A Forensic Image Description Language for Generating Test Images

Dr Gordon Russell Rich Macfarlane Robert Ludwiniak

Edinburgh Napier University
School of Computing, Merchiston Campus
10 Colinton Road, Edinburgh, UK
g.russell@napier.ac.uk, r.macfarlane@napier.ac.uk, r.ludwiniak@napier.ac.uk

Abstract

Digital Forensics is a fast developing job market, as well as being topical and interesting, and as such is an area in which University students are keen to develop and study. At Edinburgh Napier University this topic has been taught with flexible and distance learning students in mind, and to promote accessibility the practical exercises have been formed around the use of cloud-based technologies. This approach has highlighted a key issue, in that cost-effective cloud-based resources struggle to provide adequate CPU and IO capabilities to drive large simultaneous student numbers when performing forensic exercises on disk images created using physical disk acquisition techniques obtained from real systems. This paper considers the issue, and proposes a simplistic and easily reconfigurable image creation technique specifically designed to support digital forensic practical sessions.

Keywords: Computer Forensics, Digital Investigations, Digital Corpora, XML

Introduction

Many digital investigation process definitions have been defined, and can be summarised as the identification, preservation, collection, examination, analysis, and presentation of digital evidence [1]. The collection phase typically involves creating a bit level copy of media thought to contain potential evidence, which is commonly referred to as a forensic image. The various digital images typically created during a digital investigation can be from a variety of media devices, such as HDD images, USB images, mobile phone images, and RAM images. These images are necessary when teaching digital investigation examination and analysis practical classes.

When teaching digital forensics it is useful to have a range of suitable digital artefacts and images for students to perform analysis on. Varying complexity, while maintaining realistic images, is important to provide students with digital investigation scenarios which match up with the learning outcomes of the classes. The creation of the digital images also has to take into account the abilities of the various digital forensic tools which may be used to examine the artefacts, and the time each may take to perform analysis processes.

Using images of real media is one method of providing these artefacts to students for analysis exercises. Creating images from second hand media is one approach for creating these types of images, but it is well documented that this can raise privacy issues, and images could contain illegal materials [2]. Repositories of these types of images are available online, but typically are not fully documented and are difficult to tailor for specific analysis exercises. Another problem highlighted by researchers who have built repositories of real images if the problems with storage space due to the large size of the forensic corpora files [3].

Another approach is building these images by hand. This manual method is a time consuming and sometimes complex task. Hand-built images are available from online sources, but these tend to be aimed at particular tasks such as tool validation [4] and can be lacking in variety. For teaching purposes, images have to cover a range of specific investigative activities, but need to be small enough that the analysis can be able to be carried out in a reasonable timeframe by students. The traditional approach to manual disk image creation is to run a full operating system, either physically or in a virtual machine, and perform a number of tasks “live” within that system. Once complete, the machine is powered down and the physical disk is acquired as a disk image. This can then be processed using standard forensic tools. This method is advantageous as it allows full control over any aspect of the image development, and the sequence of activities performed on the media can be documented and compared with analysis results. This approach requires a large time investment, and if a range of unique but similarly complex images are to be produced for a class of students, this method is not feasible. Forming example images with many variations on a theme is a difficult process to perform by hand. The activities performed in the live machines to produce scenarios is hard to carry out in an automated way, as the instructions would tend to be “click on this link” and “run notepad.exe and enter the following text”.

When providing digital images in a teaching environment, it is also preferable to create unique artefacts for each student, or at worst per educational session. Using static images for teaching or training, with no variations between students or cohorts, can be problematic due to intra/inter-cohort collaboration [5]. Students within the same cohort may work together on analysis projects, and not get the benefit of completing entire pieces of analysis work individually. Another issue is that of images not being reusable as previous cohorts may pass on finished work to current students.

A different approach to creating images is to describe the image in a custom language, and then have an automated system read the description and produce the image on demand. Variants could then be produced simply by editing the description. There are already some forensic description languages published, such as the DFXML used in fiwalk [6] or the output of log2timeline [7]. However these currently focus on describing the output of a forensic analysis in complete detail, rather than offering an easily editable summary suitable for image creation and variation. This paper proposes an XML style specification language for disk image creation which allows summarisation of the details, allowing disk images to be rapidly created on demand. A simple application for producing the disk images from an XML specification is discussed in this paper.

