

Event-based Computer Profiling for the Forensic Reconstruction of Computer Activity

Andrew Marrington, George Mohay, Andrew Clark, Hasmukh Morarji

Information Security Institute, Queensland University of Technology
{ a.marrington, g.mohay, a.clark, h.morarji }@qut.edu.au

Abstract. In cases where an investigator has no prior knowledge of a computer system to be investigated, the significant investment of time and resources required to undertake a detailed computer forensic examination may deter investigators, given it is not known whether it will yield any relevant evidence. This problem is particularly acute in cases involving acceptable usage monitoring or intelligence operations, where an investigator has no particular expectations about the digital evidence which might be found on a collection of computer systems, or no prior knowledge of their usage. *Computer profiling* is a process by which a computer system is automatically examined, without direction, to determine whether the computer system is of interest to a human investigator. This paper proposes a new technique for automated computer forensic investigations which provides a computer profile with historical time-lining of user and application activity. A prototype software implementation of the technique is described and experimental results are provided and discussed which demonstrate the feasibility and value of incorporating activity traces into a computer profile.

Keyword. Computer profiling, digital forensics, digital evidence, event correlation.

1 Introduction

Computer forensics encompasses the examination and analysis of computer-based evidence as part of an investigation of a crime or suspect behaviour. Most computer forensic activities are conceived of as part of an investigation when there is already a hypothesis about the crime or suspect behaviour and the protagonists. Existing forensic tools and approaches are focussed on assisting an investigator who already knows, in general terms, what he or she is looking for. These traditional computer forensic tools and approaches are of limited utility in the case of investigations in which a computer system is being examined with no prior knowledge of its usage or its users.

Computer profiling is the forensic reconstruction of a computer system for the purpose of characterising its behaviour and usage, providing for the identification of computer systems of interest. Such a forensic reconstruction is conducted without any

prior knowledge of the computer to be profiled. [1] describes the extraction of information from a filesystem and the extraction of logs and meta-information to identify a static lexicon of objects representing the logical components of a computer (files, users, applications, and so on). This paper describes how the dynamic behaviour of such a system can be characterised by extracting activity traces for each object, allowing for the reconstruction of complete user sessions and the history of the computer system as a whole.

Existing forensic tools have limited functionality to support computer profiling as a distinct, automated activity. We discuss these tools and their limitations in section 2, and identify event correlation as a possible solution to address those limitations. The design of our system and our research methodology is discussed in section 3. We go on to describe, in section 4, the implementation of a proof-of-concept prototype, its capabilities, and our experimental results using that software. We discuss our results in some detail in section 5, and discuss some limitations of our approach. Finally, we finish by drawing conclusions and identifying future work in the field of computer profiling in section 6.

2 Related Work

2.1 File Analysis Forensic Tools

Some of the best known computer forensic tools, such as EnCase [2], the Forensic Toolkit (FTK) [3], and The Sleuth Kit (TSK) [4], are basically file analysis tools. They can be used by a forensic examiner to analyse individual files, as well as to discover deleted and hidden files on a target file system. They can be used to examine an image file of the target file system or to navigate through the file system's directory structure and conduct a search for evidence of the crime or suspicious event being investigated. They can be used by a forensic examiner to conduct a directed interactive search of a computer system in order to uncover evidence of a suspected crime (or other event of interest). File analysis tools can be used to discover deleted and hidden files of all types, including text and graphics files hidden inside other files. These tools also allow an examiner to search the contents of a disk for suspicious keywords and for specific file-types, thereby identifying evidence of illegal activity.

File analysis tools like EnCase, FTK and TSK were all designed to facilitate an exhaustive interactive search of a computer hard disk. However, as hard disk sizes increase dramatically, an exhaustive interactive search may be too time and resource intensive to be practical. Recognising this fact, Beebe and Clark proposed the application of data mining techniques to reduce human processing time of large datasets [5]. Such techniques could be incorporated into a file analysis tool, or perhaps into a computer profiling tool such as the one we describe here.

The functionality provided by existing file analysis tools is distinct from the functionality provided by a computer profiling tool. A computer profiling tool conducts an automated examination of the target computer system and present its findings to a human investigator. On the basis of those findings, the human investigator may decide whether or not the computer in question may be of interest to

their investigation and therefore warrant a more directed and detailed interactive investigation with a tool like EnCase, FTK, or TSK. The roles of the two types of tools are distinct – file analysis tools are used to gather evidence in support of some hypothesis to be presented to meet some burden of proof (for example, in a court of law), whereas computer profiling tools are used as an investigative aid to assist in the formulation of an hypothesis about the computer under examination.

2.2 Computer Profiling

Computer profiling is a new systematic computer forensic activity for automatically identifying computer systems of interest [1]. It can be described as the automated forensic reconstruction of a computer system for the purpose of characterising its behaviour and usage. Such a process is worthwhile in scenarios where investigators obtain a computer system with no specific knowledge of a crime or event to investigate, and want to learn about its usage. Rather than commit significant human and technical resources in a full-scale manual investigation of the system, investigators in such a scenario would employ an automated computer profiling tool. This tool may then be used to determine whether the computer system in question warranted such an interactive investigation, and provide some context and direction for such an investigation.

A practical computer profiling software tool needs to gather information from a wide variety of different data sources, and to employ a variety of techniques to assist in its data gathering. It incorporates a suite of modules designed to examine the file system and individual files with a similar level of detail to file analysis forensic tools such as those discussed above. Additionally, it incorporates a suite of modules designed to extract meta-information about files, applications, and users, in order to facilitate automated decision making about links and relationships between them. Known file filter (KFF) technology is employed not simply to eliminate so-called “uninteresting” files, as file analysis tools currently do, but to identify and categorise files. Advanced implementations can employ datamining, as advocated by Beebe and Clark, in order to improve the effectiveness and quality of the data analysis of the contents of a target computer’s filesystem [5]. Marrington et al undertook a prototype implementation of a computer profiling tool, which aggregated the output of external tools originally built to extract a very specific type of information, especially about files and users, but did not incorporate KFF technology, formal datamining techniques, nor functionality to reconstruct an activity timeline for the computer system being examined [1].

