Automated Mapping of Large Binary Objects Using Primitive Fragment Type Classification

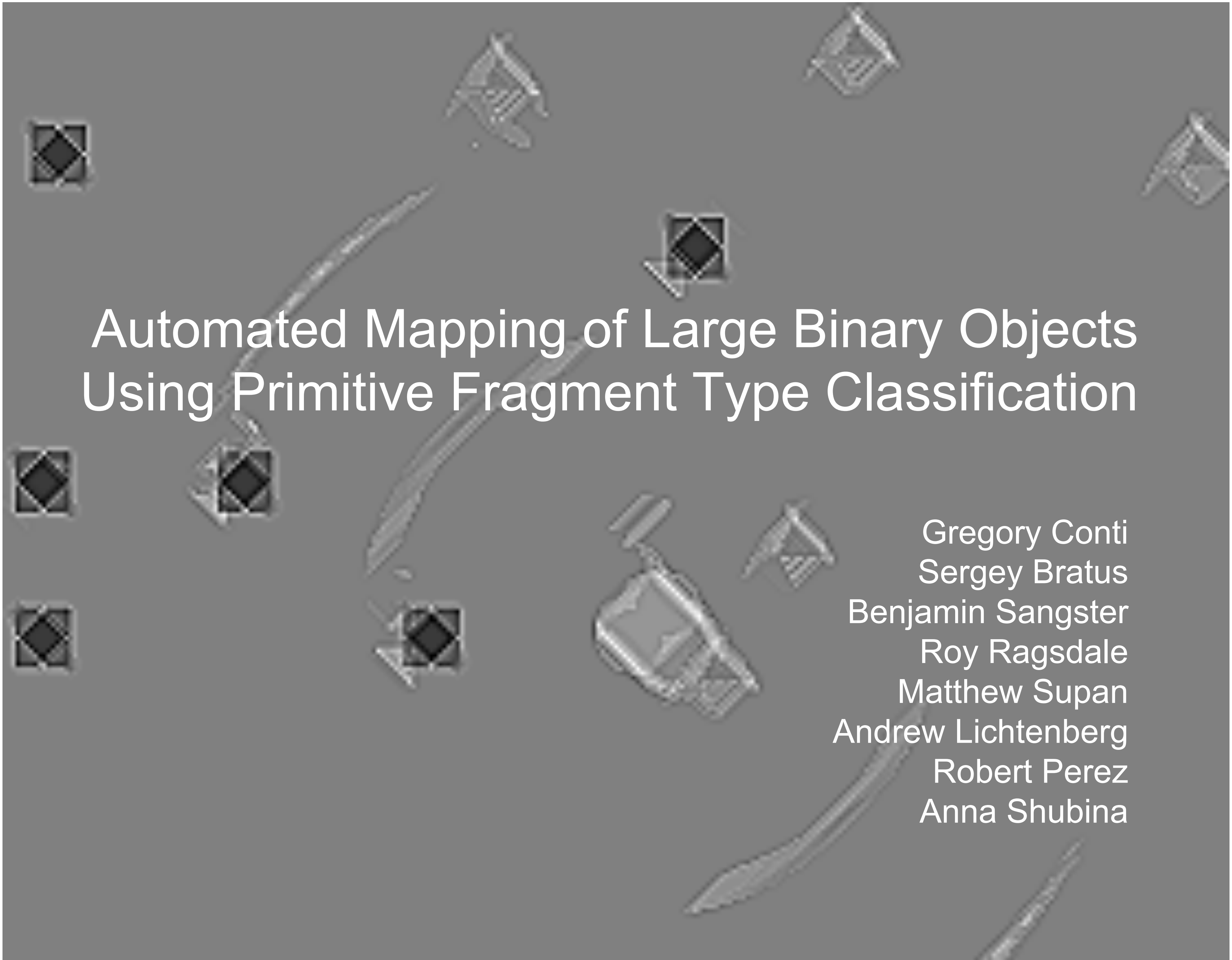
By

Gregory Conti, Sergey Bratus, Benjamin Sangster, Roy Ragsdale, Matthew Supan, Andrew Lichtenberg, Robert Perez-Alemany and Anna Shubina

Presented At

The Digital Forensic Research Conference

DFRWS 2010 USA Portland, OR (Aug 2nd - 4th)

