Digital Investigation (2004) 1, 18–23

ELSEVIER

Digital Investigation

www.elsevier.com/locate/diin

Time and date issues in forensic computing—a case study

Chris Boyd, Pete Forster*

National Technical Assistance Centre, UK

Received 8 January 2004; accepted 13 January 2004

KEYWORDS
Forensic;
Computing;
Time;
Date;
Evidence;
Microsoft;
Windows;
Internet;
Explorer;
UTC;
Local;
Translation;
Ethics

Abstract Evidence concerning times and dates in forensic computing is both important and complex. A case study is outlined in which forensic investigators were wrongly accused of tampering with computer evidence when a defence expert misinterpreted time stamps. Time structures and their use in Microsoft Internet Explorer are discussed together with local and UTC time translation issues. A checklist for examiners when producing time evidence is suggested underlining the need for the examiner to fully understand the meaning of the data that they are seeking to interpret before reaching critical conclusions.

Introduction

Date and time evidence is a fundamental part of many forensic computing examinations. Forensic examiners know that lawyers are often drawn to dates and times because they represent a concrete link between the real world and the less easily understood world of computer evidence.

Experienced examiners know that date and time evidence is not simple and contains many potential pitfalls. Usually the more knowledge and experience that an examiner possesses the less they are willing to commit to a particular time or date. They will always try and look at the fuller picture seeking corroboration or verification of their findings.

This article is based around the case study of a recent criminal investigation in the UK, which involved time and date evidence. It discusses some of the more common forensic issues when interpreting dates and times during Microsoft Windows and Internet Explorer examinations. In addition it demonstrates a common mistake that can be made and touches on some of the ethical issues encountered.

Case study part one

The investigation began when an email trace identified an individual suspected of involvement in the communication of child abuse images. A

* Corresponding author.
E-mail addresses: Chris.Boyd@squaremail.co.uk (C. Boyd), pete@pforster.co.uk (P. Forster).

1742-2876/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.diin.2004.01.002

warrant was obtained and executed, the suspect was arrested and his computer equipment seized.

The local police computer crime unit (CCU) was tasked to identify any traces of relevant email correspondence or child abuse images. This appeared to require a relatively simple forensic examination, which when conducted identified many references to the suspect email address. In addition an indecent image depicting children was recovered from the seized computer media.

During a police interview the suspect failed to provide an explanation for the unlawful images. He was charged with relevant offences and bailed to the local court where he pleaded 'not guilty' to all charges.

The police and prosecution service planned their case strategy whilst the defence employed a forensic computing expert to comment on the digital evidence. The expert was provided with forensic images of the defendant's computer together with the police forensic statement.

When the defence report was presented to the prosecution they were shocked to find that it contained serious allegations of malpractice by the police. The summary at the beginning of the report stated:

> 'The defendants computer [ID number] was used to access the Internet after it was seized and was in police custody. Approximately 750 records of Internet access are time stamped during the six hours or so after the computer was seized…'

and

> 'pages accessed included Hotmail login pages and possible child pornography site. Floppy diskettes were also used.'

The main body of the highly critical report contained the statement:

> 'There is substantial evidence that is consistent with the Defendant's computer [ID number] being altered while it was in police custody'.

The report indicated that it was likely the police had placed the indecent image depicting children onto the computer themselves and concluded by stating:

> 'However I am sure that there are so many grave problems with this evidence, and with all the computer evidence submitted by the prosecution, that the Court cannot safely rely on it.'

So what had gone wrong? Were the police guilty of malpractice or even corruption? The defence expert had made these allegations based on their own interpretation of time and date evidence. Before answering these questions it is therefore worth discussing dates and times in a forensic context in more depth. Common formats for storing dates and times will be examined followed by a look at how Internet Explorer uses time stamps.

Date and time structures

The CMOS clock

The Complementary Metal Oxide Semiconductor (CMOS) is an on-board semiconductor chip. It includes a simple clock function that calculates current time. Most desktop operating systems and applications access the CMOS clock using calls to interrupt 0x1A.

