Digital Investigation 9 (2012) S131–S138

Contents lists available at SciVerse ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin

Testing the National Software Reference Library

Neil C. Rowe

U.S. Naval Postgraduate School, GE-328, 1411 Cunningham Road, Monterey, CA 93943, USA

Keywords:
NSRL
Forensics
Files
Hash values
Coverage
Accuracy
Extensions
Directories

A B S T R A C T

The National Software Reference Library (NSRL) is an essential data source for forensic investigators, providing in its Reference Data Set (RDS) a set of hash values of known software. However, the NSRL RDS has not previously been tested against a broad spectrum of real-world data. The current work did this using a corpus of 36 million files on 2337 drives from 21 countries. These experiments answered a number of important questions about the NSRL RDS, including what fraction of files it recognizes of different types. NSRL coverage by vendor/product was also tested, finding 51% of the vendor/product names in our corpus had no hash values at all in NSRL. It is shown that coverage or "recall" of the NSRL can be improved with additions from our corpus such as frequently-occurring files and files whose paths were found previously in NSRL with a different hash value. This provided 937,570 new hash values which should be uncontroversial additions to NSRL. Several additional tests investigated the accuracy of the NSRL data. Experiments testing the hash values saw no evidence of errors. Tests of file sizes showed them to be consistent except for a few cases. On the other hand, the product types assigned by NSRL can be disputed, and it failed to recognize any of a sample of virus-infected files. The file names provided by NSRL had numerous discrepancies with the file names found in the corpus, so the discrepancies were categorized; among other things, there were apparent spelling and punctuation errors. Some file names suggest that NSRL hash values were computed on deleted files, not a safe practice. The tests had the secondary benefit of helping identify occasional errors in the metadata obtained from drive imaging on deleted files in our corpus. This research has provided much data useful in improving NSRL and the forensic tools that depend upon it. It also provides a general methodology and software for testing hash sets against corpora.

Published by Elsevier Ltd.

1. Introduction

The U.S. government organization NIST (National Institute of Standards and Technology) provides an important resource for forensic investigators in the National Software Reference Library (NSRL), available at www.nsrl.nist.gov. Its primary resource is the Reference Data Set (RDS) which gives SHA-1 and MD5 hash values on the full contents of known files plus additional information such as file name, size, operating system under which it runs, and type of software product with which it is associated (White and Ogata, 2005). Such files are usually uninteresting to investigators and can be excluded in forensic analyses. The hash values can also be used to detect unauthorized copies of files and virus modifications.

Not much has been published evaluating the NSRL RDS. Since its primary contribution is the set of hash values and these are computed by well-tested code, these are likely to be accurate. There could be errors in the manual transcription of auxiliary information associated with the files. A more important question, however, is how much relevant data is missing from the NSRL. Although NSRL both purchases and receives donations to obtain a broad collection of software on which they calculate hash values, it is hard for them to be complete. The RDS does lack files

E-mail address: ncrowe@nps.edu.

1742-2876/$ – see front matter Published by Elsevier Ltd.
doi:10.1016/j.diin.2012.05.009

created dynamically by software since by policy it does not actually run the software when creating hash values. The current work investigated the significance of these potential weaknesses of the NSRL RDS. At the same time, it developed general software tools for evaluating hash sets.

2. Background

Digital forensics is challenged to handle greater and greater volumes of data. Data reduction by elimination of irrelevant data is increasingly important (Richard and Roussev, 2006), and the NSRL provides a start with its hash values. Concept-based search criteria provide more sophisticated kinds of filtering, but require careful tuning for a particular investigation (Hoelz et al., 2009).

Work building a "Korean RDS" suggests a more general approach (Kim et al., 2009). Their goal was to provide a version of the NSRL more relevant to Korean investigations with better coverage of frequently-occurring Korean files and less coverage of files that rarely occur in Korea. To do this, they eliminated irrelevant records (those with missing pointers, zero file sizes, references to temporary or installation files, duplicate information, etc.). They then analyzed a set of Korean packages to obtain additional Korean hash values. In our view, there appears no pressing need to reduce the size of the NSRL, as its data only required 8.4 GB in our experiments, a size easily within the capabilities of current hardware. There also appears no need to eliminate records with missing data, since partial information can still aid an investigation. However, the ideas of scanning the NSRL data to find errors and systematic checking of real-world files to suggest additions to the NSRL appear useful. That motivated the current work.

