Automated Document and Media Exploitation

Simson Garfinkel, Ph.D.
Associate Professor
Naval Postgraduate School
http://faculty.nps.edu/slgarfin/

NPS is the Navy’s Research University.

Location: Monterey, CA
Campus Size: 627 acres
Students: 1500
- US Military (All 5 services)
- US Civilian (Scholarship for Service & SMART)
- Foreign Military (30 countries)
- All students are fully funded

Schools:
- Business & Public Policy
- Engineering & Applied Sciences
- Operational & Information Sciences
- International Graduate Studies
Current Projects:
- High-speed AES implementation on CellBE SPUs
  - 6 cores at a time on Sony PS3!
- User-level ATA command generator for Linux.
  - For testing ATA/SATA write-blockers
- Corpora for Forensic Research
  - Real Data Corpus (2000 “images” of hard drives, SD cards, etc.)
  - Realistic Corpus (constructed disk images)
- Sector Discrimination Project
  - What can you say about a 4096-byte sector/cluster from a hard drive?
    - A piece of a known file?
    - A JPEG / HTML / ZIP / DOC / DOCX?
    - Encrypted?
Law enforcement & military actives encounter lots of digital documents and media.

Most of this data is analyzed using trained personnel and off-the-shelf software.
Software is mostly COTS GUI tools designed for law enforcement.

- Designed for visibility & search.
- Does not scale to 100s or 1000s of drives.

Manual analysis misses opportunities for correlation.

Different analysts see different hard drives.

Keyword searches don’t connect the dots.
Tools designed for Law Enforcement do not use data once the investigation is over.

Bad guy goes to prison.  
Hard drive goes to storage.

"Analysts in a tent" do not scale.
We are building tools for performing Automated DOMEX.

Unrestricted Research Corpus
- Several thousand disk & device images

New techniques & algorithms
- Designed to exploit a data rich environment.
- Designed for autonomous operation.

Legal approaches for working with data
- “Realistic Data” — For tool testing & education.
- “Redistributable Real Data” — real but unrestricted.
- “Real Data” — For advanced research.
- IRB (Institutional Review Board) approaches and protocols.

We are building...
... an unrestricted research corpus.

Hard drives, USB sticks, Digital Cameras & cell phones:
- Purchased outside the US on the secondary market.
- Available to qualified researchers.

Multiple Corpora:
- Non-US Persons Corpus
- US Persons Corpus (at Harvard)
- “Realistic” Corpus (No PII)
We are developing...
... automated DOMEX algorithms.

Ascription:
- Batch analysis of drive contents
- Automated determination of owner & source
- Attribution of carved data

Hot Drive Identification:
- Automated triage without using keywords
- “Datasphere”–aware

Cross-Drive Analysis:
- Reconstruction of social networks

End-to-End automated analysis can increase exploitation capabilities and connect the dots.

Who? Contacts?
Anything unusual?
What was done?
Anything Encrypted?

Datasphere Repository
The Real Data Corpus

"Real Data from Real People."

Most forensic work is based on “realistic” data created in a lab.

We get real data from CN, IN, IL, MX, and other countries.

Real data provides:
- Real-world experience with data management problems.
- Unpredictable OS, software, & content
- Unanticipated faults

We have multiple corpora:
- Non-US Persons Corpus
- Releasable Real Corpus
- Realistic Corpus
We image physical devices to disk image files.

image.aff

100MB - 40GB
The images are stored in the Advanced Forensic Format (AFF)

**AFF extensible schema stores:**
- Data copied from disk
- Metadata (SN, date of image)
- Chain-of-custody information

**AFF supports:**
- Compression
- Public Key Encryption & Digital Signatures
- Incremental file transfer

**AFF Adoption:**
- Sleuth Kit
- Black Bag (Mac Forensics)
- Others have expressed interest

**AFF is designed for automating media exploitation**

To use the Real Data corpus, you need IRB approval

US Law requires approval of an Institutional Review Board to work with human subject data [45 CFR 46]
- Clearly describe what you want to do.
- Get submit your protocol to your local IRB.
- Provide your protocol and your approval.
- Approval must be renewed every year.

No approval required for “Realistic” data.
Automated Exploitation: Disk Images to XML

XML makes forensic analysis easier.