32 bit Windows/DOS time format

This is stored in a binary (or bit) packed format (i.e. the bit values can cross byte boundaries). The date and time is stored in a 32 bit (4 byte) structure as follows:

- Seconds occupy 5 bits from offset 0, minutes occupy 6 bits from offset 5, and hours occupy 5 bits from offset 11. These 5 bits cannot store 60 s so time must be incremented in 2 s (even) intervals.
- Days occupy 5 bits from offset 16, months occupy 4 bits from offset 21, and years occupy 7 bits from offset 25 (counting from 1980).

This format is used in FAT file systems to record the File Created, File Modified Dates and Times together with the Last Accessed Date. It is therefore often referred to as the MS DOS time/date format.

64 bit Windows FILETIME time format

This is stored as a 64 bit (8 byte) number being the number of 100 ns intervals since 00:00:00 on 1 January 1601; 1 ns is equal to 10^{-9} s.

This format is used in the NTFS Master File Table (MFT) to store the file's creation time, last modification time, last access time and the last modification time of the MFT record.

C/Unix time format

This is stored as a 32 bit number being the number of seconds since 00:00:00 on 1 January 1970. It is commonly found in association with Unix systems.

HFS and HFS+ time format

This is the Apple Mac file system date/time format and is stored as a 32 bit number being the number of seconds since 00:00:00 on 1 January 1904.

Local and UTC time translation

When dealing with different time zones or daylight saving time it is important to know if the time has been translated to Coordinated Universal Time (UTC) or if local time has been used. Universal Time (UT), Coordinated Universal Time (UTC) and Greenwich Mean Time (GMT) are effectively the same but the modern IT community tends to use UTC. Many file systems and applications automatically calculate the difference between local time and UTC and store a time/date structure as UTC. This provides advantages for the operating system or application but can confuse the forensic examiner. As a very simple example try this on an NTFS partition, create a text file, make a note of the file's created and modified times. Now change the time zone and look again. Explorer has automatically made the translation from the UTC time that was stored as an attribute in the MFT and displayed the local time.

It is important to realise that the time structure itself does not record UTC or local time, this is a decision made when the file system, operating system or application was designed and coded. There are certain facts that usually hold true, for example 64 bit Windows FILETIME is usually (but not always) translated from local time to UTC (and visa versa). This is the result of calls to the Windows API and a program could be written that ignores this protocol. Indeed Microsoft programmers were faced with complex decisions as to the relationship between UTC and local daylight savings time when they designed the NTFS. The Code Project (referenced below) has a detailed article on this.

Of the time structures discussed usually only 32 bit Windows/DOS time and HFS (but not HFS+) times are stored as local times but time/date evidence should always be checked. If the application storing them is not one that you are familiar with or is being used in an unfamiliar context then findings must be checked by experimentation.

Registry information

The time bias on a Windows machine is identifiable through interrogation of the system's registry information. On a live machine this can be accessed using non-forensic techniques such as the 'regedit' program run from the command prompt.

In a case where the registry file has been exported forensically a number of commercial products (such as regdat) are available for information extraction.

In a windows ME/XP machine the time zone bias is located in the registry key—HKEY_Local_Machine/System/Current ControlSet/Control/TimeZoneInformation/Bias.

ActiveTimeBias is the number of minutes (+ or −) to add to UTC.

The results for a computer correctly set in the UK would appear in the format shown below:

```
ActiveTimeBias    REG_DWORD    0x00000000
StandardName      REG_SZ       GMT Standard Time
```

If a computer's time was set to Pacific Standard Time (+480 min) the registry data were displayed differently:

```
ActiveTimeBias   REG_DWORD   0x000001e0
StandardName     REG_SZ      Pacific Standard Time
```

Similarly in the case of a time zone ahead of GMT, for example Nairobi (GMT+3 h), the bias would be negative (−180 min) and is represented as follows:

```
ActiveTimeBias  REG_DWORD  0xffffff4c
StandardName    REG_SZ     E. Africa Standard Time
```

This can be demonstrated by opening two windows in a Microsoft OS, one displaying RegEdit and the other the Time and Date Properties. Simply alter the time zone information using the properties window and examine the change in both the displayed time and the registry.