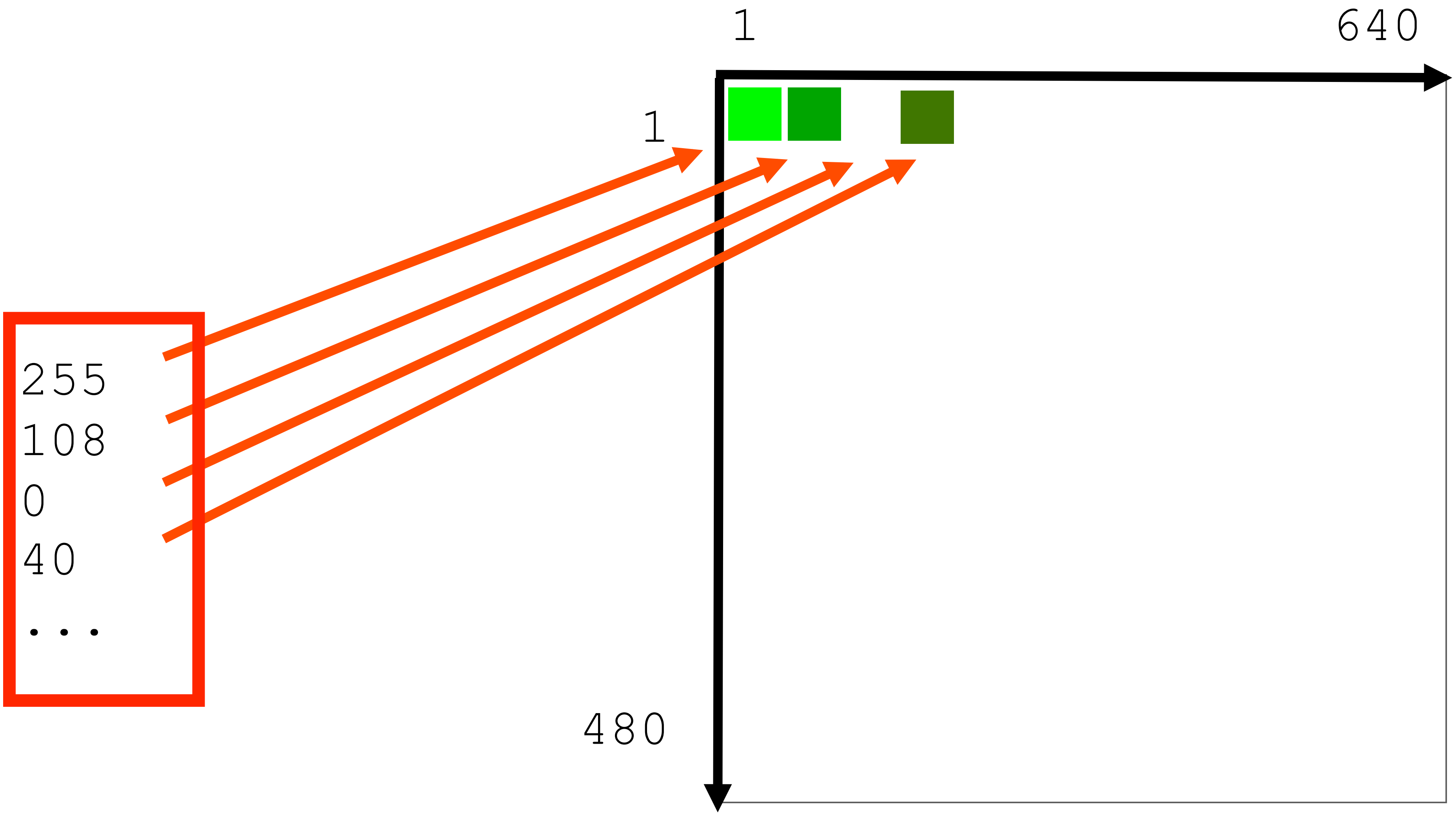


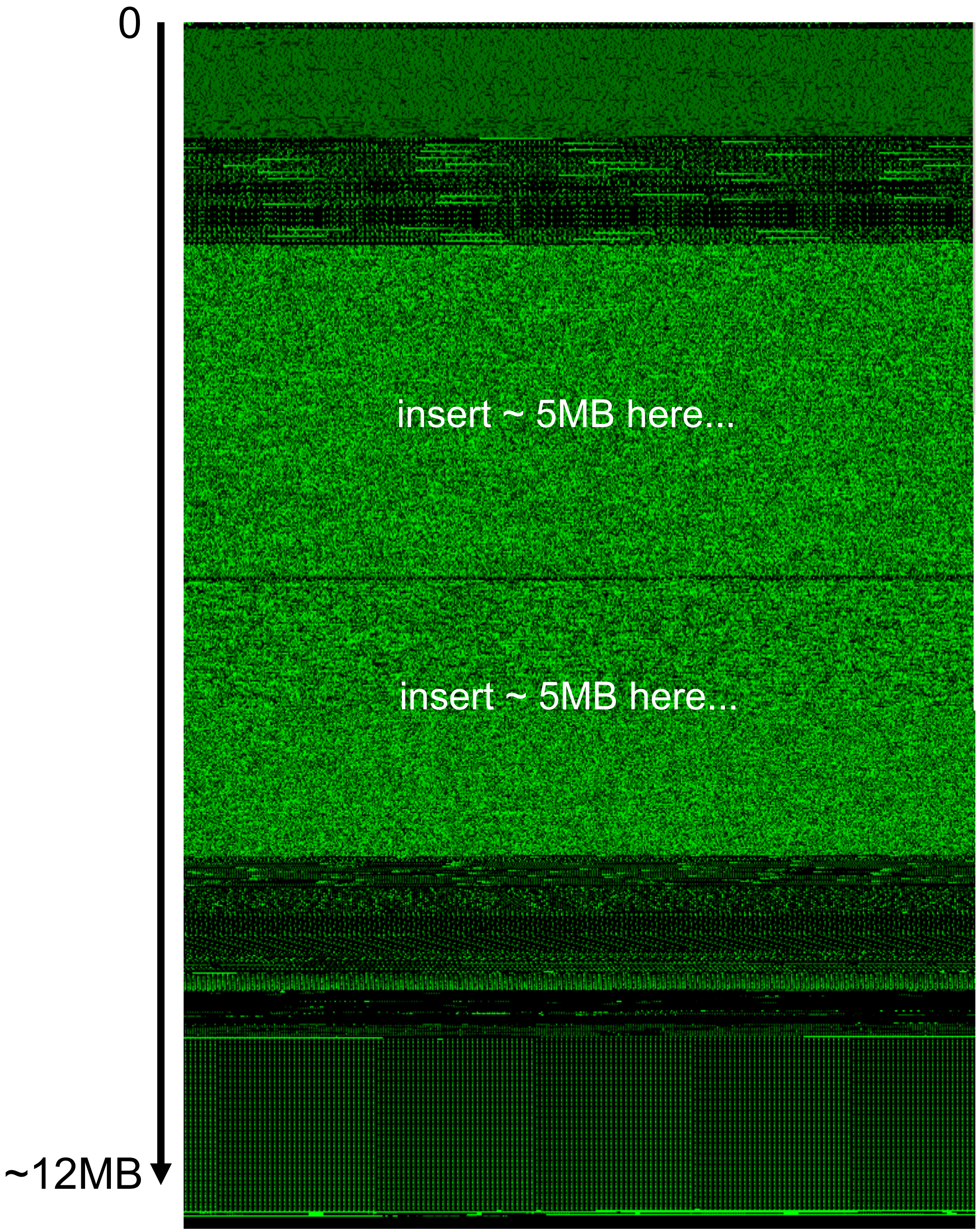
Automated Mapping of Large Binary Objects Using Primitive Fragment Type Classification

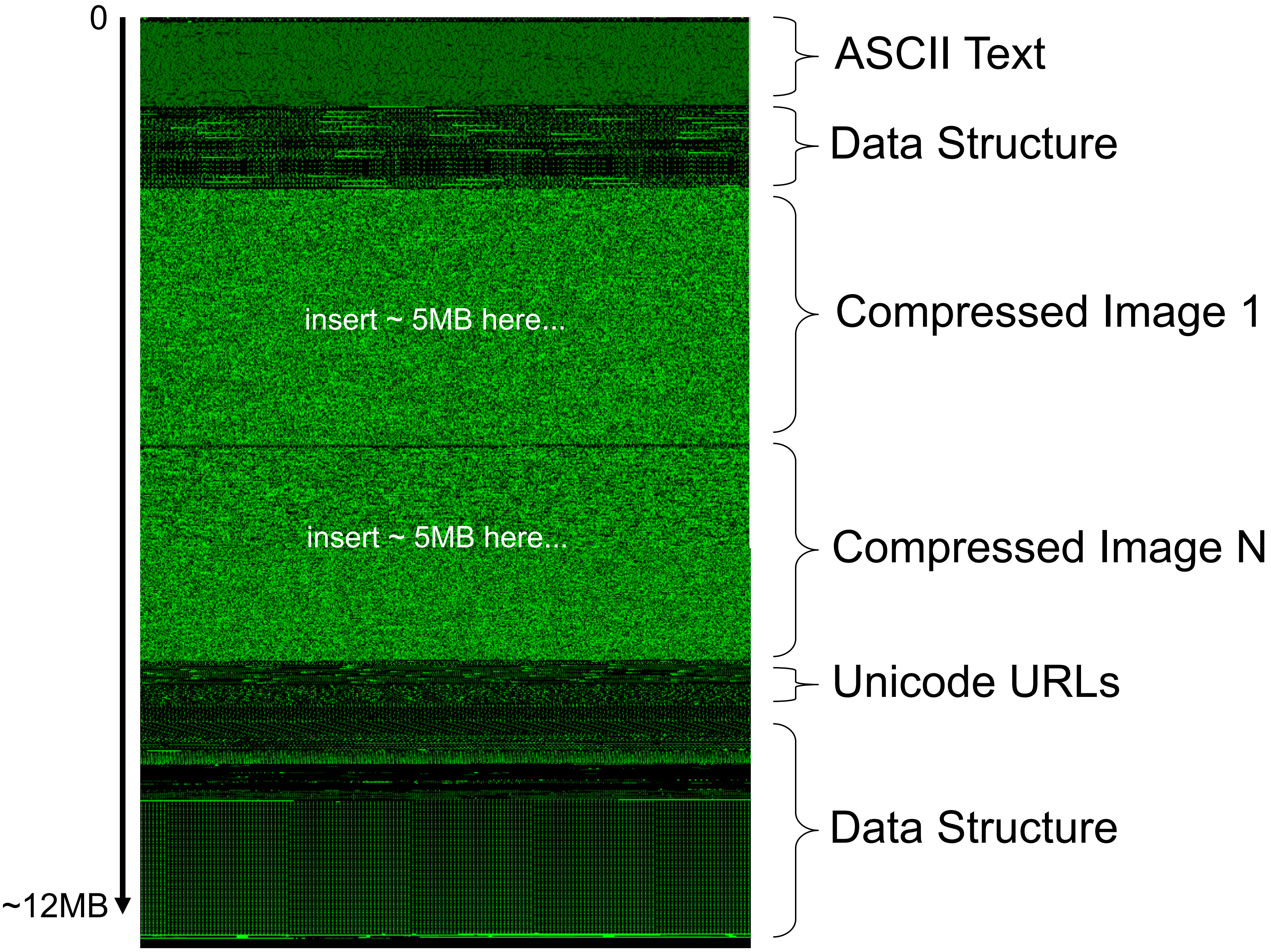
Gregory Conti
Sergey Bratus
Benjamin Sangster
Roy Ragsdale
Matthew Supan
Andrew Lichtenberg
Robert Perez
Anna Shubina

The views expressed in this presentation are those of the author and do not reflect the official policy or position of the United States Military Academy, the Department of the Army, the Department of Defense or the U.S. Government.

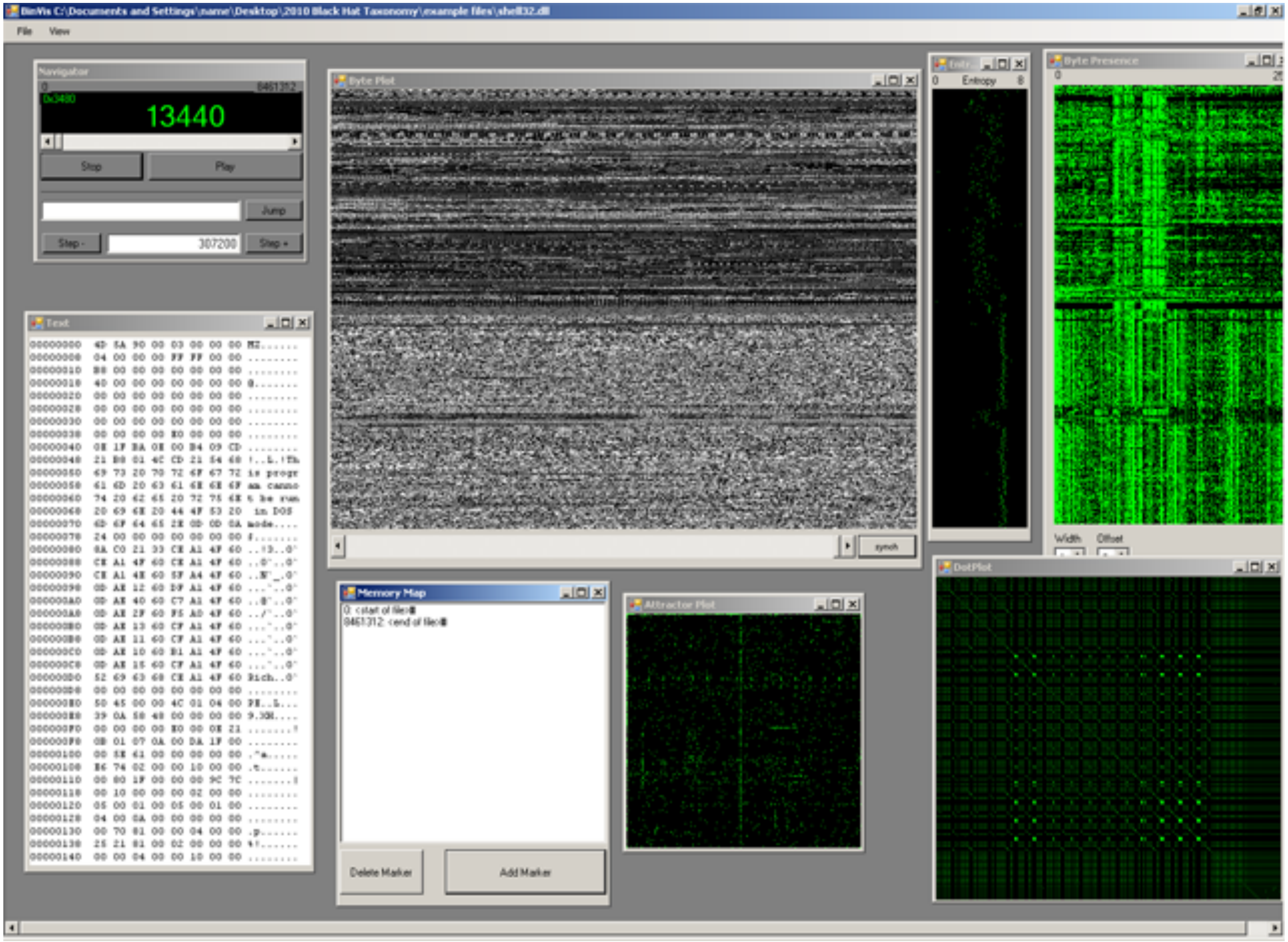
Byte Plot




0
insert ~ 5MB here...
insert ~ 5MB here...
~12MB


0
~12MB
insert ~ 5MB here...
insert ~ 5MB here...
ASCII Text
Data Structure
Compressed Image 1
Compressed Image N
Unicode URLs
Data Structure

