

**TRACING FORENSIC ARTEFACTS FROM USB-BOUND COMPUTING
ENVIRONMENTS ON WINDOWS HOSTS.**

JAN COLLIE

A submission presented in partial fulfilment of the requirements of the University of
South Wales/Prifysgol De Cymru for the degree of Doctor of Philosophy

NOVEMBER 2015

ABSTRACT

This thesis proposes that it is possible to extract and analyse artefacts of potential evidential interest from host systems where miniature computing environments have been run from USB connectable devices. The research focuses on Windows systems and includes a comparison of the results obtained following traditional ‘static’ forensic data collection after conducting a range of user initiated activities. Four software products were evaluated during this research cycle, all of which could be used as anti-forensic tools - associated advertising claims that use of the software will either leave ‘no trace’ of user activity or no ‘personal data’ on a host system.

It is shown that the USB-bound environments reviewed create numerous artefacts in both live and unallocated space on Windows hosts which will remain available to the digital forensic examiner after system halt. These include multiple references to identified software and related processes as well as user activity in Registry keys and elsewhere. Artefacts related to program use and data movements will also be retained in live memory (RAM) and it is recommended that this is captured and analysed. Where this is not possible, relevant information originally held in RAM may be written to disk on system shut down and hibernation, opening further opportunities to the analyst.

This study builds on existing knowledge within digital forensic science and expands it in three ways. Firstly, it presents and explains a previously overlooked artefact which aids investigations involving the unauthorized use of both connected and connectable devices on Windows hosts. Secondly, it explores how portable virtualisation software interacts with host systems - a relatively uncharted field of enquiry. Finally, it informs research into antifoensics by showing that, despite its ability to cover and wipe its tracks, portable virtualisation software does leave traces of user-related activity on host systems which can greatly assist a digital forensic enquiry. By means of the methodology set out in this thesis, it is possible to uncover these traces in RAM dumps and by conducting a targeted analysis of static hard drives.

Table of Contents

CHAPTER 1.....	7
INTRODUCTION.....	7
1.1 Overview.....	7
1.2 The problem area.....	8
1.3. Scope of Study	10
1.4 Aims, Objectives and Hypothesis	13
1.5 Methodology overview	14
1.6 Original contribution to knowledge	15
1.7 Structure of the Thesis.....	15
1.8 Chapter Summary	17
CHAPTER 2	19
BACKGROUND	19
Table 1: Table of topic areas and relevancies to the research program.....	19
2.1 Virtualisation and Virtual Machines (VMs)	20
Figure 1: Virtualisation conceptualized.....	20
2.2 Virtualisation Technologies for USBs.....	22
2.3 Windows Memory Handling	26
2.4 Windows artefacts.....	28
2.5 Current Best Practice guides for forensic practitioners	29
2.6. Chapter Summary	32
CHAPTER 3.	33
EXPERIMENTS.....	33
3.1 Restatement of Hypothesis and aims	33
3.2 Research methodology	33
Figure 2: Research Methodology graphic.....	35
Figure 3: Test disk preparation	37
Figure 4: Test operating systems	38
Figure 5: Test vPC applications and Windows compatibility	39
Figure 6: Taxonomy of artefacts for analysis	43
Figure 7: Identification system for test vPCs & Windows OSs during experimentation	45
3.3 Experimentation.....	45
Figure 8: Experiment A1	46
Figure 9: Experiment A2	46
Figure 10: Experiment A3	47
3.4 Results overview	47
3.5 Chapter summary	49
CHAPTER 4.	51
THE ICONCACHE DATABASE	51
4.1 Background	51

4.2 Synopsis of findings	53
4.3 Research Method.....	55
Table 2. Forensic tools used during experimentation	56
4.4 Experimentation	56
Table 3. Details of experiments devised.....	57
4.5 Icon caching mechanisms in Windows.....	59
Figure 11: Icons stored in Shell32.dll	60
Figure 12: Shell32.dll and stored icon properties.....	60
Table 4 : Numbers of icons in the IconCache.db on first creation by version of Windows OS.....	61
Table 5: Basic file signature for IconCache.db in Windows XP, Vista and 7	62
Table 6: Sample differences in IconCache.db file signature in Windows XP, Vista and 7.....	62
4.6 Experimental results	63
Example A. Sample start of file, first generation of IconCache.db.....	64
Example B. Sample end of file, first generation of IconCache.db.....	64
Example C. Sample end of file, simple system restart following first generation of IconCache.db	65
Table 7: Example result showing increase in file size and icons retained in IconCache.db resulting from user action.....	65
Figure 13: Icons from application's 'autorun' icon present on DVD in IconCache.db	66
Example D: 'Autorun' icon present in an application on DVD listed in IconCache.db	66
Example E: Path to executable run from DVD in IconCache.db file.....	67
Example F: Record of executable run from host in IconCache.db file	67
Example G: Sample end of file after text document run from USB memory stick.....	67
Figure 14: Icons from executables present on USB memory stick in IconCache.db file	68
Example H: Listing of executables on USB connectable shown in ASCII within IconCache.db	68
Example I: Icons of Executable run from USB connectable shown in IconCache.db....	69
Figure 15: Icons from Limewire application, installed from USB connectable & run on host shown within IconCache.db file	70
Example J: Executable installed from USB connectable & run on host shown within IconCache.db file.....	70
Figure 16: Icons retained in IconCache.db following uninstall of Limewire program	71
Example K: First instance of IconCache.db for new user 'Max'	71
Table 8: Sample Date and Time changes to the IconCache.db in Windows XP	72
4.7 Conclusions.....	73
4.8 Chapter summary	75
CHAPTER 5.....	76

EXPERIMENTAL METHOD, CONDUCT AND RESULTS.....	76
5.1 Experimental method and conduct.....	76
5.2. Initial analysis	78
Figure 17a): MojoPac Filestructure.....	79
Figure 17 b): MojoPac Filestructure	80
Figure 18 a): Ceedo Filestructure	81
Figure 18 b): Ceedo Filestructure	82
Figure 18 c): Ceedo Filestructure.....	83
Figure 19: LupoPenSuite Filestructure	84
5.2.4 PortableApps file structure.....	85
Figure 20: PortableApps File structure.....	85
5.3. Baseline Experiments	86
Figure 21: USB device enumeration process in Windows (simplified).....	87
Figure 22: MojoPac icon in IconCache.db textual reference	88
Figure 23: Icon and textual references in IconCache.db following first run of MojoPac.....	90
Figure 24: Icon and textual references in IconCache.db following first run of Ceedo Personal.....	92
5.4 Test scenario Experiments	93
Figure 25: Use of Notepad ++ from within Ceedo Personal identified in the UserAssist key.....	95
Figure 26: Create and save of a document within PortableApps identified in selected ComDlg32 key locations.....	96
Figure 27: Artefacts from Firefox Portable run within PortableApps retained on host.....	97
Figure 28: Artefacts retained on host following text file movements using Ceedo Portable.	99
5.5 Further results	99
Table 9. Search terms for test vPCs in Windows XP & Windows 7 (32-bit)	100
5.6 Chapter Summary.....	100
CHAPTER 6.....	102
ANALYSIS AND DISCUSSION	102
6.1 Introduction.....	102
6.2 Baseline test results.....	102
Figure 29: Executable files in the root of the drive containing Ceedo	108
6.3 Scenario-based experimental results.....	110
6.4 Chapter summary	116
CHAPTER 7.....	118
SUMMARY AND CONCLUSIONS	118
7.1 Proof of Hypothesis.....	121
7.2 Contribution to knowledge.....	121
7.3 Further Work.....	124

REFERENCES.....	126
BIBLIOGRAPHY.....	135

CHAPTER 1.

INTRODUCTION

This chapter covers:

- USB memory sticks and virtualisation: data security issues
- An outline of the thesis
- The scope of study
- Research aims and objectives

1.1 Overview

Running a functioning computer environment from a memory stick has become more and more viable thanks to developments in desktop virtualisation technologies over the past decade. Although the computer environments concerned usually provide only a sub-set of the features which users would expect to have access to when working on desktop or laptop computers, they nevertheless allow for every-day activities to be carried out. These include playing games, making, moving and copying files and accessing the Internet. As well as portability, a number of these miniature systems – which will be termed vPCs (Virtual PCs) here – are said in advertising literature to offer the user strong confidentiality. Some manufacturers claim that use of their vPC system will leave no trace of activity at all on the host machine involved (Ceedo Technologies Ltd, 2014, MojoPac, 2009). Others state that no ‘personal data’ will be left behind following vPC use (Lupo PenSuite 2013, Portable Apps, 2014). These messages imply that secrecy as well as security is ensured.

Maintaining personal privacy is an important issue from an individual’s point of view; maintaining data security is equally important to companies. The proponents of vPC technologies are naturally keen to show how virtualised computing environments address both concerns. From an information security perspective, however, they could be viewed as a new threat, expanding the risk of data loss or network corruption already posed by the use of USB memory sticks in general (Tetmeyer 2010, Goucher, 2008) and modern ways of working such as BYOD (Garrity and Weir, 2010). For the digital forensic analyst, the use of vPCs presents a different challenge – one which this

thesis suggests is similar to that encountered when dealing with data wiping and encryption. In the hands of a wrongdoer, a vPC could become an antifoensics tool, its functionality being subverted in order to evade detection when carrying out unauthorised or potentially criminal activities. While evidence can be deliberately hidden or destroyed, however, traces of those actions can usually be found and can be beneficial to a digital forensic enquiry (Carlton and Kessler, 2013, Moe, Thorkildsen and Arnes, 2009). This research seeks to show that the same can be true for vPCs and that worthwhile investigative results may be obtained using standard examination techniques. More detailed results may be acquired by using advanced techniques such as live memory capture and analysis and preliminary research into this aspect of enquiry is explored in this thesis.

1.2 The problem area

The problem area identified in this research arises where two distinct technologies meet. The first technology is the USB connectable device, a piece of hardware, the second is virtualisation, which is software. The hardware is cheap, ubiquitous and extremely portable, the software is either free or very cheap and readily available for download from the Internet. The use of either of these technologies in isolation can impact on the security of host computer systems, as will be discussed. The use of both in combination, it is suggested, raises the bar for professionals involved in either preserving data integrity or investigating data security breaches.

The uncontrolled connection of USB devices to computer systems is widely acknowledged as a serious security threat (DiRenzo, 2012; Pham et al.,2010). USB connectable memory sticks –also known as thumb drives or keys – present a particular risk, both to sensitive data and the systems used to serve and store them. Small, low-cost and obtainable, they open up opportunities for both the theft and casual loss of valuable information. Viruses and other malicious software can also be introduced to stand-alone or networked computer systems via USB, whether deliberately or inadvertently (DiRenzo, 2012; Sharma, 2011). From the perspective of digital forensic analysis, there is a danger that antifoensic programs can be run direct from a USB key, helping wrongdoers to cover up or completely wipe out any traces of their activities (Thomas and Morris, 2008).

For all the above reasons, the analysis of artefacts left behind by the use of USB connectable devices has become an integral part of computer investigations in corporate and criminal cases. As technology advances and the storage capacity of USB memory sticks grows - the highest capacity USB3 stick currently¹ available offers a full terabyte of usable space - the security problems associated with them increase. A miniature computer environment can be run from a USB stick with as little as 4 GB onboard memory, for example. In today's market place, 32 GB capacity sticks are commonplace and cost as little as £12. These high capacity devices are also capable of delivering high read/write speeds, allowing users to quickly access and transfer data.

Virtualisation technologies have risen in popularity because of the benefits they offer. From a corporate perspective, virtualisation allows companies to provide services to clients and helps them to facilitate their own business agility. The technologies most commonly adopted - network, storage and server virtualisation - have both advantages and disadvantages from the point of view of security (Kim, G, 2007). At the same time, virtualisation is evolving and becoming available to the general public. Special 'slimmed down' portable applications such as word processors, spreadsheets and web browsers have been developed for use from USB sticks, making it possible for users to work from any available computer. In the same timeframe, the pocket PC has become a reality thanks to the invention of miniaturized computing environments which incorporate both a self-contained operating system and a user area for applications – a format which effectively reproduces the workstation experience.

The development of virtual machines (VMs) and other applications specifically intended for use direct from a USB key began around the year 2000, closely following on the introduction of the high-speed USB2 interface. In 2005, a team of researchers at Stanford University, U.S.A. developed what they termed a 'LivePC'. This was a piece of software which allowed users to put one or more miniature VMs onto a portable device such as an iPod or memory stick. Stanford's LivePC, called Moka5, consisted of two separate components: a virtualized operating system and a user

¹ Kingston Hyper X USB 3, released 2013

environment containing applications. Once running, Moka5 offered a full environment from which applications could be used as normal.

The release of Moka5 and the emergence of similar software gave rise to the notion of the 'PC In Your Pocket'. In 2008, the idea of carrying a personal computing environment around on nothing bigger than a USB stick was a novelty and it was greeted with enthusiasm by the magazine PC World, which devoted five pages to a feature explaining the technology involved (Taylor, K, 2008). Security was a focal point of this article, the accent being laid on the ability to use a USB-bound VM to access the Internet without fear of virus infection - a VM can be recreated in moments so the core system is never lost. However, other security considerations, notably the danger of such miniature VM users being able to carry out online actions anonymously, were not taken into account. In fact, it was stressed that both VMs and portable applications could be used without leaving any 'data files or other traces' on the host machine, a feature important to those wishing to protect personal data security.

The boast of user anonymity, which is repeated by the developers of the miniature environments considered in this thesis, merits serious concern by anyone involved in corporate security or law enforcement, particularly since the software concerned is either free or extremely cheap and is readily available for download from the Internet. In the interests of informing digital forensic science and practice, therefore, the author decided to determine whether the stated claims were true via the research and experimentation presented in this thesis.

In sum, the **hypothesis** of this research is that it is possible to extract and analyse artefacts of potential evidential interest from system files that have been created on Windows hosts by miniature computing environments running from USB devices.

1.3. Scope of Study

This thesis offers an assessment of the placement and analysis of those artefacts of potential evidential interest which can be retained on computers running Windows operating systems following the use of miniature computing environments from a connected USB stick. It is a focused assessment and there is no intention to imply that every avenue of enquiry in this research area has been explored. Ample opportunity exists for further work to be carried out in this field and some suggestions are made in Chapter 7.

A number of research areas are touched on in this enquiry, importantly, the analysis of artefacts retained on Windows-based computers following the use of USB connectable devices. A significant amount of research has been carried out in this field (Carvey, 2005 and 2009, Mee and Jones, 2005, Mee et al, 2006) and this is drawn upon here. However, a previously undocumented source of further information was identified during this research. This information is kept in the IconCache database - a file which is normally hidden from the computer user but which is accessible to the digital forensic examiner. Examination of this artefact helps to clarify what programs may have been run from a USB key together with any associated file paths and the user name utilised for the activity concerned (Collie, 2013).

The capture and analysis of volatile memory (RAM) is also considered in this thesis. The value of this capability for the purposes of digital forensic analysis has been recognised for many years (Solomon et al, 2007, Petroni et al, 2006, Casey and Seglem, 2004). Since the technique is carried out on live computers, it raises issues in respect of the forensically sound collection of evidence. This is because some device must be linked up to the computer under examination in order to receive the captured data. Inevitably, such a link up changes the state of the target machine. Concerns over this consideration have meant that the standard approach to computer analysis remains the collection of data from static systems. This method, which is colloquially known as 'Pull The Plug', involves taking the power supply out of the back of a live desktop computer before accessing the hard drive. Changes to files and running processes are thus prevented. While arguments exist to support both means of evidence collection, live memory capture is now seen as an imperative for network and malware investigations as well as live response (Anson et al, 2012, Casey, Malin and Aquilina, 2012).

The Virtual Machine (VM) as evidence is a subject which has been explored by Brett Shavers (2008) and he has noted that the use of a VM will tend to leave artefacts on the host system. The focus of Shavers' work is on the use of VMs which have been installed on a host computer rather than run from an external device. While he has drawn attention to the fact that VMs can be run from removable media and disposed of after use, hindering the investigative process, this aspect of research has not been developed further. Barrett and Kipper (2010) also looked at the use of VMs, including some miniature environments, and monitored the changes made to a host system by use of the software. The results for the miniature VMs showed that, for Windows XP, traces of activity – notably the names of the programs involved - were retained in certain Registry keys. Evidence of network protocols being opened was also found during live testing. This thesis seeks to extend the above research by further exploring the Windows Registry and analysing memory dumps, page files and other artefacts recovered from live and static systems running Windows XP and Windows 7, both by Microsoft.

The miniature environments considered in this research are desktop virtualisations. The applications chosen for testing fall broadly into two categories: Virtual Machines and Portable Applications. Both are designed as standalone programs which will run on compatible computers without being installed. Virtual Machines allow for those applications which are installed within the provided environment to interact with one another. A picture created in one software package can be placed in a document created in another software package, for example. This facility differentiates them from Portable Applications environments, in which the various software packages made available are designed to run separately from each other (Ceedo, 2010). A third category of virtualisation, the LiveUSB, does not form part of this research project. This choice has been made because a LiveUSB consists of a portable device which contains a bootable operating system. Most LiveUSB systems are designed to run before the host computer's operating system boots. The tools evaluated here are only available to a user after the Windows operating system boots. Thus they interact with the host system, creating the potential for traces of user activity to be left behind. Whilst the Moka5 LiveUSB does run following the launch of Windows on the host, the software is now only available for commercial use. It has not, therefore, been

possible to test recent editions of the software. A further technology which may have been of interest to this enquiry was the U3 but this was discontinued in 2009. The U3 was a USB-bound technology developed by SanDisk and Microsoft which gave users access to a range of applications, including web browsers. Although the U3 was supposedly designed to leave ‘no trace’ of either user data or application usage on the host computer once it was removed, researchers found that artefacts of potential interest to digital forensic examiners could be retrieved (Bosschert, 2007; Tank and Williams, 2008).

During this research, experiments were carried out using four examples of virtualisation software. All were available for download from the Internet at the time of writing. The operating systems used for testing were: Windows XP (32 bit) and Windows 7 (32 bit). For the purposes of comparison, testing was also carried out on Windows 7 (64 bit) systems. Throughout the period of study and experimentation for this thesis, Windows XP and Windows 7 have been the most popular family of operating systems in use (W3schools.com, 2014, Net Applications, 2013). Although Windows 8 was released in 2012 and gained ground in the market place, Windows XP maintained a respectable following. Microsoft ceased support for XP on April 8, 2014 but some argue that it’s popularity will take time to diminish, even in the private sector (Casey, 2014).

1.4 Aims, Objectives and Hypothesis

The **aim** of this research project is to develop a methodology which informs the forensic analysis of Windows computers where portable virtualisation software is thought to have played a part in the unlawful or unauthorized access of an host system. The methodology also aims to assist the investigation of cases in which the potential unlawful/unauthorized use of USB connectable devices in general exists.

The **objectives** of this research are:

- 1) To explore the potential for portable computing environments to be used to commit unlawful acts in the context of existing concerns regarding the use of USB connectable devices in general.
- 2) To examine selected miniature computing environments and analyse the effects of carrying out a range of common activities via these on an host computer.
- 3) To consider currently available methods of analyzing Windows computer systems for evidence of activity involving a user-connected USB device
- 4) To create a chart of key areas to be examined and analysed when investigating the use of miniature computing environments and/or USB connectable devices.

The **hypothesis** of this research is that it is possible to extract and analyse artefacts of potential evidential interest from system files that have been created on Windows hosts by miniature computing environments running from USB devices.

1.5 Methodology overview

A series of procedures have been devised in order to meet the identified objectives of this research. First, a test environment was set up. This consisted of:

- a) a research computer to host test hard disks.
- b) a set of hard disks to receive test operating systems.
- c) a set of USB keys to receive test vPCs.
- d) forensic hardware and software for disk sanitation, disk cloning and data analysis.

Next, two sets of experiments were developed. The initial set were baseline experiments which were designed to ascertain the types of artefacts which would remain on test Windows operating systems following:

- a) the connection of a blank USB memory stick.
- b) the connection of a USB stick bearing a miniature computing environment (vPC).
- c) connecting a vPC and launching the software.

The object of this exercise was to compare outputs from the experiments so that a dataset of interest (DSOI) could be isolated and used to inform further work. A second set of scenario-based experiments were then developed, the intention being to reproduce some of the basic activities which the user of a vPC would, in the view of

the author, likely wish to carry out after introducing it to a host machine. These activities were:

1. Copy a text file ; vPC to host.
2. Copy a text file ; host to vPC.
3. Copy a picture file: vPC to host
4. Copy a picture file; host to vPC
5. Write and save a text file on the vPC.
6. Run a program executable on the vPC.
7. Launch a browser on the vPC.
8. Conduct a search from a vPC-based browser.

All experiments in the two sets were to be carried out for each test vPC application in the context of each compatible test operating system. For each experiment, data was to be obtained from live RAM and then the static host hard drive following system halt. Analysis would afterwards be conducted using proprietary forensic software.

1.6 Original contribution to knowledge

This study builds on existing knowledge within digital forensic science and expands it in three ways. Firstly, it presents and explains a previously overlooked artefact which aids investigations involving the unauthorized use of both connected and connectable devices on Windows hosts. Secondly, it explores how portable virtualisation software interacts with host systems - a relatively uncharted field of enquiry. Finally, it informs research into antifoensics by showing that, despite its ability to cover and wipe its tracks, portable virtualisation software does leave traces of user-related activity on host systems which can greatly assist a digital forensic enquiry. By means of the methodology set out in this thesis, it is possible to uncover these traces in RAM dumps and by conducting a targeted analysis of static hard drives.

1.7 Structure of the Thesis

The second chapter of this thesis aims to give the background to the research to be conducted. Aspects of five topic areas that are relevant to the project are introduced and discussed. Previous work carried out within these topic areas by other researchers

is reviewed and summarised and the ways in which this project will advance on existing studies are indicated.

The research methodology for this project is set out in detail in the third chapter and the experiments that support it are defined and described. The hypothesis is reiterated. The methods, tools and techniques which are used during testing, data collection and analysis are identified and any limitations are stated. A brief summary of the results obtained during experimentation is also given.

The fourth chapter focuses on the IconCache database, an artefact previously undocumented in terms of its usefulness to digital forensic investigation. The basic workings of the IconCache.db are explored together with the circumstances under which information comes to reside in it. The evidential potential of the IconCache.db is demonstrated, in cases where the unauthorized use of USB connectable devices is suspected in general and in cases where the unauthorised use of vPCs is suspected in particular.

Chapter five fully details the design and implementation of the experiments conducted during this research cycle. The data obtained during experimentation is condensed and reproduced. The most informative results in terms of digital forensic investigation are reported and considered and brief comments are made.

A thorough scrutiny of all relevant experimental outputs is undertaken in Chapter six. Discussion points are raised which centre around the findings made during analysis. These include not only those artefacts which were discovered but also those which were not discovered, despite a reasonable expectation that they might be present on a host system following the experimental actions taken. Possible rationales for the findings are provided.

The seventh and final chapter summarises the entire project, briefly reiterating the problem area and recapping on the design and implementation of the experiments. The aims and objectives of the research program are repeated, the hypothesis is restated and it is explained how the work which has been carried out explores,

evaluates and substantiates it. The project's contribution to the field of digital forensic science is considered and suggestions for further work are made.

1.8 Chapter Summary

This chapter introduced the subject matter of this thesis. The context for the project is set, firstly by giving an overview of the data security challenges currently posed by the use of USB connectable devices and secondly by discussing virtualisation technologies, in particular miniature computing systems which can be run from a memory stick. Cheap and easy to set up and highly portable, the USB-bound miniature computing environment, here termed the vPC, is said to offer a high level of privacy to the user. Manufacturers of the vPC software to be examined during this study claim that either no data at all or no 'personal' data will be left on host computing systems after their use. The implication is that secrecy is ensured – a concept which raises concerns for data security professionals, law enforcement agencies and digital forensic analysts since it suggests that vPCs could be used as anti-forensic tools. A problem area which encompasses the risks associated with USB devices in general and USB-bound virtualised computing environments in particular is thus identified. Aspects of this problem area have been addressed by previous authorities and this earlier work has shown that the use of USB devices will leave artefacts of potential evidential interest on host systems. It has also been found that traces of user actions are likely to be retained on hosts even when evidence has been deliberately hidden or destroyed. The hypothesis is therefore advanced that it should be possible to extract and analyse artefacts which could benefit a digital forensic enquiry from system files that have been created on Windows hosts by miniature computing environments running from USB devices.

In order to explore the hypothesis of this thesis, certain objectives need to be met. These include examining selected vPC environments, the effects of introducing them to Windows hosts and the effects of then carrying out common activities. Current methods of analysing Windows computers for evidence of USB activity must also be

reviewed. The overall aim of this research project is to develop a methodology which both informs the forensic analysis of Windows computers where portable virtualisation software is thought to have been used and assists the investigation of cases in which the unlawful or unauthorized use of USB connectable devices is suspected. To that purpose, the following chapters will fully detail the background to the project, expand the explanation of the problem area, describe the research methodology and the experiments which were devised and conducted, report on and discuss all findings made and conclusions drawn, submit that the hypothesis has been proved and suggest further work which could usefully be done in this field of study.

CHAPTER 2

BACKGROUND

This chapter is intended to set the context of this research program by discussing five main topic areas, aspects of which relate to the problem area identified for study. The order of presentation of the topics does not imply an order of importance. Rather, it has been chosen with the aim of providing the reader with the background information necessary to understand the project and the reasons for formulating it. The five main topics and their relevance to this thesis are given in Table 1, below. Those topics marked in orange pertain to virtualisation technologies, those in pink to relevant elements of the Windows operating system and digital forensic practice.

Table 1: Table of topic areas and relevancies to the research program

Topic		Relevance
1.	Virtual Machines (VMs) and virtualisation.	a) To introduce virtualisation as a concept and give a general understanding of current virtualisation technologies. b) To discuss the implications of virtualisation in terms of digital forensic investigation.
2.	Virtualisation technologies for USB memory-stick devices.	a) To present a brief history of virtualisation software for USB devices. b) To discuss the virtualisation software packages chosen for testing in this study and those excluded.
3.	Windows memory handling	To give an overview of how computers running current Windows operating systems handle memory and manage live data.
4.	Artefacts created by the use of USB devices on Windows systems.	a) To give an account of previous work carried out in this area of enquiry. b) To show how this thesis will extend this knowledge base.
5.	Current Best Practice guides for forensic practitioners.	To summarise existing guidelines for the forensically sound capture of data from digital devices, to explain their limitations and to discuss their influence on current practice.

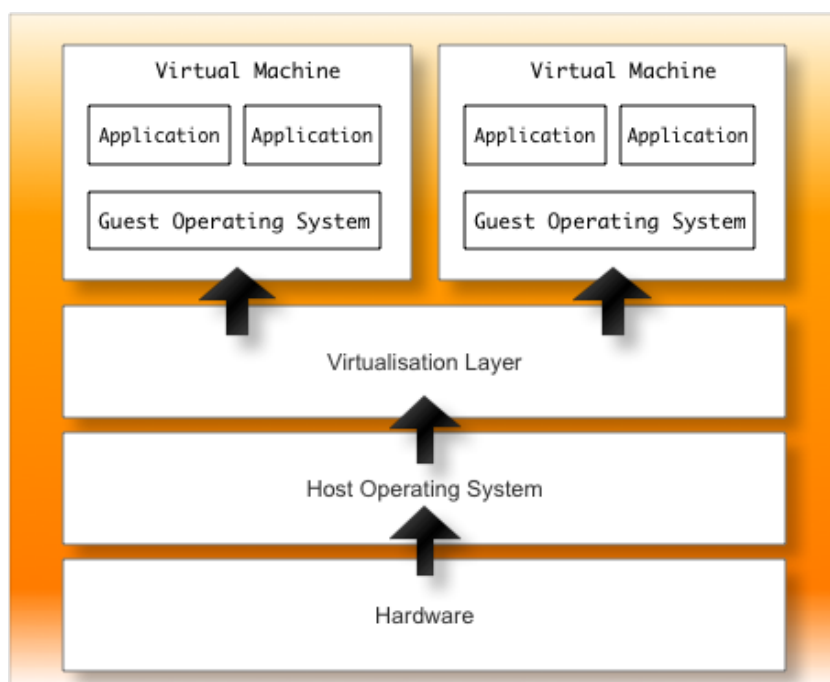
2.1 Virtualisation and Virtual Machines (VMs)

2.1.1 Overview

Virtualisation involves the creation of an entity e.g. a computer environment which appears real but is in fact a simulation running on a host system. Virtualisation software can be used, for example, to allow one computer to run multiple operating system images at the same time. An example of this type of virtual environment (Virtual Machine) will be used and studied in this research, albeit in a miniaturized format.

The term ‘Virtual Machine’ is normally used to refer to VM software, also known as a ‘hypervisor’ or ‘virtual machine monitor’. This type of software makes it possible to perform multiple identical executions on one computer. (Kietzman, S, 2010). The software itself creates what is known as a virtualisation layer which sits on top of the operating system being used by the host machine, as conceptualised in the following diagram:

Figure 1: Virtualisation conceptualised



In the wider world of commerce, virtualisation can be seen as part of an ‘overall trend in enterprise IT which includes autonomic computing and utility computing.’ Haynos, M (2008). Virtualisation is increasing in three main areas of IT: network, storage and server virtualisation. As Kim, G (2008) explains, ‘Virtualisation makes it possible to build and deploy IT releases and changes into production faster and more economically than ever before.’ However, he warns, this capability also has many disadvantages, notably where security controls have not been properly implemented in virtualised environments. The security controls which Kim focuses on are configuration settings and changes at the virtualisation layer. Physical security controls, including preventing the introduction of hardware and software are not considered in his white paper.

2.1.2 The forensic perspective

The treatment of the Virtual Machine as evidence has been considered by Brett Shavers (2008). He has also looked at the use of the VM as a tool to aid ‘antiforensics’ i.e. as a means of masking or erasing the actions carried out by someone gaining access to a computer for unethical reasons. The main focus of Shavers’ paper is on the analysis of artefacts from VMs which have been resident on a computer during its service life. However, he does note that VMs can be used from removable media and disposed of after use, hindering the investigative process. The opportunity afforded to wrongdoers by USB-bound virtualisations is explored comparatively briefly by Shavers and this research project proposes to extend and expand his work in this area. From the point of view of the digital forensic analyst, Shavers draws attention to a number of file names which are likely to be left on a host machine following activity involving a VM. He points out, though, that recovering these, ‘...will typically not yield the contents of the data residing in the VM.’ Nevertheless, finding these traces may open other avenues of enquiry, for instance, the investigator might be prompted to look for the suspect VM on other available media. Shavers also indicates that the use of a VM will tend to leave artefacts on the host operating system e.g. in the system Registry, Lnk files and Prefetch files. It is envisaged that artefacts of this type will inform this research project.

In common with other authorities, for example Penhallurick, M (2005) and Sammes, T (2007), Shavers discusses the use of VM as a tool for forensic analysis, though in much less detail. This aspect of VM technology is outside the scope of this research although it is a topic of great interest.