The registry 'TimeZoneInformation' key also holds data pertaining to any daylight saving quirks related to the time zone under 'DaylightBias' and the period when this should be applied is defined using 'DaylightStart' and 'StandardStart'. Clearly this data must also be factored into any time/date information where the recording program has used this information during storage.

Dates and times in Microsoft Internet Explorer (IE)

Microsoft Internet Explorer (IE) appears in the case study and although it is beyond the scope of this

article to discuss the format of its associated files in depth, an explanation of time evidence in this context is worthwhile. The following refers to IE versions 5.5 and 6.

IE records the URLs of Web pages that it has visited together with other information that may be of forensic interest. These records are stored in index.dat files. There are three main different types of index.dat records maintained; these are the history, the cache, and the cookies.

IE history

In respect of the history records there are three subtypes of index.dat files used by IE. These are 'root history', 'daily sort history' and 'weekly sort history' files. All have the same overall format in that each record of online activity is stored in a separate URL record. These begin at offset 0x5000 in the file. Each URL record contains varying amounts of information but two 64 bit Windows FILETIME (little endian) structures are present at offsets 0x8 and 0x10 in each record. These have different interpretations depending on the type of history file and are referred to as date 1 and date 2 in this article.

The root history is stored in the root of the '...\History\History.IE5\' folder which is variously located depending on the operating system and other factors. The root history is where IE accumulates URL records of online activity. It is important to note that these are not always in chronological order in this file. The two FILETIME dates (dates 1 and 2) are usually identical. They represent the UTC date and time that the corresponding URL was last visited using Internet Explorer.

Internet Explorer sorts and archives its history URLs on a daily and a weekly basis. These are transcribed into index.dat files, which are stored in folders off the History.IE5 root named using the dates they refer to in the following way:

MSHist0120%YYMMDD%20%yymmdd%\index.dat

where %YYMMDD% is the date the history starts and %yymmdd% is the date it finishes (not inclusive). Pairs of percentage signs (%) delimitate variables in the folder name.

Within the Sorted History URL record the entry type either consists of a host URL:

: %20YYMMDD20yymmdd%:x20
%USERNAME%@:Host:x20%URL%

or a Page URL:

: %20YYMMDD20yymmdd%:x20
% USERNAME %@%URL%

examples are:

:2004112220041123: TestName@:Host:
www.test.co.uk

and

:2004112220041123: TestName@http://
www.test.co.uk/contents.html

Each host URL reflects an index node in the History that is displayed to the user by Internet Explorer. Pages on that root index node are added to the index.dat so that a tree hierarchy can be constructed.

In respect of the daily sort date 1 is the local date and time that the corresponding URL was last visited using Internet Explorer and date 2 is the UTC date it was last visited. This will be the same as the corresponding date in the root index.dat. Clearly this has important forensic implications for establishing local time settings.

The Weekly Sort URL records are similar to the daily sort records but the date information must be interpreted differently. Date 1 is the same as date 1 in the weekly history; the date/time the URL was last visited in this period. Date 2 represents the date and time that the weekly sort took place and the URLs were transcribed into the Weekly Sort index.dat file.

IE Temporary Internet Files (cache)

The cache maintains copies of many of the files downloaded as a result of a visit to a URL and these are stored within a 'Temporary Internet Files' folder. The cache contents are accessed and maintained through an index.dat file located at

...\Temporary Internet Files\Content.IE5\
index.dat

In this file the individual records start at 0x6000. There are three types of record entry in the cache index.dat files: URL, REDR and LEAK (identified by the record header). The principal date/time evidence is to be found in the URL records, which are again stored at offsets 0x8 and 0x10.

In this case Date 1 is the associated file's original date on its originating server or host computer. It may be interpreted differently depending on the nature of the server and the actions performed on it by the server. However, in general terms, it is likely to represent the date and time that the file was uploaded to the server. Date 2 represents the UTC date and time that the cached file was last loaded by Internet Explorer.