Additionally, if near-identical images could quickly be produced containing variations on a theme, scenarios could be built to enable students to explore many variations of particular issues in a controlled manner. For instance, when exploring cylinder alignment issues, the ability to produce on demand different alignment variations of the same data could greatly improve concept understanding.

The proposed XML description language must support the definition of realistic images suitable for students learning about key digital forensic issues. This includes system files, user files, deleted files, hidden files, and application usage such as URLs visited in Internet Explorer, cookies set in Firefox, cache files generated in Chrome, and program execution history in file explorer. However the assumption should be that unspecified XML information should result in the automatic selection of appropriate values to match other specified criteria, thus maintaining the realism of the image while avoiding unnecessary definitions appearing in the XML. It should however always be possible to override automatic defaults where a particular artefact is needed. The XML should also allow interesting student challenges to be built, such as “find a file with a secret slack space message somewhere in partition 2” [8], which should assist in increasing student engagement when learning forensics.

Our digital forensic practical classes uses cloud-based virtual machines coupled with an integrated tutorial environment to support students with forensic analysis, which is offers a safe, reliable, and flexible learning approach to digital forensics [9] [10]. Analysing large images on such a platform takes time even on fast machines, and powerful cloud servers can be expensive to provide for large class sizes. This paper highlights the detrimental effect that long forensic-tool execution time has on the student learning experience, and thus it is useful to minimise such execution times. Therefore the image creation application proposed attempts to greatly reduce image sizes through various novel techniques, while maintaining a realistic partition structure and thus avoiding forensic tools highlighting any shortcuts as anomalies. The student experience of using such compact images in practical sessions is also compared to more traditional images.

A particular issue in our approach is the manipulation of windows registry hives. The windows registry has been found to be an excellent source of forensic evidence [11], but registry files have limited public documentation, so setting forensically significant registry values (such as the last accessed URL) reliable way can be challenging. A number of techniques are considered later in the paper.

The use of the new XML description language in a classroom setting is evaluated and observations presented. This considers a number of factors, including realism, efficiency, creation time, flexibility, and its impact on student engagement. Various forensic analysis tools are considered in this evaluation, including the CAINE environment [12] and Encase Forensic [13].

XML Specification Language

This paper proposes an XML-based description language for specifying problem scenarios within digital media. This Summarised Forensic XML language, SFX, allows scenarios to be specified in as few lines as possible, providing a simple but extensible definition language, which results in realistic images suitable for teaching purposes.

As well as SFX being able to specify disk and partition information, it needs to allow various operating system partitions to be built, and for scenario-specific files to be created. Some files may be marked as deleted, as might be the case in a real-world scenario. Additionally various user actions need to be specified, such as Internet use, running applications, and changes to system files.

At this stage of the research, the focus is on specifying image details which are particularly useful for the specific student practical exercises which match the learning outcomes for the programme. However, as XML is easily extensible there is no reason why our initial language could not be extended quickly to tackle new challenges.

As an example image specification, consider the SFX shown in Figure 1. Here a simple specification of an image is presented which gives a variety of partition and disk information suitable for a variety of different training exercises. This can be simply sent to our image generation application and the image will be created within a few seconds.

```
<disk size=”512M” alignment=”cylinder” diskid=”0x12345678”>
  <partition index=”p1” hidden=”0” size=”48M” type=”vfat”>
    <expand archive=”part1.zip” />
    <copy   from=”fake.dat” to=”/fake01.dat” />
    <copy   from=”fake.dat” to=”/fake02.dat” />
    <delete from=”/Thomas.jpg” />
  </partition>
  <partition index=”p2” hidden=”0” size=”48M” type=”ntfs”>
    <base os=”windows7x64” />
    <copy   from=”fake.dat” to=”/fake03.dat” />
    <copy   from=”fake.dat” to=”/fake04.dat” />
    <user   username=”Gordon”>
      <browserhistory browser=”firefox”>
         <url link=http://bbc.co.uk” time=”13:14:00 1 Jan 2013” />
      </browserhistory>
    </user>

  </partition>
  <partition index=”p3” hidden=”1” size=”64M” type=”ntfs”>
    <expand archive=”part3.zip” />
    <copy   from=”fake.dat” to=”/fake11.dat” />
    <copy   from=”fake.dat” to=”/fake12.dat” />
    <delete from=”/docs/image.exe” />
    <slackspace offset=”20” file=”/tomas.gif”
                message=”This is a secret message”
    />
  </partition>
  <partition index=”s1” hidden=”0” size=”144M” type=”ext3”>
    <base os=”fedora15x64” />
    <expand archive=”part2.tar” />
    <copy   from=”fake.dat” to=”/tmp/fake01.dat” />
    <copy   from=”fake.dat” to=”/tmp/fake02.dat” />
  </partition>
</disk>
```