Barrett and Kipper (2010) have also explored the use of VMs and the types of artefacts which may be found on a computer system as a result of any related activity. Two miniature VM environments were included in the authors' work and the changes made to a host system by use of these pieces of software were monitored. The results for the miniature VMs showed that, for Windows XP, traces of activity – notably the names of the executable files for the programs concerned - were retained in certain Registry keys. Evidence of network protocols being opened was also found during live testing. This project seeks to extend Barrett and Kipper's research by analysing other virtualised environments and by examining memory dumps, page files and other artefacts recovered from live and static systems running Windows XP and Windows 7.

2.2 Virtualisation Technologies for USBs

Two types of technologies will be considered in this research:

1. The miniature VM environment
2. Portable application software.

A handful of companies have brought VMware designed for portable storage devices to the market over the past five years. Two of the first applications were Moka5 and MojoPac, both of which were free when first released. These innovative software packages offered a familiar computing environment to users whilst freeing them from a given work space. This gave rise to the notion of the 'PC In Your Pocket', a concept explored in some depth by the magazine PC World. In an article which explained the technology involved, it was stated that the introduction of high-speed USB2 interfaces on the latest computer workstations had made it possible to run VMs and applications direct from USB connectible devices (Taylor, K, 2008). The functionality and security of the technology involved was discussed, the security aspect revolving

around the ability to use a VM to access the Internet without fear of picking up a virus infection. This is possible because, once built and configured, a VM can be recreated quickly and easily over and over again. The core system is never lost. However, other aspects of security, such as the danger of miniature VMs being used to carry out malicious acts anonymously either in the work place or online, was not taken into consideration in this article. On the contrary, it was stressed that both VMs and portable applications could be used without leaving any ‘data files or other traces’ on a host machine. At the time the article was published, both Moka5 and MojoPac could only be run if the user had administrator privileges on the host machine. This would have afforded a certain amount of security. That situation soon changed.

In 2008, the virtualisation technology offered by Moka5 enabled the user to create a portable PC – known as a ‘Live PC’ - based on the Linux operating system. The software, which was created by a team at Stanford University, U.S.A, in 2005, was based on the free VMware Player. In fact VMPlayer had to be installed onto the host system before a Live PC could be run. Once running, the Live PC offered a full environment from which applications could be used as normal. Moka5 is now only available as a commercial product and in the USA, therefore it will not be tested during experimentation for this project.

Other types of LivePC , also largely based on variations of the Linux operating system, continue to be freely available for use. These technologies consist of a bootable operating system which can be contained on a USB or DVD. Where they differ from Moka5 is that the operating systems concerned do not depend on Windows i.e. they boot independently and before the Windows operating system. This being the case, the software does not interact with Windows and will not leave any artefacts within the Windows operating system. This style of virtualisation has therefore been excluded from this study. A further exclusion is a technology known as U3. The U3 was a USB-bound miniature computing environment which gave users access to a number of applications, including web browsers. Developed by SanDisk and Microsoft, U3 created problems with certain Windows operating systems, notably Vista, and work on it was discontinued in 2009.

The miniature environments considered in this research are designed to run on Windows PCs via a USB connectible drive. Two with strong similarities are MojoPac and Ceedo. To use Ceedo's description of the technology, it, 'enables a suite of standard Windows applications to run on a host PC without installation'. This type of virtualisation gives the user a full workspace, complete with applications, application settings and user data. One of the advantages of this kind of virtualisation is that applications which are run within the environment can interact with one another in the way that they do on Windows PCs. This differentiates them from portable applications, which run separately from each other. As a white paper from Ceedo (2010) explains, 'Standard application virtualisation technology implements a virtualisation layer between each application and the operating system. A problem with this approach is that each virtualized application is isolated such that it cannot interact with other applications in the same environment. This means, for instance, that it is not possible to embed an Excel spreadsheet into a Word document.'

The white paper further notes that portable applications 'do not use the Windows registry, do not store files in the standard Windows folder, and do not use any 3rd party ActiveX components or COM objects'. Portable applications are written in a way, 'which will not use any Windows persistent services.'

This observation has also been made by Bem, D and Heubner, E (2007), who state that Windows is not designed to be portable: it relies on the Windows registry, installs or uses already installed dynamic libraries (DLLs) and stores files and profiles in various system folders. Consequently, they add, in order to make a portable application, a software developer needs to write it in such a way that it does not use the Windows registry nor store its files anywhere on the host computer.

The Windows operating system has been updated since Bem and Heubner presented their findings and, since the introduction of Windows 8 in 2012, a portable version of Windows has become available. The portable system is called 'Windows To Go' and is only available in the Enterprise version of the software. In order to run normally, Windows To Go requires the USB device that it is held on to have a minimum

capacity of 32 GB. It must also be highly specified in terms of read/write speeds (Microsoft, 2013).

2.2.1 Test miniature environments

The preceding paragraphs in this section describe those types of currently available technologies which may be termed miniature computing environments. The applications chosen for testing fall broadly into two categories: Virtual Machines and Portable Applications. The environments chosen for testing in this research are:

- 1: MojoPac
- 2: Ceedo Personal
- 3: LupoPenSuite
- 4: Portable Apps

Of these, MojoPac may be considered a miniature VM in that it emulates the functionality of a particular computer system – Windows XP. Ceedo Personal is a virtualisation software which runs on a number of Windows hosts. It may be considered as offering application virtualisation in that it encapsulates the application software it contains, allowing it to run as if it were installed on the host operating system whilst actually keeping it separate. In this sense, Ceedo Personal presents in much the same way as LupoPenSuite and Portable Apps, both of which can be categorised as portable applications.

2.2.2 The promise of privacy

For each of the above environments, direct references to the high level of personal privacy that a person can expect to enjoy when using the software are made in promotional literature. Two of the strongest claims are made by the manufacturers of MojoPac and Ceedo. The advertisement for MojoPac, for example, claims: “Your MojoPac is completely private and secure. Your applications, your browsing history and your data is never left behind on the PC it is connected to. What happens on MojoPac stays on MojoPac” (Appendix 1.1). While that for Ceedo Personal states it: “has zero-footprint, meaning once you plug-out your USB nothing from Ceedo Personal is left behind on the host PC.” (Appendix 1.2)

For the portable applications Lupo PenSuite and Portable Apps, the supporting literature is more carefully worded. For LupoPenSuite, the following is published within the FAQ section:

“A portable application is a computer software that you can carry around with you on a portable device, use it on any Windows computer and when you unplug the device none of your personal data is left behind.”²

Likewise, the PortableApps site defines what a portable app is:

“A portable app is a computer program that you can carry around with you on a portable device and use on any Windows computer. When your USB flash drive, portable hard drive, iPod or other portable device is plugged in or your cloud drive is synced, you have access to your software and personal data just as you would on your own PC. And when you unplug the device, none of your personal data is left behind.”³

The phrases ‘your data’ and ‘your personal data’ may, of course, be interpreted in a variety of ways. Nevertheless, there is a clear implication that a user’s personal privacy will be protected.

2.3 Windows Memory Handling

Operating systems handle system memory in a highly complex way. An in-depth explanation goes beyond the scope this thesis. Instead, areas of Windows operating systems which may generate or hold artefacts of evidential interest as a result of a memory handling process will be explored together with those particular processes.

In overview, the term ‘memory’ is generally used to refer to RAM (Random Access Memory). It is not the same as ‘storage’, which is normally taken to mean the capacity of a computer’s hard drive (Kingston Technologies, n.d.). RAM is used to hold temporary instructions and the data needed to complete tasks. Information is put into RAM by the operating system on an *ad hoc* basis. In simple terms, information which

² <http://www.lupopensuite.com/faq.htm>

³ http://portableapps.com/about/what_is_a_portable_app#guidelines

is being used in real time is swapped in and out of the volatile RAM memory so that the operating system can multi-task. The information is held in files known as page files. As Russinovich M., and Solomon, D. (2005), explain more technically, page files are used to store modified pages which are still in use by some processes but have had to be written to disk. Whilst much of the data held in RAM will be lost when the computer is switched off, therefore, some can remain on the system in the form of page files.

Russinovich and Solomon provide a thorough discussion of Windows memory management, showing both how Microsoft Windows implements virtual memory and how it manages the subset of virtual memory kept in physical memory. They explain that the Windows memory manager consists of several components which deal, amongst the things, with the allocation, reallocation and management of virtual memory. It is responsible for handling the paging process and for managing the size of the page file.

An important aspect of paging files is that they cannot be deleted while the computer system is running. Furthermore, if the system has not been configured to clear the page file at shut down, a quantity of the data placed there will be retained by the system and can afterwards be viewed. From the point of view of forensic examiners, therefore, paging files may be of interest.

Research on the contents of RAM has shown that although the contents of page files are held in volatile memory, large segments are unlikely to survive more than 5 minutes (Solomon et al., 2007). Nevertheless, small segments may be retained up to 2 hours after the data is committed to memory.

The value of capturing and analysing live memory during digital forensic investigations has been recognised for many years (Solomon et al, 2007, Petroni et al, 2006, Casey and Seglem, 2004). However, there is no standard approach to extracting this data. Instead, there are a number of methods, as discussed by Ruff (2007). The analysis of volatile data is a developing field but it is possible to identify rootkits,

Trojans and other viruses using the techniques evolved thus far (Schuster, 2006, Carvey, 2007).

2.4 Windows artefacts

A range of artefacts are created on Windows systems whenever a USB device is plugged in, notably in the Registry. A number of researchers have shown that the Registry is a rich resource for digital investigation, in particular Carvey, H (2005); Carvey and Altheide (2005), Mee, V and Jones, A (2005) and Mee et al (2006). These authorities have focused on the traces left by USB storage devices on computers running the Windows XP operating system. The same or similar artefacts have since been identified in newer versions of the Windows operating system e.g. Windows Vista and Windows 7.

Carvey (2009) has noted that when a USB device is first connected, the Windows plug and play manager elicits a device descriptor from it. It then finds a driver for the device and stores the information in the setupapi.log file. Once this action is complete, a key is created in the HKEY local machine portion of the Registry at:

HKEY_LOCAL_MACHINE\System Current\ControlSet\Enum\USBSTOR

A sub key is also created, which identifies the specific class of device in the following format:

DiskandVen_***andProd_***andRev_***

where *** represents manufacturer and product information data gleaned from the device by the Windows operating system.

An unique instance ID is created from this device class ID. This is usually the serial number of the device in question. If more than one device of the same class are connected e.g.. Two identical USB keys, each will nonetheless have unique IDs on the host system. Where the USB device does not have a serial number, Windows' Plug-

and-Play (PnP) Manager will create a unique instance ID for it. This ID will consist of an alphanumeric string, of which the second character is an ampersand.

Carvey also notes that a second repository of information regarding USB devices is the MountedDevices key. This records the drive letter assigned to each device connected to the host system e.g F:\. The Windows system may also store traces of USB connection activity via Link (.lnk) files, Shortcuts and the Prefetch folder. Such artefacts can help a forensic analyst trace file-related activity, such as opening a document on or copying a picture to a particular device (Roy and Jain, 2012). Other artefacts are created in the Windows Registry when programs are executed. The UserAssist key, for example, stores information about recently used and frequently used programs and the number of times these programs have been launched (Stevens, 2010). If a program is run from a USB connectable device, therefore, it is possible that a record of that action will be kept by the system in this Registry key. Using a new application on a Windows system can also populate the MUICache Registry key (NirSoft, 2011), so this key may also contain useful information.

The prior research mentioned above has all concentrated on the role of the Windows Registry in locating artefacts from USB connectable devices. The research conducted during this thesis has extended the field of enquiry by investigating the forensic potential of the IconCache database. This artefact, which lies outside of the Registry, has been shown to retain file paths to programs and processes which have run on both fixed and attached drives such as USB devices. These activities can be associated with individual user names which have been set up on the host computer. Thus, where the unauthorised or covert use of systems and programs is suspected, analysis of the IconCache database can also prove useful (Collie, 2013).

The research reviewed in this area has all focused on the artefacts created on Windows systems when USB devices are connected.

2.5 Current Best Practice guides for forensic practitioners

‘Best Practice’ procedures for handling computer evidence and mobile phone evidence have been considered by a number of authorities, mainly those concerned with law

enforcement. In the UK, the Association of Chief Police Officers (ACPO) has published a well-known set of guidelines which are primarily aimed at serving officers but also apply to investigators and practitioners of digital forensics in the private sector. The ACPO guidelines, originally approved in 1999, were updated and republished in 2012.

The Principles of Digital Evidence, as laid out in the Guide are:

Principle 1: No action taken by law enforcement agencies, persons employed within those agencies or their agents should change data which may subsequently be relied upon in court.

Principle 2: In circumstances where a person finds it necessary to access original data, that person must be competent to do so and be able to give evidence explaining the relevance and the implications of their actions.

Principle 3: An audit trail or other record of all processes applied to digital evidence should be created and preserved. An independent third party should be able to examine those processes and achieve the same result.

Principle 4: The person in charge of the investigation has overall responsibility for ensuring that the law and these principles are adhered to.

In common with other published guides in this subject area, for example, First Responder reference guides published by the U.S. Department of Justice and the U.S. Secret Service, the ACPO guidelines advise on how digital equipment should be handled in order to best preserve evidence. The accent is on not losing or inadvertently spoiling digital evidence during the seizure of equipment, in particular, at the scene of a crime. The majority of the guides written for law enforcement agencies do not cover the subsequent analysis of data although the principles laid down for the handling of items of evidential interest are relevant at all stages of investigation, from crime scene to laboratory. The latest version of the ACPO guide now contains a brief section on analysis, giving views on who should carry out such analysis and the need for analysis to be properly targeted towards gathering evidence relevant to the case in hand.

A document which goes into much more depth is the Directors' and Corporate Advisors' Guide to Digital Investigations and Evidence published by the Information

Assurance Advisory Council (IAAC) in 2009 (Sommer, P, 2009). This details relevant legal issues involved with the analysis of computer systems, media and mobile phones as well as their collection and preservation . Advice is also given on a ‘Corporate plan of action’ for digital incidents, although this paper does not go into data analysis procedure.

The Guidelines on PDA Forensics published by the National Institute of Standards and Technology (NIST) in 2004 gives a comprehensive introduction to PDA technology and goes on to describe an entire procedure, from the seizure and handling of PDAs through to their examination and analysis, detailing the location of evidence and the use of hardware and software. It also advises on subsequent reporting. This guide is focused entirely on PDA forensics, however, and does not consider other digital media. Likewise, the NIST Guidelines on Cell Phone Forensics (2007) is a comprehensive treatise on the seizure, handling and forensic examination of mobile phones alone. It may nevertheless be considered an excellent primer for anyone interested in the forensic investigation of mobile phones.

The guides mentioned above are chiefly concerned with maintaining the forensic integrity of digital data as evidence. Further resources have been made available to the digital forensics examiners in terms of where to look for evidence during analysis. For example, ‘quick guides’ for USB key forensics on Windows XP, Vista and 7 systems have been written by Rob Lee (2009) and made available through the SANS Institute website. (Appendix I a) A similar guide has been published online by the consultancy firm, Arsenal Recon (Appendix I b).

An important aspect of digital forensic examination not mentioned in the older guides reviewed here but covered in ACPO (2012) is the acquisition and analysis of volatile data. Although the process has been known for over a decade (Crescenzo et al., 1999), it is not commonly used because the technique requires prerequisite knowledge and training. Furthermore, it raises issues in respect of the forensically sound collection of evidence. As a result, the standard approach to computer analysis remains the capture of static systems, or what is colloquially known as ‘Pull The Plug’, since it involves pulling the power cable out from the back of a running computer. There are

arguments to support both methods but while live memory capture is now seen as an imperative for network and malware investigations as well as live response (Anson et al, 2012, Malin et al, 2012), law enforcement agencies tend to avoid it. One possible explanation, at least in the UK, may be a conservative attitude based on Principle 2 of the ACPO guidelines - it could be difficult for an officer to explain both the necessity and implications of his/her actions to a Judge and Jury, especially in the face of clever questioning supported by legal argument from an opposing barrister. There would also be no way of conforming to Principle 3 of the ACPO guidelines as far as the acquisition of volatile data was concerned . The process could not be repeated by a third party.

2.6. Chapter Summary

This chapter aims to set the context of this research program by identifying and discussing five headline topics, all of which relate to the problem area identified for study.

These topics include:

- Virtual Machines (VMs)
- Virtualisation technologies for USBs
- Windows memory handling
- Windows forensic artefacts
- Current Best Practice guides for forensic practitioners.

CHAPTER 3.

EXPERIMENTS

This chapter restates the hypothesis of this thesis and the aims of the overall research project. The research methodology for the project is set out in detail and the experiments are described.

3.1 Restatement of Hypothesis and aims

The **hypothesis** of this project is that it is possible to extract and analyse artefacts of potential evidential interest from system files that have been created on Windows hosts by miniature computing environments running from USB devices. The **aim** is to develop a methodology which informs the forensic analysis of Windows computers where portable virtualisation software is thought to have played a part in the unlawful or unauthorized access of an host system. The methodology also aims to assist the investigation of cases in which the potential unlawful/unauthorized use of USB connectable devices in general exists.

3.2 Research methodology

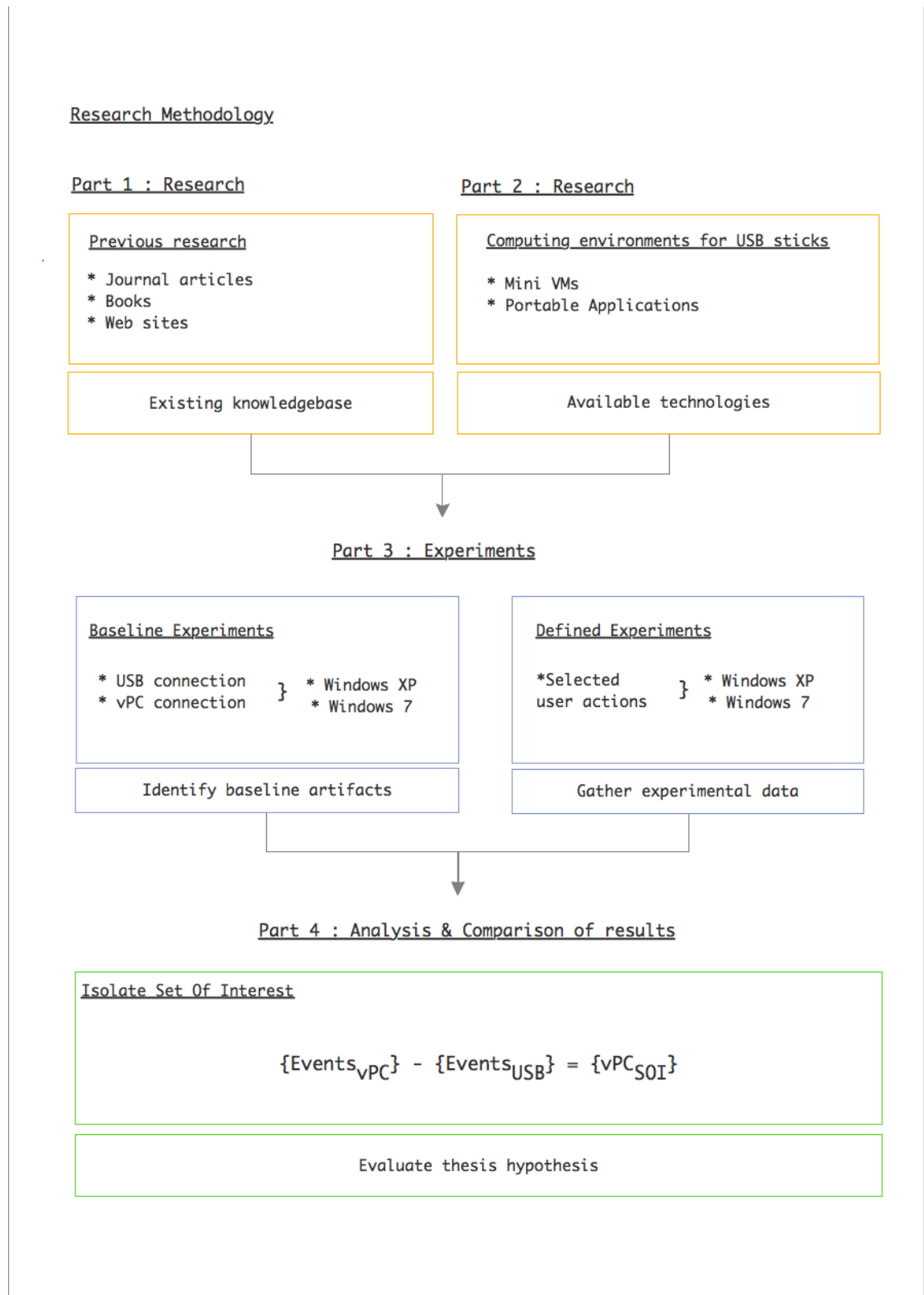
The purpose of this research is to isolate information of potential evidential interest where a miniature computing environment has been introduced to a Windows host via a USB connectable device. To this end, a methodology has been devised which is based on both reading research and experimental research. The reading research, described in Chapter 2, aims to show the current extent of knowledge which lies within the areas pertaining to the central research question of this thesis. The experimental research is described in the following subsections.

Briefly, three baseline experiments are carried out in order to identify the types of artefacts which will remain on the test operating systems following:

- a) the connection of a blank USB memory stick.
- b) the connection and use of a USB bearing a miniature computing environment (vPC).

Thereafter, a series of defined experiments are carried out in order to collect data. A Data Set Of Interest (DSOI) is isolated during the analysis stage. A comparison of results then takes place, against which the thesis hypothesis can be evaluated. A graphic expression of the research methodology is shown in Figure 2, below.

Figure 2: Research Methodology graphic



3.2.1 Test environment

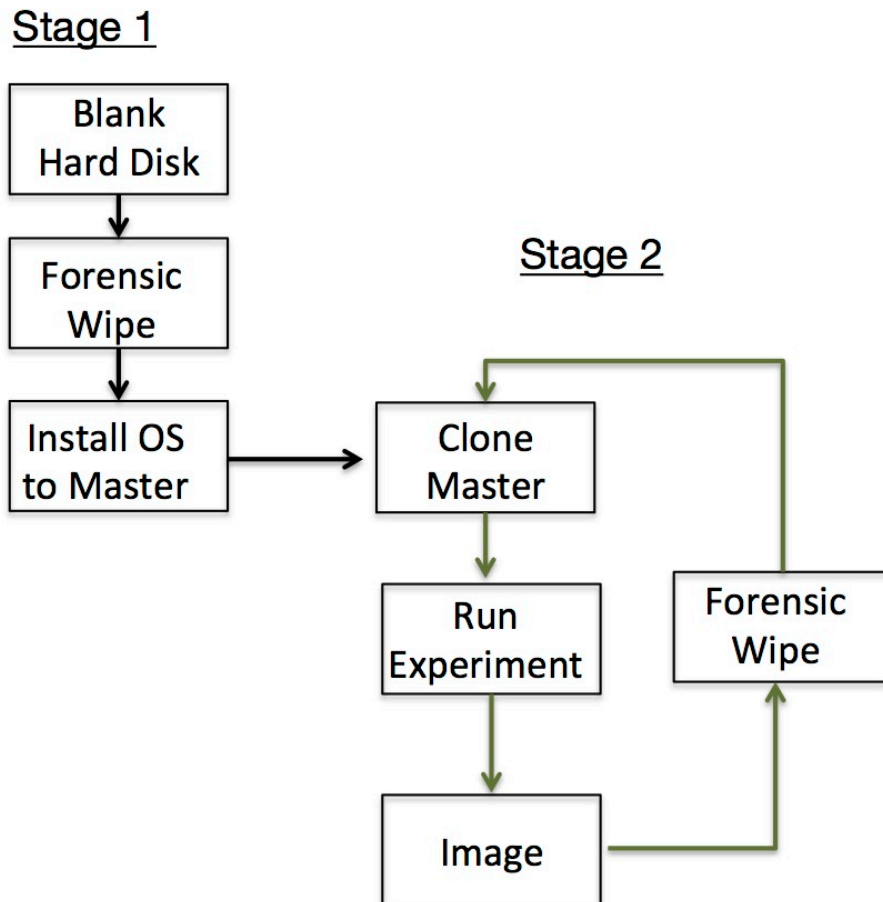
The physical hardware used was a single PC workstation with an Intel Celeron 64 bit processor (E3400 @ 2.60 Ghz), 4 GB of RAM and a standard VGA card. The computer was not initially connected to any network.

A formal test host environment was established to ensure that the same experiment could be repeated consistently using different versions of the Microsoft Windows operating system. This was achieved as follows:

A clean install of each test operating system was made onto a set of 250GB hard disks which had previously been wiped using standard forensic hardware. This was done to eliminate potential data contamination during operating system installation and use. The hard disks were also wiped between each experimental phase. Each test operating system installation was set up to run using UK English and with the time set to GMT London. A single user name and computer name was used throughout. Once created, the individual test systems were cloned to other previously sanitized disks. The latter were then used for experimentation (Figure 3).

The hardware used for wiping and imaging was: Logicube Talon and for cloning: Logicube SuperSonix.

Figure 3: Test disk preparation



3.2.2 Test Design Conditions

In order to control sources of variation during experimentation, the test environment was designed to be as uncomplicated as possible. Each OS install was created direct from an installation disk. No patches or updates were installed. The utility Process Monitor was installed onto each disk used during the preliminary testing phase for auditing purposes (See 3.2.6.3, below). No additional programs or applications were installed. With a view to replicating the type of scene a digital forensic examiner might be called into, for example, where a small to medium enterprise suspected IP theft or the introduction of malware to a computer system, the following assumptions are made for each test carried out:

- That the ‘suspect’ workstation is running when the examiner enters.
- That no enterprise solution exists to allow live system monitoring.
- That the workstation under review is the only evidence source available to the examiner.

3.2.3 Test Operating systems

For the purposes of this research, 32 bit and 64 bit versions of two Microsoft Windows operating systems were examined: Windows XP Pro, and Windows 7 Pro (Figure 4). These were chosen because, during the research period, Windows XP and Windows 7 accounted for more than 80% of all operating systems in use whilst Windows 8 accounted for only 1.77% of the market share (StatCounter, 2012; w3schools, 2012). In April 2014, when Microsoft stopped support for Windows XP, the operating system remained the most popular in terms of internet usage in Europe, South America, Asia and Africa (StatCounter, 2014). At that time, Windows 7 was reported as the leading global operating system, accounting for 54.7% of the market share. At the time of writing, therefore, it is likely that computer examiners will commonly be analyzing versions of these operating systems.

Figure 4: Test operating systems

Windows XP Pro SP3	
32 bit	64 bit
Windows 7 Pro	
32 bit	64 bit

3.2.4 Miniature computing environments to be tested

The miniature environments tested are shown in Figure 5, together with their compatibility with the test operating systems under review. Illustrations showing how the different environments appear within a Windows environment are presented in Appendix III.

Figure 5: Test vPC applications and Windows compatibility

Type of vPC	Application	Test OS compatibility			
		Win XP		Win 7	
		32	64	32	64
Mini VM/AV*	MojoPac v. 2.1.10	Y	Y	N	N
	Ceedo Personal v. 5.0.1.7	Y	Y	Y	Y
Portable App	Lupo PenSuite v. 2013.04 Lite	Y	Y	Y	Y
	Portable Apps v.11.2	Y	Y	Y	Y

* = Application Virtualisation

The technologies are briefly described further below:

a) **MojoPac**

MojoPac is a virtualisation which presents a desktop environment which closely emulates Windows XP. A user has a personal profile, files and folders and can load applications e.g. Microsoft Office into the environment from a standard installation disk. In common with a full version of Windows XP, MojoPac has its own Registry and shell. A user also has access to USB devices connected via the host. MojoPac runs independently and in isolation from the host operating system. In these ways, MojoPac acts like a virtual machine and, it is suggested, may be considered a miniature VM.

First launched in 2006, MojoPac was developed by RingCube Technologies and can be used on computers running Windows XP. Development was discontinued around 2012 after RingCube was acquired by Citrix but the software is still available for download from the Internet.

b) **Ceedo Personal**

Ceedo Personal from Ceedo Technologies Ltd is one of a suite of products based on the manufacturer's own workspace virtualisation technologies. The software presents as a taskbar at the top of a computer user's desktop.

Specially adapted applications can be loaded into this workspace from the Ceedo website. Host computer services such as access to USB ports and DVD drive can be accessed from within the Ceedo workspace.

As a software technology, Ceedo may be considered an application virtualisation i.e. the program encapsulates the application software from the host operating system on which it is executed. Applications e.g. Firefox are not installed on the host but they run as if they were. Although it has much in common with portable application software, it is more sophisticated in its operation. Privacy settings are available to the user, the default privacy settings being to keep any temporary files created during use within Ceedo and to clear temporary files on exiting the software.

c) **Lupo PenSuite** and

d) **Portable Apps**

Both these programs can be classified as portable applications – they are designed to run on a compatible computer without being installed. Both offer designated folders for user files e.g. MyDocs, MyMusic. Cut-down versions of popular applications such as browsers and word processors can be installed into the environment. Neither Lupo PenSuite nor Portable Apps offer privacy settings.

3.2.5 Data collection

For static systems, data collection was carried out by attaching a write-blocked imager to the host hard disk. Volatile data was collected by introducing forensic software to the host system via a USB connectable memory stick. The forensic software used was: FTK Imager Lite by Access Data and RamCapturer (32 & 64) by Belkasoft.

3.2.6 Data analysis

The data to be analysed was present in RAM dumps and in specific areas of the Windows operating system. These are detailed below and a classification of artefacts is presented in Figure 6, which follows at the end of this subsection.

3.2.6.1 RAM data

RAM captures were analysed using HBGary Responder Community Edition v. 2.0.2.1438. Keyword searches for the names of the software in use and related processes were carried out on the memory dumps obtained.

3.2.6.2 Static systems

For each experiment, the analysis of data collected from static test systems consisted of scrutinizing five main areas of the Windows operating system for artefacts. These areas, which were identified based on research and working knowledge, were: Registry, Prefetch, Link (.lnk) files, IconCache.db and Pagefile. In the Registry, up to twelve keys likely to retain artefacts as a result of USB-related activity were checked. In a real-life situation, an examiner would pay attention to the finer detail of the dates and times associated with such activities, correlating information gathered from Registry keys with that to be found, for example, in system event and setupapi logs.

The software used for analysis was FTK v 5.1.

3.2.6.2.1 The IconCache.db

During this research, it was discovered that artefacts of potential evidential interest are retained in the IconCache.db, a hidden file which is created and maintained on computers running Windows operating systems. This led to an extended investigation of the database and its workings. Chapter 4 is devoted to detailing the experimentation carried out during this phase of research and the findings made. Notable results were published in the academic journal 'Digital Investigation' in 2013 (Appendix IV). This research on the IconCache.db was later extended by others and the original paper was cited extensively in another article published in the same journal in 2014 (Appendix V).

