BIOSCIENCES BIOTECHNOLOGY RESEARCH ASIA, April 2015. Vol. **12**(1), 745-752

Searching for Random Data in File System During Forensic Expertise

Vesta Matveeva* and Anna Epishkina

Cybernetics and Information Security Department, National Research Nuclear University "MEPhI" (Moscow Engineering Physics Institute), 31, Kashirskoe Highway, Moscow, Russian Federation.

DOI: http://dx.doi.org/10.13005/bbra/1720

(Received: 15 February 2015; accepted: 25 March 2015)

During forensic expertise the searching for random data is an important step. Existing approaches are based on verification of statistical properties of file data by means of test suites that estimate properties of random sequences. Some tests are not adapted to file system and are resource and time consuming, others have significant type I and II error. That is why authors have conducted a research in this field and suggest a new approach to assess statistical properties of data contents by visualisation of it. This approach was used to develop a program which testing shows that type I error in searching for random data is reduced to zero and type II errors for widely spread file formats is less than 1%.

Keywords: Digital forensics; Conceal data; Random data; Statistical tests; Encrypted data; Assessment of uniformity; Localization of heterogeneity; Wavelet transform; Compressed file formats.

Searching for random data in file system is conducted during forensic expertise for the following tasks:

a) Find data encrypted by cryptographically strong algorithms as a sign of information concealment. The search also can help in detection of malicious encrypted modules, that are hidden in the file system;
b) Find files or areas in the file system that are rewritten by random data as a sign of permanent data deletion;
c) Find hidden encrypted areas and file systems as a sign of malware activity or data concealment.

* To whom all correspondence should be addressed.
E-mail: vesta.matveeva@gmail.com;
avepishkina@mephi.ru

If files with random data are hidden in system catalogues, deliberately changed, removed or imitate ordinary files their detection is impossible during usual search by an expert. We need automated search that is based on statistical properties assessment because special test suites for estimation properties of random sequences are created. Examples of test suites that can be used for the task are: NIST[1], DIEHARD[2], Knuth's tests[3], TestU01[4], AIS31 Methodology[5]. Other approaches are used to assess particular properties of random data that lead to increased type II error because of complex file formats or have significant type I error, therefore, can't be used in forensic expertise.

Statistical test suites are not adapted for searching in file system and don't use features of file formats but can be used for testing individual

files as an array of bits / bytes. Their use for the analysis of all the files in the file system is resource- and time-consuming. After tests one receives results for every test as an integral value that don't reveal local heterogeneities in assessed distribution that is why it is difficult to interpret them.

Detecting files with random data is complicated because it is usually performed by an expert manually. If they are hidden in directories with system files or intentionally renamed or modified it is impossible to reveal them through ordinary search. In order to do this one uses automated search through signatures that are added by means used for encryption (extensions or header) or with the use of the evaluation of the statistical distribution of bytes within a file.

As for existing automated means of encrypted files search, one has been using the«Passware Forensic Kit» software6, the module of which is embedded in forensic tools: BelkasoftEvidence Center[7], EnCase Forensic[8], Oxygen Forensic Kit + Passware Edition[9]. In the course of testing the module a limit on the minimum size for files with no header (> 273 KB) has been revealed as well as false positive up to 6%, which is quite significant.

Therefore, this article proposes an approach to finding files that might contain random data, taking into account the internal structures of the data. The main purpose is to quickly select files that lack for the heterogeneity of distribution of bytes, reducing type I error to zero. This approach can be applied to identify sectors or other areas different in size in the file system with random data.

In order to identify random data in traffic one uses the same methods as for the evaluation of the distribution of bits / bytes in the file. In this case, data section in the packet is represented as a sequence of bits / bytes and evaluated.

Prerequisites for the research

One of the main stages of forensic expertise is the comparison of the extension and the file format. Moreover, the file format is usually determined by the signature database of known formats. For this purpose the following forensic modules can be used: OSForensics[10], Autopsy mismatch Module[11], as well as search by using the file program in the Linux operating system of all files the format of which is defined as data.

During this procedure one detects suspicious files. Further, one faces the problem of searching among them those which may contain random data.

According to the definitions in NIST[1] random data is defined as having a uniform and independent distribution of the data elements. Test suites mentioned in Introduction aim to verify these properties.

