DIGITAL INVESTIGATION 6 (2009) S34–S42

available at www.sciencedirect.com

ScienceDirect

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

Digital Investigation

A second generation computer forensic analysis system

*Daniel Ayers**

Elementary Solutions Limited, PO Box 1756, Shortland St, Auckland 1140, New Zealand

Keywords:
Limitations of existing tools
Second generation tools
Tool architecture
Parallel processing
Tool metrics
Data design & abstraction
Forensic Workflow
Functional decomposition
Standardised tests

ABSTRACT

The architecture of existing – first generation – computer forensic tools, including the widely used EnCase and FTK products, is rapidly becoming outdated. Tools are not keeping pace with increased complexity and data volumes of modern investigations. This paper discusses the limitations of first generation computer forensic tools. Several metrics for measuring the efficacy and performance of computer forensic tools are introduced. A set of requirements for second generation tools are proposed. A high-level design for a (work in progress) second generation computer forensic analysis system is presented.
© 2009 Digital Forensic Research workshop. Published by Elsevier Ltd. All rights reserved.

1. Introduction

During the late 1990s the fledgling field of computer forensics was enhanced by the development of the first generation of dedicated forensic analysis tools. These tools facilitated convenient access to and review of evidential data in a forensically safe manner. Tools such as EnCase (Guidance Software Inc) and FTK (AccessData Corp) have become the industry standard tools for computer forensic investigation.

First generation (general purpose) computer forensic tools share a common architecture – application programs that execute on desktop computers, generally under the Microsoft Windows operating system. Although database systems, such as Oracle in the case of FTK, may be used for information storage a first generation tool executes on a single computer.

In the decade since the inception of first generation tools the limitations of this architecture have become apparent. Existing tools are failing to keep pace with the increasing complexity and evidential volumes of modern computer forensic investigations (Roussev and Richard, 2004).

In recent years researchers and tool vendors have proposed incremental improvements upon the first generation architecture. These improvements have focussed on increasing the computing capacity available so as to speed up forensic analysis. Roussev and Richard (2004, 2006) proposed a prototype system, *DELV*, that spread forensic processing workload across a commodity Beowulf cluster with evidence data stored on a central file server and in the main memory of cluster nodes. AccessData Corporation have announced a ''Professional'' version of their FTK 2 product that supports multiple processing nodes connected to a central database and analysis workstation.

The use of parallel processing to provide additional processing capacity is an important advance in computer forensic tools. However, this addresses only one of the significant limitations of first generation tools, and for that reason I describe such tools as ''Generation 1.5''. Other issues such as tool reliability,[1] auditability, data abstraction, efficient data storage and repeatability of results must also be addressed if computer forensic tools are to truly move into a ''second generation''.

* Tel.: +64 9 280 6351.
E-mail address: daniel.ayers@elementary-solutions.com
[1] Recently the Author documented several errors (Ayers, 2009) in the way the EnScript programming language and certain EnScript programs included with popular EnCase computer forensic tool handles dates and times in certain defined circumstances.
1742-2876/$ – see front matter © 2009 Digital Forensic Research workshop. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.diin.2009.06.013

Several metrics for measuring the efficacy and performance of computer forensic tools are introduced: Absolute and Relative Speed, Reliability, Accuracy, Completeness, Auditability and Repeatability.

Requirements for second generation computer forensic tools are proposed. A high-level design for a second generation computer forensic analysis system is presented. The design is a work in progress, but illustrates how some of the proposed requirements can be satisfied. The design aims to deliver:

- **Improved absolute and relative speed** through the use of distributed computing techniques, including Beowulf clusters, super computers, clustered file systems and (later) grid computing.
- **Greater reliability and accuracy** by adopting high integrity software development techniques borrowed from military, medical and aerospace systems.
- **Improved completeness, auditability and repeatability** using a formalised workflow framework to specify the sequence of analysis tasks to be undertaken and to record in detail the results of each analysis step.
- Improved human comprehension and productivity by presenting data at higher levels of abstraction than file system objects.

This work is part of an ongoing research effort to develop new computer forensic tools and techniques for the rapid analysis of computer evidence.

