Edith Cowan University
Research Online

Australian Digital Forensics Conference   Conferences, Symposia and Campus Events

2014

Towards a set of metrics to guide the generation of fake computer file systems

Ben Whitham
University of New South Wales, Canberra, Australia

Follow this and additional works at: https://ro.ecu.edu.au/adf

Part of the Computer Engineering Commons, and the Computer Sciences Commons

DOI: 10.4225/75/57b3dc72fb878
12th Australian Digital Forensics Conference. Held on the 1-3 December, 2014 at Edith Cowan University, Joondalup Campus, Perth, Western Australia.
This Conference Proceeding is posted at Research Online.
https://ro.ecu.edu.au/adf/130

TOWARDS A SET OF METRICS TO GUIDE THE GENERATION OF FAKE COMPUTER FILE SYSTEMS

Ben Whitham
University of New South Wales, Canberra, Australia
b.whitham@student.adfa.edu.au

Abstract

Fake file systems are used in the field of cyber deception to bait intruders and fool forensic investigators. File system researchers also frequently generate their own synthetic document repositories, due to data privacy and copyright concerns associated with experimenting on real-world corpora. For both these fields, realism is critical. Unfortunately, after creating a set of files and folders, there are no current testing standards that can be applied to validate their authenticity, or conversely, reliably automate their detection. This paper reviews the previous 30 years of file system surveys on real world corpora, to identify a set of discrete measures for generating synthetic file systems. Statistical distributions, such as size, age and lifetime of files, common file types, compression and duplication ratios, directory distribution and depth (and its relationship with numbers of files and sub-directories) were identified and the respective merits discussed. Additionally, this paper highlights notable absences in these surveys, which could be beneficial, such as analysing, on mass, the text content distribution, file naming habits, and comparing file access times against traditional working hours.

Keywords
Fake files, fake file systems, cyber deception, honey-files, canary files, decoy documents

INTRODUCTION

Since the implementation of the Multics Operating System 40 years ago, most current operating system software provides a facility to organise programs and data files in hierarchical structures (Henderson, 2004). File systems can contain the base executable version of the operating system in additional to applications and configuration data (Kim and Spafford, 1994). Users can also employ file systems as a central repository of their collective knowledge, such as correspondence, study material, travel information, financial records and inventions (Salminen, et al., 1997).

Fake file systems are a set of synthetic data files arranged in a manner to replicate real digital repositories. While artificially created file systems are used in cyber deception to lure unauthorised users or to sustain a falsehood, they also have other important benign applications, such as supporting research and development into replication and archiving. The key limitation in their application is an absence of common or agreed baseline metrics for building or evaluating these synthetic products. The consequence of this research gap is that fake file systems can be produced that either lack realism, contain uniquely identifiable characteristics that can be detected, or both. This paper presents the results of a literature review of the previous thirty years of published file system surveys. The aim of the research is to extract metrics and identify common themes to improve the realism and reduce detectability of synthetic file systems. These results are the first step in a multi-stage process. The next stage of the research is to test these common statistical distributions by building replica file systems.

FILE ATTRIBUTES

Most modern operating systems provide users with visibility of characteristics of accessible files, such as content, attributes and metadata (Carrier and Spafford, 2004). Characteristics such as file name, file size, location, and file type or file-extension assist users to identify specific documents and navigate the file system (Ritchie and Thompson, 1974). Time-based attributes can also be accessed by the user, such as: (1) atime - updated when the file is read; (2) mtime - updated when the file contents change (the default file time in most cases); and (3) ctime - updated when the file or certain file characteristics (owner, permissions) change (Daley and Neumann, 1965). The user and group identifier were also introduced in Unix (and later adopted by Microsoft's New Technology File System) providing information on who can read, write and execute the file (Buchholz and Spafford, 2004). The owner of the file can also modify the filename, which can provide clues to the purpose, version number and creation date (Anquetil and Lethbridge, 1997).

Additional information can also be obtained from documents with further analysis, such as inspecting the content of the file to classify the type and whether or not the file is encrypted or compressed (Erbacher and Mulholland, 2007; Li et al., 2005; McDaniel, M., & Heydari, 2003). Some file types also save additional metadata in the contents, such as the author's name, initials, organisation name, computer's name, document revisions and versions, template information and comments. Files can be differentiated through review of their content's entropy (Shannon, 2001), word and character count, strings, language and writing style analysis (Mosteller and Wallace, 1963; Somers and Tweedie, 2003; Abbasi and Chen, 2008) and cryptographic hash (Kim, 1994).

Combinations of characteristics can be used to categorise and classify files (Goncalves and Jorge, 2003; Ellard, et al., 2003), and unexpected combinations can be used to differentiate the real from the fake (Rowe, 2005).