Figure 1: SFX Example of s Problem Scenario

Having a specification language is a useful step, but it does not solve one of the key challenges. In real-world forensics an investigator would accept waiting minutes or hours for particular stages of an analysis process to complete. However students learning digital forensics are less patient, and in particular may have to attempt an analysis step many times due to errors made as part of the learning process. Thus for educational purposes the disk images used should be small and lightweight in terms of processing times, yet appear realistic to the student.

Another reason for focusing on short processing times for image analysis is that image analysis uses considerable CPU and IO resources. When such resources involve only the student’s own stand-alone PC, the actual usage is less important. However our courses are promoting the use of online learning environments, and the students can use a virtualised cloud-based environment to perform their forensic analysis. To make this environment cost-effective, each student has resources equal to a small fraction of a traditional forensic examiner’s PC. Therefore the images should ideally require low CPU and IO resources during the various stages of an investigation.

Squeezing Linux partitions

To lighten the overheads involved in forensically analysing operating system partitions the authors considered a number of approaches. It was felt that it was important to retain the file and directory structure of each operating system partition, but that the contents of most files in an operation system partition were irrelevant for our educational exercises. However it was felt that the files should not just be completely empty, so a compromise was investigated.

As a first step a Linux operating system partition was examined, namely Fedora 15. The allocated disk usage was 2.9GB for a web server install. There were 10610 directories, and 100950 other entries, of which 92303 were files. Truncating all files to 1 block (which by default is 4096 bytes) results in disk usage of 360MB. Of course there are still other overheads, such as inode information and directory entries, plus files you might not want to truncate, such as log files and user information.

The decision was taken that, rather than use files sizes truncated to 4096 bytes, the files would be truncated to the standard filesystem block size of 512 bytes. Although this does not reduce disk usage further within the disk image, it does make archives of the image much smaller (and these are needed to recreate the images on demand). The file information lost in this process did not seem useful to any forensic exercise undertaken by our students.

One option for saving even more space would be to reduce the filesystem block size in the operating partition to its minimum, which in ext4 is 512 bytes. However the goal was to avoid parameters which result in the appearance of a non-standard image, and disk block size was a factor which appeared in many forensic analysis exercises. Instead, further disk space was saved by replacing some files with hard links to similar files. In an attempt to be non-obvious, this is done only on directory branches with a depth of two or more. Links are always made to files held within the same directory as the link (in an attempt to keep some content similarity in each directory), and the replacement of files with links is determined on a random basis with a user-specified probability parameter. The probability used in our scenarios is considered at the end of the next section.

Squeezing Windows 7 partitions

A similar approach to managing Windows partitions was selected as that used in Linux partitions. Due to the nature of the NTFS drives available in Linux, NTFS partitions are more complex to manipulate than Linux partitions. There are many features of NTFS which are hard to manipulate in Linux. In this paper a workable procedure is implemented for handling NTFS, but extra work in this area could increase the image realism.

The base Windows 7 x64 image was 8.5G, with 108992 files and 18012 directories. Again by traversing a copy of the partition and performing random truncation, while preserving the registry hives of the system and the administrative user, the partition can be shrunk such that file share is reduced to 473M. This is for a cluster size of 4096 bytes.

