bulk_extractor 1.5 overview

Simson L. Garfinkel

Overview:

- What is bulk_extractor?
- What can it do?
- How does it work?
- How do I run it?
- What's new in version 1.5?

Introducing bulk_extractor

bulk_extractor is a stream-based disk forensics tool. It scans the media and extracts recognizable content.

0 → 1TB

3 hours, 20 min to *read* the data

1. Read all of the blocks in order.
2. Look for information that might be useful.
3. Identify & extract what's possible in a single pass.

Before bulk_extractor, this was done with strings & grep

```
$ strings diskimage.dd | grep '[a-z]+*@[a-z0-9]+'
```

strings → grep → **user@gmail.com**
+ lots of stuff

Problems with this approach:

- Slow (not parallelized)
- Many false positives
- Each 'search' requires a complete scan of the device
- Misses encoded data.

bulk_extractor improves on strings & grep

Finds more kinds of data:

- Structured text (email addresses, URLs, etc)
- Structured binary data (Microsoft Windows PE files, LNK files, etc)

Finds data in many kinds of situations:

- Compressed & encoded data.
- Recursive re-analysis

Expandable:

- Easy to add new features.

“Encoded data” must frequently be *decoded* to be recognized.

Compression removes redundancies in data:

```
5859 5a40 636f 6d70 616e 792e 636f 6d20  XYZ@company.com
4142 4340 636f 6d70 616e 792e 636f 6d20  ABC@company.com
4445 4640 636f 6d70 616e 792e 636f 6d20  DEF@company.com
```

Compressed with “gzip:”

```
1f8b 0800 0000 0000 0203 8b88 8c72 48ce  .............rH.
cf2d 48cc abd4 03d2 0a8e 4ece 287c 1757  .-H.......N.(|.W
3714 3e00 b455 c1c5 3000 0000            7.>..U..0...
```

Compressed email addresses do not “look” like email addresses!

— *Forensic tools must “optimistically” decompress data to search for email addresses.*

Programs encode data data in many ways.

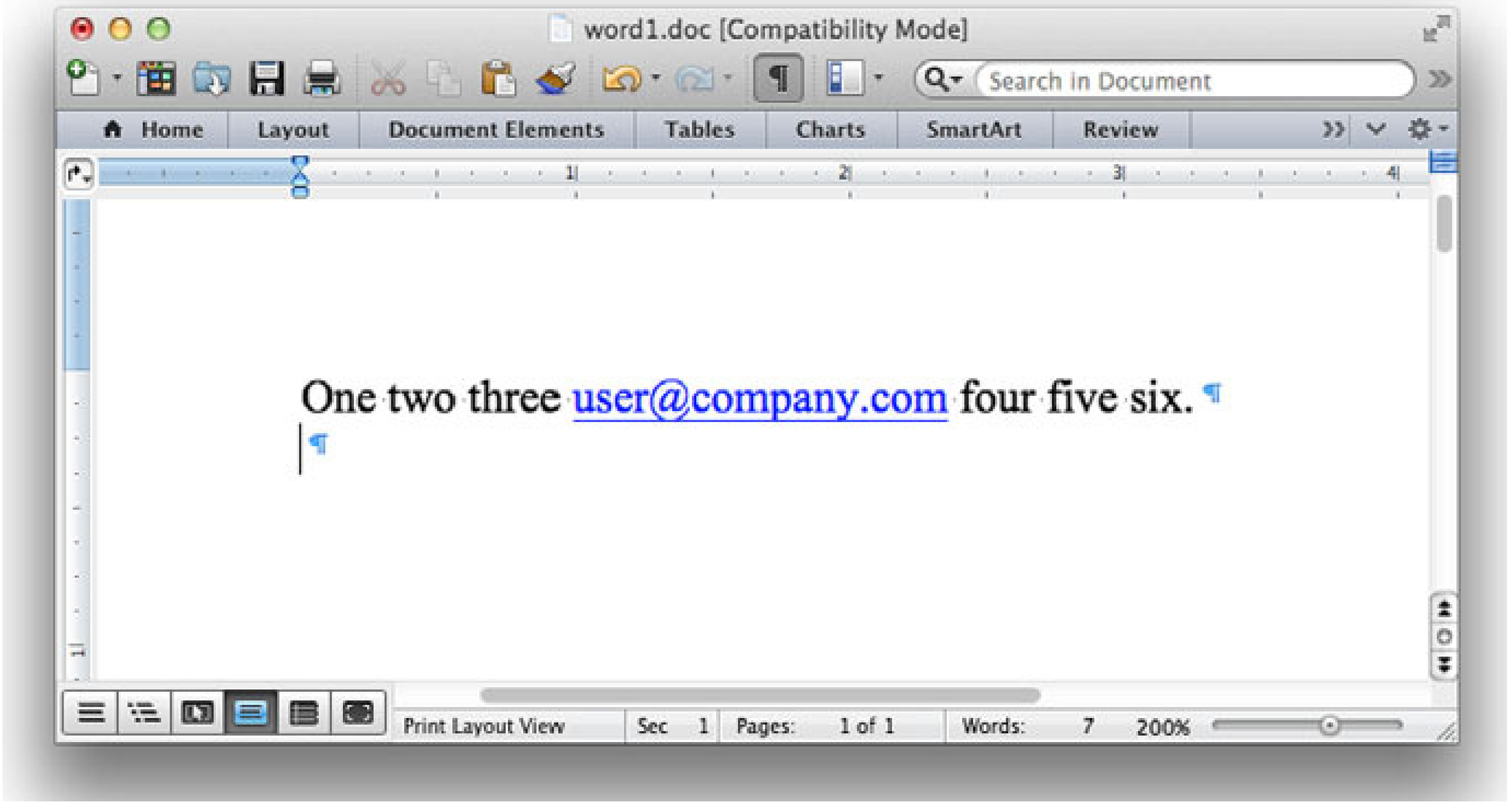
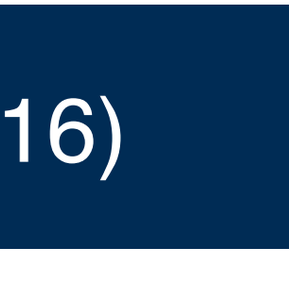


FIG. 1—*A Microsoft Word file containing a single sentence followed by a blank line.*

Word's .doc format stores plain text (UTF-8 and UTF-16)

```
00000a00: 4f6e 6520 7477 6f20 7468 7265 6520 1320  One two three .
00000a10: 4859 5045 524c 494e 4b20 226d 6169 6c74  HYPERLINK ‘‘mailt
00000a20: 6f3a 7573 6572 4063 6f6d 7061 6e79 2e63  o:user@company.c
00000a30: 6f6d 2220 1475 7365 7240 636f 6d70 616e  om’’ .user@compan
00000a40: 792e 636f 6d15 2066 6f75 7220 6669 7665  y.com. four five
00000a50: 2073 6978 2e0d 0d00 0000 0000 0000 0000   six.............
00000a60: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000a70: 0000 0000 0000 0000 0000 0000 0000 0000  ................
```



Word's .docx format stores content as compressed XML

```
00000990: 0300 504b 0304 1400 0600 0800 0000 2100  ..PK..........!.
000009a0: ea76 7d78 d702 0000 c607 0000 1100 0000  .v}x............
000009b0: 776f 7264 2f64 6f63 756d 656e 742e 786d  word/document.xm
000009c0: 6ca4 55db 729b 3010 7def 4cff 81d1 7b0c  l.U.r.0.}.L...{.
000009d0: 7673 7198 e034 b7a6 79e8 3453 b7cf 1d19  vsq..4..y.4S....
000009e0: 0468 8cb4 1a49 98ba 5fdf 95b8 d889 ddd6  .h...I.._.......
000009f0: 495e 0c98 b367 cf9e 5d2d 1797 bf44 15ac  I^...g..]-...D..
00000a00: 9836 1c64 42c6 a388 044c a690 7159 24e4  .6.dB....L..qY$.
00000a10: c7f7 4f47 5312 184b 6546 2b90 2c21 6b66  ..OGS..KeF+.,!kf
```

Uncompress

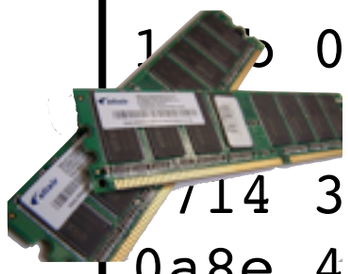
```
w:t></w:r><w:hyperlink r:id=''rId5'' w:history=''1''><w:r w:rsidRPr=
''004B377A''><w:rPr><w:rStyle w:val=''Hyperlink''/></w:rPr><w:t>user
@company.com</w:t></w:r></w:hyperlink><w:r><w:t
```

PDFs generated by Word are compressed PDF streams

```
%PDF-1.3
%\304\345\362\345\353\247\363\240\320\304\306
4 0 obj
<< /Length 5 0 R /Filter /FlateDecode >>
stream
x^A\225\222\313n\2030^PE\367\376\212\2734\213:\266^C^FvU\252n
\272\251''Y\352\242\352\242BAi^U\240\201\246\217\277\257\237
\224''\224\250^B\311^^{4>\367\316^\\261\305^QB\332?+\344\252
@\277\303^CZ\254n^F\201j^@w\337P\231<\316d\352c\273)9r^\260R
\241j\260\321$\363\231a\321^MVZ^K\306!\240k<\202\336'\2702^U
@\333]bK\201\342=l>\343U^W\216^H\3
\240\355[^B^KTBqV\346\251\372e#^[\
\313&@\253\300\3305      q\324o\31
_\202\223Y\311\224JE\200#\316\270\
\274^CR\311\377\2263}\250\235\324^
\303^[@(FK\342\325^W\233v'^N\263\2
^M\305T\333\330P\241\314\320\244\3
^H1\261\261I;'\222\357\342=j?\243K
^\\336\366^G\212q\250^D
endstream
endobj
```

```
q Q q 12 12 588 768 re W n /Cs1 cs 0 0 0 sc q 0.24 0 0 0.24 90 708.96
cm BT 50 0 0 50 0 0 Tm /TT1.0 1 Tf [ (O) -0.2 (ne) 0.2 ( t) 0.2 (w)
  -0.2 (o t) 0.2 (hre) 0.2 (e) 0.2 ( ) ] TJ ET Q 0 0 1 sc q 0.24 0 0
0.24 160.9746 708.96 cm BT 50 0 0 50 0 0 Tm /TT1.0 1 Tf [ (us) -0.2
  (e) 0.2 (r@) 0.1 (c) 0.2 (om) 0.2 (pa) 0.2 (ny.c) 0.2 (om) ] TJ ET Q
0 0 0 sc q 0.24 0 0 0.24 259.6641 708.96 cm BT 50 0 0 50 0 0 Tm /TT1.0
1 Tf ( ) Tj ET Q q 0.24 0 0 0.24 262.6641 708.96 cm BT 50 0 0 50 0 0
Tm /TT1.0 1 Tf [ (f) -0.5 (our f) -0.5 (i) 0.2 (ve) 0.2 ( s) -0.2 (i)
  0.2 (x.) ] TJ ET Q q 0.24 0 0 0.24 324.3281 708.96 cm BT 50 0 0 50 0
0 Tm /TT1.0 1 Tf ( ) Tj ET Q 0 0 1 sc 161.04 707.28 m 259.68 707.28 l
259.68 707.04 l 161.04 707.04 l h f 0 0 0 sc q 0.24 0 0 0.24 90 695.28
cm BT 50 0 0 50 0 0 Tm /TT1.0 1 Tf ( ) Tj ET Q Q
```

Encoded data may be in files or between files.



```
e327 962d 6450 3d91 c945 3bed 97a6 a4cd   .'.-dP=..E;.....
1[illegible]b 0800 0000 0000 0203 8b88 8c72 48ce   .............rH.
[illegible]8cc abd4 03d2 0a8e 4ece 287c 1757   .-H.......N.(|.W
[illegible]714 3e00 b455 c1c5 3000 0000 0000 0000   7.>..U..0.......
0a8e 4ece 287c 1757 3714 3e00 a175 10ed   ..N.(|.W7.>..u..
```

Folders.pst

Mother.JPG

Presentation.pptx

Sequestration.docx

```
a097 83a1 ed96 26a6 3c69 3d0f 750a 2399   ......&.<i=.u.#.
a2b5 bea7 692f 5847 a38a dd53 082c add5   ....i/XG...S.,..
5061 b64c 721d 864b 90b6 b55f bb04 735c   Pa.Lr..K..._..s\
9448 6730 5453 df64 813e b603 5795 2242   .Hg0TS.d.>..W."B
e[illegible]8 7454 7322 7cdc b60e 97af 2f64 2728   ..tTs"|...../d'(
[illegible]4bd 2a84 2dfe 50ea 5935 c349 1513   <XYZ@COMPANY.COM
[illegible]e92c a3f8 6e46 0530 8a88 c7a2 5d2b   ...,..nF.0....]+
d89d 77cc fe1e f637 f3f3 d0af 1b47 c09b   ..w....7.....G..
```

EnCase & FTK use Oracle's "Outside In" to extract text.

Outside In will extract text from:

- Word .doc
- Word .docx
- PDFs made by Word
- 500+ other file formats

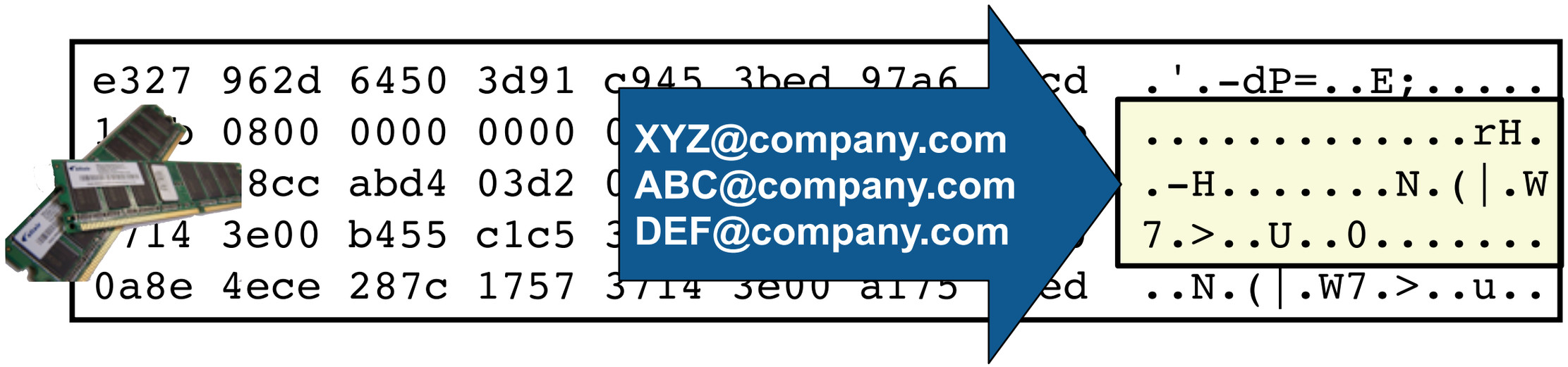
Folders.pst

Mother.JPG

Presentation.pptx

Sequestration.docx

Outside In won't extract text from non-file ("bulk") data.



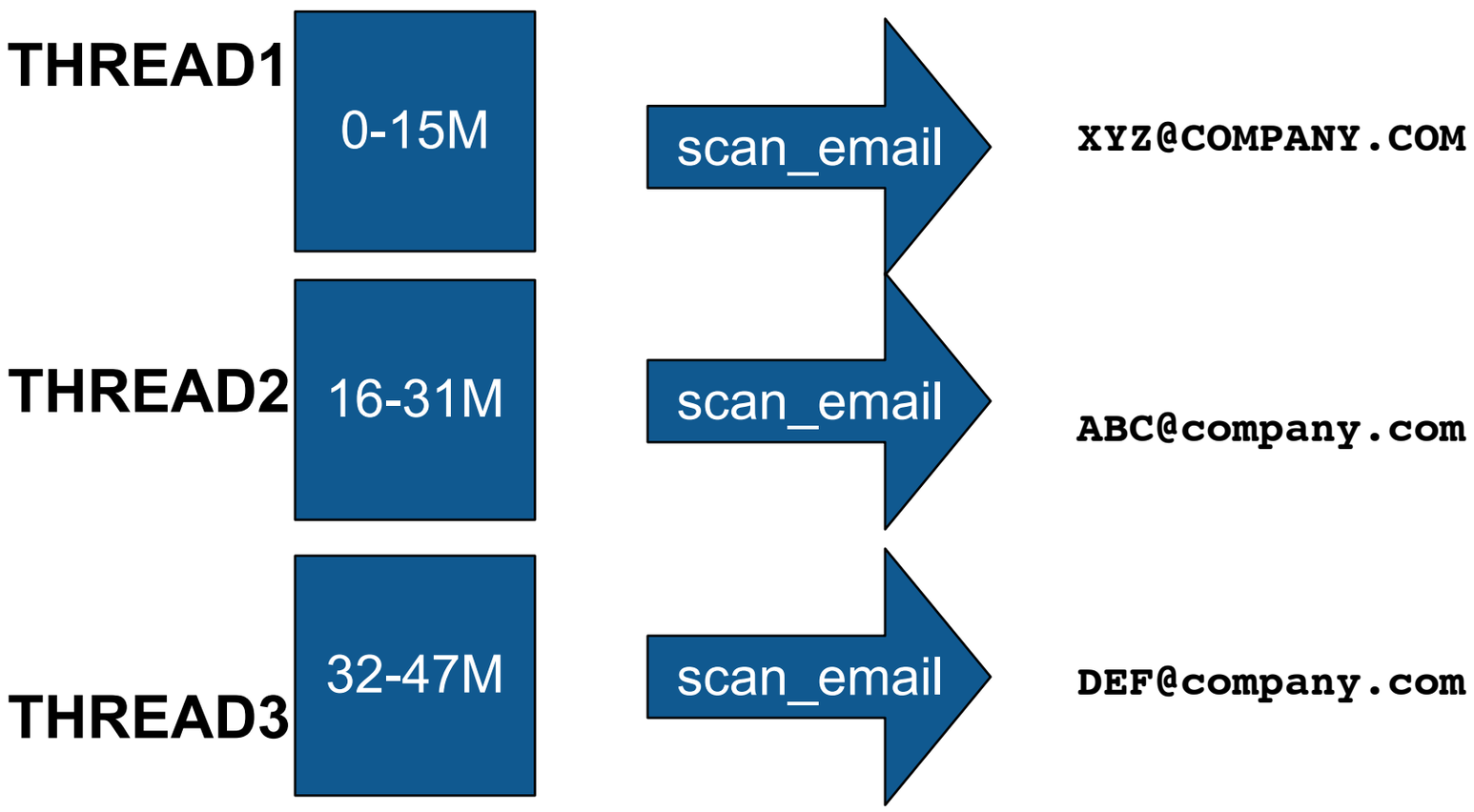
Outside In won't recognize the file type.

- The entire file may not be present.

Examples:

- Compressed — zlib (gzip, ZIP), RAR, Windows Hibernation (Microsoft Xpress)
- Encoded — BASE64
- Obfuscated — ROT13, XOR(255)

bulk_extractor splits the disk into 16M “pages” (blocks) and processes each page independently.

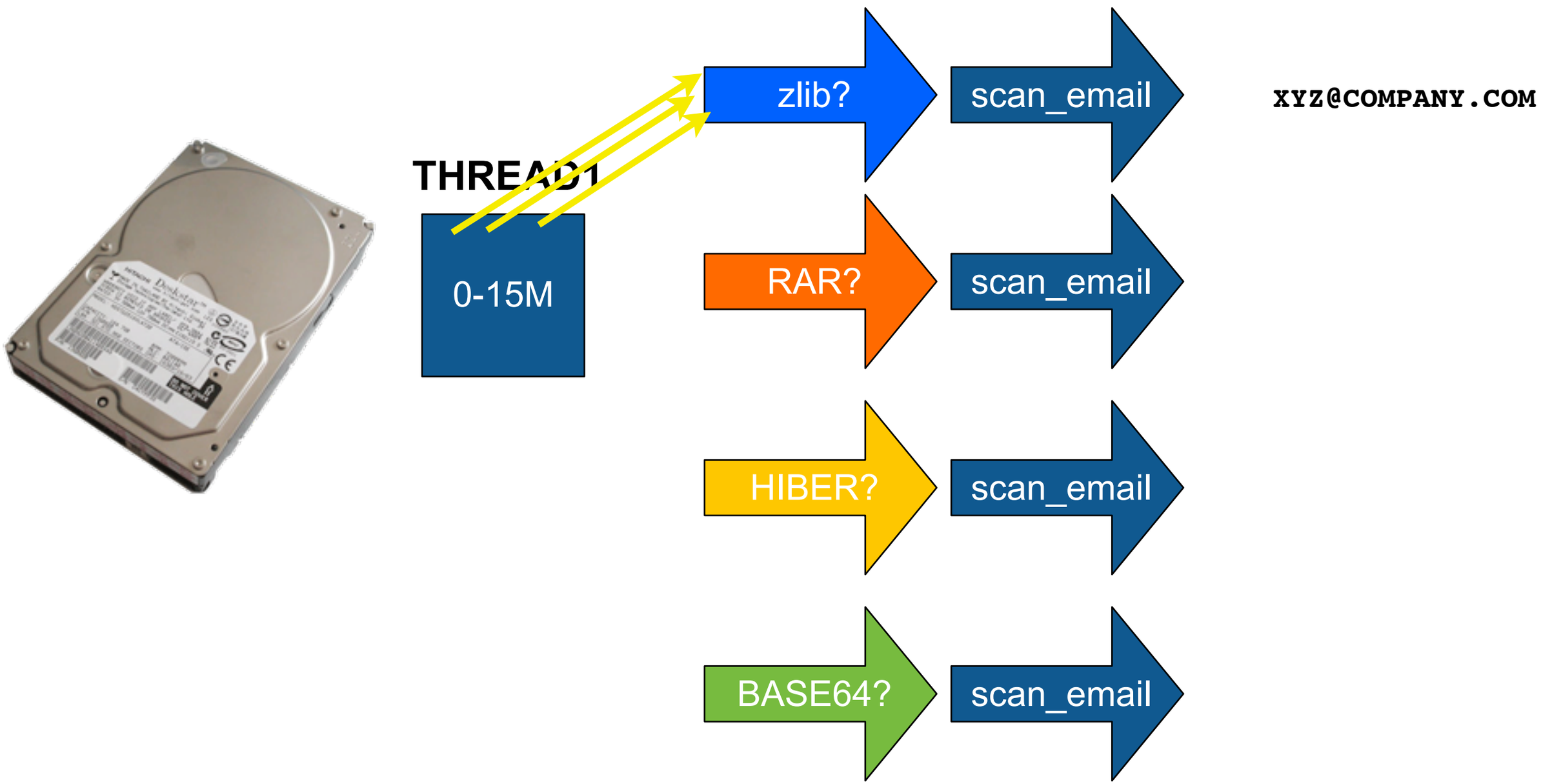


This finds obvious email addresses in bulk data:

```
a097 83a1 ed96 26a6 3c69 3d0f 750a 2399  ......&.<i=.u.#.
a2b5 bea7 692f 5847 a38a dd53 082c add5  ....i/XG...S.,..
5061 b64c 721d 864b 90b6 b55f bb04 735c  Pa.Lr..K..._..s\
9448 6730 5453 df64 813e b603 5795 2242  .Hg0TS.d.>..W."B
e9c8 7454 7322 7cdc b60e 97af 2f64 2728  ..tTs"|...../d'(
3cfb 84bd 2a84 2dfe 50ea 5935 c349 1513  <XYZ@COMPANY.COM
a9e9 e92c a3f8 6e46 0530 8a88 c7a2 5d2b  ...,..nF.0....]+
d89d 77cc fe1e f637 f3f3 d0af 1b47 c09b  ..w....7.....G..
```

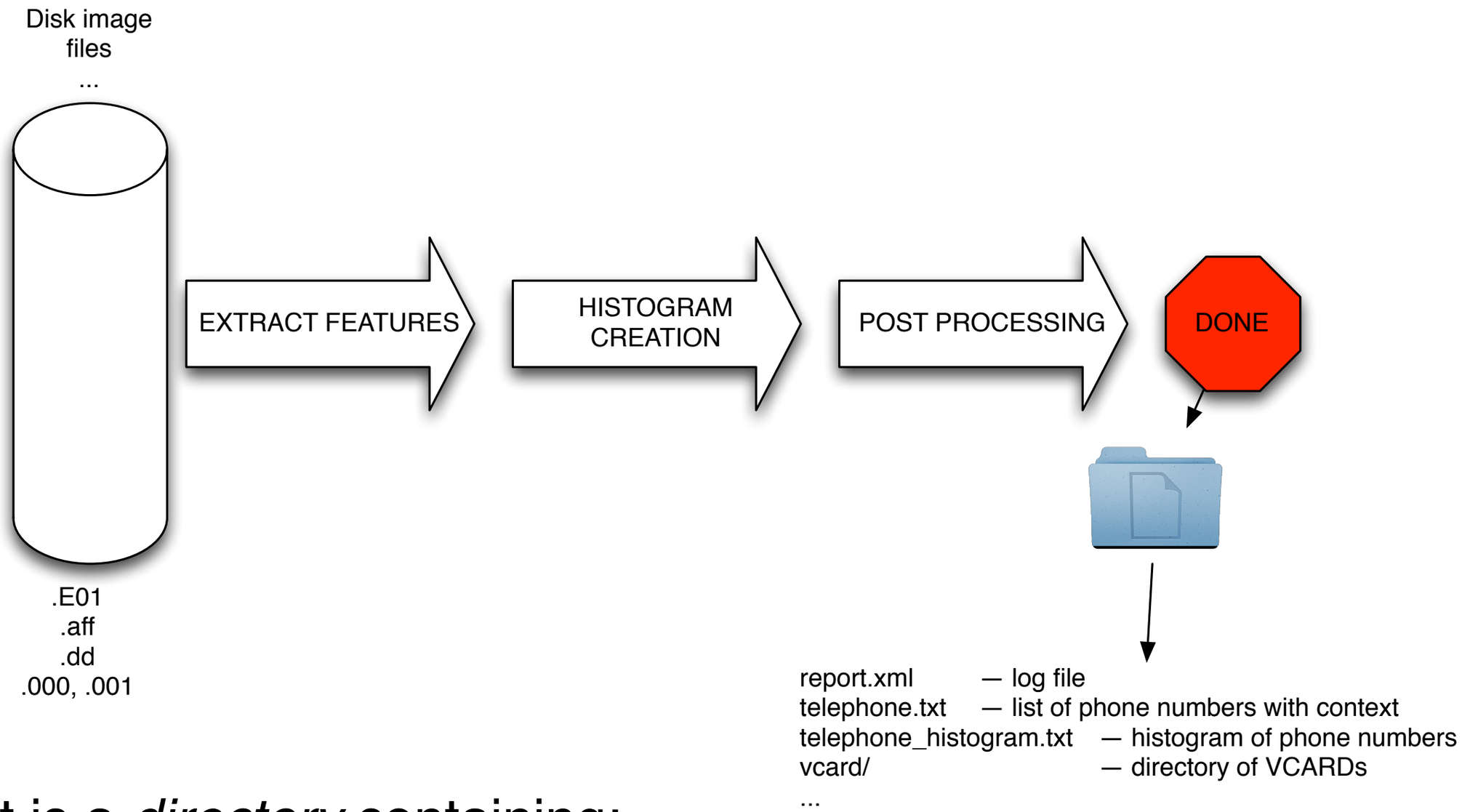
bulk_extractor examines every byte to see if it is the beginning of an “encoded” region.