FAKE FILE SYSTEMS

Individual fake files have been referred to as honeytokens (Spitzner, 2003), honeyfiles (Yuill et al., 2004), digital decoys (Kushner, 2003), decoy files (Bowen et al., 2009), and canary files (Whitham, 2013a). Stoll (2005) was the first to publicly discuss the use of fake files in cyber deception. He handcrafted several documents to successfully track and identify an unauthorised user on his network. His work inspired several others. Bowen, et al (2009) developed the Decoy Document Distributor System, which is a tool for generating and monitoring individual fake files. Yuill, et al (2004) also proposed a fake file generation, distribution and alert system to simplify the operation and management of individual fake files. Finally, Whitham (2013b) proposed a method to automate the generation of fake files to perform a watermarking role, using content from the local and nearby directories.

On a mass scale, Gerwehr and Glenn (2003), and Cohen (2003) independently noted that whole fake file systems can be generated for honeypots to suggest that real users have been using the system in normal ways. Garfinkel (2009) and Rowe (2006) both proposed generating a synthetic file system using publicly available data, Internet searches or other random text. Assuming that there are no copyright concerns, these files can be assembled into a hierarchical set of folders resembling a modern file repository.

RESEARCH APPLICATION

Realism is critically important for fake files deception (Whitham, 2014). The primary challenge with artificially constructed datasets is that while the text content may be genuine, the document arrangements are rarely a true representation of a real file system (Arnold and Bell, 1997; Ellard et al., 2003). File system corpora should include corruptions, compressed files, duplications, previous versions of documents, natural modifications to file characteristics due to operating system processes, and realistic distributions of time meta-data, directories and files (Chow et al., 2007). Artificially generated corpora can also fail to mimic the diverse behaviours of different users (Ellard et al., 2003) and human document management processes (Goncalves and Jorge, 2003).

Poor fake file system constructions can also introduce identifiable characteristics that allow a forensic examiner or intruder to develop automated processes to counter the deception; a single test function would thus easily distinguish the real from the fake (Bowen et al., 2009). Examples include: anomaly-based detection, which can be employed against the fake file system, comparing a range of metrics against those of typical computer systems (Rowe, 2006), inspecting randomly-chosen files or directories to see if they look ‘normal’ (Fu et al., 2006), checking content against other nearby material to determine if it is out of place (Lavoie and Krishnamoorthy, 2010), or looking at content to identify unusual languages, uncharacteristic formats or text that comprise random words, rather than structured sentences (Voris et al., 2012).

Simulating realistic file repositories can be useful for a range of practical research purposes, such as generating test data to improve the performance of compression and retrieval in electronic storage systems. Undertaking research on genuine data is complicated by lack of access to real world corpora, primarily due to privacy or copyright concerns (Ming et al., 2014; Tarasov et al., 2012). Synthetic data that is more aligned with real world environments are likely to yield results that are more transferable to practical outcomes.

The research is also applicable to the forensic detection of fake files. The presence of fake files could indicate malicious or unauthorised activity. Successful detection of fake files may trigger further analysis or allow an examiner to discard these files as spurious.

METHOD

Rowe (2006) proposed an approach to detect fake file systems that is relevant to their construction. He believed that they could be detected by calculating statistics on a candidate file system and its subdirectories, and comparing these results with a typical real system; significant discrepancies suggest deception. This research follows Rowe's approach. The previous 30 years of published file system surveys were reviewed in order to identify a set of important characteristics that could be used to either construct fake file systems, or detect their presence. Only recognised peer-reviewed publications were included in the literature review.

PREVIOUSLY PUBLISHED FILE SYSTEM METRICS

There have been more than 15 published surveys of file systems since 1977. Table 1 summarises these major file system surveys, in chronological order of publication. Five other notable studies were conducted on files and documents, rather than file systems (Barford and Crovella, 1998; Bouayad-Agha and Kilgarriff, 1999; Chow, 2007; Cunha, 1995; Rowe, 2010). The relevant findings are grouped by metrics below.

Distribution by File Size

Stritter's (1977) initial observation was that the distribution of file sizes within his sample set could be approximated by a Pareto distribution. Soon after, Satyanarayanan (1981) identified that his sample distribution decreased nearly monotonically with increasing file size. He was able to use this result to approximate files by size based on a hyper-exponential distribution, using Kolmogorov-Smirnov (Hájek, 1967), a common approach to compare sample distributions against a reference probability. Nearly 20 years later, Barford and Crovella (1998) refined Satyanarayanan's results to create a hybrid model where the bulk of the file sizes were small, with a long tail of much less frequently appearing larger files; Satyanarayanan's approximation split the log-normal distribution of the body (93% of the data) and a heavy tailed distribution of the larger files (above 133kB in size), with a mean file size of 9.357 and standard deviation of 1.318. This result was supported by Douceur and Bolosky (1999), who while noting that the previous hyper-exponential distribution was not as natural a fit for their data, still preferred a log-normal distribution, with a Pareto tail.