3.2.6.2 Preliminary System monitoring

Preliminary system monitoring was carried out using the utility Process Monitor v.2.94⁴, developed by Windows Sysinternals. Process Monitor is a tool which

⁴ Available from: www.technet.microsoft.com

monitors and displays file system activity on computers running the Windows operating system as the activity occurs in real time. This tool identified the ports which are opened and the processes which run following the connection of particular vPCs. These findings helped to inform the analysis to be carried out.

3.2.6.3 Recording findings

Experimental outputs were recorded into a table devised for the purpose of collecting and collating results. (Appendix VI) This consisted of a listing of Registry keys which are known to retain artefacts following the connection of a USB device plus other areas of the operating system, for which the same is true. It also contained a column for noting down any artefacts found in RAM.

Following initial experimentation, it was found that a number of Registry keys retained similar information e.g. the name of the vPC executable which had been attached to the host and run. A sub-set of seven key system locations were found to yield the most detailed artefacts. These are discussed in Chapter 5.

Figure 6: Taxonomy of artefacts for analysis

STATIC SYSTEM			
1.	Registry HKey		
1.1	SYSTEM\CurrentControlSet	\Enum\USB	
		\Enum\USBSTOR	
1.2	SYSTEM	\Control\DeviceClasses\{ <i>alphanumeric</i> }	
1.3	SYSTEM	\MountedDevices	
1.4	SOFTWARE\Microsoft	\Windows Portable Devices\Devices	
		\Windows\CurrentVersion\Explorer	
		\MountPoints2	
		\Windows\CurrentVersion\Explorer	
		\UserAssist	
		Windows\Shell	
		Windows\ShellNoRoam	
		Windows\ShellNoRoam\MUICache (Win XP)	
		\Windows\StreamMRU	
1.5	SOFTWARE\Classes	\LocalSettings\Software\Microsoft	
		\Windows\Shell\MuiCache	
		(WinVista, Win7)	
1.6	UsrClass.dat\Local Settings	\Software\Microsoft\Windows\Shell	
2.	Prefetch		
3.	Lnk files		
4.	IconCache.db		
5.	Pagefile (s)		
LIVE SYSTEM			
Artefacts in Live Memory (RAM)		Unstructured. Various search techniques appropriate	

3.2.7 Test procedure

Two main tests were carried out, the first to ascertain what artefacts remained from experimentation following a memory dump taken from a live system, the second to ascertain what could be gathered from the same system, once static. The results were then compared.

In the interests of brevity, only the most useful results out of a total of 112 outputs are reported in this thesis.

3.2.7.1 Test scenarios

A series of scenarios were developed with the aim of mimicking a set of basic general activities the user of a vPC would, in the view of the author, likely wish to carry out.

These were numbered as follows :

1. Copy a text file ; vPC to host.
2. Copy a text file ; host to vPC.
3. Copy a picture file: vPC to host
4. Copy a picture file; host to vPC
5. Write and save a text file on the vPC.
6. Run a program executable on the vPC.
7. Launch a browser on the vPC.
8. Conduct a search from a vPC-based browser.

Each of these activities were carried out for each test vPC application in the context of each compatible test operating system. A system of identifying the vPC and OS used for each experiment was devised (Figure 7). In this, each vPC and OS were designated a letter of the alphabet and the experiments to be carried out were numbered. Thus, a shorthand emerged. For example, the vPC Ceedo, designated letter B, tested against Windows XP Pro (32bit), designated letter W, for experiment 1 becomes experiment BW1. The same vPC tested against Windows 7 (64 bit) for experiment 1 becomes experiment BZ1.

A form was created using this system in order to record the results obtained during experimentation and correlate them with the record sheets filled out during analysis. This is produced in Appendix VII.

Figure 7: Identification system for test vPCs & Windows OSs during experimentation

vPC name	vPC Designation:	Windows OS:	OS Designation:
MojoPac	A	XP Pro (32)	W
Ceedo	B	XP Pro (64)	X
Lupo PenSuite	C	7 Pro (32)	Y
Portable Apps	D	7 Pro (64)	Z

3.3 Experimentation

3.3.1 Baseline experiments

Three baseline experiments were first carried out, as follows:

- a) Introduce clean USB key into the host system, nothing more (Experiment A1 – Figure 8).
- b) Introduce USB key containing vPC executable into the host system, nothing more (Experiment A2 – Figure 9).
- c) Introduce USB key containing vPC executable into the host system. Run vPC (Experiment A3 – Figure 10).

For b) and c) above, experiments were carried out for each combination of vPC and test OS.

Thereafter, a first phase of experimentation involved testing four (4) applications in two (2) versions of Windows XP for each of the eight (8) experimental test scenarios outlined in 3.2.7.1, above – a total of 64 outputs. A second phase involved testing three (3) applications in two (2) versions of Windows 7 – a total of 48 outputs.

Figure 8: Experiment A1

A1

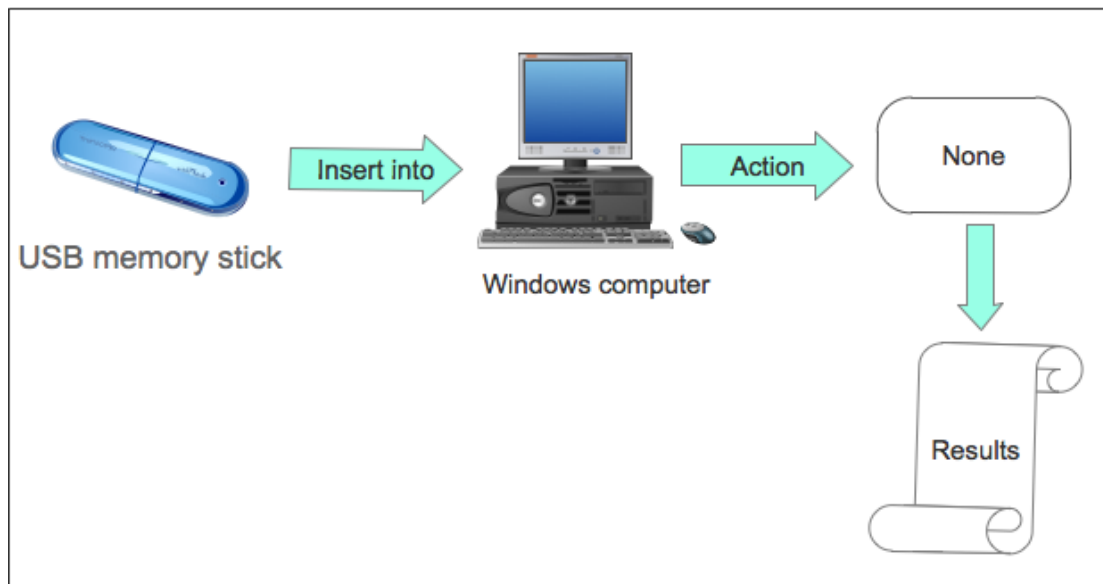


Figure 9: Experiment A2

A2

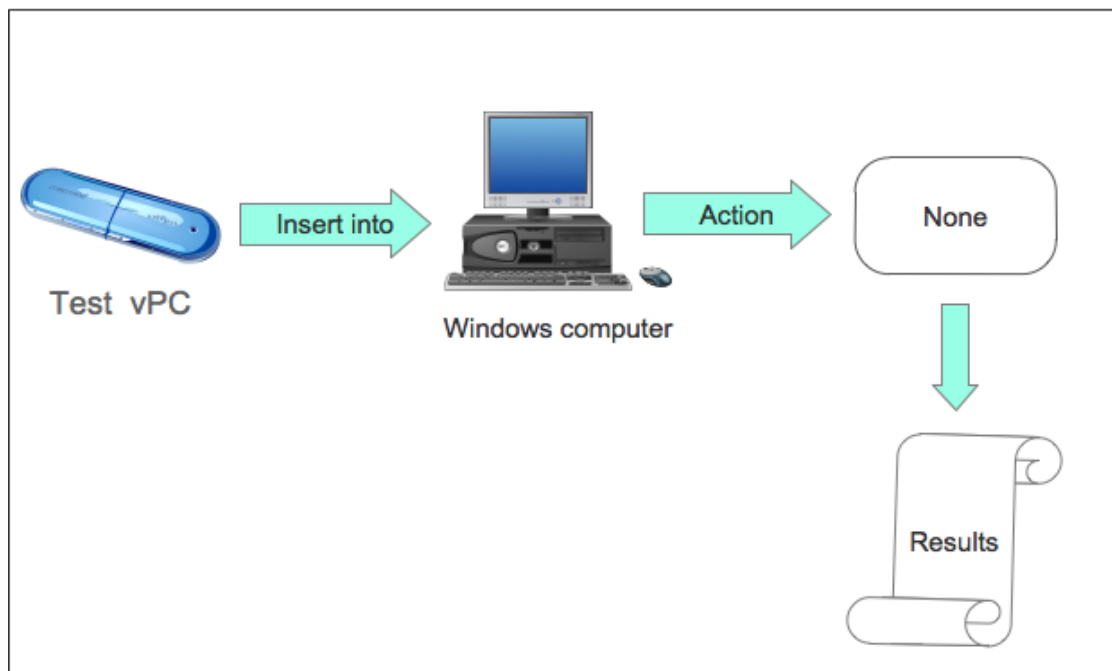
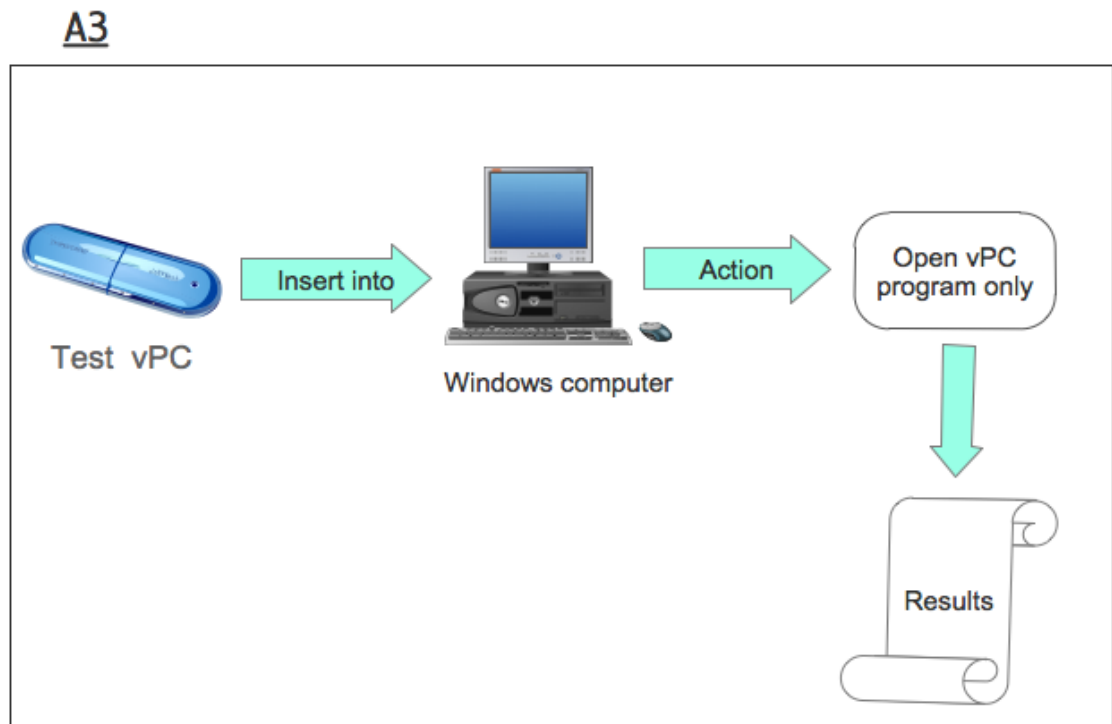


Figure 10: Experiment A3



3.4 Results overview

3.4.1 Baseline experiments

For the three baseline experiments and for every combination of vPC and OS, artefacts related to the attachment of the USB drive – such as the drive letter allocated to the device, its type and its serial number - were to be found in the Registry at:

```
HKEY_LOCAL_MACHINE\SYSTEM\CURRENTCONTROLSET\ENUMUSBSTOR  
“  
\ SYSTEM\CURRENTCONTROLSET\ENUMUSB
```

This result was expected since the USB enumeration process during which the host machine reads a connected device’s descriptors, loads the appropriate drivers for it and configures the device for use, occurs automatically in Windows. It was also possible to find a record of the first connection of the USB devices in log files, specifically

'setupapi.dev.log' in Windows XP and 'setupAPI.dev.log' and 'setupapi.app.log' in Windows 7.

For Baseline b), no artefacts relating to the name of the vPC stored on the drive were found in the Registry or elsewhere. For MojoPac alone, both the related icon and textual references to the executable file were to be found in the IconCache.db.

For Baseline c) a large number of further artefacts, which identified the vPC being used, were located in the Registry, IconCache.db and elsewhere. A table of the most useful 'quick reference' locations was assembled and is shown in Appendix VIII.

3.4.2 Test scenarios

It was found that the introduction and use of USB-bound vPCs on Windows hosts can create numerous artefacts of interest to digital forensic examiners. The most informative of these will be found in Registry keys as well as in Link files / Shortcuts, Prefetch and the IconCache database. At a minimum, analysis of these artefacts will enable an enquirer to establish the name of the vPC environment invoked, the user name under which it was introduced to the host and which programs were run from within it, together with relevant dates and times, the drive letter allocated to the containing USB key plus details enabling identification of that key, such as the make and serial number. All of this information is available when a computer has been closed down using the traditional 'pull the plug' method. The Pagefile may be a further resource on static systems.

This research has shown that the connection of a vPC does not preclude the Windows registry from retaining information which helps identify the container drive. Once a vPC is run, the icon associated with its executable file will be stored in the IconCache.db and the program name and its associated file path will be documented there.

For some vPCs, the names of files created and saved within the miniature environment are retained on the host, together with the file path. Folders temporarily created on the host when a portable browser is used from a vPC and which are afterwards automatically deleted may also be visible within forensic software. This type of

finding could further usefully inform a digital forensic investigation. Where the collection of live memory is possible, this can reveal the names of files copied between a vPC and the host although file paths may not be available. Further research is needed in order to establish whether more pertinent artefacts could be gleaned from the contents of virtual memory for this and other user related activity.

While results from pagefile analysis during this round of research did not reveal anything of great note, further testing might produce something worthwhile. The host systems considered were running the native OS alone, placing limited demands on memory. Also they were only run for short periods, therefore there was little time for artefacts to accumulate in the pagefile. Further research could also establish whether, in common with malware and encryption keys, artefacts of potential interest relating to the use of vPCs may be retained in a computer's hibernation file.

3.5 Chapter summary

This Chapter restates the hypothesis of this project, namely that it is possible to extract and analyse artefacts of potential evidential interest from system files that have been created on Windows hosts by miniature computing environments running from USB devices (vPCs). The aim of the research is to develop a methodology which informs the forensic analysis of Windows computers where portable virtualisation software is thought to have played a part in the unlawful or unauthorized access of an host system.

The research methodology is described, together with the test environment, conditions and procedures. The miniature computing environments and operating systems to be tested are identified. The experiments carried out are detailed, as are the way in which outputs are recorded and analysed.

An overview of the results obtained is given. It was found that the introduction and use of USB-bound vPCs on Windows hosts can create numerous artefacts of interest to digital forensic examiners. At a minimum, analysis of these artefacts will enable an enquirer to establish the name of the vPC environment invoked, the user name under which it was introduced to the host, the drive letter allocated to the containing USB key plus details enabling identification of that key, such as the make and serial

number. For the majority of vPCs tested, it was also possible to ascertain which programs were run from within the given environment, together with relevant dates and times,

An important discovery made during this research program was that artefacts of potential evidential interest are retained in the IconCache.db, a hidden file which is created and maintained on computers running Windows operating systems. A paper on this aspect of forensic enquiry was published in the academic journal 'Digital Investigation' in 2013 and has since been cited. Chapter 4 details the experimentation carried out during this phase of the research and the findings made.

The VM has been defined and its use as a computing environment has been described. The potential for the use of VM software to leave artefacts of possible evidential interest on a host machine has been discussed, leading to the notion that the same may be true of miniature VM software which is now available and can be run from a USB connectable memory stick. Equally, other virtualisation technologies which provide a cut-down computing environment from USBs may cause artefacts to be created on a host. Those which can be described as portable applications and which will be examined in this study have been reviewed.

An overview of how the Windows operating system handles memory has been given. The types of artefacts which the every-day use of a USB connectable device will create on a host machine have also been considered. These elements of the discourse encapsulate the background knowledge which currently informs digital forensic analysis in this area. The capture and analysis of volatile data is also looked at. It is postulated that although this can be a valuable source of case-relevant data, it is often left untapped because it requires evidence gathering techniques which are unfamiliar to many digital forensic operatives. At the same time, both legal strictures and the tight protocols recommended in 'best practice' guidelines cause law enforcement agencies to shy away from using the methodology. As a background to this argument, a digest of best practice guides is presented.

CHAPTER 4.

THE ICONCACHE DATABASE

Experimentation carried out during this research project led to the discovery of a previously undocumented resource in terms of digital forensic investigation. This resource is the IconCache database, a hidden file which is created and maintained on computers running Windows operating systems. This chapter details the findings made in relation to the IconCache database (IconCache.db) and its usefulness to the digital forensic examiner, in particular when tracking activity from USB connectable devices which have been introduced to a Windows-based system by some user. This makes the analysis of the IconCache.db an essential element of any investigation into the suspected use of vPCs. As will be shown, the IconCache.db retains the names of program executables and their file paths, thus linking the associated activity to an individual user name on the computer under review.

4.1 Background

On Windows systems, programs, files and folders are represented by icons - small symbolic pictures which provide a visual clue to what may be accessed by clicking on them. However, the desktop presented to the computer user would be impossibly cluttered if the system displayed every icon that it could. Having to retrieve all possible icon images from disk and render them, time and again, would also consume system resources unnecessarily. As a result, Windows systems save icons already retrieved e.g. from programs which run automatically at start, in a cache in memory (Holderness, 1999). Before the introduction of Windows XP, the icons associated with processes and programs that either run in the background or are otherwise initiated on Windows-based systems were stored in a hidden file at:

C:/Windows/ShellIconCache

The maximum number of cached icons which could be cached in the ShellIconCache file was set in the Registry at:

HKEY_Local_Machine\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell
Icons

Valid values were: 100 - 4096 The Default value was: 500

With the launch of Windows XP, a new mechanism was introduced - the IconCache.db file. The IconCache.db is created by Shell32.dll, a core system file which is responsible for a number of key operations when a Windows computer is running (Russinovich and Solomon, 2005). The IconCache.db is a hidden file and Folder Options must be configured to show hidden files before it will be revealed to a computer user. The file is stored in different locations, depending on the version of the host operating system. For the purposes of the research discussed here, samples of three Microsoft Windows operating systems were examined: Windows XP, Windows Vista and Windows 7. The operation of IconCache.db in Windows 8 lies outside of the scope of this enquiry however, it is noted that certain changes have been made to what the IconCache.db stores in this version of Windows.

Caching mechanisms such as the IconCache database have been discussed by systems specialists in terms of their role in Windows memory management (Russinovich and Solomon, 2005). However, the investigative potential of such mechanisms had not been explored before Collie (2013).

This part of research project gives an overview of how the IconCache.db comes into being and how it accumulates artefacts of potential evidential interest. A full explanation of the workings of the icon caching mechanism in Windows goes beyond the remit of this section of the research program. The aim is chiefly to examine the artefacts produced and retained in the IconCache.db file when a USB device containing an executable file is connected to a Windows host. The effects of introducing and running executable files from a DVD drive and from the host itself are also considered.

4.2 Synopsis of findings

The IconCache.db is a hidden file stored in different locations, depending on the version of the operating system (OS). Since a separate IconCache database is maintained for each named user of a computer, artefacts in this file can be attributed to a specific user account.

If the IconCache.db file does not already exist on a Windows system, it is automatically created at system startup and populated with a number of default system icons. The cache of icons exists in memory but will also be written to disk following a Windows shutdown or restart.

The IconCache.db grows as information is added to it by the processes and activities which occur on the machine. The changes which take place in the IconCache.db include both the addition of new visible icons and the addition of file paths to the programs or processes associated with those icons in the text portions of the data base.

Of particular interest is the information which appears in the IconCache.db when a removable USB device or CD/DVD is connected to a host system and its contents are viewed. If the removable media contains one or more executables at the root of the drive, then any associated icons are added to the database. The file path, including the drive letter associated with the removable concerned, is also retained in the database in the following example format:

E:\Program_Name.exe

It should be noted that this action occurs whether any such executable is run or not, which means that readings obtained from the IconCache.db should be interpreted with care, ideally being corroborated against other findings. Nevertheless, the database can be an useful aid to the digital analyst, especially where attempts have been made to conceal suspect activities, such as the use of a vPC, an external encryption program or the introduction of malware via some USB device or optical media. Importantly, since these types of programs can be used on a computer without being installed, the IconCache.db may retain the only record of that activity.

The IconCache.db can also help digital examiners in cases where a potentially suspect program e.g. Limewire has been installed on a system via removable media and later

uninstalled. While few traces of that activity may remain on the host, a record, complete with textual details of both the origin of the installation and the path to its location on the host, will be kept by the database.

In brief overview, the results from experimentation showed that for the operating systems under review:

a) The IconCache.db is:

- Created on system reboot following a clean OS install
- Altered on system restart
- Recreated on system reboot, if deleted
- Altered by creating a link to a Windows pre-loaded executable on the host desktop
- Altered by running a Windows pre-loaded executable on the host
- Altered by installing an executable from a DVD
- Altered by inserting a USB connectable thumb drive which contains an executable
- Altered by running an executable from a USB connectable thumb drive
- Altered by opening a file e.g. text from a USB connectable thumb drive
- Altered by writing a file out to a USB connectable thumb drive
- Altered by creating a new user account on a system
- Created for each individual user on a system

b) When changes occur in the IconCache.db:

- artefacts of potential evidential interest are viewable in Hex and readable in ASCII .
- program icons new to the system e.g. from executables on external media, are stored in the database.
- icons for .dll files extant on the host system are stored in the database once fetched by the system.

c) Changes to the IconCache.db:

- are shown in increased file size.

- affect the file's Created/Modified/Accessed dates and times.

d) Dates and times:

- The 'Created' date and time of the IconCache.db file will reflect either:
- the first date the file was ever created on the system i.e. after clean OS install and reboot
- or the last 'Modified' date of the file, where it has been deleted in a previous session and is then recreated by the computer when the system is rebooted.
- The 'Created' date does not change when content is added during the duration of a user session.
- The file's 'Modified' dates reflect the time of the last system shut down.

4.3 Research Method

A formal test environment was established to ensure that the same experiment could be repeated consistently using the different versions of the Microsoft Windows operating system. The physical hardware used was a single PC workstation with an Intel Celeron 64 bit processor (E3400 @ 2.60 Ghz), 4 GB of RAM and a standard VGA card. The computer was not initially connected to any network.

Two 250 GB hard disks were securely wiped (Table 2) and used interchangeably in the following way: Firstly, the test OS was installed and a single user account was set up on the system. Experimentation, data collection and analysis followed. To eliminate potential data contamination or inconsistency during operating system installation and use, the hard disks were wiped before first use and between all experimental phases.

The primary operating systems tested were Windows XP Home (SP2), Windows Vista Home Basic (32 bit) and Windows 7 Home Premium (32 bit). Preliminary testing was also carried out on Windows 7 Home Premium (64 bit).

In all cases, Windows was set up to run using UK English and with the time set to GMT London. To ensure consistency, a single user name and computer name was used throughout.

The forensic tools used to conduct this research are summarized in Table 2, below. For each experiment, data was collected using a USB memory stick, formatted via Windows command line.

Table 2. Forensic tools used during experimentation

Hardware/software	Purpose
Logitech Talon	a) Hard disk secure data deletion (WipeClean ⁵ method) b) Write-blocked hard disk capture
Wiebetech USB Writeblocker	Write-blocked attachment of USB devices to a forensic workstation host.
FTK Imager Lite by Access Data	Imaging test USB devices.
Forensic Tool Kit (FTK) v 3.1 & 3.2 by Access Data	Analysis of data
decWindows Thumbnail Database Viewer v. 1.8 by Dec Software ⁶	Render icons during IconCache.db analysis

4.4 Experimentation

Based on the common or expected functionality found across all the Windows operating systems, three experiment stages were derived. These were: the clean install of an operating system as a starting point to baseline the experiments, initial interaction with the operating system and finally the extended or repeated user activity. In accordance with these stages and the objectives of this research, the experiments summarized in Table 3 were formulated.

Results from the experiments listed in Table 3 are summarized in the following sections. In the interests of brevity, specific examples of findings recorded in this section relate to Windows XP only since the same basic behaviour patterns were observed during the experiments on Windows Vista and Windows 7 32 bit systems.

⁵ Writes a pattern over the whole target drive $2x n+1$ times where n is the selected number of iterations of all 0s and 1s. The last pass then writes random values to every byte of the drive. (Source: Logicube Talon manual).

⁶ Available from: <http://www.thumbnailexpert.com/en/products/commercial-tools/dec-windows-thumbnail-database-viewer/1/1>.

Table 3. Details of experiments devised

Set		Purpose	Activities
A: Baseline IconCache.db Research		Establish file provenance.	
	A1		Ascertain: <ol style="list-style-type: none"> 1. When the IconCache.db file is created. 2. Its contents immediately following creation. 3. Its contents following a system restart. 4. What happens if the IconCache.db file is deleted
	A2	Further research on IconCache.db to corroborate initial state.	Ascertain: <ol style="list-style-type: none"> 1. Assessment of IconCache.db default system icons immediately following creation. 2. Review of default system icons in Shell32.dll
B: Broad empirical observation of the workings of the IconCache.db		To investigate the effects of common user activity on the database.	
	B1	To investigate the effects of basic user activity on the host alone.	Ascertain: <ol style="list-style-type: none"> 1. IconCache.db contents after opening files & folders on the host system. 2. Its contents after placing a link

			<p>on the desktop from a host-based program.</p> <p>3. Its contents after running a host-based executable.</p>
	B2	To investigate the effects of user initiated DVD activity.	<p>Ascertain:</p> <ol style="list-style-type: none"> 1. IconCache.db contents after loading a DVD containing an executable file into the host DVD tray. 2. Its contents after installing a program on the host from a DVD. 3. Its contents after running the newly installed executable from the host.
	B3	To investigate the effects of connecting a USB thumb drive to the host	<p>Ascertain:</p> <ol style="list-style-type: none"> 1. IconCache.db contents following the connection of a clean USB thumb drive on the host. 2. Its contents after connecting a USB drive containing files and folders only. 3. Its contents after opening a file from a connected USB. 4. Its contents after connecting a USB drive containing one or more executables. 5. Its contents after an executable file is run from a thumb drive.
	B4	Further research on IconCache.db to corroborate findings	<ol style="list-style-type: none"> 1. Creating a new user account 2. File size changes for experiments in Set. 3. Date and time changes for experiments in Set .

4.5 Icon caching mechanisms in Windows

Holderness (1999) has observed that: ‘Every time the shell displays a folder full of files it needs to obtain icons for each of those items from somewhere... By saving icons that has already retrieved in a cache in memory, the shell is relieved of the need to constantly retrieve icons from disk.’ He also noted that, as well as caching icon information in memory, Windows systems attempt to save the information to disk before shutdown. By this means, the cache persists from one session to another. At the time of writing his observations, Holderness was referring to Windows operating systems which existed at or before 1999. Windows XP (32-bit) was not launched until 2001. Previous to Windows XP, icons called by the system were stored in a hidden file at:

C:/Windows/ShellIconCache

With the introduction of Windows XP , the functionality of ShellIconCache was taken over by the IconCache.db file. During this study, the baseline state of the IconCache.db was examined in order to establish its properties. The object was to distinguish artefacts present in the file on its initiation from those created by user activities.

4.5.1 Shell32.dll

Much of the functionality of Windows operating systems and the programs run on these systems is provided by special files known as dynamic link libraries (DLL). The use of DLLs helps both operating systems and programs to load and run more efficiently and therefore quickly (Microsoft, 2013). Shell32.dll is the main kernel of the Windows OS. It is a library which contains Windows Shell API (Application Programming Interface) functions, thus it is a core system file which is responsible for a number of key operations when a Windows computer is running.

Shell32.dll is located at:

C:\windows\system32\Shell32.dll

As well as containing API functions, the file also contains a number of icons. Tests carried out during this research showed that the Shell32.dll in Windows XP (32-bit) contained 306 icons, each of which exist in a number of versions and which have differing dimensions, numbers of colours and file sizes, as illustrated in Figures 11 and

12, below. Shell32.dll creates the IconCache.db in memory following a clean OS install, as will be shown. The database is written to disk on system halt or reboot.