2. Metrics for computer forensic tools

This paper introduces several metrics for measuring the efficacy and performance of computer forensic tools:

- **Absolute speed** – the elapsed (wall clock) time required to complete analysis.
- **Relative speed** – the average rate at which the tool is able to process evidence compared with the rate at which data can be read from the original evidential media.
- **Accuracy** – the proportion of analysis results that are correct.
- **Completeness** – the proportion of forensic artefacts present in the evidence that are identified and reported by the tool.[2]
- **Reliability** – the proportion of tests where the tool executes successfully, does not crash or hang, and provides output in the documented format. (Note that this metric is not concerned with the accuracy of results.)
- **Auditability** – the proportion of results which are fully auditable back to original evidence data, including documenting all computations performed to derive results and all assumptions and other inputs (such as configuration information set by the analyst) that are capable of influencing results.
- **Repeatability** – the proportion of tests where it can be established, for example using detailed logs generated by the tool, that the process employed for analysis of an evidence item was exactly the process specified.

These metrics provide a framework for measuring the performance of existing tools against each other, against proposed new tools and in the context of theoretical limits on performance.

3. Limitations of first generation forensic tools

General purpose computer forensic tools, such as EnCase and FTK, provide a convenient and user-friendly environment for conducting forensic analysis. Prior to the inception of these tools, just over ten years ago, even simple analysis was difficult and frequently required hexadecimal-level examination of data and manual interpretation of file system structures.

These first generation packages are now the industry standard tools for conducting computer forensic analysis. They facilitate read-only access to file system contents using a familiar GUI-based user interface. Analysts are able to browse file contents, conduct keyword searches and employ a range of other analysis techniques.

Although the first generation tools have greatly enhanced investigators' ability to analyse digital evidence, in the ensuing decade the limitations of the tools have become apparent.

3.1. *Processing speed*

First generation computer forensic tools are struggling to keep pace with modern analysis workloads. Even when deployed on expensive high-end workstations with multiple processor cores, large amounts of memory and fast disk storage the ability of a single (even multi-threaded) application to quickly process evidence data is constrained.

Offloading the responsibility of data management to an external database system, which is the approach used by version 2 of FTK, is of marginal benefit with a single analysis system.

It is now common for analysts to face delays of several hours or even days when processing average volumes of evidential data. In some instances computer forensic analysis is conducted under urgency as part of a serious crime investigation where there is a very real risk to the safety of the public. There is an obvious need to improve the speed with which these investigations can be completed.

3.2. *Data design and I/O bottlenecks*

Most computer forensic analysis tasks, with the exception of password cracking, are I/O bound. The performance of a forensic analysis system will therefore be determined by the speed at which evidence data can be accessed. This is in turn influenced by data and algorithm design.

With first generation tools I/O optimisation is usually limited to selection and configuration of the fastest possible

[2] Completeness can be either relative or absolute, depending on whether or not the true number of artefacts present in the evidence is known. For test corpora the actual number of artefacts may be known, making an absolute measure of completeness possible. However in real investigations the only possible metric is relative – where tools are measured against each other, the tool finding the largest number of artefacts being defined as 100%.

disk storage systems. While analysts will take some care, and spend significant money, to install high performance disk systems such as multi-drive RAID and SAN configurations, this achieves only a modest improvement in data throughput.

The greatest improvement in I/O throughput will be achieved through efficient design of on-disk data storage and use of parallel processing. The on-disk data storage formats used by first generation tools have changed little in the past decade. Most tools use either a flat image file, such as those created by the UNIX tool *dd*[3] or by hardware devices such as the Voom HardCopy data capture unit, or a flat image encapsulated in a proprietary data format to store metadata and ensure evidential integrity.

EnCase uses flat image files[4] and re-parses the image each time the case is loaded. This can introduce significant delays when a new case is loaded or when recovering from a crash, particularly if large compound files such as PST archives or registry files have been opened. Furthermore, since evidential data is effectively retained in its raw format any fragmentation in the original evidence continues to slow down the parsing of evidence for the duration of the case.

Notably, FTK takes a different approach. Significant time is invested at the start of each case parsing evidential artefacts and, in the case of FTK version 2, storing information in an external database. This means the process of starting a new case is slow, but once that is completed analysis queries are fast.

3.3. *Software errors*

Software errors have troubled forensic tools for some time. The most common problems experienced by analysts are unexplained crashes that lead to disruption and, often, loss of work. Crashes of commercial forensic tools seem to be caused by a combination of factors:

- The requirement for the tool to be able to parse data that may be incomplete or corrupt, coupled with inadequate validation of input data.
- The use of relatively ''unsafe'' programming languages such as C and C++ where programming errors or unexpected data are likely to result in a software crash.
- Design errors in tools – including incorrect algorithms for interpreting evidential data.
- A lack of high integrity software development practices within the tool vendors.
- A desire, driven by commercial imperatives, to deliver new features to market as quickly as possible.