What was the Motivation?



```
00000000  4D 5A 90 00 03 00 00 00 MZ......
00000008  04 00 00 00 FF FF 00 00 ........
00000010  B8 00 00 00 00 00 00 00 ........
00000018  40 00 00 00 00 00 00 00 @.......
00000020  00 00 00 00 00 00 00 00 ........
00000028  00 00 00 00 00 00 00 00 ........
00000030  00 00 00 00 00 00 00 00 ........
00000038  00 00 00 00 E0 00 00 00 ........
00000040  0E 1F BA 0E 00 B4 09 CD ........
00000048  21 B8 01 4C CD 21 54 68 !..L.!Th
00000050  69 73 20 70 72 6F 67 72 is progr
00000058  61 6D 20 63 61 6E 6E 6F am canno
00000060  74 20 62 65 20 72 75 6E t be run
00000068  20 69 6E 20 44 4F 53 20  in DOS
00000070  6D 6F 64 65 2E 0D 0D 0A mode....
00000078  24 00 00 00 00 00 00 00 $.......
00000080  8A C0 21 33 CE A1 4F 60 ..!3..O`
00000088  CE A1 4F 60 CE A1 4F 60 ..O`..O`
00000090  CE A1 4E 60 5F A4 4F 60 ..N`_.O`
00000098  0D AE 12 60 DF A1 4F 60 ...`..O`
000000A0  0D AE 40 60 C7 A1 4F 60 ..@`..O`
000000A8  0D AE 2F 60 F5 A0 4F 60 ../`..O`
000000B0  0D AE 13 60 CF A1 4F 60 ...`..O`
000000B8  0D AE 11 60 CF A1 4F 60 ...`..O`
000000C0  0D AE 10 60 B1 A1 4F 60 ...`..O`
000000C8  0D AE 15 60 CF A1 4F 60 ...`..O`
000000D0  52 69 63 68 CE A1 4F 60 Rich..O`
000000D8  00 00 00 00 00 00 00 00 ........
000000E0  50 45 00 00 4C 01 04 00 PE..L...
000000E8  39 0A 58 48 00 00 00 00 9.XH....
000000F0  00 00 00 00 E0 00 0E 21 .......!
000000F8  0B 01 07 0A 00 DA 1F 00 ........
00000100  00 5E 61 00 00 00 00 00 .^a.....
00000108  E6 74 02 00 00 10 00 00 .t......
00000110  00 80 1F 00 00 00 9C 7C .......|
00000118  00 10 00 00 00 02 00 00 ........
00000120  05 00 01 00 05 00 01 00 ........
00000128  04 00 0A 00 00 00 00 00 ........
00000130  00 70 81 00 00 04 00 00 .p......
00000138  25 21 81 00 02 00 00 00 %!......
00000140  00 00 04 00 00 10 00 00 ........
```

```
0: <start of file>
8461312: <end of file>
```

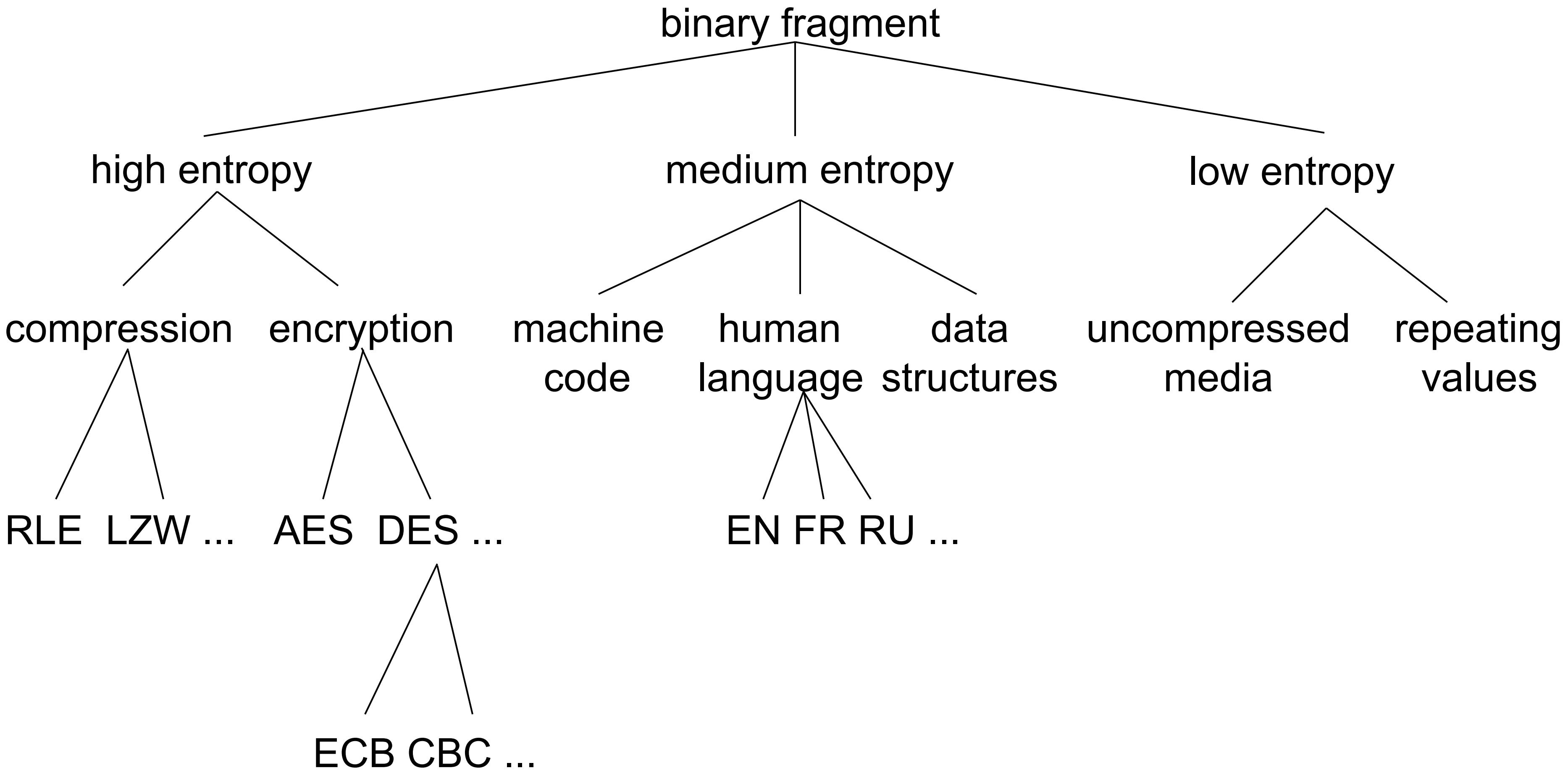
Why?

- Facilitate deep understanding
- Reversing
- Fuzzing
- Memory forensics
- File carving
- Interactive filtering

What is a “Primitive Type?”

{int, long, char, string …} **< Primitive Type** < {.doc, .jar, .exe …}

Example Hierarchy of Primitive Types



binary fragment

high entropy

medium entropy

low entropy

compression

encryption

machine code

human language

data structures

uncompressed media

repeating values

RLE LZW ...

AES DES ...

EN FR RU ...

ECB CBC ...