There are many forensic tools.
Many are programmable
- EnCase — EScript
- PyFlag — Flash Script & Python
- Sleuth Kit — C/C++

But *writing programs* for these systems is hard:
- Programming languages are *procedural* and *mechanism-oriented*
- Data is separated from actions on the data.
- Many of the programs are not designed for easy automation.
XML is ideally suited for forensic analysis

Forensic data is tree-structured.
- Case > Devices > Partitions > Directories > Files
  - Files
    - file system metadata
    - file meta data
    - file content
  - Container Files (ZIP, tar, CAB)
    - We can exactly represent the container structure
    - PyFlag does this with “virtual files”
    - No easy way to do this with the current TSK/EnCase/FTK structure
      -(Note: Container files not currently implemented.)

fiwalk extracts disk images into XML.

XML usage:

$ fiwalk [-Xfile.xml] [-x] imagefile
  -X file.xml produces an XML file with a full DTD
  -x sends XML to standard out

Other options:
- -c config.txt — metadata extraction plug-ins
- -C nn — only process nn files
- -S — single threaded
- -z — don’t calculate MD5 or SHA1
- -s dir — save all files in dir
- -Afile.arff — creates an ARFF file with output
- -Bfile — bloomfilter support
fiwalk produces three kinds of XML tags.

Per-Image tags

```xml
<fiwalk> — outer tag
<fiwalk_version>0.4</fiwalk_version>
<Start_time>Mon Oct 13 19:12:09 2008</Start_time>
<Imagefile>dosfs.dmg</Imagefile>
<volume startsector="512">

Per <volume> tags:

```
<Partition_Offset>512</Partition_Offset>
<block_size>512</block_size>
<ftype>4</ftype>
<ftype_str>fat16</ftype_str>
<block_count>81982</block_count>
```

Per <fileobject> tags:

```xml
<filesize>4096</filesize>
<partition>1</partition>
<filename>linedash.gif</filename>
<libmagic>GIF image data, version 89a, 410 x 143</libmagic>
```

fiwalk XML example

```xml
<fileobject>
<filename>WINDOWS/system32/config/systemprofile/「开始」菜单/程序/附件/_rf55.tmp</filename>
<filesize>1391</filesize>
<unalloc>1</unalloc>
<used>1</used>
<mtime>1150873922</mtime>
<ctime>1160927826</ctime>
<atime>1160884800</atime>
<fragments>0</fragments>
<md5>d41d8cd98f00b204e9800998ecf8427e</md5>
<sha1>da39a3ee5e6b4b0d3255bfef95601890afd80709</sha1>
<partition>1</partition>
-byte_runs type='resident'>
  <run file_offset='0' len='65536'
    fs_offset='871588864' img_offset='871621120'/>
  <run file_offset='65536' len='25920'
    fs_offset='871748608' img_offset='871780864'/>
</byte_runs>
</fileobject>
```
The `<byte_runs>` array specifies the physical location on the disk.

One or more `<run>` elements may be present:

```
<byte_runs type='resident'>
  <run file_offset='0' len='65536'
       fs_offset='871588864' img_offset='871621120'/>
  <run file_offset='65536' len='25920'
       fs_offset='871748608' img_offset='871780864'/>
</byte_runs>
```

This file has two fragments:
- 64K starting at sector 1702385 (871621120 ÷ 512)
- 25,920 bytes starting at sector 1702697 (871780864 ÷ 512)

Additional XML attributes may specify compression or encryption.

---

**fiwalk has a plugable metadata extraction system**

Metadata extractors are specified in the *configuration file*

```
*.jpg   dgi     ../plugins/jpeg_extract
*.pdf   dgi     java -classpath plugins.jar Libextract_plugin
```

- Currently the extractor is chosen by the file extension
- fiwalk runs the plugins in a different process
- We have specs for a native JVM interface which uses IPC and 1 process.

Metadata extractors produce name:value pairs on STDOUT

Manufacturer: SONY
Model: CYBERSHOT
Orientation: top - left

fiwalk incorporates into XML:

```
<fileobject>
  ...
  <Manufacturer>SONY</Manufacturer>
  <Model>CYBERSHOT</Model>
  <Orientation>top - left</Orientation>
  ...
</fileobject>
```
fiwalk.py: a Python module for automated forensics.

Key Features:
- Automatically runs fiwalk with correct options if given a disk image
- Reads XML file if present (faster than regenerating)
- Creates fileobject objects.

Multiple interfaces:
- SAX callback interface
  \[ \text{fiwalk}\_\text{using}\_\text{sax} (\text{imagefile}, \text{xmlfile}, \text{flags}, \text{callback}) \]
  -- Very fast and minimal memory footprint

- SAX procedural interface
  \[ \text{objs = fileobjects}\_\text{using}\_\text{sax} (\text{imagefile}, \text{xmlfile}, \text{flags}) \]
  -- Reasonably fast; returns a list of all file objects with XML in dictionary

- DOM procedural interface
  \[ (\text{doc, objs}) = \text{fileobjects}\_\text{using}\_\text{dom} (\text{imagefile}, \text{xmlfile}, \text{flags}) \]
  -- Allows modification of XML that's returned.

The SAX and DOM interfaces both return fileobjects!

Our Python fileobject class is an easy-to-use abstract class for working with file system data.

\[
\begin{align*}
\text{fileobject}\_\text{sax}(\text{fileobject}) & \quad \text{-- for the SAX interface} \\
\text{fileobject}\_\text{dom}(\text{fileobject}) & \quad \text{-- for the DOM interface}
\end{align*}
\]

Both classes support the same interface.
- fi.partition()
- fi.filename(), fi.ext()
- fi.filesize()
- fi.ctime(), fi.atime(), fi.crtime(), fi.mtime()
- fi.sha1(), fi.md5()
- fi.byteruns(), fi.fragments()
Example: calculate average file size on a disk

Using DOM interface:

```python
import fiwalk

objs = fileobjects_using_sax(imagefile, xmlfile, flags)
print "average file size: ",sum([fi.filesize() for fi in objs]) / len(objs)
```

Or, for the Python-impaired:

```python
import fiwalk

objs = fileobjects_using_sax(imagefile, xmlfile, flags)
sum_of_sizes = 0
for fi in objs:
    sum_of_sizes += fi.filesize()
print "average file size: ",sum_of_sizes / len(objs)
```

Example: Find and print all the files 15-bytes in length.

Using DOM interface:

```python
import fiwalk

objs = fileobjects_using_sax(imagefile, xmlfile, flags)
for fi in filter(lambda x:x.filesize()==15, objs):
    print fi
```

Or, for the Python-impaired:

```python
import fiwalk

objs = fileobjects_using_sax(imagefile, xmlfile, flags)
for fi in objs:
    if fi.filesize()==15:
        print fi
```
The fileobject class allows direct access to file data.

byteruns() is an array of “runs.”

\[
\begin{align*}
\text{byteruns} & \text{type='resident'} \\
\text{<run file_offset='0' len='65536' fs_offset='871588864' img_offset='871621120'/>} \\
\text{<run file_offset='65536' len='25920' fs_offset='871748608' img_offset='871780864'/>} \\
\end{align*}
\]

Becomes:

\[
[[0, 65536, 871621120], [65536, 25920, 871780864]]
\]

Each run has:

- \textit{run[0]} — Start byte offset within the file
- \textit{run[1]} — Number of bytes
- \textit{run[2]} — Start offset from the beginning of the disk

byteruns() returns that array of “runs” for both the DOM and SAX-based file objects.

>>> print fi.byteruns()
[[0, 65536, 871621120],
[65536, 25920, 871780864]]

Accessor Methods:

- \texttt{fi.contents\_for\_run(run)} — \textit{Returns the bytes from the linked disk image}
- \texttt{fi.contents()} — \textit{Returns all of the contents}
- \texttt{fi.file\_present(imagefile=None)} — Validates MD5/SHA1 to see if image has file
- \texttt{fi.tempfile(calMD5,calcSHA1)} — Creates a tempfile, optionally calculating hash
We are building several interconnected applications with this framework.

imap.py
- reads a disk image or XML file and prints a “map” of a disk image.

igroundtruth.py
- reads multiple disk images (different generations of the same disk)
- uses earlier images as “maps” for later images.
- Outputs new XML file

iverify.py
- Reads an image file and XML file.
- Reports which files in the XML file are actually resident in the image.

iredact.py
- reads a disk image (or XML file) and a “redaction file”
- Produces new disk image.

The redaction language is flexible.

Language: \{CONDITION\} \{ACTION\}

Conditions:
- FILENAME filename
- FILEPAT file*name
- DIRNAME dirname/
- MD5 d41d8cd98f00b204e9800998ecf8427e
- SHA1 da39a3ee5e6b4b0d3255bfef95601890af05250e
- FILE CONTAINS user@company.com
- SECTOR CONTAINS user@company.com

Actions:
- FILL 0x44
- ENCRYPT
- FUZZ (changes instructions but not strings)
We have also built a USB transfer kiosk.

The kiosk:
- Reads a USB drive using fiwalk & fiwalk.py
- Displays list of files in GUI
- Transfers selected files to “quarantine” without mounting the disk image.
- Virus scans
- Transfers scanned files to SMB server without mounting the file server.

Key features:
- Functionality could not be implemented without forensic tools
- fiwalk & fiwalk.py allows forensics to be abstracted away
- Kiosk program is mostly GUI, not forensics
  - filelist.py — 110 lines
  - kiosk.py — 368 lines
  - loginpanel.py — 70 lines
  - smb.py — 90 lines
  - watcher.py — 152 lines

Publishing XML for disk images enables our remote exploitation methodology...

Extract metadata in Boston.
Search from Monterey.
Just download what you need.
Publishing XML for disk images enables our remote exploitation methodology...

Extract metadata in Boston.
Search from Monterey.
Just download what you need.
Publishing XML for disk images enables our remote exploitation methodology...

Extract metadata in Boston.
Search from Monterey.
Just download what you need.
Publishing XML for disk images enables our remote exploitation methodology...


Realistic Images
We have created six disk images.

IMAGE 1 - CANON2 (FAT32)
IMAGE 2 - NTFS1 (NTFS)
IMAGE 3 - UBNIST1 (FAT32)
IMAGE 4 - UBNIST1 (embedded EXT3)
IMAGE 5 - DOMEXUSERS (NTFS)
IMAGE 6 - HFSNIST1 (HFS+)

Each image has:
- Narrative of how the image was created and expected uses.
- Image file in RAW/SPLITRAW, AFF and E01 formats
- SHA1 of raw image
- “Ground truth” report

“Disk” is a 32MB SD card shot in a Canon camera.

All operations carried out by camera:
- Disk formatting (-o51 !)
- JPEG creation
- JPEG deletion

Disk was repeatedly removed from camera & imaged.
JPEGs were created and deleted in such a way to assure:
- Fragmented files
- Files that can only be recovered through file carving (name overwritten, not data.)
nps-2009-canon2 is boring...

... perhaps not so boring

It’s hard to avoid placing information in images...
... fortunately nothing here is a problem.
Three directories:
  d/d 28-144-8: Compressed
  d/d 29-144-10: Encrypted
  d/d 27-144-7: RAW

EFS key information:
  r/r 46-128-1: EFS-key-info.txt
  r/r 43-128-4: EFS-key-no-password.pfx
  r/r 45-128-4: EFS-key-password-strong-protection.pfx
  r/r 44-128-4: EFS-key-password.pfx

The same files are in each directory:
  r/r 42-128-1: RAW/20076517123273.pdf
  r/r 47-128-0: RAW/logfile1.txt
  r/r 36-128-1: RAW/NISTSP800-88_rev1.pdf
  r/r 33-128-1: RAW/NIST_logo.jpg
  r/r 39-128-1: RAW/report02-3.pdf

No existing program will automatically identify the key and then try it on the encrypted files!

The “logfile.txt” file was written a line-at-a-time to each directory and is fragmented...

<fileobject>
<filename>RAW/logfile1.txt</filename>
<filesize>21888890</filesize>
<partition>1</partition>
<ALLOC>1</ALLOC>
<USED>1</USED>
<mtime>1231192883</mtime>
<ctime>1231192883</ctime>
<atime>1231192883</atime>
<crtime>1231192820</crtime>
<libmagic>ASCII text, with CRLF line terminators</libmagic>
<byte_runs type='resident'>
  <run fs_offset='237428736' img_offset='237428736' file_offset='0' len='1024'/>
  <run fs_offset='243657728' img_offset='243657728' file_offset='1024' len='3072'/>
  <run fs_offset='240057344' img_offset='240057344' file_offset='4096' len='5120'/>
</byte_runs>

1628 fragments in all!

This is a realistic model for the writing of log files.
UBNIST1 is a bootable USB flash drive running Ubuntu 8.10.

The “outer” file system is FAT32:

```bash
$ fls -o63 ubnist1-2009-01-07.aff
r/r 4:  ldlinux.sys
d/d 6:  casper
d/d 8:  dists
d/d 10:  install
r/r 12:  syslinux.cfg
d/d 14:  pics
d/d 16:  pool
d/d 18:  preseed
d/d 20:  .disk
r/r 22:  autorun.inf
r/r 24:  md5sum.txt
r/r 27:  README.diskdefines
r/r 29:  umenu.exe
r/r 31:  wubi.exe
d/d 33:  syslinux
r/r 35:  casper-rw
```

Inside UBNIST1 is an ext3 file system:

```bash
$ icat -o63 ubnist1.gen2.aff 35 > ubnist1.gen2.casper-rw
$ fls ubnist1.gen2.casper-rw
```

```bash
d/d 11:  lost+found
r/r 12:  .wh..wh.aufs
d/d 7681:  .wh..wh.plnk
d/d 23041:  .wh..wh..tmp
d/d 7682:  rofs
d/d 23042:  etc
d/d 23044:  cdrom
d/d 7683:  var
d/d 15361:  home
d/d 30721:  tmp
d/d 30722:  lib
d/d 15377:  usr
d/d 7712:  sbin
d/d 13:  root
r/r * 20(realloc):  .aufs.xino
```

```bash
$OrphanFiles
```
Image created by LT Paul Farrell as part of his master’s thesis.

```bash
$ mmls realistic.aff
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