Once the region is found, it’s decoded, then processed.



This “optimistic” approach also recovers data from fragments of files.

bulk_extractor has three phases of operation: Feature extraction; histogram creation; post processing



Output is a *directory* containing:

- feature files; histograms; carved objects
- Mostly in UTF-8; some XML
- Can be bundled into a ZIP file and processed with bulk_extractor_reader.py

bulk_extractor is run from the command line and creates a directory of “feature files” and carved results.

Command line:

```
$ bulk_extractor -o output_dir INPUT.E01
```

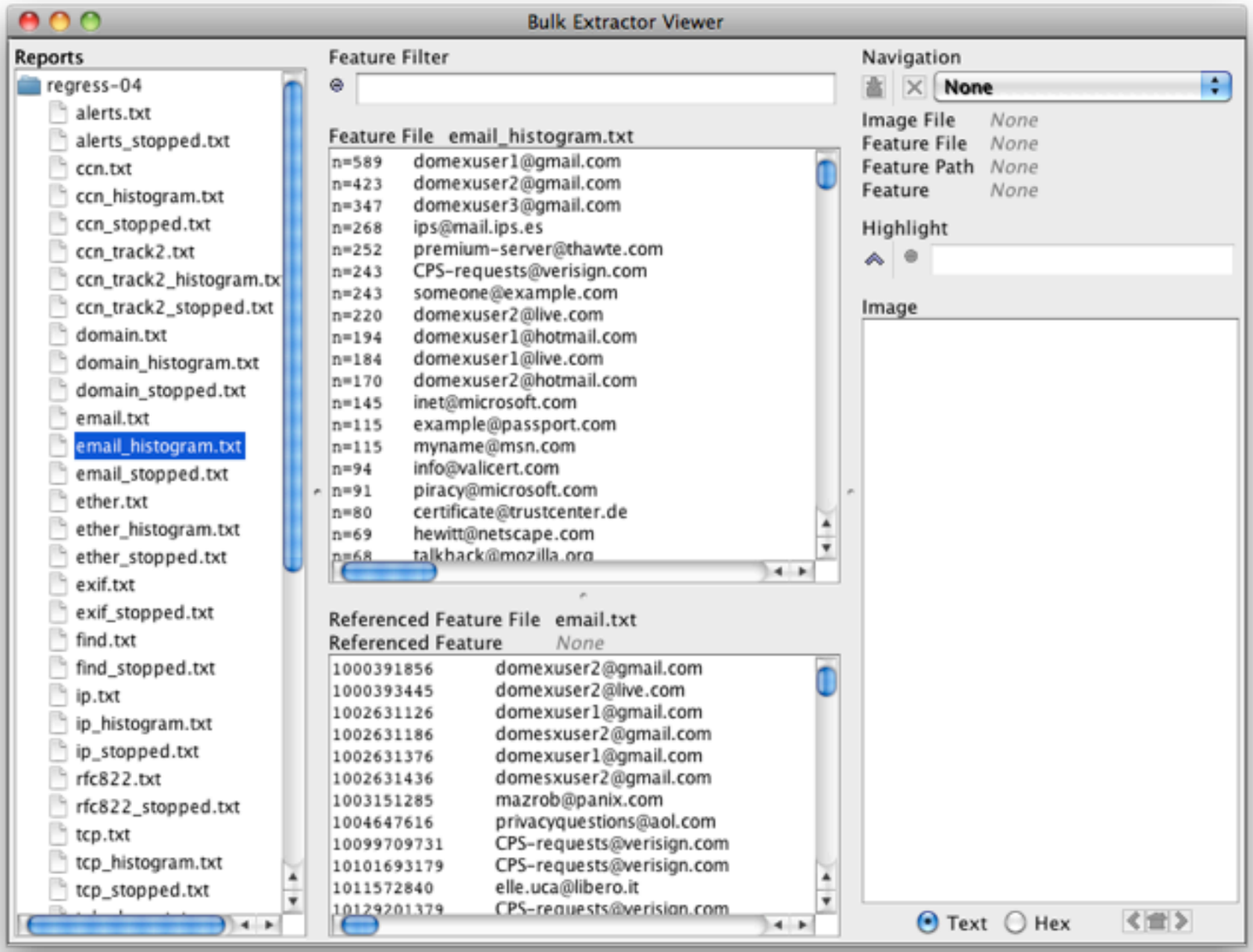
```
$ ls -l out-team
total 228
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 aes_keys.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 alerts.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 ccn.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 ccn_histogram.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 ccn_track2.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 ccn_track2_histogr
-rw-r-----+ 1 simsong  staff   18918 Apr 21 13:15 domain.txt
-rw-r-----+ 1 simsong  staff     854 Apr 21 13:15 domain_histogram.tx
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 elf.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 email.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 email_domain_histogram.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 email_histogram.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 ether.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 ether_histogram.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 exif.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 find.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 find_histogram.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 gps.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 ip.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 ip_histogram.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 jpeg_carved.txt
-rw-r-----+ 1 simsong  staff    6646 Apr 21 13:15 json.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 kml.txt
-rw-r-----+ 1 simsong  staff  147729 Apr 21 13:15 pii.txt
-rw-r-----+ 1 simsong  staff     313 Apr 21 13:15 pii_teamviewer_from.txt
-rw-r-----+ 1 simsong  staff       0 Apr 21 13:15 rar.txt
```

“0” means scanner ran, nothing found

teamviewer found

BEViewer: GUI runs on Windows, Mac & Linux
Launches bulk_extractor; views results

Uses bulk_extractor to decode encoded data.

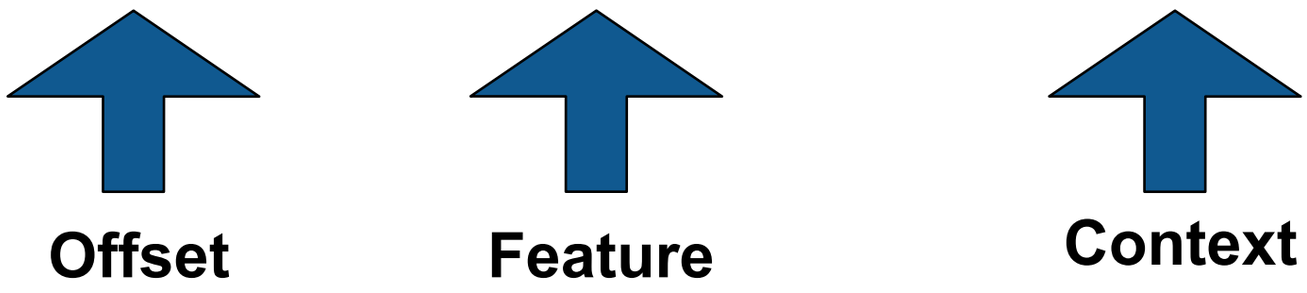


The viewer can run on an existing report.

“Feature files” contain the results of scanners.

Line-by-line output.

```
# Filename: /corp/nps/drives/nps-2009-m57-patents/charlie-2009-12-11.E01
# Feature-Recorder: telephone
# Feature-File-Version: 1.1
...
6489225486    (316) 788-7300   Corrine Porter (316) 788-7300,,,,,,Phase I En
6489230027    620-723-2638    ,,,,Dan Hayse - 620-723-2638,,,,,,Phase I En
6489230346    620-376-4499    Bertha Mangold -620-376-4499,,,,,,Phase I En
```



Designed for easy processing by grep, Notepad++, python or perl

- “Loosely ordered.”
- UTF-8 clean (non-UTF-8 characters are escaped with Python-notation)

The “context” helps you decide if the hit is valid or not.

These are not valid credit card numbers:

```
85364212076     346646148754362 /photo.php?fbid=346646148754362&set=a.345157228
85480391075     346344211011114 <t~x~\x80\x8A\x85vvx\x8A\x8Cm;/346344211011114/U}
dlrzy}\x83\x89\x8A\x8B\x87\x89\x88
87433290505     349446478403970 /photo.php?fbid=349446478403970&set=a.182950211
87434352827     347642328585048 /photo.php?fbid=347642328585048&set=t.100000231
87435156790     371055846243033 /photo.php?fbid=371055846243033&set=o.221006364
87435883832     347439688656505 /photo.php?fbid=347439688656505&set=a.292909134
```

These probably are:

```
18462439451     4519********3362        \x0A\x0D\x0ALynn:\x0D\x0ACard:
                                        4519********3362\x0D\x0APassword: rosi

18462439594     4519********3362        DINEROLYNN:CARD:
                                        4519********3362PASSWORD:ROSIETO
```

CCNs are validated with the Luhn algorithm & heuristics.

The “offset” also indicates how the data were decoded. We call this the “forensic path.”

```
# Filename: /corp/nps/drives/nps-2009-m57-patents/charlie-2009-12-11.E01
# Feature-Recorder: telephone
# Feature-File-Version: 1.1
...
6489225486    (316) 788-7300   Corrine Porter (316) 788-7300,,,,,,Phase I En
6489230027    620-723-2638    ,,,,Dan Hayse - 620-723-2638,,,,,,Phase I En
6489230346    620-376-4499    Bertha Mangold -620-376-4499,,,,,,Phase I En
...
3772517888-GZIP-28322  (831) 373-5555  onterey-<nobr>(831) 373-5555</nobr>
3772517888-GZIP-29518  (831) 899-8300  Seaside - <nobr>(831) 899-8300</nobr>
5054604751     716-871-2929     a%,888-571-2048,716-871-2929\x0D\x0ACPV,,,%Cape
```

Offset **Feature** **Context**

- “-GZIP-” indicates that data was decompressed

Decoding transformations can be stacked.

This output comes from a GZIP stream in a Windows Hibernation File.

```
...
…6464-HIBER-49691-GZIP-1526   groups-noreply@linkedin.com   3d\134"groups-noreply@linkedin.com
…6464-HIBER-49691-GZIP-2018   m*****************@gmail.com  3d\134"m*****************@gmail.co
…6464-HIBER-49691-GZIP-2128   sur*******1@gmail.com         3d\134"sur*******1@gmail.com\134"\
…6464-HIBER-49691-GZIP-2625   *******.consultancy@gmail.com 3d\134"*******.consultancy@gmail.c
…6464-HIBER-49691-GZIP-2736   sur*******1@gmail.com         3d\134"sur*******1@gmail.com\134"\
…6464-HIBER-49691-GZIP-3186   san****@***********.com       \134" "san****@***********.com\134"\134u
…6464-HIBER-49691-GZIP-3685   Careers@******bank.com        3d\134"Careers@******bank.com\134"
…6464-HIBER-49691-GZIP-4124   par****@team******.com        3d\134"par****@team******.com\134"
…6464-HIBER-49691-GZIP-4149   u003epar****@team******.com   \134u003epar****@team******.com\13
…6464-HIBER-49691-GZIP-4607   d****.*****@gmail.com         3d\134"d****.*****@gmail.com\134"\
…6464-HIBER-49691-GZIP-4631   u003ed****.*****@gmail.com    \134u003ed****.*****@gmail.com\134
…6464-HIBER-49691-GZIP-5114   raj******@bsnl.in             3d\134"raj******@bsnl.in\134"\134u
…6464-HIBER-49691-GZIP-5558   kiran.***@****technology.com  3d\134"kiran.***@****technology.co
…6464-HIBER-49691-GZIP-5671   sur*******1@gmail.com         3d\134"sur*******1@gmail.com\134"\
...
```

- JSON object downloaded from Facebook by compressed HTTP
- In RAM, written to HIBER on disk when the system went into sleep.

  —All same from same HIBER section & same GZIP object.

The file “report.xml” is an XML log of the processing.

```
-rw-r-----+ 1 simsong  staff   32766 Apr 21 13:15 report.xml
```

report.xml contains:

- Invoking command:

```
$ grep command out-M1234/report.xml
      <command_line>src/bulk_extractor -Z -o out-M1234 M1234.img</command_line>
$
```

- Compiler used to compile executable & linked libraries.
- System on which executable ran.
- Scanners that ran
- Statistics
- Performance counters

“tests/regress.py --analyze” will report time spent by each scanner

```
% python tests/regress.py --analyze ~/IN10-0512-fb
Analyze /home/simsong/IN10-0512-fb
bulk_extractor version: 1.5.0-alpha5
Image filename:         /corp/nus/drives/IN/IN10-0512/IN10-0512.E01
Scanner paths by time and calls
```

name	calls	sec	sec/call	% total
EMAIL	1813	4778.2158	2.6355	23.03%
ZIP	1813	3110.5149	1.7157	14.99%
NET	1813	2718.7876	1.4996	13.10%
ACCTS	1813	2134.9463	1.1776	10.29%
AES	1813	1656.8833	0.9139	7.99%
ZIP-NET	42060	832.9504	0.0198	4.01%
ZIP-EMAIL	42060	745.8298	0.0177	3.59%
HTTPLOGS	1813	483.8336	0.2669	2.33%
FIND	1813	439.8189	0.2426	2.12%
ZIP-ACCTS	42060	374.7072	0.0089	1.81%
ZIP-AES	42060	361.4514	0.0086	1.74%
BASE64	1813	338.8786	0.1869	1.63%
WINLNK	1813	309.9835	0.1710	1.49%
RAR	1813	298.1150	0.1644	1.44%
WINPE	1813	265.6627	0.1465	1.28%
HIBER	1813	265.1910	0.1463	1.28%
EXIF	1813	185.5110	0.1023	0.89%

Inside bulk_extractor

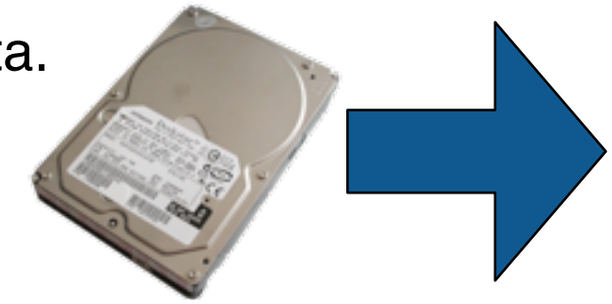
bulk_extractor: architectural overview

Written in C++, GNU flex and Java (GUI)

- Command-line tool.
- Linux, MacOS, Windows (compiled with mingw)
- BEViewer command-line tool and views results
- Overall size:
  - *23,911 lines C++*
  - *1,654 lines GNU Flex*
  - *17,861 lines of Java*

Key Features:

- "Scanners" look for information of interest in typical investigations.
- Recursively re-analyzes compressed data.
- Results stored in "feature files"
- Multi-threaded



http://www.nps.edu/

202-555-1212
user@domain.com

202-555-1212

http://www.nps.edu/

user@domain.com

bulk_extractor: system diagram

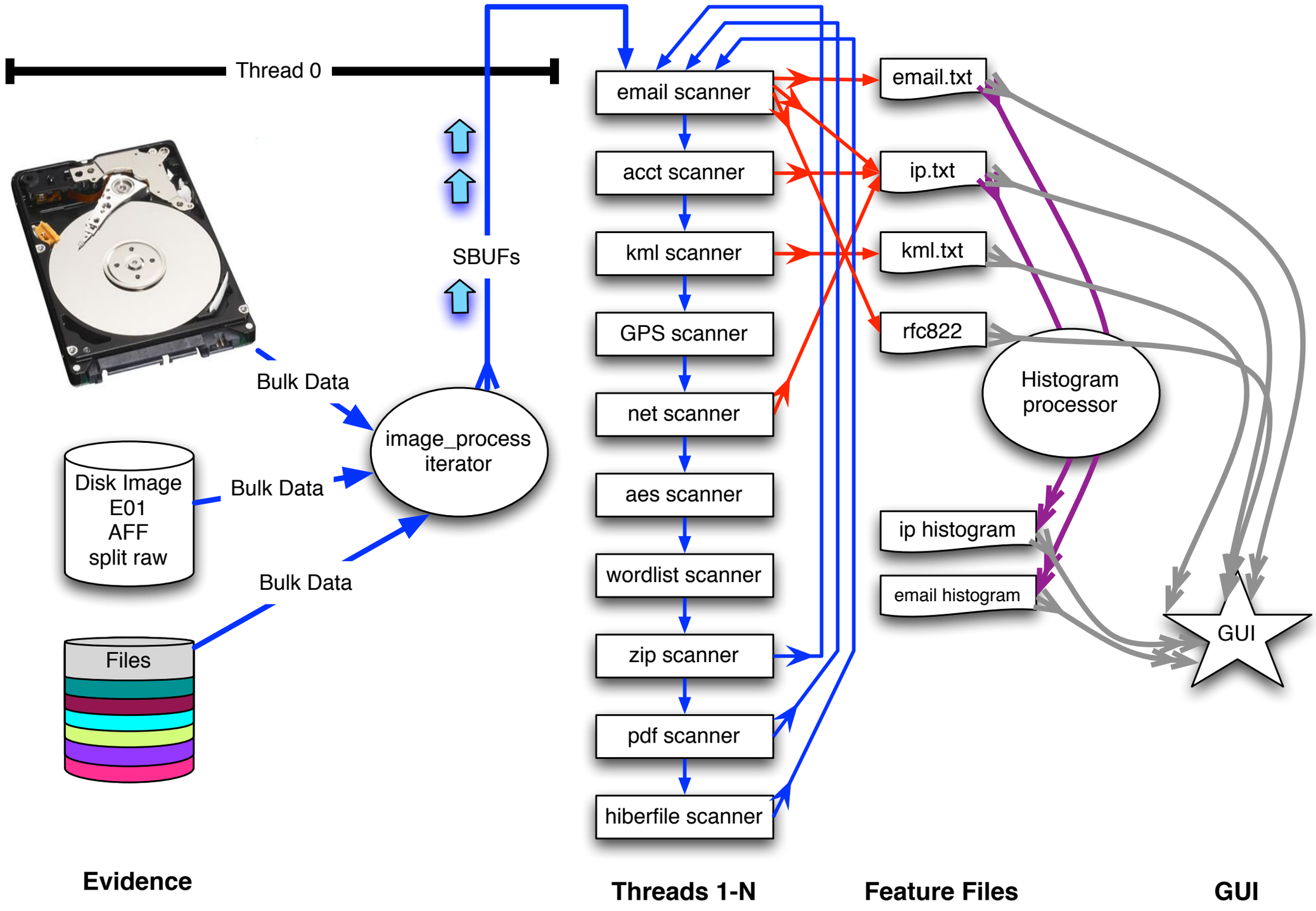
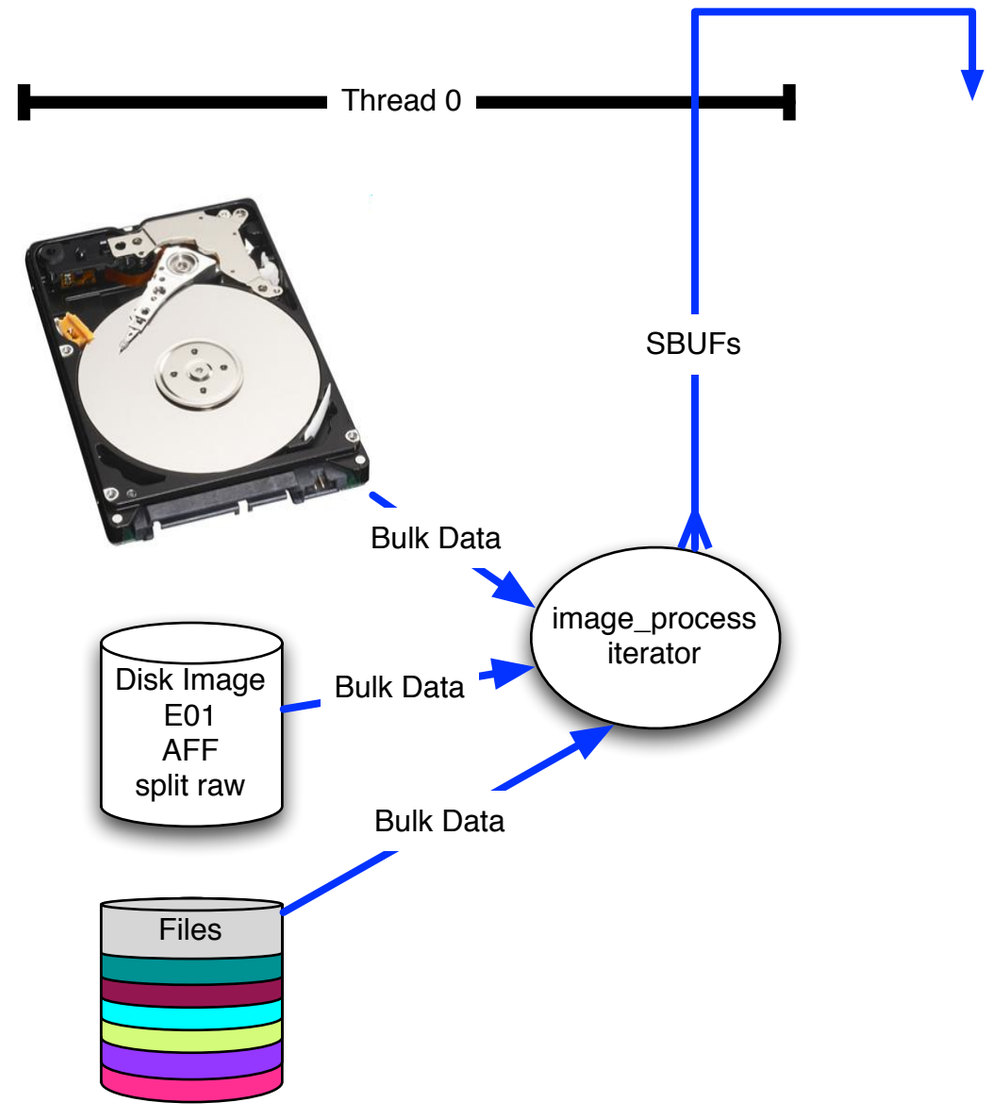


image processing
C++ iterator handles disks, images and files

Works with multiple disk formats.

- E01
- raw
- split raw
- individual disk files
- (AFF)



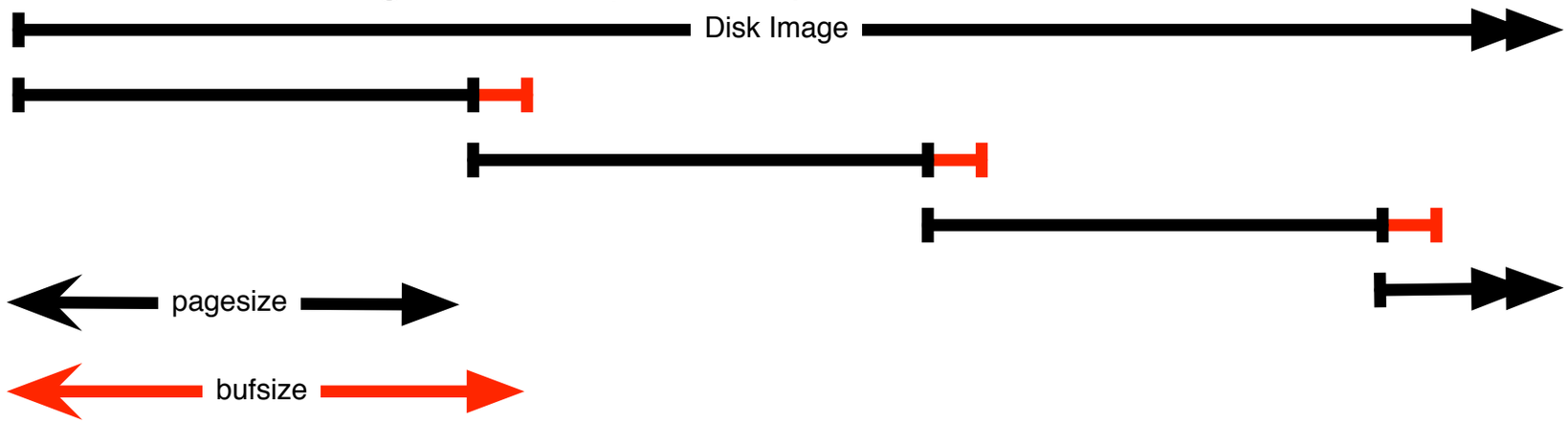
Evidence

We chop the 1TB disk into 65,536 x 16MiB “pages” for processing.