Investigators can of course obtain other hash sets. Bit9.com and www.hashsets.com, among others, sell hash sets but provide no testing data. Users can develop their own lists of additional includable and excludable hash values for their needs such as The Sleuth Kit's "Ignore Database" and "Alert Database" (www.sleuthkit.org). However, it would seem best to try to improve the NSRL RDS because a single standard free list would simplify investigations and better justify their results in court.

3. Setup of a corpus for testing

The experiments we conducted used only hash values for files plus the directory metadata of file systems since much can be learned about a drive quickly from this alone (Buchholz and Spafford, 2004). Experiments were conducted on directory information of 2337 drive images from the Real Drive Corpus (RDC) (Garfinkel et al., 2009) as of April 2012. These contained 36.0 million files of which 13.0 million were unique. 18.0% of the files were unallocated (deleted) with varying amounts of metadata. The drives were purchased as used equipment in 21 countries (Bangladesh, Bosnia, Canada, China, the Czech republic, Egypt, Germany, Ghana, Greece, India, Iraq, Israel, Mexico, Monaco, New Zealand, the Palestinian Authority, Pakistan, Singapore, Turkey, Thailand, and United Arab Emirates) plus a few from our research group. Some were computer disks (almost entirely Microsoft operating systems) and some were storage for portable devices, and the ages of files on them ranged from 0 to 25 years. They represent a variety of users and usages including business users, home users, and servers of several kinds.

Preliminary processing of the drive data is described in Rowe and Garfinkel (2011). File metadata was extracted including file path and name, file size, MAC times, NTFS flags (allocated, empty, compressed, and encrypted), and fragmentation status using our open-source Fiwalk utility. The hash values used in this study were SHA-1 since the raw RDS download is sorted by its values. Deleted files are especially interesting for investigators, and reconstructing them is definitely possible (Naiqi et al., 2008). So names and paths to deleted files are corrected when possible from modifications by the file system (appearing as underscores in the front or end of the file name) by reconstructing first and last characters of directory and file names using analogous names on the corpus; on our sample corpus, this permitted correction of 56% of the deleted file names with such symbols.

To make sense of a corpus, it is important to assign files to semantically meaningful groups like spreadsheets and programs; work such as Agrawal et al. (2007) demonstrates the benefits of this. 78 kinds of file groups were defined in three categories: for file extensions (like "htm"), for top-level directories in which the files occurred (like "Windows"), and for immediate directories in which the files occurred (like "pics"). It is valuable to classify files by directories as well as extension because files with the same extension like "txt" can serve very different purposes in different software. For immediate-directory names that are ambiguous, their parent directories are searched recursively to find one with an unambiguous meaning, something important with the increasing number of data "filestores" found in directory structures (Fisher et al., 2011). Currently 8224 extensions and directories are mapped into the 78 categories. Those that are insufficiently precise (like a top-level directory "new folder") are mapped to a default "miscellaneous" category. Currently all extensions and directories occurring at least 500 times in the corpus are mapped, and those occurring less than that are also assigned to "miscellaneous". 296 file extensions serve a sufficient diversity of function that they are ambiguous, and they are assigned to a special "multiuse" extension category. Mappings were built from laborious manual research.

The version of the NSRL RDS data used was the one available in August 2011. It was 8.4 GB when uncompressed. First experiments accessed it through a Web service to a local machine, but processing was slow. The current approach selects the needed columns from the RDS and collects all data for the same hash value together, reducing the size to 2.9 GB split into 69 separate files. This ran 80 times faster on the same corpus, so this approach was used for the experiments reported here.

4. Coverage of corpus files in the NSRL

NSRL matched SHA-1 hash values for 32.7% of the 36.0 million files in our corpus. Match rates did vary

considerably along several dimensions. Hash values were found in NSRL for 46.5% of the German files, 43.5% of the Canadian files, 32.6% of the Chinese files, 24.1% of the Mexican files, 23.0% of the Pakistani files, and 12.3% of the Israeli files. Table 1 shows the breakdown by file extension group, Table 2 the breakdown by top-level directory group, and Table 3 the breakdown by immediate-directory group. Some fractions are surprisingly low; executable files are recognized by NSRL only 64.8% of the time despite NIST's extensive software-collection program. Other fractions are surprisingly high, like the 68.0% for presentations; this probably reflects their occurrence in games and documentation.