A Bit of History…

0400-07FF	1024-2047	Screen memory
0800-9FFF	2048-40959	Basic ROM memory
8000-9FFF	32758-40959	Alternate: Rom plug-in area
A000-BFFF	40960-49151	ROM : Basic
A000-BFFF	49060-59151	Alternate: RAM
C000-CFFF	49152-53247	RAM memory, including alternate
D000-D02E	53248-53294	Video Chip (6566)
D400-D41C	54272-54300	Sound Chip (6581 SID)
D800-DBFF	55296-56319	Color nybble memory
DC00-DC0F	56320-56335	Interface chip 1, IRQ (6526 CIA)
DD00-DD0F	56576-56591	Interface chip 2, NMI (6526 CIA)
D000-DFFF	53248-53294	Alternate: Character set
E000-FFFF	57344-65535	ROM: Operating System
E000-FFFF	57344-65535	Alternate : RAM
FF81-FFF5	65409-65525	Jump Table

Goal

0400-07FF	1024-2047	ASCII Text (English)
0800-9FFF	2048-40959	Pointer Table
8000-9FFF	32758-40959	Variable Length Array
A000-BFFF	40960-49151	Compressed Data
A000-BFFF	49060-59151	Unicode (Basic Latin)
C000-CFFF	49152-53247	Unknown Region
D000-D02E	53248-53294	Repeating Value (0xFF)
D400-D41C	54272-54300	Encrypted Region (AES)
D800-DBFF	55296-56319	PNG Image
DC00-DC0F	56320-56335	JavaScript
DD00-DD0F	56576-56591	Encrypted Region (RSA Key?)
D000-DFFF	53248-53294	Unknown Region
E000-FFFF	57344-65535	BMP Image
E000-FFFF	57344-65535	Unicode (Hyperlinks?)
FF81-FFF5	65409-65525	Repeating Value (0x00)

Statistical Tests

- Shannon Entropy

$$H(X) = -\sum_{i=0}^{n-1} p(x_i) \log_b p(x_i)$$

- Arithmetic Mean

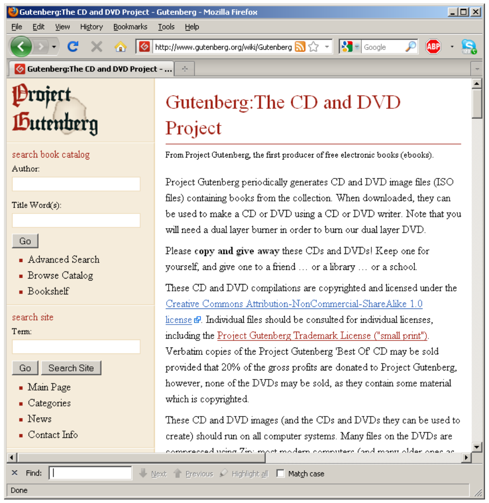
- Chi Square

$$X^2 = \sum_{i=0}^{n-1} \frac{(\text{observed} - \text{expected})^2}{\text{expected}}$$

- Hamming Weight

Corpus Creation

- random
- text
- encrypt
  - AES256/text
- compress
  - bzip2/text
  - compress/text
  - deflate/png
  - LZW/gif
  - mpeg/audio
  - jpeg/image
- encoded
  - base64/zip
  - uuencoded/zip
- machine code
  - linux elf/.text
  - windows PE/.text
- bitmap



Gutenberg:The CD and DVD Project

From Project Gutenberg, the first producer of free electronic books (ebooks).

Project Gutenberg periodically generates CD and DVD image files (ISO files) containing books from the collection. When downloaded, they can be used to make a CD or DVD using a CD or DVD writer. Note that you will need a dual layer burner in order to burn our dual layer DVD.

Please **copy and give away** these CDs and DVDs! Keep one for yourself, and give one to a friend … or a library … or a school.

These CD and DVD compilations are copyrighted and licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 1.0 license. Individual files should be consulted for individual licenses, including the Project Gutenberg Trademark License ("small print"). Verbatim copies of the Project Gutenberg 'Best Of' CD may be sold provided that 20% of the gross profits are donated to Project Gutenberg, however, none of the DVDs may be sold, as they contain some material which is copyrighted.

These CD and DVD images (and the CDs and DVDs they can be used to create) should run on all computer systems. Many files on the DVDs are

What's this fuss about *true* randomness?

Perhaps you have wondered how predictable machines like computers can generate randomness. In reality, most random numbers used in computer programs are *pseudo-random*, which means they are a generated in a predictable fashion using a mathematical formula. This is fine for many purposes, but it may not be random in the way you expect if you're used to dice rolls and lottery drawings.

RANDOM.ORG offers *true* random numbers to anyone on the Internet. The randomness comes from atmospheric noise, which for many purposes is better than the pseudo-random number algorithms typically used in computer programs. People use RANDOM.ORG for holding drawings, lotteries and sweepstakes, to drive games and gambling sites, for scientific applications and for art and music. The service has existed since 1998 and was built and is being operated by Mads Haahr of the School of Computer Science and Statistics at Trinity College, Dublin in Ireland.

Games & Gambling: Coin Flipper | Dice Roller | Playing Card Shuffler | Keno Quick Pick | Birdie Funds | Lottery Quick Pi

Random Drawings: Random Draw Service | Guide to Drawings | Public Records | Drawing Result Widget | Drawing FA

Numbers: Integers | Sequences | Gaussians | Decimal Fractions | Integer Widget

Security: Passwords | Raw Bytes | iGoogle Password Generator

The Arts: Bitmaps | Audio Noise | Jazz Scales | Samuel Beckett

Primary Sources

- random.org (random numbers)
- Project Gutenberg (text, zip)
- Local image archive (jpg)
- XP / Ubuntu (exe)

Conversions

- Linux CLI utilities (encoding, compress, encrypt)
- Photoshop (images)
- Custom scripts (.text)

Examples

bitmap (.bmp)

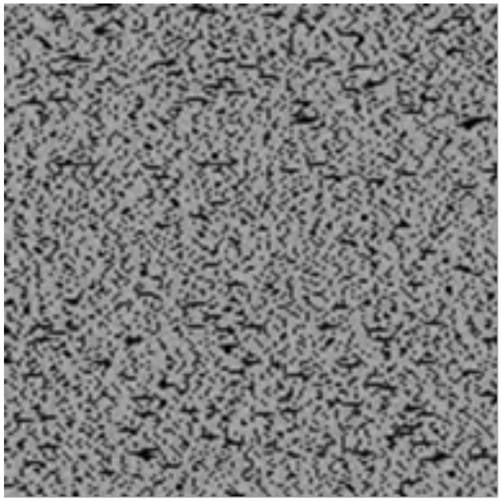
bitmap (process memory)

audio (.wav)