The “pages” overlap to avoid dropping features that cross buffer boundaries.

The overlap area is called the *margin*.

- Each sbuf can be processed in parallel — they don’t depend on each other.
- Features that start in the page but end in the margin are *reported*.
- Features that start in the margin are *ignored* (we get them later)
  - *Assumes that the feature size is smaller than the margin size.*
  - *BE version 1.5 margin: 4MB (tunable)*



Entire system is automatic:

- Image_process iterator makes **sbuf_t** buffers.
- Each buffer is processed by every scanner
- Features are automatically combined.

bulk_extractor has *many* scanners.
Each scanner runs sequentially on all the data.

Scanners can be turned on or off

- Use for tuning & debugging
  - *Turn off scanners you don't need.*
  - *Turn off scanners if you get a crash.*

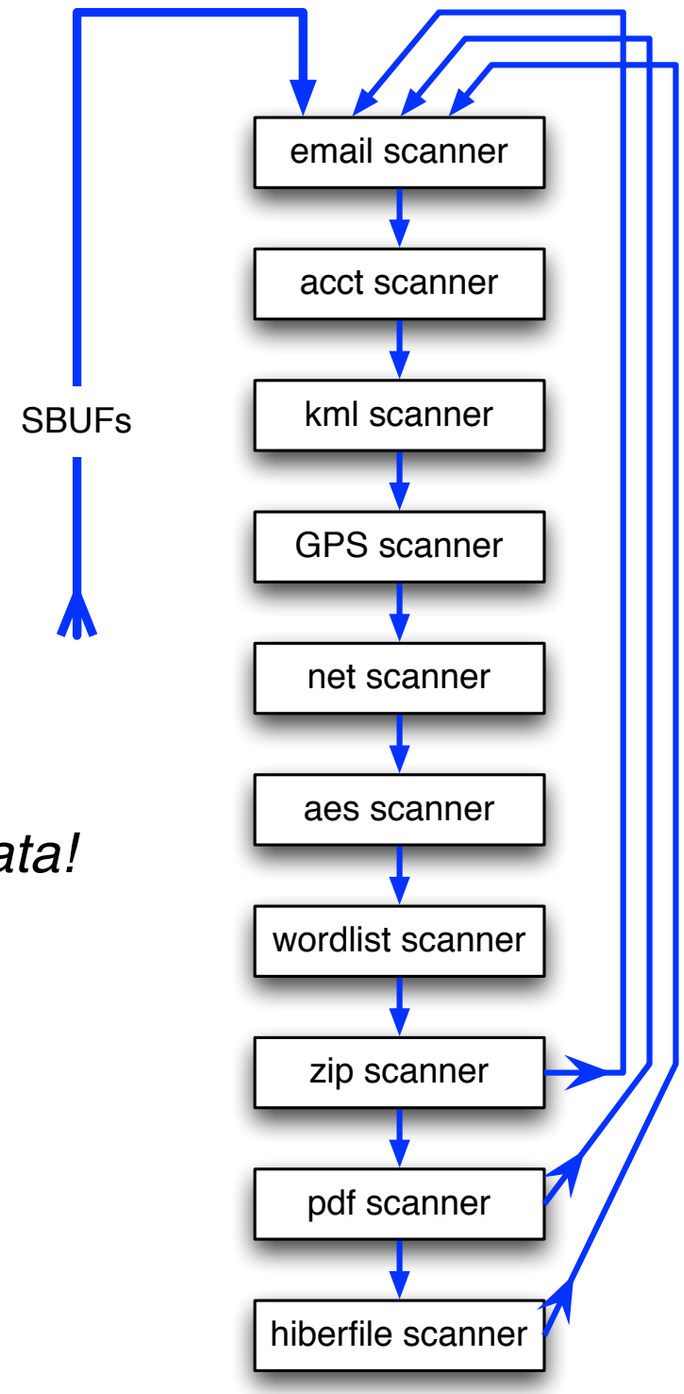
Some scanners are *recursive.*

- *e.g.* scan_zip will find zlib-compressed regions
- An **sbuf** is made for the decompressed data
- The data is re-analyzed by the other scanners
  - *This finds email addresses in compressed data!*

Some scanners can "carve"

Recursion used for:

- Decompressing ZLIB, Windows HIBERFILE,
- Extracting text from PDFs



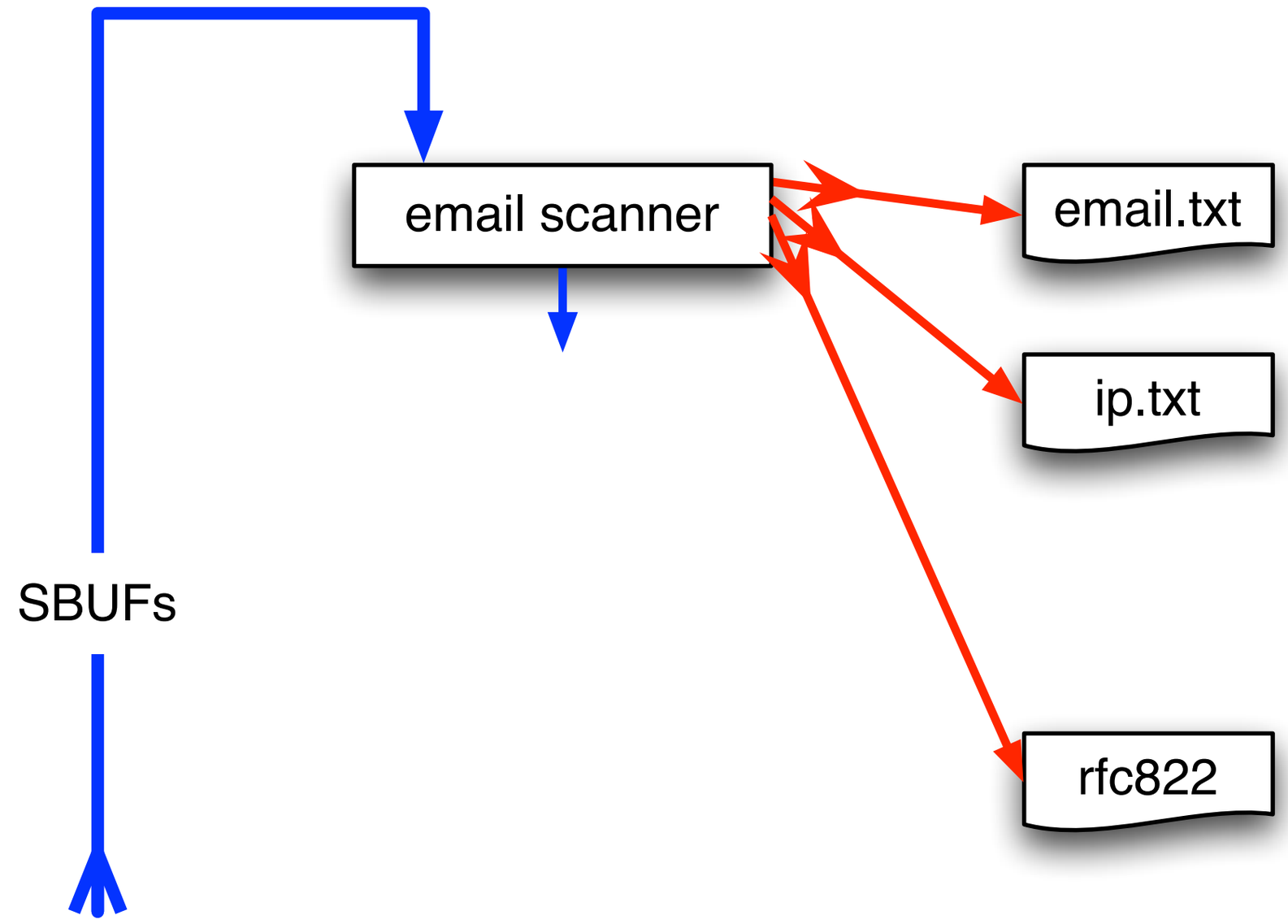
Scanners process each page and extract features

scan_email is the email scanner.

- inputs: **sbuf** objects

outputs:

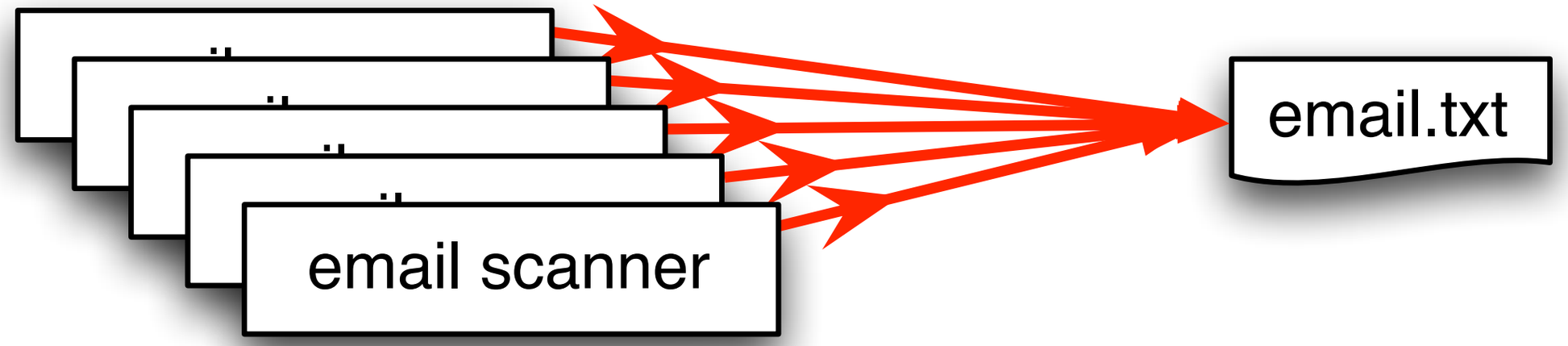
- **email.txt**
  - —*Email addresses*
- **rfc822.txt**
  - —*Message-ID*
  - —*Date:*
  - —*Subject:*
  - —*Cookie:*
  - —*Host:*
- **domain.txt**
  - —*IP addresses*
  - —*host names*



The feature recording system saves features to disk.

Feature Recorder objects store the features.

- Scanners are given a (feature_recorder *) pointer
- Feature recorders are *thread safe*.



Features are stored in a *feature file:*

```
48198832  domexuser2@gmail.com    tocol>____<name>domexuser2@gmail.com/Home</name>____
48200361  domexuser2@live.com     tocol>____<name>domexuser2@live.com</name>____<pass
48413829  siege@preoccupied.net   siege) O'Brien <siege@preoccupied.net>_hp://meanwhi
48481542  danilo@gnome.org        Danilo __egan <danilo@gnome.org>_Language-Team:
48481589  gnom@prevod.org         : Serbian (sr) <gnom@prevod.org>_MIME-Version:
49421069  domexuser1@gmail.com    server2.name", "domexuser1@gmail.com");__user_pref("
49421279  domexuser1@gmail.com    er2.userName", "domexuser1@gmail.com");__user_pref("
49421608  domexuser1@gmail.com    tp1.username", "domexuser1@gmail.com");__user_pref("
```

offset **feature** **feature in evidence context**

Features can also be stored in a SQLite3 database*

Each “feature table” that stores:

- offset — bytes from start of disk (integer)
- path — forensic path (string)
- feature_eutf8 — feature, escaped UTF8 (string)
- feature_utf8 — feature, pure UTF8 (string)
- context_eutf8 — escaped UTF8 (string)

```
sqlite> select * from f_email limit 10;
```

offset	path	feature_eutf8	feature_utf8
42716701	42716701-BASE64-60	pki@microsoft.com	pki@micro...
42716701	42716701-BASE64-396	pki@microsoft.com	pki@micro...
24900678	24900678	grafta@bl.com	grafta@bl.com \x028\x0...
26735686	26735686	grafta@bl.com	grafta@bl.com \x028\x0...
32597062	32597062	grafta@bl.com	grafta@bl.com \x028\x0...
50392739	50392739	inet@microsoft.com	inet@microsoft.co...
51781228	51781228	dbaron@dbaron.org	dbaron@dbaron.org...
51788157	51788157	bzbarsky@mit.edu	bzbarsky@mit.edu ...
51789901	51789901	bzbarsky@mit.edu	bzbarsky@mit.edu ...
51791829	51791829	roland.mainz@informatik.med.uni-giessen.d...	

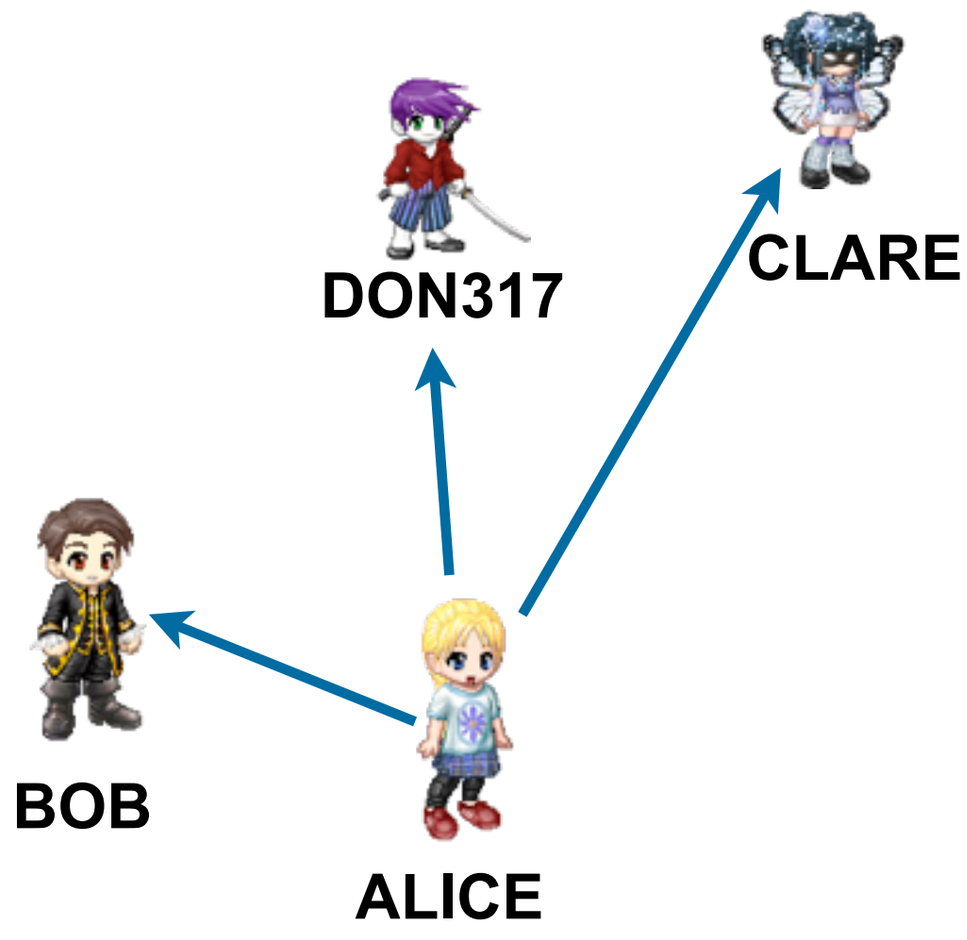
Enable with command line option:

```
-S write_feature_sqlite3=YES
```

Feature histograms are created at the end of processing. They are a powerful tool for understanding relations.

Email address histogram allows us to rapidly determine:

- Drive's primary user
- User's organization
- Primary correspondents
- Other email addresses



Drive #51 (Anonymized)

ALICE@DOMAIN1.com	8133
BOB@DOMAIN1.com	3504
ALICE@mail.adhost.com	2956
JobInfo@alumni-gsb.stanford.edu	2108
CLARE@aol.com	1579
DON317@earthlink.net	1206
ERIC@DOMAIN1.com	1118
GABBY10@aol.com	1030
HAROLD@HAROLD.com	989
ISHMAEL@JACK.wolfe.net	960
KIM@prodigy.net	947
ISHMAEL-list@rcia.com	845
JACK@nwlink.com	802
LEN@wolfenet.com	790
natcom-list@rcia.com	763

Histograms can be based on regular expressions to extract strings from feature files.

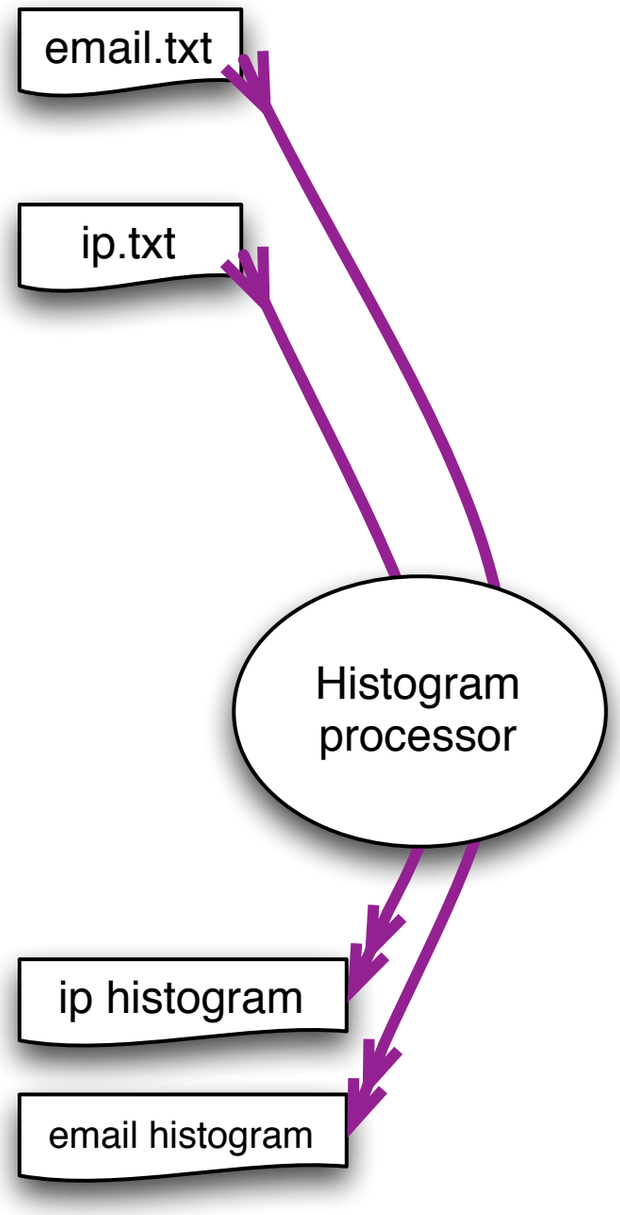
Simple histogram based on “pure” feature:

```
n=579    domexuser1@gmail.com
n=432    domexuser2@gmail.com
n=340    domexuser3@gmail.com
n=268    ips@mail.ips.es
n=252    premium-server@thawte.com
n=244    CPS-requests@verisign.com
n=242    someone@example.com
```

Based on regular expression extraction:

- For example, extract search terms with **.*search.*q=(.*)**

```
n=18     pidgin
n=10     hotmail+thunderbird
n=3      Grey+Gardens+cousins
n=3      dvd
n=2      %TERMS%
n=2      cache:
n=2      p
n=2      pi
n=2      pid
n=1      Abolish+income+tax
n=1      Brad+and+Angelina+nanny+help
n=1      Build+Windmill
n=1      Carol+Alt
```



Use the histograms to get an overview of the media.

Histograms created by default in bulk_extractor 1.5:

- ccn_histogram — credit card numbers
- ccn_track2_histogram — track2 information
- domain_histogram — All domains found on the drive
- email_domain_histogram — Domains from email addresses
- email_histogram — Email addresses
- ip_histogram — IP addresses (frequently false positives)
- pii_teamviewer* — TeamViewer IDs
- telephone_histogram — Telephone numbers
- url_facebook — Facebook URLs
- url_histogram — JURLs
- url_microsoft_live — Microsoft Live URLs
- url_searches — What people searched for
- url_services — Domains from URLs

url_searches.txt is a list of search URLs from media.

```
$ more out-domexusers-baseline/url_searches.txt
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 1.5.0-beta1 ($Rev: 10844 $)
# Feature-Recorder: url
# Filename: /corp/nps/drives/nps-2009-domexusers/nps-2009-domexusers.E01
# Histogram-File-Version: 1.1
n=18        pidgin
n=10        hotmail+thunderbird
n=7         %s
n=3         Grey+Gardens+cousins
n=3         dvd
n=2         %TERMS%
n=2         p
n=2         pi
n=2         pid
n=1         Abolish+income+tax
n=1         Brad+and+Angelina+nanny+help
n=1         Build+Windmill
n=1         Carol+Alt
n=1         DVD
n=1         Don+Quixote
n=1         Ivanka+Trump
n=1         John+Updike
n=1         Kate+Hudson
n=1         Obama+McCain+Palin+Biden
n=1         Patrick+Swayze
n=1         StarCaps
```

BE1.5 can also store histograms in SQLite3*

```
sqlite> select * from h_email order by count desc limit 10;
count   feature_utf8
589     domexuser1@gmail.com
423     domexuser2@gmail.com
347     domexuser3@gmail.com
268     ips@mail.ips.es
252     premium-server@thawte.com
243     someone@example.com
243     CPS-requests@verisign.com
220     domexuser2@live.com
194     domexuser1@hotmail.com
184     domexuser1@live.com
sqlite>
```

Enable with command line option:

```
-S write_feature_sqlite3=YES
```

```
$ bulk_extractor -o output mydisk.raw
```

Running bulk_extractor

bulk_extractor is a command line tool.

```
$ ./bulk_extractor -o out-1 disk.img
bulk_extractor version: 1.5.0-alpha9
Hostname: mncrnpsedu.local
Input file: disk.img
Output directory: out-1
Disk Size: 5164080
Threads: 24
Attempt to open bulk_extractor
All data are read; waiting for threads to finish...
Time elapsed waiting for 1 thread to finish:
     (timeout in 60 min.)
All Threads Finished!
Producer time spent waiting: 0 sec.
Average consumer time spent waiting: 1.96071 sec.
MD5 of Disk Image: d85cf891e6297e71c8d7ae4368bf5eb6
Phase 2. Shutting down scanners
Phase 3. Creating Histograms
Elapsed time: 2.05625 sec.
Total MB processed: 5
Overall performance: 2.51141 MBytes/sec (0.104642 MBytes/sec/thread)
Total email features found: 0
$
```

bulk_extractor help is always available

Help is always available:

```
$ src/bulk_extractor -h
bulk_extractor version 1.3b6 $Rev: 10046 $
Usage: src/bulk_extractor [options] imagefile
  runs bulk extractor and outputs to stdout a summary of what was found where

Required parameters:
   imagefile     - the file to extract
 or  -R filedir  - recurse through a directory of files
                   SUPPORT FOR E01 FILES COMPILED IN
                   SUPPORT FOR AFF FILES COMPILED IN
                   EXIV2 COMPILED IN
-o outdir      - specifies output directory. Must not exist.
                 bulk_extractor creates this directory.
```

-h updates automatically depending on how bulk_extractor is compiled.

- Disk image formats supported (E01, AFF)
- Compiled-in scanners that are enabled
- Plug-ins that are loaded at startup.

bulk_extractor input and output functions

Options change the behavior of the scanner:

```
Options:
   -b banner.txt - Add banner.txt contents to the top of every output file.
   -r alert_list.txt  - a file containing the alert list of features to alert
                        (can be a feature file or a list of globs)
                        (can be repeated.)
   -w stop_list.txt   - a file containing the stop list of features (white list
                        (can be a feature file or a list of globs)s
                        (can be repeated.)
   -F <rfile>   - Read a list of regular expressions from <rfile> to find
   -f <regex>   - find occurrences of <regex>; may be repeated.
                  results go into find.txt
```

-b — Adds a “classification banner” to every output file.

-r, -w — alert list (red list) and stop list (white list) features.

-F, -f — Search for keywords in all data

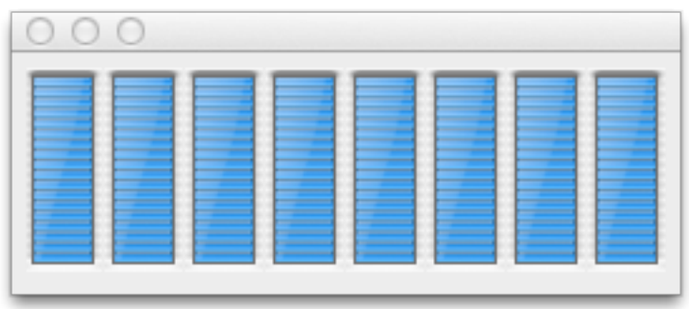
bulk_extractor threading control

Normally bulk_extractor will run one analysis thread per core.