There have been incidents where forensic tools have been shown to capture or interpret evidential data incorrectly. The Author recently documented several errors in the way EnCase handles date and time values (Ayers, 2009).

Loss of productivity due to software crashes continues to be a significant concern for analysts and improvements to the robustness of forensic tools are required for this reason alone.

[3] Including *dcfldd*, the forensic derivative of *dd*.

[4] Including the EnCase proprietary evidence format, which is effectively a flat image file with metadata and integrity verification.

3.4. *Auditability*

The leading first generation tools, EnCase and FTK, are closed-source commercial applications. The ability of analysts to gain insight into their inner workings, how evidential data has been interpreted and the accuracy (or otherwise) of that interpretation is limited. Typically, analysts must rely on product documentation or information provided by the software vendor.

Tools rarely provide detailed logging or debugging information that would provide sufficient information to allow an analyst to validate the operation of a tool with respect to particular evidence. Often all the analyst can do is compare the output of several tools to ensure they deliver the same result.

In the case of open-source tools, analysts are free to browse the source code for the tool and even use a debugger to trace the operation of the tool in respect of particular evidence.

3.5. *Planning and control of analysis tasks*

First generation tools provide relatively poor support for detailed planning of investigative tasks and recording the detailed results of investigative processes. Much of this responsibility is left to the analyst – who will frequently use nothing more sophisticated than a pen and a notebook.

The diligence of analysts in planning the analysis process and recording results varies widely. But even the most diligent analyst is human, and subject to errors and lapses in concentration. Computer forensic analysis is a very complex undertaking so whenever the process is under manual control mistakes will be made and bias could be introduced, even inadvertently.

For example, an analyst may skip a particular analysis step that in the previous 200 cases has failed to yield any useful results. The analyst may believe that the step is of no practical use and unlikely to ever yield a result – but missing that step may cause important evidence to be overlooked in one case out of 500.

Auditability of forensic results is enhanced if detailed information is available regarding the results of each analysis step completed in an investigation.

EnCase also has limited support, in the form of bookmarks and the console log, for recording the results of analysis steps. But the software does not keep a comprehensive log recording each step in the analysis process. Likewise, FTK can highlight important artefacts within the evidence and store some analysis results – but this feature is not comprehensive.

3.6. *Automation*

Repeatability and robustness is improved if the detailed sequencing of analysis tasks is automated, following an investigation plan set by the analyst or as a matter of policy in the computer forensic laboratory.

EnCase has limited support for automated execution via the EnScript language and scripts, such as the Case Processor, supplied with the software. Analysts are free to develop and distribute their own scripts. But scripting is a partial solution at best as there are many EnCase functions that cannot be automated in scripts. Furthermore, the EnScript documentation is

sparse and does not provide sufficient detail for an analyst to develop scripts and be confident that the results generated will be as expected.

FTK has less support for automation than EnCase, lacking the sophisticated scripting facility. Analysts can control which of the available pre-processing steps are executed in each situation.

Consequently the operation of first generation tools remains a largely manual process. The requirement for a high degree of human monitoring and intervention slows investigations, increases cost and increases the likelihood of errors.

3.7. *Data abstraction*

Over a decade ago, first generation tools made it possible to review a captured file system in a forensically sound manner. This was accomplished by taking an image of the physical storage device and implementing a read-only file system parser within the tool. Analysts could then browse and search evidential data using a familiar GUI interface depicting files and folders.

First generation tools thus became focussed around the abstraction of a hierarchical file system. Despite the advances in forensic tools in the last decade, this has changed little. Most current forensic tools remain focussed at the file system level rather on higher level artefacts such as documents, email messages and images.[5] The notable exception is FTK.

Analysts must realise that the objective of a computer forensic analysis is not to locate *relevant files* but to identify *relevant evidence*. The analyst's ability to recognise and analyse certain types of relevant evidence can be improved if that evidence is presented at a higher level of abstraction than computer files.

For example email messages have the same basic attributes whether they are individual .MSG files, web based email in HTML format, contained within a PST archive or an .MSG email inside a ZIP attachment to an email stored in a PST archive. Tools should list email messages in a single list and display them in a uniform format, regardless of the particular data format in which they were found.