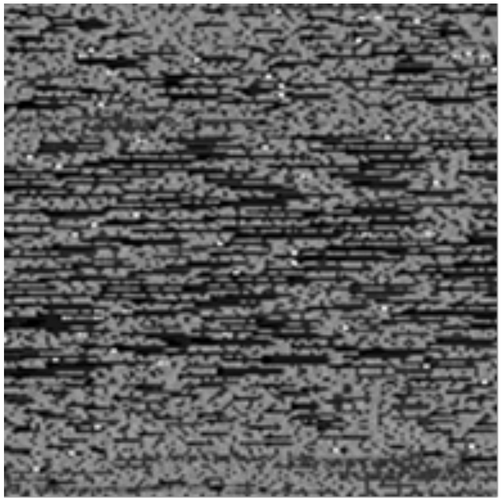
random

Examples

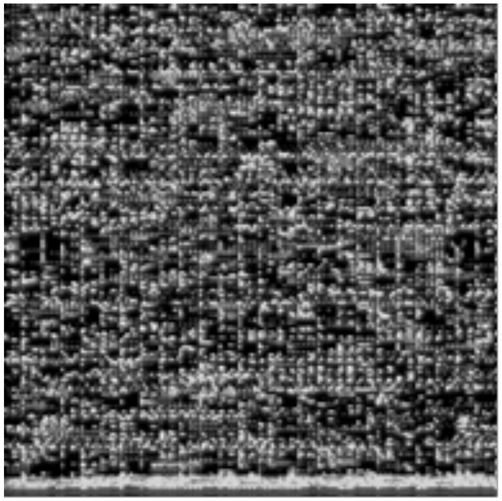
C++ source code



ASCII encoded English text

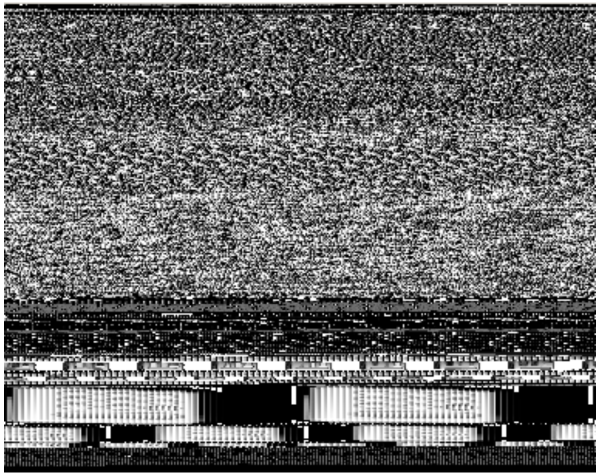


ASCII encoded HTML



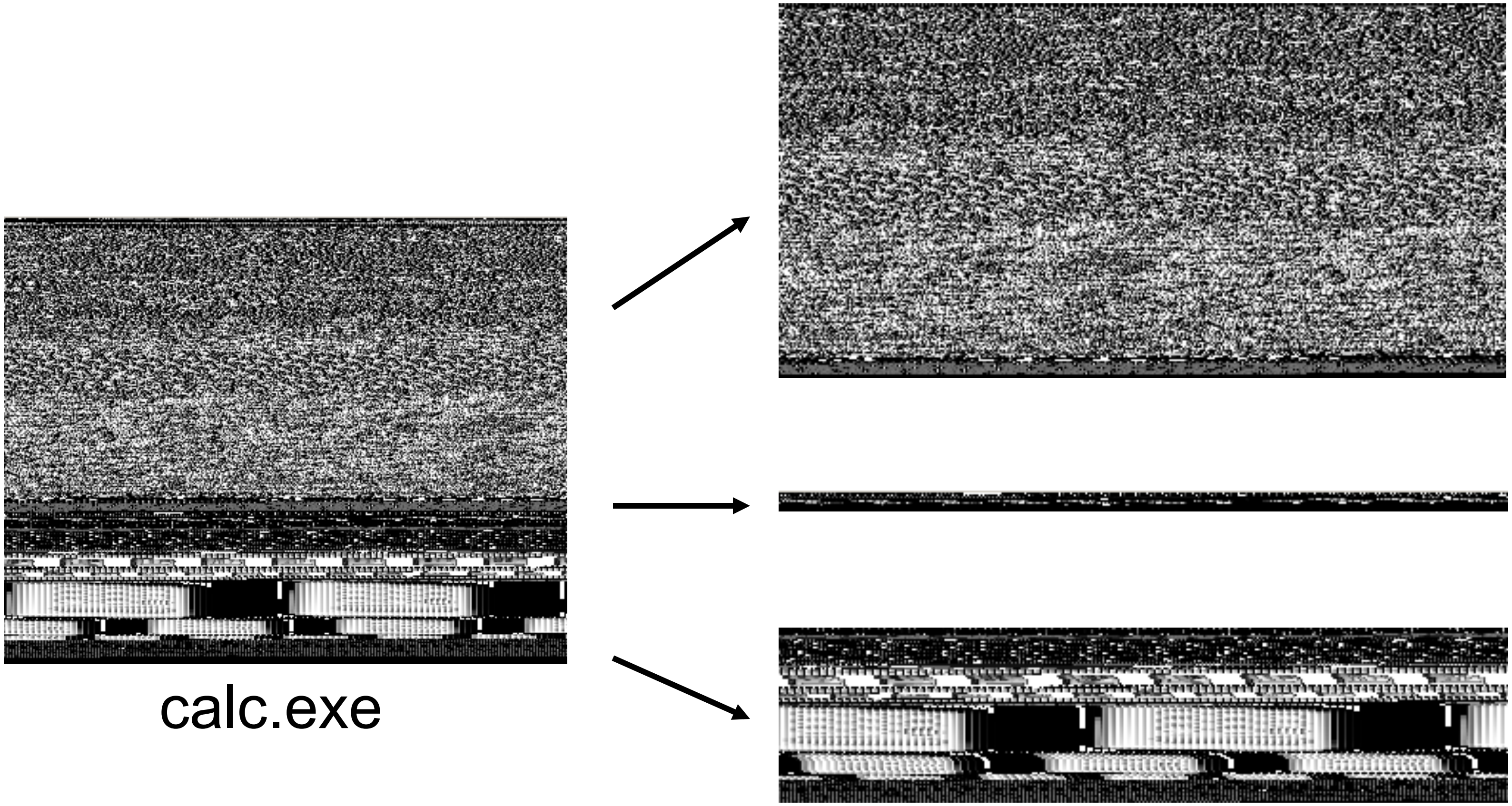
Basic Latin Unicode

Windows PE



calc.exe

Windows PE



calc.exe

.text

.data

.rsrc

Exemplars

	Mean		Shannon Entropy		CHI SQUARE		Hamming Weight	
		σ		σ		σ		σ
random	127.4039	2.3436	9.9826	0.0055	0.4873	0.2968	0.5627	0.0050
encrypt (AES256/text)	127.4778	2.3122	9.9830	0.0055	0.5008	0.2925	0.5627	0.0052
compress (bzip2/text)	126.6846	4.2372	9.9802	0.0069	0.2118	0.2480	0.5597	0.0134
compress (compress/text)	113.7279	8.8724	9.9662	0.0475	0.0681	0.1594	0.5316	0.0149
compress (deflate (png)	121.7824	12.9482	9.7103	0.7053	0.0460	0.1294	0.5430	0.0444
compress (LZW (gif) / image)	113.7543	8.2331	9.9455	0.0551	0.0203	0.0932	0.5153	0.0265
compress (mpeg/music)	126.2643	7.2295	9.8747	0.4421	0.0463	0.1260	0.5560	0.0245
compress (jpeg/image)	130.7620	12.7763	9.7314	0.8792	0.0647	0.1555	0.5744	0.0412
encoded (base64/zip)	84.4643	0.7402	9.7672	0.0192	0.0000	0.0000	0.5306	0.0037
encoded (uuencoded/zip)	63.7171	0.6968	9.7026	0.0209	0.0000	0.0000	0.4991	0.0053
machine code (linux elf)	116.4212	14.9786	7.6141	0.4381	0.0000	0.0000	0.4940	0.0429
machine code (windows PE)	107.3952	18.4625	8.0671	0.7279	0.0022	0.0385	0.4819	0.0497
bitmap	156.4776	69.1200	6.2298	3.6235	0.0000	0.0000	0.6635	0.1905
text (mixed)	88.5252	7.4828	7.4389	0.2427	0.0000	0.0000	0.5140	0.0146

Exemplars

	Mean		Shannon Entropy		CHI SQUARE		Hamming Weight	
		σ		σ		σ		σ
random	127.4039	2.3436	9.9826	0.0055	0.4873	0.2968	0.5627	0.0050
encrypt (AES256/text)	127.4778	2.3122	9.9830	0.0055	0.5008	0.2925	0.5627	0.0052
compress (bzip2/text)	126.6846	4.2372	9.9802	0.0069	0.2118	0.2480	0.5597	0.0134
compress (compress/text)	113.7279	8.8724	9.9662	0.0475	0.0681	0.1594	0.5316	0.0149
compress (deflate (png)	121.7824	12.9482	9.7103	0.7053	0.0460	0.1294	0.5430	0.0444
compress (LZW (gif) / image)	113.7543	8.2331	9.9455	0.0551	0.0203	0.0932	0.5153	0.0265
compress (mpeg/music)	126.2643	7.2295	9.8747	0.4421	0.0463	0.1260	0.5560	0.0245
compress (jpeg/image)	130.7620	12.7763	9.7314	0.8792	0.0647	0.1555	0.5744	0.0412
encoded (base64/zip)	84.4643	0.7402	9.7672	0.0192	0.0000	0.0000	0.5306	0.0037
encoded (uuencoded/zip)	63.7171	0.6968	9.7026	0.0209	0.0000	0.0000	0.4991	0.0053
machine code (linux elf)	116.4212	14.9786	7.6141	0.4381	0.0000	0.0000	0.4940	0.0429
machine code (windows PE)	107.3952	18.4625	8.0671	0.7279	0.0022	0.0385	0.4819	0.0497
bitmap	156.4776	69.1200	6.2298	3.6235	0.0000	0.0000	0.6635	0.1905
text (mixed)	88.5252	7.4828	7.4389	0.2427	0.0000	0.0000	0.5140	0.0146

Exemplars

	Mean		Shannon Entropy		CHI SQUARE		Hamming Weight	
		σ		σ		σ		σ
random	127.4039	2.3436	9.9826	0.0055	0.4873	0.2968	0.5627	0.0050
encrypt (AES256/text)	127.4778	2.3122	9.9830	0.0055	0.5008	0.2925	0.5627	0.0052
compress (bzip2/text)	126.6846	4.2372	9.9802	0.0069	0.2118	0.2480	0.5597	0.0134
compress (compress/text)	113.7279	8.8724	9.9662	0.0475	0.0681	0.1594	0.5316	0.0149
compress (deflate (png)	121.7824	12.9482	9.7103	0.7053	0.0460	0.1294	0.5430	0.0444
compress (LZW (gif) / image)	113.7543	8.2331	9.9455	0.0551	0.0203	0.0932	0.5153	0.0265
compress (mpeg/music)	126.2643	7.2295	9.8747	0.4421	0.0463	0.1260	0.5560	0.0245
compress (jpeg/image)	130.7620	12.7763	9.7314	0.8792	0.0647	0.1555	0.5744	0.0412
encoded (base64/zip)	84.4643	0.7402	9.7672	0.0192	0.0000	0.0000	0.5306	0.0037
encoded (uuencoded/zip)	63.7171	0.6968	9.7026	0.0209	0.0000	0.0000	0.4991	0.0053
machine code (linux elf)	116.4212	14.9786	7.6141	0.4381	0.0000	0.0000	0.4940	0.0429
machine code (windows PE)	107.3952	18.4625	8.0671	0.7279	0.0022	0.0385	0.4819	0.0497
bitmap	156.4776	69.1200	6.2298	3.6235	0.0000	0.0000	0.6635	0.1905
text (mixed)	88.5252	7.4828	7.4389	0.2427	0.0000	0.0000	0.5140	0.0146

Exemplars

	Mean		Shannon Entropy		CHI SQUARE		Hamming Weight	
		σ		σ		σ		σ
random	127.4039	2.3436	9.9826	0.0055	0.4873	0.2968	0.5627	0.0050
encrypt (AES256/text)	127.4778	2.3122	9.9830	0.0055	0.5008	0.2925	0.5627	0.0052
compress (bzip2/text)	126.6846	4.2372	9.9802	0.0069	0.2118	0.2480	0.5597	0.0134
compress (compress/text)	113.7279	8.8724	9.9662	0.0475	0.0681	0.1594	0.5316	0.0149
compress (deflate (png)	121.7824	12.9482	9.7103	0.7053	0.0460	0.1294	0.5430	0.0444
compress (LZW (gif) / image)	113.7543	8.2331	9.9455	0.0551	0.0203	0.0932	0.5153	0.0265
compress (mpeg/music)	126.2643	7.2295	9.8747	0.4421	0.0463	0.1260	0.5560	0.0245
compress (jpeg/image)	130.7620	12.7763	9.7314	0.8792	0.0647	0.1555	0.5744	0.0412
encoded (base64/zip)	84.4643	0.7402	9.7672	0.0192	0.0000	0.0000	0.5306	0.0037
encoded (uuencoded/zip)	63.7171	0.6968	9.7026	0.0209	0.0000	0.0000	0.4991	0.0053
machine code (linux elf)	116.4212	14.9786	7.6141	0.4381	0.0000	0.0000	0.4940	0.0429
machine code (windows PE)	107.3952	18.4625	8.0671	0.7279	0.0022	0.0385	0.4819	0.0497
bitmap	156.4776	69.1200	6.2298	3.6235	0.0000	0.0000	0.6635	0.1905
text (mixed)	88.5252	7.4828	7.4389	0.2427	0.0000	0.0000	0.5140	0.0146