You can make it use less cores if you need to leave some free:

```
-j NN          - Number of analysis threads to run (default 8)
```

```
Input file: /corp/nps/drives/nps-2009-ubnist1/ubnist1.gen3.E01
Output directory: regress-1.3b7-norm-01
Disk Size: 2106589184
Threads: 8
Phase 1.
 9:33:28 Offset 0MB (0.00%) Done in n/a at 09:33:27
 9:33:29 Offset 16MB (0.80%) Done in  0:00:58 at 09:34:27
 9:33:29 Offset 33MB (1.59%) Done in  0:00:48 at 09:34:17
 9:33:41 Offset 50MB (2.39%) Done in  0:08:27 at 09:42:08
 9:33:41 Offset 67MB (3.19%) Done in  0:06:42 at 09:40:23
```



There are many configurable options.

The -s option allows scanners to have settable tuning parameters:

```
-S work_start_work_end=YES    Record work start and end of each scanner in
report.xml file ()

-S enable_histograms=YES    Disable generation of histograms ()

-S hash_alg=md5    Specifies hash algorithm to be used for all hash calculations

-S dup_data_alerts=NO    Notify when duplicate data is not processed ()

-S write_feature_files=YES    Write features to flat files ()

-S write_feature_sqlite3=NO    Write feature files to report.sqlite3 ()

-S report_read_errors=YES    Report read errors ()

-S ssn_mode=0    0=Normal; 1=No `SSN' required; 2=No dashes required (accts)

-S min_phone_digits=6    Min. digits required in a phone (accts)

-S carve_net_memory=NO    Carve network  memory structures (net)
```

Use -h for a list of all options.

Individual scanners can be enabled or disabled.

These scanners disabled by default; enable with -e:

```
-e base16 - enable scanner base16
-e facebook - enable scanner facebook
-e outlook - enable scanner outlook
-e sceadan - enable scanner sceadan
-e wordlist - enable scanner wordlist
-e xor - enable scanner xor
```

These scanners enabled by default; disable with -x:

```
-x accts - disable scanner accts
-x aes - disable scanner aes
-x base64 - disable scanner base64
-x elf - disable scanner elf
-x email - disable scanner email
-x exif - disable scanner exif
-x find - disable scanner find
-x gps - disable scanner gps
-x gzip - disable scanner gzip
-x hiber - disable scanner hiber
-x httplogs - disable scanner httplogs
-x json - disable scanner json
-x kml - disable scanner kml
-x net - disable scanner net
-x pdf - disable scanner pdf
-x rar - disable scanner rar
-x sqlite - disable scanner sqlite
-x vcard - disable scanner vcard
-x windirs - disable scanner windirs
-x winlnk - disable scanner winlnk
-x winpe - disable scanner winpe
-x winprefetch - disable scanner winprefetch
-x zip - disable scanner zip
```

Don't assume that you should enable every scanner!

Discussed in next sections

```
-rw-r--r--@ 1 simsong  staff       476 Jul  7 23:50 aes_keys.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 alerts.txt
-rw-r--r--@ 1 simsong  staff      2743 Jul  7 23:59 ccn.txt
-rw-r--r--@ 1 simsong  staff       454 Jul  8 00:03 ccn_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 ccn_track2.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 ccn_track2_histogram.txt
-rw-r--r--@ 1 simsong  staff  23369167 Jul  8 00:03 domain.txt
-rw-r--r--@ 1 simsong  staff    185266 Jul  8 00:03 domain_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 elf.txt
-rw-r--r--@ 1 simsong  staff   1719842 Jul  8 00:03 email.txt
-rw-r--r--@ 1 simsong  staff     35073 Jul  8 00:03 email_histogram.txt
-rw-r--r--@ 1 simsong  staff     23961 Jul  8 00:00 ether.txt
-rw-r--r--@ 1 simsong  staff       337 Jul  8 00:03 ether_histogram.txt
-rw-r--r--@ 1 simsong  staff  11188830 Jul  8 00:03 exif.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 find.txt
-rw-r--r--@ 1 simsong  staff      1112 Jul  8 00:01 gps.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 hex.txt
-rw-r--r--@ 1 simsong  staff     95835 Jul  8 00:03 ip.txt
-rw-r--r--@ 1 simsong  staff     11603 Jul  8 00:03 ip_histogram.txt
-rw-r--r--@ 1 simsong  staff   2025702 Jul  8 00:03 json.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 kml.txt
-rw-r--r--@ 1 simsong  staff    194991 Jul  8 00:03 packets.pcap
-rw-r--r--@ 1 simsong  staff     21343 Jul  8 00:03 report.xml
-rw-r--r--@ 1 simsong  staff   3782598 Jul  8 00:03 rfc822.txt
-rw-r--r--@ 1 simsong  staff    213746 Jul  8 00:03 tcp.txt
-rw-r--r--@ 1 simsong  staff     61255 Jul  8 00:03 tcp_histogram.txt
-rw-r--r--@ 1 simsong  staff     59469 Jul  8 00:03 telephone.txt
-rw-r--r--@ 1 simsong  staff      6612 Jul  8 00:03 telephone_histogram.txt
-rw-r--r--@ 1 simsong  staff  67205326 Jul  8 00:03 url.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_facebook-id.txt
-rw-r--r--@ 1 simsong  staff   5706665 Jul  8 00:03 url_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_microsoft-live.txt
-rw-r--r--@ 1 simsong  staff      8504 Jul  8 00:03 url_searches.txt
-rw-r--r--@ 1 simsong  staff    151673 Jul  8 00:03 url_services.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 vcard.txt
-rw-r--r--@ 1 simsong  staff  18549729 Jul  8 00:03 windirs.txt
-rw-r--r--@ 1 simsong  staff  29051041 Jul  8 00:03 winpe.txt
-rw-r--r--@ 1 simsong  staff   1984759 Jul  8 00:03 winprefetch.txt
-rw-r--r--@ 1 simsong  staff  34128889 Jul  8 00:03 zip.txt
```

bulk_extractor 1.5 scanners and output

Scanners in bulk_extractor 1.5

Optimistic Decoding scanners

- **base64*** — Decodes Base64 encoded data (email attachments & SSL certificates)
- **gzip** — Browser cache
- **hiberfile** — Decompresses fragments of Windows hibernation files.
- **pdf** — Extracts text from PDF files
- **rar** — Decompressed unencrypted RAR files; carves encrypted RAR files
- **zip** — decompresses ZIP'ed files, optionally carves ZIP components.
- **base16** (disabled by default*) — finds HEXADECIMAL strings and decodes them]
- **outlook*** (disabled by default*) — Outlook Compressible Encryption; obsolete
- **xor** (disabled by default) — applies any XOR mask to all input data; good for malware.)

Identity scanners:

- **accts** — Credit Card numbers, Track 2 data, Telephone Numbers, and "PII"
  - —*DOB; SSNs; TeamViewer*; etc.*
- **email** — email addresses, IP addresses, Email message headers, MAC addresses, URLs
- **gps —** Looks for Garmin-formatted trackpoint XML

More scanners in bulk_extractor 1.5

Information on Linux and Windows operating system structures

- **elf —** Linux Executables
- **windirs —** Windows directory entries (NTFS and FAT32)
- **winpe —** Windows executables
- **winprefetch —** Windows prefetch files
- **winlnk* —** Windows LNK files

Web-related scanners:

- **json** — Properly-formatted JSON
- **kml** — KML files
- **facebook*** — Facebook HTML
- **httplogs* —** Looks for fragments of HTTP log files
- **vcard** — VCARD entries

More scanners in bulk_extractor 1.5

Misc:

- **aes** — Searches for scheduled AES keys in RAM dumps
- **wordlist** — Simple wordlist generator (la-strings is better)
- **net** — IP packets & TCP/IP memory structures
- **find** — Simple word search
- **sqlite*** — identifies SQLite3 databases and optionally carves them.

Plug-ins for other NPS projects:

- **hashdb*** — Builds or Searches an NPS “sector hash database.”
- **sceadan*** — Runs UTSA file type identification system

Why use bulk_extractor to find Windows executables?

BE works with incomplete executables:

- Only needs the first 4K
- Produces MD5 hash of first 4K (distinct for most executables)
- Decodes all available PE header fields

BE finds executables other approaches miss:

- Orphans not in the file system
- Fragments in RAM
- BASE64 encoded PE files (email attachments)
- GZIP-encoded PE files (HTTP downloads)
- RAR-encoded PE files

Why use bulk_extractor to find Windows LNK files?

Windows LNK files provide information about:

- Most Recently Used files
- Shortcuts
- Other system information.

LNK information is provided as DFXML data in the feature file:

```
263671296        E:\x5Cz-netvampire33_2\x5C10.jpg
<lnk><atime>2003-08-09T16:00:00Z</atime><ctime>2000-10-25T14:49:40Z</
ctime><path>E:\x5Cz-netvampire33_2\x5C10.jpg</path><wtime>1999-06-02T23:46:08Z</
wtime></lnk>
```

- Reformats as:

```
263671296        E:\x5Cz-netvampire33_2\x5C10.jpg
      <lnk> <atime>2003-08-09T16:00:00Z</atime>
            <ctime>2000-10-25T14:49:40Z</ctime>
            <path>E:\x5Cz-netvampire33_2\x5C10.jpg</path>
            <wtime>1999-06-02T23:46:08Z</wtime> </lnk>
```

BE will find LNK files that were deleted and “lost”

Why use bulk_extractor to find Prefetch files?

Windows prefetch files contain:

- Name of executable
- Time last run
- Total Runs
- Linked DLLs

Windows deletes prefetch files when there are more than 129.

bulk_extractor finds all of the prefetch files on the drive and decodes the contents as XML:

```
13688320        SHMGRATE.EXE    <prefetch><os>Windows XP</
os><filename>SHMGRATE.EXE<filename><header_size>152<header_size><atime>2005-08-08
T14:42:40Z</atime><runs>1</runs><filenames>...
```

Drive 0844.aff has 400+ prefetch files on the drive. Use them to:

- Infer when a program was last run.
- Gauge a program's popularity

Why use bulk_extractor to find RAR files?

rar.txt — A large file says that there were lots of RAR files

- rar.txt will give you the file names, flags, file size, and other data:

```
44871848980     MsgPlusLive Agreement (es).rtf  <rar_component><name>MsgPlusLive
Agreement (es).rtf</name><flags>0x0000</
flags><version>29<version><compression_method>smallest</
compression_method><uncompr_size>9396</uncompr_size><compr_size>2942</
compr_size><file_attr>0x20</file_attr><lastmoddate>2006-06-10T09:36:46Z</
lastmoddate><host_os>Windows</host_os><crc32>0x626F65C1</crc32></rar_component>..
```

You can scan for encrypted files.

```
<encrypted>true</encrypted>
```

You can carve objects from within RAR files

- JPEGs, PDFs, etc.

Why use bulk_extractor for web server logs?

Because they are sometimes present! Unknowingly! e.g. IN10-0512:

```
4406559744  GET  /n/p?module=8814&error=0&language=09.01&product=Norton
%20AntiVirus&version=16.0.0.125&hbguid=839fa0aa-d7bf-11df-
bb3c-001cc0560f34&c=1425011&psn=CHT8VVMM9GBY&m=14483463&b=200&a=0&h=351 HTTP/
1.1\x0DGET /n/p?module=8814&error=0&language=09.01&product=Norton
%20AntiVirus&version=16.0.0.125&hbguid=839fa0aa-d7bf-11df-
bb3c-001cc0560f34&c=1425011&psn=CHT8VVMM9GBY&m=14483463&b=200&a=0&h=351 HTTP/
1.1\x0D
```

Found on the NPS 2TB disk:

```
159319740968 122.162.197.206 - - [14/May/2007:05:03:21 -0500] ``GET /lc/meetings/
delhi03/talks/AMitra.pdf HTTP/1.1'' 206 28425122.162.197.206 - - [14/May/
2007:05:03:21 -0500] ``GET /lc/meetings/delhi03/talks/AMitra.pdf HTTP/1.1'' 206
28425

159319741080 122.162.197.206 - - [14/May/2007:05:03:23 -0500] ``GET /lc/meetings/
delhi03/talks/AMitra.pdf HTTP/1.1'' 206 31400122.162.197.206 - - [14/May/
2007:05:03:23 -0500] ``GET /lc/meetings/delhi03/talks/AMitra.pdf HTTP/1.1'' 206
31400
```

-

bulk_extractor 1.5 will carve encoded files. These are missed by other scanners.

Some scanners can “carve.” Each carving scanner has a carving mode.

- Carving mode 0 — carve nothing
- Carving mode 1 — Only carve if file was “encoded.”
  - — *e.g. a JPEG won’t be carved, but a BASE64-encoded JPEG will be carved.*
  - — *These are the files missed by conventional carvers.*
- Carving mode 2 — Carve everything.

Carving “dedups”

Carved files are binned in the output directory:

```
$ ls -l out-TH0001_0028/jpeg_carved/000/
   2905 Jul  2 06:16 10410502381-ZIP-29.jpg
  15452 Jul  2 06:16 10602684084-BASE64-0.jpg
   1395 Jul  2 06:18 11547112759-ZIP-2184645-ZIP-1320-ZIP-0.jpg
```

Filename is forensic path

Carving scanners:

- exif — JPEGs
- kml — KML files
- net — IP packets
- RAR — RAR-compressed files

sqlite - Sqlite3 databases (only contiguous)

vcard — Contacts (VCARDs)

Bugfix #1:
Fix to scan_base64 scanner

Bulk_extractor 1.4 scan_base64 missed many BASE64 regions.

- Fixed in BE1.5!

Disk Image	BE14 base64 emails	BE15 base64 emails
domexusers	**0**	**11**
NPS 2TB Drive	**1**	**4,760**

Most BASE64-encoded email addresses are from:

- SSL certificates
- email attachments

Bugfix #2: Improved stoplist performance

bulk_extractor supports two kinds of “stop lists” (black lists)

- Word-based:

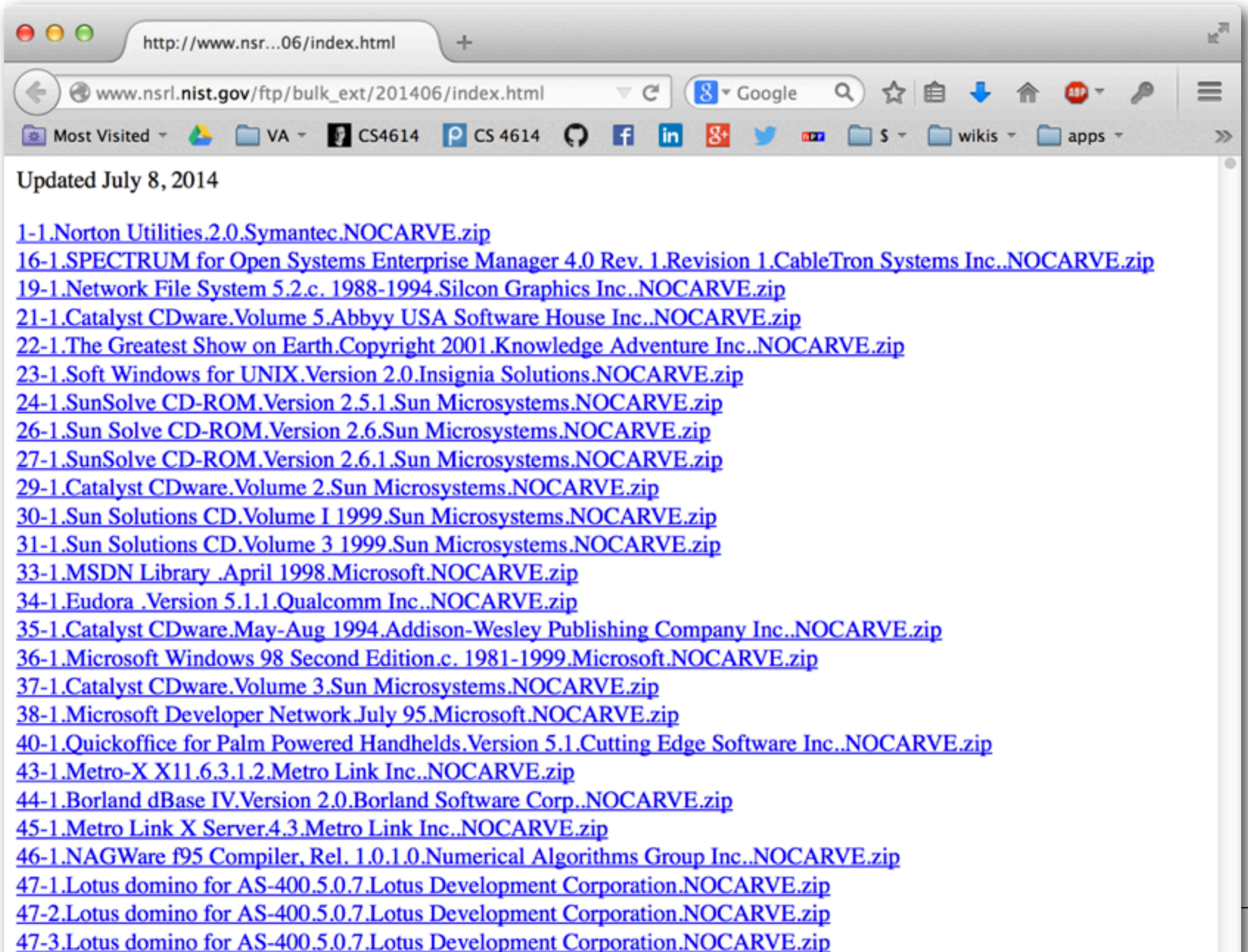
```
0%4@2gy2kj.es
0%f@m4.bh
0%p@vcp.sr
0%s@earthlink.net
0+anwwhbp8fa@hpqe.qa
```

- Context-based:

```
11996200    CPS-requests@verisign.com    m; by E-mail at CPS-
requests@verisign.com; or\x0Aby mail at
124792693   Thierry.Mayeur@lotus.com     ing addresses:\x0D
\x0DThierry.Mayeur@lotus.com\x0DTMayeur@ca.ibm.
124792718   TMayeur@ca.ibm.com ayeur@lotus.com\x0DTMayeur@ca.ibm.com\x0D\x0D2.
What's new\x0D
212163670   CPS-requests@verisign.com    m; by E-mail at CPS-
requests@verisign.com; or\x0Aby mail at
352717318   firstname_lastname@company.com        main, such as:
'firstname_lastname@company.com(foobar)@Notes_d
```

Context-based stop-lists are feature files concatenated together.

BE1.5 stoplist:
NIST ran bulk_extractor over the entire NSRL



Each NSRL output is a ZIP file — a bulk_extractor report

```
    $ unzip -l /corp/nus/bulk_output/nsrl/43-1.Metro-X\ X11.6.3.1.2.Metro\ Link\ Inc..NOCARVE.zip
Archive:  /corp/nus/bulk_output/nsrl/43-1.Metro-X X11.6.3.1.2.Metro Link Inc..NOCARVE.zip
  Length      Date    Time    Name
---------  ---------- -----   ----
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./aes_keys.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./alerts.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./ccn.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./ccn_histogram.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./ccn_track2.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./ccn_track2_histogram.txt
   162721  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./domain.txt
     2733  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./domain_histogram.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./elf.txt
   126866  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./email.txt
      430  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./email_domain_histogram.txt
    13919  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./email_histogram.txt
      484  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./ether.txt
      332  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./ether_histogram.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./exif.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./find.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./find_histogram.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./gps.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./httplogs.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./ip.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./ip_histogram.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./jpeg_carved.txt
      360  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./json.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./kml.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./pii.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./pii_teamviewer.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./rar.txt
    17993  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./report.xml
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./rfc822.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./sqlite.txt
    14652  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./telephone.txt
     2386  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./telephone_histogram.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./unrar_carved.txt
        0  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./unzip_carved.txt
   107450  07-05-2014 21:15   43-1.Metro-X X11.6.3.1.2.Metro Link Inc./url.txt
```

Example: email.txt

```
# BULK_EXTRACTOR-Version: 1.5.0-alpha2 ($Rev: 10844 $)
# Feature-Recorder: email
# Filename: /Volumes/SanVol1/iRepository/media/43-1/ff81e9459a5fa057c2922ba0656414e336635248/
ff81e9459a5fa057c2922ba
0656414e336635248.img
# Feature-File-Version: 1.1
36079262 gary@mods.com.au    \x0DGary Smith at gary@mods.com.au\x0Dor visit http:/
36079706 gary@mods.com.au   \x00\x00Gary Smith at gary@mods.com.au\x00\x00\xFF\xFF\xFF\xFF&\x00\x00\x00o
r vis
102564008 CPS-requests@verisign.com   m; by E-mail at CPS-requests@verisign.com; or\x0Aby mail at
104028694 CPS-requests@verisign.com   m; by E-mail at CPS-requests@verisign.com; or\x0Aby mail at
104704969 CPS-requests@verisign.com   m; by E-mail at CPS-requests@verisign.com; or\x0Aby mail at
105144833 CPS-requests@verisign.com   m; by E-mail at CPS-requests@verisign.com; or\x0Aby mail at
117767109 CPS-requests@verisign.com   m; by E-mail at CPS-requests@verisign.com; or\x0Aby mail at
117979823 CPS-requests@verisign.com   m; by E-mail at CPS-requests@verisign.com; or\x0Aby mail at
108010472 CPS-requests@verisign.com   m; by E-mail at CPS-requests@verisign.com; or\x0Aby mail at
119507772 CPS-requests@verisign.com   m; by E-mail at CPS-requests@verisign.com; or\x0Aby mail at
...
223775147 james.king@prd-foods.com 5671\x04\x02 \x001\x00\x05\x00\x0F\x00\x18\x00james.king@prd-foods.c
om\x04\x02\x13\x002\x00\x00\x00\x0F\x00\x0B\x00Jori
223775296 jorie.meyer@prd-foods.com   5672\x04\x02!\x002\x00\x05\x00\x0F\x00\x19\x00jorie.meyer@prd-foods.
com\x04\x02\x17\x003\x00\x00\x00\x0F\x00\x0F\x00Glor
223775470 gloria.reinaldo@prd-foods.com  4357\x04\x02%\x003\x00\x05\x00\x0F\x00\x1D\x00gloria.reinaldo@prd-
fo
ods.com\x04\x02\x17\x004\x00\x00\x00\x0F\x00\x0F\x00Oliv
223775640 oliver.sullivan@prd-foods.com  4358\x04\x02%\x004\x00\x05\x00\x0F\x00\x1D\x00oliver.sullivan@prd-
fo
ods.com\x04\x02\x13\x005\x00\x00\x00\x0F\x00\x0B\x00Bill
223775806 bill.hughes@prd-foods.com   4359\x04\x02!\x005\x00\x05\x00\x0F\x00\x19\x00bill.hughes@prd-foods.
com\x04\x02\x16\x006\x00\x00\x00\x0F\x00\x0E\x00Crai
223775971 craig.anderson@prd-foods.com   4360\x04\x02$\x006\x00\x05\x00\x0F\x00\x1C\x00craig.anderson@prd-
fo
```

example: email_histogram.txt

```
# BULK_EXTRACTOR-Version: 1.5.0-alpha2 ($Rev: 10844 $)
# Feature-Recorder: email
# Filename: /Volumes/SanVol1/iRepository/media/43-1/ff81e9459a5fa057c2922ba0656414e336635248/
ff81e9459a5fa057c2922ba
0656414e336635248.img
# Histogram-File-Version: 1.1
n=101cps-requests@verisign.com
n=8  msccatus@microsoft.com
n=6  expresstools@autodesk.com
n=3  anonymous@microsoft.com
n=3  cindyl@neucom.com (utf16=3)
n=3  zeusprod@aol.com  (utf16=3)
n=2  aaron.ryan@prd-foods.com
n=2  aj.stensen@prd-foods.com
n=2  alfonso.camacho@prd-foods.com
n=2  bill.hughes@prd-foods.com
n=2  bill.mcgregor@prd-foods.com
n=2  bruce.gillmore@prd-foods.com
n=2  bryan.partridge@prd-foods.com
n=2  carlos.perez@prd-foods.com
n=2  carroll.engdahl@prd-foods.com
n=2  chandra.singh@prd-foods.com
n=2  chela.james@prd-foods.com
n=2  christian.ludwig@prd-foods.com
n=2  claytonc@neucom.com  (utf16=2)
n=2  cliffordy@neucom.com (utf16=2)
n=2  craig.anderson@prd-foods.com
```

bulk_extractor 1.5 stoplist bug fix

In BE1.4, all words with "." were treated as regular expressions.

- stoplists of email addresses were too slow to be usable.