Several years after EnCase first appeared Guidance Software devoted significant effort to implementing parsers for common types of compound files – PST archives, Windows registry files, ZIP archives, etc. But the analyst had to manually find and open these files, or use a script to accomplish that task.

Opening a compound file could reveal other compound files, up to an arbitrary level of recursion. Instead of the software automatically locating and parsing compound files and presenting, for example, a global list of all email messages located on the system (in whatever format) this task was left to the analyst, possibly aided by scripts. The analyst was left to deal with all possible combinations of compound files, each combination requiring subtly different analysis techniques. At best this increased complexity and at worst relevant evidence was overlooked because it could not be easily located and examined.

[5] A single file – for example a PST email archive – may contain tens of thousands of documents, emails and images. The same document, email or image may be stored in different formats inside different files in the file system.

4. Requirements for second generation tools

This section defines the term ''second generation computer forensic tool'' by setting out a series of requirements that such tools *must* meet – plus optional requirements that tools *should* or *may* meet:

- **Parallel processing.** The tool *must* be able to use the computational resources of many separate processors (i.e. processors that do not share main memory or I/O bus bandwidth) so as to be capable of improved absolute and relative speed. The tool *must* be able to process data volumes that exceed the aggregate RAM of all processors by at least an order of magnitude. An automated method for distributing evidence data around processors and collecting results *must* be provided. The tool *should* be able to use processors with different operating systems and processor architectures. The tool *may* make use of distant ''grid computing'' resources providing that evidential integrity and confidentiality is maintained.[6]
- **Data storage and I/O bandwidth.** The tool *must* support a fault tolerant, high performance and scalable data storage medium so that analysts can implement a data storage solution to meet arbitrary capacity and throughput criteria.[7] The data storage medium *should* support a range of computer architectures and operating systems. The tool *should* store evidential data in a form that ensures it may be efficiently accessed.
- **Accuracy and reliability.** The tool *must* be designed and coded to provide a high level of assurance that analysis results will be correct and software operation free from error under all circumstances. Accuracy and reliability metrics must be 100% in all validation tests. The tool *must* validate all input data, including evidence data, and *must not* assume that any data input by a user or read from evidence is valid or free from corruption. The tool *must* trap all unexpected error conditions and *must* take all reasonable steps to recover from any error. In the event that the normal operation of the tool is interrupted due to software error, the tool *must* ensure that evidence data is not corrupted and that partial results or analyst work product is preserved and available when the tool restarts.
- **Auditability.** Source code for forensic analysis functions *should* be available for independent review by a qualified third party. Ideally this requirement would be met by making source code publicly available, although in the case of closed-source tools a detailed source code review and acceptance test by an independent auditor could be substituted. The tool *must* be able to generate a detailed log of all evidence parsing, analysis and searching activities. The tool *must* maintain an audit trail record of the actions of each analyst. The tool *must* be able to record and display to the analyst, in a convenient form, details of all

[6] This implies that remote grid nodes must be fully trusted and be reachable by a trusted communications medium, or that special techniques be adopted to allow the use of semi-trusted or untrusted nodes or communications media.

[7] So that analysts can design their storage solution to eliminate the central I/O bottleneck.

computations undertaken to produce any result together with details of any assumptions used in those computations and any other factor (such as configuration data or information provided by the operating system) capable of influencing the outcome of a particular computation. The tool *must* clearly identify which results are merely displayed and which are the result of computations or are influenced by configuration data or information provided by the operating system. It *must* be possible to trace each result and/or fragment of evidence back to the original raw data in an evidence file. The tool *must* ensure that the integrity of evidential data is provably maintained.

- **Repeatability.** The tool *must* support the automation of all analysis functions and processes, except those where interactive human involvement is unavoidable. The tool *must* provide a mechanism for analysts to codify a sequence of analysis steps to be undertaken automatically on a set of evidence. Dynamic automated modifications to the programmed analysis sequence *should* be supported.[8] The tool *must* provide a facility for results, documents or findings to be simultaneously reviewed by a group of analysts.
- **Data abstraction.** The tool *must* provide high-level abstractions for at least the following types of documents: (a) Generic text document; (b) Email message; (c) Picture or Image; (d) Video file; (e) Audio file; (f) Arbitrary binary data. Common data formats, including compound and archive files, *should* be parsed and instances of those file types converted into the generic form and listed in a single list. Listed documents *should* be identified using a convenient and unique numeric or alphanumeric identifier and *should not* be identified primarily by path and/or filename.