The main and the quickest way to identify randomness of bytes distribution in a file is counting its entropy. The evaluation of the content using entropy is based on counting the occurrence probabilities of symbols in a file, which evaluates the uniformity of their distribution. Testings[12, 13] have shown that the entropy values for compressed file formats pdf, docx, rar, zip, wmv, jpg, png, mpg, etc. intersect with the values for encrypted files. Because of this, entropy cannot be used to search for files with random data, but it allows to reveal the main "competitors" in the search for them, namely, the files with compressed data.

Modern symmetric algorithms, that are used by encryption software, have the following properties:

a) Good mixing of input data, i.e. when an input is changed slightly (for example, flipping a single bit) the output changes significantly. The algorithms satisfies strict avalanche criterion;
b) Using combination of substitution and permutation ciphers;
c) Uniformity and independence are the properties of output data.

No restrictions are imposed on the statistical properties of the compressed data. The main aim of the compression operation is to reduce the size of the compressed data and to provide the ability to recover the original data from the compressed data with or without loss depending on the task: archives use lossless compression, but images, audio, video files – lossy compression. Compression algorithms do not satisfy strict avalanche criterion and depend on properties of data to be compressed.

Lossless compression used for archives is based on probability algorithms that equalize the frequency of occurrence of compressed data elements: common elements are replaced by short

code words, rare ones by longer ones. Thus, in the compressed data one equalizes the probability of the occurrence of symbols, which is knowingly equal for various element values in random and encrypted data. Plain text data is compressed in volume by several times, which does not hold true with regard to the compression of encrypted or random data. Using of probability algorithms leads to high entropy for compressed data that, therefore, satisfy tests for uniformity.

Mentioned compression algorithms use substitution only: RLE, LPC, DC, MTF, VQ, SEM, Subband Coding, Discrete Wavelet Transform, PPM, Huffman and Shannon-Fano coding, arithmetic and probability coding, etc. Repeated sequences of elements or sequences of elements defined in the codebook are replaced by codes. The codebook can be constructed in the process of coding or specified at the beginning or coding.

However, it should be noted that in order to improve the quality of compression one applies permutation algorithms: BWT, MFT, etc. The purpose of the permutation is a transformation of the compressed sequence which would allow to place elements with the same values together with the possibility of a clear inverse transformation after the permutation. Therefore, during the permutation one achieves a high dependency between the elements, which are subsequently replaced by codes.

Other statistic properties of compressed data are based on properties of data being compressed, the presence of certain symbol values in it and combination of compression algorithms

Table 1. The list of files for testing

Filename	Entropy	Chi-square test value	Monte Carlo value	Arithmetic mean
test.jpg	7.97	429711.58	3.23	124.59
test.ods	7.99	1956.72	3.13	126.94
test.aes	7.99	248.46	3.14	127.47
test.wmv	7.97	99963.81	3.09	126.15
test.mp3	7.94	1542810.27	3.11	125.11
test.pdf	7.90	218024.01	3.14	124.79
test.mpeg	7.89	769440.90	3.28	116.38
test.7z	7.99	231.99	3.14	127.49
test.zip	7.99	933.50	3.16	127.48
test.m4a	7.87	757648.04	3.17	118.03
test.mp4	7.98	244899.91	3.04	130.73

Table 2. Type I error as a result of the experiment

Evaluated Value Name of the collection	Number of files	The proposed approach Type I error (100x100)	The proposed approach Type I error (2000x50)	The proposed approach Type I error (100x100 and 2000x50 and statistics)	Chi-squared test Type I error
File types1(random)	445	0%	0%	0%	1,798%

Table 3. Type II error as a result of the experiment

Evaluated Value Name of the collection	Number of files	The proposed approach Type I error (100x100)	The proposed approach Type I error (2000x50)	The proposed approach Type I error (100x100 and 2000x50 and statistics)	Chi-squared test Type I error
filetypes1	1 348	8,853%	7,336%	6,026%	5,778%