5. Discrepancies in file names between the corpus and the NSRL

Files which are recorded in NSRL under a different name than that in the corpus could be interesting to the forensic investigator. It could indicate legitimate renaming of files by vendors. But it could also indicate illegal copies or attempts to conceal information. Also interesting are files with the same name but different contents, possibly bogus software or camouflaged malware.

5.1. Finding the best match for a hash value

One challenge in comparing file names between the corpus and the NSRL is that NSRL can have multiple entries for the same hash value; 2,078,143 of the 5,266,496 hash values covered in the NSRL RDS version used in these experiments have more than one referent. This occurs when the same file appears for different operating system versions or as part of different products, and the NSRL provides a separate entry for each. Then the best matching entry to that of a given corpus file must be found. A rating scheme was used based on a sum of differences in three things: the file name before the extension, the extension, and the operating system. Differences where one string forms the front of the other are considered less severe, and differences in just a few characters are considered less severe. Differences between upper and lower case were ignored. So for instance the file "setup.dll" under Windows XP could match "SETUP2.dl!" under Windows XP with a difference of one character for the file name, one character for the extension, and 0 for the operating system, for a total difference of 2. When there were ties between the best-scoring records, the most recent NSRL entry was preferred.

An issue with finding hash code matches is the currently 225,430 records for zero-size files in the NSRL, all of which have the same hash value. They must be stored separately and require exact matches on the file name to match to corpus files, for otherwise they slow processing greatly.

The operating system version for each drive in the corpus was inferred by collecting the best matching NSRL entries for each file on the drive using just their file name and extension. Assuming a file had K such matches with the best matching value, a weight of 1/K is added to the totals for each operating system version mentioned in those K matches. The operating system version with the highest total over all files is assumed. This approach reduces the chances of misidentifying a drive with legacy data from more than one operating system.

The NIST product codes (reference numbers of the software packages in which the files were originally found) were also tracked. The product code from the best match in the NSRL data for each file is used, assuming that just a single file for a product is sufficient to indicate use of the product. Drives averaged about 50 distinct product codes, not a large number.

5.2. Types of discrepancies between file names in NSRL and our corpus

NSRL-supplied file names were compared to their best matches in the corpus with the same hash value, and a program proposed possible explanations for discrepancies. Hash collisions are an unlikely explanation since the probability of even a single occurrence of one on our corpus when using SHA-1 is $2^{-160}*0.5*(3.6*10^7)^2 \approx 4.4*10^{-34}$. But other explanations are possible. Table 3 shows the counts of the best explanations calculated by our tool. Some further explanation of the categories:

- Some of the corpus file names are default names like "..".
- Some NSRL file names appear to be temporary or preliminary, such as "__0X01FC" for "00000025.query" in the corpus and "_ECE3E78EB29F498DAD23059-C0ED928E" for "erv43260.dll" in the corpus. Some NSRL names appear to have appended hash values, like "gray.pf.2b708bc3_5b4d_47c0_bcc5_3e1bd2c51e5b" for

Table 1
Fraction of corpus files found in the NSRL, grouped by extension.

Group	Fraction	Group	Fraction	Group	Fraction	Group	Fraction
No ext.	.038	Windows OS	.402	Graphics	.449	Camera image	.172
Temp.	.064	Web	.394	General document	.247	Micro. Word	.163
Presentation	.680	Database	.193	Other Microsoft Office	.667	Spreadsheet	.164
Email	.303	Link	.063	Encoded	.246	Help	.724
Audio	.125	Video	.282	Program source	.487	Executables	.648
Disk image	.346	XML	.482	Log	.037	Geographic	.136
Copies	.228	Dictionary	.154	Query	.404	Integer	.052
Index	.030	Form	.076	Configuration	.476	Update	.015
Security	.439	Known malware	.016	Map	.092	Multipurpose	.372
Directory	.503	Lexicon	.944	Games	.188	Engineering	.421
Science	.505	Signals	.088	Virtual machine	.135	Miscellaneous	.210

Table 2
Fraction of corpus files found in the NSRL, grouped by top-level directory name.