5. Design of proposed second generation computer forensic system

To illustrate how the requirements set out in the previous section may be met, and to demonstrate the benefits of doing so, early design work is underway on a proposed new second generation computer forensic analysis system.

The system will provide an environment within which analysts can store evidential data and conduct computer forensic analysis. It is up to the analyst to decide which analysis techniques should be applied, and how, to each investigation.

5.1. *Processing architecture*

Two different processing architectures will be implemented and tested.

5.1.1. *Beowulf cluster*

The prototype system will initially be implemented on a modest Beowulf cluster with approximately 50–100 dedicated processor cores and a similar number of ad-hoc processors available. This system is considered to be a realistic practical deployment for computer forensic analysts in law enforcement and consulting firms.

Initially all cluster nodes will run Linux, but at a later date software will be ported to Microsoft Windows so that additional ad-hoc cluster nodes (i.e. desktop workstations with idle processing cycles) can be used.

Experiments with a popular virtual machine environment are planned to ascertain the impact, positive or negative, of deploying analysis nodes as virtual machines within such an environment.

5.1.2. *IBM BlueGene/L super computer*

Another prototype system will be implemented on an IBM BlueGene/L super computer with 2048 processors. Each processor is a dual-core 700 MHz PPC CPU, giving a total of 4096 cores.

While few forensic analysts will have access to an IBM BlueGene/L super computer, and therefore it cannot be considered to be a realistic practical system, experimentation with forensic analysis workloads on the machine is expected to:

- Promote an understanding of how forensic workloads scale to very large computer systems
- Provide an opportunity to experiment with highly parallel analysis techniques that are only feasible on tightly coupled multi-processor systems
- Demonstrate the efficacy, or otherwise, of super computers for the large scale and/or urgent forensic analysis requirements of government agencies

5.1.3. *Distributed process scheduling*

The proposed system includes a workflow management capability (see below) that will be responsible for controlling which operations are scheduled on the cluster and maintaining an audit trail documenting each individual operation, its inputs and outputs.

The workflow manager will generate a queue of processing tasks for execution on the cluster. There may be interdependencies between tasks so execution order will matter in some instances.

The Condor (Litzkow et al., 1988) scheduling system will be used to manage the execution of tasks across available computing resources. Condor supports multiple architectures and allows tasks to be scheduled on nodes that meet certain specified prerequisites (such as amount of memory or disk space).

The proposed system will cache blocks of evidential data on the local hard disks of cluster nodes. For optimum performance of the cluster, a task requiring a particular block of evidence should by preference be scheduled to run on a node that has that evidence cached locally. Some minor customisation will be required of the Condor scheduler to implement this optimisation.

5.1.4. *Grid computing*

The Condor scheduling system includes support for ''Grid Computing''(Thain et al., 2005) where portions of a local cluster computing workload may be delegated to

[8] For example, if a password protected file is located new analysis tasks could be automatically scheduled to initiate password cracking attempts against the file.

geographically distant clusters operated by third parties. Grid computing provides an even greater pool of processing capacity, although there are some challenges if the technology is to be used for computer forensic processing:

- Remote nodes may not be trusted to maintain the integrity or confidentiality of evidential data.
- Remote nodes may not be trusted to deliver accurate analysis results.
- Communications paths between the local evidence storage and remote nodes may be inadequate to transfer evidential data (in which case remote nodes may only be able to be used for CPU-intensive workloads such as password cracking).
- Grid computing is so different to the normal way of conducting computer forensic analysis that it may be difficult to convince a Court that results obtained using that technique are reliable and therefore admissible.

It is not yet known whether or not it will be possible to overcome these challenges so that grid computing may be used for computer forensic workloads in the future. This question will be examined towards the end of the project.

5.1.5. *Data and results management*

The MySQL relational database management system will be used to store analysis results, workflow information and other data required to manage the operation of the cluster.

5.2. *Evidence storage*

Computer forensic analysis workloads are characterised by large volumes of data and the need for a very high data throughput. Analysis is often I/O bound, so any improvement to data transfer rates will improve overall system performance.

Traditionally evidence data has been stored on (in order of increasing performance) networked file servers, NAS, local hard disks or fibre-connected SAN. In a cluster environment some form of centralised storage is required, which implies a file server of some type.

Standard file servers are used by many forensic analysts, but are relatively slow even when accessed by only a single client due to network bandwidth constraints. When multiple clients are accessing the server at once, which is what might be expected in a cluster situation, performance degrades even further due to both network and disk contention.