Shortly after, Downey (2001) compared the performance of log-normal and hybrid Pareto distributions using data from Irlam (1993), Arlitt and Jin (2000), Satyanarayanan (1981), and Douceur and Bolosky (1999), also using Kolmogorov-Smirnov. He was unable to confirm that the previous distributions were long-tailed (i.e. Pareto), but he still identified that the data could be approximated more accurately by either a single or two mode log-normal with the distribution skewed towards smaller file sizes. He concluded that even if his model is not entirely realistic, it is robust to violations of the assumptions.

While distribution by file size may appear to be a common measurement of file systems, there are limitations for its application in generating synthetic content. Firstly, there is no consensus on a uniform model of distribution; some researchers were unable to match previous published file size results (Evans and Kuenning 2002; Vogels, 1999). There does appear to be consensus on the overall characteristics.

Secondly, several researchers observed that the average file size increased over time (Agrawal et al., 2007; Bennett, 1992; Douceur and Bolosky, 1999; Roselli et al., 2000; Sienknecht et al., 1994), which may account for the variance between surveys. The latter may require the development of a generic generation or detection metric, which can be adjusted periodically to account for technological change, such as the growth of virtual machine snapshots and multimedia. This variation may focus on the tail of the distribution elongating, but with the overall split between large and small file sizes remaining consistent.

Finally, file size metrics are heavily influenced by user activity; different file types exhibit divergent size distributions (Evans and Kuenning 2002; Satyanarayanan, 1981). For instance, a file repository supporting a graphics company is likely to have a different set of common file types compared with an elderly home user, both legitimate real world data. File size models may also be heavily influenced by the inclusion of the underlying operating system, rather than just user data. This might require the creation of multiple file size models, which can be applied depending on the mix of content that is required, with the operating system included as a constant.

Year Published	Researcher	File System Owner	Method	Number of File Systems	Volume of Sample Set
1977	Stritter	Production servers from a research organisation	Installed a program on three IBM 360 and 370 machines, and over the period of a year, collected a snapshot of the files system, and recorded file creations and deletions.	3	Unknown
1981	Satyana-rayanan	Production servers from a research organisation	Captured data from the eight 200-MB disk drives of a Digital PDP-10 running the TOPS-10 operating system. He recorded each file's size, timestamps, and type according to the three character file-name extensions allowable in the TOPS-10 operating system.	8	36,000 files
1984	Mullender and Tanenbaum	Production server from a research organisation	Captured file size data from a single Unix server to support the investigation into the design of a free-standing transaction-based server.	1	19,978 files
1991	Bennett, Bauer, and Kinchlea	Production server from a research organisation	Captured quarter-hourly snapshots, recording each file's size, timestamps, and type according to attribute flags.	3	304,847 files
1991	Baker, et al.	Production server from a research organisation	Analysed the user level file access patterns and caching behaviour of a Unix BSD file system. As part of their data collection they captured statistics on the file size and lifetime distributions.	1	Unknown
1993	Irlam	Various	Posted a message on Usenet asking UNIX system administrators to run a script on their machines and mail in the results. The script uses the find utility to traverse the file system and report a histogram of the sizes of the files.	>600	Unknown
1994	Sienknecht et al.	Production servers from a commercial business	Conducted a snapshot of file and directory data of HP-UX computer systems supporting 1845 active users.	267	> 2 million files
1999	Vogels	Individual file systems	Captured snapshots of file system activity from Windows NT systems over a period of four weeks. His snapshot data included system files and temporary web cache data activity in addition to user document repositories.	45	Unknown
1999	Douceur and Bolosky	Individual file systems	Collected and analysed files from the home directories of Microsoft Windows personal computers. Their survey included both system and user files. They developed analytical approximations for distributions of file size, file age, file functional lifetime, directory size, and directory depth, and compared these to previously derived distributions.	4801	> 140 million files
2000	Roselli, et al.	Various	Collected approximately one month of trace data from Unix and NT systems to understand how modern workloads affect the ability of file systems to provide high performance to users.	42	Unknown