To improve the NTFS imaging process a number of features should be considered for the future. In particular the NTFS extended attributes are not currently well handled in this design. NTFS attributes allow files to be marked as non-standard files, such as for *hidden* files, *compressed* folders, and *system* files. This information can be accessed in Linux using getfattr, e.g.

```
> getfattr -h -e hex -n system.ntfs_attrib_be /mnt/\$Recycle.Bin/
# file: mnt/$Recycle.Bin/
system.ntfs_attrib_be=0x00000016
```

This cryptic big endian bitmap [14] indicates the NTFS file attributes set. In this case this indicates HIDDEN, SYSTEM, and DIRECTORY. Using tar on this information is difficult, so it would have to be stored separately and reapplied after restoration. However, as this paper is focused on Caine tools (which use a tolerant NTFS driver which can ignore attribute inconsistences) and on practical exercises where the attributes are not relevant, this aspect of reimaging is considered future work.

Additionally file ownership handling in the NTFS 3g is a multistep process, requiring first that the image to be cloned is scanned for user SIDs, and then have these added to the target image by first mounting it, writing the file, unmounting it, then remounting it, before finally copying in the files [15]. This is handled via the *ntfs-3g.usermap* utility, operating on the raw device file for the image. In our case this command produces:

```
:nobody:S-1-5-21-2592120054-4195220499-4132615206-513
operator:operator:S-1-5-21-2798420391-3814106432-3213135885-1000
```

This allows us to define that the windows user “Default” is mapped to the Linux user “nobody” and the administrative login “theSystem” is “operator” in group “operator”. This is saved in the mount directory as “/.NTFS-3g/UserMapping”. The file would have to be edited to accommodate any additional users added to the base image to support specific scenarios, and thus the SIDs would have to be managed.

One final approach might be to replace tar archives with ntfsclone, which can produce a stream which can be stored and reapplied to a partition. This could then be sized appropriately using ntfsresize. However, in the current system, file ownership in NTFS is not covered in any analysis so maintaining NTFS ownership and extended attributes are also considered future work.

Registry and Browser Information

Not only do the disk images have to contain partition and operating system files, but they also have to be configured to match each SFX specification of the user’s activities. Initially this has been restricted to creating Internet history information and setting some key registry entries in Windows.

Browser history and cookie information can be accessed and updated relatively easily. Firefox related browsers store their information in sqlite databases, and these can be updated using sql INSERT statements via the sqlite3 command line interface. For internet explorer the process is more complex, but the file format itself is relatively well known and simple [16], allowing routines to be easily written to create the requested browser history.

Updating the Windows registry, for instance to show which applications a particular user has recently executed, is more complex than updating the browser history. The registry hive format is not completely documented, and thus managing the files is not straight forward. The authors are still researching this aspect of image creation, but the use of hivexsh [17] seems to be the most promising route.