IE cookies

Cookies are stored in the folder '...\Cookies\' together with the associated index.dat, which maintains the cookies. The cookies index.dat has a similar format to the history index.dat. Similarly the URL records have two dates at 0x8 and 0x10.

Date 1 is the UTC date/time that the cookie was originally uploaded to the computer. Date 2 is the date/time that the cookie was last accessed.

Case study part two

In response to the defence report further forensic examinations were conducted by the original police Computer Crime Unit and two other experts. These identified the reasoning behind the defence claims.

The computer was operating on a Windows ME platform and the time zone information was set to an offset of Hex 0x00001e1 (+480 min) or Pacific Standard Time (PST). Despite having identified correctly that the computer's operating system was set to PST this had not been factored into the dates reported by one of the forensic software packages used by the defence expert.

The expert had used Encase V4 to interpret file time/date information and extract registry data including the time zone bias. In addition they had used the NetAnalysis software application to analyse Internet activity. This requires importing the index.dat files and setting the time zone bias. Unfortunately the expert had failed to configure Net Analysis in this way and their report therefore quoted times without subtracting the 8 h bias.

Fortunately after service of a further prosecution reports detailing these facts the defence expert wrote a second report retracting their allegations and correcting many of their fundamental errors. They conceded:

> 'There is now no evidence that the computer was operated in any way after the time it was seized...'

and

> 'The computer [id number] was being used to access Web pages associated with pornography, including some possibly indicative of child pornography between [times on a date before it was seized].'

A short time later the defendant pleaded guilty to the majority of offences charged.

Checklist for date/time evidence

The following suggestions may form the core of a checklist for date time analysis.

- Record the CMOS time on seized or examined system units in relation to actual time, obtainable using radio signal clocks or via the Internet using reliable time servers.
- Establish the computer's current time zone from the registry.
- Establish if daylight saving times may have an effect on the times relevant to the investigation.
- Identify the types of time structures that you are using and establish if they are displaying local time or UTC. Use a tool such as Dcode Date from Digital Detective for this, or better still write your own in order to increase your understanding of the subject.
- Look for corroboration in the form of additional times, dates and activities both on the computer and away from it that confirms your understanding of the dates and times you are interpreting.
- Test your results using the same operating systems and application versions that are present on the computer being examined.

Conclusion

From an ethical viewpoint this case has shown the importance of establishing exactly what is happing forensically before anyone, prosecution or defence, commit themselves to a line of reasoning or a strong opinion. It does not assist either side to make excessive attacks on the other's evidence. Recently a high profile case in the UK has led to one expert publicly criticising the other on a website. Is this behaviour ethically sound or constructive to the community as a whole? In many cases if the experts were to meet and discuss the case they would agree with a large section of each other's findings and avoid potential embarrassment both to individuals and the community.

The increased availability and diversity of easy to use forensic applications over the past few years is, in many ways, a good thing as it makes case work and reporting more efficient. This case shows that these applications cannot be a substitute for completely understanding the evidence yourself.

It is not the purpose of this article to highlight the errors that were made by the defence expert. More importantly it is to underline that experimentation and testing are the key to strong, reliable computer forensics. It may seem that this causes unnecessary

additional work and in many cases the initial report or statement may not require, or justify that. However, examiners will eventually be questioned in detail about forensic issues such as time/date evidence in a courtroom and there is no guarantee, as we have seen, that the forensic software alone will correctly interpret the raw data.

References and useful software

An article at 'The Code Project' entitled 'Beating the Daylight Savings Time bug and getting correct file modification times' gives an indication of the problems facing programmers dealing with date/time issues.

http://www.codeproject.com/datetime/dstbugs.asp?print=true

The Digital Detective contains many informative papers including some useful information on dates and times. It promotes several useful tools including Dcode Date (free) and NetAnalysis.

www.digital-detective.co.uk

Chris Boyd is a law enforcement officer working with the National Technical Assistance Centre (NTAC), Home Office, London. He currently specialises as a forensic technician working in the field of stored data recovery and analysis within the stored data laboratory.

Pete Forster is a detective with a background in commercial fraud, computer crime and forensic computing. He is currently seconded to NTAC where he manages the stored data laboratory.