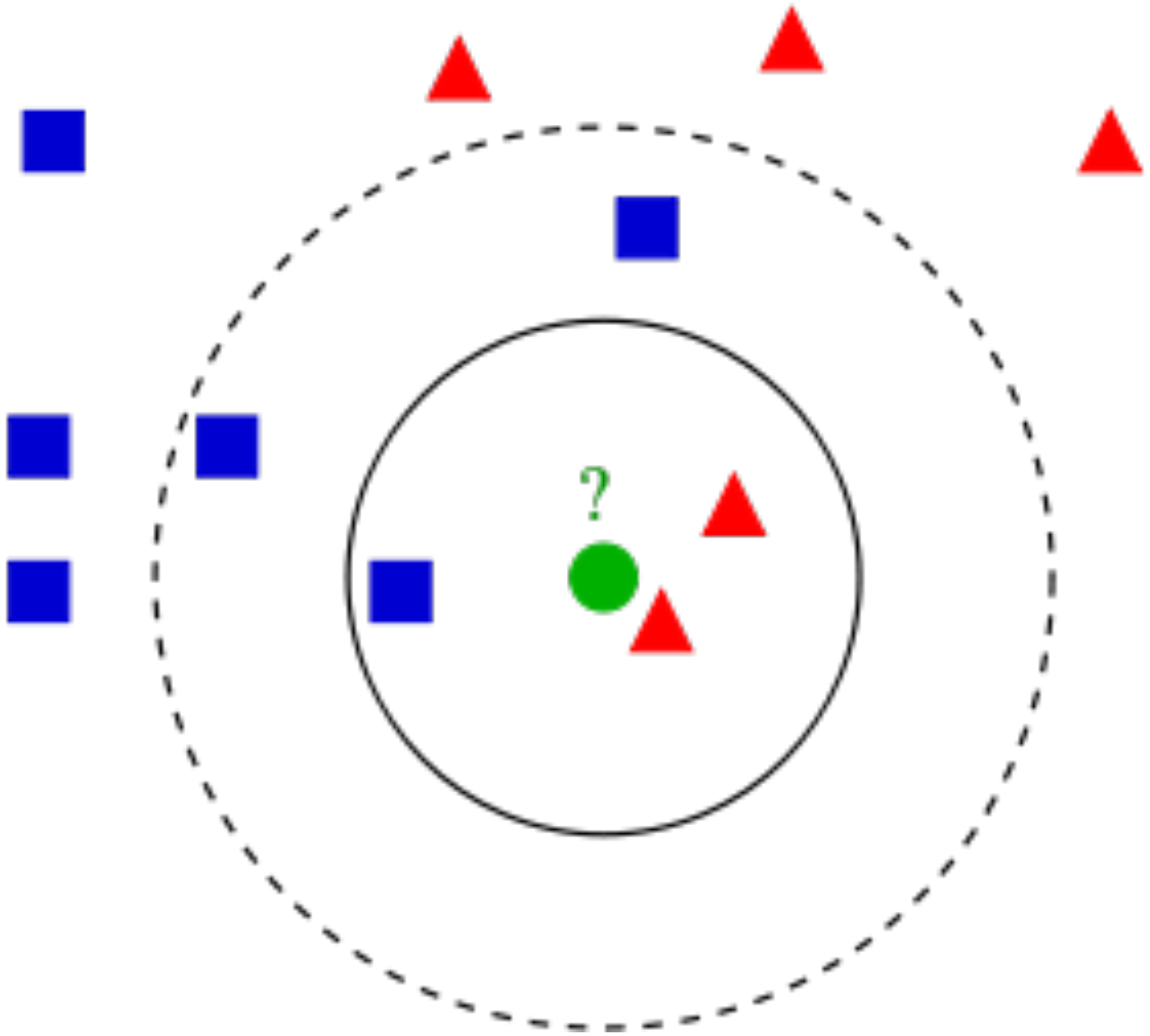
Exemplars

	Mean		Shannon Entropy		CHI SQUARE		Hamming Weight	
		σ		σ		σ		σ
random	127.4039	2.3436	9.9826	0.0055	0.4873	0.2968	0.5627	0.0050
encrypt (AES256/text)	127.4778	2.3122	9.9830	0.0055	0.5008	0.2925	0.5627	0.0052
compress (bzip2/text)	126.6846	4.2372	9.9802	0.0069	0.2118	0.2480	0.5597	0.0134
compress (compress/text)	113.7279	8.8724	9.9662	0.0475	0.0681	0.1594	0.5316	0.0149
compress (deflate (png)	121.7824	12.9482	9.7103	0.7053	0.0460	0.1294	0.5430	0.0444
compress (LZW (gif) / image)	113.7543	8.2331	9.9455	0.0551	0.0203	0.0932	0.5153	0.0265
compress (mpeg/music)	126.2643	7.2295	9.8747	0.4421	0.0463	0.1260	0.5560	0.0245
compress (jpeg/image)	130.7620	12.7763	9.7314	0.8792	0.0647	0.1555	0.5744	0.0412
encoded (base64/zip)	84.4643	0.7402	9.7672	0.0192	0.0000	0.0000	0.5306	0.0037
encoded (uuencoded/zip)	63.7171	0.6968	9.7026	0.0209	0.0000	0.0000	0.4991	0.0053
machine code (linux elf)	116.4212	14.9786	7.6141	0.4381	0.0000	0.0000	0.4940	0.0429
machine code (windows PE)	107.3952	18.4625	8.0671	0.7279	0.0022	0.0385	0.4819	0.0497
bitmap	156.4776	69.1200	6.2298	3.6235	0.0000	0.0000	0.6635	0.1905
text (mixed)	88.5252	7.4828	7.4389	0.2427	0.0000	0.0000	0.5140	0.0146

Exemplars

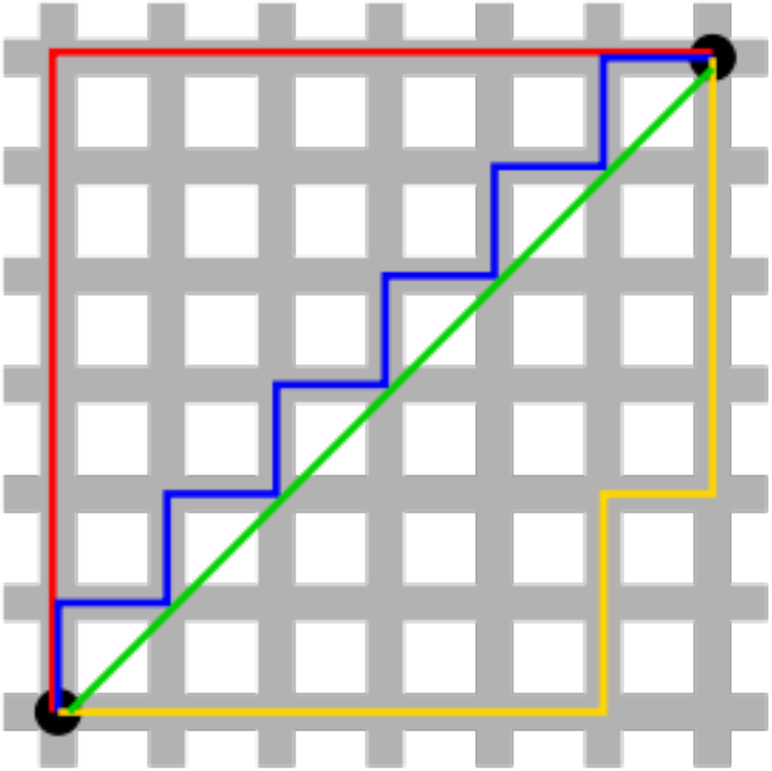
	Mean		Shannon Entropy		CHI SQUARE		Hamming Weight	
		σ		σ		σ		σ
random	127.4039	2.3436	9.9826	0.0055	0.4873	0.2968	0.5627	0.0050
encrypt (AES256/text)	127.4778	2.3122	9.9830	0.0055	0.5008	0.2925	0.5627	0.0052
compress (bzip2/text)	126.6846	4.2372	9.9802	0.0069	0.2118	0.2480	0.5597	0.0134
compress (compress/text)	113.7279	8.8724	9.9662	0.0475	0.0681	0.1594	0.5316	0.0149
compress (deflate (png)	121.7824	12.9482	9.7103	0.7053	0.0460	0.1294	0.5430	0.0444
compress (LZW (gif) / image)	113.7543	8.2331	9.9455	0.0551	0.0203	0.0932	0.5153	0.0265
compress (mpeg/music)	126.2643	7.2295	9.8747	0.4421	0.0463	0.1260	0.5560	0.0245
compress (jpeg/image)	130.7620	12.7763	9.7314	0.8792	0.0647	0.1555	0.5744	0.0412
encoded (base64/zip)	84.4643	0.7402	9.7672	0.0192	0.0000	0.0000	0.5306	0.0037
encoded (uuencoded/zip)	63.7171	0.6968	9.7026	0.0209	0.0000	0.0000	0.4991	0.0053
machine code (linux elf)	116.4212	14.9786	7.6141	0.4381	0.0000	0.0000	0.4940	0.0429
machine code (windows PE)	107.3952	18.4625	8.0671	0.7279	0.0022	0.0385	0.4819	0.0497
bitmap	156.4776	69.1200	6.2298	3.6235	0.0000	0.0000	0.6635	0.1905
text (mixed)	88.5252	7.4828	7.4389	0.2427	0.0000	0.0000	0.5140	0.0146

kNN Overview



http://en.wikipedia.org/wiki/File:KnnClassification.svg

Distance Metrics



Manhattan Distance (12)