Figure 11: Icons stored in Shell32.dll

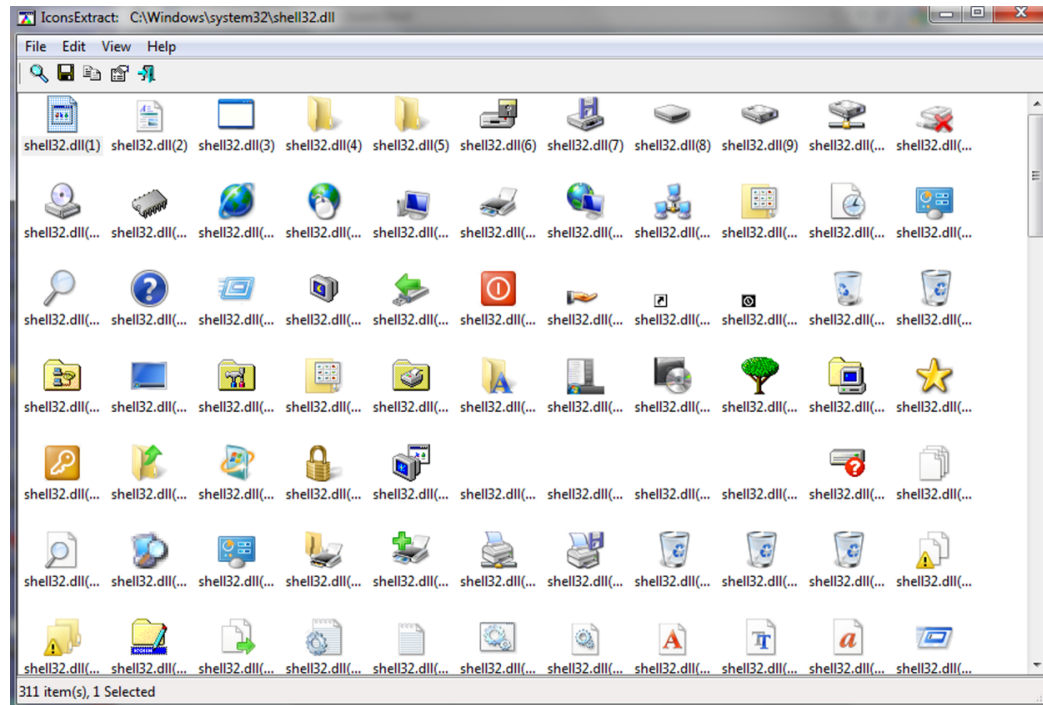
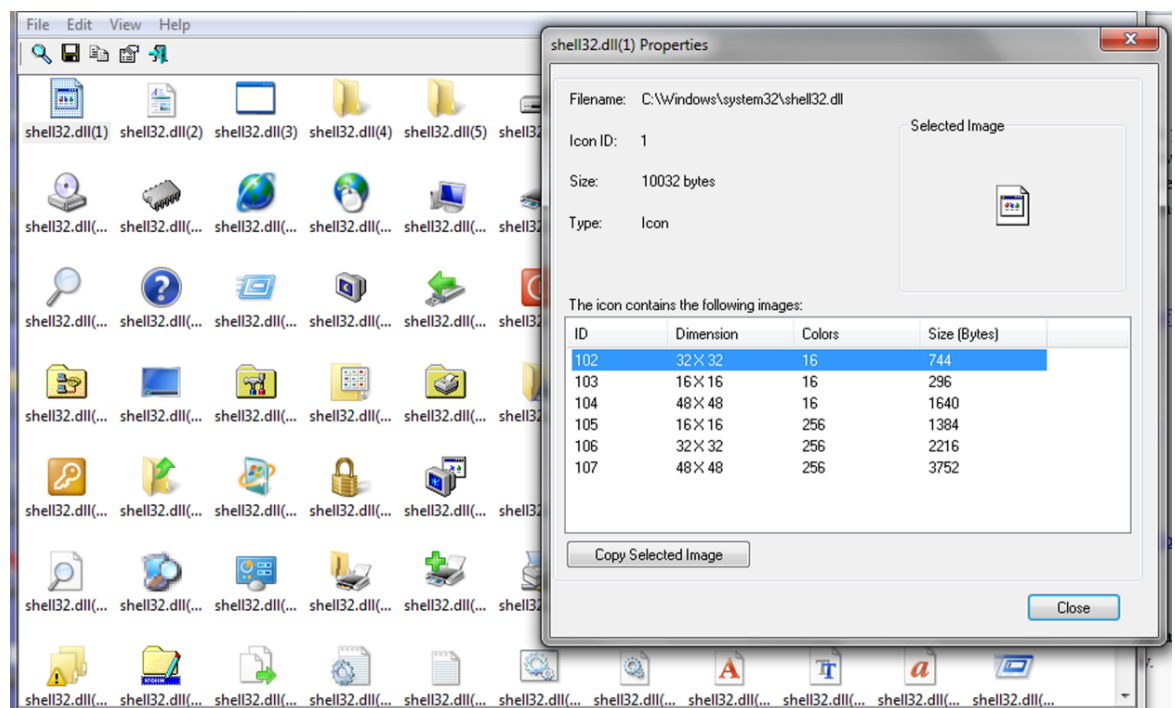


Figure 12: Shell32.dll and stored icon properties



4.5.2 IconCache.db

The IconCache.db is a hidden file and Folder Options must be configured to show hidden files before it will be revealed.

4.5.2.1 Location

Depending on the operating system, the IconCache.db may be located as follows:

- Windows XP: C:\Documents and Settings\Username\Local Settings\Application Data\IconCache.db
- Windows Vista or Windows 7:
C:\Users\Username\AppData\Local\IconCache.db

4.5.2.2 Creation/initialisation

The IconCache.db is created by Shell32.dll and it will be initialised by the shell already containing a number of system icons. The number will vary depending on the operating system in use and the configuration settings chosen during its installation. For the systems tested during this research, using the settings described in 4.3, above, the numbers of icons present in the IconCache.db when first created were as documented in Table 4.

Table 4 : Numbers of icons in the IconCache.db on first creation by version of Windows OS

Windows Operating System Version	Numbers of icons in IconCache.db on first creation of the file
Windows XP Home Edition SP. 2 (32bit)	172
Windows Vista Home Basic (32 bit)	426
Windows 7 Home Premium (32 bit)	158

Windows Operating System Version	Numbers of icons in IconCache.db on first creation of the file
Windows 7 Home Premium (64 bit)	183

4.5.2.3 File signature

The file signature for the IconCache.db appears to be consistent across Windows XP, Windows Vista and Windows 7 at offset 4 for four bytes, where the hexadecimal values: 57 69 6E 34, read as text: Win4, are recorded (Table 5). However, variances in the IconCache.db are found to occur at the beginning of the file, at offsets 0, 8 and 12 in the three versions of the operating system. Examples are given in Table 6.

Table 5: Basic file signature for IconCache.db in Windows XP, Vista and 7

Signature offset	Hex	Text
Offset 4 for 4 bytes	57 69 6E 34	Win4

Table 6: Sample differences in IconCache.db file signature in Windows XP, Vista and 7

Operating System	Offset 0 For 1 byte		Offset 8 For 2 bytes		Offset 12 For 1 byte	
	Hex	Text	Hex	Text	Hex	Text
Windows XP (32 bit)	50	P	05 05	..	54	T
Windows Vista (32 bit)	40	@	06 05	..	71	q
Windows 7 Home Premium (32 & 64 bit)	40	@	06 05	..	B0	°
Windows 7 Professional (64 bit)	40	@	06 05	..	B1	±

4.6 Experimental results

Results from the experiments listed in Table 3 are summarized in the following sections. In the interest of brevity, specific examples of findings recorded in this section relate to Windows XP only, since the same basic behaviour patterns were observed during the experiments on Windows Vista and Windows 7 32-bit systems.

4.6.1 IconCache.db baseline behaviour

Conducting Experiment A from Table 3 for the Windows operating systems chosen for this research resulted in the following findings:

- i. The IconCache.db is not created during a clean install of the operating system.
- ii. The IconCache.db is created on system reboot following a clean OS install.
- iii. Where no user action apart from a system restart has taken place, an IconCache.db which contains a number of default system icon images is created (as shown in Figure 10.)
- iv. The numbers of icons present in the IconCache.db at creation may vary depending upon the operating system in use and the configuration settings chosen during installation.

4.6.2 Abbreviated findings

4.6.2.1 Initial creation of the IconCache.db

Following a clean install of the Windows XP operating system, during which one user was set up on the system but no other action was taken apart from a system restart, an IconCache.db which contained 172 icon images was created (Table 4).

When viewed using the forensic software, FTK, a listing of programs and processes associated with those icons was viewable in ASCII. Example A, below, shows a sample start of the IconCache.db file, where references to shell32.dll is first noticeable, closely followed by the loading of Internet Explorer from the program files folder on the host hard drive, allocated letter C:, as is commonly the case. A record of other automatic processes which run on system restart comes next, with those which access the user's documents and settings folder clearly shown. Example B gives the end of the file for this sample and shows that certain executables, e.g. that for the solitaire game which forms part of the Windows install package, have been loaded.

The record: d:\setup.exe , is also visible. This refers to the Windows XP executable, which has been run from the DVD drive, here allocated drive letter D: by the system.

Example A. Sample start of file, first generation of IconCache.db

```
shell32.dll shell32.dll shell32.dll shell32.dll shell32.dll shell32.dll shell32.dll
shell32.dll shell32.dll shell32.dll/c:\program files\internet
explorer\iexplore.exeGc:\documents and settings\jan\start
menu\programs\internet explorer.Ink*c:\program files\outlook
express\msimn.exeEc:\documents and settings\jan\start menu\programs\outlook
express.Ink!%systemroot%\system32\rcimlby.exeGc:\documents and
settings\jan\start menu\programs\remote assistance.Ink/c:\program
files\internet explorer\iexplore.exe2c:\program files\windows media
player\wmplayer.
```

Example B. Sample end of file, first generation of IconCache.db

```
c:\windows\system32\spider.exeRc:\documents and settings\all users\start
menu\programs\games\spider solitaire.Ink c:\windows\system32\mydocs.dll
shell32.dll c:\windows\explorer.exe c:\windows\explorer.exe d:\setup.exe
shell32.dll c:\windows\system32\mydocs.dll shell32.dll shell32.dll shell32.dll
c:\windows\system32\shdocvw.dll
```

4.6.2.2 Impact of second restart on the IconCache.db

A second system restart, immediately following the first, added data to the IconCache.db file. For instance, the fact that the logon screen had run was recorded by the “logon.scr” entry (Example C).

Two conclusions were drawn from the findings in Examples B and C. Firstly, since this information did not exist in the IconCache.db at inception, it must have been added to the file during use of the system. Secondly, since the information was not visible in the database whilst the computer was running, it must have been added to the database on system shut down.

Example C. Sample end of file, simple system restart following first generation of IconCache.db

```
c:\windows\explorer.exe c:\windows\explorer.exe d:\setup.exe shell32.dll  
c:\windows\system32\mydocs.dll shell32.dll shell32.dll shell32.dll  
c:\windows\system32\shdocvw.dll c:\windows\system32\logon.scr shell32.dll
```

4.6.2.3 Impact of program use on the IconCache.db

The listing of programs and processes in the IconCache.db was shown to grow as they were newly used, as did the number of associated icons retained. This finding was corroborated by further tests to establish file size changes prompted by experimentation. An example is given below. Further examples may be found in Appendix IX. It should be noted that in the following table (Table 7) and for all other examples produced, ‘Baseline IconCache.db’ indicates the properties of the .db file before the listed action was taken, not a virgin IconCache.db file.

Table 7: Example result showing increase in file size and icons retained in IconCache.db resulting from user action

Action	Number of icons in file	New icons added	File size (Bytes)
Baseline IconCache.db	181	n/a	2,149,330
Create link to ‘WordPad.exe’ on Desktop	188	7	2,149,870
Create ‘WordPad’ .txt file, save on Desktop	189	1	2,149,994

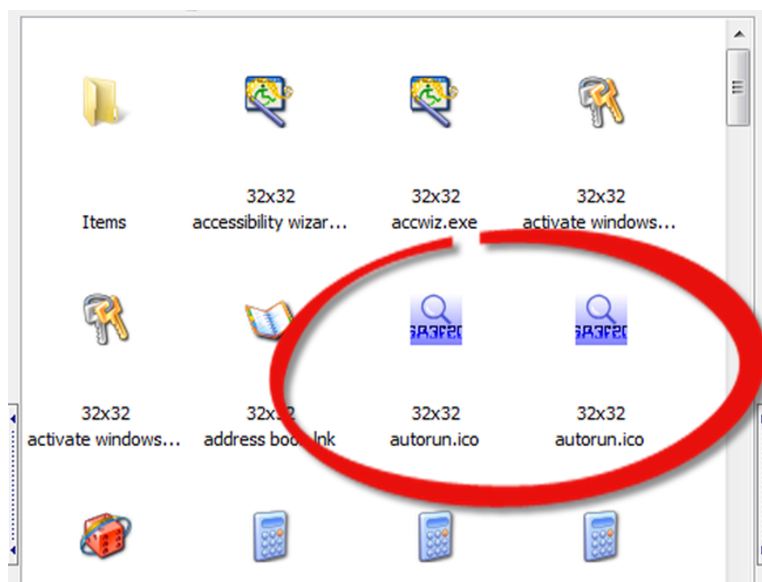
4.6.2.4 Replicating the effects of sample user actions on the IconCache.db

4.6.2.4.1 The effects of introducing a DVD-based executable to the system

a) Loading a DVD containing an executable file.

When a disk containing an application which had an 'autorun' icon was placed into the host's DVD drive, that icon was stored in the IconCache.db. Figure 13 illustrates how, when a disk containing the forensic software application 'XWays' was loaded, two 'autorun' icons (the same graphic presented in different resolutions) were stored in the IconCache.db. The textual reading from the database, shows the presence of both of these (Example D). It also shows that the path to the application, including the drive letter associated with the DVD drive was retained.

Figure 13: Icons from application's 'autorun' icon present on DVD in IconCache.db



Example D: 'Autorun' icon present in an application on DVD listed in IconCache.db

```
c:\windows\system32\spider.exeRc:\documents and settings\all users\start  
menu\programs\games\spider  
solitaire.lnk:\windows\explorer.exec:\windows\system32\mydocs.dll  
shell32.dllc:\windows\explorer.exe shell32.dll shell32.dll shell32.dll  
shell32.dllc:\windows\system32\shdocvw.dll shell32.dll shell32.dll shell32.dll  
d:\autorun.ico shell32.dll d:\autorun.ico
```

b) Installing an executable from a DVD

When an application was installed on the host from an executable file on DVD, the icons for the application were stored in the IconCache.db. The path to the executable, both on the DVD and in the 'program files' folder on the host's C: drive, were also recorded in the file (Example E).

Example E: Path to executable run from DVD in IconCache.db file

```
c:\windows\system32\mspaint.exec:\windows\system32\shimgvw.dll"d:\x-ways forensics 14.9\setup.exe(d:\x-ways forensics 14.9\xwforensics.exe"d:\x-ways forensics 14.9\setup.exe1c:\program files\x-ways forensics\xwforensics.exe1
```

c) Running an installed executable

When the newly installed executable was then run on the host, a record this action was retained in the IconCache.db (Example F).

Example F: Record of executable run from host in IconCache.db file

```
c:\windows\system32\mspaint.exec:\windows\system32\shimgvw.dll"d:\x-ways forensics 14.9\setup.exe(d:\x-ways forensics 14.9\xwforensics.exe"d:\x-ways forensics 14.9\setup.exe1c:\program files\x-ways forensics\xwforensics.exe1c:\program files\x-ways forensics\xwforensics.exec:\windows\winhlp32.exe
```

4.6.2.3.2 The effects of USB device usage on the IconCache.db

a) Opening a text file

When a plain text file was opened from a memory stick, the IconCache.db recorded the running of both *notepad.exe* and *wordpad.exe* from the C: drive but did not retain the drive letter for the stick (Example G). Icons for both programs were also stored in the database.

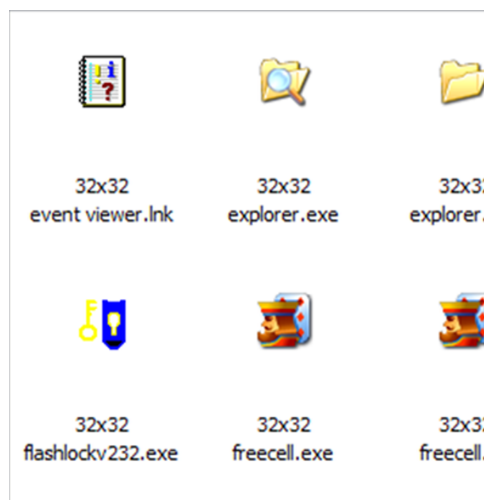
Example G: Sample end of file after text document run from USB memory stick

```
c:\windows\system32\mydocs.dll shell32.dll shell32.dll shell32.dll
c:\windows\system32\shdocvw.dll c:\windows\system32\logon.scr shell32.dll
c:\windows\system32\notepad.exe3c:\program files\windows
nt\accessories\wordpad.exe
```

b) Viewing a USB containing executable files

When a USB memory stick containing an executable in the root of the drive was connected and viewed on the host, icons associated with that executable were stored in the IconCache.db despite the fact that it had not been run. As illustrated in Figure 14, in a test in which a USB with the integral encryption program, Flashlockv232.exe, was connected to the host, its icon was stored in the IconCache.db file (first icon, second row down). The path to the associated executable files, together with the drive letter allocated to the USB stick, was also retained (Example H).

Figure 14: Icons from executables present on USB memory stick in IconCache.db file



Example H: Listing of executables on USB connectable shown in ASCII within IconCache.db

```
shell32.dllc:\windows\system32\logon.scr"%systemroot%\system32\wiaacmgr.
exe shell32.dll shell32.dll shell32.dll
shell32.dll!%systemroot%\system32\shimgvw.dll e:\osf.exe
c:\windows\system32\zipfldr.dll e:\flashlockv232.exe shell32.dll
c:\windows\system32\wiaacmgr.exe
```

c) Running an executable file from USB

When an executable was run from a USB drive, icons associated with it were stored in the IconCache.db. In a test in which the forensic software FTK imager lite was run from a USB, two icons associated with it were shown in the IconCache.db. The textual reading from the database showed that these were both recorded, along with the path to the executable (Example I).

Example I: Icons of Executable run from USB connectable shown in IconCache.db

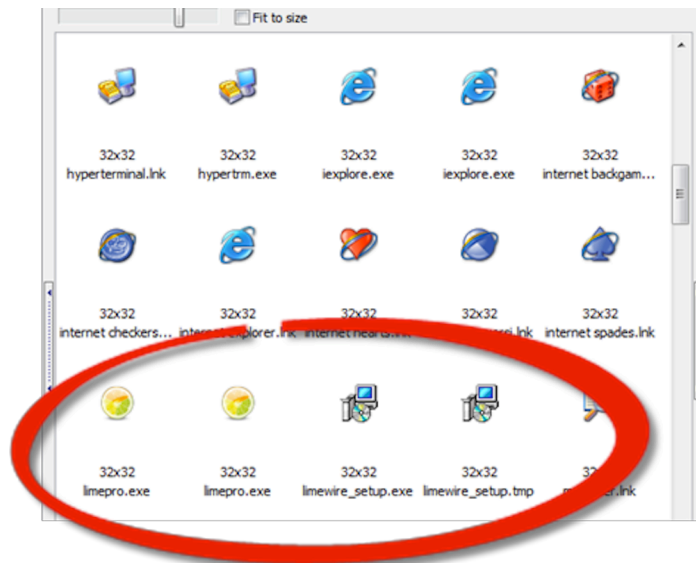
```
\system32\logon.scr:c:\windows\explorer.exec:c:\windows\system32\mydocs.dllshe
ll32.dllc:\windows\explorer.exec:c:\windows\system32\shdocvw.dllshell32.dllshell3
2.dllshell32.dllc:\windows\system32\mshta.exec:c:\windows\system32\cryptui.dll!f
:\imager lite 2-1\ftk imager.exe!f:\imager lite 2-1\ftk imager.exe
```

Significantly, for the overall research project, the same types of artefacts were created when a vPC was run from a USB drive. For example, starting Ceedo Personal caused both the program icon and a textual record to be retained in the IconCache.db. Further details are given in Chapters 5 and 6.

d) Installing an executable from a USB

When an executable was installed on a host from a USB drive, the icons associated with it were stored in the IconCache.db. In a test involving the once popular peer-to-peer application, Limewire, two instances of its executable icon and two associated setup icons were retained in the IconCache.db, (Figure 15). The path to the executable on the USB, allocated letter 'F:' was also revealed in the ASCII (Example J).

Figure 15: Icons from Limewire application, installed from USB connectable & run on host shown within IconCache.db file



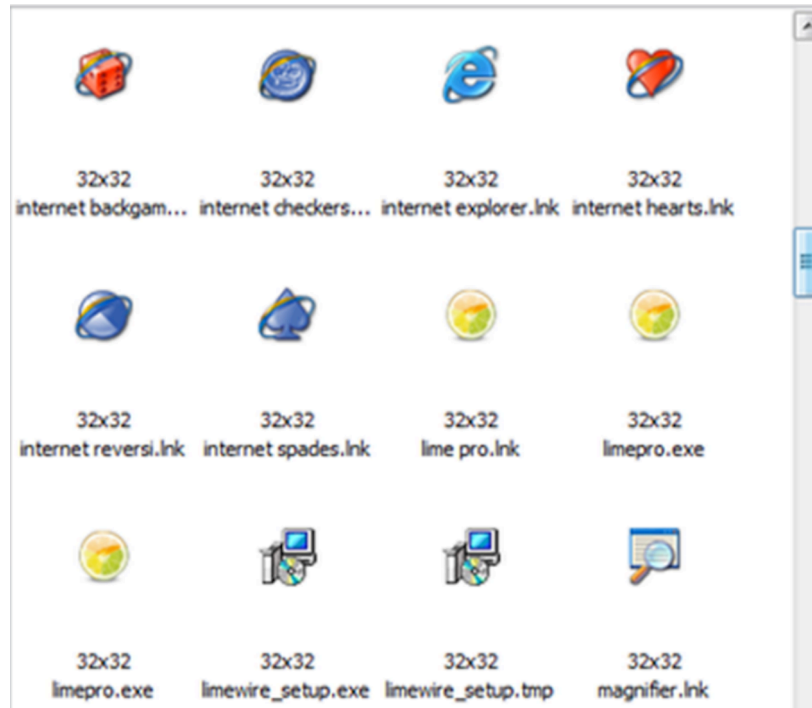
Example J: Executable installed from USB connectable & run on host shown within IconCache.db file

```
shell32.dllc:\windows\system32\shdocvw.dllshell32.dllc:\windows\system32\sync
ui.dll
shimgvw.dll3c:\programfiles\windowsnt\accessories\wordpad.exe!c:\progra~1\wi
ndow~2\wmplayer.exec:c:\windows\system32\zipfldr.dllf:\limewire_setup.exe=c:\d
ocume~1\max\locals~1\temp\is-trm8s.tmp\limewire_setup.tmp%c:\program
files\lime pro\limepro.exe%c:\program files\lime pro\limepro.exe&c:\program
files\lime pro\unins000.exe
```

e) Uninstalling an executable placed on the host via USB

When the Limewire executable was subsequently deleted from the Windows host, using the program's own uninstaller, five icons related to it nevertheless remained in the IconCache.db, as shown in Figure 16 (icons 3 & 4 second row and 1, 2 & 3, third row).

Figure 16: Icons retained in IconCache.db following uninstall of Limewire program



4.6.2.4.3 Creating a new user on a computer

When a new user was set up on the host machine, an IconCache.db for that user was created in their profile, separate to and different from the IconCache.db for the original user, as shown in the ASCII output in Example K. The creation of the new user did not update the original user's IconCache.db with any information about new user activity. The original user's IconCache.db was merely updated with a call to System 32\mshta.exe. This is a system file associated with the Microsoft HTML application host.

Example K: First instance of IconCache.db for new user 'Max'

```
shell32.dll/c:\program files\internet explorer\iexplore.exeGc:\documents and  
settings\max\start menu\programs\internet explorer.lnk*c:\program files\outlook  
express\msimn.exeEc:\documents and settings\max\start menu\programs\outlook  
express.lnk/c:\program files\internet
```

4.6.2.5 IconCache.db Date and Time information

Changes in date and time stamps for the IconCache.db were recorded during experimentation. A sample is given in Table 8, below. It was observed that, for the 32-bit operating systems tested during this research:

- a) When the IconCache.db was deleted, it was recreated on system reboot with its original created date/time and with a modified time which reflected the time of its deletion. This finding supports the supposition that the database is written to disk at system shut down or reboot.

- b) The IconCache.db file modification times were updated following actions which caused new information to be retained in it – the example chosen here was creating a text file and saving it to a USB memory stick.

Table 8: Sample Date and Time changes to the IconCache.db in Windows XP

Action	Created		Modified	
First instance IconCache.db	28/02/12	21.07	28/02/12	21.07
Delete IconCache.db@ 21.33, restart	28/02/12	21.07	28/02/12	21.33
Create .txt file, save to USB, restart	28/02/12	21.07	28/02/12	21.12

Preliminary testing on a Windows 7 Home Basic 64-bit installation produced a variation in results, as follows:

- a) When the IconCache.db was deleted, it was not immediately recreated on system re-boot. In fact, no IconCache.db was apparent.

- b) A new IconCache.db was created following a second system reboot. It was smaller in size than the original (804 kb as opposed to 1.08 mb following a fresh install).

However, further testing showed that this difference only occurred when the host was shut down by disconnecting the power cable from the back of the machine. When the host was shut down normally, the IconCache.db was recreated in the same way as in 32-bit Windows systems. It was also found that, in common with the 32-bit systems examined, modification times for the IconCache.db were updated following actions which caused new information to be retained in it.

4.7 Conclusions

The IconCache.db is a hidden file which is created by Shell32.dll, a Windows system file which is loaded during the boot process. The file is first created following a clean install and reboot containing a base number of icons, which varies depending on the version of the operating system in use and the settings chosen. The IconCache.db exists in memory but is written to disk following a Windows shutdown (either manual or via the 'Pull The Plug' method) or restart. The database regenerates itself from the information held on the computer system at shutdown when the system is restarted. Its persistence, plus the information that it retains in respect of program usage and installation, makes it a useful resource for the computer analyst.

This research has shown that the IconCache.db not only stores system and program related icons but also a textual record of the activities which it facilitates. These will include numerous user-initiated activities both on the host and peripherals attached to the host, such as DVD players and USB connectable devices. This ability gives the IconCache.db the potential to inform and assist forensic investigation, particularly those involving the suspected use of a vPC.

Where external media, such as DVDs and USB connectable devices have been introduced to the system, for example, artefacts retained in the IconCache.db can be especially useful in identifying executable files which either existed on them or were run or installed from them. The ability to trace the installation or use of a foreign program to a USB connectable or other media can help to narrow down the search for a culprit. It can also help suggest where other evidence might exist. Care should be taken, however, in interpreting findings since the IconCache.db may contain the icons

of executables which were merely stored on a USB connectable, not actually run from it. Opening and viewing the contents of a USB device which contains a program executable file will cause any related icon to be stored in the database.

The information gleaned from the IconCache.db can be used to corroborate other findings, for instance artefacts remaining in the Registry, link files and log files. It can be used to show that certain programs e.g. LimeWire existed and were used on a machine, even if those programs have been uninstalled. Where an external encryption program has been used, it may provide the only indication. Likewise where antifoensics have been used to clear information in registry keys.

The IconCache.db contains a wealth of information including file paths to the programs and processes which have been invoked on fixed and attached drives. Since an IconCache.db exists for each named user of a computer, the file paths also reveal which activities occurred under which user name.

Further research is needed in order to gain a better understanding of the IconCache.db in terms of its structure and functionality. This would greatly aid the interpretation of information to be found in the file. An opportunity also exists to develop software capable of rendering icons stored in the database together with properly parsed date and time data.

Further research is needed in order to clarify how the IconCache.db behaves both in 64 bit versions of the Windows operating system and in Windows 8. Initial experimentation with Windows 7 Home Premium 64 bit has indicated that, once created, the file stores artefacts associated with user-based computer activity in a similar way to the 32-bit operating systems which were tested during this research. Initial experimentation in Windows 8 has indicated that the IconCache.db no longer stores images of icons associated with applications which have been invoked at start time or run as a result of user activity. These are now stored at:

USER\AppData\Local\Microsoft\Windows\Explorer

It does, however, retain a textual record of such activities, in the manner which has been observed in previous versions of Windows.

4.8 Chapter summary

This Chapter has discussed the IconCache database and its potential as a resource for digital forensic examiners, especially where the unauthorized use of USB connectable devices is suspected. This makes it an essential component of any investigation into the unauthorized use of vPCs. It is shown that as well as storing system and program-related icons, the IconCache.db also retains a textual record of the activities which it facilitates. Of particular interest are the artefacts retained in the database as a result of user-initiated activity. These include the names of executable files that were run or installed from external media and the associated file path.

The ability to trace the installation or use of a foreign program such as a vPC to a USB connectable or other media is important during a digital forensic investigation. The information could help to narrow down the search for a culprit. It could also help suggest where other evidence might exist. Care should be taken in interpreting findings, however. For example, opening and viewing the contents of a USB device which contains a program executable file in the root of the drive will cause any related icon to be stored in the IconCache.db. Thus the database may contain the icons of executables which were merely stored on some attached media, not actually run from it.

The IconCache.db contains a wealth of information including file paths to the programs and processes which have been invoked on fixed and attached drives. Since an IconCache.db exists for each named user of a computer, the file paths also reveal which activities occurred under which user name.

The information gleaned from the IconCache.db can be used to corroborate other findings, for instance artefacts remaining in the Registry, link files and log files. It can be used to show that certain programs existed and were used on a machine, even if those programs have been uninstalled. Where a vPC or another external software, for instance an encryption program, has been used it may provide the only indication. Likewise where antiforensics have been used from an external source to clear information in Registry keys.

CHAPTER 5.

EXPERIMENTAL METHOD, CONDUCT AND RESULTS

This chapter describes the design and implementation of the experiments conducted during this research. A set of initial experiments, designed to garner baseline results against which main test results are to be measured, are detailed. The effect of connecting the individual vPCs chosen for testing to Windows-based host systems and carrying out various user-initiated activities is then monitored and the results reported.

5.1 Experimental method and conduct

The design conditions for the experiments described in this section are detailed in Chapter 3 where a full account of the test environment and the hardware and software used during testing is also given. The purpose of these experiments is to test the hypothesis of this thesis by:

- a) Creating a baseline dataset to act as a measure for experimental results. This is achieved by:
 - i. Identifying those artefacts which are created when a clean USB stick is introduced to a test Windows system.
 - ii. Identifying those artefacts which are created when a USB stick containing a test vPC is introduced to a Windows system.
 - iii. Identifying those artefacts which are created when a vPC is run on a Windows system.
- b) Isolating artefacts created by the active use of vPCs on test Windows systems. This is achieved by comparing the results obtained during a second set of experiments with those obtained during the initial set of experiments.

5.1.1 Test system monitoring

The utility Process Monitor v. 2.94 was pre-installed on a set of test hard drives before experimentation began. The purpose of this was to audit any activity which occurred on the host system as a result of conducting the baseline experiments described below. The aim was to assemble a set of artefacts to act as a standard against which further experimental outputs would be measured. Since Process Monitor records file system,

Registry, process, thread and DLL activity in real time, it was possible to investigate the behaviour of the individual vPCs, once introduced to the host, by analysing these records,. It was therefore possible to gain an insight into which areas of the operating system would retain traces of vPC activity.

5.1.2 RAM capture

For each of the experiments listed below, RAM was captured before the system was terminated. Once an experiment was complete, the computer was halted by pulling the plug from the back of the machine. As explained in Chapter 2, this method was used in order to reproduce field conditions: when confronted with a running ‘suspect’ workstation, a digital forensic examiner will most commonly halt it in this way before commencing imaging. The object of doing so is to prevent any changes from being made to the host system, so maintaining the integrity of the evidence captured.

5.1.3 Control Experiment

The first experiment carried out was a control experiment. This involved introducing a clean USB key to a Windows XP 32-bit system. The USB key had been wiped using the software Eraser v. 6.07.1893, which securely overwrites data several times. The key was then reformatted FAT32 using Windows.

Process Monitor was set to run before the USB key was inserted into the host machine. Outputs from the software clearly showed the enumeration process, which occurs automatically on Windows systems, taking place. During enumeration, a USB connectable drive is allocated a drive letter and the host system sends a number of queries to the drive. Amongst other things, these queries elicit a creation time for the volume, its serial number and its label. As shown by Carvey (2009) and discussed in Chapter 2, this information is stored in a key which is then created in the HKEY local machine/System portion of the Registry.

As a result of this experiment, a list of events which occur when a USB key is connected to a Windows host was drawn up. This was used as the minimum set of events which could be expected to occur during further experimentation. By comparing this set of results with those obtained from Process Monitor after

introducing a test vPC to a host, it was possible to isolate activity initiated by the vPC and thus derive a Data Set Of Interest (DSOI) which was then used for the analysis and comparison of results.