Group	Fraction	Group	Fraction	Group	Fraction	Group	Fraction
Root	.067	Deleted	.080	Windows	.644	Hardware	.331
Microsoft Office	.576	Documents and settings	.064	Programs	.384	Documents	.167
Temporaries	.225	Unix and Mac	.275	Games	.214	Miscellaneous	.158

"gray.pf" in the corpus. These names should be updated in NSRL.

- The NSRL name may have an extra file extension on the name used in the corpus, as for instance, "module1.xba.old" in NSRL and "module1.xba" in the corpus. This suggests that NSRL obtained an early or restricted form of software. The extra extensions should be deleted in NSRL.
- The corpus may have an extra file extension on the name in NSRL, for instance "canon-s600.xml.nodrv" in the corpus matching "canon-s600.xml" in NSRL. This suggests a different product uses a different name for the same software, a name that should be added to NSRL.
- The file extension alone may differ on an update, as "netapi.os2" in NSRL which matches "netapi.dll" in the corpus. This suggests additions to NSRL.
- One file name may have added numbers to designate a version number, such as "setup[2].dll" in the corpus for "setup.dll" in NSRL.
- One file name may have substituted underscores or exclamation points for the first or last character in the file name, e.g. "MDMGL004.IN_" in NSRL for "mdmgl004.inf" in the corpus, and "wz.pi!" in NSRL for "wz.pif" in the corpus. These appear to be associated with file deletions as discussed in Section 3. The many cases of these suggest that a significant number of the NSRL files collected for hash values are in fact deleted files. Computing hash values on deleted files could be risky for NIST as more than just the name of the file can be unreliable after deletion.
- The file name in NSRL may be misspelled. These were identified conservatively as cases where the NSRL file name was the same as the corpus file name except for a missing character, an added character, or a transposition of two adjacent characters. Examples are "dirtbike.bmp" in NSRL for "dirtbike.bmp" in the corpus, and "gnome-dice2.png" in NSRL for "gnome-die2.png" in the corpus. These appear to be errors in the NSRL.
- The file name in NSRL may be mispunctuated, as for instance "which_2.sh" instead of "which-2.sh" in the corpus. These appear to be errors in the NSRL.
- One file name may be an abbreviation of the other, as for example "EVTMGMT.MDZ" in NSRL matching "event management.mdz" in the corpus. These appear to be cases where NSRL had a preliminary version of software.
- The name in NSRL may be a singular when the name in the corpus is a plural, or vice versa. These appear to be errors in the NSRL.
- The name in NSRL may be a translation of a name in a foreign language, as "conejo.wmf" on a Spanish-language corpus drive matching "rabbit.wmf" in NSRL. These can be ignored. This is discussed more in the next section.
- Some hash values are for files serving multiple purposes under different file names. Some are default files and some are common graphics. See Section 7.1.
- Small files such as zero-size files may be accidentally identical to one another.
- The file may have been copied and renamed. This may be the case with some of the more radically different file names. When done with commercial software it may suggest theft; when done with sensitive information it may suggest espionage.
- The file name in the corpus may be in error. Evidence for this was more frequent on deleted files where partial garbage collection could have destroyed the true file name. See the discussion in Section 6.
- Someone may have manufactured a malware file to match a specific NSRL signature in the hopes of deceiving investigators into thinking it was an acceptable known file. This possibility has been discussed in the literature, but the space of SHA-1 hash values is sufficiently large that engineering such a file with a specific functionality would be very difficult. No evidence of this was found in our corpus.

Table 4 shows the counts on these discrepancies within the 11.8 million files of our corpus that matched hash values in NSRL. The categories are designed to be mutually exclusive, so the more general categories only apply to items to which the more specific categories do not.

5.3. Foreign language translation

Since our corpus is international, some discrepancies in file names are due to differences in languages. This did not

Table 3
Fraction of corpus files found in the NSRL, grouped by immediate-directory name.

