what to look for.
- Statistical properties are different after data is overwritten or hidden.

AF tools that minimize footprint avoiding leaving traces for later analysis.

- Memory injection and syscall proxying
- Live CDs, Bootable USB Tokens
- Virtual Machines—VMWare, QEMU, etc.
- Anonymous Identities and Storage

Memory Injection and Userland Execve:

Running a program without loading the code.

Memory Injection loads code without having the code on the disk.

- **Buffer overflow** exploits — run code supplied as (oversized) input

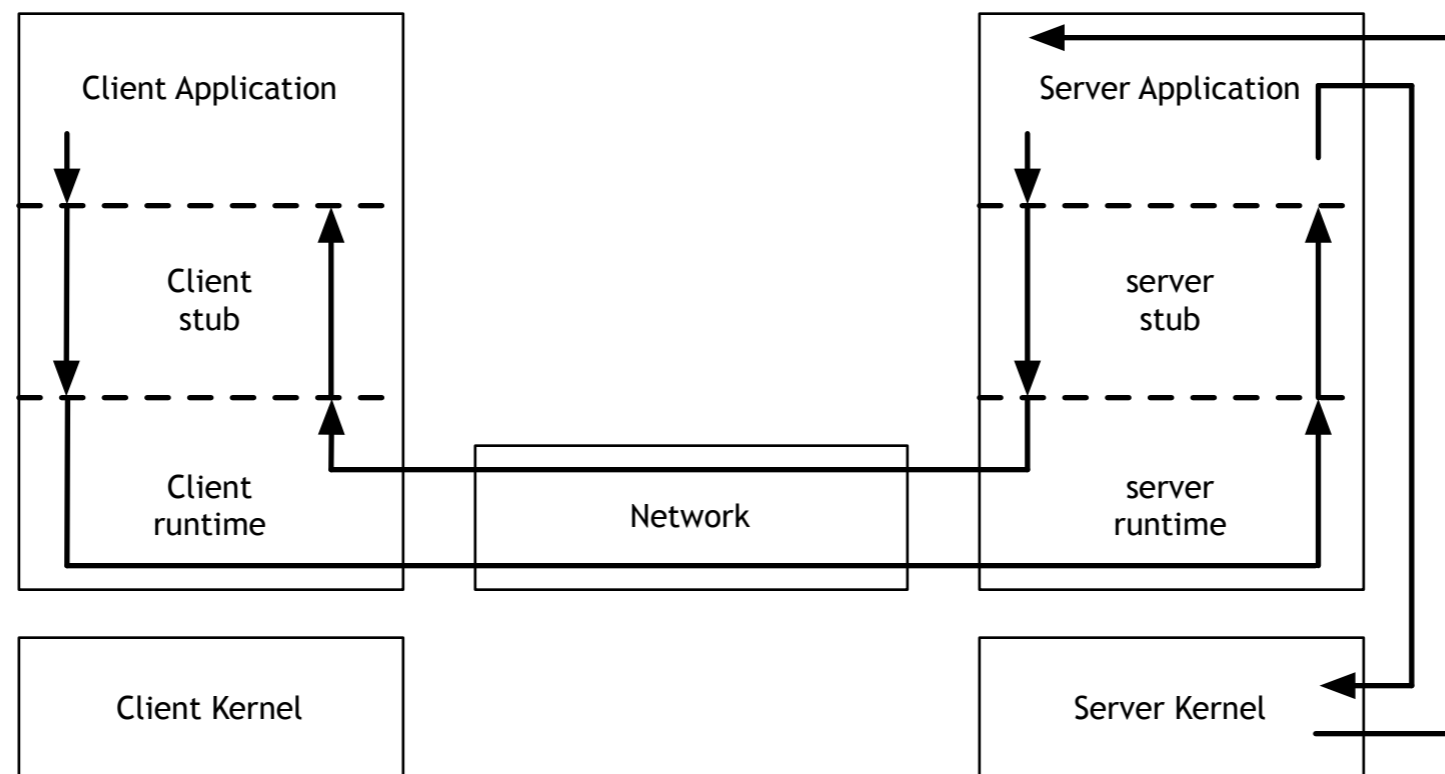
Userland Execve

- Runs program without using `execve()`
- Bypasses logging and access control
- Works with code from disk or read from network

Syscall proxying: Running a program without the code!

Syscall Proxying

- Program runs on one computer, syscalls executed on another.
- Program not available for analysis
- May generate a lot of network traffic
- Developed by Core Security; used in **Impact**



Live CDs, Bootable USB Tokens, Virtual Machines: Running code without leaving a trace.

Most forensic information is left in the file system of the running computer.

These approaches keep the attacker's file system segregated:

- In RAM (CDs & Bootable USB Tokens)
- In the Virtual Machine file (where it can be securely deleted)



Anonymous Identities and Storage:

The attacker's data may be anywhere.

Attackers have long made use of anonymous e-mail accounts. Today these accounts are far more powerful.

- Yahoo and GMail both have 2GB of storage
- APIs allow this storage to be used as if it were a file system

Amazon's Elastic Compute Cloud (EC2) and Simple Storage Service (S3) provide high-capability, little-patrolled services to anyone with a credit card

- EC2: 10 ¢/CPU hour (Xen-based virtual machines)
- S3: 10 ¢/GB-Month

With BGP, it's possible to have “anonymous IP addresses.”

1. Announce BGP route
2. Conduct attack
3. Withdraw BGP address

Being used by spammers today
(<http://www.nanog.org/mtg-0602/pdf/feamster.pdf>)

Attacking the Investigator: AF techniques that exploit CFT bugs.

Craft packets to exploit buffer-overflow bugs in network monitoring programs like **tcpdump**, **snort** and **ethereal**.