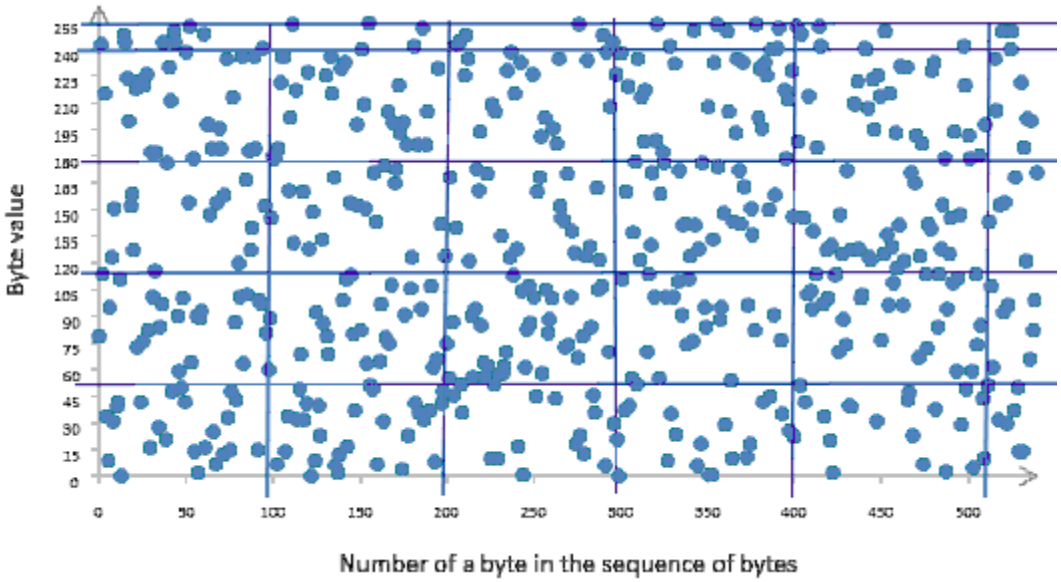

Fig. 1. A fragment of a plane where points are represented by elements of sequence of bytes

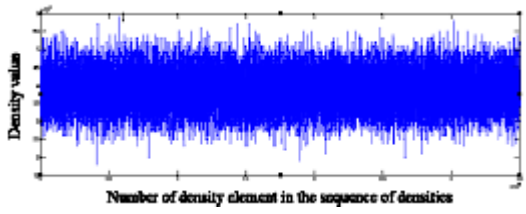

Fig. 2. The fragment of the distribution of density values for a file encrypted with AES algorithm

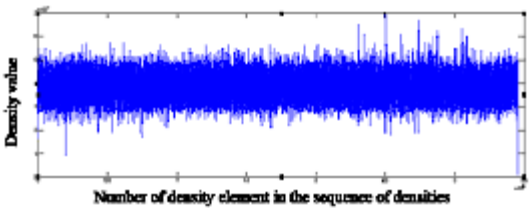

Fig. 3. The fragment of the distribution of density values for a zip format file

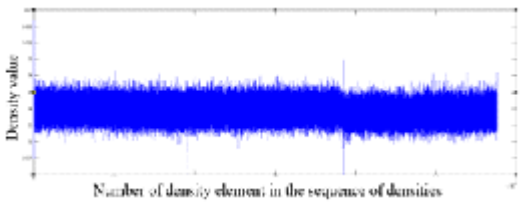

Fig. 4. The fragment of the distribution of density values for a jpg format file

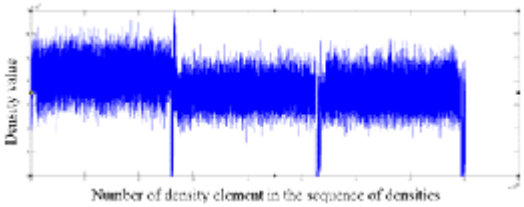

Fig. 5. The fragment of the distribution of density values for an m4a format file

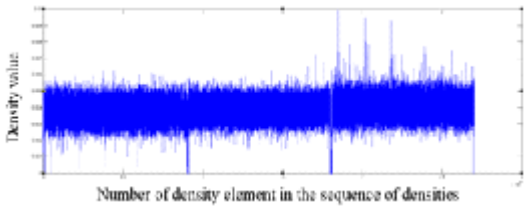

Fig. 6. The fragment of the distribution of density values for an mp4 format file