5.2.1. *Cluster file systems*

The solution is to use a *clustered* or *parallel* file system where the abstraction of a single file system is, in reality, provided by multiple physical servers.

Data may be striped across the servers in the same way data is striped across hard disks in RAID-1 and RAID-5 configurations. A client reading file system data is thus receiving data from multiple physical servers at once, meaning the client's aggregated read rate can exceed the maximum disk or network I/O bandwidth of a single server. The performance of clustered file systems is greatly enhanced when both servers and clients use teamed network adapters to increase network bandwidth.

Several clustered file systems, including GPFS (Barrios et al., 1998), PVFS (Ligon et al., 1999) and Lustre (Sun Microsystems Inc, 2008) were examined and found to have broadly similar capabilities. Lustre was selected for our Beowulf implementation because it provides POSIX file system semantics, and thus is easily mounted, and is freely available.

The Lustre architecture comprises a small number (usually one or two) of metadata servers responsible for maintaining file system metadata, plus several or many object servers storing file system data. The capacity and performance of the file system scales with the number of object servers. The size of the file system can be dynamically increased through the deployment of new object servers.

The proposed system will use a central Lustre file system for evidence storage. Evidence data will also be cached on the local hard disks of cluster nodes.

IBM's GPFS is provided on the BlueGene/L system and will be used in that implementation.

5.2.2. *Data distribution and optimisation*

First generation computer forensic tools store evidence data in raw image files or proprietary evidence formats, which are usually raw image files thinly wrapped with metadata and information to prove evidential integrity.

While the raw image of an evidence item is considered to be the ''original'' version of the evidence and must be retained, it is often not the optimal format to store evidence in for later processing, because:

- The physical layout of files in the raw image will almost certainly not be optimal for traversal and searching by the analysis tool. Tools usually implement a tree traversal algorithm of some kind when processing files (e.g. for keyword searching). It is inevitable that the order files are read during traversal will not match the order in which they are stored in the image file. The situation may be exacerbated by some files being fragmented. Therefore the tree traversal implemented by the forensic tool will usually result in a sub-optimal access pattern into the image file.
- Evidential data must be distributed across, and locally cached on, nodes in the forensic processing cluster. It is not efficient to distribute a complete copy of the evidential image to every cluster node.
- Certain types of files in the raw image might contain evidential data that must be searched or examined, but is not stored in a readily accessible format. One example is Microsoft Outlook PST email files, which can contain tens of thousands of items but require significant processing effort to parse.[9] Other examples are compressed archive files and deleted files that must be recovered from unallocated space using file carving techniques.

Since computer forensic analysis is heavily I/O bound, there is strong incentive to improve absolute and relative speed by storing evidential data in the most efficient format –

[9] EnCase re-parses these files whenever a case file is reopened, often causing a lengthy delay.

even if that format is different to the original raw image file. The raw image must be retained as "original" evidence, and detailed records must be kept showing how the original evidential data was processed into the alternative form. But providing it is possible to establish that the integrity of the evidence has not been affected there is no practical reason why evidence data cannot be stored in the more efficient and convenient format. And the performance benefits in doing so may be significant.

The proposed system will iteratively process evidence data, starting with raw image files, into a form that is more convenient for efficient distribution across and analysis on a forensic processing cluster:

- **Level 0**. Original raw image file
- **Level 1**. Raw image (Level 0) with sectors re-ordered so that all files are contiguous and arranged in the appropriate order so that traversal of the file system can be achieved using a single linear read. An MD5 hash value will be computed and stored with each file to aid analysis and establish evidential integrity.
- **Level 2**. Level 1 image split into pieces of a convenient size for distribution across a forensic cluster (e.g. 100 Mb). Piece size may vary as no allocated file will be split across pieces. Optionally, the order of files may be rearranged so that pieces contain files of a similar type (for example so that all Microsoft Office files may be assigned to a cluster node specialising in that type of file).

5.3. *Workflow framework*

The proposed system will include a facility for managing the execution of forensic processing tasks, managing results and maintaining a detailed record of all tasks executed.

5.3.1. *Functional decomposition – analysis modules*

Functional decomposition is a widely used technique in software engineering where a large programming problem is iteratively broken down into smaller and smaller parts until each component is a simple programming problem with clearly defined inputs and outputs. It is one of the classic software design techniques for imperative programming languages.