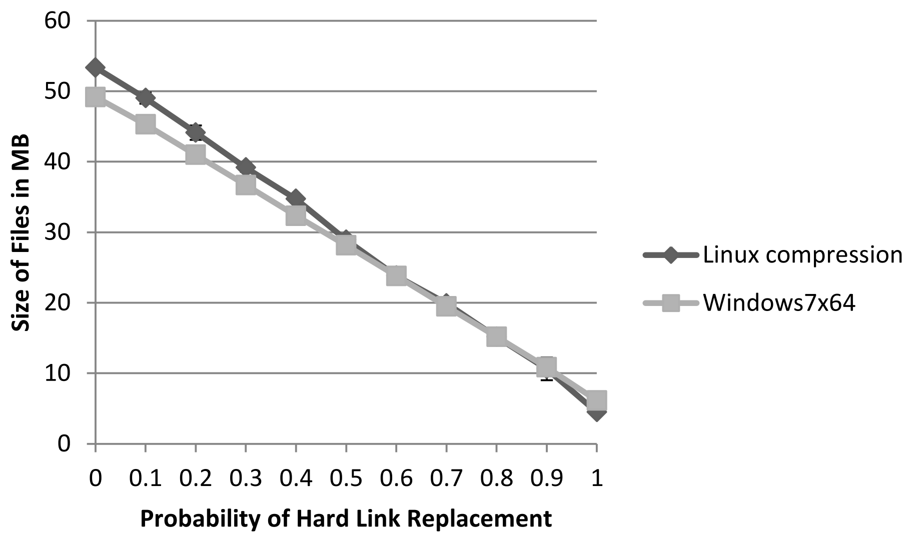

Figure 2: Fedora 15 image compressed size versus link insertion probability

Evaluation of Disk Usage Analysis

The windows 7 x64 and fedora 15 images were compressed using a variety of hard link replacement probabilities. The actual total file size (which ignores metadata overheads) in each case was plotted in Figure 2. As expected the relationship between size and probability is mostly linear. However the graph crossing suggests that in Windows there are more directories with fewer files in each directory, and as hard links only involved files within a directory the Linux image compressed more.

For use in the image creation application, the base fedora 15 image was produced with a 60% replacement probability, and the files tarred and gzipped, resulting in an archive size, including metadata, of 13MB. For Windows7x64, 60% was also selected as the replacement probability, resulting in an archive size of 24MB. Thus overall the archives are very small, and the image creation process takes only a few seconds.

Effect of Image Size on Teaching

Initially students were taught using full-sized image files, and this class is named *prechange*. In the following trimester a second class was taught similar material using similar practical labs, but this time the images were replaced with smaller more compact images. This second course is named *postchange*. The impact of this change on student engagement and achievement was one of the driving forces behind the SFX description language and associated application, and in this section the actual impact on the students is analysed.

In prechange there were 34 students, and in postchange there were 68. The two courses were not identical, but where they differ this has been considered in the analysis. As the LinuxZoo cloud-based learning environment was written in-house, and thus creates the opportunity to capture a wide variety of data on how students used the system. It is this data which has been used to form the analysis.

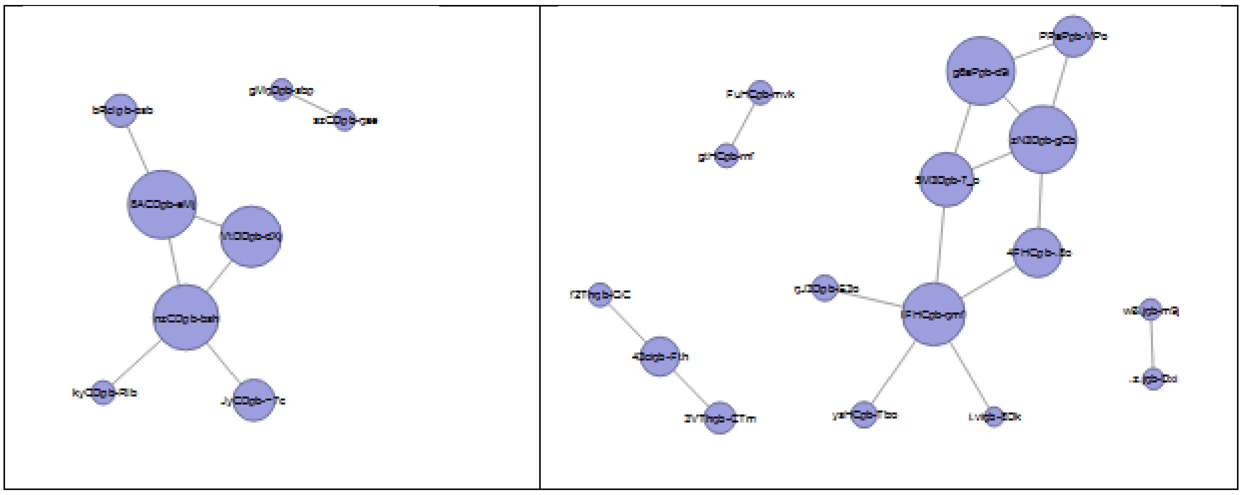
Figure 3: Peer support analysis for prechange (left) and postchange (right)

The first step in the analysis was to confirm that the student's perception of the course's overall difficulty level had not changed. The authors have developed a technique for measuring this, based on the fact that cohorts who find the course material challenging will form tight social groups as a coping measure [10]. Thus students who worked together on the material were identified by their similar IP numbers (e.g. sitting in a lab close together), the nature of the activity, and if this happened often enough outside timetabled events then the students in question were considered to have formed a support group. Figure 3 shows this relationship depicted in a spring model. Remembering that postchange is twice the size as prechange, both cohorts actually formed a similar number of similar sized groups. Thus, in both groups perceived difficulty in carrying out the analysis challenges was unchanged.