Group	Fraction	Group	Fraction	Group	Fraction	Group	Fraction
Root	.058	Operating system	.453	Hardware	.579	Backup	.150
Temps.	.303	Help	.669	Visual images	.499	Audio	.068
Video	.185	Web	.412	Data	.201	Programs	.301
Documents	.348	Sharing	.310	Security	.072	Email	.301
Games	.115	Installation	.345	Applications	.382	Miscellaneous	.146

Table 4
Counts of file name discrepancies between the NSRL and our corpus.

Type of inconsistency	Count on corpus	Type of inconsistency	Count on corpus
Missing extension in NSRL	103,036	Missing extension in corpus	18,905
Additional 128-bit hash code on NSRL extension	19,959	Additional backup extension on corpus item	18,330
Other additional extension on NSRL item	2814	Other additional extension on corpus item	2311
More detailed extension on NSRL item	273,558	More detailed extension on corpus item	9835
Same file name with complex difference in extension	19,900		
NSRL file name has numeric addition	2932	Corpus file name has numeric addition	4983
Corpus file name same except has a bracketed number	16,536	Only difference in file names is punctuation	6102
Apparent misspelling in NSRL file name	5550	Period on end of NSRL file name	280
"__OX" code in lieu of a NSRL file name	169,708	Corpus file name is a placeholder like "."	1056
Plural in NSRL and singular in corpus	139	Plural in corpus and singular in NSRL	359
Foreign language version identical in contents to the English version	67	Foreign language in corpus, translation in NSRL	1873
Exclamation substitution at end of NSRL file name	42,267	Exclamation substitution at end of corpus file name	0
Underscore substitution at end of NSRL file name	351,989	Underscore substitution at end of corpus file name	4938
NSRL file name is abbreviation of the corpus	35,698	Corpus file name is abbreviation of NSRL	1600
Same extension but NSRL file name more detailed	117,292	Same extension but corpus file name more detailed	17,950
Match on files under 100 bytes	71,917	Files occurring with more than three names in corpus	19,706
Significant differences in both file name and extension	277,231		

occur as often as might be expected, as much software uses English file names even when it does not originate in an English-speaking country. Nonetheless, there were a significant number of translations in file names.

To check this, the Systran automated translation system was used. It covers most of the foreign languages in our corpus, most importantly the Chinese, Spanish, German, and Arabic words. The language is inferred as the predominant language of the country the drive comes from (only the country of origin is known). File paths were split into component words, sent to Systran for a translation, and results reassembled into a path. If Systran failed or returned the input, it was ignored. But there were 1873 file paths for which it found something, mostly Spanish to English. Systran is less successful on finding the proper syntax for translations, and permutations and rephrasings were not checked, so there were likely more acceptable matches than we counted.

6. Precision of the NSRL

An important question is the precision of the NSRL data, or the degree to which the data offered is correct. Hash values are unlikely to be erroneous, but file name, size, and product type could be.

To test errors in file names, all hash values were collected which occurred at least five times in the corpus but whose NSRL name never occurred in the corpus. Such cases are likely to be errors in file names because the space of SHA-1 hash values is so sparse that an incorrect one is unlikely to map to any files at all in the corpus. There were 43,384 such hash values for our corpus. There were simple explanations of some when the NSRL name was compared with the most popular name in the corpus. The NSRL name was the same except for a final underscore in 15,865 cases, the NSRL name was the same except for a final exclamation point in 3163 cases, the corpus name was embedded in the NSRL name in 903 cases, the NSRL name had additional characters in 1535 cases, and there were 111 additional minor differences. There were 4363 cases of the same extension with very different file names, which were probably alternative names for the file because the odds of an incorrect name matching in extension are small. There were 15,500 remaining cases where both the extension and file name did not match. These should be investigated as possible errors in NSRL file names, or at least as suggesting a new additional file name in NSRL for that hash value.