Integral to the computer profiling process is the automated identification of the logical components of a computer system and the classification of those components according to an object model. Objects in the model correspond to identified logical components, and have a type which is part of a hierarchy of types. This approach is similar to the approach employed in object-oriented programming. The National Centre for Forensic Science has advocated the application of an object-oriented paradigm to the representation of digital evidence, in order to facilitate both the description of digital evidence in a logical form and to enable the development of extensible schemas. To this end, the Digital Evidence Markup Language (DEML) was

developed [6]. In the computer profiling process the usefulness of an object-oriented approach extends beyond logical representation to automated reasoning. The object model proposed by Marrington et al classifies objects as belonging to one of four categories – *Application*, *Content*, *Principal*, and *System*. These categories are defined by broad super-types, and most objects will actually be instances of narrower sub-types (as can be seen in Figure A).

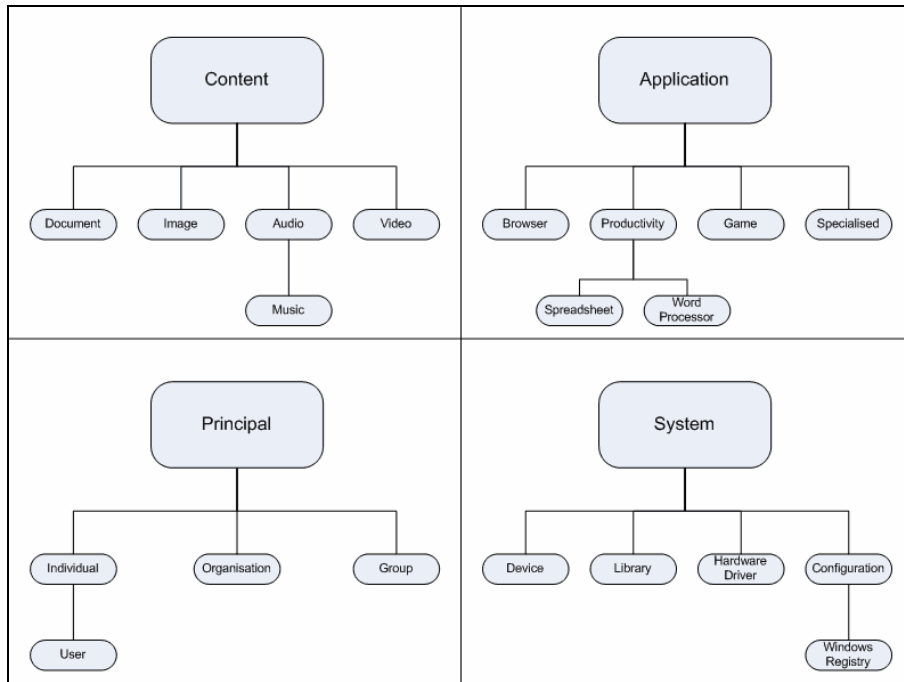


Fig A The four categories, defined by the super-types, with example sub-types becoming more specific.

After creating objects to represent the logical components of the target computer system, relationships between those objects can be discovered. This facilitates a meaningful understanding of the computer system by understanding the relationship between its various logical components. For instance, a User object (A) would have a relationship with a Word Processor object (B) and a Document object (C) where A represented a user who used the software represented by B to create a text file represented by C. This simple web of relationships is illustrated in Figure B. In an interactive computer forensic investigation, a human investigator may be able to recognise such relationships intuitively; however, an automated process has no intuition. Identifying such relationships automatically is the key feature of the computer profiling process, allowing the human investigator’s time to be best prioritised in computer forensic investigations.

Marrington et al identified several areas for future work in the field of computer profiling, especially the discovery and representation of the history of the target computer system, through the representation of historical states of objects. This would

allow for a time-dependent or history profile of the entire target computer system to be constructed via composition of information about object states [1]. This paper proposes a technique for reconstructing the history of a computer system through event correlation.

2.3 Event Correlation in Digital Forensics

Event logs are recorded by the operating system as well as by various subsystems and applications. There has been significant research about the examination (or auditing) of such logs for forensic purposes. Event correlation, in the context of digital forensics, is the examination of recorded events in sets of logs in order to provide various kinds of forensic trace information, including:

- user activity,
- application activity,
- identification of anomalous, unusual or even criminal activity, and
- correlation of activity traces, e.g. “user A downloaded images X, Y and Z – and so did user B”.

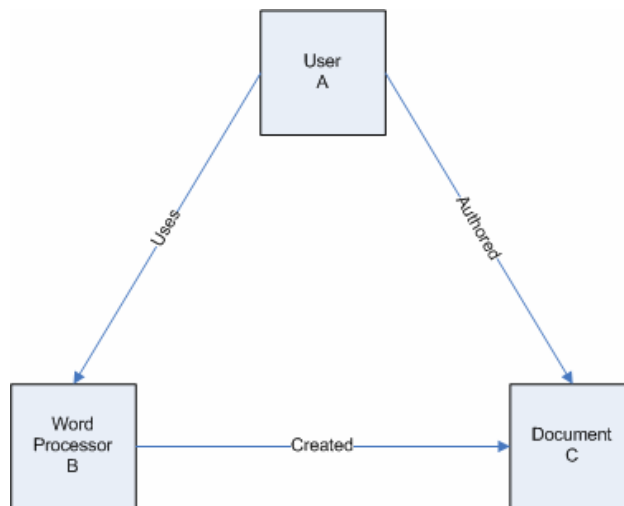


Fig B A simple relationship where A created C using B.