Functional decomposition will have been employed by the developers of existing forensic tools to structure the code used to implement forensic processing steps "behind the scenes" within tools. It is proposed that functional decomposition be further applied as a means for analysts to manage forensic analysis tasks so that each analysis step is implemented as a single software module (which may in turn be decomposed further, yielding a subtree of software modules).

Each module would have defined inputs and outputs, and may have side effects that influence future modules (for example by passing data to them, or by arranging for additional modules to be executed). The output from one module may be used as the input to another. Modules may recurse and iterate over a list or subtree of files.

The details of every module invocation – including inputs, outputs and side effects – would be logged to provide an audit trail of forensic processing.

The analyst would specify at the start of processing which modules would be executed by arranging modules in a tree structure that is similar in semantics to a *Makefile* used by the "make" utility.

5.3.2. *Module libraries and standardised tests*

Modules may be reused and shared between analysts, and arranged into libraries. Open-source modules may be developed and standardised within the computer forensic community, leading to standardised tests and analysis techniques that are widely recognised and therefore more likely to be accepted in Court.

5.3.3. *Queues and manual review*

It is not possible to automate all types of forensic analysis. Some tasks can only be undertaken by a human, for example because special training or complex reasoning is required. Examples include the review of documents to assess relevance and privilege in legal discovery, or the review of images to assess whether or not they are contraband or in violation of corporate policy.

The proposed workflow framework will support FIFO queues where modules can add items, such as documents or pictures, into a queue where they can wait to be reviewed by a human. It will be possible for a team of people to be withdrawing and reviewing queued items. Reviewers may then classify items by assigning them into a new queue (where they may be the subject of further automated processing by a new set of modules) or exclude items from further analysis.

All actions of every reviewer will be logged for audit trail purposes.

5.4. *Software reliability*

Recognising the importance of reliability in forensic analysis software, the proposed system will be carefully designed and implemented so as to be as robust as possible in the face of corrupt evidential data and other unforseen circumstances.

Substantial research into software reliability has been undertaken in the aerospace, defence and medical industries. Software will be implemented in the Ada language, which is widely accepted as the language of choice for reliable software development. Extensive use will be made of software exceptions to handle, as gracefully as possible, any unforeseen software errors while minimising the risk of lost work or results.

6. Related work

6.1. *The DELV prototype system*

Roussev and Richard (2004, 2006) describe a prototype Distributed Digital Forensics system, *DELV*, implemented using a small Beowulf cluster of eight Linux nodes, a file server and a controller system interconnected by a high speed Ethernet network. A simple communications protocol was

used to control the distribution of evidence data and to instruct nodes to perform specific analysis steps.

Two keyword searches – one a simple string and the other a regular expression search – were used to compare the performance of DELV to that of FTK. DELV completed the string search 18 times faster than FTK, and the regular expression search 89 times faster.

A key design element of DELV is that evidence data distributed across the cluster nodes is held in RAM rather than being saved to local disk.[10] Therefore the performance improvement of DELV relative to FTK is due not only to the distributed architecture but the fact that DELV holds all evidence in RAM.

In the general case it is not realistic to limit the total size of evidence data to the sum of physical (or virtual) memory available on cluster nodes. Clusters are more likely to be used on larger investigations and the aggregate virtual memory available even in a larger cluster is small relative to the data size of large computer forensic investigations.

DELV demonstrates that even a modest Beowulf cluster can achieve a significant performance improvement when compared to a single workstation. However the observed improvement of 18–89 times is unlikely to be realised in the general case where local or centralised disk would be required for evidence storage. In that case the performance improvement factor could be expected to be closer to the number of nodes in the cluster.

6.2. *XIRAF*

Alink et al. (2006) describes XIRAF, a novel XML-based approach for managing and querying forensic traces extracted from digital evidence.

The XIRAF system includes a *feature extraction framework* that includes a repository of tools and a feature extraction manager. XIRAF's feature extraction framework is similar in some respects to the Workflow Framework proposed herein.

Importantly, XIRAF does include the concept that the system's model of the case may be iteratively extended through invocations of various tools from the tool library. This is analogous to the ability of analysis modules in the proposed framework to influence which modules execute in the future as a result of locating some type of artefact in the evidence data.

However, XIRAF's implementation using shell scripts and Xquery would not be suitable for use in a real world forensic tool. The system also adopts an unusual data model whereby evidence data is considered at the lowest possible level of abstraction (as ''BLOBs'' – Binary Large Objects) and some tools that parse evidence data are referred to as ''BLOB-extending tools'' because instead of their output being considered at a higher level of abstraction, the resulting data is simply appending to the low-level BLOB.