<table>
<thead>
<tr>
<th>Slot</th>
<th>Start</th>
<th>End</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:</td>
<td>Meta</td>
<td>0000000000</td>
<td>0000000000</td>
<td>0000000001 Primary Table (#0)</td>
</tr>
<tr>
<td>01:</td>
<td>-----</td>
<td>0000000000</td>
<td>0000000062</td>
<td>Unallocated</td>
</tr>
<tr>
<td>02:</td>
<td>00:00</td>
<td>0083859299</td>
<td>0083859237</td>
<td>Unallocated NTFS (0x07)</td>
</tr>
<tr>
<td>03:</td>
<td>-----</td>
<td>0083859300</td>
<td>0083886079</td>
<td>Unallocated</td>
</tr>
</tbody>
</table>
```

Contents: NTFS Windows XP installation.
Two users: domex1 and domex2
Email, Chat, limited web browsing.

There is a lot of work to do with this image before it is released:

- Redact Microsoft © data.
- Verify that image contains no PII or other copyrighted data.
- Detailed characterization report.
This is a very simple image:

- Two files — file1.txt and file2.txt:

File1 has two sets of contents

- “This is file 1 - snarf”
- “New file 1 contents - snarf”

The first set of contents is in the HFS journal.

Automated Ascription of Multi-User Data
Problem: How do you ascribe carved data to a particular person?

Alice, Bob, Carol and Dave all share a computer:

File carving reveals “contraband information.”

Who put the data on the disk?

Prior work has used content analysis datamining to determining authorship

For example:

- Alice always misspells “kill” as “kilz”
- Everything Bob writes is boring
Prior work has used content analysis datamining to determining authorship

For example:

- Alice always misspells “kill” as “kilz”

- Everything Bob writes is boring
This research uses metadata to infer *ownership* or *agency* — who is *responsible* for the data.

Available sector metadata (“extrinsic”):
- Fragmentation patterns (disk usage)
- Where the file is on the hard drive (sector numbers)

Available file metadata (“intrinsic”):
- When the file was created (internal timestamps)
- Make & model of digital cameras
- Usage patterns.

Let’s look at the hard drive:

Whose data is this?
Alice's information is all in
/Documents and Settings/Alice

All Alice's data is green.

All data is
3 blocks long.

Bob's information is in
/Documents and Settings/Bob

Bob's information is all monochrome.

It's all labeled "Bob"

It's kind of boring.
Carol's information is in
/Documents and Settings/Carol

Carol's information is all JPEGs

It's all little dots

Dave's information is all in
/Documents and Settings/Dave

It's all jumbled.
But a lot of the data starts two blocks in....

Some of the data starts
10 blocks in.
So who is responsible for the contraband data?

Dave is responsible for the carved data.

Carved data has more in common with Dave's data and violates others' rules.

-✓ starts two blocks over.
-✗ It's not a JPEG.
-✗ It's not green.
-✗ It's not monochrome
We are developing a system for ascribing carved information on multi-user systems.

**Inputs 1:**
- One or more disk images
- A rule for ascribing the exemplars.
  Here the rule was `/Documents and Settings/username/*`
- Plug-in metadata extractors
We are developing a system for ascribing carved information on multi-user systems.

Inputs 1:
- One or more disk images
- A rule for ascribing the exemplars.
  Here the rule was /Documents and Settings/username/*
- Plug-in metadata extractors

Outputs 1:
- A rule-based system for ascribing files
- The accuracy of the system using N-fold validation

Input 2:
- Carved data
We are developing a system for ascribing carved information on multi-user systems.

Inputs 1:
- One or more disk images
- A rule for ascribing the exemplars. Here the rule was /Documents and Settings/username/*
- Plug-in metadata extractors

Outputs 1:
- A rule-based system for ascribing files
- The accuracy of the system using N-fold validation

Input 2:
- Carved data

Output 2:
- Ascribed individual & confidence of match

Key points of data ascription

The accuracy rates we have seen are phenomenal:
- 98.5% on laboratory data
- 99% - 100% on real data.
- Mostly based on sector number & embedded timestamps
- Classification Algorithm: J48 / C4.5

This technique relies upon exemplars to train the machine learning algorithm.
- You need a multi-user system;
- It could be applied to USB memory sticks, if each one identifies to a different user.
Cross-Drive Analysis

Making use of the datasphere

The **datasphere** is information:
- on devices taken from people-of-interest
- on remote systems used by people-of-interest
- on devices and systems co-located in time and space with these systems.

Examples:
- All of the laptops, desktops and servers in the Washington DC area.

**Hypothesis 1**: *Datasphere sampling* can build better single-drive analysis tools.

**Hypothesis 2**: *Datasphere correlation* can identify and characterize previously unknown social networks.
1. Media is collected in theater