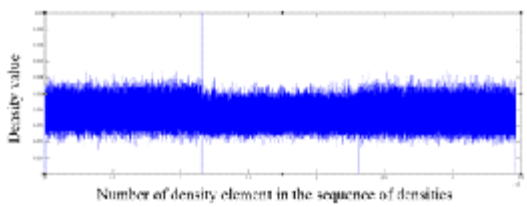

Fig. 7. The fragment of the distribution of density values for an mp3format file

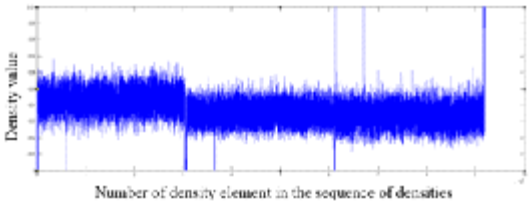

Fig. 8. The fragment of the distribution of density values for an mpeg format file

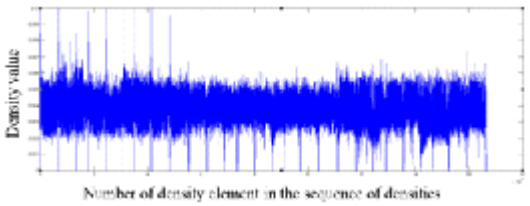

Fig. 10. The fragment of the distribution of density values for an odt format file

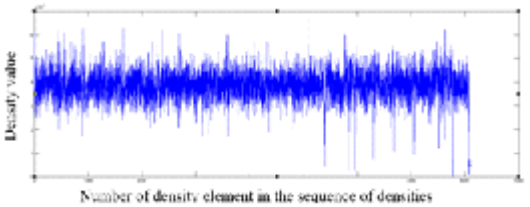

Fig. 9. The fragment of the distribution of density values for an ods format file

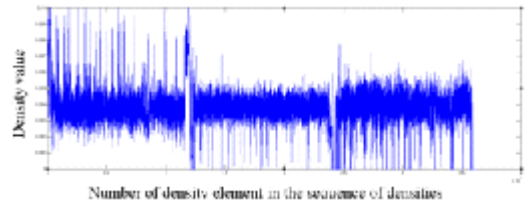

Fig. 11. The fragment of the distribution of density values for a pdf format file

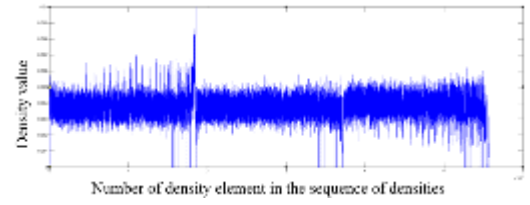

Fig. 12. The fragment of the distribution of density values for a wmv format file

that were used.

Lossy compression used for audio, video files and images is based on revealing of significant components of data and application lossless compression to them. Such transformation leads to high entropy for compressed data that, therefore, satisfy some tests for uniformity.

It clear that the compression does not provide the mixing properties of the data being compressed and do not aim to provide properties of random data for output data; however, due to the use of multiple algorithms together and equalizing the probability of the occurrence of symbols, they become competitors of files with random data, since they satisfy the uniformity property and some tests for independence.

Authors conclude that the character of the distribution of bytes in compressed files and encrypted files (or files with random data) will vary with the condition of its evaluation based on the location of data in the file, i.e. the positions of bytes relative to each other.

For this purpose authors suggest a simplified model of evaluation which is given in the "Evaluation of the statistical properties of a file" section.

The Evaluation of the Statistical Properties of a File

Existing approaches which have been described above deal with numerical sequences, therefore we will represent file f as a finite sequence of bytes $\{y_1, y_2,....y_N\}$of length N, where $y_i \in Y=\{0,1...,255\}$ $|Y|=k=256$ is the cardinality of Y, and N is file size The sequence is usually analyzed byte by byte or by blocks from left to right in order to set the uniformity and independence of elements in the sequence relative to each other.

Authors consider this sequence in a different way. One will plot the sequence onto the plane, the x-axis of which is plotted with values from 0 to N-1, and the y-axis with those from 0 to 255. One will fill the plane with points P (x, y), such that x is the number of a byte in the sequence of bytes of the file, and y is the value of the byte.

Authors have introduce a definition.

Definition. Points on the plane have uniform distribution if the probability of finding a point in the plane in a fragment of minimum size: 1x1 is the same for any fragment of this size and is equal to $P_{1x1}=1/k=1/256$. Thus, there are no areas of high

and low concentration of points on the plane.