Another question is whether the file sizes given in NSRL are correct. So NSRL-reported file sizes were compared to those found for the same hash value in the corpus. They differed on 7461 corpus files, and usually by large amounts. This was a surprise was because the size of a file should be easy to compute in building the NSRL, and the chances of a file of different length mapping to the same SHA-1 hash value are vanishingly small. An example of a size discrepancy is the file test2.zeros of size 4096 in NSRL which had the same hash value as file AA00389B.71 of size 2,490,544 in the corpus. The possible explanations:

- This is a hash collision, unlikely given the size of the SHA-1 hash space.
- The hash values could be incorrect. This is unlikely since they were computed afresh for each file in both the corpus and the NSRL.
- Nearly all the NSRL file sizes in these cases were powers of 2, so NSRL may be measuring a block size containing the file rather than the file itself. File sizes that were powers of 2 occurred in only 2.2% of the corpus files with NSRL hash values, so this is cannot be a coincidence.
- The file sizes that Fiwalk retrieved for our corpus files could be incorrect. All the cases involved unallocated files in the corpus, which tend to have less reliable metadata. This explanation appears to be most likely.

Also investigated was whether there were possible errors in hash values given in NSRL by finding records in the corpus for hash values that never occurred in NSRL but

which had the same name as a file in an entry in NSRL. No such cases occurred. This was surprising because file names can be short and there was a definite probability of a false alarm at least. So there was no evidence of errors in NSRL hash values by this test.

Also checked was the product code information provided by the NSRL by comparing our own group classification of the files in the corpus (based on lowest recognizable directory in the path of the file) with a classification based on the NSRL product code. The latter was obtained by manually mapping the 887 distinct product descriptions in the NSRL to our groups. Only 12.3% of the group classifications agreed, from which one concludes that the data is too different to be directly comparable. For instance, Program Files/Microsoft Office/media/CntCD1/Photo1/j0180794.jpg is classified as an image file by our taxonomy but as applications software by NSRL. Both classifications are justifiable but not comparable.

NSRL also maps from product to product type. One particularly interesting NSRL product type is "Hacker Tool" which covers malware. To test its use, ClamAV antivirus software was run on a subset of 48 drives of our corpus, finding 6875 files containing signatures of malware. Looking these up in NSRL found 93 items (35 distinct) with NSRL hash values. In none of these cases did NSRL classify the file as a "Hacker Tool", and most were identified as "MSDN Library". So NSRL completely failed at recognizing malware on these drives. NSRL does not claim to be good at doing so, but users should not be misled by the presence of the "Hacker Tool" category in the NSRL classification. More importantly, the presence of what appear to be 93 virus-infected files with hash values in the NSRL RDS is disturbing. These could represent mistakes in identifying the virus file since the virus checker reported only the name of the file, not its path, and there were multiple files with the same name in these cases. This is a subject for future study.

7. Recall of the NSRL

An even more important question is the recall (coverage) of the NSRL, or the fraction of appropriate files for which it has hash values. This is somewhat subjective, since one can argue whether files only created after software is installed should be included, and if so, which. Updates, configuration files, and default documents would seem important to include, but not temporary files. NIST does not consider dynamically created files as part of its mandate, just files contained in the packaging of software. But a 32.7% success rate in matching files in our corpus sounds low considering how few files a typical user creates themselves.

7.1. Multidrive files missed by the NSRL

The 67.3% of the corpus whose hash values were not recognized by NSRL include a variety of files. Hash values that occurred on at least five different drives in the corpus provide relatively uncontroversial proposed additions to the NSRL hash values. They provided 166,113 distinct hash values. Some examples:

- The file "machine.inf" of size 103496 occurs 40 times in the corpus with the same hash value, under both "WINNT/inf" and "WINNT/ServicePackFiles/i386". Judging by the path and extension, this appears to be part of the Windows NT operating system missed by the NSRL.
- As mentioned in Section 5, the same file can serve multiple purposes, and this often happens with basic graphics files. For instance, the following names were used in the corpus for the same 56-byte GIF image: "BTN-DO ~ 1.GIF", "TB_SRH ~ 1.GIF", "DF_REV ~ 1.GIF", "ICON_A ~ 1.GIF", "RND03_ ~ 2.GIF", "vlog.idx", and "ESQUIN ~ 1.GIF".
- Another hash value of size 56 occurred 912,013 times in the corpus, usually under the name "." or ".." but in many different directories. This appears to be default contents for a basic operating system file.
- Cache files can be identical when the system state did not change between writes. If their contents originated from the operating system or the software, they may appear on multiple drives. An example is a hash value of size 10 that appeared under many names in our corpus including "A0021284.ini", "A0021290.ini", "A0022464.ini", "A0022478.ini", "A0022490.ini", etc., and usually under "System Volume Information" in Windows.