BE1.5 performance scanning domexusers (40GB test image)

stop list	Lines	Time to process	Found email
None	**0**	**304 seconds**	**1050 distinct 8744 total**
NSRL email words	**502,697 (12MB)**	**296 seconds**	**286 distinct 3822 total**
NSRL email context	**45,231,320 (4.4GB)**	**371 seconds**	**364 distinct 3737 total**

Example output with and without stoplist

Baseline:

```
n=609 domexuser1@gmail.com(utf16=303)
n=455 domexuser2@gmail.com(utf16=225)
n=359 domexuser3@gmail.com(utf16=204)
n=268 ips@mail.ips.es
n=252 premium-server@thawte.com
n=243 cps-requests@verisign.com (utf16=3)
n=243 someone@example.com (utf16=234)
n=221 domexuser2@live.com (utf16=59)
n=198 domexuser1@hotmail.com (utf16=80)
n=185 domexuser1@live.com (utf16=59)
n=175 domexuser2@hotmail.com (utf16=97)
n=145 inet@microsoft.com
n=115 example@passport.com(utf16=115)
n=115 myname@msn.com(utf16=115)
n=94  info@valicert.com
n=91  piracy@microsoft.com(utf16=91)
n=80  certificate@trustcenter.de
n=78  name_123@hotmail.com(utf16=78)
n=74  talkback@mozilla.org(utf16=12)
n=69  hewitt@netscape.com (utf16=1)
n=64  lord@netscape.com
n=53  someone@microsoft.com  (utf16=48)
n=51  mcgreer@netscape.com
n=48  domexuser1%40gmail.com@imap.gmail.com
n=47  neil@parkwaycc.co.uk
n=43  49091023.6070302@gmail.com(utf16=22)
n=43  73a94919-ff6b-4e3f-938e-fb39bbc7497c@gmail
n=43  9name_123@hotmail.com  (utf16=43)
n=43  mazrob@panix.com
n=42  domex2@rad.li  (utf16=42)
```

With Stoplist:

```
n=609 domexuser1@gmail.com(utf16=303)
n=455 domexuser2@gmail.com(utf16=225)
n=359 domexuser3@gmail.com(utf16=204)
n=221 domexuser2@live.com (utf16=59)
n=198 domexuser1@hotmail.com (utf16=80)
n=185 domexuser1@live.com (utf16=59)
n=175 domexuser2@hotmail.com (utf16=97)
n=67  talkback@mozilla.org(utf16=6)
n=48  domexuser1%40gmail.com@imap.gmail.com  (utf16=16)
n=43  49091023.6070302@gmail.com(utf16=22)
n=43  73a94919-ff6b-4e3f-938e-fb39bbc7497c@gmail.com(ut
n=43  9name_123@hotmail.com  (utf16=43)
n=42  domex2@rad.li  (utf16=42)
n=36  49091664.70508@gmail.com  (utf16=22)
n=35  domex1@rad.ms  (utf16=35)
n=33  domesxuser2@gmail.com  (utf16=8)
n=27  domex1@www.ms  (utf16=27)
n=26  premium-server@thawte.com
n=25  outldomexuser2@gmail.com-00000002.ps  (utf16=25)
n=23  2domexuser2@gmail.com  (utf16=23)
n=23  314d3a220810291941w4b52597fh206faba1e5063365@mail
n=23  domex2@adopt.eu  (utf16=23)
n=20  domex2@bl135w.blu135.mail.li (utf16=20)
n=19  alguem@exemplo.pt(utf16=19)
n=19  cph@99841.pa
n=18  outldomexuser2@gmail.com-00000002.pst.tm  (utf16
n=17  managelinks.aspx%3fmkt%3den-us%26noteid%3dnote.li
      %3d1%26notesec%3d0%26username%3ddomexuser1@
```

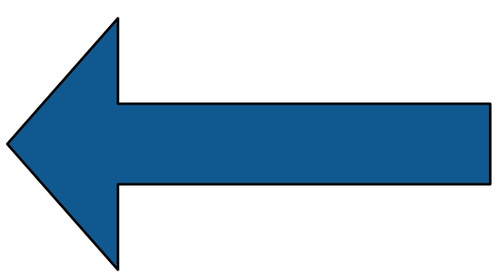
The included program bulk_diff.py will compare the two reports

```
$ python bulk_diff.py --help
Usage: usage: bulk_diff.py [options] <pre> <post>
<pre> and <post> may be a bulk_extractor output directory or a zip file of a
report.


Options:
  -h, --help          show this help message and exit
  --smaller           Also show values that didn't change or got smaller
  --same              Also show values that didn't change
  --tabdel=TABDEL     Specify a tab-delimited output file for easy import into
                      Excel
  --html=HTML         HTML output. Argument is file name base
```

Usage:

```
$ python bulk_diff.py  out-domexusers-stop-context out-domexusers-baseline
Files only in out-domexusers-stop-context:
     ip_stopped.txt (0 lines)
     email_stopped.txt (5007 lines)
     rar_stopped.txt (0 lines)
     rfc822_stopped.txt (0 lines)
     pii_stopped.txt (0 lines)
...
```



The differences are due to email addresses in distribution software being suppressed.

```
ccn_histogram.txt: No differences

ccn_track2_histogram.txt: No differences

domain_histogram.txt: No differences

email_histogram.txt:
# in PRE # in POST   Δ Value
-----------------------------------------------------------------
       0       268 268 ips@mail.ips.es
       0       243 243 cps-requests@verisign.com
       7       243 236 someone@example.com
      26       252 226 premium-server@thawte.com
       0       145 145 inet@microsoft.com
       0       115 115 example@passport.com
       0       115 115 myname@msn.com
       0        91  91 piracy@microsoft.com
       8        94  86 info@valicert.com
       0        80  80 certificate@trustcenter.de
       0        78  78 name_123@hotmail.com
       0        69  69 hewitt@netscape.com
       0        64  64 lord@netscape.com
       0        51  51 mcgreer@netscape.com
       4        53  49 someone@microsoft.com
       0        47  47 neil@parkwaycc.co.uk
       3        43  40 mazrob@panix.com
       0        37  37 server-certs@thawte.com
```

```
-rw-r--r--@ 1 simsong  staff       476 Jul  7 23:50 aes_keys.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 alerts.txt
-rw-r--r--@ 1 simsong  staff      2743 Jul  7 23:59 ccn.txt
-rw-r--r--@ 1 simsong  staff       454 Jul  8 00:03 ccn_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 ccn_track2.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 ccn_track2_histogram.txt
-rw-r--r--@ 1 simsong  staff  23369167 Jul  8 00:03 domain.txt
-rw-r--r--@ 1 simsong  staff    185266 Jul  8 00:03 domain_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 elf.txt
-rw-r--r--@ 1 simsong  staff   1719842 Jul  8 00:03 email.txt
-rw-r--r--@ 1 simsong  staff     35073 Jul  8 00:03 email_histogram.txt
-rw-r--r--@ 1 simsong  staff     23961 Jul  8 00:00 ether.txt
-rw-r--r--@ 1 simsong  staff       337 Jul  8 00:03 ether_histogram.txt
-rw-r--r--@ 1 simsong  staff  11188830 Jul  8 00:03 exif.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 find.txt
-rw-r--r--@ 1 simsong  staff      1112 Jul  8 00:01 gps.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 hex.txt
-rw-r--r--@ 1 simsong  staff     95835 Jul  8 00:03 ip.txt
-rw-r--r--@ 1 simsong  staff     11603 Jul  8 00:03 ip_histogram.txt
-rw-r--r--@ 1 simsong  staff   2025702 Jul  8 00:03 json.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 kml.txt
-rw-r--r--@ 1 simsong  staff    194991 Jul  8 00:03 packets.pcap
-rw-r--r--@ 1 simsong  staff     21343 Jul  8 00:03 report.xml
-rw-r--r--@ 1 simsong  staff   3782598 Jul  8 00:03 rfc822.txt
-rw-r--r--@ 1 simsong  staff    213746 Jul  8 00:03 tcp.txt
-rw-r--r--@ 1 simsong  staff     61255 Jul  8 00:03 tcp_histogram.txt
-rw-r--r--@ 1 simsong  staff     59469 Jul  8 00:03 telephone.txt
-rw-r--r--@ 1 simsong  staff      6612 Jul  8 00:03 telephone_histogram.txt
-rw-r--r--@ 1 simsong  staff  67205326 Jul  8 00:03 url.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_facebook-id.txt
-rw-r--r--@ 1 simsong  staff   5706665 Jul  8 00:03 url_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_microsoft-live.txt
-rw-r--r--@ 1 simsong  staff      8504 Jul  8 00:03 url_searches.txt
-rw-r--r--@ 1 simsong  staff    151673 Jul  8 00:03 url_services.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 vcard.txt
-rw-r--r--@ 1 simsong  staff  18549729 Jul  8 00:03 windirs.txt
-rw-r--r--@ 1 simsong  staff  29051041 Jul  8 00:03 winpe.txt
-rw-r--r--@ 1 simsong  staff   1984759 Jul  8 00:03 winprefetch.txt
-rw-r--r--@ 1 simsong  staff  34128889 Jul  8 00:03 zip.txt
```

bulk_extractor 1.5 scanners and output

There are four main categories of feature files:

Identity Information:

- Domain Names; Email addresses; URLs
- Search terms; Facebook IDs; JSON data
- KML files
- VCARDs
- find output

```
-rw-r--r--@ 1 simsong  staff       476 Jul  7 23:50 aes_keys.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 alerts.txt
-rw-r--r--@ 1 simsong  staff      2743 Jul  7 23:59 ccn.txt
-rw-r--r--@ 1 simsong  staff       454 Jul  8 00:03 ccn_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 ccn_track2.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 ccn_track2_histogram.txt
-rw-r--r--@ 1 simsong  staff  23369167 Jul  8 00:03 domain.txt
-rw-r--r--@ 1 simsong  staff    185266 Jul  8 00:03 domain_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 elf.txt
-rw-r--r--@ 1 simsong  staff   1719842 Jul  8 00:03 email.txt
-rw-r--r--@ 1 simsong  staff     35073 Jul  8 00:03 email_histogram.txt
-rw-r--r--@ 1 simsong  staff     23961 Jul  8 00:00 ether.txt
-rw-r--r--@ 1 simsong  staff       337 Jul  8 00:03 ether_histogram.txt
-rw-r--r--@ 1 simsong  staff  11188830 Jul  8 00:03 exif.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 find.txt
-rw-r--r--@ 1 simsong  staff      1112 Jul  8 00:01 gps.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 hex.txt
-rw-r--r--@ 1 simsong  staff     95835 Jul  8 00:03 ip.txt
-rw-r--r--@ 1 simsong  staff     11603 Jul  8 00:03 ip_histogram.txt
-rw-r--r--@ 1 simsong  staff   2025702 Jul  8 00:03 json.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 kml.txt
-rw-r--r--@ 1 simsong  staff    194991 Jul  8 00:03 packets.pcap
-rw-r--r--@ 1 simsong  staff     21343 Jul  8 00:03 report.xml
-rw-r--r--@ 1 simsong  staff   3782598 Jul  8 00:03 rfc822.txt
-rw-r--r--@ 1 simsong  staff    213746 Jul  8 00:03 tcp.txt
-rw-r--r--@ 1 simsong  staff     61255 Jul  8 00:03 tcp_histogram.txt
-rw-r--r--@ 1 simsong  staff     59469 Jul  8 00:03 telephone.txt
-rw-r--r--@ 1 simsong  staff      6612 Jul  8 00:03 telephone_histogram.txt
-rw-r--r--@ 1 simsong  staff  67205326 Jul  8 00:03 url.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_facebook-id.txt
-rw-r--r--@ 1 simsong  staff   5706665 Jul  8 00:03 url_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_microsoft-live.txt
-rw-r--r--@ 1 simsong  staff      8504 Jul  8 00:03 url_searches.txt
-rw-r--r--@ 1 simsong  staff    151673 Jul  8 00:03 url_services.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 vcard.txt
-rw-r--r--@ 1 simsong  staff  18549729 Jul  8 00:03 windirs.txt
-rw-r--r--@ 1 simsong  staff  29051041 Jul  8 00:03 winpe.txt
-rw-r--r--@ 1 simsong  staff   1984759 Jul  8 00:03 winprefetch.txt
-rw-r--r--@ 1 simsong  staff  34128889 Jul  8 00:03 zip.txt
```

Technical Info:

- ZIP files; EXIF data

Network Information:

- PCAP files; Ethernet Addresses; TCP/IP Connections; etc.

Information about executables:

- ELF & PE headers; Windows Prefetch files

There are four main categories of feature files:

Identity Information:

- Domain Names; Email addresses; URLs
- Search terms; Facebook IDs; JSON data
- KML files
- VCARDs
- find output

Technical Info:

- ZIP files; EXIF data

Network Information:

- PCAP files; Ethernet Addresses; TCP/IP Connections; etc.

Information about executables:

- ELF & PE headers; Windows Prefetch files

```
-rw-r--r--@ 1 simsong  staff       476 Jul  7 23:50 aes_keys.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 alerts.txt
-rw-r--r--@ 1 simsong  staff      2743 Jul  7 23:59 ccn.txt
-rw-r--r--@ 1 simsong  staff       454 Jul  8 00:03 ccn_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 ccn_track2.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 ccn_track2_histogram.txt
-rw-r--r--@ 1 simsong  staff  23369167 Jul  8 00:03 domain.txt
-rw-r--r--@ 1 simsong  staff    185266 Jul  8 00:03 domain_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 elf.txt
-rw-r--r--@ 1 simsong  staff   1719842 Jul  8 00:03 email.txt
-rw-r--r--@ 1 simsong  staff     35073 Jul  8 00:03 email_histogram.txt
-rw-r--r--@ 1 simsong  staff     23961 Jul  8 00:00 ether.txt
-rw-r--r--@ 1 simsong  staff       337 Jul  8 00:03 ether_histogram.txt
-rw-r--r--@ 1 simsong  staff  11188830 Jul  8 00:03 exif.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 find.txt
-rw-r--r--@ 1 simsong  staff      1112 Jul  8 00:01 gps.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 hex.txt
-rw-r--r--@ 1 simsong  staff     95835 Jul  8 00:03 ip.txt
-rw-r--r--@ 1 simsong  staff     11603 Jul  8 00:03 ip_histogram.txt
-rw-r--r--@ 1 simsong  staff   2025702 Jul  8 00:03 json.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 kml.txt
-rw-r--r--@ 1 simsong  staff    194991 Jul  8 00:03 packets.pcap
-rw-r--r--@ 1 simsong  staff     21343 Jul  8 00:03 report.xml
-rw-r--r--@ 1 simsong  staff   3782598 Jul  8 00:03 rfc822.txt
-rw-r--r--@ 1 simsong  staff    213746 Jul  8 00:03 tcp.txt
-rw-r--r--@ 1 simsong  staff     61255 Jul  8 00:03 tcp_histogram.txt
-rw-r--r--@ 1 simsong  staff     59469 Jul  8 00:03 telephone.txt
-rw-r--r--@ 1 simsong  staff      6612 Jul  8 00:03 telephone_histogram.txt
-rw-r--r--@ 1 simsong  staff  67205326 Jul  8 00:03 url.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_facebook-id.txt
-rw-r--r--@ 1 simsong  staff   5706665 Jul  8 00:03 url_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_microsoft-live.txt
-rw-r--r--@ 1 simsong  staff      8504 Jul  8 00:03 url_searches.txt
-rw-r--r--@ 1 simsong  staff    151673 Jul  8 00:03 url_services.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 vcard.txt
-rw-r--r--@ 1 simsong  staff  18549729 Jul  8 00:03 windirs.txt
-rw-r--r--@ 1 simsong  staff  29051041 Jul  8 00:03 winpe.txt
-rw-r--r--@ 1 simsong  staff   1984759 Jul  8 00:03 winprefetch.txt
-rw-r--r--@ 1 simsong  staff  34128889 Jul  8 00:03 zip.txt
```

ccn.txt — potential credit card numbers.

bulk_extractor's credit card number finder:

- Considers pattern of digits; Luhn algorithm; distribution of digits; local context
- Frequently alerts on "false positives," so be careful!

```
# Feature-Recorder: ccn
88284672-GZIP-177427    5273347458642687        73A4B55CE2234D5\x0A5273347458642687\x0AC0841BAFA1B4C28
4909069775      6543210123456788        \x0Addadd7540 add '6543210123456788' 0.499999999
4909069811      6543210123456788        499999999   -> '6543210123456788' Inexact Rounde
4909069861      6543210123456788        \x0Addadd7541 add '6543210123456788' 0.5
4909069897      6543210123456788        5           -> '6543210123456788' Inexact Rounde
4909069947      6543210123456788        \x0Addadd7542 add '6543210123456788' 0.500000001
4814857216-GZIP-793     4015751530102097        ebO.d=0;ebO.rnd=4015751530102097;ebO.title="";eb
5304221350      5678901234560000                +4   -> 5678901234560000\x0D\x0Addshi052 shift
5612375618      6543210123456788        \x0D\x0Aaddx6240 add '6543210123456788' 0.499999999
5612375654      6543210123456788        499999999   -> '6543210123456788' Inexact Rounde
5612375703      6543210123456788        \x0D\x0Aaddx6241 add '6543210123456788' 0.5
5612375739      6543210123456788        5           -> '6543210123456788' Inexact Rounde
5612375788      6543210123456788        \x0D\x0Aaddx6242 add '6543210123456788' 0.500000001
```

In this example:

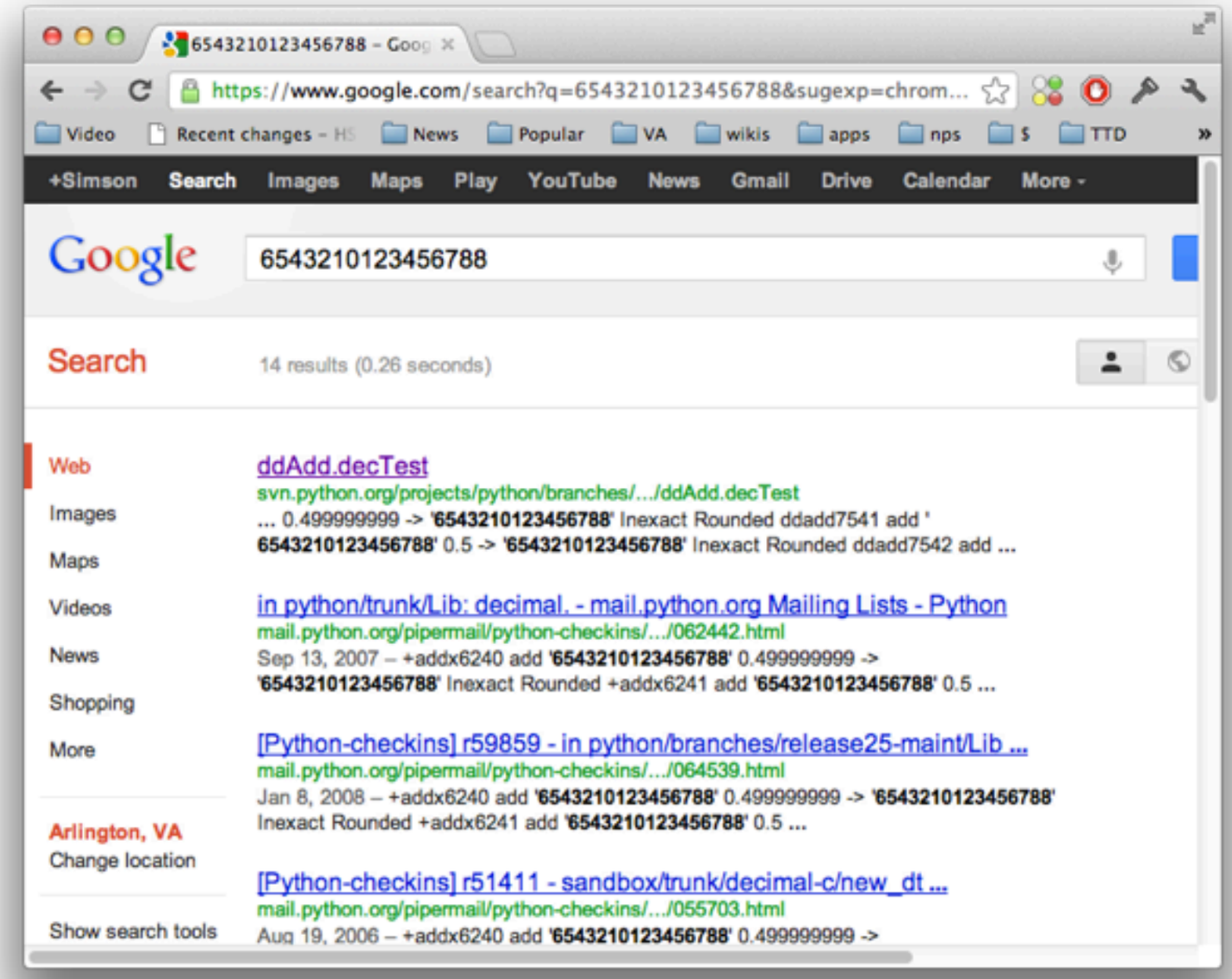
- 5273347458642687 looks like a valid CCN from the context (\x0A is a new line)
- 4015751530102097 looks like a random number in a piece of JavaScript
  - *Notice it was compressed! offset 4814857216 starts a GZIP stream; +793 bytes is CCN*
- "Inexact Rounde" is actually from the Python source code
  - *http://svn.python.org/projects/python/branches/pep-0384/Lib/test/decimaltestdata/ddAdd.decTest*

ccn_histogram.txt: a histogram of the potential credit card numbers

Normally this is a great way to find the real numbers...

```
n=20     6543210123456788
n=2      4015751530102097
n=2      4920919202474441
n=1      4857994530998756
n=1      4909616081396134
n=1      5235714985079914
n=1      5273347458642687
n=1      5578481572827551
n=1      5678901234560000
n=1      5700122152274696
```

This time it's a great way to find that python test data!



ccn_track2.txt contains potential “track 2” credit card number information

```
  Length  Date          Time      File
--------  ----------  --------  ----------------------------------------
     476   8-Jul-2012  01:50:32  charlie-2009-12-11/aes_keys.txt
       0   8-Jul-2012  01:48:36  charlie-2009-12-11/alerts.txt
    2743   8-Jul-2012  01:59:24  charlie-2009-12-11/ccn.txt
     454   8-Jul-2012  02:03:14  charlie-2009-12-11/ccn_histogram.txt
       0   8-Jul-2012  01:48:36  charlie-2009-12-11/ccn_track2.txt
       0   8-Jul-2012  02:03:14  charlie-2009-12-11/ccn_track2_histogram.txt
...
```

In this case we don’t have any track 2 data...

domain.txt is a list of all the potential "domains" and host names that were found. Sources include potential URLs, email, dotted quads.

```
50395405        \x00h\x00o\x00t\x00m\x00a\x00i\x00l\x00.\x00c\x00o\x00m\x00
\x00b\x00r\x00e\x00_\x001\x002\x003\x00@\x00h\x00o\x00t\x00m\x00a\x00i\x00l
\x00.\x00c\x00o\x00m\x00\x0A\x00\x09\x00m\x00i\x00n\x00o\x00m\x00b\x00


235154  www.microsoft.com       teUrl = "http://www.microsoft.com/isapi/redir.dll

257091  www.DocURL.com  _404.htm#http://www.DocURL.com/bar.htm \x0D\x0A\x0D\x0A

169692672-GZIP-4139     us.ard.yahoo.com        8" href="http://us.ard.yahoo.com/
SIG=15s920d26/M

148770304-GZIP-63217    www.bakersfield.com     n value="http://
www.bakersfield.com">CA, Bakersfiel

148770304-GZIP-63295    www.thebakersfieldchannel.com   n value="http://
www.thebakersfieldchannel.com">CA, Bakersfiel

27766700        205.155.65.61   ustang.nps.edu [205.155.65.61])\x0D\x0A\x09(using
27766902        m57.biz \x0D\x0A\x09for <charlie@m57.biz>; Mon, 16 Nov 2
```

Note:

- UTF-16 is "escaped" as Python-style — \x00h\x00o\x00t means "hot"
- Domains are common in compressed data

domain_histogram.txt is a histogram of the domains...

```
# UTF-8 Byte Order Marker; see http://unicode.org/faq/utf_bom.html
# bulk_extractor-Version: 1.3b1-dev2
# Filename: /corp/nps/drives/nps-2009-m57-patents-redacted/charlie-2009-12-11.E01
# Feature-Recorder: domain
# Histogram-File-Version: 1.1
n=10749 www.w3.org
n=6670  chroniclingamerica.loc.gov
n=6384  openoffice.org
n=5998  www.uspto.gov
n=5733  www.mozilla.org
n=5212  www.osti.gov
n=4952  www.microsoft.com
n=4474  patft.uspto.gov
n=4468  www.gpo.gov
n=3653  www.verisign.com
n=3167  www.google.com
n=3150  www.wipo.int
n=2733  news.bbc.co.uk
n=2595  crl.microsoft.com
```

Many of these domains are part of the operating system. Some aren't.

email.txt is similar to domain.txt, but has the potential email addresses!

```
50395384        n\x00o\x00m\x00b\x00r\x00e\x00_\x001\x002\x003\x00@\x00h\x00o
\x00t\x00m\x00a\x00i\x00l\x00.\x00c\x00o\x00m\x00  e\x00m\x00p\x001\x00o
\x00:\x00\x0A\x00\x09\x00n\x00o\x00m\x00b\x00r\x00e\x00_\x001\x002\x003\x00@\x00h
\x00o\x00t\x00m\x00a\x00i\x00l\x00.\x00c\x00o\x00m\x00\x0A\x00\x09\x00m\x00i\x00n
\x00o\x00m\x00b\x00

50395432        m\x00i\x00n\x00o\x00m\x00b\x00r\x00e\x00@\x00m\x00s\x00n
\x00.\x00c\x00o\x00m\x00        i\x00l\x00.\x00c\x00o\x00m\x00\x0A\x00\x09\x00m
\x00i\x00n\x00o\x00m\x00b\x00r\x00e\x00@\x00m\x00s\x00n\x00.\x00c\x00o\x00m
\x00\x0A\x00\x09\x00e\x00j\x00e\x00m\x00p\x001\x00
```

— *minombre@msn.com* — *myname@msn.com?*

— *50395384 is very early in the disk...*

Further down we see:

```
828564544       charlie@m57.biz (37190)\x0D\x0A\x09 for <charlie@m57.biz>; Fri,
20 Nov 2

828564992       4B01C378.3060603@m57.biz        0\x0D\x0AReferences:
<4B01C378.3060603@m57.biz>\x0D\x0ATo: charlie@m

828565023       charlie@m57.biz 3@m57.biz>\x0D\x0ATo: charlie@m57.biz\x0D
\x0ASubject: Still
```

email_histogram.txt shows a histogram of all potential email addresses

Clearly the histogram makes a difference:

```
# UTF-8 Byte Order Marker; see http://unicode.org/faq/utf_bom.html
# bulk_extractor-Version: 1.3b1-dev2
# Filename: /corp/nps/drives/nps-2009-m57-patents-redacted/charlie-2009-12-11.E01
# Feature-Recorder: email
# Histogram-File-Version: 1.1
n=875   mozilla@kewis.ch          (utf16=3)
n=651   charlie@m57.biz           (utf16=120)
n=605   ajbanck@planet.nl
n=411   mikep@oeone.com
n=395   belhaire@ief.u-psud.fr
n=379   premium-server@thawte.com   (utf16=11)
n=356   lilmatt@mozilla.com
n=312   cedric.corazza@wanadoo.fr
```

Notice:

- Charlie's email is #2 (it would probably be #1 if the disk had been used for more than 3 weeks)
- Charlie's email appeared 651 times; 120 of those were in UTF-16.
- Many of these email addresses are from the software (ajbanck@planet.nl is in Mozilla Calendar)

find.txt is the result of the ‘find’ command

```
-rw-r--r--@  1 simsong  staff         0 Jul  7 23:48 find.txt
```

But we can run with the find command (-f) to do a string search.