6.3. *Carrier – layers of abstraction*

Carrier (2003) discusses the role of abstraction layers in computer forensic analysis. He states that the appropriate level of abstraction depends on the skill level of the investigator and the investigation requirements.

Abstraction layers are modelled as having inputs, rules, outputs and margins of error. This model provides a useful framework for validating the novel abstraction layers discussed above (Level 1 and Level 2 images) and motivating the need for some evidence to be presented to the analyst at a higher level of abstraction than files within a file system.

[10] The authors plan to improve the system to remove this restriction.

7. Conclusion

Existing general purpose computer forensic analysis tools are rapidly becoming inadequate for modern analysis workloads. The outdated – first generation – architecture restricts their ability to scale and accommodate current and future analysis requirements.

In recent years researchers have cited the need for more capable forensic analysis tools, and proposed the use of Beowulf clusters to deliver greater processing capability. While this is a valuable advance, improving processing capacity alone does not address all of the key limitations of existing general purpose computer forensic tools.

This paper has proposed several metrics that may be used to measure the efficacy and performance of computer forensic tools. These metrics will allow existing first generation tools to be benchmarked against each other and against theoretical performance limits.(These tests are now underway as part of the Author's research project).

Deficiencies in first generation computer forensic tools were discussed. Requirements for second generation tools have been proposed. An early high-level design for a practical second generation computer forensic analysis system was presented. Design and implementation work on this system is ongoing. The system includes novel strategies for optimising and managing evidential data, managing and recording analysis steps, and proposes the use of super computers and grid computing resources for computer forensic analysis.

Once the proposed second generation system has been implemented, tests will be conducted to quantify its performance in terms of the metrics discussed above.

The value of developing standardised tests and analysis strategies, implemented in open-source and reusable software modules widely accepted by Courts and the analysis community, was reiterated.

REFERENCES

AccessData Corp. Accessdata website, http://www.accessdata.com.

Alink W, Bhoedjang RAF, Boncz PA, de Vries AP. Xiraf – xml-based indexing and querying for digital forensics. Digital Investigation 2006:S50–8.

Ayers Daniel. Errors in handling of dates and times in encase. Technical report. Elementary Solutions Ltd., http://www.elementary-solutions.com; 2009.

Barrios M, et al. Gpfs: a parallel file system. Technical report SG24-5165-00. IBM Corporation, http://www.redbooks.ibm.com; 1998.

Carrier Brian. Defining digital forensic examination and analysis tools using abstraction layers. International Journal of Digital Evidence 2003;1(4):2003.

Guidance Software Inc. EnCase Forensic Version 6. User Manual, Guidance Software Inc, http://www.encase.com; 2006.
Ligon Walter B., Robert Iii, Ross B. An overview of the parallel virtual file system. In: Proceedings of the 1999 Extreme Linux Workshop; 1999.
Litzkow Michael, Livny Miron, Mutka Matthew. Condor – a hunter of idle workstations. In: Proceedings of the eighth international conference of distributed computing systems; June 1988.
Richard III Golden, Roussev Vassil. Digital forensics tools: the next generation. Idea Group Inc.; 2006. p. 76–91 [chapter IV].
Roussev Vassil, Richard III Golden G. Breaking the performance wall: the case for distributed digital forensics. In: Proceedings of the 2004 Digital Forensics Research Workshop (DFRWS 2004); 2004.
Sun Microsystems Inc. Lustre file system – high performance storage architecture and scalable cluster file system, http://www.sun.com/software/products/lustre/docs/lustrefilesystem_wp.pdf; 2008.
Thain Douglas, Tannenbaum Todd, Livny Miron. Distributed computing in practice: the condor experience. Concurrency: Practice and Experience 2005;17(2–4):323–56.

Daniel Ayers holds Honours and Masters degrees in Computer Science from the University of Canterbury and is currently a part-time PhD candidate at that institution. He is the owner and Principal Consultant of Elementary Solutions (http://www.elementary-solutions.com), one of the major providers of computer forensic services in New Zealand, and founder of http://forensic-validation.com a global service dedicated to computer forensic tool testing and independent analysis of the reliability of forensic tools and methodologies. His research interests include event reconstruction, high performance computing and reliable software. Daniel has 26 years IT experience and twelve years experience giving expert evidence in Court. He has completed hundreds of computer forensic investigations during his career.