The specific value of event correlation in the context of automatic computer profiling lies in the creation of activity traces of objects which are as complete as possible because they are derived not from a single source, but from every event log source on the target computer system, and ordered into a meaningful sequence. The Event Correlation for Forensics (ECF) framework was developed by Chen et al with the strategic objective “to develop a means by which a consolidated repository of event information can be constituted and then queried in order to provide an investigator with post hoc event correlation” [7]. Using the ECF it is possible to create activity traces and to match them against scenarios. This is accomplished by using a

manual query-based system which allows an investigator to query a database composed of event information parsed from heterogeneous sources and stored in a standardised canonical format. This work was subsequently expanded upon into an Auto-ECF system, which provides automated recognition of event scenarios [8]. Using the Auto-ECF system, Abbott et al were able to recognise three test scenarios, one involving the distribution of pornographic material/pirated software, another involving an exploit of the Windows 2000 operating system, and a third involving the CaseSen attack from the 1999 DARPA IDS evaluation data set. We address the research problem of providing historical timelines for computer activity by incorporating an approach similar to the ECF with computer profiling.

3 Design

3.1 Rationale

Computer profiling is fundamentally an activity about reconstructing and characterising the usage and behaviour of a computer system, which has an obvious historic dimension. An overview of a computer system at the time of seizure may allow an investigator to infer certain facts about its usage over time. Support for and confirmation of such inferences can be provided by historical information extracted from the system in question, and this may be of critical importance in subsequently finding and providing actual admissible evidence. We hypothesise that event correlation can be useful, employed as part of the computer profiling process, in order to construct timeline activity traces for the objects discovered on a given computer system. There are three obvious forms of historical activity tracing when considering the forensic examination of stand alone computer systems:

- File activity tracing
- Application activity tracing
- User activity tracing through the reconstruction of user sessions

Many file analysis tools, such as those discussed in section 2, support file activity tracing. EnCase, for example, provides a calendar-like view of file activity over time [9]. Investigators employing file analysis tools can construct application and user activity timelines by consulting the logs of the computer system being investigated, but this is a manual activity, or must be facilitated by purpose-built scripts. Computer profiling is better suited to creating activity traces of applications and users because such timelines can be placed as part of a model about the computer system as a whole, rather than its filesystem exclusively. A complete computer profile, constructed by a practical computer profiling tool, would create application and user activity timelines automatically and present them to an investigator as part of its results summary.

3.2 Discovered Events

An approach similar to the ECF allows events occurring over time to be combined and understood as part of the computer system's history, or as part of the history of

specific objects. Events can be discovered through the examination of the various logs on the target computer system. When an event of some description occurs, the operating system or the application in question records one or more entries in one or more logs on a target computer system. System or application settings select which events are recorded and which are allowed to pass without being logged. In regards to the Windows Event Logs, for instance, without enabling auditing, only certain limited types of events are logged and visible in the Event Viewer, whereas with auditing enabled many more events are recorded. These settings are recorded in the Windows Registry and controlled through a graphical user interface for Local Security Policy settings [10]. Our design identifies and records such events into a database table of events along with the unique identifiers of the objects concerned by the event. Employing a database in this fashion rather than constructing a simple list allows for the subsequent reconstruction of timelines concerning specific objects.

We utilise an ECF-like database design to support this time-lining functionality, similar to the design employed in [7], as an event correlation subsystem for incorporation into an implementation of the computer profiling process. Chen et al's design employs a canonical form for all events irrespective of the different logs or systems from which the events were initially drawn in the log parsing stage. We utilise a table which performs the same function for the computer profiling event correlation subsystem, although our table has less fields than Chen's. This is possible because of the computer profiling object model described in section 2 – we require no fields specifically relating to the type of the subject and object as this information is easily determined using the object ID of each. Our table is called the *Discovered Events* table, which comprises the following fields:

- Event ID – Primary key for the event
- Time – The date and time of the event
- Subject – The ID of the object representing the instigator of the action
- Object – The ID of the object being manipulated
- Action – The action being performed
- Result – The outcome of the action (success/failure/unknown)

3.3 Inferred Events

Not all available historical information appears as event records in computer logs. The objects discovered by our computer profiling software provide additional historical information which is also of significant potential value. It is important that this information be harnessed and incorporated along with the information gathered from the various logs discovered on a computer system under investigation. Doing so will ensure that the timelines constructed using the event correlation subsystem are as complete as possible. To incorporate this information we utilise a second table, of identical design to the *Discovered Events* table, called the *Inferred Events* table. For instance, where a causal conditional exists it is possible to infer that a particular event took place, even if no direct record of it exists. If P causes Q, and a log record of event P can be discovered, we can infer that event Q also took place. Some attributes of objects in our model of a computer system may be used to infer such events.

For example, the “modified time” of a Content object representing a file which is associated with a particular program on the computer system, represented by an Application object, yields an inferred event being that of the Application object modifying the Content object. Of course, the “modified time” field is provided by the target computer system's filesystem, and could have been changed through the modification of the file by some other program than that represented by the Application object. The fact that the event in question is drawn from the *Inferred Events* table allows an investigator to be informed that the event is inferred only, so that the investigator can differentiate between events of which there is direct evidence and events which are simply inferred to have occurred.

3.4 Time-lining of Application and User Activity

Having collected event information and inferred other event information and stored this data in the *Discovered Events* and *Inferred Events* tables, it is possible to construct timelines through querying the two tables. By querying both tables simultaneously, a timeline can be constructed which will consist of both discovered and inferred events. An activity trace can be constructed for a particular object by searching for its object ID in the “Subject” and/or “Object” fields. For instance, using our prototype, the following SQL query shows an ordered list of events for which a given object (“%OBJECT%”) is the subject:

```
SELECT * FROM (SELECT * FROM InferredEvents UNION ALL
SELECT * FROM DiscoveredEvents) AS universalevents
WHERE universalevents.Subject LIKE "%OBJECT%" ORDER BY
universalevents.Time;
```

In this query, “%OBJECT%” can be replaced by a User object’s object ID to show an activity timeline of those events in which that User object was the actor. The same field can also be replaced by a given Application object’s object ID to show an application activity trace for that Application object. A more complete timeline could be constructed by searching for the object ID in question in the Object field as well as in the Subject field.