- Here we look for any mentions of ‘nps.edu’ (any case) in charlie-2009-12-11

```
$ bulk_extractor -f '[nN][pP][sS].[eE][dD][uU]' -o charlie-2009-12-11-find /corp/
nps/scenarios/2009-m57-patents/drives-redacted/charlie-2009-12-11.E01
...
elapsed time: 1787.12 seconds
$
```

- The string search is executed as a first-class scanner (so it goes in compressed data)

```
27766691    nps.edu   ps.edu (mustang.nps.edu [205.155.65.61]
27767031    nps.edu    http://mustang.nps.edu:80/cgi-bin/mark
...
3449724105  nps.edu   wall at mustang.nps.edu. I'm sorry to i
3445904906  nps.edu   ED0A1F1@mustang.nps.edu)(25C=fe0)(261\x0D\x0A
...
9976666871  nps.edu   $\xA0"\x1B cifs/domex.nps.edu@DOMEX.NPS.EDU\x00\x00
9976666885  NPS.EDU   x.nps.edu@DOMEX.NPS.EDU\x00\x00\x00\x00\x0D\x00\x01\x00Fil
\xE5\x01\x00\x15\x02
```

Note that these “domains” are *not* included in the domain histogram!

json.txt — JavaScript Object Notation (Facebook, etc.)

Provides offset, JSON, and MD5 of JSON

- Use the MD5 for deduplication

```
# UTF-8 Byte Order Marker; see http://unicode.org/faq/utf_bom.html
# bulk_extractor-Version: 1.3b1-dev2
# Filename: /corp/nps/drives/nps-2009-m57-patents-redacted/charlie-2009-12-11.E01
# Feature-Recorder: json
# Feature-File-Version: 1.1
```

```
5091418457      [6, 4, 6, 4]    7ea5995a7acbd301b98e15b50b723e2b
5091418525      [6, 4, 6, 4]    7ea5995a7acbd301b98e15b50b723e2b

10002203123     {"url":"http://patft.uspto.gov/netacgi/nph-Parser?
Sect1=PTO2&Sect2=HITOFF&p=1&u=%2Fnetahtml%2FPTO%2Fsearch-
bool.html&r=26265.169749810822&f=G&l=50&co1=AND&d=PTXT&s1=mortality&OS=mortality&
RS=mortality","title":"United States Patent: 4035489","ID":63,"scroll":"0,0"}
81e95912dbb0e7e0966a9becf1c9f74a
```

Excellent for web pages:

- bulk_extractor is great at finding JSON in compressed streams, HIBER files, etc.
- There is a huge amount of stuff here

rfc822.txt — potential Email headers, HTTP headers, and more

```
114072068       SUBJECT:no investment   l investment\x5Cb\x00\x00SUBJECT:no
investment\x00\x00\x00SUBJECT:no gi
114072092       SUBJECT:no gimmick\x5Cb no investment\x00\x00\x00SUBJECT:no
gimmick\x5Cb\x00\x00\x00\x00SUBJECT:\x5Cbno
114072116       SUBJECT:\x5Cbno refund  no gimmick\x5Cb\x00\x00\x00\x00SUBJECT:
\x5Cbno refund\x00SUBJECT:\x5Cbno ag
114072136       SUBJECT:\x5Cbno age (restriction|limit) ECT:\x5Cbno refund
\x00SUBJECT:\x5Cbno age (restriction|limit)\x00\x00\x00\x00SUBJECT:\x5Cbno
114072176       SUBJECT:\x5Cbno medical exam    ction|limit)
\x00\x00\x00\x00SUBJECT:\x5Cbno medical exam\x00\x00\x00SUBJECT:no st
114072204       SUBJECT:no strings attached\x5Cb         medical exam
\x00\x00\x00SUBJECT:no strings attached\x5Cb\x00\x00\x00SUBJECT:no pu

1167648701      Host: www.ferrari.com   ss.js HTTP/1.1\x0D\x0AHost:
www.ferrari.com\x0D\x0AUser-Agent: Mo
1167649049      Cookie:
__utma=157168684.1746400801.1258507160.1260220504.1260306908.3; __utmz=157168684
ages/Home.aspx\x0D\x0ACookie:
__utma=157168684.1746400801.1258507160.1260220504.1260306908.3;
__utmz=157168684.1258507160.1.1.
```

We would like to have better reporting of mail headers.

— *Combining email address and name*

telephone.txt — potential Phone numbers!

Beware — many are tech support!

```
88850883        (800) 563-9048  rmation centre: (800) 563-9048\x0D\x0A<BR><b><i>Tech
88850995        (905) 568-4494  indows 95: (905) 568-4494\x0D\x0A<BR> Microsoft
88851056        (905) 568-2294  ice components: (905) 568-2294\x0D\x0A<BR> Other sta
88851111        (905) 568-3503  hnical support: (905) 568-3503\x0D\x0A<BR> Priority
88851162        (800) 668-7975  rt information: (800) 668-7975\x0D\x0A<BR> Text Tele
88851208        (905) 568-9641  phone (TTY/TDD) (905) 568-9641</P>\x0D\x0A\x0D\x0A<id ID="
88851367        (809) 273-3600  nc.\x0D\x0A<BR>Phone: (809) 273-3600\x0D\x0A<BR>Fax: (809)
```

Some are bogus:

```
9935481243      177-188-1984    chives/techbull/177-188-1984.pdf\x0Ahttp://chan
10038401194     252.227-7013    clause at DFARS 252.227-7013 or subparagraph
10051668126     177-188-1984    chives/techbull/177-188-1984.pdf\x0Ahttp://www.
10051721420     118/150/1746    /filestorage/78/118/150/1746/1159/1268/Augus
10051801958     177-188-1984    chives/techbull/177-188-1984.pdf\x0Ahttp://www.
```

And some are clearly legit:

```
6561037824-GZIP-28322   (831) 373-5555  onterey - <nobr>(831) 373-5555</nobr><br><a cl
6561037824-GZIP-29518   (831) 899-8300  Seaside - <nobr>(831) 899-8300</nobr><br><a cl
6561037824-GZIP-31176   (831) 899-8300  Seaside - <nobr>(831) 899-8300</nobr><br><a cl
```

telephone_histogram.txt: Usually a better place to look for potential phone numbers

```
# Filename: /corp/nps/drives/nps-2009-m57-patents-redacted/charlie-2009-12-11.E01
# Feature-Recorder: telephone
# Histogram-File-Version: 1.1
n=42    4159618830
n=35    8477180400
n=24    2225552222
n=24    +27112570000
n=18    8005043248
n=15    2225551111
n=12    8772768437
n=11    2522277013
n=11    8662347350
n=9     1115554444
n=9     1771881984
n=8     4253532287
```

In version 1.3, non-digits are extracted from phone number.

url.txt: potential URLs

Note that UTF-16 data is escaped:

```
53946664        http://patimg2.uspto.gov/.piw?Docid=07626151&homeurl=http%3A%2F
%2Fpatft.uspto.gov%2Fnetacgi%2Fnph-Parser%3FSect1%3DPTO2%2526Sect2%3DHIT center>
\x0A<a href=http://patimg2.uspto.gov/.piw?Docid=07626151&homeurl=http%3A%2F
%2Fpatft.uspto.gov%2Fnetacgi%2Fnph-Parser%3FSect1%3DPTO2%2526Sect2%3DHITOFF%2526p
%3D1%25

34913630        H\x00T\x00T\x00P\x00:\x00/\x00/\x00     n\x00a\x00s\x00 \x00W
\x00e\x00b\x00 \x00H\x00T\x00T\x00P\x00:\x00/\x00/\x00 \x00e\x00 \x00H\x00T\x00T
\x00P\x00S\x00

34913650        H\x00T\x00T\x00P\x00S\x00:\x00/\x00/\x00.\x00   T\x00P
\x00:\x00/\x00/\x00 \x00e\x00 \x00H\x00T\x00T\x00P\x00S\x00:\x00/\x00/\x00.\x00
\x00N\x00o\x00 \x00e\x00n\x00t\x00a\x00

53952231        http://appft1.uspto.gov/netacgi/nph-Parser?
TERM1=20020186464&Sect1=PTO1&Sect2=HITOFF&d=PG01&p=1&u=%2Fnetahtml%2FPTO
%2Fsrchnum.html&r=0& =left>\x0A<a href="http://appft1.uspto.gov/netacgi/nph-
Parser?TERM1=20020186464&Sect1=PTO1&Sect2=HITOFF&d=PG01&p=1&u=%2Fnetahtml%2FPTO
%2Fsrchnum.html&r=0&f=S&l=50" target
```

url_histogram.txt: potential URLs from the disk UTF-16 is converted to UTF8

```
n=2     http://ebiz1.uspto.gov/vision-service/ShoppingCart_P/AddToShoppingCart?
docNumber=7626465&backUrl1=http%3A//patft1.uspto.gov/netacgi/nph
n=2     http://ebiz1.uspto.gov/vision-service/ShoppingCart_P/AddToShoppingCart?
docNumber=7627056&backUrl1=http%3A//patft1.uspto.gov/netacgi/nph
```

Note:

```
n=1022  http://www.uspto.gov/patft/help/help.htm        (utf16=3)
n=992   http://www.uspto.gov/patft/index.html           (utf16=4)
```

Not all URLs are accurate:

```
n=3922  http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul   (utf16=2609)

n=859   http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xu    (utf16=858)
```

url_facebook, url_histogram, url_microsoft-live, url_searches and url_services pull info out of URLs...

```
-rw-r--r--@  1 simsong  staff        0 Jul  8 00:03 url_facebook-id.txt
-rw-r--r--@  1 simsong  staff  5706665 Jul  8 00:03 url_histogram.txt
-rw-r--r--@  1 simsong  staff        0 Jul  8 00:03 url_microsoft-live.txt
-rw-r--r--@  1 simsong  staff     8504 Jul  8 00:03 url_searches.txt
-rw-r--r--@  1 simsong  staff   151673 Jul  8 00:03 url_services.txt
```

The most useful is url_searches.txt:

```
n=59    1
n=53    exotic+car+dealer
n=41    ford+car+dealer
n=34    2009+Shelby
n=25    steganography
n=23    General+Electric
n=23    time+travel
n=19    steganography+tool+free
n=19    vacation+packages
n=16    firefox
n=16    quicktime
n=14    7zip
n=14    fox+news
n=13    hex+editor
```

Searches frequently convey intent.

There are four main categories of feature files:

Identity Information:

- Domain Names; Email addresses; URLs
- Search terms; Facebook IDs; JSON data
- KML files
- VCARDs
- find output

```
-rw-r--r--@ 1 simsong  staff       476 Jul  7 23:50 aes_keys.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 alerts.txt
-rw-r--r--@ 1 simsong  staff      2743 Jul  7 23:59 ccn.txt
-rw-r--r--@ 1 simsong  staff       454 Jul  8 00:03 ccn_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 ccn_track2.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 ccn_track2_histogram.txt
-rw-r--r--@ 1 simsong  staff  23369167 Jul  8 00:03 domain.txt
-rw-r--r--@ 1 simsong  staff    185266 Jul  8 00:03 domain_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 elf.txt
-rw-r--r--@ 1 simsong  staff   1719842 Jul  8 00:03 email.txt
-rw-r--r--@ 1 simsong  staff     35073 Jul  8 00:03 email_histogram.txt
-rw-r--r--@ 1 simsong  staff     23961 Jul  8 00:00 ether.txt
-rw-r--r--@ 1 simsong  staff       337 Jul  8 00:03 ether_histogram.txt
-rw-r--r--@ 1 simsong  staff  11188830 Jul  8 00:03 exif.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 find.txt
-rw-r--r--@ 1 simsong  staff      1112 Jul  8 00:01 gps.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 hex.txt
-rw-r--r--@ 1 simsong  staff     95835 Jul  8 00:03 ip.txt
-rw-r--r--@ 1 simsong  staff     11603 Jul  8 00:03 ip_histogram.txt
-rw-r--r--@ 1 simsong  staff   2025702 Jul  8 00:03 json.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 kml.txt
-rw-r--r--@ 1 simsong  staff    194991 Jul  8 00:03 packets.pcap
-rw-r--r--@ 1 simsong  staff     21343 Jul  8 00:03 report.xml
-rw-r--r--@ 1 simsong  staff   3782598 Jul  8 00:03 rfc822.txt
-rw-r--r--@ 1 simsong  staff    213746 Jul  8 00:03 tcp.txt
-rw-r--r--@ 1 simsong  staff     61255 Jul  8 00:03 tcp_histogram.txt
-rw-r--r--@ 1 simsong  staff     59469 Jul  8 00:03 telephone.txt
-rw-r--r--@ 1 simsong  staff      6612 Jul  8 00:03 telephone_histogram.txt
-rw-r--r--@ 1 simsong  staff  67205326 Jul  8 00:03 url.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_facebook-id.txt
-rw-r--r--@ 1 simsong  staff   5706665 Jul  8 00:03 url_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_microsoft-live.txt
-rw-r--r--@ 1 simsong  staff      8504 Jul  8 00:03 url_searches.txt
-rw-r--r--@ 1 simsong  staff    151673 Jul  8 00:03 url_services.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 vcard.txt
-rw-r--r--@ 1 simsong  staff  18549729 Jul  8 00:03 windirs.txt
-rw-r--r--@ 1 simsong  staff  29051041 Jul  8 00:03 winpe.txt
-rw-r--r--@ 1 simsong  staff   1984759 Jul  8 00:03 winprefetch.txt
-rw-r--r--@ 1 simsong  staff  34128889 Jul  8 00:03 zip.txt
```

Technical Info:

- ZIP files; EXIF data

Network Information:

- PCAP files; Ethernet Addresses; TCP/IP Connections; etc.

Information about executables:

- ELF & PE headers; Windows Prefetch files

aes_keys.txt — scheduled potential AES encryption keys, usually found in RAM, Swap, or hibernation files

```
     # BULK_EXTRACTOR-Version: 1.5.0-alpha8 ($Rev: 10844 $)
# Feature-Recorder: aes_keys
# Filename: /home/slgarfin/corp/nus/drives/HU/HU001-0001/HU001-0001.E01
# Feature-File-Version: 1.1
2358637804      15 4d 75 26 33 71 92 ef 0a ea 2a d7 f2 5f 41 a1 7f 87 da aa ec 89 e2 83 ef 2d 18 d6 7c 22 6a b8         AES256
2358638380      eb 5f 49 3a 8c a6 d8 c3 82 a7 09 0a b2 e9 19 74 d4 7c f8 d2 0a 3a 72 44 39 af ce 7b ca ea 1e e1         AES256
5841949156      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5842372252      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5851473204      d1 21 8d f6 3f 50 8b 10 86 26 cc 3c 01 55 ac 19         AES128
5851743468      d1 21 8d f6 3f 50 8b 10 86 26 cc 3c 01 55 ac 19         AES128
5853526188      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5756942508      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5758633628      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5868052964      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5805737044      d1 21 8d f6 3f 50 8b 10 86 26 cc 3c 01 55 ac 19         AES128
5811143140      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5936750156      d1 21 8d f6 3f 50 8b 10 86 26 cc 3c 01 55 ac 19         AES128
5937668260      d1 21 8d f6 3f 50 8b 10 86 26 cc 3c 01 55 ac 19         AES128
5946947756      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5893453996      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5897967260      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5797285772      d1 21 8d f6 3f 50 8b 10 86 26 cc 3c 01 55 ac 19         AES128
5799444364      d1 21 8d f6 3f 50 8b 10 86 26 cc 3c 01 55 ac 19         AES128
5802143628      d1 21 8d f6 3f 50 8b 10 86 26 cc 3c 01 55 ac 19         AES128
5803496876      d1 21 8d f6 3f 50 8b 10 86 26 cc 3c 01 55 ac 19         AES128
5994456732      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5994477740      00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
5715677156      d1 21 8d f6 3f 50 8b 10 86 26 cc 3c 01 55 ac 19         AES128
10563805900     00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
57118442068     00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f         AES256
```

- Three AES keys, appearing 1, 2 and 10 times
- Keys “01 02 03 04 05 06 07 … “ is a test vector seen on many Windows systems.
-

exif.txt is a list of every potential EXIF that is found on the drive

This feature file has a different internal formatting:

[offset] [MD5 of first 4K of JPEG] [XML encoding of EXIF]

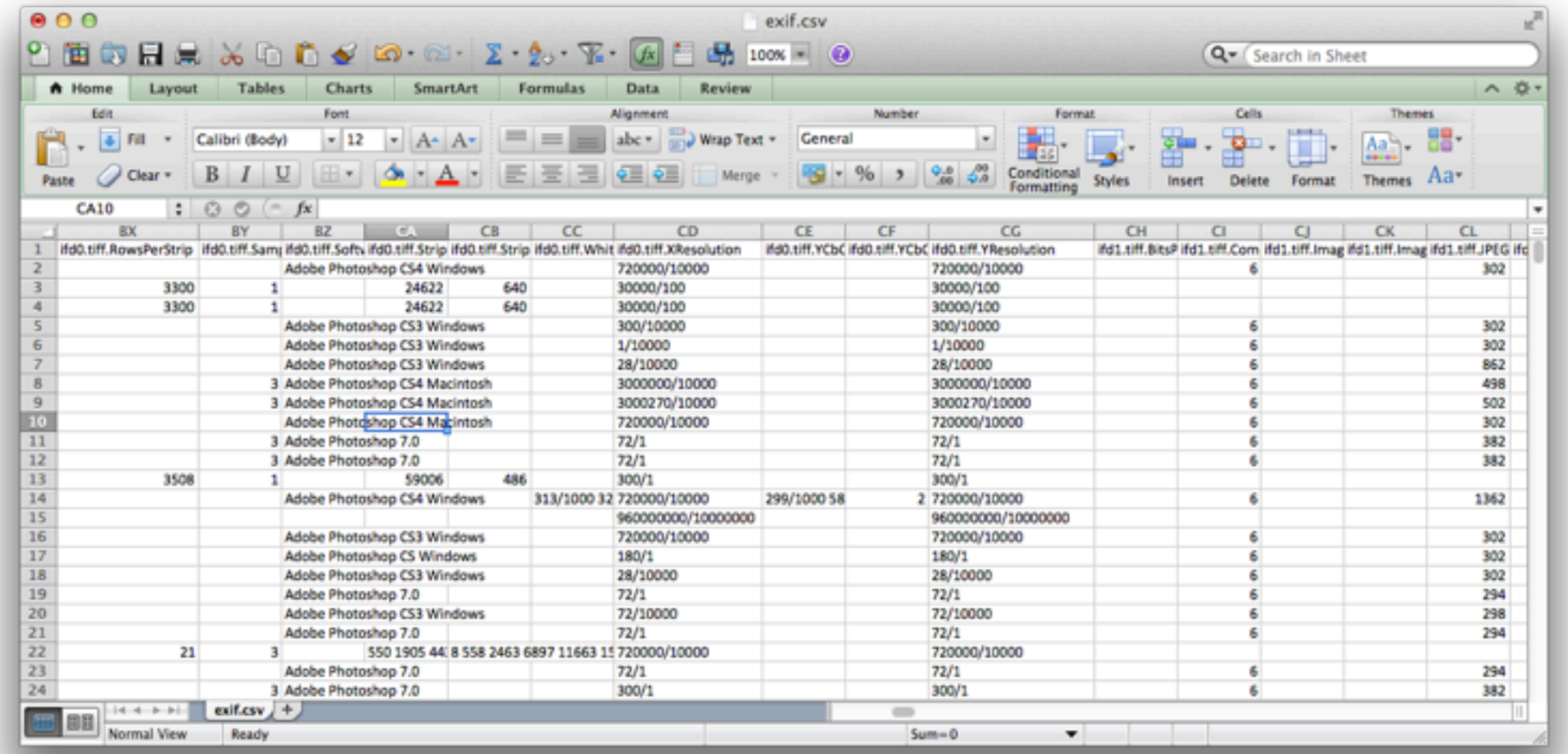
These files are *really* hard to understand...

```
75177472        15c1c6c52b34d64e368ffbf04bd14596        <exif><ifd0.tiff.Orientation>1</ifd0.tiff.Or\
ientation><ifd0.tiff.XResolution>720000/10000</ifd0.tiff.XResolution><ifd0.tiff.YResolution>720000/1\
0000</ifd0.tiff.YResolution><ifd0.tiff.ResolutionUnit>2</ifd0.tiff.ResolutionUnit><ifd0.tiff.Softwar\
e>Adobe Photoshop CS4 Windows</ifd0.tiff.Software><ifd0.tiff.DateTime>2009:09:03 14:30:07</ifd0.tiff\
.DateTime><ifd0.exif.ColorSpace>65535</ifd0.exif.ColorSpace><ifd0.exif.PixelXDimension>140</ifd0.exi\
f.PixelXDimension><ifd0.exif.PixelYDimension>96</ifd0.exif.PixelYDimension><ifd1.tiff.Compression>6<\
/ifd1.tiff.Compression><ifd1.tiff.XResolution>72/1</ifd1.tiff.XResolution><ifd1.tiff.YResolution>72/\
1</ifd1.tiff.YResolution><ifd1.tiff.ResolutionUnit>2</ifd1.tiff.ResolutionUnit><ifd1.tiff.JPEGInterc\
hangeFormat>302</ifd1.tiff.JPEGInterchangeFormat><ifd1.tiff.JPEGInterchangeFormatLength>4542</ifd1.t\
iff.JPEGInterchangeFormatLength></exif>
78016000        00000000000000000000000000000000        <exif><ifd0.tiff.entry_0x00fe>0</ifd0.tiff.e\
ntry_0x00fe><ifd0.tiff.entry_0x00ff>1</ifd0.tiff.entry_0x00ff><ifd0.tiff.ImageWidth>2560</ifd0.tiff.\
ImageWidth><ifd0.tiff.ImageLength>3300</ifd0.tiff.ImageLength><ifd0.tiff.BitsPerSample>1</ifd0.tiff.\
BitsPerSample><ifd0.tiff.Compression>4</ifd0.tiff.Compression><ifd0.tiff.PhotometricInterpreation>0<\
/ifd0.tiff.PhotometricInterpreation><ifd0.tiff.entry_0x010a>1</ifd0.tiff.entry_0x010a><ifd0.tiff.ent\
ry_0x010d>US020090196419A120090806</ifd0.tiff.entry_0x010d><ifd0.tiff.ImageDescription>00010001</ifd\
0.tiff.ImageDescription><ifd0.tiff.StripOffsets>640</ifd0.tiff.StripOffsets><ifd0.tiff.Orientation>1\
</ifd0.tiff.Orientation><ifd0.tiff.SamplesPerPixel>1</ifd0.tiff.SamplesPerPixel><ifd0.tiff.RowsPerSt\
rip>3300</ifd0.tiff.RowsPerStrip><ifd0.tiff.StripByteCounts>24622</ifd0.tiff.StripByteCounts><ifd0.t\
iff.entry_0x0118>0</ifd0.tiff.entry_0x0118><ifd0.tiff.entry_0x0119>1</ifd0.tiff.entry_0x0119><ifd0.t\
iff.XResolution>30000/100</ifd0.tiff.XResolution><ifd0.tiff.YResolution>30000/100</ifd0.tiff.YResolu\
tion><ifd0.tiff.entry_0x0125>0</ifd0.tiff.entry_0x0125><ifd0.tiff.ResolutionUnit>2</ifd0.tiff.Resolu\
tionUnit><ifd0.tiff.DateTime>2009:07:20 13:00:56</ifd0.tiff.DateTime><ifd0.tiff.entry_0x03e7>       \
                                                                                                \
```

Fortunately, we have a program that turns it into a spreadsheet...

python/post_process_exif.py

```
$ python bulk_extractor-1.3/python/post_process_exif.py exif.txt exif.csv
Input file: exif.txt
Output file: exif.csv
Scanning for EXIF tags...
There are 856 exif tags
$
$ open exif.csv
```



- Still not great, but at least you can search it and re-arrange the columns.

gps.txt shows times and potential GPS info extracted from JPEGs and Garmin XML files.

```
# UTF-8 Byte Order Marker; see http://unicode.org/faq/utf_bom.html
# bulk_extractor-Version: 1.3b1-dev2
# Filename: /corp/nps/drives/nps-2009-m57-patents-redacted/charlie-2009-12-11.E01
# Feature-Recorder: gps
# Feature-File-Version: 1.1
258600448       fc0bae6e33d1bd9f286461b816d957d9        2009-10-21T17:16:59,,,,,
3775933952      ff9fd0fbe87c91be5a74c4f1eadf75c4        2009-05-30T18:32:56,,,,,
3776724480      67d71def1274fd56d718063da0023247        2009-12-07T16:18:29,,,,,
4549410304      8d491ae0e448c466bd2ecc36554f0e03        2008-10-16T17:54:44,,,,,
5185666560      9b258122d51cf340d680e88504ccc23f        2009-09-23T10:11:50,,,,,
6619950592      de9017779919ea62e3a31fbb5f8c31ed        2009-11-09T16:13:00,,,,,
6620278272      1b0ce3f082d96cd9009b68e936c19da8        2009-03-01T12:55:23,,,,,
9066278400      9b258122d51cf340d680e88504ccc23f        2009-09-23T10:11:50,,,,,
9066794496      ff9fd0fbe87c91be5a74c4f1eadf75c4        2009-05-30T18:32:56,,,,,
9069837939      00000000000000000000000000000000        2009-12-07T16:18:29,,,,,
9069891072      67d71def1274fd56d718063da0023247        2009-12-07T16:18:29,,,,,
9076293235      00000000000000000000000000000000        ,invalid entry type code:
3328,,0.000000,,invalid entry type code: 0
```

This is interesting because it's data from other devices (cameras, etc.)