The next step in the analysis was to consider the time in hours each group put into the practical work. The author's suggest that large image sizes not only were slow to process, but the impact of a slow "try something and then view the result" learning cycle was further slowing the learning process. In particular written course feedback from prechange was that the cloud system was slow and unresponsive (even though it was actually operating correctly).

Figure 4 shows productive time spent on average by each student each week. It is partitioned into time within timetabled events (tuttime) and time outside such events (nontut). Productive time is measured where a student is productive if they click on something in the learning environment at least once every 5 minutes. As the tutorial system is integrated into the environment, this should happen normally if the student is actively studying.

It should be noted that, although the two courses covered similar material, they were assessed differently. In particular week 7 in postchange was a written test in the tutorial event, and thus 0 hours of that event were used online. In prechange there was a practical test in week 6, which perhaps explains the peak in week 5 nontut time, but does not explain the size and width of the peak. Both modules were assessed using linuxzoo in week 13, but clearly more non tutorial time was used by prechange.

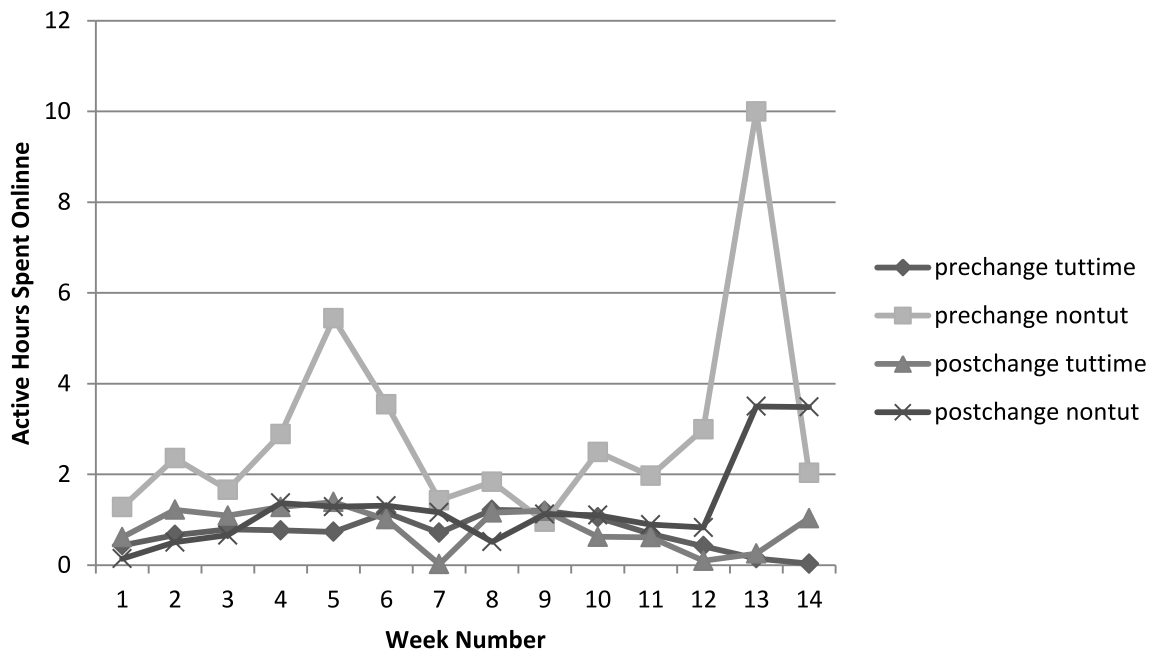

Figure 4: Time online in hours per week

Even taking into account differences between the modules, it is hard to ignore the significant additional hours outside of normal practical time which was used by prechange. The prechange students spent 16% less time in the practicals, but spent 129% more time outside the timetabled events working on the tutorials.