2001	Evans and Kuenning	Individual and production server(s) from a research organisation	Collected data on media files as part of research into large files. *nix machines were collected by a shell script that used find (as root) to locate all files. A similar approach was used for the Windows systems. Directory and file names were discarded to maintain privacy, keeping only the size in bytes, type, modification date, directory depth, extension, and the number of blocks used by the file.	562	6, 156, 581 files
2002	Downey	Various	Examined data previously collected by Irlam, Satyanarayanan and Douceur in order to propose a user model that explains the shape of the distribution of file sizes in local file systems.	22	4, 489, 298 files
2003	Gonçalves, et al	Individual file systems	Installed a bespoke Python script on to the each of the individual file systems to collect the following data: (1) the size of personal document spaces, (2) the distribution of their contents, and (3) the directory tree topological structure. Information was not collected on individual files due to privacy concerns.	11	87, 340 files
2007	Agrawal, et al.	Individual file systems	A longitudinal extension of Douceur and Bolosky. Research was conducted, over the period 2000 to 2004. They collected snapshots of metadata from Windows desktop computers, comparing their results to the earlier study.	> 60, 000	> 5 billion files
2008	Leung, et al	Production server(s)	Examined two production, large-scale enterprise network file system workloads over a three month period. The research focused on: (1) changes in file access patterns and lifetimes since previous studies, (2) properties of file I/O and file sharing, and (3) the relationship between file type and client access patterns. Data was collected by copying and analysing the network traffic to and from the servers.	2	22 TB of data
2009	Henderson, et al	Individual file systems	A study of individual Windows XP file systems. Bespoke software was written and installed on each of the participant's file systems to record all open and close events, document creation, deletion, renaming, copying and moving. Each file system was monitored for between 1 - 5 days.	73	427, 050 files
2012	Meyer	Individual file systems	Collected file system content data over a span of 4 weeks. A file system scanner was installed on each workstation, autonomously collecting snapshots once per week.	857	162 TB of data

Table 1 – Summary of Previous File System Surveys

Distribution by File Age

Stritter (1977) was the first to model the age distributions, discovering an exponential distribution. He defined file age as the elapsed time since ctime. Douceur and Bolosky (1999) published the next significant distribution. They discovered that the median file age on their sample set was 48 days, three times the values collected by Stritter (1977), as reported by Smith (1981). Douceur and Bolosky (1999) were able to approximate their results using a 2-stage hyper-exponential distribution.

Douceur and Bolosky (1999) quantified the fit according to the maximum displacement of the cumulative curves. Unfortunately, this result failed both the Kolmogorov-Smirnov test and the chi-square test, which limits their application as governing distributions, even for their own observations.

Agrawal, et al. (2007) and Meyer (2012) believed that Douceur and Bolosky's (1999) previous results were still valid. Agrawal, et al. went further to add that: "since the distribution of file age has not appreciably changed across the years, we can expect that a prediction algorithm developed today based on the latest distribution will apply for several years to come". This is a promising result for synthetic file system generation.

There are, however, several limitations of employing previous file age distributions in a general theorem for sythentic file system generation. Firstly, the metric relies on ctime. As previously discussed, most modern file systems track three time fields for each file: (1) atime (2) mtime and (3) ctime. The latter can be confused with 'creation' time, but the ctime attribute can be modified other than at creation, such as a change in ownership (Rowe, 2010).

Secondly, file time attributes are unreliable due to: (1) variances in the way in which time stamps attributes are managed when copying files (Vogels, 1999); and (2) the file system routinely modifying these timestamps as part of normal system processes (searching files, anti-virus, etc) (Chow, 2007).

Thirdly, the distribution of file ages varies significantly across file systems and user job function. For instance, the results varied between files systems that provided collective repositories for academic research, as opposed to individual engineers working for a commercial entity. Potentially separate algorithms may be required for operating system files and those managed by users and organisations.

Finally, some researchers used trace data, allowing them to observe all file actions, whereas, the snapshot process (commonly used) missed: (1) creation, modification and access in-between the periodic snapshots; and (2) changes to the file system whilst scanning other parts of the file system.

Distribution by Functional Lifetime

Satyanarayanan (1981) was the first to introduce the concept of a file's functional lifetime (f-lifetime). He defined f-lifetime as the difference between the file's mtime and atime. That is the difference between the last access compared to the last modification of the file. He proposed that f-lifetime indicates the time span over which the data in the file has been demonstrably useful. He discovered that the distribution of files decreases nearly monotonically with increasing f-lifetime. This finding led him to find hyper-exponential distributions that approximate these two distributions. Mullender and Tanenbaum (1984), Bennett, et al. (1992), and Douceur (1999) also supported his findings.

Douceur and Bolosky (1999) noted a reduction in median f-lifetimes between their data set (12 days) and the original study undertaken by Satyanarayanan (30 days). Both these figures are exclusive of zero f-lifetimes, which can be the result of system processes, such as temporary Internet files (Rowe, 2010). They also observed that the distribution of f-lifetimes varies widely. On 50% of file systems, the median f-lifetime ranges from zero to 6 days, and on 90% of file systems, it ranges from zero to 97 days, reflecting the strong bi-modality. Douceur and Bolosky (1999) were able to approximate their distribution with a mixture of a constant distribution for zero f-lifetimes and a 3-stage hyper-exponential distribution. This result is similar in distribution to Satyanarayanan (1981), but with different parameters, which exposes potential for a general theorem.

Analysing and collecting data on file times is limited by the same challenges previously discussed in the distribution of files by age. In addition, the principle of locality may be important, which essentially states that (1) information in recent use is likely to be reused, and (2) information logically adjacent to recently used information is likely to be referenced soon (Denning, 1972).