As computer profiling is an automated process, the above process of querying the database and constructing activity traces must be automated. Queries are pre-written and stored in a list of queries to be performed automatically after all the objects on a computer system have been discovered and after the *Discovered Events* and *Inferred Events* tables have been fully populated. Mindful that overwhelming an investigator with too much information up front compromises the usefulness of computer profiling as a technique (as discussed in [1]), this information must be provided in an accessible but not overwhelming fashion. User activity traces of each user need to be presented with the other results relevant to that user. Application activity traces too need to be presented with the relevant results for a given application.

Our design therefore incorporates the above event correlation subsystem which in turn includes a relational database with two tables in which to store events – the *Discovered Events* table and the *Inferred Events* table. Our design incorporates the

automated querying of these tables in such a way as to reconstruct a series of events as an activity trace for a user, application, or a particular file. The inclusion of the event correlation subsystem also necessitates the incorporation of log parsers to process the event logs on the target computer system. To evaluate our approach, we have implemented a prototype system which implements the above computer profiling technique and which incorporates an event correlation subsystem. We have employed this prototype software to construct a computer profile of a test case desktop computer and gathered the events necessary to build two exemplar timelines. The first is an activity trace of a particular application, the other an activity trace of a particular user of the desktop computer. Our implementation and experiment are discussed in more detail, below.

4 Experimental Results

4.1 Experiment

As stated earlier, our research objective has been to identify and design a means by which to create timelines of computer activity so as to allow investigators to trace the activity of a particular user or application. Having proposed a design to do this in section 3, we needed to evaluate the viability of our approach. To this end we have conducted an experiment focussed on assessing the usefulness of event correlation as a technique to construct user and application activity traces. We sought to construct timelines of user sessions, listing a user's activity during each session, and timelines of application activity, showing, for instance, the files opened and modified by an application.

4.2 Implementation

The core of our proof-of-concept implementation was undertaken in the Java programming language, and our software incorporates several different tools and libraries which are executed by this core and which pass their results back to the core. These tools and libraries are:

- libextractor (<http://gnunet.org/libextractor/>) - a library which can extract meta-information from files of arbitrary type. This information is stored as attributes of objects.
- GrokEVT (<http://projects.sentinelchicken.org/grokevt/>) - a collection of Python scripts built for reading Windows NT/2000/XP event log files stored on a mounted Windows partition.
- A series of Perl scripts which extract modified, accessed and created (MAC) times of files, extract meta-information from Microsoft Word document files, and give a detailed list of the users of a Windows host. These scripts have been heavily modified but their original forms are distributed in [10].

Our prototype software communicates with a MySQL database, which contains three tables, the *Keyword Associations* table (in which object attributes are stored to

allow relationships between objects to be discovered through querying) [1], and the *Discovered Events* and *Inferred Events* tables, as described in Section 3. Our prototype software runs in Linux, and only builds profiles of machines running the Windows operating system, although our approach is generic and can be applied to other operating systems. The filesystem of the Windows host to be profiled is mounted as a read-only partition, allowing our profiler software to be used to examine an offline system without altering the digital evidence. In practical environments a write blocker and/or bit-wise image of the target filesystem may be incorporated.

Our prototype implementation is intended to evaluate the viability and usefulness of event correlation as a sub-activity of computer profiling. Our prototype software discovers objects from each of the following three categories: Content, Application, and Principal objects, describing respectively data files, programs, and people. It does not include the implementation of identifying System objects. The prototype discovers objects belonging to each of these categories, and extracts the meta-information about each object, providing especially rich information about Content objects. This meta-information is saved as attributes of the relevant object, and is inserted into the *Keyword Associations* table of the database for subsequent querying so that relationships between objects on the basis of their attributes may be discovered. The objects themselves may be saved in a serialised form and reloaded (along with the database) for subsequent re-examination.

4.3 Evaluation

We used an office desktop PC belonging to one of the authors as the target computer in our experiment. This computer runs Windows XP Professional, and a variety of programs including the Microsoft Office 2003 application suite. Only one user primarily uses the computer, although it is part of a domain, and occasionally other users from the same organisation use it. It is used for word processing (using Microsoft Word), web browsing, and very little else. It would be reasonable to characterise this computer as a standard office computer.

All three Windows event logs were available for this computer. The Application log contained 9382 records, the Security log 25683, and the System log 9249. Our analysis focussed on Application and Content objects situated on the D: drive partition of this office desktop, which contained 204.8MB of data in 4551 files in 389 sub-folders.

4.4 Results

After completing the generation of the computer profile, our software presents the investigator with a summary screen which shows the number of objects which have been discovered from each of the three categories. Our profiling software's automated examination discovered 16 objects belonging to the Principal category, 42 Application objects, and 4551 Content objects. From the Windows event logs, we extracted 44314 events to insert into the *Discovered Events* table. Given that there were 44314 records in total across the three Windows event logs, this means that there

were there were no events which were not inserted into the *Discovered Events* table. However, because our computer profiling tool does not implement System objects, many of the events (especially those extracted from the System log) have a subject or object field containing simply the word “SYSTEM”. In a complete implementation, System objects would be implemented, and the fields populated by the word “SYSTEM” in our results would instead be populated by the object ID of a specific System object. Our software allows the investigator to browse through the lexicon of objects discovered on the computer system (as seen in Figure C), and then display an activity timeline for each of these, subject to the limitations on our software implementation discussed above.