5.2. Initial analysis

A brief analysis of the file system structures for each vPC was conducted in order to gain some basic background knowledge on the various folders and their contents with the object of gaining a better understanding of the outputs reported by Process Monitor during experimental testing. The analysis was done by viewing the software in the forensic tool FTK Imager 3.1.1.8. Commentaries and screen shots follow.

5.2.1 MojoPac file structure

MojoPac's file structure appeared to mimic that of the Windows operating system, complete with folders for Program Files, Documents and Settings etc., as shown in Figures 17 a) & 17 b), below:

Figure 17a): MojoPac Filestructure

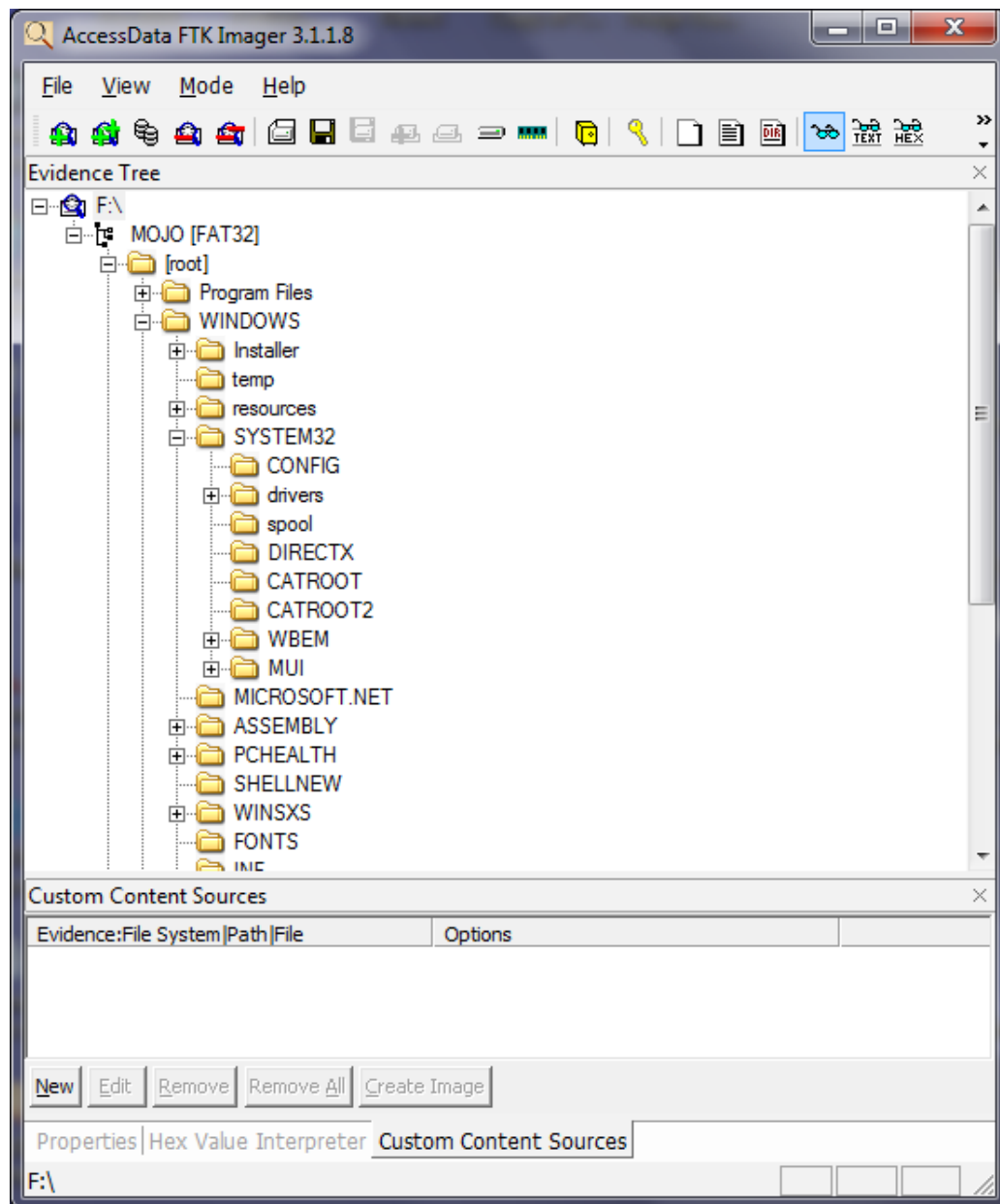
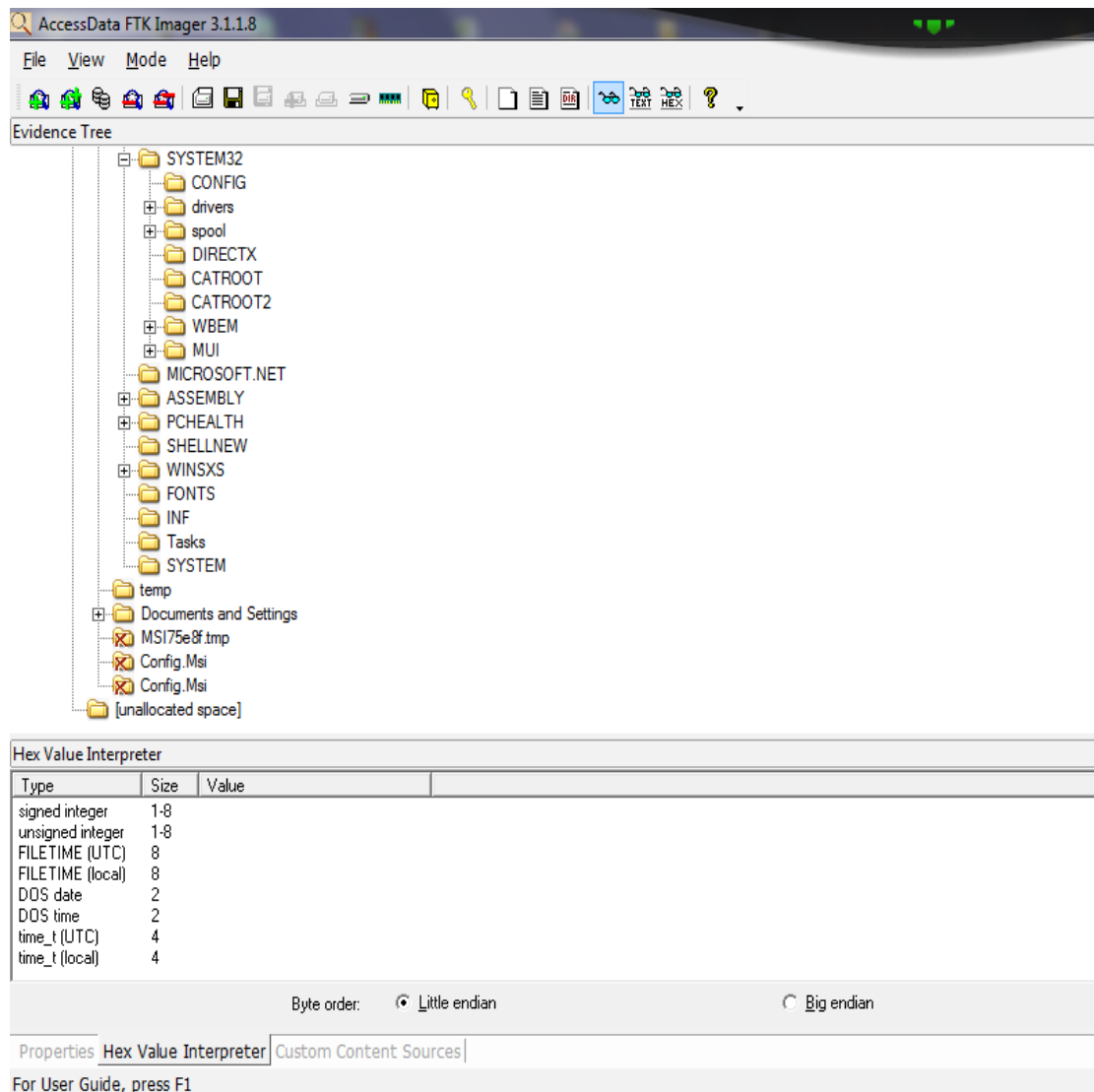


Figure 17 b): MojoPac Filestructure



5.2.2 Ceedo file structure

Ceedo's file structure included folders for 'Program Files' and 'Program Files (x64)' as well as a 'Windows' folder (Figures 18 a) & 18 b)) . There was a 'User' folder containing subfolders related to installed applications. (Figure 18 c)) and a 'My Documents' folder for the storage of user created files.

Figure 18 a): Ceedo Filestructure

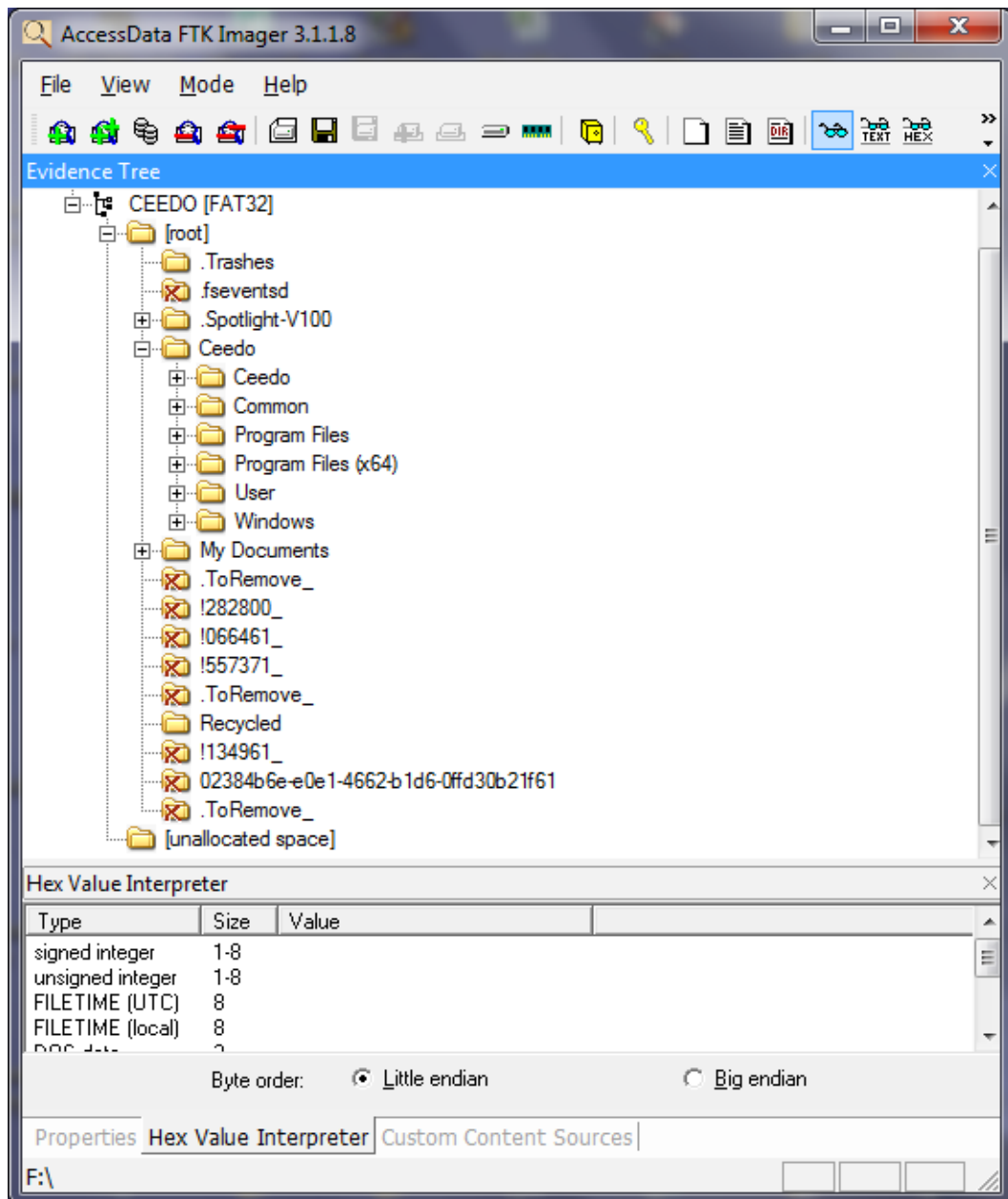


Figure 18 b): Ceedo Filestructure

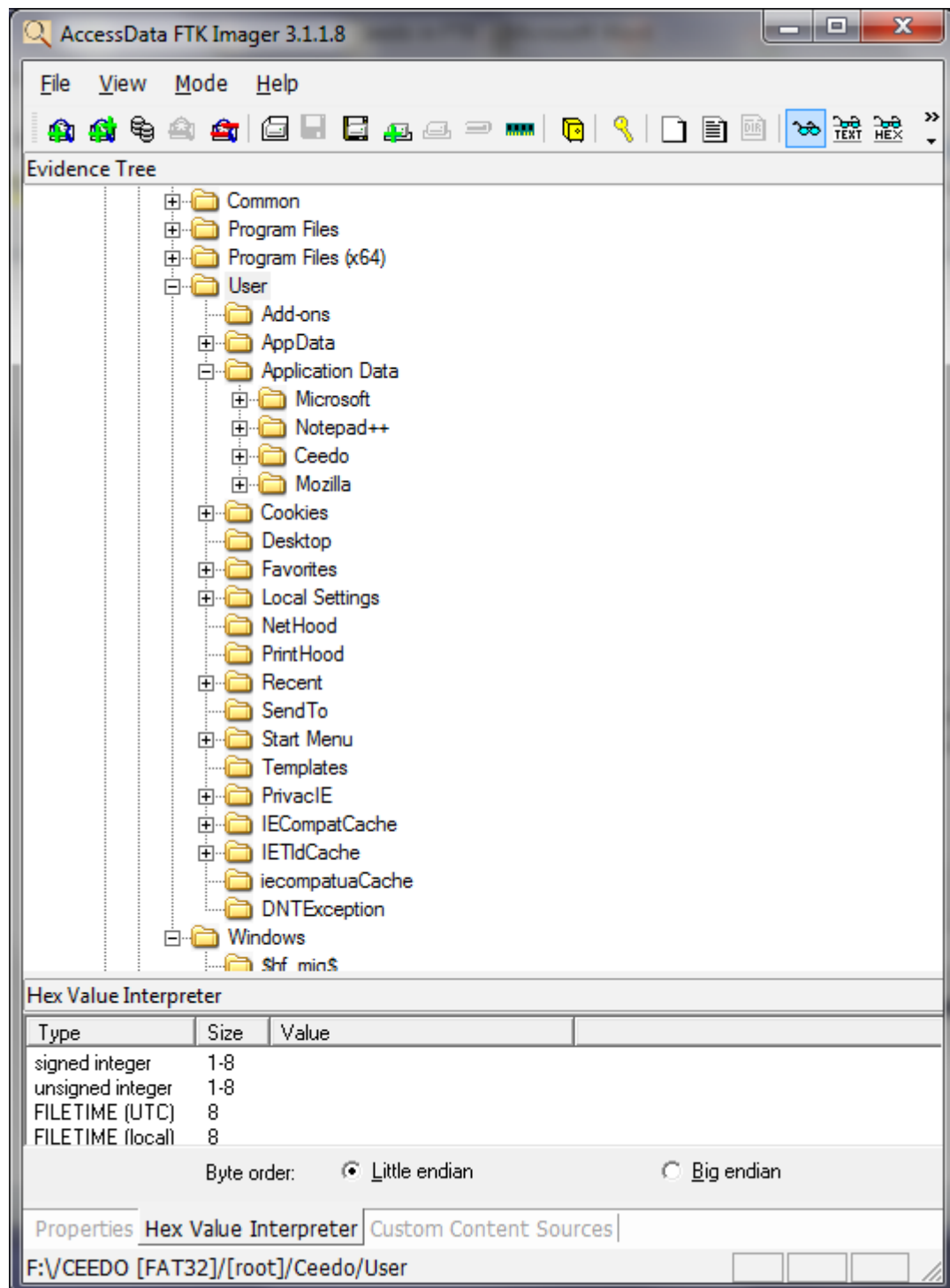
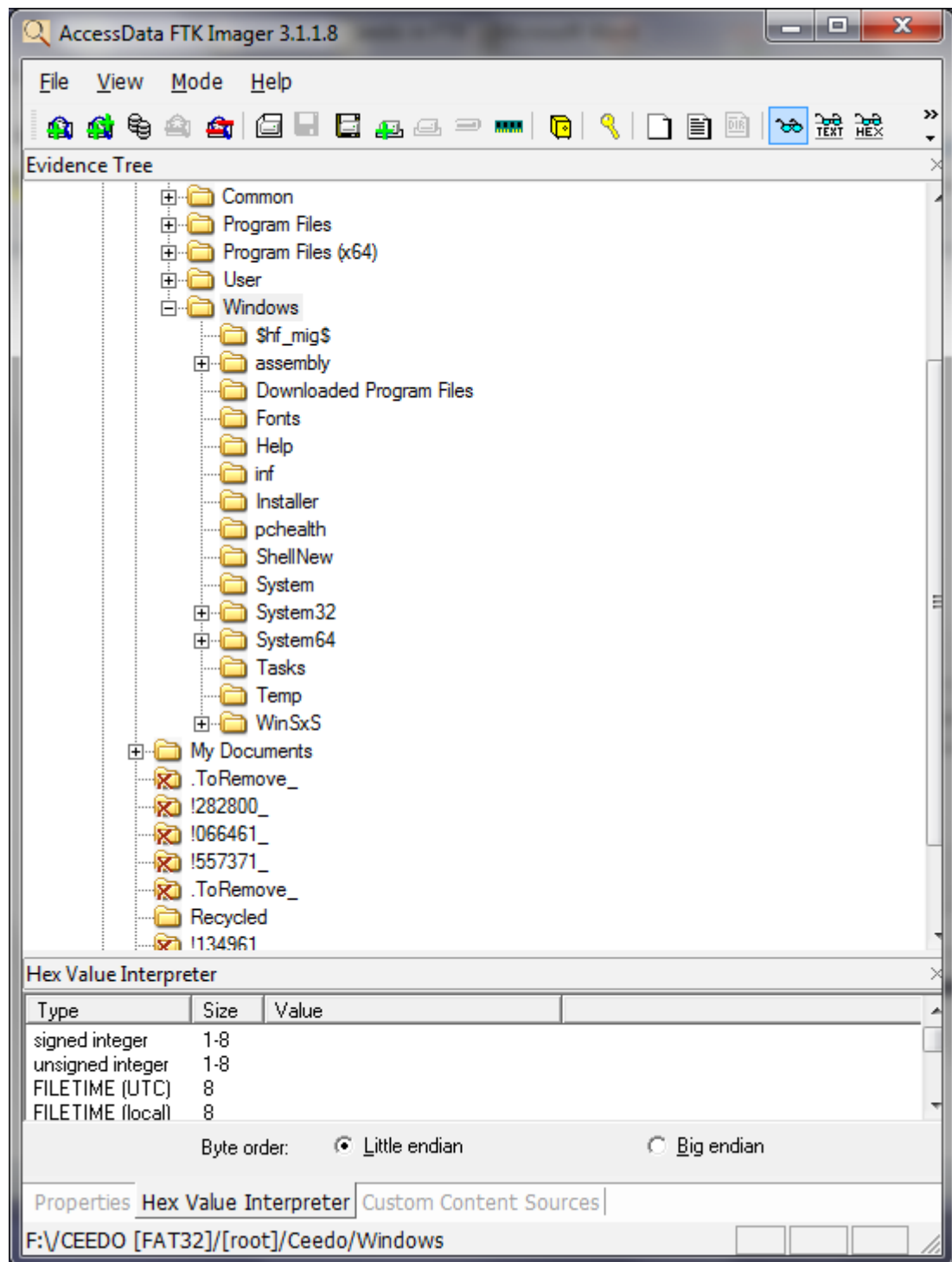


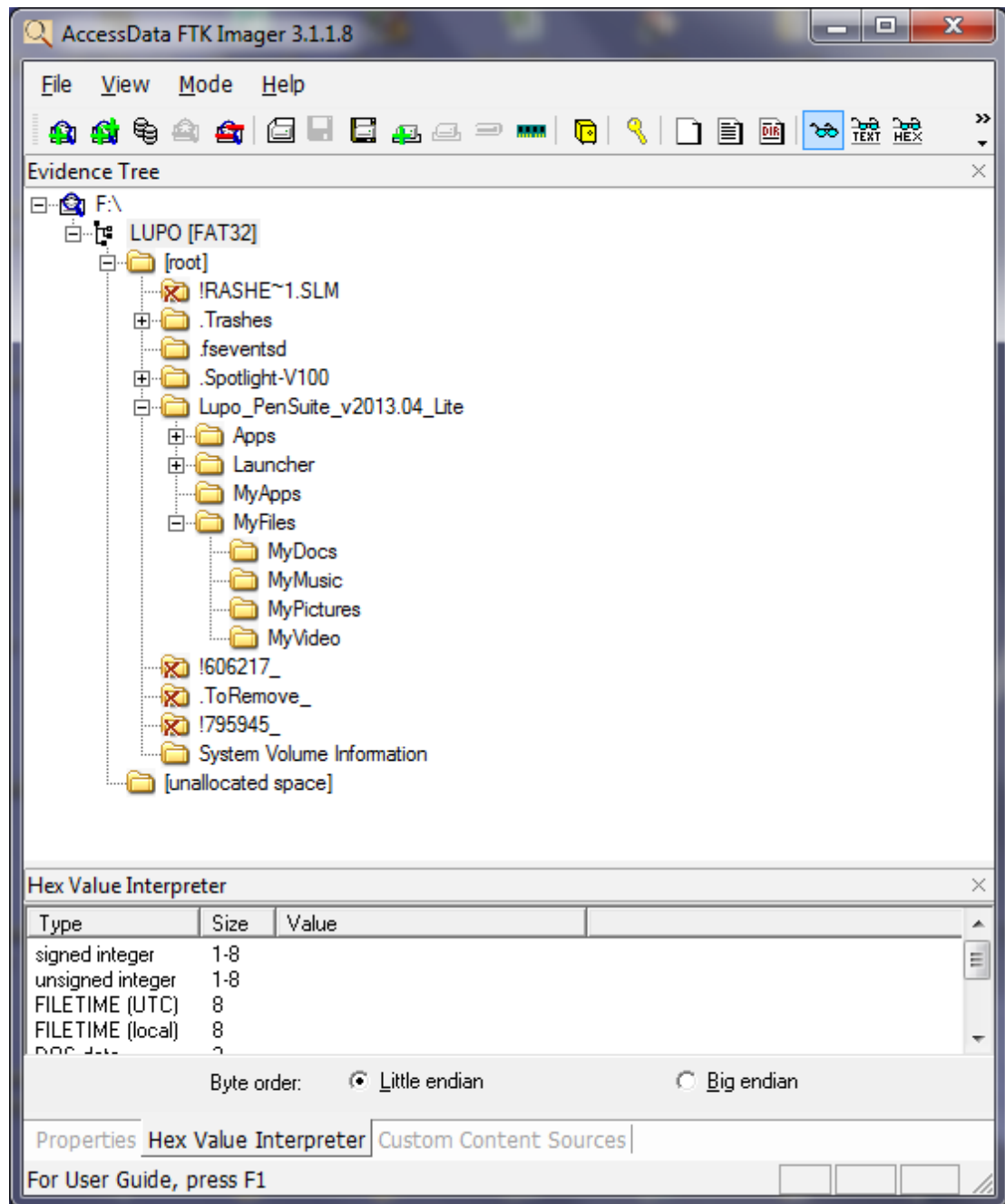
Figure 18 c): Ceedo Filestructure



5.2.3 LupoPenSuite file structure

LupoPenSuite had a folder for installed applications and a 'MyFiles' folder within which a user could store various types of files. (Figure 19).

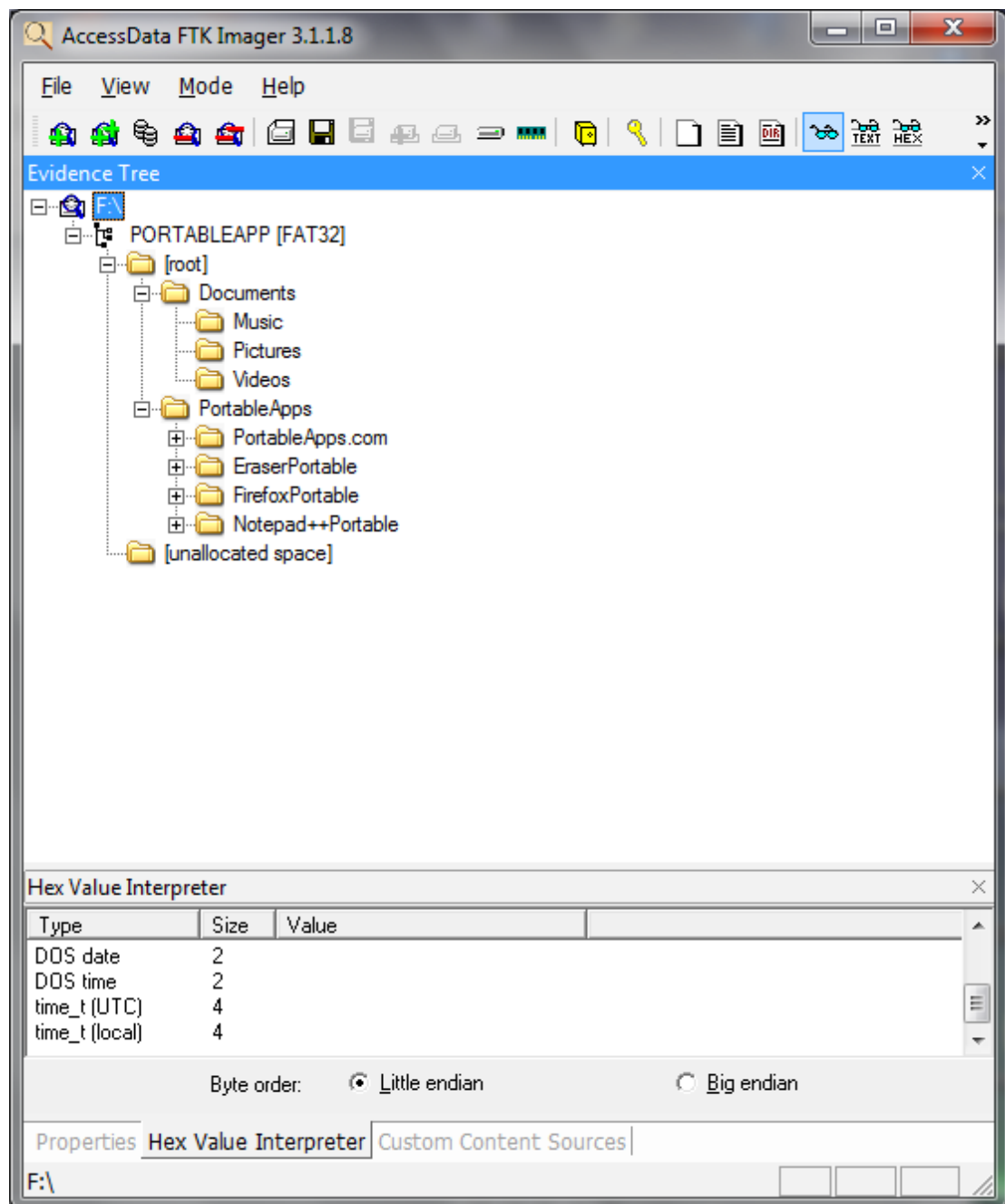
Figure 19: LupoPenSuite Filestructure



5.2.4 PortableApps file structure

The PortableApps file structure appeared similar to that of LupoPenSuite. There was a ‘Documents’ folder for user files and ‘PortableApps’ folder containing various installed applications e.g. Firefox Portable (Figure 20).

Figure 20: PortableApps File structure



5.3. Baseline Experiments

Baseline experiments were conducted for each combination of vPC and test OS, as follows:

- a) Introduce USB key containing vPC executable into the host system, no further action.
- b) Introduce USB key containing vPC executable into the host system. Run vPC.

As explained in 5.1, above, the aim was to identify a baseline dataset to act as a measure for the second set of experimental results.

5.3.1 Baseline experiments – results

A full set of results for the baseline experiments conducted on MojoPac and Ceedo Personal in Windows XP 32 bit are given below. These were obtained by analysing the test hard drives used during experimentation, as indicated in Chapter 3. Following analysis, it was found that the results obtained on Windows XP 64 bit systems were very similar to those obtained on 32 bit versions of the same OS.

A summarised report of the results obtained for baseline experiment testing on LupoPenSuite and PortableApps follows at the end of this section.

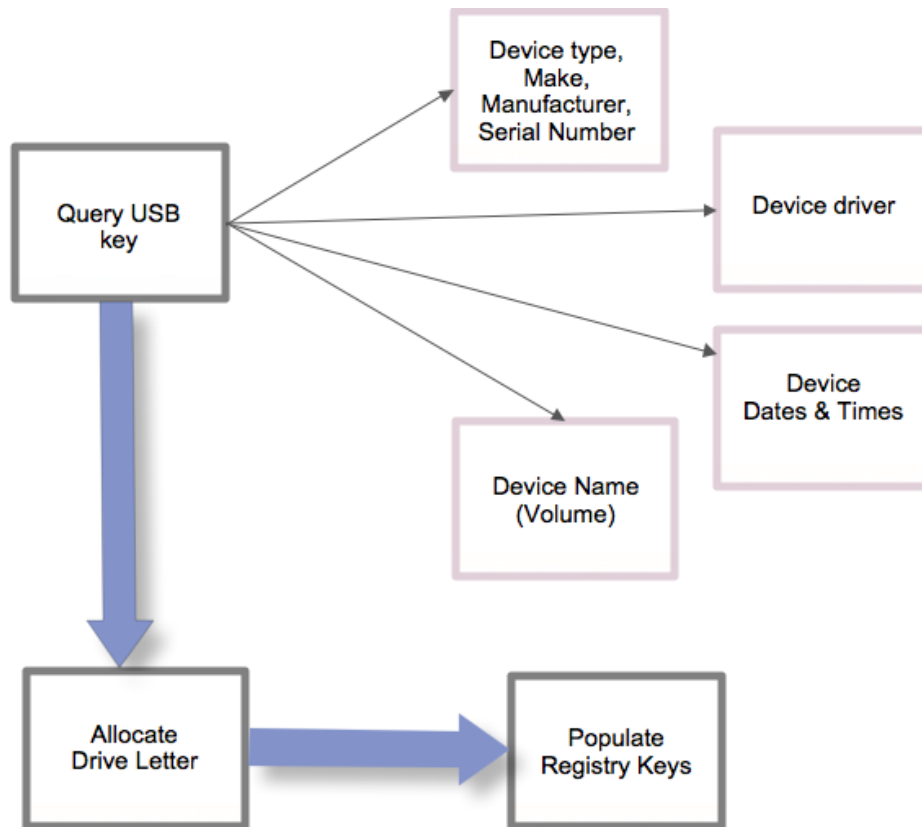
- a) Introduce USB key containing MojoPac executable file.