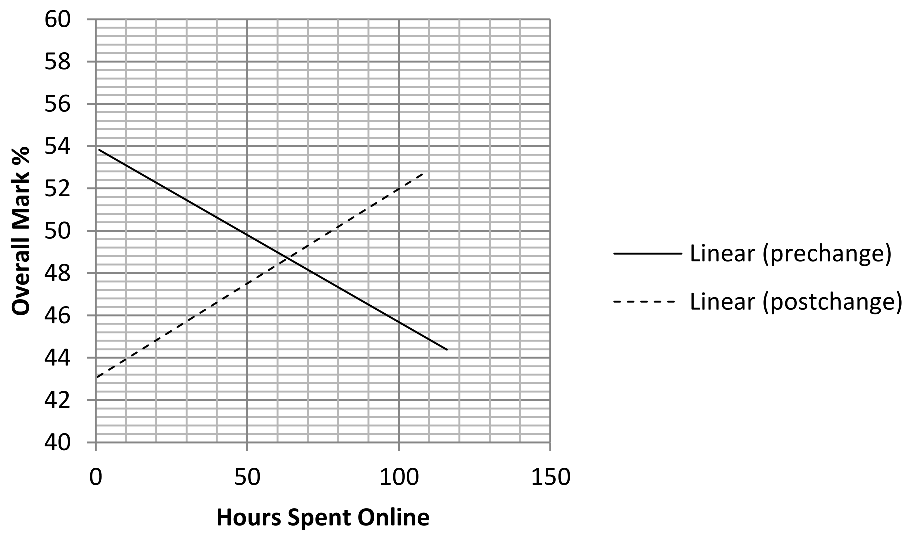

Figure 5: Trendline analysis of overall course performance

Figure 5 shows the linear trend lines for overall student performance in each course. Although there is naturally not a strong correlation, nevertheless the two courses are clearly different, with the postchange course showing the more desirable relationship of *more time spent, more marks received*.

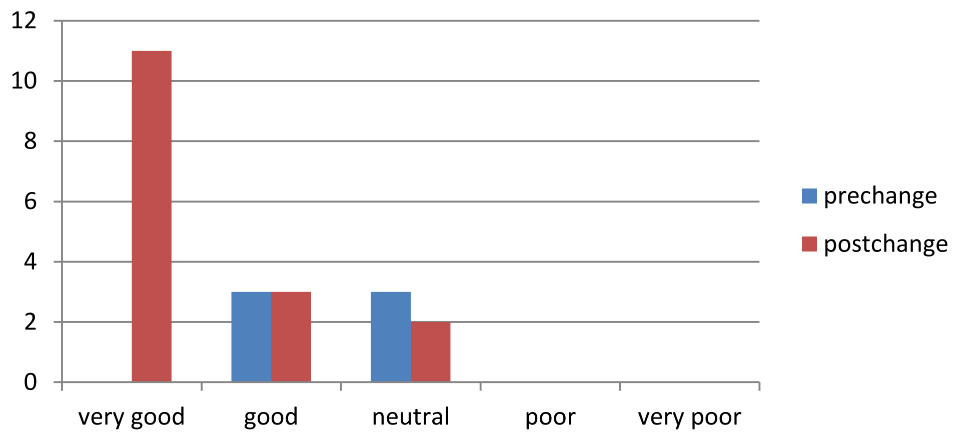

Figure 6: Student feedback for both student groups

Figure 6 represents overall student satisfaction in each course. The information was gathered through questionnaires, where students had a chance to comment on their learning experience on the module. The turnover of the feedback forms was rather low, only 17% of students from prechange and 23% of students from postchange group have returned the forms. 60% of the students from prechange group indicated that the practical sessions were too long and rather confusing. One student stated that long practical session did not enhance learning experience of the theoretical material provided through the associated lectures. The use of smaller images allowed the teaching team to provide greater granularity in delivery of the forensics concepts by simplifying some of the tasks performed by the students in the postchange group. The feedback from the students generally indicated a more positive learning experience. 50% of the students from postchange group had very positive experiences during practical session delivered via the virtualised learning environment and 30% of students indicated that the learning experience have enhanced their forensic knowledge and should help further their forensic career.