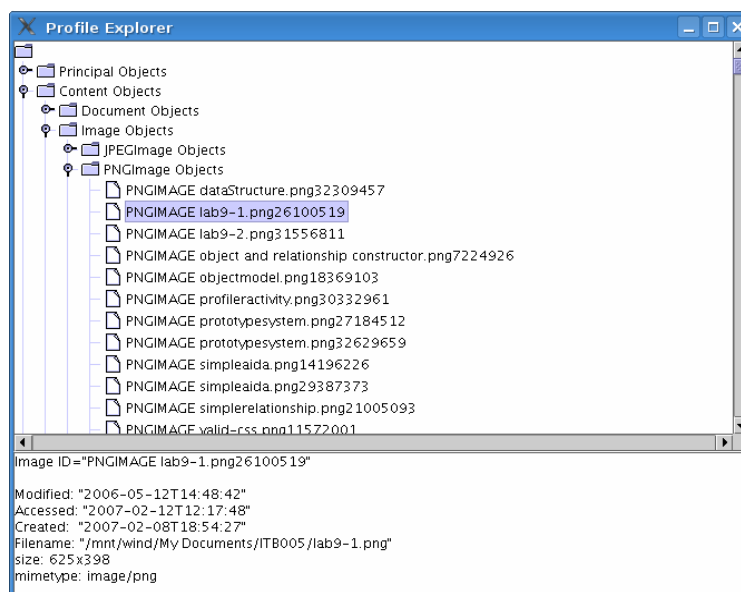


Fig C A screenshot showing our prototype's object lexicon browser.

To exercise the application timelining functionality, we created an activity trace for the Microsoft Word Application object. This object’s unique object ID was “APPLICATION MSWORD16327700”. Our software inferred 254 events with this object ID as the subject. These inferred events were constructed on the basis of the fields of Document (a sub-class of Content) objects, which were initially extracted from the meta-information embedded in Word files. These inferred events were ordered by time to show a sequential list of the Documents created, modified, and opened by Word which were stored on the target partition. Our software extracted only six discovered events, which were also included in the timeline – these represented error messages or application crashes recorded in the Application log. An excerpt of this timeline is shown in Table A, below. Note that the only discovered event in this excerpt corresponds to Microsoft Word encountering an error.

Through combining information from the *Discovered Events* and *Inferred Events* tables, we were also able to create simple activity traces for user sessions. The *Discovered Events* table contained logon/logoff information which allowed the bounds of a user session to be established. This information was combined with inferred events with the user as the subject in order to reconstruct the user’s activities during each session. The complete user activity timeline for the computer’s primary user (who has been de-identified as “INDIVIDUAL 1 anonymous”) consisted of 1166 events. An excerpt of this timeline, showing several sessions of the same user, is shown in Table B.

EventID	Source Table	Time	Subject	Object	Action	Result
44361	Inferred	01/03/05 05:23	APPLICATION MSWORD16327700	WORDDOC confirmation.doc9110923	CREATED	Success
44367	Inferred	01/03/05 05:23	APPLICATION MSWORD16327700	WORDDOC old_confirmation.doc17940412	CREATED	Success
44379	Inferred	01/03/05 05:23	APPLICATION MSWORD16327700	WORDDOC -WRL0005.tmp29477163	CREATED	Success
44316	Inferred	04/03/05 05:41	APPLICATION MSWORD16327700	WORDDOC 2005 Top Up Application.doc33320810	CREATED	Success
44570	Inferred	22/04/05 18:26	APPLICATION MSWORD16327700	WORDDOC -WRL0005.tmp29477163	MODIFIED	Success
44368	Inferred	12/05/05 06:36	APPLICATION MSWORD16327700	WORDDOC model.doc28868898	CREATED	Success
44331	Inferred	19/05/05 06:55	APPLICATION MSWORD16327700	WORDDOC CS1 - AUP Violation.doc4977982	CREATED	Success
44426	Inferred	20/05/05 13:57	APPLICATION MSWORD16327700	WORDDOC CS1 - AUP Violation.doc4977982	MODIFIED	Success
25	Discovered	21/06/05 11:22	APPLICATION MSWORD16327700	APPLICATION MSWORD16327700	FAULT	Unknown
44399	Inferred	13/09/05 19:45	APPLICATION MSWORD16327700	WORDDOC ASWEC 2005.doc394365	MODIFIED	Success
44324	Inferred	26/09/05 04:52	APPLICATION MSWORD16327700	WORDDOC Annual Progress Report 2005.doc6554172	CREATED	Success
44325	Inferred	26/09/05 04:52	APPLICATION MSWORD16327700	WORDDOC Annual Progress Report 2006.doc4167406	CREATED	Success
44534	Inferred	27/09/05 11:17	APPLICATION MSWORD16327700	WORDDOC old_confirmation.doc17940412	MODIFIED	Success
44347	Inferred	18/10/05 07:00	APPLICATION MSWORD16327700	WORDDOC SRS0.1.doc21409163	CREATED	Success
44444	Inferred	15/12/05 13:56	APPLICATION MSWORD16327700	WORDDOC Modelling Operating Systems.doc28110456	MODIFIED	Success
44447	Inferred	15/12/05 13:56	APPLICATION MSWORD16327700	WORDDOC Object Definition.doc21263661	MODIFIED	Success
44405	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC Annual Progress Report 2005.doc6554172	MODIFIED	Success
44531	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC model.doc28868898	MODIFIED	Success
44465	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC Quality Assurance in a Student Based Agile S	MODIFIED	Success
44492	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC ace52.doc28532785	MODIFIED	Success
44480	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC ScheduleModule3.doc19736274	MODIFIED	Success
44483	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC SourceMaterialsForExam.doc5482965	MODIFIED	Success
44498	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC assignment 3.doc12085572	MODIFIED	Success
44429	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC Computer related crimes.doc11320634	MODIFIED	Success
44441	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC InstructionsPARTB.doc19356212	MODIFIED	Success
44438	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC Implications for the Disc.doc10311571	MODIFIED	Success
44501	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC assignment1_question1n2.doc2526406	MODIFIED	Success
44435	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC ITN673CollectedReport2.doc19061461	MODIFIED	Success
44390	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC A1PartBRubric.doc1801334	MODIFIED	Success
44555	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC pauls_submission.doc16655704	MODIFIED	Success
44411	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC Assignment 1 Part A.doc8306728	MODIFIED	Success
44432	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC ITN673.Exam.2003.2.doc7312507	MODIFIED	Success
44504	Inferred	15/12/05 17:57	APPLICATION MSWORD16327700	WORDDOC assignment4.doc26760685	MODIFIED	Success

Table A – The first 33 events in the *application activity timeline* for the “APPLICATION MSWORD16327700” object.