It is logical to assume that points will be applied evenly on the plane for the random data and the encrypted data. This follows from the fact that the subsequence of the random sequence is also subsequent, and thus has a uniform and independent distribution of all its values. Since elements are applied to the plane in accordance with their position in the sequence and the value, they will be arranged evenly on this plane. An example of a fragment of such a plane (hereinafter - the plane of distribution) is shown in figure 1.

If the distribution in the plane is uniform, the density of element distribution is close to1/256 in each fragment.

For each fragment, the density of points in a fragment is relative to the size of that fragment and is calculated, i.e.:

$$\rho_i = \frac{\text{(number of points from the file in the i – th fragment)}}{\text{(number of all points in the i – th fragment} = W * H)}$$

Calculating the density of each fragment a sequence of densities is formed, thus a rearrangement is carried out:

$$S : \{y_1, y_2, \ldots, y_N\} \rightarrow \{\rho_1, \rho_2, \ldots, \rho_L\},$$

equation

where L is the number of the resulting fragments as a result of moving through the contents of the plane using the method of a sliding window sized *WxH*.

There is a good reason to carry out the move from fragment to fragment using the sliding window method from left to right, since the accumulation of the values of bytes of one value will be placed horizontally. At vertical motion all the values that are available in the subsequence of length W will be captured, in connection with which a significant surge or drop in the density values will be smoothed.

It is also worth noting that it is advisable to choose the window size based on the size of the sequence (file) being tested, since the aim is to get a larger size of densities vector, which is achieved by reducing the size of the fragment given a small file size.

For random data the density for each fragment is close to 1/*k*=1/256. Therefore, the distribution of densities can be described by the following statistic values:

$$\begin{cases} \mu_1(\rho) = \frac{1}{k} = \frac{1}{256} \approx 0{,}0039 \\ \mu_2(\rho) = \frac{k - H}{WHk^2} \end{cases}.$$

where μ_k – central moment of power *k*, $k \in N$

These statistical values can be used to differentiate between encrypted files and files of other formats.

Density sequences obtained for different file formats will differ greatly from density sequences for files with encrypted data, due to the presence of operational information in the files that defines the distribution of the values of bytes in the file, and the features of file structure. For random data the densities for the fragment of any size should be close to the 1/256 reference value. While for other file formats operational information expressed in a cluster of the predetermined range of byte values in the plane will lead to the presence of surges or drops in density values in the sequence. In this connection, it is possible to produce the identification of encrypted and random data by means of checking density sequences for the presence of expressed deviations from the mean density value in the density vector. Examples are given in the "Visualization" of file contents" section.

Visualization of file contents

The distributions of densities for various file formats with high entropy were obtained. One will take files sized from 1MB to 10MB, window size: 100x100, and obtain a sequence of densities for each of them.

The following table lists the files and the values of some mathematical statistics for them.

As one can see from the table, all the selected files have high entropy which is characteristic of encrypted files and random data. Below in Fig. 2 - 12 fragments of the distribution of densities for different file formats are shown.

As one can see from the graphs, file formats other than encrypted with AES algorithm contain in the density distribution visible surges relative to the mean valuewhich can serve as an indicator of the inhomogeneous distribution of

bytes in the file, and consequently of the lack of encryption.

For the compressed formats and random data mean density value fluctuates around the value of 1/256~0,0039.

Constructing such graphs gives an expert a visual representation of the distribution within a file and allows to identify the encryption by the absence of local deviations in the distribution of densities. Based on mentioned method of files visualization a program was developed, that can detect group and local deviations in the distribution by using mentioned statistics and wavelet analysis[14].

To localize deviations the following formula for wavelet transform with Haar mother wavelet was used:

$$W_{ab} = \frac{1}{\sqrt{a}} \sum_{i=b}^{b+a} \rho_i * \psi_{HAAR}\left(\frac{i-b}{a}\right), \quad (4)$$

where a – scale parameter (i.e., the number values of elements in a sequence ρ_{diff}, used to calculate the wavelet coefficient), b – shift parameter (i.e., the position in the sequence ρ_{diff}, from which the count of elements in the sequence ρ_{diff} begins), ρ_i – the value of i element in the sequence of densities. The selection of this function as a mother wavelet has been due to its asymmetrical shape and first moment equal to zero. Results of program testing are given in section "Results and discussion".