Other Time-based Measurements:

Chow, et al. (2007) provided seven rules associated with attributes times, which are useful for individual file classification during forensic analysis. These include: (1) when mtime = ctime, the file has neither been modified since its creation, nor copied from another disk location; (2) when mtime < ctime, the file has been copied or moved; and (3) when a large number of files with 'close' atimes are found inside the hard drive, those files are likely to be scanned by some tool, e.g. anti-virus software.

Rowe and Garfinkel (2010) expanded on Chow's results. They grouped documents into three categories: (1) modified after creation (mtime > ctime), (2) accessed after creation (atime > ctime), and/or (3) accessed after modification (atime > mtime). They concurred with previous findings from Agrawal, et al. (2007) that: (1) files that were modified or accessed at least one day before creation (mtime < ctime or atime < ctime), suggest that the files were downloaded or copied from an external location; (2) clusters of these files in a directory have ctimes within a minute, suggest that they were downloaded or copied at the same time; and (3) the transfer of data is actually very common in user controlled file systems. Real-world files systems could contain evidence of all of the above cases.

Prior to publishing these results, Rowe (2006) suggested analysing file times to identify the mean within the day, week, and year to see periodic patterns. Expanding on this observation, it might also be useful to compare the file times against the normal working hours of the owner. For instance a commercial business might have the majority of ctimes between 0730 and 1930 Monday to Friday, whereas an individual home user times might primarily record activity in the evenings, after work.

Analysing and collecting data on file times is limited by the same challenges previously discussed in the distribution of files by age. Moreover, user activity is likely to heavily influence these results. For instance, individual users with a passion for collecting music or movie files may have a higher than average volume of externally created files.

Directory Size Distribution

Rowe (2006) noted that statistics on directories are less variable than those relating to files, and therefore present an important design consideration for fake file system generation. Several researchers (Agrawal et al., 2007; Bennett et al., 1992; Douceur and Bolosky, 1999; Goncalves and Jorge, 2003; Henderson and Srinivasan, 2009; Meyer, 2012; Sienknecht et al., 1994) reported statistics on the count of files by directory. Douceur and Bolosky (1999) were the first to present a distribution function for directory sizes. Their observations were relatively consistent across their sample file systems. Douceur and Bolosky (1999) were able to demonstrate that their directory size distribution fit offset inverse-polynomial distributions, which allowed them to develop an approximation with a mixture of a constant distribution for zero-size directories and an inverse-square. These results were supported by their extended study several years later (Agrawal et al., 2007).

Two interesting observations suggest that a general distribution function for directory sizes could be sustained. Firstly, Agrawal, et al. (2007) noted that the directory size distribution didn't change over their five-year observation period. Secondly, Henderson and Srinivasan (2009) remarked that there was no correlation between the number of files or folders a person manages and any of the demographic data collected (age, gender, academic or general staff status, department, position and/or employment tenure).

Four challenges may limit the wider applicability of directory size measurements: (1) there was significant variation between published means - results ranged from 2 to 13 files without a consistent value or pattern; (2) there was potentially a growth over time of the number of reported directories that did not contain files (but could contain directories). Agrawal, et al. (2007) reported that approximately 25% of directories were empty. This figure is up from 18% recorded by Douceur and Bolosky (1999), who surveyed a similar data set five years earlier. These results are higher than the 14% observed in large industry file servers by Sienknecht et al. (1994). (3) Vogels (1999) reported that there was no uniformity in size or content of files between individual user document repositories; and (4) Goncalves, et al. (2003) also observed that directories generated or managed automatically by applications tend to have large numbers of files, whereas users tend to separate and classify documents into sub-directories whenever possible. The latter result might be useful if a fake file system is required to mimic applications in addition to user data.

Sub-directory Distribution

Sub-directory distribution is the count of sub-directories in each directory (Sienknecht et al., 1994). Henderson and Srinivasan (2009) suggested a relationship between directory size and the depth. They observed that the average number of files per folder is greatest at the top levels of the tree (up to a depth of five), and then drops off sharply. Agrawal, et al. (2007) noted: "a slight downward trend in this ratio with increasing depth, punctuated by three depths whose directories have greater-than-typical counts of files: at depth two are files in the Windows and Program Files directories; at depth three are files in the System and System32 directories; and at depth seven are files in the web cache directories". This last result may not hold true for other operating systems.

Henderson and Srinivasan (2009) also discovered that on average 74% of folders in their samples did not contain any subfolders. They termed these directories 'leaf folder'. They also found that there were a high average number of subfolders at the root of the tree (~9), which sharply drops off (to < 2) by two or three folders depth. Douceur and Bolosky (1999) also observed similar results. They found that 69% of all directories contain no subdirectories, 16% contain one, and less than 0.5% contain more than 20. These percentages align with those reported by Sienknecht et al. (1994), who found 74%, 11%, and 1%, respectively. Douceur and Bolosky (1999) approximated their results using a mixture of constant distribution for zero-size directories and an inverse-cube distribution.