Result:

- i) Process Monitor (PM) identified changes in the Registry as a result of connecting the USB key. As discussed in Chapters 2 and 3, the Windows operating system automatically runs an enumeration process when a USB device is connected. The information gathered during this process is used to populate keys in the Registry. The basic process is simplified in Figure 21, below. Thus, the outputs from PM showed the host system querying the USB key for various pieces of information e.g. the volume serial number, label, its creation date and time and its last access, write and change dates and times (Appendix X a)). The key was labelled ‘MOJO’ and this was clearly reported.

A 'setup' executable was looked for but not found. The MojoPac installer executable, was found, however, and this was reported.

Figure 21: USB device enumeration process in Windows (simplified)



Other results of interest in the data set were as follows:

- a) the dates and times recorded for the 'setup' executable were all 01/01/1601 at 01:00:00 – this is the epoch date used by Windows NTFS 32 & 64 bit systems. The USB memory sticks used for testing were formatted in the FAT 32 file system. The timestamps recorded for the 'setup' executable here most probably came about when zero-value DosDATE timestamps on the FAT device were converted to the FILETIME format on connection to the host.
- b) the 'Created' and 'LastWrite' dates and times recorded for the Mojopac installation executable were 15/10/2008, the date when the Mojopac program was expanded onto the USB key upon which it was held. The 'Last Access'

time was 22/09/2014, the date when the last test was run. The 'Change' date remained 01/01/1601.

- c) Artefacts were present on the host in captured RAM (Appendix X d)). No specific references to MojoPac were found in Registry keys, however both the related icon and textual references to the executable file were to be found in the IconCache.db for the single named user of the host following system restart. (Figure 22).

Figure 22: MojoPac icon in IconCache.db textual reference

```
c:\windows\system32\spider.exeRc:\documents and settings\all users\start
menu\programs\games\spider solitaire.lnk c:\windows\explorer.exe
c:\windows\system32\logon.scr c:\windows\system32\mydocs.dll shell32.dll
c:\windows\explorer.exe c:\windows\system32\mshtml.dll/c:\program files\internet
explorer\iexplore.exe c:\windows\system32\url.dll shell32.dll e:\mojopacinstaller.exe
e:\mojopacinstaller.exe Lāñāğāčā t̄aiāēā fāñāĜāČā ŠāīāĒā ūāLāēā
```

- b) Place key containing MojoPac executable file into host. Run MojoPac.

Result:

- i. Process Monitor identified a large amount of activity which was caused by running MojoPac. Both the name of the program executable and related software e.g. Ringthree - part of the MojoPac virtualisation package - were found to be associated with the drive letter allocated to the USB which contained the vPC. (Appendix X b))
- ii. An analysis of the host hard drive found that a large number of artefacts had been retained by the system, both in live and deleted space. Sample results are given in (Appendix Xc)). It was also shown that a batch file named “_R3Cleanup_.bat” had been run automatically, apparently with the intention of wiping traces of MojoPac usage, as follows:

```
"C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\_R3Cleanup__.exe" /gd:E: E:\ %1 %2 %3
%4 %5
```



```

IF ERRORLEVEL 9 GOTO END
IF "%1" == "keep" GOTO END
del "C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\__R3Cleanup__.exe"
del "C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\reconFigure.exe"
del "C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\cmojo.exe"
del "C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\dne-fastuninstall.bat"
del "C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\ciscopec-fastuninstall.bat"
del "C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\__R3Cleanup__.bat" :END

```

Further information about the application was found in Prefetch records, as follows:

[root]/WINDOWS/Prefetch/Regedit.exe-25EEFE2F.pf

```

\DEVICE\HARDDISK1\DP(1)0-0+5\PROGRAM
FILES\RINGTHREE\CONFIG\TABLEPATCH.DAT
\DEVICE\HARDDISK1\DP(1)0-0+5\PROGRAM
FILES\RINGTHREE\SETTINGS\R3MOJOPATCH.DAT
\DEVICE\HARDDISK1\DP(1)0-0+5
\DEVICE\HARDDISK1\DP(1)0-0+5\
\DEVICE\HARDDISK1\DP(1)0-0+5\PROGRAM FILES\
5\DEVICE\HARDDISK1\DP(1)0-0+5\PROGRAM FILES\RINGTHREE\
\DEVICE\HARDDISK1\DP(1)0-0+5\PROGRAM FILES\RINGTHREE\CONFIG\
\DEVICE\HARDDISK1\DP(1)0-0+5\PROGRAM FILES\RINGTHREE\SETTINGS\

```

- i. An analysis of the host Registry found that artefacts had been retained in the NTUser.dat file at the MUICache key located at:
Software\Microsoft\Windows\ShellNoRoam\MuiCache. (Appendix X d). An executable named 'Start.exe' was shown as having run from a connectable drive at drive letter: E. An executable named 'Ringthreemainwin32.exe' was also shown as having run from drive letter E:. Both of these were also referenced in the IconCache database (see item v), below). As mentioned in item i) above, Ringthree is part of the MojoPac virtualisation package.
- ii. Artefacts were present on the host both in captured RAM and the Pagefile (Appendix X e); Appendix X f).
- iii. Running MojoPac caused references to be created in the IconCache.db for the named user 'Jan' (Figure 23). These referred to two executables, one named: 'start.exe', the other named: 'ringthreemainwin32'.

Figure 23: Icon and textual references in IconCache.db following first run of MojoPac.

c:\documents .and.
.settings\jan\d.es.k.t.o.p.\p.r.o.c.m.o.n...e.x.e.+e:\...\p.r.o.g.r.a.m.
.files\ring.thr.e.e.\bin\ring.o.s...i.c.o...e:\start...e.x.e.
.shell3.2...dll...e:\m.o.j.o.p.a.c.i.n.s.t.a.l.l.e.r...e.x.e.5.e:\p.r.o.g.r.a.m.
.files\ring.thr.e.e.\bin\ring.thr.e.e.m.a.i.n.w.i.n.3.2...e.x.e.
.shell3.2...dll.

5.3.2 Ceedo Personal baseline experiments - results

a) Introduce USB key containing Ceedo executable file.

- i) PM identified changes in the Registry as a result of connecting the USB key. The outputs showed the host system querying the USB key for the volume serial number, label, its creation date and time and its last access, write and change dates and times. During the enumeration process, the initial dates and times recorded for the USB key Volume Creation was: 01/01/1601. The Last Access date was 28/08/2014 – the last time the key was used. The last write time was 05/08/2013 – the last time a file was saved to the key. The key was labelled ‘CEEDO’ and this was clearly reported. (Appendix XI a)).
- ii) Five executable files and a number of folders e.g. ‘Program Files’ and ‘My Documents’ were found to be associated with drive letter E:, which had been allocated to the USB containing the vPC. (Appendix XI a)).
- iii) Artefacts were present on the host in captured RAM (Appendix XI b)).
- iv) No references to Ceedo were found in the Registry keys designated for checking (See Appendix VII). No reference to the executable files referenced by PM were to be found in the IconCache.db following system restart.

b) Place key containing Ceedo executable file into Host. Run Ceedo.

Result:

- i) Process Monitor identified a range of activity which was caused by running Ceedo. Notably, whilst the drive containing the program was allocated letter E: and this can be plainly seen, numerous responses to the Host queries which are run across the USB key reference data at drive letter G: (Appendix XI c)).
- ii) Artefacts were present on the host in captured RAM (Appendix XI d)).
- iii) In the Registry, artefacts were retained in the NTUser.dat file at the MUICache key: Software\Microsoft\Windows\ShellNoRoam\MUICache

Readings here showed the executable starting followed by the integral 'SmartPlayer' application, as follows:

G:\Ceedo\Ceedo\SmartPlayer\napplay.exe

A screenshot of this activity forms Appendix XI e).

- iv) An analysis of the host hard drive found that numerous artefacts had been retained by the system, up to and including a reference to Ceedo Technologies Ltd's base address in Rosh Haayin, Israel. Sample results are given in (Appendix XI f)).
- v) Monitoring with PM picked up references to the Ceedo application located on:

\DEVICE\HARDDISK1\DP(1)0-0+5\

in the Prefetch record:

[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf

(Samples are shown Appendix X f).) Whilst a 'Ceedo.exe' prefetch record was retained on the host, it did not reveal that location, referring only to the host hard drive:

\DEVICE\HARDDISKVOLUME1\

- vi) Running Ceedo caused references to be created in the IconCache.db for the named user 'Jan'. These referred to five executables related to the Ceedo program e.g. startceedo.exe, and 'foudme.exe'. (Figure 24).

Figure 24: Icon and textual references in IconCache.db following first run of Ceedo Personal.

```
s.h.e.l.l.3.2...d.l.l.s.h.e.l.l.3.2...d.l.l.". %s.y.s.t.e.m.r.o.o.t.%.\s.y.s.t.e.m.3.2.\w.i.a.a.c.m.g.r...e
.x.e.l.%s.y.s.t.e.m.r.o.o.t.%.\s.y.s.t.e.m.3.2.\s.h.i.m.g.v.w...d.l.l...e.:\c.e.e.d.o.\c.e.e.d.o.\c.
e.e.d.o.i.c.o...d.l.l.

c.:\w.i.n.d.o.w.s.\s.y.s.t.e.m.3.2.\l.o.g.o.n...s.c.r...e.:\s.t.a.r.t.c.e.e.d.o...e.x.e...e.:\f.o.u.n.
d.m.e...e.x.e...e.:\c.p._g.a...e.x.e...e.:\a.u.t.o.r.u.n...e.x.e...e.:\a.u.t.o.d.e.t.e.c.t...e.x.e...e.
\c.e.e.d.o.\c.e.e.d.o.\c.e.e.d.o.i.c.o...d.l.l...e.:\a.u.t.o.r.u.n...e.x.e.,c.:\d.o.c.u.m.e.~1\j.a.
n.\l.o.c.a.l.s.~1\l.t.e.m.p.\a.u.t.o.d.e.t.e.c.t...e.x.e.
```

5.3.3 *LupoPenSuite and PortableApps baseline experiments – results*

Conducting Baseline tests a) and b) on LupoPenSuite and PortableApps produced the following summarised results:

Test a)

For both LupoPenSuite and PortableApps , artefacts naming the vPCs were found in RAM, for example:

```
E:\Lupo_PenSuite_v2013.04_Lite
E:\Lupo_PenSuite_v2013.04_Lite\Apps\*
E:\Lupo_PenSuite_v2013.04_Lite\Apps\eMule
```

And:

```
E:\PortableApps\PortbleApps.com\PortableAppsPlatform.exe
E:\PortableApps\PortbleApps.com\Data\PortableAppsMenu.ini
E:\PortableApps\PortbleApps.com\App\Graphics\Themes\Default\PATheme.ini
```

Nothing to identify the two pieces of software was retained in the Registry or in the IconCache database after system restart.

The volume labels for both vPCs were marked with the names of the software and these were reported by PM.

Test b)

When running LupoPenSuite and PortableApps, the names of the vPC software packages were reported by PM (Appendix XII). In the case of LupoPenSuite, the names of further programs contained within the vPC e.g. CCleaner, were identified by PM (Appendix XII a). In the case of PortableApps, PM picked up references to E:\setup.exe and 'E:\Autorun.inf', both of which are associated with the software (Appendix XII c)). For both vPCs, artefacts identifying the software were retained in Prefetch records and in the NTUser.dat file at the MUICache key. For example, in Prefetch:

\Prefetch\PORTABLEAPPSPLATFORM.EXE-0F20D083.pf

\Prefetch\LUPO_PENSUITE-V2013.04_LITE.E -3A06A9D8.pf

In common with findings made in respect of MojoPac and Ceedo during these experiments, references to the program executable files for both LupoPenSuite and PortableApps were found in the IconCache.db for the named user on the Host following system restart. No references to any hard disk other than the host hard disk were found either during monitoring with PM or subsequent analysis of the host.

5.4 Test scenario Experiments

5.4.1 Test scenarios

The following test scenarios were envisaged with the aim of replicating the types of activity that, in the view of the author, a vPC user would likely wish to carry out once having gained access to a host machine. Each of the activities listed were carried out for each test vPC application in the context of each test operating system.

1. Copy a text file ; vPC to host.
2. Copy a text file ; host to vPC.
3. Copy a picture file: vPC to host
4. Copy a picture file; host to vPC

5. Write and save a text file on the vPC.
6. Run a program executable on the vPC.
7. Launch a browser on the vPC.
8. Conduct a search from a vPC-based browser.

Since MojoPac v.2.1.1.0 tested incompatible with Windows 7, experimentation on this vPC was limited to Windows XP professional. All of the other test results which follow were recorded on Windows 7 Professional 32 bit systems using Ceedo Personal v. 5.0.1.7, PortableApps v.11.2 and Lupo PenSuite v. 2013.04_Lite. Identical procedures were followed in every case.

In the interests of brevity, only the most noteworthy results are recorded in the following sections.

5.4.2 Tests 1 - 4

Using copy and paste, when either a text file or a picture file was copied from the host to the vPC, no artefacts which pointed to this action having happened were apparent in the key system areas chosen for scrutiny on static systems although the 'Accessed' date/time of the file copied from the host was altered. When a text or picture file was copied from the vPC to the host, no artefacts to show the source drive or vPC were apparent in the key areas examined. However, the 'Modified' date and time of the file preceded the 'Created' and 'Accessed' dates and times. This type of finding commonly indicates that a file has been created on some device other than the host and has been transferred from an external drive to the host.

During testing on MojoPac, it was noted that when the vPC is running, it is allocated drive letter C:. On a host with one partition, the host drive was not accessible from within the MojoPac environment. Transferring files from vPC to host via copy and paste was not supported. It was possible to right click a file on the host and choose the 'Send to' option, thus transferring it to the MojoPac environment. No artefacts were found on the host as a result of this action. On a host with two partitions it was possible to copy a file directly from the MojoPac user's 'Documents' folder to the host hard drive and vice versa. The sole artefact recovered following a copy-paste action from vPC to host was retained in the StreamsMRU key on the host, which contained the following data:

G:\Documents and settings.....RINGCUBE

The name of the document was not revealed.

In all cases, following experiments 1 - 4, the names of the files copied between host systems and vPCs during testing were found to be present in live memory dumps but there was no clear indications as to where the files had been copied to or from. No artefacts were found in the pagefile.

5.4.3 Test 5

MojoPac allows a full application to be installed within the vPC environment. For this experiment, Microsoft Word 2003 had been preloaded onto the vPC. No version of Word existed on the host. Word was launched and used to create and save a document within the vPC. No artefacts to suggest that a document had been created and saved within MojoPac were found.

For Ceedo, LupoPenSuite and PortableApps, Notepad ++ was used to write and save a text document within the vPC. The results monitored showed that for all these vPCs, evidence that Notepad ++ had been run from within the named vPC on an external drive was held in the UserAssist key (Figure 25.). The name of the document which had been created was not discernible when using Ceedo Personal. However, for both PortableApps and LupoPenSuite, artefacts were found. In both cases, the named file could clearly be identified as existing within the program's 'Documents' folder on the external drive both within the Registry's ComDlg32 key (Figure 26) and in an associated .lnk file.

Figure 25: Use of Notepad ++ from within Ceedo Personal identified in the UserAssist key

Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{

Last Written Time	20/03/2014 19:36:45 UTC
-------------------	-------------------------

<i>Raw:</i> R:\Prrqb\Cebtenz Svyrf\Abgrcnq++\abgrcnq++.rkr <i>ROT13:</i> E:\Ceedo\Program Files\Notepad++\notepad++.exe
--

Figure 26: Create and save of a document within PortableApps identified in selected ComDlg32 key locations

**Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\OpenSavePid
IMRU***

Last Written Time 04/12/2014 09:54:07 UTC

(ASCII String) ...P.O.

.i.....+00.../E:\.....X.1.....=D5...DOCUME~1..@.....=D5.IE..*.....
D.o.c.u.m.e.n.t.s....j.2.....MoreMalware.txt.L.....*.....M.o.r.e.M.a
.l.w.a.r.e...t.x.t.....

5.4.3 Test 6

When a program executable was run from within a vPC, in the case of MojoPac alone, no artefacts to suggest that the action had occurred were visible either in the host's Registry or the IconCache.db. For all the other vPCs under consideration, evidence that a program executable had been run was recorded in the UserAssist Registry key. Where the deletion software 'Eraser' was started from within PortableApps, for example, the named executable was retained as follows:

F:\PortableApps\EraserPortable\App\eraser\Eraser.exe

As was found during baseline experimental testing, icons for the parent vPC executable programs under examination were retained in the IconCache.db, along with textual records. However, icons for programs run from within the vPC environments, such as Eraser and Notepad++, were not recorded. No artefacts were apparent in the pagefile.

5.2.4 Test 7

When the browser Firefox Portable was launched on test vPCs, a record of the executable having run on the external drive was retained in the UserAssist key when using PortableApps and Lupo PenSuite. For PortableApps, a record was also located at:

[root]/Windows/System32/config/System.Log

as follows:

PortableApps\FirefoxPortable\FirefoxPortable.exe

and for Lupo PenSuite a record was located at:

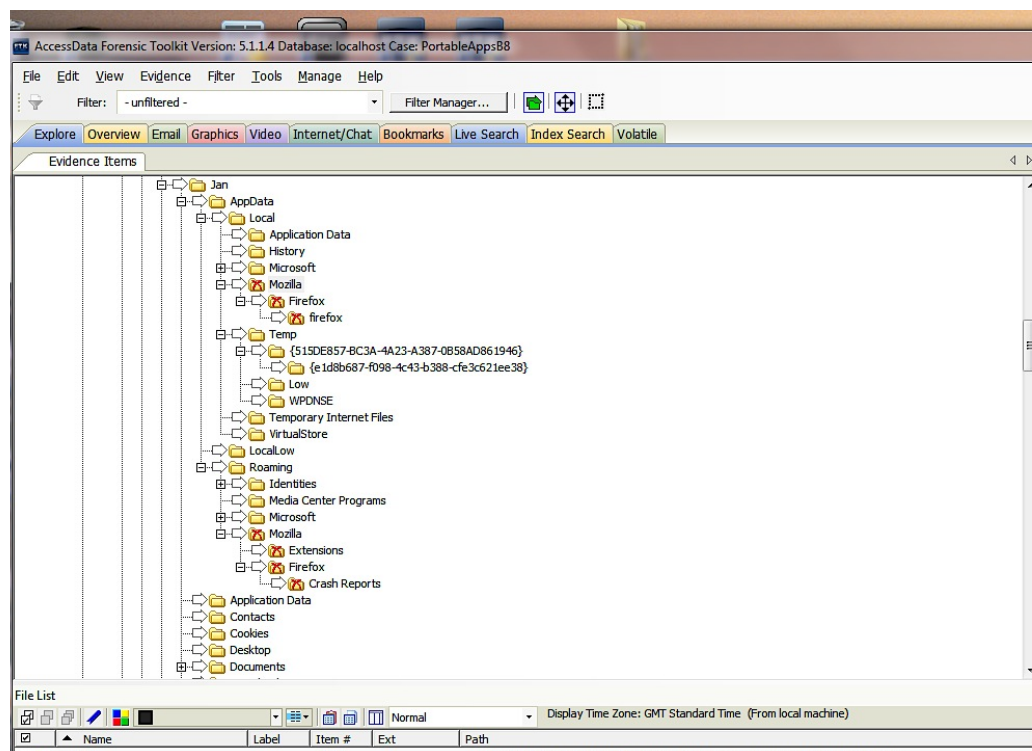
[root] /Windows/System32/Config/System

as follows:

\Lupo_PenSuite_v2013.04_Lite\Apps\Firefox Portable\FirefoxPortable.exe

In the case of PortableApps, further artefacts were found the \Explorer\Software key in the user's NTUser.dat file. Running the browser in PortableApps also resulted in deleted folders being kept on the host system which were clearly viewable in forensic software (Figure 27). No data was retained in the deleted folders.

Figure 27: Artefacts from Firefox Portable run within PortableApps retained on host



5.4.5 Test 8

The browser preloaded in Ceedo was Firefox rather than Firefox Portable. It was found that an uninstall record was left in the UserAssist key after running the browser from within the tool and closing out of the vPC, as follows:

E:\Ceedo\Program Files\Mozilla Firefox\uninstall\helper.exe

The browser preloaded in MojoPac was Explorer. No evidence to suggest that the browser had been used was found on the host.

Search terms were entered into each browser on each vPC after connecting the host system to the internet. No artefacts identifying the search terms used were found on the host systems during static analysis. Testing with PM indicated that internet usage data were written back to the vPCs themselves and analysis using IEF v.6.3.2 showed this to be the case: the search terms were located within the vPC volumes.

5.4.6 Tests in Windows 7 64-bit

For all the Windows 7 compatible vPCs tested, experimental results indicated that artefacts were retained in a similar way to 32-bit systems. i.e. the same Registry keys were populated with information, as were other areas selected for investigation, for example prefetch files and the IconCache.db. Additional findings of interest included the following: for Ceedo Personal, on opening the program, artefacts were retained both in the MountPoints2 key within the user's NTUser.dat file and in the IconCache.db. Use of the program Notepad++ was also shown in the UserAssist key but use of Firefox was not. No trace of a document created and saved within the software was apparent in the Registry. However, a keyword search for the text files copied Host to vPC and vPC to Host located a log file containing both names at:

[root]/Users/User/ntuser.dat.Log 1

The content of this file requires further in-depth analysis but it seems to associate the two text files with the writing package used to create them (Notepad++) within Ceedo at drive letter E:. In the following extract (Figure 28), the file copied Host to vPC was 'StateSecret.rtf' and the file copied vPC to Host was 'MyHakz.txt'.

Figure 28: Artefacts retained on host following text file movements using Ceedo Portable.

```
BDCDA39A394A}.notification.1 ItemPos800x600x96(1) STATES~1.RTF tD1dtD1d*  
StateSecret.rtf MYHAKZ~1.TXT tDadtDad* MyHakz.txt Ceedo Repair  
\Users\User\AppData\Local\Temp\AutoDetect.exe /drive=E: /repair /name=Ceedo  
/id={8ACD63E8-70AF-4728-AAB7-9C1308A2776F} /params="" R:\Prrqb\Cebtenz  
Svyrf\Abgrcnq++\abgrcnq++.rkr
```

The final line of this extract is encoded Rot13 and translates as:

E:\Ceedo\Program Files\Notepad++\notepad++.exe

This value matched that found in the UserAssist key following this experiment.

5.5 Further results

As a further result of the experiments carried out, it was also possible to draw up a table of useful search terms for each vPC tested. These terms, which revealed artefacts present in both live and unallocated space, are given in Table 9.

Table 9. Search terms for test vPCs in Windows XP & Windows 7 (32-bit)

Ceedo Portable	A	U	MojoPac	A	U	Lupo PenSuite	A	U	PortableApps	A	U
ceedo	Y	N	mojo	Y	Y	Lupo_pensuite	N	Y	portableapps	N	Y
ceedoico	Y	Y	mojopac	Y	Y	Lupo_pensuite_v2013	Y	N	portableappsplatform	Y	N
ceedologs	Y	N	ringcube	Y	Y						
ceedodriveinfo	Y	N	ringthree	Y	Y						
ceedodrive	Y	N	juniper	Y	Y						
ceedodatadrive	Y	N	r3cleanup	N	Y						
ceedoblockedreport	Y	N	ringthree_computer	Y	Y						
ceedo_processenforcement	Y	N	mojopac_id	N	Y						
ceedoenvironment	Y	N	mojopac_guestbar	N	Y						
ceedomutex	Y	N									
smartplayer	Y	N									
ceedort	Y	N									
napplay	Y	Y									

Key= A = Allocated space U = Unallocated space

5.6 Chapter Summary

This chapter has detailed a range of experiments which have been conducted in order to monitor the effect of connecting USB-bound vPCs to Windows XP and Windows 7 host systems and carrying out various user-initiated activities. The utility Process Monitor was used to audit the activity which occurred, initially by introducing four test vPCs to a host, then by introducing the test vPCs to a host and running them. The results from these baseline tests were compared against those obtained from a control experiment in which a virgin USB stick was introduced to a host. A data set of interest was thus derived.

Further experimentation then took place in which a number of activities e.g. copying and pasting a file, writing and saving a document, were carried out using the test vPCs. For every experiment, a RAM dump was taken before system close. This was inspected for findings of potential interest, as was the test host hard drive during subsequent static analysis.

It was found that, for the systems tested, connecting a vPC to a Windows hosts can create numerous artefacts of interest to digital forensic examiners. The most informative of these will be found in Registry keys as well as in Link files / Shortcuts, Prefetch and the IconCache database. At a minimum, analysis of these artefacts will enable an enquirer to establish the name of the vPC environment invoked, the user

name under which it was introduced to the host and which programs were run from within it, together with relevant dates and times, the drive letter allocated to the containing USB key plus details enabling identification of that key, such as the make and serial number. All of this information is available when a computer has been closed down using the traditional 'pull the plug' method. The pagefile may be a further resource on static systems. Where the collection of live memory is possible, this can reveal the names of files copied between a vPC and the host although file paths may not be available. However, for some vPCs, the names of files created and saved within the miniature environment are retained on the host, together with the file path. Folders temporarily created on the host when a portable browser is used from a vPC and which are afterwards automatically deleted may also be visible within forensic software. This type of finding could further usefully inform a digital forensic investigation.

CHAPTER 6.

ANALYSIS AND DISCUSSION

6.1 Introduction

This chapter reviews the analysis of data reported in Chapter 5 and discusses the findings, from which conclusions are drawn. The *modus operandi* is that of a digital forensic examination which occurs after a suspected breach of data security where the perpetrator is thought to be a member of staff working at the target organisation's premises. The type of organisation envisaged is the small to medium enterprise where computer network activity is not continually monitored using specialist software. Where IT staff are employed, these are primarily engaged in keeping the network available for use, observing user behaviour only when some unusual event occurs e.g. suspected data compromise.

The discussion points presented here centre around the artefacts which were either discovered during analysis or not discovered, despite a reasonable expectation that they might be present on a host system following the experimental actions taken. Possible reasons as to why such artefacts were created and retained, or not, are explored. The intention is not to offer a comprehensive reference work which identifies every Registry hive or other potential repository of artefacts on Windows computers which may be of investigative interest. Instead, the aim is to justify the hypothesis of this thesis by demonstrating that it is possible, using the methods described in preceding chapters, to extract and analyse artefacts of potential evidential interest from system files that have been created on Windows hosts by miniature computing environments running from USB devices.

6.2 Baseline test results.

6.2.1 Baseline Test a), 'Standard practice' analysis

A number of initial steps are undertaken when a digital forensic examiner suspects that an unauthorised USB device or devices may have been connected to a host system. These steps were followed during the analysis of baseline test results.

The majority of modern USB connectable devices are ‘plug and play’ (PnP) compatible. That is, they come to the marketplace complete with the driver files needed to make them work on Windows computers. As noted in Chapters 2 and 5, the Windows operating system queries a connected device during installation. The query process elicits information from the USB device such as its particular ID, serial number, make, manufacturer etc. These events are recorded by the host system in specific files e.g. setupapi.log and certain keys are created and populated in the Registry. A drive letter is allocated to the connected device and it is made available to the user.

As was found during initial experimentation when Process Monitor was used to record activity on test systems, in every case where a vPC was attached to a host machine, Windows ran the enumeration process, eliciting information about the container USB stick. This action caused data to be retained in the Registry at:

```
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet00x\Enum\USBSTOR
“ “ \SYSTEM\ControlSet00x\Enum\USB
```

In every case, it was thus possible to discover that a USB memory stick had been introduced to the host, the make and model of that device, its serial number and its unique ID. Also, in every case it was possible to discover the drive letter which was assigned to the container USB by the host system at the Registry key:

```
HKEY_LOCAL_MACHINE\SYSTEM\MountedDevices
```

Since this value is repeated in the ntuser.dat hive of the user profile under which the action of connecting the USB drive is taken, it was possible to link the two pieces of data. Furthermore, the date and time of first connection of the test devices could be verified by analysing the setupapi.dev.log in Windows XP and setupAPI.dev.log and setupapi.app.log in Windows 7, as would be expected during a standard digital forensic investigation.

6.2.2 Baseline Test a), IconCache.db analysis

A new technique for investigating the connection of USB devices plus the presence and potential use of program executable files from such devices has been developed

during this research. This technique centres around analysis of the IconCache database.

As explained in Chapter 4, the IconCache.db is a hidden file which is created and maintained on computers running Windows operating systems. The file not only stores icons associated with any programs which are invoked on a Windows host, it also stores icons associated with any programs which are invoked from a connected drive. A textual record is kept of these activities and since separate IconCache records are kept for each named user of a computer, artefacts in this file can be attributed to a specific user account.

During baseline experimental testing for Set a) it was found that the IconCache.db can retain icons from programs which exist on a USB drive but have not been used. This proved to be the case with MojoPac, for example. Attaching a USB containing this vPC resulted in a record being kept although the software had not actually been run.

6.2.3 Baseline Test a), RAM & Pagefile analysis

RAM was captured after conducting each experiment and the Pagefile was analysed for relevant artefacts. For all vPCs, references to the named software could be gleaned from RAM together with the allocated drive letter. No artefacts were discovered in the Pagefile after completing Test a) for each vPC.

- Summary Result

This research has shown that:

- a) connecting a vPC to a Windows host does not preclude the operating system from running the enumeration process for USB connectable drives.
Information which helps to identify the drive containing the vPC is therefore stored in the Registry.
- b) Existing analysis techniques for static hard drives can be used to locate this information plus the dates and times of suspect device connection.
- c) The IconCache.db can retain artefacts of potential evidential interest and is therefore a useful new resource.

- d) The analysis of RAM has limited potential under these test conditions.
Analysis of the Pagefile may not add to the enquiry.

6.2.4 Baseline Test b), 'Standard practice' analysis

A number of Registry keys or 'hives' are automatically populated when a USB device is connected to a Windows system, as has been discussed above. Any further action taken – in the case of Test b), running a program executable – may cause other artefacts to be created in the Registry. Starting a new application, for example, should populate the UserAssist key and MUICache key. These are located in the NTUSER.Dat file at:

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\User Assist

and

HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\MUICache

HKEY_CURRENT_USER\Software\Classes\LocalSettings\Software\Microsoft\Windows\Shell\MuiCache

HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\MUICache

The UserAssist key keeps a record of the applications which have been launched on a system, the number of times those applications have been launched and associated date and time data. However, none the vPCs tested created artefacts in this key when the executable files were run.

Similarly, a record is also created in the MUICache key when new applications are started although date and time values are not retained. During experimentation it was found that, in every case, launching a vPC did populate this key.

Outside of the Registry, the prefetch folder would be expected to retain a record of any programs which may have been run on a Windows system. The mechanism, which saves information as a number of small files, is intended to help speed the computer start-up process. Analysis carried out on test hard drives after experimentation showed that prefetch records which identified the vPCs in use did exist after running

the software. Interestingly, the record created by using MojoPac showed the host recognizing the vPC as a 'hard disk'. This was indicated by the instruction:

```
\DEVICE\HARDDISK1\DP(1)0-0+5\
```

This instruction is intended to help the host to map events to the correct volume. According to Microsoft (2015)⁷, the drive type indicated here is a CD Rom (Type 5) rather than a removable drive (Type 2).