Conclusion

The use of a simple image description language offers a useful opportunity to dynamically and quickly produce disk images suitable for many training purposes. By having an automated process to produce disk images, there is also an opportunity to optimise the images in ways which can make them significantly faster to analyse, as well as easier to move between machines. This in turn should increase student engagement, and ultimately improve student performance on digital forensic courses. This paper proposes a simple description language, and offers a cohort analysis of the effect that carefully designed disk images can have on improving the educational process.

There is however much work still to be done in this area, including the generation of a range of images with similar complexity, and the configuration of additional aspects of a specific disk image. In particular there are challenges in creating some aspects of realistic browser history, as well as setting registry entries. Lastly, the manipulation of NTFS information from Linux tools remains a difficulty, but even here a number of solutions are possible in supporting Linux-based creation of realistic NTFS partition information.

References

[1] M. C. C. G. G. Reith, “An Examination of Digital Forensic Models,” in *International Journal of Digital Evidence*, 2002.

[2] P. F. V. R. G. D. Simson Garfinkel, “Bringing science to digital forensics with standardized forensic corpora,” *Digital Investigation,* vol. 6, pp. S2-S11, 2009.

[3] S. Garfinkel, “Lessons learned writing digital forensics tools and managing a 30TB digital evidence corpus,” *Digital Investigation,* vol. 9, pp. S80-S89, 2012.

[4] B. Carrier, “Digital forensic tool testing images,” Sourceforge, 2010. [Online]. Available: http://dftt.sourceforge.net/. [Accessed 15 July 2012].

[5] M. Jones, “A Digital Forensics Case Generator,” in *4th International Conference on Cybercrime Forensics Education and Training*, Canterbury, 2010.

[6] S. Garfinkel, “Digital forensics XML and the DFXML toolset,” *Digital Investigation,* vol. Volume 8, no. 3-4, pp. 161-174, February 2012.

[7] K. Guðjónsson, “Mastering the super timeline with log2timeline,” SANS Institute, 2010.

[8] E. Huebner, D. Bern and C. Kai Wee, “Data hiding in the NTFS file system,” *Digital Investigation,* vol. 3, no. 4, pp. 211-226, December 2006.

[9] G. Russell and R. Macfarlane, “Security Issues of a Publicly Accessible Cloud Computing Infrastructure,” in *Proceedings of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom-2012)*, Liverpool, 2012.

[10] G. Russell and A. Cumming, “Student Behaviour in a Flexible Learning Course Framework,” in *Proceedings of the IADIS International Conference on e-Learning*, Rome, 2011.

[11] H. Carvey, “The Windows Registry as a forensic resource,” *Digital Investigation,* vol. 2, no. 3, pp. 201-205, 2005.

[12] CAINE, “CAINE,” CAINE, 2012. [Online]. Available: http://www.caine-live.net/. [Accessed Aug 2012].

[13] G. Software, “EnCase Forensic,” 2012. [Online]. Available: http://www.guidancesoftware.com/encase-forensic.htm. [Accessed Aug 2012].

[14] J.-P. André, “Extended Attributes | Tuxera,” Tuxera, [Online]. Available: http://www.tuxera.com/community/ntfs-3g-advanced/extended-attributes/. [Accessed 5 June 2012].

[15] J.-P. André, “NTFS-3G: Building a User Mapping File,” [Online]. Available: http://b.andre.pagesperso-orange.fr/usermap.html. [Accessed 5 June 2012].

[16] “Internet Explorer History File Format,” [Online]. Available: http://www.forensicswiki.org/wiki/Internet_Explorer_History_File_Format. [Accessed 26 July 2012].

[17] R. W. M. Jones, “hivexsh - Windows Registry Hive Shell,” [Online]. Available: http://libguestfs.org/hivexsh.1.html. [Accessed 26 July 2012].