Goncalves and Jorge (2003) remarked that their directory count data resembled a Poisson distribution. The results from Agrawal, et al. (2007), the most comprehensive evaluation of directory distribution for individual users, also matched a Poisson distribution. This is an notable result for fake file system creation, however, this distribution is yet to be tested on organisational document repositories.

Other Noted Characteristics

Several researchers provided statistics on file type (including numbers of zip and gzip files), however; only Goncalves and Jorge (2003) provided a compression ratio (7%). They defined compression ratio as the number of text-based files with a corpus that have been processed by a compression algorithm within the file system divided by the total files in the corpus. Their compression ratio excluded images, audio and video content. It is possible that real-world file systems may contain a consistent data volume or number of compressed files within their contents; however, more work is required to develop predictable algorithm.

Henderson and Srinivasan (2009) defined duplication ratio as the number of files with an exact replica copy within the file system divided by the total files in the corpus. Bouayad-Agha and Kilgarriff (1999) reported that 15% of their documents had an exact replica copy in their corpus. Henderson and Srinivasan (2009) noted that 21.8% files had the same name as another file in their file system. While matching file name provides some level of confidence that the file could contain identical or similar content, a more precise measure may assess resemblance using such tools as fuzzy hashes.

Finally, there is a conspicuous absence of analysis on file content from the previous surveys. Partial text duplication arises for many reasons, such as: popular templates, the compulsory application of warning messages or classification banners, and organisational jargon (Bouayad-Agha and Kilgarriff, 1999). Plotting the proximity of content reuse may also be important. Its distribution is unlikely to be random. Similar data is likely to reside within the same or nearby directories, a characteristic of versioning and common users. It may be possible to identify consistent patterns, such as clustering of frequency of word pairs (n-grams), file naming conventions and common authorship identification through natural language processing that could assist in detection and more realistic generations.

CONCLUSIONS AND FURTHER RESEARCH

Simulating realistic hierarchical file structures is important for research, testing and cyber deception. Currently, there is an absence of metrics to guide the generation of synthetic file systems. This paper reviewed the previous 30 years of file system surveys in order to assess common metrics that could be applied to this problem. Ten potential characteristics were discovered: (1) file size, (2) file age, (3) functional lifetime, (4) directory size, (5) file origin analysis using time attributes, (6) file creation times against normal work hours, (7) directory size, (8) sub-directory distribution, (9) duplication ratio, and (10) compression ratio. Of these, directory distributions appear to provide the greatest potential, especially for individual file systems running Windows. There appear to be established trends in age and size distributions, but the specifics are far too dependent on the situation for a general theorem, without adjusting thresholds based on ownership (individual or collective), expected working hours and roles of the file system, and the employed collection method. The final sets of measures involving ratios are too immature without further experimentation.

Regardless of maturity, the composition of fake file systems should confirm with all identified real-world characteristics. Failure to consider one or more attributes could result in a simple detection process exposing the deception.

Future research should test these identified metrics against a range of individual and collective file systems to confirm algorithms and determine acceptable tolerances, particularly with a view to assigning variable thresholds for ownership and employment. Other algorithms could analyse file and folder names, and their conventions. One notable in these surveys was content analysis. Metrics could be developed to identify proximity and clustering of similar data content, authorship and vocabulary as a means of detecting data of different origin placed randomly within folders.

REFERENCES

Abbasi, A., & Chen, H. (2008). Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Transactions on Information Systems (TOIS)*, *26*(2), 7.

Agrawal, N, Bolosky, W J, Douceur J R, and Lorch J R (2007). A five-year study of file-system metadata, *in ACM Transactions on Storage (TOS)*, vol. 3, no. 3, p. 9.

Anquetil, N., & Lethbridge, T. (1997, November). File clustering using naming conventions for legacy systems. In *Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research* (p. 2). IBM Press. Arlitt M and Jin T (2000). A workload characterization study of the 1998 world cup web site, *in Network, IEEE*, vol. 14, no. 3, pp. 30–37, 2000.

Arnold R and Bell T (1997). A corpus for the evaluation of lossless compression algorithms, *in Data Compression Conference, 1997. DCC'97. Proceedings*. IEEE, pp. 201–210.

Baker M G, Hartman J H, Kupfer M D, Shirriff K W, and Ousterhout J K (1991). Measurements of a distributed file system, *in ACM SIGOPS Operating Systems Review*, vol. 25, no. 5. ACM, pp. 198–212.

Barford P and Crovella M (1998). Generating representative web workloads for network and server performance evaluation, *in ACM SIGMETRICS Performance Evaluation Review*, vol. 26, no. 1, pp. 151–160.

Bennett, J M Bauer M A and Kinchlea D (1992). Characteristics of files in NFS environments, *in ACM SIGSMALL/PC Notes*, vol. 18, no. 3-4, pp. 18–25.

Bouayad-Agha N and Kilgarriff A (1999). Duplication in corpora, *in Proceedings of the Second CLUK Colloquium*.