A side benefit to finding multidrive hash values in the corpus is that they can indicate metadata errors in our drive imaging process. For instance, "WINNT/inf/machine.inf" also occurred once listed as "WINNT/Fonts/Impact.ttf", a graphics file. The latter is unlikely to be correct since a graphics file cannot serve as an INF executable; probably the drive imaging extracted the wrong metadata for a deleted file. Such errors appeared on around 1% of the files with hash values occurring on five or more drives. However, such errors do not affect the validity for exclusion of the hash values themselves that have been found many times since they occur so rarely.

7.2. Similarly named files missed by the NSRL

Other candidates for possible addition of hash values to the NSRL are files that are closely related to files already in the NSRL. These may represent software versions that NSRL has not yet had an opportunity to analyze. To test this, all cases in our corpus of files with hash values not known to NSRL were extracted where a match or a variant on that name with a NSRL hash value appears in the same directory (though not necessarily on the same drive). This includes:

- Identical file names and paths, like file "OLETHK32.DLL" found in directory "WINDOWS/SYSTEM" on different systems with different hash values. These appear to be upgrades or patches and should be uncontroversial additions to the NSRL. However, they could represent fake software or modifications of software by malware, so antivirus software should be run on them. Such cases contributed 166,113 hash values from our corpus.
- Names identical except for extension, like "eng.xtr" and "eng.lex" found in directory "Program Files/Hewlett-

Packard/HP PrecisionScan/ISTech/OCR". These are very likely related; though they could be data specific to a user, most software would use a different naming scheme. These contributed 279,849 hash values from our corpus.
- Names identical except for numbers just before the extension, like "taunto11.wav" and "taunto13.wav" in the directory "Games/Age of Empires/sound". These appear to be parts of the same software package; perhaps the one without a hash value was added to the software after NSRL obtained a copy. Although caching often numbers files, if one file has a hash value in the NSRL, the other is unlikely to be a cache. These contributed 276,383 hash values from our corpus.

Altogether 937,570 additional hash values were found using both multidrive occurrences and similarities in names, enabling reduction of the corpus files for analysis by 2,559,749.

7.3. File-type filtering of files

The busy investigator may also want to exclude other kinds of files depending on the type of investigation. Our classification of files based on extension and directory permits this to be done automatically. Good candidates for such additional hash values are:

- Files with executable extensions like "exe" and "dll", which were 16.1% of our corpus. An exception would be investigations involving malware.
- Files in operating system directories such as "Windows" (30.1% of the corpus) or applications software directories such as "Program Files" (16.0% of the corpus). But this loses user-related information stored their such as preferences and activity history which could be important in an investigation.
- Configuration files (3.9% of the corpus by extension), since many of these are uninteresting defaults for a user.

7.4. Vendor coverage of the NSRL

Particularly useful for NIST to know are the applications software directories in our corpus with large numbers of files not in NSRL, since it suggests vendors and products that they should target to improve the completeness of their collection. To find this information, a tool was written to find all files in directories under "Program Files" in Windows, its variants and translations in languages of the corpus, and "bin" on Macintosh and Linux machines. It also sought vendor and product directories at the top-level of the file system like "python26" since some vendors put software there too, particularly on older machines; but this only considered directories recognized in the group mapping described in Section 3 as indicators of software, hardware, operating system, and game directories, of which there are currently 402.

The metric used was the fraction of files in NSRL for all files under those directories and their subdirectories, categorized by the inferred vendor/product name (the first directory under "Program Files" or its variants, or the top-level directory name if that was used). Table 5 shows a sample data for the corpus. Vendors with low fractions are prime candidates for attention by NIST. 51.1% of the software directories (1718 out of 3360) in fact did not have a single file with a hash value in NSRL, and 74.3% of them had hash values for less than 10% of their files, so the NSRL clearly has some gaps. Not all of these suggest action by NIST, as some of these files are old and unlikely to be encountered again, and some were data files. But the percentages are worrisome.