-

hex.txt is extracted hexdecimal strings of special lengths.

This disk image doesn’t have any...

Uses:

- emailed strings of MD5 codes, AES keys, etc.
- Anything else?

windirs.txt — potential FAT32 and NTFS directory entries

You will find most of the disk entries:

```
3230706176      EXCH_ntfsdrv.dll        <fileobject src='mft'>
  <atime>2009-11-09T01:24:59Z</atime><attr_flags>2080</attr_flags>
  <crtime>2009-11-09T01:24:59Z</crtime><ctime>2009-11-09T01:24:59Z</ctime>
  <filename>EXCH_ntfsdrv.dll</filename><filesize>38912</filesize>
  <filesize_alloc>40960</filesize_alloc><lsn>123102367</lsn>
  <mtime>2001-08-18T06:36:28Z</mtime><nlink>2</nlink><par_ref>71</par_ref>
  <par_seq>1</par_seq><seq>2</seq></fileobject>

3230707200      ntio404.sys     <fileobject src='mft'>
  <atime>2009-11-09T01:24:59Z</atime><attr_flags>2080</attr_flags>
  <crtime>2008-04-14T12:00:00Z</crtime><ctime>2009-11-08T17:08:04Z</ctime>
  <filename>ntio404.sys</filename><filesize>34560</filesize>
  <filesize_alloc>65536</filesize_alloc><lsn>29295332</lsn>
  <mtime>2008-04-14T12:00:00Z</mtime><nlink>1</nlink><par_ref>71</par_ref>
  <par_seq>1</par_seq><seq>1</seq></fileobject>
```

Error rate for FAT32 is high; ignore these if drive is not FAT:

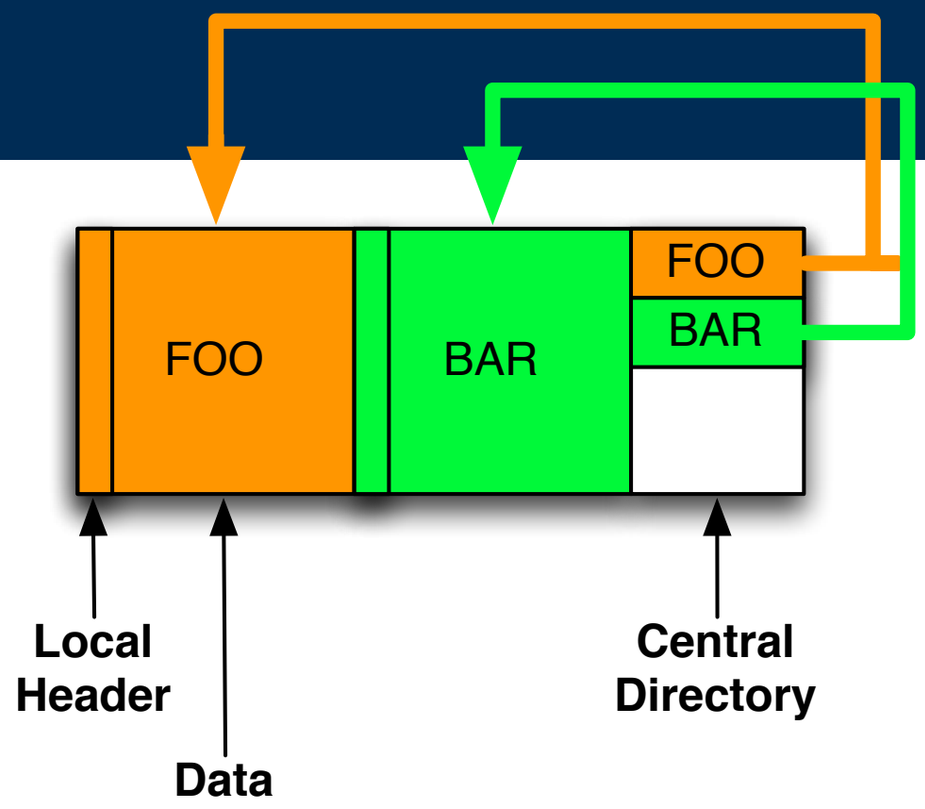
```
159466528       -eSigPol.icy    <fileobject src='fat'>
  <atime>2037-09-13T00:00:00</atime><attrib>45</attrib><ctime>2030-03-09T00:00:00</ctime>
  <ctimeten>56</ctimeten><filename>-eSigPol.icy</filename><filesize>1937007917</filesize>
  <mtime>2034-09-13T12:43:13</mtime><startcluster>1701667951</startcluster></fileobject>

173063680-GZIP-470016   dukdxd1o.lH7    <fileobject src='fat'>
  <atime>2010-09-29T00:00:00</atime><attrib>32</attrib><ctime>1999-09-25T06:34:01</ctime>
  <ctimeten>50</ctimeten><filename>dukdxd1o.lH7</filename><filesize>1632198449</filesize>
  <mtime>2007-01-18T15:01:17</mtime><startcluster>2016504113</startcluster></fileobject>
```

zip.txt — potential zipfile headers

ZIP has become the *defacto* archive format.

- zip, jar, docx, pptx, etc.
- ZIP64 provides for files larger than 4GiB
- Allows faster access to components that .tar.gz



bulk_extractor finds local file headers.

```
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)
```

zip.txt decodes every potential header of every zip archive

```
# Filename: /corp/nps/scenarios/2009-m57-patents/drives-redacted/
charlie-2009-12-11.E01
# Feature-Recorder: zip
# Feature-File-Version: 1.1
62865144        000024.tif      <zipinfo><name>000024.tif</name><name_len>10</
name_len><version>20</version><compression_method>8</
compression_method><uncompr_size>0</uncompr_size><compr_size>0</
compr_size><lastmodtime>8</lastmodtime><lastmoddate>34592</lastmoddate><crc32>0</
crc32><extra_field_len>0</extra_field_len><disposition
bytes='10846'>decompressed</disposition></zipinfo>

62874091        000025.tif      <zipinfo><name>000025.tif</name><name_len>10</
name_len><version>20</version><compression_method>8</
compression_method><uncompr_size>0</uncompr_size><compr_size>0</
compr_size><lastmodtime>8</lastmodtime><lastmoddate>34592</lastmoddate><crc32>0</
crc32><extra_field_len>0</extra_field_len><disposition
bytes='67680'>decompressed</disposition></zipinfo>
```

Possible uses:

- Identify MSOffice and OpenOffice documents
- Identify Java programs
- Reconstruct hierarchy

There are four main categories of feature files:

Identity Information:

- Domain Names; Email addresses; URLs
- Search terms; Facebook IDs; JSON data
- KML files
- VCARDs
- find output

Technical Info:

- ZIP files; EXIF data

Network Information:

- PCAP files; Ethernet Addresses; TCP/IP Connections; etc.

Information about executables:

- ELF & PE headers; Windows Prefetch files

```
-rw-r--r--@ 1 simsong  staff       476 Jul  7 23:50 aes_keys.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 alerts.txt
-rw-r--r--@ 1 simsong  staff      2743 Jul  7 23:59 ccn.txt
-rw-r--r--@ 1 simsong  staff       454 Jul  8 00:03 ccn_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 ccn_track2.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 ccn_track2_histogram.txt
-rw-r--r--@ 1 simsong  staff  23369167 Jul  8 00:03 domain.txt
-rw-r--r--@ 1 simsong  staff    185266 Jul  8 00:03 domain_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 elf.txt
-rw-r--r--@ 1 simsong  staff   1719842 Jul  8 00:03 email.txt
-rw-r--r--@ 1 simsong  staff     35073 Jul  8 00:03 email_histogram.txt
-rw-r--r--@ 1 simsong  staff     23961 Jul  8 00:00 ether.txt
-rw-r--r--@ 1 simsong  staff       337 Jul  8 00:03 ether_histogram.txt
-rw-r--r--@ 1 simsong  staff  11188830 Jul  8 00:03 exif.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 find.txt
-rw-r--r--@ 1 simsong  staff      1112 Jul  8 00:01 gps.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 hex.txt
-rw-r--r--@ 1 simsong  staff     95835 Jul  8 00:03 ip.txt
-rw-r--r--@ 1 simsong  staff     11603 Jul  8 00:03 ip_histogram.txt
-rw-r--r--@ 1 simsong  staff   2025702 Jul  8 00:03 json.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 kml.txt
-rw-r--r--@ 1 simsong  staff    194991 Jul  8 00:03 packets.pcap
-rw-r--r--@ 1 simsong  staff     21343 Jul  8 00:03 report.xml
-rw-r--r--@ 1 simsong  staff   3782598 Jul  8 00:03 rfc822.txt
-rw-r--r--@ 1 simsong  staff    213746 Jul  8 00:03 tcp.txt
-rw-r--r--@ 1 simsong  staff     61255 Jul  8 00:03 tcp_histogram.txt
-rw-r--r--@ 1 simsong  staff     59469 Jul  8 00:03 telephone.txt
-rw-r--r--@ 1 simsong  staff      6612 Jul  8 00:03 telephone_histogram.txt
-rw-r--r--@ 1 simsong  staff  67205326 Jul  8 00:03 url.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_facebook-id.txt
-rw-r--r--@ 1 simsong  staff   5706665 Jul  8 00:03 url_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_microsoft-live.txt
-rw-r--r--@ 1 simsong  staff      8504 Jul  8 00:03 url_searches.txt
-rw-r--r--@ 1 simsong  staff    151673 Jul  8 00:03 url_services.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 vcard.txt
-rw-r--r--@ 1 simsong  staff  18549729 Jul  8 00:03 windirs.txt
-rw-r--r--@ 1 simsong  staff  29051041 Jul  8 00:03 winpe.txt
-rw-r--r--@ 1 simsong  staff   1984759 Jul  8 00:03 winprefetch.txt
-rw-r--r--@ 1 simsong  staff  34128889 Jul  8 00:03 zip.txt
```

ether.txt and ether_histogram.txt: a list of potential ethernet addresses (from packets and ASCII)

```
342699417       00:80:77:31:01:07       n008077310107 1 00:80:77:31:01:07 192.168.1.2  an
342700437       00:80:77:31:01:07       the following:  00:80:77:31:01:07 brn008077310107
342703371       00:80:77:31:01:07        -s 192.168.1.2 00:80:77:31:01:07   ping 192.168.
559251251       00:80:77:31:01:07       N008077310107 1 00:80:77:31:01:07 192.168.1.2</sp
567912435       00:80:77:31:01:07       class="command">00:80:77:31:01:07   BRN0080773101
684600847       00:80:77:31:01:07        -s 192.168.1.2 00:80:77:31:01:07</span></div><a
6341279242      00:0B:DB:4F:6B:10        (ether_dhost)
6341279242      00:19:E3:E7:5D:23        (ether_shost)
6341283338      00:0B:DB:4F:6B:10        (ether_dhost)
6341283338      00:19:E3:E7:5D:23        (ether_shost)
6341287434      00:0B:DB:4F:6B:10        (ether_dhost)
```

```
# UTF-8 Byte Order Marker; see http://unicode.org/faq/utf_bom.html
# bulk_extractor-Version: 1.3b1-dev2
# Filename: /corp/nps/drives/nps-2009-m57-patents-redacted/charlie-2009-12-11.E01
# Feature-Recorder: ether
# Histogram-File-Version: 1.1
n=255   00:19:E3:E7:5D:23
n=254   00:0B:DB:4F:6B:10
n=6     00:80:77:31:01:07
n=3     00:80:C7:8F:6C:96
```

Note:

- Packets clearly traveled from 00:19:E5:E7:5D:23 to 00:0B:DB:4F:6B:10
- Other usage appears to have Ethernet addresses in HTML!

ip.txt: potential ip addresses from packet carving (scan_net) (not from dotted quads)

```
# Filename: /corp/nps/drives/nps-2009-m57-patents-redacted/charlie-2009-12-11.E01
# Feature-Recorder: ip
# Feature-File-Version: 1.1
117942521       20.137.78.24    struct ip R (src) cksum-bad
117942521       94.89.93.194    struct ip L (dst) cksum-bad
118342942       20.137.78.24    struct ip R (src) cksum-bad
118342942       94.89.93.194    struct ip L (dst) cksum-bad

9977306594      192.168.1.1     sockaddr_in
9977393926      63.245.209.93   sockaddr_in

5839793854-HIBER-17952268       90.4.162.232    struct ip L (dst) cksum-bad
5839793854-HIBER-17960460       78.0.3.185      struct ip R (src) cksum-bad
5839793854-HIBER-17960460       90.4.162.232    struct ip L (dst) cksum-bad
6339825268      192.168.1.104   struct ip L (src) cksum-ok
6339825268      192.168.1.1     struct ip R (dst) cksum-ok
6339825320      192.168.1.104   struct ip L (src) cksum-ok

5839793854-HIBER-129985200      8.3.2.3 sockaddr_in
```

- Local (“L”) or Remote (“R”)
- chksum-bad/chksum-ok — IP checksum good or bad
- sockaddr_in — IP address from sockaddr_in structure.

ip_histogram.txt removes random noise (1.3 histogram is only of chksum-ok values)

Histogram of all values:

```
# Filename: /corp/nps/drives/nps-2009-m57-patents-redacted/charlie-2009-12-11.E01
# Feature-Recorder: ip
# Histogram-File-Version: 1.1
n=93    108.5.218.9
n=93    7.90.102.193
n=64    20.137.78.24
n=64    94.89.93.194
n=31    176.69.248.3
n=30    5.225.0.252
n=26    120.23.102.15
n=26    182.210.102.137
n=24    152.6.0.164
n=24    152.6.0.220
n=19    192.168.1.1
n=14    192.168.1.104
n=13    141.77.252.81
n=13    80.4.139.6
```

chksum-ok:

packets.pcap — pcap file made from carved potential packets.

Use any packet analysis tool you like...

```
$ tcpdump -r packets.pcap
-5:-59:-59.0000 IP 192.168.1.1.microsoft-ds > 192.168.1.104.udpradio: Flags [.],
ack 416616880, win 65535, length 0
-5:-59:-59.0000 IP 192.168.1.1.microsoft-ds > 192.168.1.104.udpradio: Flags [.],
ack 4294967234, win 65535, length 0
-5:-59:-59.0000 IP 192.168.1.1.microsoft-ds > 192.168.1.104.udpradio: Flags [.],
ack 4294967084, win 65535, length 0

-5:-59:-59.0000 IP 192.168.1.1.microsoft-ds > 192.168.1.104.udpradio: Flags [P.],
seq 4294966956:4294967060, ack 4294967008, win 65535, length 104SMB PACKET:
SMBtrans2 (REPLY)
...
```

Notice time is -5:-59:-59.000

- My local time zone was -0600
- The time in the packet file is “1”

```
/* Possibly a valid ethernet frame but not preceeded by a pcap_record_header.
 * Write it out with time of 1.
 */
```

- Only packets carved from a PCAP file will have a the correct time.

tcp.txt — Details about TCP (and UDP) network flows

More detail than ip.txt

```
117942521       20.137.78.24:2048 -> 94.89.93.194:32824 (TCP)    Size: 232
118342942       20.137.78.24:2048 -> 94.89.93.194:32824 (TCP)    Size: 232
119672053       255.144.140.1:3972 -> 0.0.133.192:52224 (TCP)    Size: 3973
122908648       1.0.0.0:0 -> 117.17.2.0:0 (UDP)  Size: 512
101356868       56.141.76.36:65490 -> 28.81.139.206:35832 (TCP)  Size: 3972
101727492       56.141.76.36:65490 -> 28.81.139.206:35832 (TCP)  Size: 3972
102361428       56.141.76.36:65490 -> 28.81.139.206:35832 (TCP)  Size: 3972
102380242       20.137.78.24:2048 -> 94.89.93.194:21899 (TCP)    Size: 232
68852207        7.90.102.193:13311 -> 108.5.218.9:18387 (TCP)    Size: 3973
```

Be careful of false positives:

```
336089314-HIBER-100696361       0.0.0.0:101 -> 0.0.0.0:19829 (TCP)       Size: 77
336089314-HIBER-113107975       48.144.141.49:0 -> 176.61.0.0:0 (TCP)    Size: 70
336089314-HIBER-161355043       7.86.252.232:55425 -> 47.0.250.69:21841 (TCP)    Size: 1419
336089314-HIBER-166154373       255.118.14.233:57600 -> 255.164.149.80:52428 (UDP)       Size: 768
336089314-HIBER-166162086       255.118.14.233:57600 -> 255.164.149.80:52428 (UDP)       Size: 768
336089314-HIBER-166169799       255.118.14.233:57600 -> 255.164.149.80:52428 (UDP)       Size: 768
336089314-HIBER-166194316       255.118.14.233:57600 -> 255.164.149.80:52428 (UDP)       Size: 768
336089314-HIBER-166202507       255.118.14.233:57600 -> 255.164.149.80:52428 (UDP)       Size: 768
336089314-HIBER-166210698       255.118.14.233:57600 -> 255.164.149.80:52428 (UDP)       Size: 768
336089314-HIBER-166218889       255.118.14.233:57600 -> 255.164.149.80:52428 (UDP)       Size: 768
336089314-HIBER-166227080       255.118.14.233:57600 -> 255.164.149.80:52428 (UDP)       Size: 768
336089314-HIBER-168358773       57.93.93.93:3968 -> 8.141.88.247:17843 (TCP)     Size: 5631
336089314-HIBER-168361526       57.93.93.93:3968 -> 8.141.88.247:17843 (TCP)     Size: 5631
```

tcp_histogram.txt — would be nice to have total flow info

These packets:

```
101727492       56.141.76.36:65490 -> 28.81.139.206:35832 (TCP)  Size: 3972
102361428       56.141.76.36:65490 -> 28.81.139.206:35832 (TCP)  Size: 3972
```

Become this histogram:

```
n=93    7.90.102.193:13311 -> 108.5.218.9:18387 (TCP)
n=53    0.0.123.55:12288 -> 56.49.57.65:12336 (TCP)
n=48    5.100.228.83:64 -> 15.134.211.0:15 (TCP)
n=38    252.21.212.255:34048 -> 83.0.0.0:17792 (TCP)
n=38    252.21.212.255:34048 -> 83.0.16.16:17792 (TCP)
n=30    104.48.235.16:60160 -> 232.235.16.232:35701 (TCP)
n=30    5.225.0.252:61133 -> 176.69.248.3:63488 (TCP)
n=28    0.106.37.95:23179 -> 102.59.199.117:52968 (TCP)
n=27    20.137.78.24:2048 -> 94.89.93.194:21899 (TCP)
n=26    120.23.102.15:4160 -> 182.210.102.137:16449 (UDP)
n=24    106.0.80.83:51457 -> 141.74.255.139:65382 (UDP)
```

Caveats:

- Still a lot of false positives.
- The current histogram system can’t do math...

There are four main categories of feature files:

Identity Information:

- Domain Names; Email addresses; URLs
- Search terms; Facebook IDs; JSON data
- KML files
- VCARDs
- find output

Technical Info:

- ZIP files; EXIF data

Network Information:

- PCAP files; Ethernet Addresses; TCP/IP Connections; etc.

Information about executables:

- ELF & PE headers; Windows Prefetch files

```
-rw-r--r--@ 1 simsong  staff       476 Jul  7 23:50 aes_keys.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 alerts.txt
-rw-r--r--@ 1 simsong  staff      2743 Jul  7 23:59 ccn.txt
-rw-r--r--@ 1 simsong  staff       454 Jul  8 00:03 ccn_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 ccn_track2.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 ccn_track2_histogram.txt
-rw-r--r--@ 1 simsong  staff  23369167 Jul  8 00:03 domain.txt
-rw-r--r--@ 1 simsong  staff    185266 Jul  8 00:03 domain_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 elf.txt
-rw-r--r--@ 1 simsong  staff   1719842 Jul  8 00:03 email.txt
-rw-r--r--@ 1 simsong  staff     35073 Jul  8 00:03 email_histogram.txt
-rw-r--r--@ 1 simsong  staff     23961 Jul  8 00:00 ether.txt
-rw-r--r--@ 1 simsong  staff       337 Jul  8 00:03 ether_histogram.txt
-rw-r--r--@ 1 simsong  staff  11188830 Jul  8 00:03 exif.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 find.txt
-rw-r--r--@ 1 simsong  staff      1112 Jul  8 00:01 gps.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 hex.txt
-rw-r--r--@ 1 simsong  staff     95835 Jul  8 00:03 ip.txt
-rw-r--r--@ 1 simsong  staff     11603 Jul  8 00:03 ip_histogram.txt
-rw-r--r--@ 1 simsong  staff   2025702 Jul  8 00:03 json.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 kml.txt
-rw-r--r--@ 1 simsong  staff    194991 Jul  8 00:03 packets.pcap
-rw-r--r--@ 1 simsong  staff     21343 Jul  8 00:03 report.xml
-rw-r--r--@ 1 simsong  staff   3782598 Jul  8 00:03 rfc822.txt
-rw-r--r--@ 1 simsong  staff    213746 Jul  8 00:03 tcp.txt
-rw-r--r--@ 1 simsong  staff     61255 Jul  8 00:03 tcp_histogram.txt
-rw-r--r--@ 1 simsong  staff     59469 Jul  8 00:03 telephone.txt
-rw-r--r--@ 1 simsong  staff      6612 Jul  8 00:03 telephone_histogram.txt
-rw-r--r--@ 1 simsong  staff  67205326 Jul  8 00:03 url.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_facebook-id.txt
-rw-r--r--@ 1 simsong  staff   5706665 Jul  8 00:03 url_histogram.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  8 00:03 url_microsoft-live.txt
-rw-r--r--@ 1 simsong  staff      8504 Jul  8 00:03 url_searches.txt
-rw-r--r--@ 1 simsong  staff    151673 Jul  8 00:03 url_services.txt
-rw-r--r--@ 1 simsong  staff         0 Jul  7 23:48 vcard.txt
-rw-r--r--@ 1 simsong  staff  18549729 Jul  8 00:03 windirs.txt
-rw-r--r--@ 1 simsong  staff  29051041 Jul  8 00:03 winpe.txt
-rw-r--r--@ 1 simsong  staff   1984759 Jul  8 00:03 winprefetch.txt
-rw-r--r--@ 1 simsong  staff  34128889 Jul  8 00:03 zip.txt
```

elf.txt records potential ELF executables

charlie-2009-12-11 doesn't have any:

```
-rw-r--r--+   1 simsong  simsong         0 Jul 20 16:54 elf.txt
```

But nps-2009-ubnist1.gen3 does:

```
-rw-r--r--+  1 simsong  staff   5691737 Aug  3 12:39 elf.txt
```

Here is a sample:

```
# Feature-File-Version: 1.1
727114768-GZIP-2048     1b5984e4365278bee12c9be8849439f4        <ELF
class="ELFCLASS32" data="ELFDATA2LSB" osabi="ELFOSABI_NONE" abiversion="0 ><ehdr
type="ET_EXEC" machine="EM_386" version="1" entry="134514864" phoff="52"
shoff="19000" flags="0" ehsize="52" phentsize="32" phnum="8" shentsize="40"
shnum="27" shstrndx="26" /><sections><section name="" type="SHT_NULL" addr="0x0"
offset="0" size="0" link="0" info="0" addralign="0" shentsize="0"><flags></
flags></section><section name=".interp" type="SHT_PROGBITS" addr="0x8048134"
offset="134" size="13" link="0" info="0" addralign="1"
shentsize="0"><flags><SHF_ALLOC /></flags></section><section name=".note.ABI-tag"
type="SHT_NOTE" addr="0x8048148" offset="148" size="20" link="0" info="0"
addralign="4" shentsize="0"><flags><SHF_ALLOC /></flags></section><section
name=".hash" type="SHT_HASH" addr="0x8048168" offset="168" size="c0" link="5"
info="0" addralign="4" shentsize="4"><flags><SHF_ALLOC /></flags></
section><section name=".gnu.hash" type="SHT_GNU_HASH" addr="0x8048228" offset\
```

Decoding the <ELF> record...