2. Some drives belong to IED-planting cells
Network Determination: The Big Idea

1. Media is collected in theater

2. Some drives belong to IED-planting cells

3. Automatically identify links

4. Identify new network members.
Network Determination: The Big Idea

1. Media is collected in theater
2. Some drives belong to IED-planting cells
3. Automatically identify links
4. Identify new network members.
5. New drive arrives
6. Find links with existing drives
Network Determination: The Big Idea

1. Media is collected in theater
2. Some drives belong to IED-planting cells
3. Automatically identify links
4. Identify new network members.
5. New drive arrives
6. Find links with existing drives
7. Determine network membership

"Datasphere"
We efficiently process the datasphere by separating metadata, identities, and content.

file names, email messages, etc.
We efficiently process the datasphere by separating metadata, identities, and content.

file names, email messages, etc.

human names
photographs
camera serial numbers

Data space and feature extraction tools

Two tools for data space and feature extraction:

Fiwalk (File and inode Walk):
1. Ingests disk images (raw, EnCase, or AFF format)
2. Automatically extracts metadata (EXIFs, Microsoft Office Doc Properties)
3. Outputs in ASCII, XML, CSV, or ARFF format

“Bulk Extractor”
1. Processes any disk, even damaged/incomplete
2. Scans for email addresses.
3. Outputs list of email addresses & count (sorted by frequency)
Hot Drive Detection:
Automatically finding “the good stuff.”

We were interested in showing that sensitive information had been left behind on hard drives.

We wrote a Credit Card Number detector.

Bulk Data

CCN Detector: written in flex and C++

Disk #105:

<table>
<thead>
<tr>
<th>Test</th>
<th># pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>pattern</td>
<td>3857</td>
</tr>
<tr>
<td>Known prefixes</td>
<td>90</td>
</tr>
<tr>
<td>CCV1</td>
<td>43</td>
</tr>
<tr>
<td>patterns &amp; histogram</td>
<td>38</td>
</tr>
</tbody>
</table>

Sample output:

CHASE NA 5422-4128-3008-3685 | pos=13152133
DISCOVER 6011-0052-8056-4504 | pos=13152440
GE CARD 4055-9000-0378-1959 | pos=13152589
BANK ONE 4332-2213-0038-0832 | pos=13152740
NORWEST 4829-0000-4102-9233 | pos=13153182
CNB CARD 5419-7213-0101-3624 | pos=13153332
Histogram analysis: 
importance on multiple drives is $k/(\text{drive count})$

Experiment: 429 Drives  
Source Countries: China and Mexico  
Automatically Constructed Stop List:

<table>
<thead>
<tr>
<th>Email Address</th>
<th>Drive Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="mailto:cps-requests@verisign.com">cps-requests@verisign.com</a></td>
<td>121</td>
</tr>
<tr>
<td><a href="mailto:server-certs@thawte.com">server-certs@thawte.com</a></td>
<td>115</td>
</tr>
<tr>
<td><a href="mailto:premium-server@thawte.com">premium-server@thawte.com</a></td>
<td>115</td>
</tr>
<tr>
<td><a href="mailto:c2le@mz.um">c2le@mz.um</a></td>
<td>105</td>
</tr>
<tr>
<td><a href="mailto:mazrob@panix.com">mazrob@panix.com</a></td>
<td>98</td>
</tr>
<tr>
<td><a href="mailto:personal-premium@thawte.com">personal-premium@thawte.com</a></td>
<td>95</td>
</tr>
<tr>
<td><a href="mailto:inet@microsoft.com">inet@microsoft.com</a></td>
<td>95</td>
</tr>
<tr>
<td><a href="mailto:personal-basic@thawte.com">personal-basic@thawte.com</a></td>
<td>94</td>
</tr>
<tr>
<td><a href="mailto:personal-freemail@thawte.com">personal-freemail@thawte.com</a></td>
<td>93</td>
</tr>
<tr>
<td><a href="mailto:keywitness@keywitness.ca">keywitness@keywitness.ca</a></td>
<td>83</td>
</tr>
<tr>
<td><a href="mailto:icw5@gn.microsoft.com">icw5@gn.microsoft.com</a></td>
<td>79</td>
</tr>
<tr>
<td><a href="mailto:java-io@java.sun.com">java-io@java.sun.com</a></td>
<td>78</td>
</tr>
<tr>
<td><a href="mailto:java-security@java.sun.com">java-security@java.sun.com</a></td>
<td>78</td>
</tr>
</tbody>
</table>

4 “Subject” Drives withheld from ingest

Questions: Can we find which networks they associate with?  
Can we learn why they associate?

Results: Drive X5-22 correlates with X5-25 and X5-28

<table>
<thead>
<tr>
<th>Drive Pair</th>
<th>Correlation Coefficient (linear scale)</th>
</tr>
</thead>
<tbody>
<tr>
<td>X5-22 and X5-25</td>
<td><strong>9.25</strong></td>
</tr>
<tr>
<td>X5-22 and X5-28</td>
<td><strong>9.13</strong></td>
</tr>
<tr>
<td>X5-22 and X5-27</td>
<td><strong>6.86</strong></td>
</tr>
<tr>
<td>X5-22 and X5-26</td>
<td><strong>5.95</strong></td>
</tr>
<tr>
<td>X5-22 and X5-03</td>
<td><strong>3.43</strong></td>
</tr>
<tr>
<td>X5-22 and X2-09</td>
<td><strong>2.20</strong></td>
</tr>
<tr>
<td>X5-22 and X3-07</td>
<td><strong>2.10</strong></td>
</tr>
<tr>
<td>X5-22 and X3-21</td>
<td><strong>2.08</strong></td>
</tr>
</tbody>
</table>

Most drive pairs have CC $<< 0.1$
TF/IDF: Those most important email addresses are those seen on fewest # of drives

<table>