While constructing the user activity timeline shown in Table B, we were initially surprised to discover that our *Discovered Events* table only contained logon failures, and no successes. Nevertheless, we found that logon failures often framed the activities of a user session. An examination of the test computer’s logging settings revealed that only failure audits were enabled for logon/logoff events, which explained the absence of logon successes in the *Discovered Events* table. We concluded that user sessions were often framed by logon failures as a result of the user in question mistyping his/her password one or more times before successfully logging in.

We then enabled success audits for logon/logoff events, and several days later we used our software to create a new timeline for the same user (once again de-identified and called “INDIVIDUAL 1 anonymous”). This second timeline appears in Table C. In Table C, sessions are framed by successful logons and logoffs. With the new level of auditing enabled, we can also see “CREATE PROCESS” events, although because our software does not implement System objects, we can’t see which processes have

been spawned. As in Table B, we can see the user's file activity, inferred from the fields of Content objects discovered during the computer profiling process.

Event ID	Source Table	Time	Subject	Object	Action	Result
44733	Inferred	30/01/07 21:19	INDIVIDUAL 1 anonymous	WORDDOC Relationship_Definitions.doc17242295	MODIFIED	Success
44633	Inferred	30/01/07 21:35	INDIVIDUAL 1 anonymous	CONTENT 6435687	MODIFIED	Success
32840	Discovered	31/01/07 15:10	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
44763	Inferred	31/01/07 21:43	INDIVIDUAL 1 anonymous	WORDDOC confirmation.doc4102111	MODIFIED	Success
32839	Discovered	01/02/07 12:19	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
32835	Discovered	01/02/07 13:23	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
32831	Discovered	01/02/07 13:30	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
32825	Discovered	01/02/07 13:34	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
32821	Discovered	02/02/07 14:27	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
32819	Discovered	04/02/07 16:02	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
32813	Discovered	05/02/07 14:26	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
32812	Discovered	06/02/07 16:00	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
32804	Discovered	06/02/07 16:05	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
32802	Discovered	08/02/07 18:44	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
44666	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	DOCUMENT timeline2006.xls31344098	CREATED	Success
44672	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	DOCUMENT totaltimeline.xls10014334	CREATED	Success
44678	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	WORDDOC 6_Monthly_Report_2006.doc11845181	CREATED	Success
44690	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	WORDDOC Annual Progress Report 2005.doc6554172	CREATED	Success
44692	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	WORDDOC Annual Progress Report 2006.doc4167406	CREATED	Success
44704	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	WORDDOC CS1 - AUP Violation.doc4977982	CREATED	Success
44738	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	WORDDOC Sample Cases for Computer Profiling.doc10140210	CREATED	Success
44758	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	WORDDOC computerprofilingecf.doc18206828	CREATED	Success
44762	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	WORDDOC confirmation.doc4102111	CREATED	Success
44624	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	CONTENT 31447144	CREATED	Success
44800	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	WORDDOC -WRL0005.tmp29477163	CREATED	Success
44660	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	DOCUMENT timeline.xls14871751	CREATED	Success
44776	Inferred	08/02/07 18:54	INDIVIDUAL 1 anonymous	WORDDOC old_confirmation.doc17940412	CREATED	Success

Table B – *User activity timeline* excerpt with individual user sessions highlighted. Windows Event Log success auditing was turned off, thus the only “LOGON” events recorded are failures.

It should be noted that in addition to reconstructing the sessions of a particular user, it is also possible to reconstruct the sessions of all the users of a computer system and display this in a single timeline of user sessions. Using our prototype, simultaneous sessions are displayed in order of the logon time, so that in the event of two users being logged into the computer system simultaneously, the complete session of the user who logged on first is displayed before the session of the user who logged on next. In practical investigations of multi-user machines, a more advanced representation of simultaneous user sessions may be desirable.

5 Discussion

Our results demonstrate the viability of event correlation as a technique for composing activity traces for users and applications. Although we have focussed our application activity tracing efforts on a single application, Microsoft Word, our approach is generic and can be applied to gather information about any and all applications. Our user activity tracing would also benefit from cross-application support, as this would allow us infer more types of events about other applications.

In regards to the reconstruction of user sessions, our technique produces more detail about more recent sessions than it does for earlier sessions. This is because we infer many more events about recent user sessions, for the simple reason that in many cases, the only timestamps associated with files represent the latest instance of activity, such as the “last printed” date, or “last modified” date. This means that

events will not be inferred, for instance, for modifications which took place before the latest modification. This deficiency might be partially overcome in a practical computer profiling tool by examining temporary files and fragments of deleted temporary files in an attempt to reconstruct previous versions of a file. It might also be mitigated in cases involving journaling file systems, where richer historical information would be available. By inferring events from Content objects and combining this information with events from the *Discovered Events* table, extracted from the operating system's audit logs, our approach presents richer historical information than a file analysis tool. Our approach also presents this historical information in a more intuitive form, by reconstructing a series of events automatically and presenting them in a timeline.

An obvious limitation of any time-lining activity based on timestamps provided by a computer's system clock is the inaccuracy inherent in such clocks. This is a limitation shared by our approach and by file analysis tools such as EnCase [9]. The solution for addressing this issue suggested most frequently in the literature is to note the system clock time of a computer under investigation at the time of its examination and to determine the discrepancy between that time and the time of a reference clock [11, 12]. However, this solution does not address the issue of clock skew varying over time prior to the examination of the computer system. Schatz et al proposed a technique for establishing the provenance of timestamps employing correlation of events stored in logs on two different hosts in a client/server relationship [13]. This technique might prove useful for determining the accuracy of timelines constructed with the method we have proposed in cases where investigators also have access to a server which has had frequent contact with the computer under investigation (for instance, in internal investigations in a corporate environment).