The path indicates that the ELF is inside a GZIP stream:

```
# Feature-File-Version: 1.1
727114768-GZIP-2048     ...
```

The MD5 is the hash of the first 4KiB:

```
1b5984e4365278bee12c9be8849439f4
```

Next comes the XML for the header:

```
<ELF class="ELFCLASS32" data="ELFDATA2LSB" osabi="ELFOSABI_NONE" abiversion="0 >
<ehdr type="ET_EXEC" machine="EM_386" version="1" entry="134514864" phoff="52"
shoff="19000" flags="0" ehsize="52" phentsize="32" phnum="8" shentsize="40"
shnum="27" shstrndx="26" />

<sections>
  <section name="" type="SHT_NULL" addr="0x0" offset="0" size="0" link="0"
info="0" addralign="0" shentsize="0">
  <flags></flags>
  </section>
...
</sections>
<shared_objects><so>libc.so.6</so></shared_objects>
</ELF>
```

winpe.txt — Potential Windows executables


```
117886464       0316eaac06e782616036639824c04ceb        <PE>
  <FileHeader Machine="IMAGE_FILE_MACHINE_I386" NumberOfSections="5" TimeDateStamp="1255540604"
PointerToSymbolTable="0" NumberOfSymbols="0" SizeOfOptionalHeader="224">
<Characteristics><IMAGE_FILE_EXECUTABLE_IMAGE /><IMAGE_FILE_32BIT_MACHINE /><IMAGE_FILE_DLL /></
Characteristics></FileHeader><OptionalHeaderStandard Magic="PE32" MajorLinkerVersion="8"
MinorLinkerVersion="0" SizeOfCode="260096" SizeOfInitializedData="89088"
SizeOfUninitializedData="0" AddressOfEntryPoint="0x3963c" BaseOfCode="0x1000" /
><OptionalHeaderWindows ImageBase="0x6a520000" SectionAlignment="1000" FileAlignment="200"
MajorOperatingSystemVersion="4" MinorOperatingSystemVersion="0" MajorImageVersion="0"
MinorImageVersion="0" MajorSubsystemVersion="4" MinorSubsystemVersion="0" Win32VersionValue="0"
SizeOfImage="59000" SizeOfHeaders="400" CheckSum="0x5aedb" SubSystem=""
SizeOfStackReserve="100000" SizeOfStackCommit="1000" SizeOfHeapReserve="100000"
SizeOfHeapCommit="1000" LoaderFlags="0" NumberOfRvaAndSizes="10"><DllCharacteristics></
DllCharacteristics></OptionalHeaderWindows><Sections><SectionHeader Name=".text"
VirtualSize="3f73a" VirtualAddress="1000" SizeOfRawData="3f800" PointerToRawData="400"
PointerToRelocations="0" PointerToLinenumbers="0" ><Characteristics><IMAGE_SCN_CNT_CODE /
><IMAGE_SCN_MEM_EXECUTE /><IMAGE_SCN_MEM_READ /></Characteristics></SectionHeader><SectionHeader
Name=".rdata" VirtualSize="df22" VirtualAddress="41000" SizeOfRawData="e000"
PointerToRawData="3fc00" PointerToRelocations="0" PointerToLinenumbers="0"
><Characteristics><IMAGE_SCN_CNT_INITIALIZED_DATA /><IMAGE_SCN_MEM_READ /></Characteristics></
SectionHeader><SectionHeader Name=".data" VirtualSize="1128" VirtualAddress="4f000"
SizeOfRawData="a00" PointerToRawData="4dc00" PointerToRelocations="0" PointerToLinenumbers="0"
><Characteristics><IMAGE_SCN_CNT_INITIALIZED_DATA /><IMAGE_SCN_MEM_READ /><IMAGE_SCN_MEM_WRITE /
></Characteristics></SectionHeader><SectionHeader Name=".rsrc" VirtualSize="848"
VirtualAddress="51000" SizeOfRawData="a00" PointerToRawData="4e600" PointerToRelocations="0"
PointerToLinenumbers="0" ><Characteristics><IMAGE_SCN_CNT_INITIALIZED_DATA /><IMAGE_SCN_MEM_READ /
></Characteristics></SectionHeader><SectionHeader Name=".reloc" VirtualSize="672c"
VirtualAddress="52000" SizeOfRawData="6800" PointerToRawData="4f000" PointerToRelocations="0"
PointerToLinenumbers="0" ><Characteristics><IMAGE_SCN_CNT_INITIALIZED_DATA /
><IMAGE_SCN_MEM_DISCARDABLE /><IMAGE_SCN_MEM_READ /></Characteristics></SectionHeader></
Sections><dlls><dll>ADVAPI32.dll\x00\x006\x05memcpy\x00\x008\x05memmov</dll><dll>WS2_32.dll
\x00\x00,\x02InterlockedIncreme</dll></dlls></PE>
```


winpe.txt — Potential Windows executables
First line is the offset, MD5(first 4K), XML of data

```
117886464       0316eaac06e782616036639824c04ceb        <PE>
  <FileHeader Machine=...
```

Uses:

- Offset tells you were the find the file (most executables are not fragmented)
- MD5 can be used to deduplicate and look up in hash database
- <PE> XML block breaks out all of the PE headers.

<PE> <FileHeader> provides information on header

```
<?xml version="1.0"?>
<PE>
  <FileHeader
      Machine="IMAGE_FILE_MACHINE_I386"
      NumberOfSections="5"
      TimeDateStamp="1255540604"
      PointerToSymbolTable="0"
      NumberOfSymbols="0"
      SizeOfOptionalHeader="224"
    >
  <Characteristics>
      <IMAGE_FILE_EXECUTABLE_IMAGE/>
      <IMAGE_FILE_32BIT_MACHINE/>
      <IMAGE_FILE_DLL/>
  </Characteristics>
</FileHeader
```

<PE><OptionalHeaderStandard>

```
<OptionalHeaderStandard
      Magic="PE32"
      MajorLinkerVersion="8"
      MinorLinkerVersion="0"
      SizeOfCode="260096"
      SizeOfInitializedData="89088"
      SizeOfUninitializedData="0"
      AddressOfEntryPoint="0x3963c"
      BaseOfCode="0x1000"/>
```

<PE> <OptionalHeaderWindows>

```
<OptionalHeaderWindows
      ImageBase="0x6a520000"
      SectionAlignment="1000"
      FileAlignment="200"
      MajorOperatingSystemVersion="4"
      MinorOperatingSystemVersion="0"
      MajorImageVersion="0"
      MinorImageVersion="0"
      MajorSubsystemVersion="4"
      MinorSubsystemVersion="0"
      Win32VersionValue="0"
      SizeOfImage="59000"
      SizeOfHeaders="400"
      CheckSum="0x5aedb"
      SubSystem=""
      SizeOfStackReserve="100000"
      SizeOfStackCommit="1000"
      SizeOfHeapReserve="100000"
      SizeOfHeapCommit="1000"
      LoaderFlags="0"
      NumberOfRvaAndSizes="10">
   <DllCharacteristics/>
</OptionalHeaderWindows>
```

<PE><Sections><SectionHeader>
Provides details of each PE section

```
<Sections>
  <SectionHeader
        Name=".text"
        VirtualSize="3f73a"
        VirtualAddress="1000"
        SizeOfRawData="3f800"
        PointerToRawData="400"
        PointerToRelocations="0"
        PointerToLinenumbers="0">
    <Characteristics>
      <IMAGE_SCN_CNT_CODE/>
      <IMAGE_SCN_MEM_EXECUTE/>
      <IMAGE_SCN_MEM_READ/>
    </Characteristics>
</SectionHeader>
```

```
<SectionHeader
        Name=".rdata"
        VirtualSize="df22"
        VirtualAddress="41000"
        SizeOfRawData="e000"
        PointerToRawData="3fc00"
        PointerToRelocations="0"
        PointerToLinenumbers="0">
<Characteristics>
 <IMAGE_SCN_CNT_INITIALIZED_DATA/>
 <IMAGE_SCN_MEM_READ/>
</Characteristics>
</SectionHeader>
```

winprefetch.txt - bulk_extractor will carve prefetch files! Useful because PREFETCH files are frequently deleted

Prefetch files give you:

- *Name of executable*
- *Name of DLLs*
- *atime*
- *Number of runs*
- *Serial number*
- *Directory of DLLs*
- *ctime*

```
62123520        WMIPRVSE.EXE    <prefetch><os>Windows XP</os>

<filename>WMIPRVSE.EXE</filename>
<header_size>152</header_size>
<atime>2009-12-11T15:31:12Z</atime>
<runs>251</runs>
<filenames>
<file>\x5CDEVICE\x5CHARDDISKVOLUME1\x5CWINDOWS\x5CSYSTEM32\x5CNTDLL.DLL</file>
<file>\x5CDEVICE\x5CHARDDISKVOLUME1\x5CWINDOWS\x5CSYSTEM32\x5CKERNEL32.DLL</file>
<file>\x5CDEVICE\x5CHARDDISKVOLUME1\x5CWINDOWS\x5CSYSTEM32\x5CUNICODE.NLS</file>
<file>\x5CDEVICE\x5CHARDDISKVOLUME1\x5CWINDOWS\x5CSYSTEM32\x5CLOCALE.NLS</file>
...
</filenames>
<volume><path>\x5CDEVICE\x5CHARDDISKVOLUME1</path>
<creation>2009-11-08T16:58:56Z</creation>
<serial_number>d8cc759a</serial_number>
<dirnames><dir>\x5CDEVICE\x5CHARDDISKVOLUME1\x5C</dir>
<dir>\x5CDEVICE\x5CHARDDISKVOLUME1\x5CWINDOWS\x5C</dir>
```

False positives must be expected.

When TB of data are scanned, there will be false positives.

```
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 1.5.0-alpha8 ($Rev: 10844 $)
# Feature-Recorder: ip
# Filename: /home/slgarfin/corp/nus/drives/HU/HU001-0001/HU001-0001.E01
# Feature-File-Version: 1.1
1818108534      3f2a:1136:3f2a:45a1:443:f05e:800:b70    struct ip6_hdr R (src) cksum-ok
1818108534      3014:d400:54f2:d82:f008:352:647:8       struct ip6_hdr L (dst) cksum-ok
19369431508     2c0e:2a2c:c675:4938:5d1:6766:880b:bc40  struct ip6_hdr R (src) cksum-ok
19369431508     2240:b842:7180:2bd1:347e:37a1:28be:802f struct ip6_hdr L (dst) cksum-ok
20790114094     28c0:d363:e9d3:e9a5:a485:69b3:3228:24aa struct ip6_hdr R (src) cksum-ok
20790114094     2eba:75b2:a575:7a9d:482e:e667:56cb:44b2 struct ip6_hdr L (dst) cksum-ok
```

These IPv6 addresses are from packets with valid checksums.

- But the IPv6 checksum is just 32 bits, so there is a 1-in-4-billion chance of a false positive.

These are probably invalid SSNs:

```
48857564047     SSN759057878    ==|\x98b\x04M\x81=Q9R4187SSN759057878QP12OQOQPO977b3V
48858284573     SSN759057878    P4O<\xF1%\x03M\x02Q9R4187SSN759057878QP12OQOQPO977\x0Cb%
```

Objects with internal binary structures will rarely be in error.

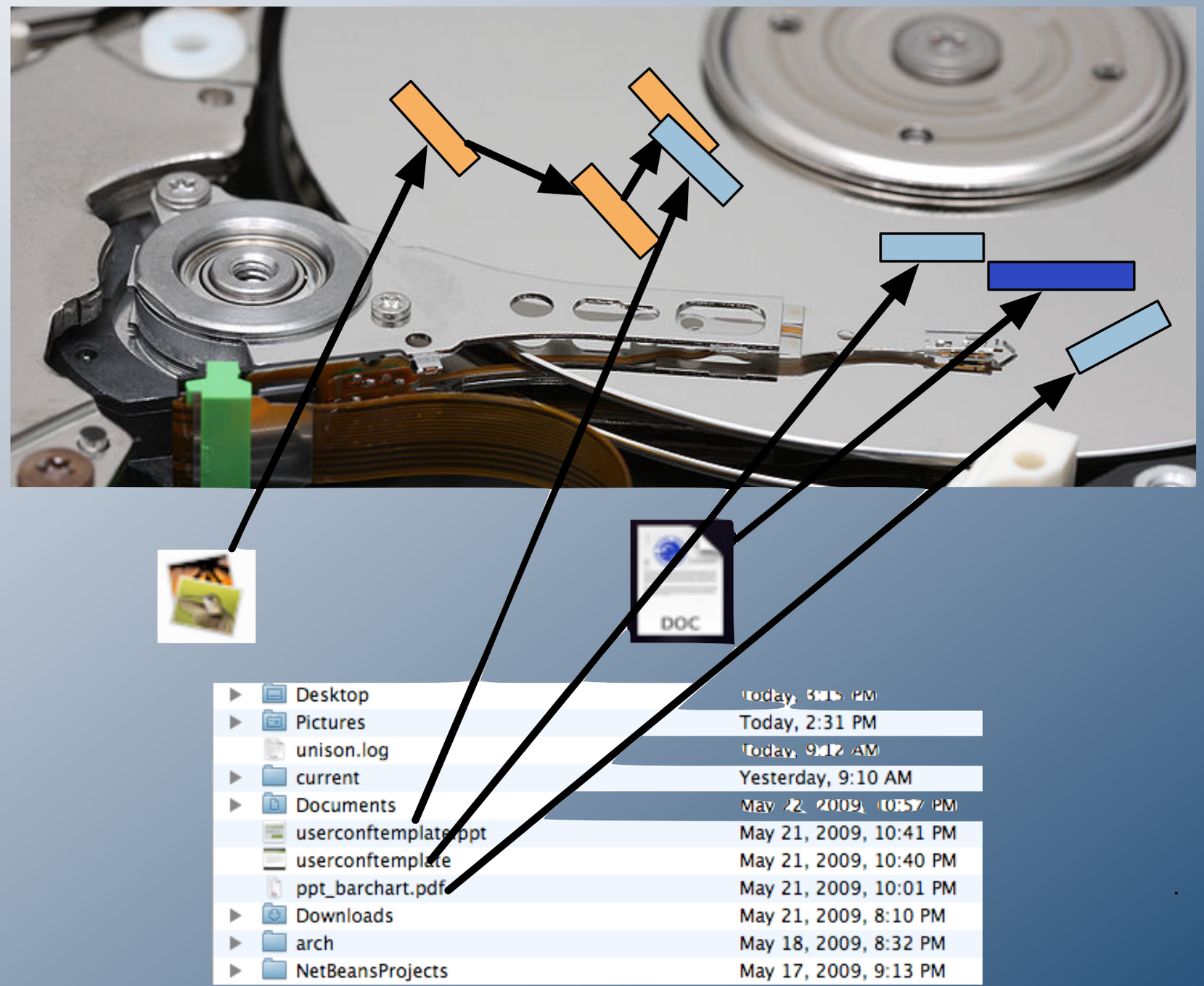
- Windows PE files, LNK files, JPEGs, etc.

scan_sqlite — Finds all SQLite3 headers; carves database

Only works with contiguous databases

For example, in testing on the drive AE10-1156, 12 sqlite databases were carved, 7 of which had recoverable schemas, and 2 of which had recoverable data:

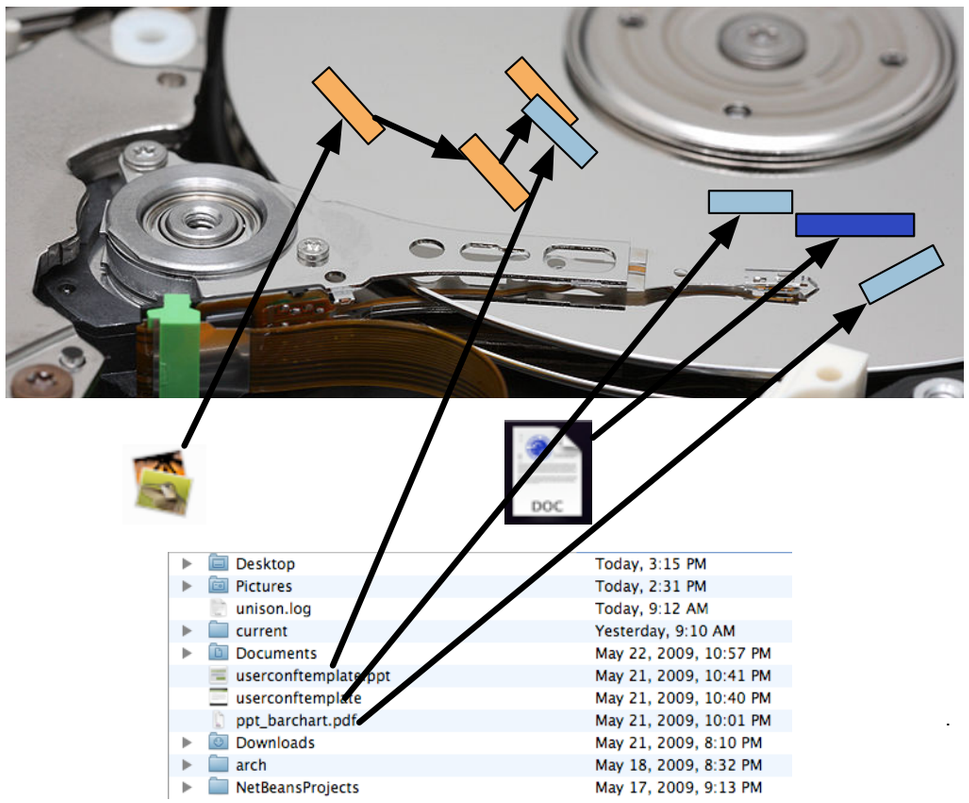
name	size	contents
33554432.sqlite3	9,269,760	moz_downloads (empty)
402653184.sqlite3	9,008,640	moz_hosts (empty)
452984832.sqlite3	6,022,656	engine_data (empty)
469762048.sqlite3	8,785,408	moz_cookies (7 cookies)
486539264.sqlite3	10,107,392	moz_logins (empty)
570425344.sqlite3	4,379,136	moz groups, prefs, settings (empty)
587202560.sqlite3	8,445,440	moz bookmarks, history, keywords, etc. (98 bookmarks
822083584.sqlite3	3,988,992	moz classifier, subs and tables _classifer,
1426063360.sqlite3	8,258,804	(nothing recoverable)
2684354560.sqlite3	12,385,876	(nothing recoverable)
2717908992.sqlite3	178,688	(nothing recoverable)
2936012800.sqlite3	16,983,284	(nothing recoverable)



Finding File Names

identify_filenames.py: Determines the file name for each feature.

bulk_extractor reports the *offset in the disk image* for each feature.



To get the file names, you need to map the disk block to a file.

- Make a map of the blocks in DFXML with **fiwalk** (https://github.com/kfairbanks/sleuthkit)
  - *Soon to be integrated into SleuthKit*
- Then use **python/identify_filenames.py** to create an *annotated feature file.*

identify_filenames correlation the feature file and the DFXML file!

python/identify_filenames.py

```
$ python3.2 python/identify_filenames.py --help
usage: identify_filenames.py [-h] [--all] [--featurefiles FEATUREFILES]
                             [--imagefile IMAGEFILE] [--xmlfile XMLFILE]
                             [--list] [-t] [-v] [--verbose] [--debug]
                             bulk_extractor_output outdir

Identify filenames from "bulk_extractor" output

positional arguments:
  bulk_extractor_output
                        Directory or ZIP file of bulk_extractor output
  outdir                Output directory; must not exist

optional arguments:
  -h, --help            show this help message and exit
  --all                 Process all feature files
  --featurefiles FEATUREFILES
                        Specific feature file to process; separate with commas
  --imagefile IMAGEFILE
                        Overwrite location of image file from bulk_extractor
                        output
  --xmlfile XMLFILE     Don't run fiwalk; use the provided XML file instead
  --list                List feature files in bulk_extractor_output and exit
  -t                    Terse output
  -v                    Print Version and exit
  --verbose             Verbose mode
  --debug               Debug mode
```

identify_filenames.py tries to use the information in the report.xml file to make operation automatic.

report.xml is a DFXML file that contains:

- Disk image that was processed
- Location of feature files

identify_filenames can work with:

- bulk_extractor output file
- a ZIP of a bulk_extractor output file
- disk image *or* DFXML of disk image

identify_filenames will run fiwalk if...

- no XML file is provided
- fiwalk is in the path
- But it's faster to provide the XML file!

```
$ python3.2 identify_filenames.py --list
charlie-2009-12-11.zip
Feature files in /Users/simsong/charlie-2009-12-11.zip:
ccn.txt
exif.txt
url.txt
url_searches.txt
url_services.txt
ether.txt
domain.txt
windirs.txt
email.txt
ip.txt
aes_keys.txt
zip.txt
rfc822.txt
json.txt
tcp.txt
winpe.txt
gps.txt
winprefetch.txt
telephone.txt
$
```

The DFXML file has to be re-read for each feature file.

identify_filenames typically takes hours to run.

Time is proportional to (# of features) * (# of file fragments)

```
$ python3.2 python/identify_filenames.py ~/charlie-2009-12-10.zip
charlie-2009-12-10-id2 --xmlfile charlie-2009-12-10.xml --all
Adding features from aes_keys.txt
Using XML file /corp/nps/scenarios/2009-m57-patents/drives_dfxml/
charlie-2009-12-10.xml
Processed 1000 fileobjects in DFXML file
...
Processed 39000 fileobjects in DFXML file
Processed 40000 fileobjects in DFXML file
Generating output...
real      10298.68
user      10286.50
sys           8.25
$
```

Roughly 3 hours for a 60GB disk image.

Output is “annotated” feature files.

```
$ ls -l
total 166088
-rw-r--r--+ 1 simsong  simsong       511 Aug  4 18:04 annotated_aes_keys.txt
-rw-r--r--+ 1 simsong  simsong      3511 Aug  4 15:39 annotated_ccn.txt
-rw-r--r--+ 1 simsong  simsong  24986176 Aug  4 17:53 annotated_domain.txt
-rw-r--r--+ 1 simsong  simsong   1882453 Aug  4 18:03 annotated_email.txt
-rw-r--r--+ 1 simsong  simsong     24451 Aug  4 16:48 annotated_ether.txt
-rw-r--r--+ 1 simsong  simsong  11208045 Aug  4 15:39 annotated_exif.txt
-rw-r--r--+ 1 simsong  simsong    125580 Aug  4 18:03 annotated_ip.txt
-rw-r--r--+ 1 simsong  simsong   3465286 Aug  4 21:40 annotated_json.txt
-rw-r--r--+ 1 simsong  simsong   3823218 Aug  4 18:26 annotated_rfc822.txt
-rw-r--r--+ 1 simsong  simsong    268678 Aug  4 21:41 annotated_tcp.txt
-rw-r--r--+ 1 simsong  simsong     79345 Aug  4 21:42 annotated_telephone.txt
-rw-r--r--+ 1 simsong  simsong  69150534 Aug  4 16:48 annotated_url.txt
-rw-r--r--+ 1 simsong  simsong  18776356 Aug  4 18:00 annotated_windirs.txt
-rw-r--r--+ 1 simsong  simsong   1944968 Aug  4 22:15 annotated_winprefetch.txt
-rw-r--r--+ 1 simsong  simsong  34263928 Aug  4 18:20 annotated_zip.txt
$
```

Added column 4: Filename from original drive
Added column 5: File's MD5

```
# Position      Feature Context Filename        File MD5
...
7277995794      485799453098756         ible-price/
&rnd=485799453098756\x00request-method\x00  Documents and Settings/Charlie/Local
Settings/Application Data/Mozilla/Firefox/Profiles/2usvf7i1.default/Cache/
_CACHE_001_      eca068c08645e300edd7530362d80a97
```

- position: 7277995794
- Feature: 485799453098756
- Context: ible-price/&rnd=485799453098756\x00request-method\x00
- Filename: Documents and Settings/Charlie/Local Settings/Application Data/Mozilla/Firefox/Profiles/2usvf7i1.default/Cache/_CACHE_001_
- File MD5: eca068c08645e300edd7530362d80a97

```
3598712863-ZIP-100622   michael.buettner@sun.com        chael Büttner
<michael.buettner@sun.com>\x0A   -   Philipp      Documents and Settings/Charlie/
My Documents/Downloads/lightning-0.9-tb-win.xpi  70eebfacfe1227e50db99556cf98161e
```

- position: 3598712863-ZIP-100622
- Feature: michael.buettner@sun.com
- Context: chael Büttner <michael.buettner@sun.com>\x0A - Philip
- Filename: Documents and Settings/Charlie/My Documents/Downloads/lightning-0.9-tb-win.xpi
- File MD5: 70eebfacfe1227e50db99556cf98161e

Getting more information

Bulk_Extractor:

- Programmer's Manual
- User's Manual
- https://github.com/simsong/bulk_extractor/wiki/Documentation
- http://digitalcorpora.org/downloads/bulk_extractor/

See also:

- https://github.com/simsong/bulk_extractor/wiki/Documentation