<thead>
<tr>
<th>Email Address (Sanitized)</th>
<th>#</th>
<th>Correlation Contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="mailto:jdelacerXX@sedesXX.gXX.X">jdelacerXX@sedesXX.gXX.X</a></td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td><a href="mailto:administraXX@google.com">administraXX@google.com</a></td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td><a href="mailto:avegXX@sedesXX.gXX.X">avegXX@sedesXX.gXX.X</a></td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td><a href="mailto:jcoronXX@sedesXX.gXX.X">jcoronXX@sedesXX.gXX.X</a></td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td><a href="mailto:guiraXXXX@hotmail.com">guiraXXXX@hotmail.com</a></td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td><a href="mailto:eescobeXXXX@sedesXX.gXX.X">eescobeXXXX@sedesXX.gXX.X</a></td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td><a href="mailto:lperXXXXX@sedesXX.gXX.X">lperXXXXX@sedesXX.gXX.X</a></td>
<td>3</td>
<td>0.3</td>
</tr>
<tr>
<td><a href="mailto:n_indexXXXXX@sedesXX.gXX.X">n_indexXXXXX@sedesXX.gXX.X</a></td>
<td>3</td>
<td>0.1</td>
</tr>
<tr>
<td><a href="mailto:brivXXXXX@sedesXX.gXX.X">brivXXXXX@sedesXX.gXX.X</a></td>
<td>3</td>
<td>0.1</td>
</tr>
<tr>
<td><a href="mailto:anyuse30000@www.mercadolibre.com">anyuse30000@www.mercadolibre.com</a></td>
<td>3</td>
<td>0.3</td>
</tr>
<tr>
<td><a href="mailto:shernandXXXXX@sedesXX.gXX.X">shernandXXXXX@sedesXX.gXX.X</a></td>
<td>4</td>
<td>0.25</td>
</tr>
<tr>
<td><a href="mailto:demandasociXXXXX@sedesXX.gXX.X">demandasociXXXXX@sedesXX.gXX.X</a></td>
<td>4</td>
<td>0.25</td>
</tr>
<tr>
<td><a href="mailto:bourXXX@mygale.org">bourXXX@mygale.org</a></td>
<td>4</td>
<td>0.25</td>
</tr>
<tr>
<td><a href="mailto:anyuse3000@terra.com">anyuse3000@terra.com</a></td>
<td>4</td>
<td>0.25</td>
</tr>
<tr>
<td><a href="mailto:ivanpetroXXX@hotmail.com">ivanpetroXXX@hotmail.com</a></td>
<td>4</td>
<td>0.25</td>
</tr>
<tr>
<td><a href="mailto:notificationXXXXX@groups.msn.com">notificationXXXXX@groups.msn.com</a></td>
<td>5</td>
<td>0.2</td>
</tr>
<tr>
<td><a href="mailto:dOO@inhtlt.net">dOO@inhtlt.net</a></td>
<td>5</td>
<td>0.2</td>
</tr>
<tr>
<td><a href="mailto:invXXXXXX@rincondelvago.com">invXXXXXX@rincondelvago.com</a></td>
<td>5</td>
<td>0.2</td>
</tr>
<tr>
<td><a href="mailto:pnrXXXXX@po.pl">pnrXXXXX@po.pl</a></td>
<td>5</td>
<td>0.2</td>
</tr>
<tr>
<td><a href="mailto:notificarolestevariaXXXXX@osmas.com.X">notificarolestevariaXXXXX@osmas.com.X</a></td>
<td>5</td>
<td>0.2</td>
</tr>
<tr>
<td>TOTAL</td>
<td>9.25</td>
<td></td>
</tr>
</tbody>
</table>

Most common domain reveals organization of previous owner

Network Membership Tool

Cross Drive Analysis (with email addresses)
1. Rank all email addresses by drive frequency
2. Top 1/3 most popular email addresses are placed on a stop list.
3. Rate email addresses by inverse drive frequency
   - More common email addresses are less important

Network Determination Tool
1. Extract email addresses from Subject drive
2. Compute score for each email address with other drives in datasphere
3. Report highest-scoring drive matches
   - Show email addresses most responsible for drive score.
Needed: Visualization Approaches

This visualization plots $C(d_1,d_2)$ for all drives:

- Drives #74 & #77: 25 CCNS in common
- Drives #171 & #172: 13 CCNS in common
- Drives #179 & #206: 13 CCNS in common

Use encrypted bloom filters for secure field deployment.

**Encrypted Bloom filter tool:**
- Background BF created from non-Network drives
- In-Network drives are processed with tool:
  - Background features removed
  - Remaining features produce in-Network BF
- Captured drives are filtered against in-Network BF:
  - Presence of features indicates network membership.

**Advantages:**
- Searches all drives simultaneously
- Bloom filter cannot be reverse-engineered by enemy

✓ “Practical Applications of Bloom filters to the NIST RDS and hard drive triage,” Farrell, Garfinkel & White, ACSAC 2008
Different targets will have different behavioral profiles.

Behavioral templates can help identify the target.

Future work: searching the datasphere with behavior profiles and templates.

Future work: Clustering algorithms to identify "unknown unknowns."
Conclusions

Research Collaborations

Data-Sharing Partnerships:
- DC3
- MITRE
- I.D.E.A.L. Technology Corporation (STRIKE)
- Professor Patrick Wolfe, Harvard University, NSF EXP program
- Professor Lorrie Cranor, Carnegie Mellon University

Research Partnerships:
- FBI Silicon Valley Regional Computer Forensic Lab
- Police Department, San Louis Obispo, California
In Summary

Automated exploitation systems can be made more powerful by making them aware of the "Datasphere."

The key is to use every piece of available data—don’t stop with the obvious content.

Datamining algorithms can be profitably applied to forensic problems.

Questions?