5 Conclusion

Event correlation is an activity which can be used to characterise activity on a computer system or systems. As such, it has significant value to computer profiling, which is a methodology conceived for the automated reconstruction of a computer system in order to provide direction for digital forensic investigations.

The event correlation subsystem for computer forensics which we specified in section 3 stores information found in the operating system's event logs in a *Discovered Events* table in a relational database. This information is combined with events inferred during an examination of the filesystem itself and stored in an *Inferred Events* table of the same database. By querying both tables, it is possible to correlate the disparate sources of events and reconstruct a timeline of application or user activity.

Event ID	Source Table	Time	Subject	Object	Action	Result
32262	Discovered	13/02/07 14:23	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
32265	Discovered	13/02/07 14:23	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Success
32177	Discovered	13/02/07 14:24	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
32176	Discovered	13/02/07 14:25	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
32169	Discovered	13/02/07 14:25	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
32165	Discovered	13/02/07 14:27	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
32050	Discovered	13/02/07 16:49	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
32039	Discovered	13/02/07 16:49	INDIVIDUAL 1 anonymous	SYSTEM	LOGOFF	Success
32041	Discovered	13/02/07 16:49	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Success
44799	Inferred	13/02/07 16:49	INDIVIDUAL 1 anonymous	WORDDOC structured objectives.doc17111494	MODIFIED	Success
32033	Discovered	13/02/07 16:50	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
32027	Discovered	13/02/07 16:51	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31938	Discovered	13/02/07 18:20	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31928	Discovered	13/02/07 18:20	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31927	Discovered	13/02/07 18:20	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31925	Discovered	13/02/07 18:21	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31924	Discovered	13/02/07 18:21	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31921	Discovered	13/02/07 18:21	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31920	Discovered	13/02/07 18:21	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31919	Discovered	13/02/07 18:21	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31913	Discovered	13/02/07 18:23	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31912	Discovered	13/02/07 18:23	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31908	Discovered	13/02/07 18:23	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31897	Discovered	13/02/07 18:28	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31896	Discovered	13/02/07 18:29	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
44796	Inferred	13/02/07 18:29	INDIVIDUAL 1 anonymous	WORDDOC qy\policy.doc25624563	CREATED	Success
44797	Inferred	13/02/07 18:29	INDIVIDUAL 1 anonymous	WORDDOC qy\policy.doc25624563	MODIFIED	Success
31895	Discovered	13/02/07 18:29	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31894	Discovered	13/02/07 18:29	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31893	Discovered	13/02/07 18:29	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31888	Discovered	13/02/07 18:29	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
31931	Discovered	13/02/07 18:30	INDIVIDUAL 1 anonymous	SYSTEM	LOGOFF	Success
2830	Discovered	14/02/07 14:10	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Success
2835	Discovered	14/02/07 14:10	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
2816	Discovered	14/02/07 14:11	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
2790	Discovered	14/02/07 14:12	INDIVIDUAL 1 anonymous	SYSTEM	LOGOFF	Success
2003	Discovered	14/02/07 14:42	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
2006	Discovered	14/02/07 14:42	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Success
1929	Discovered	14/02/07 14:52	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1928	Discovered	14/02/07 14:52	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1916	Discovered	14/02/07 15:02	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1909	Discovered	14/02/07 15:12	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1874	Discovered	14/02/07 15:58	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1861	Discovered	14/02/07 16:03	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1857	Discovered	14/02/07 16:07	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1856	Discovered	14/02/07 16:07	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1812	Discovered	14/02/07 16:34	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1777	Discovered	14/02/07 17:30	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1766	Discovered	14/02/07 17:43	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1755	Discovered	14/02/07 17:56	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1735	Discovered	14/02/07 18:07	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1730	Discovered	14/02/07 18:07	INDIVIDUAL 1 anonymous	SYSTEM	LOCK	Success
1732	Discovered	14/02/07 18:07	INDIVIDUAL 1 anonymous	SYSTEM	UNLOCK	Success
1709	Discovered	14/02/07 18:18	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1703	Discovered	14/02/07 18:28	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1694	Discovered	14/02/07 18:39	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1688	Discovered	14/02/07 18:40	INDIVIDUAL 1 anonymous	SYSTEM	LOCK	Success
1690	Discovered	14/02/07 18:40	INDIVIDUAL 1 anonymous	SYSTEM	UNLOCK	Success
44785	Inferred	14/02/07 18:40	INDIVIDUAL 1 anonymous	WORDDOC paper.doc16042569	MODIFIED	Success
1676	Discovered	14/02/07 18:42	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1675	Discovered	14/02/07 18:42	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1674	Discovered	14/02/07 18:42	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1672	Discovered	14/02/07 18:42	INDIVIDUAL 1 anonymous	SYSTEM	LOGOFF	Success
1556	Discovered	15/02/07 13:55	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Failure
1553	Discovered	15/02/07 13:55	INDIVIDUAL 1 anonymous	SYSTEM	LOGON	Success
1470	Discovered	15/02/07 13:56	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1467	Discovered	15/02/07 13:56	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1461	Discovered	15/02/07 13:57	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1458	Discovered	15/02/07 13:57	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1457	Discovered	15/02/07 13:57	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1454	Discovered	15/02/07 13:58	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1453	Discovered	15/02/07 13:58	INDIVIDUAL 1 anonymous	SYSTEM	LOCK	Success
1258	Discovered	15/02/07 14:06	INDIVIDUAL 1 anonymous	SYSTEM	UNLOCK	Failure
1255	Discovered	15/02/07 14:06	INDIVIDUAL 1 anonymous	SYSTEM	UNLOCK	Success
1145	Discovered	15/02/07 14:50	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1139	Discovered	15/02/07 14:52	INDIVIDUAL 1 anonymous	SYSTEM	LOCK	Success
1141	Discovered	15/02/07 14:52	INDIVIDUAL 1 anonymous	SYSTEM	UNLOCK	Success
1122	Discovered	15/02/07 15:16	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1117	Discovered	15/02/07 15:16	INDIVIDUAL 1 anonymous	SYSTEM	LOCK	Success
1107	Discovered	15/02/07 15:16	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1106	Discovered	15/02/07 15:16	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1096	Discovered	15/02/07 15:31	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1027	Discovered	15/02/07 16:47	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
1020	Discovered	15/02/07 16:47	INDIVIDUAL 1 anonymous	SYSTEM	UNLOCK	Success
44788	Inferred	15/02/07 17:58	INDIVIDUAL 1 anonymous	WORDDOC paper.doc9518166	CREATED	Success
44789	Inferred	15/02/07 17:58	INDIVIDUAL 1 anonymous	WORDDOC paper.doc9518166	MODIFIED	Success
942	Discovered	15/02/07 17:58	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
941	Discovered	15/02/07 17:58	INDIVIDUAL 1 anonymous	SYSTEM	CREATE PROCESS	Success
940	Discovered	15/02/07 18:01	INDIVIDUAL 1 anonymous	SYSTEM	LOGOFF	Success