Red

Blue

Yellow

Euclidean Distance (~8.48)

Green

$$d(\mathbf{p},\mathbf{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}.$$

http://en.wikipedia.org/wiki/Euclidean_distance

http://en.wikipedia.org/wiki/File:Manhattan_distance.svg

Normalization (0..1)

Arithmetic Mean	- n/255
Hamming Weight	- count of ones/total bits
Shannon Entropy	- H/10
Chi Square Probability	- not applicable

Overall Results

Random/Compressed/Encrypted	98.6%
Base64 Encoded	100.0%
Uuencoded	100.0%
Machine Code (ELF and PE)	96.7%
Text	98.7%
Bitmap	82.5%

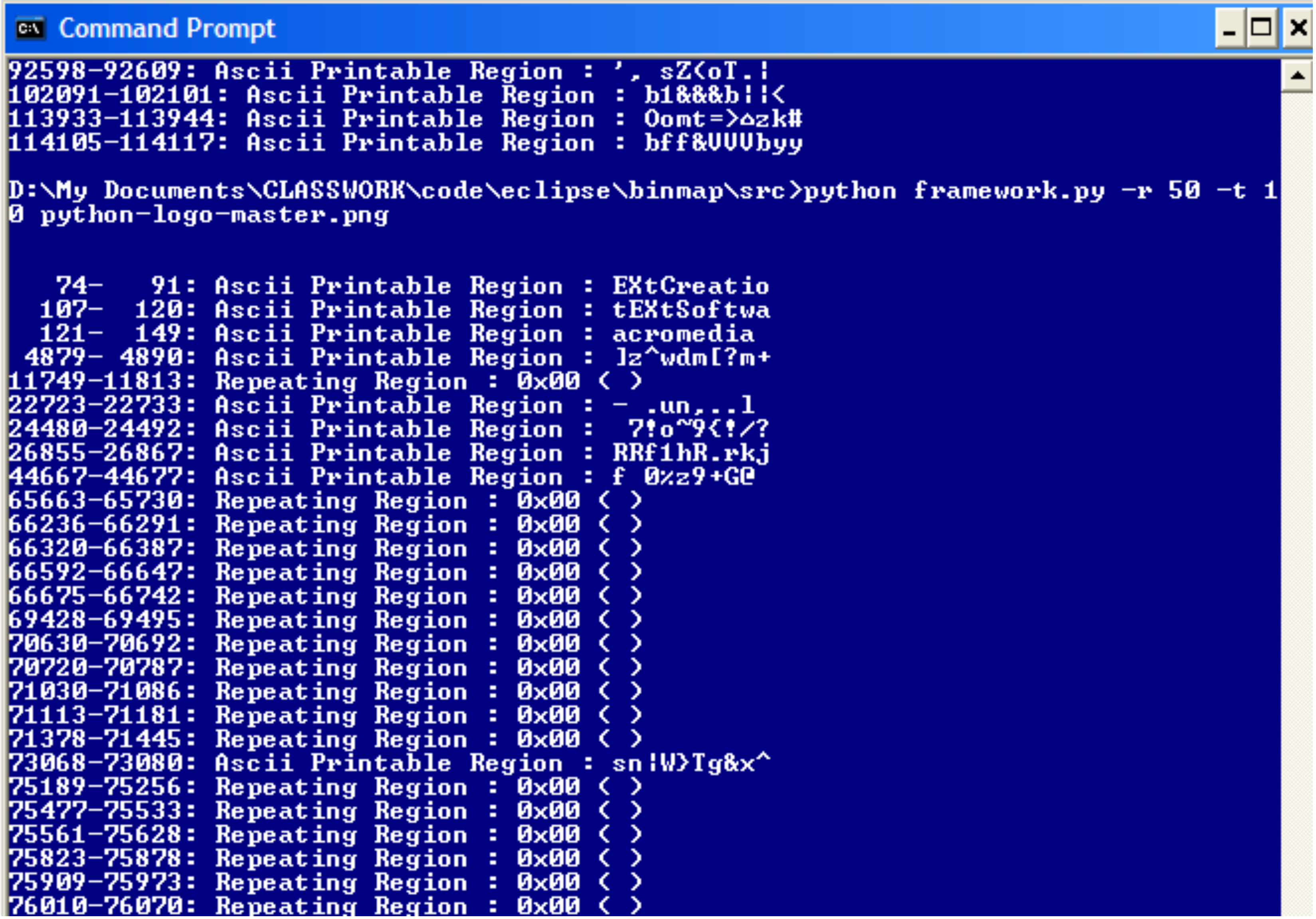
A bin mapping application

0-1023	(1.0 KB)	compressed/random/encrypted
1024-7679	(6.5 KB)	bitmap
7680-13823	(6.0 KB)	machine code
13824-15359	(1.5 KB)	bitmap
15360-15871	(0.5 KB)	machine code
15872-16383	(0.5 KB)	compressed/random/encrypted
16384-17407	(1.0 KB)	bitmap
17408-17919	(0.5 KB)	compressed/random/encrypted
17920-18943	(1.0 KB)	machine code
18944-19455	(0.5 KB)	compressed/random/encrypted
19456-20479	(1.0 KB)	machine code

…

- Firefox process memory dump (above)
- 1K window size
- .5K step size
- Perl
- kNN
- 14,000 exemplars
- Tested on variety of files: data, executable, and memory dumps

A Variant



```
92598-92609: Ascii Printable Region : ', sZ<oT.¦
102091-102101: Ascii Printable Region : b1&&&b¦¦<
113933-113944: Ascii Printable Region : Oomt=>⌂zk#
114105-114117: Ascii Printable Region : bff&VVVbyy

D:\My Documents\CLASSWORK\code\eclipse\binmap\src>python framework.py -r 50 -t 1
0 python-logo-master.png


   74-   91: Ascii Printable Region : EXtCreatio
  107-  120: Ascii Printable Region : tEXtSoftwa
  121-  149: Ascii Printable Region : acromedia
 4879- 4890: Ascii Printable Region : ]z^wdm[?m+
11749-11813: Repeating Region : 0x00 ( )
22723-22733: Ascii Printable Region : -  .un,..l
24480-24492: Ascii Printable Region :  7!o~9{!/?
26855-26867: Ascii Printable Region : RRf1hR.rkj
44667-44677: Ascii Printable Region : f 0%z9+G@
65663-65730: Repeating Region : 0x00 ( )
66236-66291: Repeating Region : 0x00 ( )
66320-66387: Repeating Region : 0x00 ( )
66592-66647: Repeating Region : 0x00 ( )
66675-66742: Repeating Region : 0x00 ( )
69428-69495: Repeating Region : 0x00 ( )
70630-70692: Repeating Region : 0x00 ( )
70720-70787: Repeating Region : 0x00 ( )
71030-71086: Repeating Region : 0x00 ( )
71113-71181: Repeating Region : 0x00 ( )
71378-71445: Repeating Region : 0x00 ( )
73068-73080: Ascii Printable Region : sn¦W>Tg&x^
75189-75256: Repeating Region : 0x00 ( )
75477-75533: Repeating Region : 0x00 ( )
75561-75628: Repeating Region : 0x00 ( )
75823-75878: Repeating Region : 0x00 ( )
75909-75973: Repeating Region : 0x00 ( )
76010-76070: Repeating Region : 0x00 ( )
```

Analysis Summary

- Bitmap confusion
- High entropy cluster
- New primitive types (or variants)
- Too many exemplars → Clustering
- Compiled language
- Weighting
- Confusion at transitions

Future

- Decision Tree
- API
- Plug-ins
- More primitive types
- Much improved interaction metaphors
- Importance of automating insights
- Obfuscation

A Parting Thought…

Dan Lunceford: MIT T-Shirt: "If you torture the data long enough, they will confess."

Brian Borchers: And just like we've learned about torturing prisoners, the data will tell you whatever you want to hear.

http://chamewco.livejournal.com/171857.html

See Also…

G. Conti and S. Bratus. “Voyage of the Reverser: A Visual Study of Binary Species;” *Black Hat USA;* August 2010.

B. Sangster, R. Ragsdale, G. Conti; “Automated Mapping of Large Binary Objects;” *Shmoocon*; Work in Progress Talk; February 2009.

G. Conti, E. Dean, M. Sinda, and B. Sangster; “Visual Reverse Engineering of Binary and Data Files;” *Workshop on Visualization for Computer Security (VizSEC)*; September 2008.