Create files that cause EnCase to crash.

Successful attacks provide:

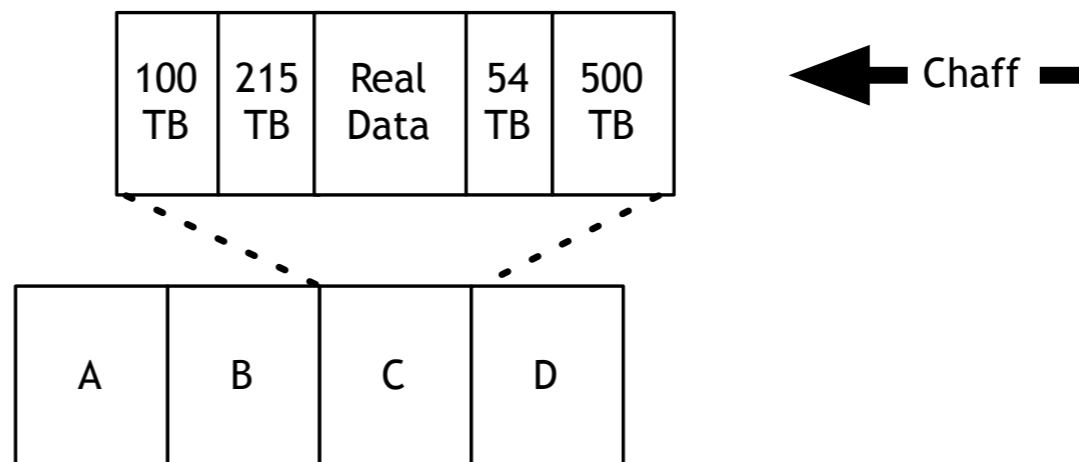
- ➡ Ability to run code on the forensic appliance
- ➡ Erase collected evidence
- ➡ Break the investigative software
- ➡ Leak information about the analyst or the investigation
- ➡ Implicate the investigator

Attacking the Investigator: Denial-of-Service Attacks against the CFT

Any CFT resource whose use is determined by input can be overwhelmed.

- Create millions of files or identities
- Overwhelm the logging facility
- Compression bombs — 42.zip

The clever adversary will combine this **chaff** with real data, e.g.:



Anti-Forensic Tools can detect

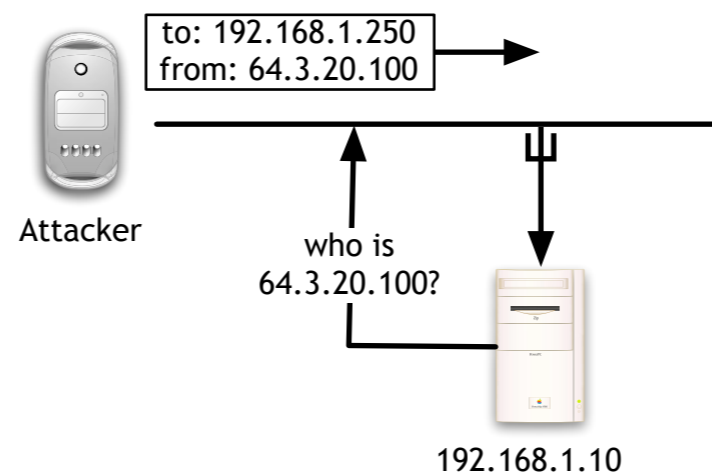
Computer Forensic Tools: cat-and-mouse.

SMART (Self-Monitoring, Analysis and Reporting Technology) drives report:

- Total number of power cycles
- Total time hard drive has been on

Network Forensics can be detected with:

- Hosts in “promiscuous” mode responding differently
 - to PINGs.
 - to malformed packets
 - to ARPs
- Hosts responding to traffic not intended to them (MAC vs. IP address)
- Reverse DNS queries for packets sent to unused IP addresses



Countermeasures for Anti-Forensics

Improve the tools — many CFTs are poorly written.

Save data where the attacker can't get at it:

- Log hosts
- CD-Rs

Develop new tools:

- Defeat encrypted file systems with keyloggers.
- Augment network sniffers with traffic analysis

Research directions in Computer Forensics

Environmental Data Survey Projects

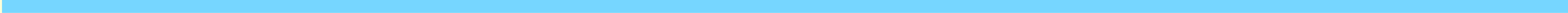
- Phone systems
- Hard drives & data storage devices
- Network hosts and traffic

Theory and Algorithm Development:

- Theoretical basis to forensics (Brian Carrier 2006 PhD)
- Cross-Drive Analysis (Garfinkel)
- Carving Fragmented Objects with Validation

Tool Development

- Easy-to-use tools
- Batch tools
- Data correlation



Conclusion

Forensic analysis is a growth area.

- You can do a lot, even if you don't understand it all.
- The law will get you if you don't watch out.

We discussed three technical areas:

- Policy
- Unicode & File Formats
- Disk Forensics
- Memory Forensics.

Anti-forensics are troubling, and are going to get worse.

Other Resources

US DoJ Computer Crime & Intellectual Property Section:

- <http://www.cybercrime.gov/>

Wiki:

- <http://www.forensicswiki.org/>

Blogs and Communities:

- <http://computer.forensikblog.de/en/>
- <http://sansforensics.wordpress.com/>


Link Farms

- <http://staff.washington.edu/dittrich/forensics.html>

Academic Program

University of Central Florida National Center for Forensic Science

- <http://ncfs.ucf.edu/>
- ["A Guide for Planning and Implementing a Computer Forensic Unit"](#)
- ["A Managers Guide for a Computer Forensic Unit"](#)



**Please fill
out your
evaluations.**