RESULTS AND DISCUSSION

Testing of developed program was performed on collection «filetypes1»[15] that contains files of different formats and random data. The collection consists 1 348 different files and 445 files with random data.

The proposed approach was tested using 100x100 and 2000x50 window fragments excluding file headers. Test results are shown in Table 2, Table 3 and are compared with test results of Chi-square test for uniformity with level of significance 0.01 for the same collection.

Experiment shows that the proposed approach reduces type I error to zero, type II error is contributed by files of jb2 format that is a graphical one and is not spread widely. Moreover, all existing tests define files of this format as random. Excluding this format makes type II error less than 1%. By the way, type I error for Chi-square test for uniformity can be reduced by decreasing the value of significance level but this leads to the considerable increase of type II error.

When testing the proposed approach on the same collection of files, including their headers, type II error decreased. Using different window sizes for every file leads to the decrease of type II error.

CONCLUSION

During forensic expertise the searching for random data is an important step. Existing approaches are based on statistical tests for assessment of properties of random data. Some tests are not adapted for use in file system and don't take into account file formats specific. Moreover, their use for the analysis of all the files in the file system is resource- and time-consuming because they are used together to overlap errors of each other. Other test have significant type I and II errors.

For the clarity and speed of analysis, authors have suggested an approach to visualize the distribution of bytes in a file. The result of this analysis is the graphs of the distribution of densities in which random data is characterized by the absence of pronounced deviations from the uniform distribution which definition is introduces in the article.

One should separately note that the proposed approach can be applied not only to files but also to individual clusters of the file system and allows to find encrypted or random data in them, which means hidden encrypted file systems or the fact of data overwriting. One can also search for distribution deviations in the file / sector / cluster itself on the basis of the proposed approach of visualizing the content.

Based on proposed approach of visualization a program was developed that performs searching for random data in the file system automatically using statistic values and wavelet transform. Program testing shows that type I error is reduced to zero and type II error for files of spread formats is less than 1%.

REFERENCES

1. NIST SP800-22. A Statistical Test Suite for Random and Random Number Generators for Cryptographic Applications. NIST, 2010; 131.
2. Marsaglia G.: The Marsaglia random number CDROM including the diehard battery of tests, 1995; http://stat.fsu.edu/pub/diehard/ (last accessed 03/17/2015).
3. Knuth, D. E.: The Art of Computer Programming, Volume 2: Seminumerical *Algorithms*, 3rd ed., Addison-Wesley Pearson Education, Canada, 1997.
4. L'Ecuyer P. TestU01: A C library for empirical testing of random number generators / P. L'Ecuyer, R. Simard // ACM Transactions on Mathematical Software (TOMS), 2007; **33**(4).
5. Killmann W.: A proposal for: Functionality classes for random number generators. *Tech. Rep. 2. Bundesamtf¨ur Sicherheit in der Informationstechnik (BSI)*, 2011; 133.
6. Passware Kit Forensic 2015; http://www.lostpassword.com/kit-forensic.htm (last accessed 03/17/2015)
7. Belkasoft Evidence Center 2015; http://ru.belkasoft.com/ru/ec (last accessed 03/17/2015)
8. Using Passware Kit Forensic with EnCase; http://www.lostpassword.com/encase.htm (last accessed 03/17/2015)
9. Oxygen Forensic Suite; www.oxygen-forensic.com/ (last accessed 03/17/2015)
10. Identify suspicious files and activity, 2015; http://www.osforensics.com/identify.html (last accessed 03/17/2015)
11. Autopsy:History, 2014; http://www.sleuthkit.org/autopsy/history.php (last accessed 03/17/2015)
12. Jozwiak, I., Kedziora, M. and Melinska, A.: Theoretical and Practical Aspects of Encrypted !ontainers Detection. *Digital Forensics Approach*, *Dependable Computer Systems*. – Springer Berlin Heidelberg, Berlin, Germany, 2011; 75-85.
13. Weston P.: Forensic Entropy Analysis of Microsoft Windows Storage Volume. *Information Security for South Africa*, 2013; 1-7.
14. Farge, M.: Wavelet transforms and their applications to turbulence, *Annual Review of Fluid Mechanic*s, 1992, **24**: 395–458.
15. Digital corpora Filetypes1, 2014; http://digitalcorpora.org/corp/nps/files/filetypes1/ (last accessed 03/17/2015)