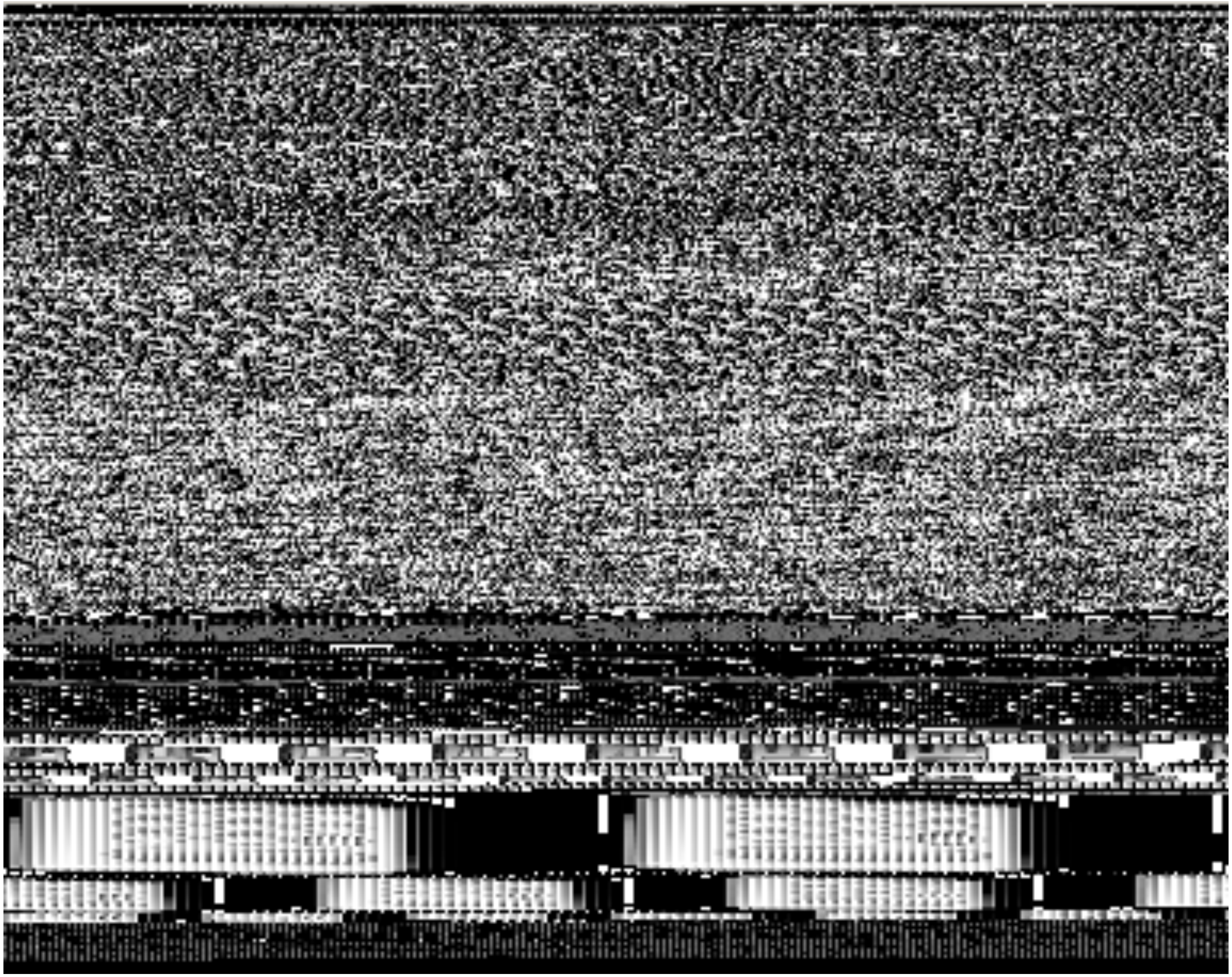
G. Conti and E. Dean; “Visual Forensic Analysis and Reverse Engineering of Binary Data;” *Black Hat USA*; August 2008.

…and of course this paper

Corpus Location:

Code Location:

Questions



Greg Conti // gregory.conti@usma.edu

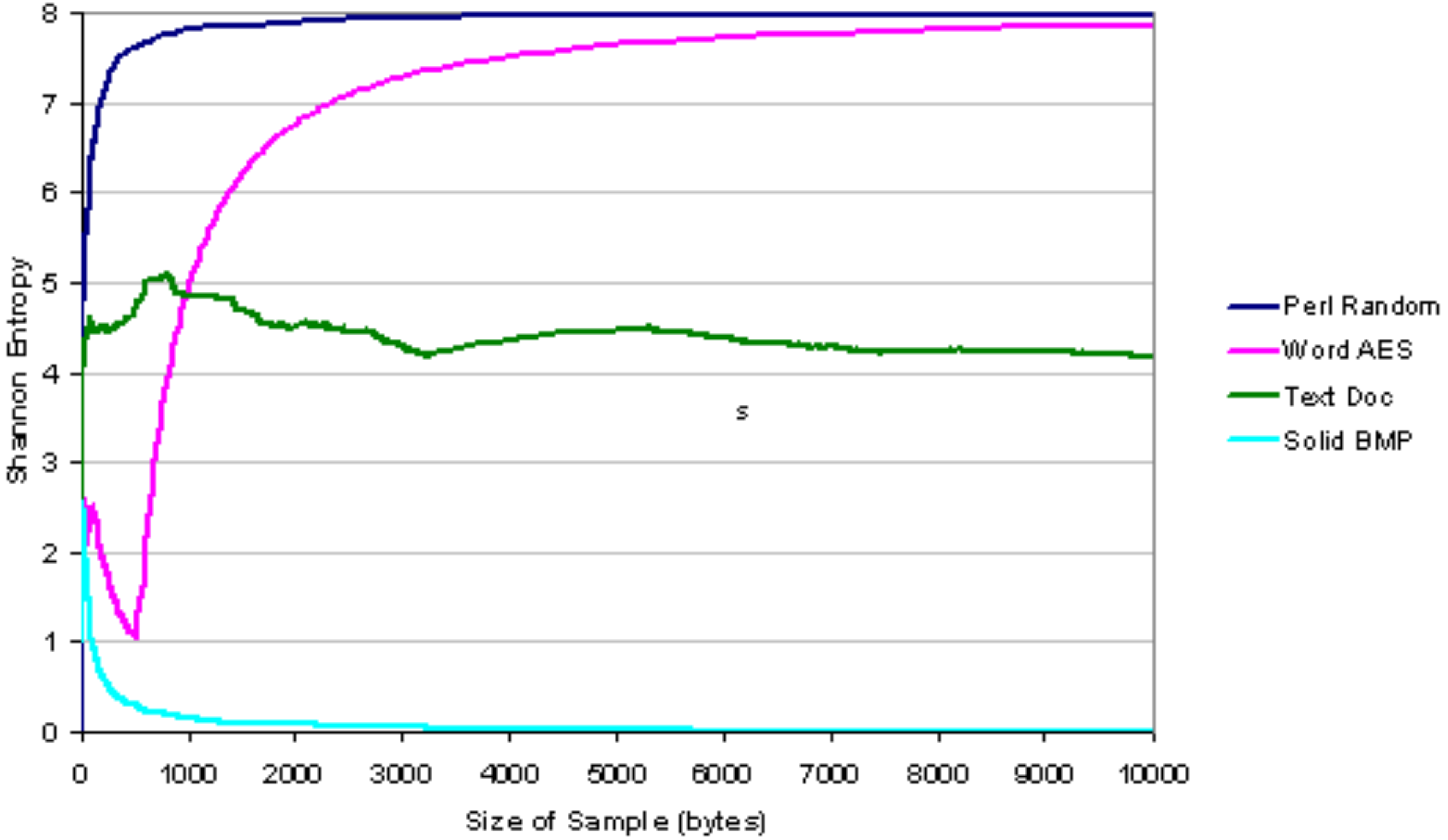
[redacted]

Confusion Matrix

random	.375	.37	.141	.018	.004	.022	.029	.041	0	0	0	0	0	0
encrypt(AES256/text)	.363	.386	.133	.019	.003	.024	.026	.044	0	0	0	.002	0	0
compress(bzip2/text)	.16	.163	.306	.078	.049	.073	.072	.097	0	0	0	.002	0	0
compress(compress/text)	.022	.03	.072	.588	.176	.04	.035	.031	0	0	0	.002	0	.004
compress(LZW(gif)/image)	.009	.007	.054	.148	.661	.041	.056	.024	0	0	0	0	0	0
compress(mpeg/audio)	.033	.036	.093	.031	.048	.455	.16	.13	0	0	0	0	0	.014
compress(deflate(png)/image)	.03	.037	.081	.027	.061	.177	.424	.101	0	0	.007	.043	0	.012
compress(jpeg/image)	.055	.054	.119	.031	.039	.116	.115	.441	0	0	.006	.009	0	.015
encode(base64/zip)	0	0	0	0	0	0	0	0	1	0	0	0	0	0
encode(uuencode/zip)	0	0	0	0	0	0	0	0	0	1	0	0	0	0
machine code(linux elf)	0	0	0	0	0	0	0	0	0	0	.823	.166	0	.011
machine code(windows PE)	0	.003	.001	.002	.001	0	.02	.002	0	0	.224	.721	.012	.014
text	0	0	0	0	0	0	0	0	0	0	0	.007	.987	.006
bitmap	0	0	0	.008	.002	.02	.034	.032	.006	.007	.024	.03	.012	.825

Window Size

(Shannon Entropy of 4 file types)


Shannon Entropy
Size of Sample (bytes)
s
Perl Random
Word AES
Text Doc
Solid BMP