Bowen B, Hershkop S, Keromytis A, and Stolfo S (2009). Baiting inside attackers using decoy documents, *in Conference on Security and Privacy in Communication Networks*.

Buchholz, F., & Spafford, E. (2004). On the role of file system metadata in digital forensics. *Digital Investigation*, *1*(4), 298-309.

Carrier, B. D., & Spafford, E. H. (2004). Defining event reconstruction of digital crime scenes. *Journal of forensic sciences*, *49*(6), 1291-1298.

Chow K P, Law F Y, Kwan M Y, and Lai P K (2007). The rules of time on NTFS file system, *in Systematic Approaches to Digital Forensic Engineering, 2007. SADFE 2007. Second International Workshop on*. IEEE, 2007, pp. 71–85.

Cohen F (2000). A mathematical structure of simple defensive network deception, *in Computers & Security*, vol. 19, no. 6, pp. 520–528.

Cohen F, and Thomas E (2001). Red teaming experiments with deception technologies, taken from http://all.net/journal/deception/experiments/experiments.html.

Cohen F (2003). Leading attackers through attack graphs with deceptions, *in Computers and Security*, vol. 22, no. 5, pp. 402–411.

Cunha C R, Bestavros A, and Crovella M. E (1995). Characteristics of www client-based traces. Boston University Computer Science Department, Tech. Rep.

Daley R C and Neumann P G (1965). A general-purpose file system for secondary storage, *in Proceedings of the November 30–December 1, 1965, fall joint computer conference, part I*. ACM, pp. 213–229.

Denning, P. J. (1972, May). On modeling program behavior. In *Proceedings of the May 16-18, 1972, spring joint computer conference* (pp. 937-944). ACM.

Douceur J R and Bolosky W J (1999). A large-scale study of file-system contents, *in Proc. of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*. ACM Press, pp. 59–70.

Downey A. B (2001). The structural cause of file size distributions, *in Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2001. Proceedings. Ninth International Symposium on*. IEEE, pp. 361–370.

Ellard D, Mesnier M, Thereska E, Ganger G. R., and Seltzer M (2003). Attribute-based prediction of file properties, *Harvard Computer Science Group Technical Report TR-14-03*.

Erbacher, R. F., & Mulholland, J. (2007, April). Identification and localization of data types within large-scale file systems. In *Systematic Approaches to Digital Forensic Engineering, 2007. SADFE 2007. Second International Workshop on* (pp. 55-70). IEEE.

Evans K M and Kuenning G. H (2002). A study of irregularities in file-size distributions, *in Proceedings of the 2002 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*. Citeseer.

Fu X, Yu W, Cheng D, Tan X, Streff K and Graham S (2006). On recognizing virtual honeypots and countermeasures, *in Dependable, Autonomic and Secure Computing*, 2nd IEEE International Symposium on, IEEE, pp. 211–218.

Garfinkel S, Farrell P, Roussev V, and Dinolt G (2009). Bringing science to digital forensics with standardized forensic corpora, *digital investigation*, vol. 6, pp. S2–S11.

Gerwehr, S., & Glenn, R. W. (2003). *Unweaving the web: Deception and adaptation in future urban operations*. Rand Corporation.

Gerwehr S, Weissler R and Rothenberg J (2000). Employing deception in information systems to thwart adversary reconnaissance-phase activities, *in National Defense Research Institute Project Memorandum PM-1124-NSA*, RAND.

Goncalves D J and Jorge J A (2003). *An empirical study of personal document spaces*. Springer, pp. 46–60.

Hájek, J., Šidák, Z., & Sen, P. K. (1967). *Theory of rank tests* (p. 297). New York: Academic press.

Henderson S (2004). How do people organize their desktops? *in CHI'04 Extended Abstracts on Human Factors in Computing Systems*. ACM, pp. 1047–1048.

Henderson S and Srinivasan A (2009). *An empirical analysis of personal digital document structures*. Springer, pp. 394–403.

Irlam G (1993). Unix file size survey, available at http://www.base.com/gordoni/ufs93.html.

Kim, G. H., & Spafford, E. H. (1994, November). The design and implementation of tripwire: A file system integrity checker. In *Proceedings of the 2nd ACM Conference on Computer and Communications Security* (pp. 18-29). ACM.

Kushner D (2003). Digital decoys [fake MP3 song files to deter music pirating], *Spectrum*, IEEE 40(5), 27.

Lavoie A and Krishnamoorthy M (2010). Algorithmic detection of computer generated text, *arXiv preprint arXiv:1008.0706*.

Leung A W, Pasupathy S, Goodson G R, and Miller E L (2008). Measurement and analysis of large-scale network file system workloads, *in USENIX Annual Technical Conference*, vol. 1, no. 2, pp. 5–2.