Table 5
Sample product/vendor fractions of files having NSRL hash values.

Fraction	Product name	Count	Fraction	Product name	Count
.000	mcafee security scan	252	.000	the bitmap brothers	4156
.000	videolan	185,455	.000	xunchitools	1734
.000	apple software update	2641	.000	media player classic	137
.000	informic	1171	.000	winutilities	549
.000	hypertechnologies	495	.001	webacus	871
.005	infograes interactive	12,142	.091	zone labs	439
.296	sonysz32audio	517	.538	macromedia	44570
.589	openoffice.org.2.2	4798	.798	microsoft plus!	2329
.900	norton utilities	178	.930	roxio	1268

8. Conclusions

Our analysis should give a clearer picture of the strengths and weaknesses of the NSRL RDS. Its precision is very good, but its recall is imperfect and users should be aware of that. Coverage was observed to be spread over a wide range of categories, and was not just confined to executables. There were a surprising number of gaps in the coverage of NSRL in regard to vendors, and 74% of vendors in our corpus were substantially uncovered. Coverage of malware was poor despite some misleading wording in the NSRL documentation.

Some errors and possibilities for improvement on auxiliary information of the NSRL were also found such as incorrect file names and unhelpful product descriptions. Comparison of our corpus with NSRL did pinpoint some errors in our own drive imaging, most obviously on file sizes.

NIST considers that its mandate is to compute hash codes from software downloads and media without running the software, but this is unreasonable in today's world of complex software installation procedures. This paper showed that two simple kinds of additions to the NSRL RDS could improve its coverage by almost a million hash values without any added work of acquiring, mounting, and analyzing files. While this is small fraction of the size of the RDS, few of these hashes could be obtained by NIST's traditional methodology of hiring interns to mount each software package and check it. Commercial hash sets like that of bit9.com may be a better source of hash sets for users who can afford them. However, our additional hashes are more consistent with NIST's

philosophy for NSRL of finding files that are likely to occur frequently on drives.

This analysis has resulted in much useful data that will be provided to the NSRL managers, and since the analysis is automated, it can be run periodically to track the NSRL RDS. The analysis software is general and will apply to any hash set that can supply associated file names and sizes. Both the software, part of the Dirim suite of tools, and analysis results are freely available.

Acknowledgements

The views expressed are those of the author and do not necessarily represent those of the U.S. Government. Thanks to Albert Wong and Riqui Schwamm.

References

Agrawal N, Bolosky W, Douceur J, Lorch J. A five-year study of file-system metadata. ACM Transactions on Storage 2007;3(3):9.

Buchholz F, Spafford E. On the role of file system metadata in digital forensics. Digital Investigation 2004;1:298–309.

Fisher K, Foster N, Walker D, Shu K. Forest: a language and toolkit for programming with filestores. In: Proc. ICFP, Tokyo, Japan; 2011. p. 292–306.

Garfinkel S, Farrell P, Roussev V, Dinolt G. Bringing science to digital forensics with standardized forensic corpora. Digital Investigation 2009;6:S2–11.

Hoelz B, Ralha C, Geeverghese R. Artificial intelligence applied to computer forensics. In: Proc. security, access, and privacy, Honolulu, Hawaii; 2009. p. 883–8.

Kim K, Park S, Chang T, Lee C, Baek S. Lessons learned from the construction of a Korean software reference data set for digital forensics. Digital Investigation 2009;6:S108–13.

Naiqi L, Zhongshan W, Yujie H. QuiKe: computer forensics research and implementation based on NTFS mfile system. In: Proc. Intl. colloquium on computing, communication, control, and management, Guangzhou, China; 2008. p. 519–23.

Richard G, Roussev V. Next-generation digital forensics. Communications of the ACM 2006;49(2):76–80.

Rowe N, Garfinkel S. Finding anomalous and suspicious files from directory metadata on a large corpus. In: 3rd Intl. ICST conference on digital forensics and cyber crime, Dublin, Ireland; 2011.

White D, Ogata M. Identification of known files on computer systems. Retrieved 17.02.12 from, www.nsrl.nist.gov/Documents/aafs2005/aafs2005.pdf; 2005.