Table C – User activity timeline excerpt from a period with Windows Event Log success auditing turned on. Individual sessions have been highlighted in alternating

There are a number of areas which require future research in the field of computer profiling. An obvious research problem lies in the automated characterisation of Content objects. In future work, we intend to investigate the application of known-file filter technology to characterise and categorise hard disk contents. Existing forensic techniques typically employ KFF technology to eliminate operating system files and common application files from an investigation, allowing an investigator to focus his or her efforts on the remaining files. We believe that KFF technology could be used in computer profiling to allow an “educated guess” to be made about the contents of a filesystem or subdirectory on the basis of positive identification of files belonging to suspicious categories (e.g. pornography, copyrighted music, etc) using a KFF database. In investigations of distribution rings involving multiple computers, some of the same files would be found on each computer in the ring.

We believe that some theoretical aspects of computer profiling, especially relationships between objects in the model described in [1] have been inadequately defined. Future research should aim to provide formal definitions of objects and the relationships between them which are discovered as part of the computer profiling process. This may entail a formal representation of objects, the relationships between them, and their history, perhaps employing a mark-up language such as the Web Ontology Language (OWL), or the Digital Evidence Mark-up Language [6]. Creating formal definitions for objects and relationships will allow future research to expand upon the usefulness of computer profiling to investigators by adding more granularity and detail.

In this paper we have described how event correlation can be incorporated into the computer profiling process to create timelines of user and application activity. We believe that computer profiling is an important digital forensics activity with the potential to significantly improve the efficiency of forensic investigations, and that via an analysis of event logs and the historical information stored in computer files, a more complete and useful view of a computer’s history can be obtained than has been possible with other sorts of computer forensic tools. We have presented a proof-of-concept implementation of the computer profiling process incorporating an event correlation subsystem of our own design, and have employed that prototype software to construct timelines of an example application’s activity, and an example user’s activity. We have demonstrated the value and viability of event correlation as a technique in a forensic examination of a stand-alone computer system, and conclude that it is a worthwhile addition to computer profiling in the automated forensic reconstruction of a computer system.

Acknowledgements. The authors would like to acknowledge the input they received from Mark Branagan.

References

- [1] A. Marrington, G. Mohay, H. Morarji, and A. Clark, "Computer Profiling to Assist Computer Forensic Investigations," presented at RNSA Security Technology Conference, Canberra, 2006.

- [2] Guidance Software, "EnCase Forensic: Product Overview," http://www.guidancesoftware.com/products/ef_index.aspx, 2007, Accessed: 14 February, 2007.
- [3] AccessData, "Forensic Toolkit - Overview," <http://www.accessdata.com/catalog/partdetail.aspx?partno=11000>, 2007, Accessed: 14 February, 2007.
- [4] B. Carrier, "The Sleuthkit & Autopsy: Forensic Tools for Linux and other Unixes," <http://www.sleuthkit.org>, 2005, Accessed: 15 April, 2005.
- [5] N. L. Beebe and J. G. Clark, "Dealing with Terabyte Datasets in Digital Investigations," in *Research Advances in Digital Forensics*, M. Pollitt and S. Sheno, Eds. Norwell: Springer, 2005, pp. 3-16.
- [6] R. Eaglin and J. P. Craiger, "Data Sharing and the Digital Evidence Markup Language," presented at 1st Annual GJXDM Users Conference, Atlanta, 2005.
- [7] K. Chen, A. Clark, O. de Vel, and G. Mohay, "ECF - Event Correlation for Forensics," presented at Australian Computer, Network & Information Forensics Conference (ACNIFC), Scarborough, Australia, 2003.
- [8] J. Abbott, J. Bell, A. Clark, O. de Vel, and G. Mohay, "Automated recognition of event scenarios for digital forensics," in *Proceedings of the 2006 ACM symposium on Applied computing*. Dijon, France: ACM Press, 2006, pp. 293-300.
- [9] G. Mohay, A. Anderson, B. Collie, O. de Vel, and R. McKemmish, *Computer and Intrusion Forensics*. Norwood: Artech House, 2003.
- [10] H. Carvey, *Windows Forensics and Incident Recovery*, 1st ed. Boston: Addison-Wesley, 2005.
- [11] C. Boyd and P. Forster, "Time and date issues in forensic computing - a case study," *Digital Investigation*, pp. 18-23, 2004.
- [12] R. Nolan, C. O'Sullivan, J. Branson, and C. Waits, "First responder's guide to computer forensics." Pittsburgh: Software Engineering Institute, Carnegie Mellon University, 2005.
- [13] B. Schatz, G. Mohay, and A. Clark, "A correlation method for establishing provenance of timestamps in digital evidence," *Digital Investigation - The Proceedings of the 6th Annual Digital Forensic Research Workshop (DFRWS '06)*, vol. 3, pp. 98-107, 2006.