Li, W. J., Wang, K., Stolfo, S. J., & Herzog, B. (2005, June). Fileprints: Identifying file types by n-gram analysis. In *Information Assurance Workshop, 2005. IAW'05. Proceedings from the Sixth Annual IEEE SMC* (pp. 64-71). IEEE.

McDaniel, M., & Heydari, M. H. (2003, January). Content based file type detection algorithms. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on* (pp. 10-pp). IEEE.

Meyer D T and Bolosky W J (2012). A study of practical de-duplication, *in ACM Transactions on Storage (TOS)*, vol. 7, no. 4, p. 14.

Ming Z, Luo C, Gao W, Han R, Yang Q, Wang L, and Zhan J (2014). Bdgs: A scalable big data generator suite in big data benchmarking, *in arXiv preprint arXiv:1401.5465*.

Mosteller, F., & Wallace, D. L. (1963). Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed Federalist Papers. *Journal of the American Statistical Association*, *58*(302), 275-309.

Mullender S J and Tanenbaum A S (1984). Immediate files, *in Software: Practice and Experience*, vol. 14, no. 4, pp. 365–368.

Ousterhout J K (1985). Da Costa H., Harrison D., Kunze J. A., Kupfer M., and Thompson J. G., *A trace-driven analysis of the UNIX 4.2 BSD file system*. ACM, vol. 19, no. 5.

Ritchie, O. M., & Thompson, K. (1978). The UNIX time-sharing system. *Bell System Technical Journal, The*, *57*(6), 1905-1929.

Roselli D S, Lorch J R and Anderson T E (2000). A comparison of file system workloads. *in USENIX Annual Technical Conference, General Track*, pp. 41–54.

Rowe N C (2005). Automatic detection of fake file systems, *in International Conference on Intelligence Analysis Methods and Tools*.

Rowe N (2006). Measuring the effectiveness of honeypot counter-counterdeception, in *39th Hawaii International Conference on Systems Sciences*, Poipu, HI.

Rowe N C and Garfinkel S L (2010). Global analysis of drive file times, *in Systematic Approaches to Digital Forensic Engineering (SADFE), 2010 Fifth IEEE International Workshop on*. IEEE, pp. 97–108.

Salminen, A., Kauppinen, K., & Lehtovaara, M. (1997). Towards a methodology for document analysis. *Journal of the American Society for Information Science*, *48*(7), 644-655.

Satyanarayanan M (1981). A study of file sizes and functional lifetimes, *in Proceedings of the 8th SOSP*. ACM, pp. 96–108.

Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, *5*(1), 3-55.

Sienknecht T F, Friedrich R J, Martinka J J, and Friedenbach P M (1994). The implications of distributed data in a commercial environment on the design of hierarchical storage management, i*n Performance Evaluation*, vol. 20, no. 1, pp. 3–25.

Smith A J (1981). Long term file migration: development and evaluation of algorithms, *in Communications of the ACM*, vol. 24, no. 8, pp. 521–532.

Somers, H., & Tweedie, F. (2003). Authorship attribution and pastiche. *Computers and the Humanities*, *37*(4), 407-429.

Spitzner, L. (2003). Honeypots: Catching the insider threat. In *Computer Security Applications Conference, 2003. Proceedings. 19th Annual* (pp. 170-179). IEEE.

Stoll, C. (2005). *The cuckoo's egg: tracking a spy through the maze of computer espionage*. Simon and Schuster.

Stritter E P (1977). File migration, Stanford Linear Accelerator Center, CA (USA), Tech. Rep.

Tarasov V, Mudrankit A, Buik W, Shilane P, Kuenning G, and Zadok E (2012). Generating realistic datasets for de-duplication analysis, *in Proceedings of the Annual USENIX Technical Conference, Boston, MA*.

Vogels W (1999). File system usage in Windows NT 4.0, *in ACM SIGOPS Operating Systems Review*, vol. 33, no. 5, pp. 93–109.

Voris J, Boggs N and Stolfo S J (2012). Lost in Translation: Improving Decoy Documents via Automated Translation, *in Security and Privacy Workshops (SPW)*, 2012 IEEE Symposium on, IEEE, pp. 129–133.

Whitham B (2013a). Canary Files: Generating Fake Files to Detect Critical Data Loss From Complex Computer Networks, *in The Second International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec2013)*, The Society of Digital Information and Wireless Communication, pp. 170–179.

Whitham B (2013b). Automating the Generation of Fake Documents to Detect Network Intruders, *International Journal of Cyber-Security and Digital Forensics (IJCSDF)* 2(1), 103–118.

Whitham, B. (2014). Design Requirements for Generating Deceptive Content to Protect Document Repositories. *Proceedings in the 15th Australian Information Warfare Conference*, *Perth, Australia.*

Yuill J, Zappe M, Denning D and Feer F (2004). Honeyfiles: deceptive files for intrusion detection, *in Information Assurance Workshop, 2004. Proceedings from the Fifth Annual IEEE SMC*, IEEE, pp. 116–122.