Monitoring with PM showed that Ceedo identified itself as existing on a Type 5 volume in the same way and a prefetch record revealing that location was retained. Monitoring LupoPenSuite and PortableApps with PM did not produce the same finding - no reference to either vPC as existing on an external 'hard disk' or device was discovered during analysis.

In a scenario where a digital forensic examiner had checked both the Registry and the prefetch folder and therefore established the name of any vPC which may have been used on a host, the examiner might then be likely to sweep the host drive for any other references to the vPC and associated software. This could be achieved by indexing the hard drive and searching for specific known terms. As has been shown, a large number of artefacts would be found in allocated and unallocated space to help to confirm and/or flesh out previous findings.

A further standard method of analysis would be to check through a listing of .log files and .exe files. This could identify unusual or unexpected data. Checking .bat files – an activity usually carried out where virus infection or hacking attempts are suspected - would also be advisable. Launching MojoPac, for example, caused the 'R3Cleanup_.bat' to be run, leaving artefacts on the host.

6.2.5 Baseline Test b), IconCache.db analysis

Running all test vPCs resulted in artefacts being retained in the IconCache.db. Apart from MojoPac, the action of running a vPC was the first to cause artefacts to be created in the database. It has been shown that the MojoPac icon can appear in the

⁷ [http://msdn.microsoft.com/en-us/library/aa394173\(v=VS.85\).aspx](http://msdn.microsoft.com/en-us/library/aa394173(v=VS.85).aspx)

IconCache.db even when the software has not been launched, however the textual record will only refer to the program's installer executable. Running MojoPac resulted in more information being written to IconCache.db. The 'start' executable was shown to have run, as was the 'ringthreemainwin32' executable. These findings correlate with those found in the MUICache key, verifying that these activities occurred.

For Ceedo, the textual record in the IconCache could also be compared with findings in MUICache, where four related executables were shown to have run. Notably, however, the drive letters for the activity were different in the two records. During this experiment, monitoring with PM showed the USB containing Ceedo being allocated drive letter E: whereas other queries made by the operating system during enumeration of the drive referenced drive letter G:. The record retained in IconCache.db in respect of executable files linked with Ceedo consistently referenced drive E: the record retained in MUICache for matching executable file names referenced drive G: Further research would be needed in order to ascertain why this occurred. Nevertheless, the activity can be cross-referenced between the two records. Where, for example, in the MUICache record, we have:

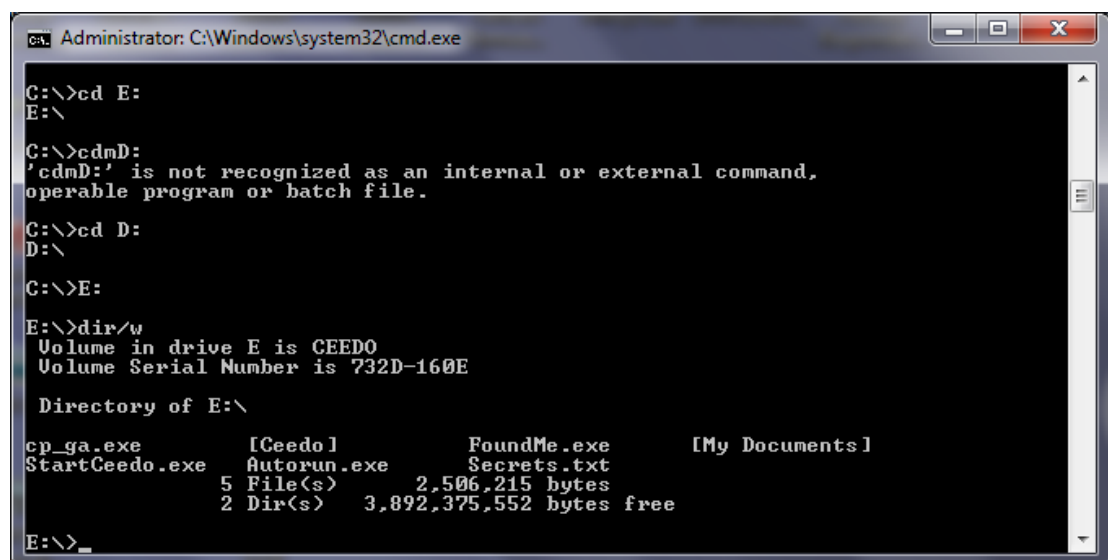
```
G:\StartCeedo.exe  
G:\Autorun.exe  
G:\Ceedo\Ceedo\SmartPlayer\napplay.exe  
C:\Docume~1\Jan\LOCALS~1\Temp\AutoDetect.exe
```

In the IconCache.db textual record we have:

```
e:\s.t.a.r.t.c.e.e.d.o...e.x.e.  
e:\f.o.u.n.d.m.e...e.x.e.  
e:\c.p._g.a...e.x.e  
e:\a.u.t.o.r.u.n...e.x.e.  
e:\a.u.t.o.d.e.t.e.c.t...e.x.e.  
e:\c.e.e.d.o.\c.e.e.d.o.\c.e.e.d.o.i.c.o...d.l.l  
e:\a.u.t.o.r.u.n...e.x.e.  
c:\d.o.c.u.m.e.~1\j.a.n.\l.o.c.a.l.s.~1\l.t.e.m.p.\a.u.t.o.d.e.t.e.c.t...e.x.e.
```

While the 'napplay.exe', which is associated with Ceedo's 'SmartPlayer' programme was not directly referenced in the IconCache.db, the line: E:\ceedo\ceedo\ceedoico.dll' was present and would appear to refer to the 'SmartPlayer' activity recorded in MUICache. This can be surmised from the MUICache record, since activities are ordered as they occur, the most recent activity being the last record shown in the key's listing. The IconCache.db also appears to record activities in order of occurrence. As noted in Chapter 4, previous research on the IconCache.db has found that the names of executable files present on attached drives may be recorded in the artefact if those executable files exist in the root of the drive. The names of executable files found in the IconCache.db but not shown in MUICache following this experiment were located in the root of the drive (Figure 29). Where the name of an executable file is found both in the IconCache.db and in MUICache, one can infer, from this finding, that it has run. Uncertainty would remain about the other executables named in the IconCache – certain malware may be capable of running from an attached drive without leaving a trace in MUICache, for example. Further testing would be required in order to research this possibility.

Figure 29: Executable files in the root of the drive containing Ceedo



```
Administrator: C:\Windows\system32\cmd.exe
C:\>cd E:
E:\
C:\>cdmD:
'cdmD:' is not recognized as an internal or external command,
operable program or batch file.
C:\>cd D:
D:\
C:\>E:
E:\>dir/w
Volume in drive E is CEEDO
Volume Serial Number is 732D-160E

Directory of E:\

cp_ga.exe          [Ceedo]          FoundMe.exe      [My Documents]
StartCeedo.exe    Autorun.exe     Secrets.txt
                  5 File(s)      2,506,215 bytes
                  2 Dir(s)      3,892,375,552 bytes free

E:\>_
```

6.2.6 Baseline Test b), RAM & Pagefile analysis

For all vPCs, references to the main executable for the named software were found in RAM together with associated drive letters allocated to the containing drives. Activity occurring automatically e.g. the launch of ‘SmartPlayer’ following the launch of the Ceedo executable was discernable. Readings were difficult to interpret for MojoPac as the vPC allocates itself drive letter C: . Since MojoPac has its own Registry and other Windows operating system features, activity attributed to drive letter C: might refer to events occurring on the host or to events occurring in the MojoPac environment. Nevertheless, it was possible to associate the vPC with the drive letter allocated to its containing drive, E:.

For MojoPac alone, artefacts were discovered in the Pagefile after Test b). While this result was disappointing, the results which might be obtained from a computer in every-day use could be much more informative. It should be noted that the host systems used during experimentation were running the native OS alone; no applications had been installed. This would mean that limited demands were being made upon memory. Also the systems were only run for short periods, therefore there was little time for artefacts to accumulate in the pagefile. Analysis of the pagefile in a real-life investigation, where a suspect computer would be highly likely to have multiple applications installed and would have been active for many months or years should therefore be considered worthwhile.

- Summary Result

This research has shown that:

- a) Running a vPC on a Windows host will cause numerous artefacts to be created on the host. These will be retained in allocated and unallocated space and may be discovered using existing analysis techniques for static hard drives.
- b) Artefacts of potential evidential interest will be created in the IconCache.db. Readings obtained from this file can be compared with those obtained from the MUICache key in the Registry, helping to identify what programs have been run from a vPC.
- c) The analysis of RAM can also help to identify activity which has taken place as a result of running a vPC on a Windows host.

- d) Analysis of the Pagefile may result in findings, this avenue of enquiry is therefore worthwhile.

6.3 Scenario-based experimental results.

6.3.1 Copy and paste a text or picture file (Experiments 1 – 4).

Save for MojoPac, copying and pasting test files from vPC to host did not create any unique artefacts on the host. Copying and pasting a file from one volume to another will not create a link (.lnk) file. A link file is only created when a file is opened on or from some source. For this experiment, therefore, the absence of .lnk files was not surprising. As noted in Chapter 5, however, the ‘Modified’ date and time of files copied from a vPC to the host preceded the ‘Created’ and ‘Accessed’ dates and times. This type of finding commonly indicates that a file has been created on some device other than the host and has been transferred from an external drive to the host.

The result obtained when copying a file from MojoPac to a host was interesting. The sole artefact was a reference to ‘Ringcube’ in the Registry at the StreamMRU key located under:

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Streams.
```

In Windows XP, the StreamMRU key stores up to 28 entries which relate to recent ‘View’ preferences for the desktop and any windows which have been opened within it. When a window is closed, the ‘View’ preference is written to a subkey. That MojoPac created an artefact here suggests that the action of opening and closing the ‘Documents and Settings’ folder within the vPC file system populated StreamMRU. Although launching the vPC opens a window on the host desktop, this key was not populated during Baseline experiment b), when the software was run but no other action took place.

The Windows 7 Registry does not contain a StreamMRU category. Data which was stored in this area under Windows XP, as well as previously available Shell and ShellNoRoam data, is stored in two subkeys in Windows 7 at:

NTUSER.DAT\Software\Microsoft\Windows\Shell

and

USRCLASS.DAT\Local Settings\Software\Microsoft\Windows\Shell

which is found in the user profile of interest at: \AppData\Local\Microsoft\Windows. When this experiment was conducted for all vPCs except MojoPac on a Windows 7 host, however, no entries of potential evidential interest were found under these Shell subkeys.

6.3.2 Write and save a text file on the vPC (Experiment 5)

Following this experiment, the UserAssist key in the Registry was found to hold evidence that a word processing program had been used from an external drive for all vPCs except MojoPac. As noted in 6.2.4, above, the UserAssist key keeps a record of the applications which have been launched on a system, the number of times those applications have been launched, together with associated date and time data. A possible reason why data was not retained in this key when using MojoPac is that the vPC has its own Registry, avoiding the need to use that on the host.

For the vPCs PortableApps and LupoPenSuite, it was possible to discover the names of the documents created during this experiment and the associated file paths. This information was found in the ComDlg32 subkey in the Registry. Under normal conditions, the names of opened and saved files will be stored here as a list, the most recently used files in terms of date and time being shown under the key name MRUList. In Windows 7, though, software developers can set a flag in their applications so that this information is not recorded. As Ceedo Personal left no data in this key, it is possible that its developers programmed the software in this way.

6.3.3 Run a program executable on the vPC (Experiment 6)

The UserAssist key was again the chief source of information following this experiment for the reasons documented above. While the IconCache.db would normally be expected to retain a record of executable files which have been run on the

host or from other media, it was noted that the database did not keep a record of executable files run from within a vPC environment. A possible explanation is that the icons for these executables are not held in the root of the vPC container drive but instead are kept in program folders in the vPC's file system. They are potentially therefore referenced and rendered from within the vPC.

6.3.4 Launch a browser on the vPC (Experiment 7)

A browser is usually launched by double clicking on the icon for the relevant program executable file. This action would normally cause a record to be kept in the UserAssist key in the Registry. MojoPac has the browser Internet Explorer preinstalled. Launching this from within the program did not populate the UserAssist key in Windows XP. A possible explanation is that the information was collected within the Registry on the vPC, thus bypassing that on the host.

Other test vPCs had versions of the Firefox browser installed. In the case of Ceedo, launching the application and afterwards closing out of the vPC did cause an artefact to be left in the UserAssist key in Windows 7. It was shown that a Firefox uninstaller program had run. This result suggests that Ceedo is programmed to clean up after itself, at least where Firefox has been used. A similar action appeared to have occurred after launching the Firefox Portable browser in Portable Apps. While running Firefox Portable within this vPC did populate the UserAssist key, further evidence to corroborate its use was found during analysis at:

%username/AppData/Roaming

The AppData/Roaming folder holds data which can move with the user profile from PC to PC where the machines in question are on the same network so that synchronisation of information such as web page favourites and bookmarks can occur. A deleted folder marked 'Mozilla' existed in this location on the host, together with associated subfolders. No data was visible within the deleted folders, nevertheless, dates and times associated with the creation of the folders were retained. These tallied with the dates and times retained in the UserAssist key for the launch of Firefox Portable. Other records that the software had been run were created in the Prefetch

folder and a System log file, however, specific date and time information was not available in the log file.

Similar Prefetch and System records were found during analysis after this experiment was conducted on LupoPenSuite and the launch of Firefox Portable likewise caused artefacts to be retained in the UserAssist key but for this vPC, no traces of its use were found in the AppData/Roaming folder.

6.3.5 Conduct a search from a vPC-based browser (Experiment 8)

Following this experiment, each host hard disk was imaged and a full index of the contents was made. As well as investigating common areas for evidence of browser usage and searching, such as browser cache, history and cookies, keyword searches were made for the specific terms used during experimentation. For all the vPCs tested, none of the specific terms e.g. YellowAlligator were found. Process Monitor was used to identify activity during each experiment and it was observed that browser usage data was being written back to the vPC in play, rather than to the host. When the vPCs were analysed individually using IEF v.6.3.2, the search terms which had been entered in at each browser were in fact found to be stored on the vPC concerned.

6.3.6 Comparative results - Windows 7 64-bit

For all compatible vPCs, the scenario-based experiments discussed above were again conducted on Windows 7 64-bit systems. The areas of the operating system which yielded results of investigative interest on 32-bit systems were found to do the same on 64-bit systems. An additional finding of note was that running the Ceedo Personal vPC caused artefacts to be written to the MountPoints2 key within the named user's NTUser.dat file. Data found in the MountPoints2 key can be used to tie a particular user to a particular suspect device – the GUID (Globally Unique Identifier) of the suspect device would potentially be retained here. However, on Windows 7 systems, a number of users can be logged in at the same time. When one logged-in user inserts a USB device, the MountPoints2 key for all logged-in users will automatically be updated with that information. This means that a forensic examiner needs to interpret any reading found here with caution. During this research experimentation, the interpretation was straightforward since – barring one experiment - only one named

user (Jan) had been set up on the test host system. The introduction of Ceedo Personal to the test host could therefore be attributed to the named user. Further corroboration was found in the IconCache.db where a textual record showing the launch of Ceedo from an external drive was retained. This discovery underpinned, once more, the usefulness of analysing the IconCache.db. This author's research on the database has shown that an IconCache.db record is kept for each individual user of a Windows computer. During testing, actions such as running a program executable caused a record to be created in the IconCache of the user responsible. In a further experiment, when a second user (Max) was set up on a test system, no update was made to the IconCache.db record for user Max when this user was logged in at the same time as the first user on the same machine. Whilst further research would be appropriate in this area, this finding was encouraging. Multiple user log-in is enabled by default on Windows 7 through the operating system's Fast User Switching mechanism. Although the mechanism exists in Windows XP, it is disabled by default thus on a multiple user system, one active user must log out before another can log in.

Another finding of interest which was made during experimentation with Windows 7 64-bit systems was that a record identifying documents copied to and from a Ceedo vPC was retained on the host. The record occurred in a log file associated with the NTuser.dat file for the named user. Further experimentation would be necessary in order to clarify why this artefact was created but some form of arbitrary data corruption is suggested in that a 'Ceedo Repair' appears to have been instigated from the vPC via the software's 'AutoDetect.exe' program. Whatever the rationale, the information written back to the host was very useful. In a real-life investigation where, for instance, it was suspected that a sensitive document had been copied from a corporate host machine, the name of the document would be known and could be searched for; a search which might well locate this type of data. At a minimum, an analyst would be made aware that a Ceedo vPC had been introduced to the host – something that may not have been known before – and that Notepad++ , a program not installed as standard on Windows 7 , had also been run. Furthermore, the existence of a rogue text file such as 'MyHakz.txt' would be a cause for concern. This unusual or unexpected file could now be searched for on the host. Finally, that the host operating system stored this information in a log file affirms the importance, during a digital

forensic examination, of taking time to check the contents of log files since they can often prove useful in this way.

- Summary Result

This research has shown that:

- a) Copying and pasting files between a Windows host system and a vPC is likely to cause artefacts of potential evidential interest to be created on the host, though these may be subtle e.g. Updated MAC dates and times on a host-resident file may not engender instant suspicion but could be important if an investigation involving potential data theft was already underway.
- b) Creating and saving a document on a vPC is likely to cause artefacts to be created on a host which can be located during a static analysis. Depending on the vPC being used, details of the names of such saved documents and relevant file paths may be retained on the host, particularly in the Registry. An analysis of any Log files retained on the host can also be worthwhile.
- c) Running a program executable from a vPC is likely to cause artefacts to be created in the UserAssist key within a named user's NTUser.dat file but not within the IconCache.db for that user if the executable concerned is not located in the root of the drive.
- d) Running a browser from a vPC is likely to cause artefacts to be created in the UserAssist key. Depending on the vPC, other records may also be retained in system files.
- e) Conducting a word search in a browser running from a vPC is unlikely to cause artefacts that identify the search terms used to be retained on the host system.
- f) Evidence that user-initiated activities have been conducted from a vPC can be found during analysis of a static host hard disk. These activities can be attributed to a specific user name existing on the host.
- g) Where it is possible to capture live RAM, further corroborative evidence of vPC user-initiated activity will be obtained.

6.4 Chapter summary

The hypothesis of this thesis is that it is possible to extract and analyse artefacts of potential evidential interest from system files that have been created on Windows hosts by miniature computing environments running from USB devices.

With that aim in mind, this chapter reviewed and discussed the findings made as a result of the experimentation conducted during this research project and detailed in Chapter 5. The discussion centered around the artefacts of potential evidential interest which were created on test host computers as a result of introducing vPC software to them in a variety of ways. The absence of artefacts which a digital forensic examiner would normally expect to be created under circumstances where an unauthorized USB device had been attached to a workstation and certain user-initiated activities had been carried out was also considered. The analysis techniques employed included those which are currently standard practice in the industry where a static hard disk is the data source. A new technique, developed by the author, which involved analyzing the IconCache database was also used. Live RAM was captured during experimentation by means of taking a memory dump from the live test host before inducing a system halt. The contents of these memory dumps were discussed in this chapter in the context of other findings where the additional information discovered was helpful.

The outcomes from two ‘baseline’ tests were analysed. Where a vPC was introduced to a host but the software was not run, it was found that the Windows host nevertheless carried out an enumeration process for the containing USB. As a result, through examining Registry keys and other system files, it is possible for a computer analyst to discover that a suspect USB memory stick has been introduced to the host, when the incident occurred and under which user name, the make and model of the device involved, its serial number, its unique ID and the drive letter which was assigned to it. The IconCache.db was also shown to have potential as a resource – an identifiable trace of the vPC MojoPac was left in this artefact, even though the software was not run. This type of finding could alert a computer analyst to look for further traces of vPC software, something which may change the course and face of an investigation.

In a second baseline experiment, the test vPCs were launched. For every vPC under review, this action alone caused a large number of artefacts to be created on host systems both in allocated and unallocated space. These could be discovered using existing analysis techniques for static hard drives. Analysis of the IconCache.db was again shown to be worthwhile. Artefacts of potential evidential interest were created in the IconCache.db following this experiment. It was possible to compare readings from the database with those obtained from the MUICache key in the Registry, helping to identify which programs were run from the vPC under consideration.

A number of scenario-based experiments were conducted next. These were based on user-initiated activity, such as copying and pasting files between host machines and vPCs, writing and saving a file on a vPC and carrying out a search via a web browser from a vPC. Results were obtained in almost every test case. While evidence of file copying from a host to a vPC was limited to date changes on the host file, this was nevertheless a useful finding which could inform an existing investigation into potential data theft. Where a document was created and saved on a vPC, it was found that in some cases, the name of the document together with the relevant file path was retained on the host – a significant finding which would very helpfully inform an enquiry. When a browser was run from a vPC, this action could be verified in every test case save one, as the name of the browser software could be identified. It was not possible, however, to discover any search terms which had been used within the test browsers.

CHAPTER 7.

SUMMARY AND CONCLUSIONS

Developments in desktop virtualisation technologies over the past decade have made it possible to run a separate, functioning computer environment from a USB memory stick connected to a host system. Once introduced to the host, these miniature systems – which are termed vPCs, in this thesis - allow a user to carry out every-day activities such as playing games, making, moving and copying files and accessing the Internet. Where a user is keen to preserve personal privacy, the vPC offers the advantage of strong confidentiality. From an information security perspective the technology can be seen as a new threat, expanding the risk of data loss or network corruption already posed by the use of USB memory sticks in general and modern ways of working such as BYOD. An equal concern is that in the hands of a wrongdoer, a vPC could be used to evade detection when carrying out unauthorised or potentially criminal activities. As documented at the beginning of this study, some manufacturers of vPC software claim that running it will either leave ‘no trace’ of user activity on a host machine or that ‘no personal data’ will be retained. Statements such as these imply that secrecy as well as security is ensured, an idea which could be appealing to people with nefarious intentions.

The potential dangers posed by the uncontrolled connection of USB devices have been discussed in previous studies. These dangers are so widely acknowledged that a significant amount of research has centred around the effects of connecting USB devices to Windows-based computers and the analysis of any ensuing artefacts. This is the background against which this research program has been developed, as discussed in Chapter 1. The program builds on the existing body of knowledge in this area of digital forensic science and expands it by presenting and explaining a new artefact which aids investigations involving the unauthorized use of USB devices and USB-bound software. Further, this thesis informs research into antifoensics by showing – as proposed in the hypothesis - that it is possible to uncover evidence that portable virtualisation software has been introduced to and used on a Windows host computer. This is a novel and relatively unexplored field of enquiry.

The problem area identified in this research is broad and five main topics are presented and discussed in Chapter 2. These are: Virtual Machines (VMs), Virtualisation technologies for USBs, Windows memory handling, Windows forensic artefacts and current Best Practice guides for forensic practitioners. This breadth of enquiry is necessary in order to provide a proper context for the design and implementation of the experimentation which is to be conducted. Thus, the VM is defined and its use as a computing environment is described. The potential for the use of VM software to leave artefacts of possible evidential interest on a host machine is discussed, leading to the supposition that the same could be true of miniature VM software which is capable of running from a USB connectable memory stick. The idea that other virtualisation technologies might do the same follows. An overview of how the Windows operating system handles memory is given in order to give a basic understanding of how a running computer deals with data. The types of artefacts that the every-day use of a USB connectable device will create on a host machine are also considered, the better to explain current digital forensic techniques. As is discussed, these can include the capture and analysis of both live and static data but in real-life situations, 'best practice' guidelines can impact on the choices made and the options may anyway be limited by circumstances at the scene of the enquiry.

The software packages chosen for testing are also presented in Chapter 2. These are desktop virtualisations which have been designed as standalone programs: all will run on compatible computers without being installed. The test programs fall broadly into two categories: Virtual Machines and Portable Applications. Virtual Machines allow for programs which are installed within the provided environment to interact with one another whereas in Portable Applications the various software packages made available are designed to run separately. The objectives of the research are set out, these are:

- 1) To explore the potential for portable computing environments to be used to commit unlawful acts in the context of existing concerns regarding the use of USB connectable devices in general.
- 2) To examine selected miniature computing environments and analyse the effects of carrying out a range of common activities via these on an host computer.

- 3) To consider currently available methods of analyzing Windows computer systems for evidence of activity involving a user-connected USB device
- 4) To create a chart of key areas to be examined and analysed when investigating the use of miniature computing environments and/or USB connectable devices.

To this end, the aim of the research project has been to develop a methodology which both informs the forensic analysis of Windows computers where portable virtualisation software is thought to have played a part in the unlawful or unauthorized access of an host system and generally assists the investigation of cases in which the potential unlawful/unauthorized use of USB connectable devices exists. During the project, as detailed in Chapter 3, a series of procedures have therefore been devised in order to meet these objectives. Firstly, baseline experiments were conducted to identify the types of artefacts which would remain on test Windows XP and Windows 7 operating systems following:

- a) the connection of a blank USB memory stick.
- b) the connection of a USB bearing a miniature computing environment (vPC).
- c) connecting a vPC and launching the software.

Comparisons were made between the data sets collected and by this means it was possible to isolate a data set of interest (DSOI) to inform further work. Secondly, a series of defined experiments were conducted. The intention was to reproduce some of the basic activities which the computer user in possession of a vPC would, in the view of the author, likely wish to carry out. These included copying and pasting text and picture files to and from a host, writing and saving a text file on a vPC, launching a web browser and searching for specific terms on the Internet. Each of these activities were carried out for each test vPC application in the context of each compatible test operating system. For every experiment, a RAM dump was taken before system close. This was inspected for findings of potential interest, as was the test host hard drive during subsequent static analysis. The outputs were recorded and analysed and the results were compared and discussed so that the hypothesis of this thesis could be fully evaluated.

7.1 Proof of Hypothesis

It is proposed that the **hypothesis** of this project, namely that it is possible to extract and analyse artefacts of potential evidential interest from system files that have been created on Windows hosts by miniature computing environments running from USB devices (vPCs), has been proved. It was found that, for the systems tested, the introduction and use of USB-bound vPCs on Windows hosts can create numerous artefacts of interest to digital forensic examiners and that the majority of these can be located on a static hard disk using existing methods. At a minimum, analysis of these artefacts will enable an enquirer to establish the name of the vPC environment invoked, the user name under which it was introduced to the host and which programs were run from within it, together with relevant dates and times, the drive letter allocated to the containing USB key plus details enabling identification of that key, such as the make and serial number. The Pagefile may be a further resource. Where the collection of live memory is possible, this can reveal the names of files copied between a vPC and a host, together with relevant file paths. The names of files created and saved within the miniature environment may also be found in RAM. Importantly, for some vPCs, these can be retained on the host, too, as can folders that are created and afterwards deleted when a portable browser is used. These are the types of findings which helpfully inform a digital forensic investigation.

7.2 Contribution to knowledge

Furthermore, it is also proposed that this research program has added to the **existing knowledge base** in the field of digital forensics both by exploring how vPCs interact with hosts and by discovering and investigating the evidential potential of the IconCache database. It is shown, in Chapter 4 of this thesis, that as well as storing system and program-related icons, the IconCache.db also retains a textual record of the activities which it facilitates. Of particular interest are the artefacts retained in the database as a result of user-initiated activity. These include the names of executable files that were run or installed from external media and the associated file path. This finding greatly assisted the hunt for evidence related to vPC activity on a host, as documented in Chapters 5 and 6. Data gathered from the IconCache.db could be correlated with that gathered from traditional areas of investigation within the

Windows operating system, such as the Registry and Prefetch folder, which helped to support and clarify the findings made. Besides being a useful resource where the unauthorized use of vPCs in particular or USB connectable devices in general is suspected, the IconCache.db can aid other kinds of digital forensic investigation. For example, it can be used to show that certain programs existed and were used on a machine, even if those programs have been uninstalled. Likewise, in cases where an external encryption program, malware or antifoensics tool has been used, it may provide the only indication since malware is often written to bypass the Registry and clean up any of its other traces on a host and antifoensics tools normally target certain Registry keys, clearing any information that may be held there. It is advised that analysis of the IconCache.db should be approached with due caution as it may contain the icons of executables which were merely stored on some attached media, not actually run from it. However, this same proclivity might help to reveal the potential source of a rogue program found elsewhere on a computer network. Where the analysis of a single workstation is involved, comparing readings from the IconCache with those obtained from the MUICache key in the Registry can help to establish which programs have been used on that system.

For the above reasons, it is suggested that an analysis of the IconCache database should routinely be incorporated into digital forensic investigations. As well as fleshing out information gathered during an analysis of the Registry and Prefetch folder, readings from the IconCache.db can help to identify new lines of enquiry. This has been proved to be the case in two complex investigations recently carried out by the author, who is a practicing digital forensic examiner. In one instance, involving rival bids for a multi-million pound building contract, an inspection of the IconCache.db revealed the potential use of encryption software by an employee who was suspected of divulging the details of secret negotiations between the parties. No markers from the encryption software had been found in the Registry or elsewhere on the employee's workstation, therefore the IconCache.db was a lone resource. In a second matter, where it was feared that the servers of a secure industrial site had been hacked, an unauthorised user name was discovered and the IconCache.db for it was examined. The names of the malware packages used by the intruder were retained in the IconCache, a finding which greatly increased the speed at which damage-limiting measures could be introduced and the investigation could be completed.

The author's research on the IconCache.db, as detailed in this thesis, has been considered important enough to merit publication in Digital Investigation, a journal widely respected by the digital forensic community worldwide, in 2013. The UK's digital forensic community, as represented by F3 - The First Forensic Forum⁸ - also showed its interest in this research by inviting the author to present her findings at its annual conference the same year.

Information stored in the IconCache.db is especially useful where it is suspected that some unauthorised media has been connected to a system via USB. Devices such as vPCs, which can be used for data movements, Internet browsing and antiforensics will leave their mark in the artefact. The focus of a computer investigation may shift as a result, prompting a deeper and wider search for related artefacts in both allocated and unallocated space. As shown in this thesis, a great deal of helpful information can be discovered during a targeted search for vPC activity on Windows systems. While existing analysis techniques may alert an examiner to some of these activities, additional techniques, as identified in this research, need to be employed in order to obtain a fuller picture of events.

A digest of the author's research in this area was presented at the 11th Annual International Conference on Information Technology and Computer Science of the Athens Institute for Education and Research (ATINER) in May 2015. It was published in ATINER's Conference Paper Series in October 2015 and has been chosen for further publication in a forthcoming issue of the Athens Journal of Sciences.

This research program has been designed with the needs of information security professionals, law enforcement agencies and digital forensic examiners in mind. In an increasingly wired world, the threat landscape broadens daily. System hacks and malware attacks which result in data loss or theft are commonplace, frequently followed by financial and legal recriminations. In this environment, the information security professional must constantly ask: 'Is the data safe?' and 'How can I best

⁸ <https://www.f3.org.uk>

secure data?’ In investigating the vPC, this research helps to raise awareness of a trending technology which could easily infiltrate the corporate environment. If data escapes or is stolen, the questions will become: ‘What happened? When did it happen?’ and ‘Who’s responsible?’ These are questions which law enforcement officers will also ask, together with: ‘How can we prove what happened?’ and ‘How can we prove who’s responsible?’. The digital forensic examiner will be needed to produce the answers, insofar as answers are possible. In the vast majority of cases, computer evidence is circumstantial: it is very difficult to prove whose hands were on the keyboard at any given time. The diligent digital forensic analyst will therefore go to great lengths to build a body of evidence which keeps leading back to a given suspect or source. Any artefact created by the computer operating system which can help the analyst to find out what was happening, when and - ideally - who was responsible for which actions, is a welcome addition to the armory. As well as uncovering and explaining evidence which can show that portable virtualisation software has been introduced to and used on a host computer, this research presents the IconCache.db, a previously unexplored artefact, which fundamentally aids investigations involving the unauthorized use of USB devices and USB-bound software. The ability to trace the installation or use of a foreign program such as a vPC to a USB connectable or other media is important during a digital forensic enquiry. The information could help to narrow down the search for a culprit. It could also help to suggest where other evidence might exist or reveal the potential source of an unwanted or prohibited program found elsewhere on a computer system.

7.3 Further Work

In respect of vPCs and user-activity related activity, further research is needed on static systems in order to establish whether artefacts of potential evidential interest might be recovered from other areas of the Windows operating system, such as volume shadow copies and hibernation files. While results from pagefile analysis during this round of research did not reveal much of note, further testing might also produce something worthwhile. In respect of live data collection, further in-depth research is needed on the contents of RAM. A number of new tools have been developed to aid this type of analysis in the past 18 months and outputs from these could usefully be compared and contrasted with those obtained from older tools. It

would also be useful to explore how various vPCs interact with computer systems running Windows 8 and the upcoming Windows 10 operating system.

In respect of the IconCache.db, the author's research has already been extended by other researchers whose work has helped to clarify its structure and functionality. While these researchers have developed a tool that parses file paths from the IconCache.db under Windows 7 and 8, an opportunity to develop software capable of rendering icons stored in the database together with these file paths remains. Further research is also necessary into the type of information which is stored in the IconCache.db as a result of differences between Windows 8 and Windows 10 and previous operating systems – for example, from the Start Screen 'Live Tiles' used by Windows 8, which also feature in Windows 10.

REFERENCES

ACPO Good Practice Guide for Digital Evidence [Online]

Available from:

<http://www.acpo.police.uk/documents/crime/2011/2011110-cba-digital-evidence-v5.pdf>

[Accessed: June 8, 2014]

ANSON, S., BUNTING, S., JOHNSON, R. and PEARSON, S. (2012) Mastering Windows Network Forensics and Investigation, 2nd Ed., Indiana, USA: John Wiley & Sons. [Chapter 6].

BARRETT, D. and KIPPER, G. (2010), *Virtualisation and Forensics. A Digital Forensic Investigator's Guide to Virtual Environments*. USA., Syngress/Elsevier. [Chapter 5, P102].

BEM, D. and HEUBNER, E. (2007), Analysis of USB Flash Drives in a Virtual Environment, *Small Scale Digital Device Forensics Journal*. [Online]. 1 (1). p. 1 – 6.

Available from: <http://www.mislan.com>

[Accessed: May 3, 2009]

BOSSCHERT, T. (2007) Battling anti-forensics: beating the U3 stick, *Journal of Digital Forensic Practice*, [Online] 1 (4). p. 265-273.

Available from: <http://www.tandfonline.com>

[Accessed: October 25, 2010]

CARLTON, G. and KESSLER, G. (2013), Identifying Trace Evidence from Target-Specific Data Wiping Application Software. *Journal of Digital Forensics. Security and Law*, 7(2). p.113 -141.

CARVEY, H. (2005) The Windows Registry as a forensic resource.

Digital Investigation. 2 (3). p.201-5.

CARVERY, H. and ALTHEIDE, C. (2005) Tracking USB storage: Analysis of windows artifacts generated by USB storage devices. *Digital Investigation*. 2 (2). p. 94 - 100.

CARVEY, H. (2009) *Windows Forensic Analysis*. [Chapter 4] p.206- 213. USA: Elsevier.

CASEY, E. and SEGLEM, K. (2004). *Handbook of Computer Crime Investigation*. [Chapter 1] p. 2 – 3. London: Elsevier.

CASEY, K.(2014) Windows XP won't Go Quietly. *Information Week* 07/01/14 [Online]
Available from: <http://www.informationweek.com/software/operating-systems/windows-xp-wont-go-quietly/d/d-id/1113331>
[Accessed: January, 2014]

CASH, J (2008) ShellBags Registry Forensics, *SANS Computer Forensics*. [Online]
Available from: <http://blogs.sans.org/computer-forensics/2008/10/31/shellbags-registry-forensics/>
[Accessed: June 19, 2009].

CEEDO TECHNOLOGIES LTD (2010) *Ceedo Virtualisation Technology Overview*. London: Elsevier.
Available from: <http://www.ceedo.com>
[Accessed: December 5, 2011].

CEEDO PERSONAL. Available from: <http://www.ceedo.com/products/ceedo-personal.html>.
[Accessed: March 24, 2013].

COLLIE, J. (2013) The windows IconCache.db: A resource for forensic artifacts from USB connectable devices. *Digital Investigation* 9.(3-4), p200–210.

COLLIE, J. (2015) Tracing Forensic Artifacts from USB-Bound Computing Environments on Windows Hosts. Athens: ATINER'S Conference Paper Series, No: COM2015-1669.

COWEN, D. Hacking Exposed Computer Forensics Blog. Daily Blog 66: Understanding the artefacts setupapi.log/setupapi.dev.log. [Online]. Available from: <http://hackingexposedcomputerforensicsblog.blogspot.co.uk/2013/08/daily-blog-66-understanding-artefacts.html>. [Accessed: January 28, 2014].

CRESCENZO, G.D., FERGUSON, N, IMPAGLIAZZO, R, and JAKOBSSON, M. (1999). How to forget a secret. *Lecture Notes in Computer Science* 1563: p. 500–9.

DEC. WINDOWS SHELL ICON CACHE. [Online]. Available from: <http://www.thumbnailexpert.com/en/formats/windows-shell-icon-cache/> [Accessed: September 6, 2011].

DIRENZO, J. (2012), The Risk of Disruption or Destruction of Critical U.S. Infrastructure by an Offensive Cyber Attack.[Online]. Available from: <http://blog.havagan.com/wp-content/uploads/2012/05/The-Risk-of-Disruption-or-Destruction-of-Critical-U.S.-Infrastructure-by-an-Offensive-Cyber-Attack.pdf> [Accessed: July 23, 2012].

DWAN, B. (2004), Identity Theft. *Computer Fraud and Security*, 2004 (4), p. 14-17.

EDWARDS. J (2007) USB Thumb Drives: Tiny IT Security Terrors. *IT Security*. [Online]. Available from: <http://www.itsecurity.com/features/usb-thumb-drive-threat-102907> [Accessed: December 12, 2009].

FTK IMAGER from Access Data. [Online]. Available from: <http://accessdata.com/product-download> [Accessed: January 25, 2014].

GARRITY, S and WEIR, G.(2010) Balancing the threat of personal technology in the workplace. *International Journal of Electronic Security and Digital Forensics*. 3(1): p.73–81.

GOUCHER, W. (2008) The problem of the double-edged USB key. *Computer Fraud and Security*, 2008, (12), p. 10-12

HAYNOS, M. (2006) *Infrastructure Virtualisation: Extending SOA Throughout The Enterprise*. [Online] Available from:
<http://www.developer.com/design/article.php/3637006/Infrastructure-Virtualisation-Extending-SOA-Throughout-The-Enterprise.htm>
[Accessed: January 27, 2009].

HB GARY RESPONDER (Community Edition). Available from
<http://www.hbgary.com/free-tools>.
[Accessed: March 14, 2010].

HEIKKILA, F. M. (2007) Encryption: Security Considerations for Portable Media Devices. [Online] *IEEE Security & Privacy* 5 (1). p.22 – 27.

HOLDERNESS, J. (1999) Undocumented Windows 95, ‘The Shell Icon Cache’. [Online]. Available from: <http://reocities.com/SiliconValley/4942/iconcache.html>
[Accessed September 28, 2011].

JONES, A. (2008). Industrial espionage in a hi-tech world. *Computer Fraud and Security*, 2008, (1). p. 7-13.

KIM, G (2008) *Gene Kim’s Practical Steps to Mitigate Virtualisation Security Risks*. Tripwire Inc. [Online]
Available from: <http://www.tripwire.com>
[Accessed: January 8, 2010].

KINGSTON TECHNOLOGIES (undated) *Ultimate Memory Guide* [Online]

Available from: <http://www.kingston.com/tools/umg/umg01a.asp>
[Accessed: August 20, 2010].

LEE, CHAN-YOUN and LEE SANGJIN. (2014), Structure and application of IconCache.db files for digital forensics. *Digital Investigation*. 11 (2). p. 102 – 110.

LEE, R. (2009) *USB Key Analysis vs. USB Drive Enclosure*. SANS DFIR [Online]
Available from: <http://blogs.sans.org/computer-forensics/category/computer-forensics/usb-device-analysis>
[Accessed: January 12, 2010].

LUPO PENSUITE. [Online] . Available from: <http://www.lupopensuite.com>.
[Accessed: March 29, 2013].

MAARTMANN-MOE, C., THORKILDSEN, S, and ARNES, A. (2009). The persistence of memory: Forensic identification and extraction of cryptographic keys, *Digital Investigation* . 6. p.132-140.

MALIN, C., CASEY, E. and AQUILINA, J. (2012), *Malware Forensics Field Guide for Windows Systems*. Waltham, MA, USA: Elsevier Inc; [Chapter 2] p. 93-96.

MEE, V and JONES, A. (2005) *The Windows operating system registry - a central repository of evidence*. In: Proceedings of E-crime and Computer Evidence Conference, Monaco.

MEE, V., TRYFONAS, T, and SUTHERLAND, I, (2006), The Windows Registry as a forensic artifact: Illustrating evidence collection for Internet usage. *Digital Investigation* 3. p.166-173.

MICROSOFT. *Windows for your Business, Announcing Windows To Go Certified Drive Partners*. [Online]. Available from:
<http://blogs.windows.com/windows/b/business/archive/2013/02/01/announcing-windows-to-go-certified-drive-partners.aspx>
[Accessed: April 30, 2013.]

MICROSOFT SUPPORT. *How to determine the appropriate page file size for 64-bit versions of Windows*. [Online].

Available from: <https://support.microsoft.com/kb/889654>.

[Accessed: January 7, 2014]

MOKAFIVE. [Online]

Available from: <http://www.moka5.com/>

[Accessed: September 19, 2007].

MOJOPAC [Online]

Available from: <http://www.mojopac.com/>

[Accessed: December 13, 2009].

MOJOPAC. [Online]

Available from: http://download.cnet.com/Mojopac/3000-2064_4-10618349.html.

[Accessed: March 5, 2014].

NET APPLICATIONS

<http://www.netmarketshare.com>

[Accessed: January 9, 2014].

NIRSOFT,(u.d). MUICacheView v1.01

Available from: http://www.nirsoft.net/utils/muicache_view.html

[Accessed: September 28, 2011]

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (NIST)., (2004) *Guidelines on PDA Forensics*. Jansen, W. and Ayers, R. Technology Administration: U.S. Department of Commerce.

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (NIST)., (2004) *Guidelines on Cell Phone Forensics*. Jansen, W. and Ayers, R. Technology Administration: U.S. Department of Commerce.

PENHALLURICK, M. A, (2005) *Methodologies for the use of VMware to Boot Cloned/Mounted Subject Hard Disk Images*. MSc Thesis. Centre for Forensic Computing. Cranfield University.

PETRONI, N; WALTERS, A; FRASER, T; ARBAUGH, W. FATKIT. A, (2006) Framework for the extraction and analysis of digital forensic data from volatile system memory. *Digital Investigation* 3(4). p. 197–210.

PHAM, D. V., HALGAMUGE, M. N. and MENDIS P. (2010) *Optimizing Windows Security Features to Block Malware and Hack Tools on USB Storage Devices*, PIERS Proceedings, Cambridge, USA. (July). p. 5-8,

PORTABLE APPS. [Online]

Available from: <http://portableapps.com>

[Accessed: February 23, 2014].

PROCESS MONITOR. [Online]

Available from: <http://technet.microsoft.com/en-gb/sysinternals/bb896645.aspx>

[Accessed: January 9, 2014].

RAMCAPTURER from Belkasoft.

Available from: <http://www.belkasoft.com>

[Accessed: April 18, 2011].

ROY TANUSHREE and JAIN ARUNA (2012). Windows registry forensics: an imperative step in tracking data theft via USB devices. *International Journal of Computer Science and Information Technologies (IJCSIT)*. 3(3). P. 4427–33.

RUFF, N. (2008). Windows memory forensics. *Journal in Computer Virology*. 4. p. 83-100.

RUSSINOVICH M, SOLOMON D. (2005). *Microsoft windows internals. Microsoft windows server 2003, windows XP and windows 2000*. 4th ed. Washington: Microsoft Press. [Chapter 6], p. 311–313.

RUSSINOVICH M, SOLOMON D, IONESCU, A. (2012) *Windows Internals Part 2*. 6th ed, Washington: Microsoft Press. [Chapter 10], p. 187–190.

SAMMES, T (2007), *VMware Forensics*. F3 Workshop delivered at Runcorn, June 12.

SEARCHSERVERVIRTUALISATION.COM (Undated), *Virtualisation*. Rouse, M.

Available from:

http://searchservvirtualisation.techtarget.com/sDefinition/0,,sid94_gci499539,00.htm

1

[Accessed: January 16, 2010].

SCHUSTER, A. (2006). Searching for processes and threads in Microsoft Windows memory dumps. The Proceedings of the 6th Annual Digital Forensic Research Workshop. 3, pp. 10-16. *Digital Investigation*.

SHARMA, V. (2011). An Analytical Survey of Recent Worm Attacks, *IJCSNS International Journal of Computer Science and Network Security*, 11 (11), p.99-103.

SHAVERS, B (2008) *Virtual Forensics – A Discussion of Virtual Machines Related to Forensics Analysis*.

Available from: <http://ebookbrowse.net/a-discussion-of-virtual-machines-related-to-forensics-analysis-by-brett-shavers-pdf-d297508581>

[Accessed: February 23, 2010].

SOLOMON, J; E. HUEBNER, E; BEM, D; SZEZYNSKA, M. (2007). User data persistence in physical memory, *Digital Investigation* 4(2). p. 68-72.

SOMMER, P, (2009) *Directors' and Corporate Advisors' Guide to Digital Investigations and Evidence v. 2.1*. Information Assurance Advisory Council.

STATCOUNTER [Online].

Available from: <http://statcounter.com>

[Accessed: November 23, 2012 & January 12, 2014].

STEVENS, D, (undated) UserAssist [online]

Available from: <http://blog.didierstevens.com/programs/userassist/>

[Accessed: December 18, 2010].

TANK, R and WILLIAMS, P (2008), *The impact of U3 devices on forensic analysis*, in Proceedings of the 6th Australian Digital Forensics Conference. Edith Cowan University. [Online].

Available from: <http://ro.ecu.edu.au/cgi/viewcontent.cgi?article=1056&context=adf>
[Accessed: August 17, 2014].

TAYLOR, Kelvyn, (2008) Turn your iPod into a PC. *Personal Computer World*. (January), P.44 - 48.

TETMEYER, A. and SAIEDIAN, H. (2010). Security threats and mitigating risk for USB devices. *Technology and Society Magazine*, IEEE 29(4). p. 44–9.

THOMAS, P. and MORRIS, A. (2008) *An investigation into the development of an anti-forensic tool to obscure USB flash drive device information on a windows XP platform*. ADFIA'08. Third International Annual Workshop on Digital Forensics and Incident Analysis. p. 60 - 66.

UNITED STATES SECRET SERVICE (2009) *Best Practice for Seizing Electronic Evidence V.3*. U.S. Department of Homeland Security.

W3SCHOOLS. *Web statistics and trends, OS platform statistics*. [Online]

Available from:

<http://www.w3schools.com>

[Accessed: January 12, 2014].

BIBLIOGRAPHY

ACCESS DATA (2006) *Windows Forensics Training Manual*, Utah, USA: Access Data Corporation.

BECKER, J.C. and PARK A., (1991) Analysis of the Paging Behavior of UNIX, *Performance Evaluation Review*. 19 (2). p.. 36- 43

BK FORENSICS (2009) *First Responder Mobile Phone Seizure Flowchart* [Online]. Available from: <http://www.BKForensics.com>
[Accessed: May 23, 2009].

CHANCE, R., (2005) *Understanding USB Flash Drives As Portable Infrastructure*. [Whitepaper]. Browsercraft LLC.

CITRIX SYSTEMS INC. (2009). *Desktop virtualisation: A revolution in desktop computing for everyone*. [Online]
Available from: <http://www.citrix.com>
[Accessed: December 12, 2009].

DATA PROTECTION ACT, 1998, [Online]
Available from: http://www.opsi.gov.uk/Acts/Acts1998/ukpga_19980029_en_1
[Accessed: April 5, 2008]

DANKER, S. AYERS, R. AND MISLAN, R. P., (2009) Hashing Techniques for Mobile Device Forensics. *Small Scale Digital Device Forensics Journal*, 1(3), p. 1-6.

DENNING, PETER J., (1970) Virtual Memory. *ACM Computing Surveys*, 2 (3). p. 153 – 189.

HAK5 – USB HACKSAW. [Online]
Available from: <http://www.hak5.org/usb-hacksaw>
[Accessed January 16, 2010].

HANDSCHUH, HELENA (2007) *Securing Flash Technology*, Spansion Internet. Inc. Workshop on Fault Diagnosis and Tolerance in Cryptography. p. 3 – 17.

HARRINGTON, M., (Undated). *Understanding SMS: Practitioner's Basics*. [Article]
Available from:
https://mobileforensics.files.wordpress.com/2007/06/understanding_sms.pdf
[Accessed: May 6, 2010].

HAY, A.S. (2005) Windows File Analyser Guidance. [Online]
Available from: <http://www.mitec.cz/Downloads/WFA%20Guidance.pdf>
[Accessed: March 11, 2010].

HM TREASURY (2008) *Fraud Report 2007 – 08: An Analysis of reported fraud in government departments*. [Online]
Available from: http://www.hm-treasury.gov.uk/d/govt_fraudreport031008.pdf
(Accessed: March 30, 2009).

HOLDER, E., ROBINSON, L. and ROSE, K. (2009) *Crime Scene Investigation: An On-the-Scene Reference for First Responders*. National Institute of Justice, U.S. Department of Justice.

INOUE, A AND WONG, D (2004), *NAND Flash Applications Design Guide*. Revision 2.0. Toshiba America Electronic Components Inc.
Available from: <http://wenku.baidu.com/view/3963b33043323968011c9293.html>
[Accessed: April 11, 2010]

KAVALLARIS, T. and KATOS, V. (2010) On the detection of pod slurping attacks. [Online]. *Computers and Security*, 29 (6) p 680 – 685.
Available from: <http://dx.doi.org/10.1016/j.cose.2010.01.002>
[Accessed: May 30, 2012.]

KEITZMAN, S. (2010) *What is a Virtual machine?* wiseGEEK [Online]
Available from: <http://www.wisegeek.com/what-is-a-virtual-machine.htm>
[Accessed; January 8, 2010].

KOH WEH, (2004) Memory Device Packaging – *From Leadframe Packages to Wafer Level Packages*. IEEE. Proceedings of HDP'04. p. 21 – 24.

KORNBLUM, J. (2007). Using every part of the buffalo in windows memory analysis. *Digital Investigation*, 4(1), p. 24-29

MARCELLA, A. and MENENDEZ, D. (2008) *Cyberforensics. A Field Manual for Collecting, Examining and Preserving Evidence of Computer Crimes*, 2nd Ed., New York, USA: Auerbach Publications.

MRDOVIC, S, and HUSEINOVIC, A. (2011). *Forensic Analysis of Encrypted Volumes Using Hibernation File*. Faculty of Electrical Engineering, Sarajevo.
Available from:
http://people.etf.unsa.ba/~smrdovic/publications/Telfor2011_Mrdovic_Huseinovic.pdf
[Accessed: January, 2014].

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (NIST) (2004), *Computer Security Incident Handling Guide; Guide to Integrating Forensic Techniques into Incident Response*. Abridged by Guidance Software Inc., (2007)

PRICEWATERHOUSECOOPERS (2006) *DTI Information Security Breaches Survey 2006*.

REGAN, J., (2009) *The Forensic Potential of Flash Memory*. MSc Thesis. Naval Postgraduate School, Monterey, California.
Available from: www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA509258
[Accessed: March 23, 2010]

SANS INSTITUTE (2009) *USB Key-Guide* [Online]
Available from: <http://computer-forensics.sans.org/>
[Accessed: July 24, 2009]

SHELDON, B., (2002) Forensic Analysis of Windows Systems. In: E. Casey, ed. 2004 *Handbook of Computer Crime Investigation*. San Diego, USA: Elsevier Academic Press. [Ch. 7].

SMITH, J AND NAIR, R (2005) *Virtual Machines: Versatile Platforms for Systems and Processes*. USA: Morgan Kaufmann.

STIMSON, J.M, *Forensic Analysis of Windows Virtual Memory Incorporating the System's pagefile*. (2008). Master's Thesis. Postgraduate School, California, USA.

STORR, W, (2010) The MOSFET. Scientific Working Group on Digital Evidence (SWGDE), *Best Practices for Mobile Phone Examinations. V.1.0*.

Available from: http://www.electronics-tutorials.ws/transistor/tran_6.html

[Accessed: February 12, 2010].

SUICHE, M, (2008) *Windows hibernation file for fun 'n' profit*, Black Hat, USA.

Available from: [http://www.blackhat.com/presentations/bh-usa-](http://www.blackhat.com/presentations/bh-usa-08/Suiche/BH_US_08_Suiche_Windows_hibernation.pdf)

[08/Suiche/BH_US_08_Suiche_Windows_hibernation.pdf](http://www.blackhat.com/presentations/bh-usa-08/Suiche/BH_US_08_Suiche_Windows_hibernation.pdf)

[Accessed: January 23, 2014].

TARDIEU, SAMUEL., (2006) *Wiping unused space in a file system* [Online]

Available at: <http://bit.ly/cQ4AVR>

[Accessed: January 16, 2009].

TREND MICRO (2010), *Security Spotlight Understanding USB Malware*

Available from: [https://imperia.trendmicro-](https://imperia.trendmicro-europe.com/imperia/md/content/us/trendwatch/researchandanalysis/)

[europe.com/imperia/md/content/us/trendwatch/researchandanalysis/](https://imperia.trendmicro-europe.com/imperia/md/content/us/trendwatch/researchandanalysis/)

[56_understanding_usb_malware__april_19__2010_.pdf](https://imperia.trendmicro-europe.com/imperia/md/content/us/trendwatch/researchandanalysis/56_understanding_usb_malware__april_19__2010_.pdf)

[Accessed: 25th July 25, 2012]

TURLE M. et al, (2011), *Data Protection: Laws of the World*, London, UK: Sweet & Maxwell.

WELLS, J.T. (2004), *Corporate Fraud Handbook. Prevention and Detection*. New Jersey, USA: John Wiley & Sons.

APPENDICES

Table of Contents

APPENDICES	1
APPENDIX I	4
Privacy Claims – Miniature Virtualization	4
APPENDIX I a):	5
vPC Promotional literature: MojoPac	5
APPENDIX 1 b) :	6
vPC Promotional literature: Ceedo Personal	6
APPENDIX II.....	7
USB Artifacts in Windows – Investigation Guides	7
APPENDIX II a):.....	8
SANS Institute Guide.....	8
APPENDIX II b).....	12
Arsenal Recon Guide	12
APPENDIX III	14
vPC environment presentation on Windows host desktop.....	14
APPENDIX III a:	15
MojoPac Environment run from USB drive on Windows XP	15
APPENDIX III b):.....	16
Ceedo Personal Environment run from USB drive on Windows XP	16
APPENDIX III c):	17
Lupo PenSuite Environment run from USB drive on Windows XP	17
APPENDIX III d):.....	18
Portable Apps Environment run from USB drive on Windows XP.....	18
APPENDIX IV.....	19
Published Paper: the Windows IconCache.db	19
APPENDIX V	40
IconCache.db research work cited and extended.....	40
APPENDIX VI.....	41
Form for identifying vPCs and Windows OSs and monitoring experiments carried out.....	41
APPENDIX VII.....	43
Form devised for collecting experimental results	43
APPENDIX VIII.....	46
‘Quick reference’ chart for vPC artefacts	46
in Windows XP and 7 locations.....	46
APPENDIX IX.....	48
IconCache.db sample file size changes during experiments.....	48
APPENDIX X	51
Process Monitor sample readings for MojoPac	51
Appendix X a)	52
Sample Outputs Baseline Test a) MojoPac.....	52
Appendix X b).....	65
Sample Outputs Baseline Test b) MojoPac.....	65
Appendix X c)	69
Sample findings on Host HD - Baseline Test b) MojoPac.....	69
Appendix X d).....	73
Findings in Host Registry - Baseline Test b) MojoPac	73
Appendix X e)	74
Sample findings in Host RAM - Baseline Tests a) and b) MojoPac	74
Appendix X f).....	77
Findings in Host Pagefile - Baseline Tests b) MojoPac.....	77
APPENDIX XI.....	79

Baseline test results for Ceedo Portable.....	79
Appendix XI a)	80
Sample PM Outputs Baseline Test a) Ceedo	80
Appendix X b).....	81
Sample findings in RAM - Baseline Test a) Ceedo	81
Appendix X c)	83
Sample PM outputs - Baseline Test b) Ceedo	83
Appendix X d).....	85
Sample findings in RAM - Baseline Test b) Ceedo.....	85
Appendix X e).....	86
Sample findings in Host Registry - Baseline Test b) Ceedo	87
Appendix X f).....	88
Sample findings on Host HD - Baseline Test b) Ceedo.....	88
APPENDIX XII	93
Sample Baseline test results for	93
LupoPenSuite & PortableApps	93
Appendix XII a).....	94
Sample PM Outputs Baseline Test b) LupoPenSuite.....	94
Appendix XII b).....	95
Sample PM Outputs Baseline Test b) PortableApps	95
Appendix XII c)	98
Programs associated with PortableApps	98

APPENDIX I
Privacy Claims – Miniature Virtualization

APPENDIX I a): vPC Promotional literature: MojoPac



This promotional slide features a central image of a computer monitor displaying a Windows XP desktop environment. The desktop includes icons for 'My Computer', 'My Recent Places', and 'My Places'. Several application windows are open, including a web browser showing an Apple website, a Microsoft Word document, and a 'Firefox Start' dialog box. A black MojoPac USB drive is positioned in the foreground on the left. An orange callout box on the right contains the text: 'Your MojoPac is completely private and secure. Your applications, your browsing history, and your data is never left behind on the PC it is connected to.' Below this, another orange box states: 'What happens on MojoPac, stays on MojoPac'. At the bottom of the slide, there is a navigation bar with four buttons: 'about mojpac', 'community', 'download', and 'register now'. A small MojoPac logo is on the left of the bar. Below the navigation bar, the text 'Copyright 2008 - RingCube Technologies, Inc.' is displayed.



This promotional slide features a central image of a black MojoPac USB drive on the left and a large orange square with a white stylized 'M' logo on the right. An orange callout box on the right contains the text: 'MojoPac is portable, private and personal and works on any Windows XP computer.' Below this, another orange box states: 'It will not affect the PC it is connected to, and the PC you connect to will not affect MojoPac.' At the bottom of the slide, there is a navigation bar with four buttons: 'about mojpac', 'community', 'download', and 'register now'. A small MojoPac logo is on the left of the bar. Below the navigation bar, the text 'Copyright 2008 - RingCube Technologies, Inc.' is displayed.

APPENDIX 1 b) : vPC Promotional literature: Ceedo Personal


Buy Software | Support | Contact | Login

HOME PRODUCTS SOLUTIONS PARTNERS COMPANY

▶ Ceedo Personal

Take your applications with you in a plug-'n'-play Workspace on any USB storage device

Ceedo Personal, which costs only \$39.95, enables individuals to install all of their favorite applications into a plug-'n'-play Workspace that can be mounted on any USB storage device.

Highly-praised and used by millions of users worldwide, Ceedo Personal is the ultimate portable desktop solution, allowing you to use your applications on any PC in a plug-'n'-play fashion, use a complete integrated Workspace, and run on any PC, regardless of Windows versions, and with no need for administrator rights.



■ Invoke Your Desktop Anywhere

Zero-Install Plug-'n'-Play Environment With All Your Applications Inside

With Ceedo Personal, you can take your computing experience with you on-the-go. Whether you're across the street or across the globe, Ceedo Personal gives you access to the same familiar applications and content that you're used to using. It's as if you never left home.

■ Vast Application Repository Online

A large verity of applications are just waiting to be added

Apart from installing your own applications, Ceedo Personal also allows you to enjoy the Ceedo Programs Directory, which lets you add applications on-the-fly with just a few mouse clicks. Whether you are eight or eighty, you are bound to find programs that suit your taste in the Ceedo Programs Directory.

■ User-Friendly Launch Panel

Anywhere at any time, and by anyone!

Free Trial

Watch Video

Buy Now

Data Sheet

Presentation

Watch on youku

Windows8downloads
EDITOR'S PICK

- **Plug-'n'-Play Experience**
 Your data, settings and application stored in a zero-install plug-'n'-play workspace that works anytime, anywhere.
- **No Need for Admin Rights**
 Ceedo Personal lets you run applications even on machines with no administrator rights like internet cafes and public PCs. Just plug you USB device and work as if it was you home PC.
- **Works on Any PC**
 Ceedo Personal supports Windows™ XP, Vista and 7, 8 & 8.1.
- **Leaves Nothing Behind**
 Ceedo Personal has zero-footprint, meaning once you plug-out your USB nothing from Ceedo Personal is left behind on the host PC.

APPENDIX II
USB Artifacts in Windows – Investigation Guides

**APPENDIX II a):
SANS Institute Guide**



Profile Windows XP USB Keys/Thumbdrives



XP USB KEY/Thumbdrive	
1. Write Down Vendor, Product, Version	
SYSTEM\CurrentControlSet\Enum\USBSTOR	Vendor = Product = Version =
2. Write Down Serial Numbers	
SYSTEM\CurrentControlSet\Enum\USBSTOR	Serial Number =
3. Determine Parent Prefix ID	
SYSTEM\CurrentControlSet\Enum\USBSTOR	Parent Prefix ID=
4. Determine Vendor-ID (VID) and Product-(PID)	
SYSTEM\CurrentControlSet\Enum\USB -> Perform search for S/N	VID_XXXX = PID_YYYY =
5. Determine Drive Letter Device Mapped To	
SYSTEM\MountedDevices-> Perform search for Parent Prefix ID in the Drive Letter	Drive =
6. Write Down Volume GUIDs	
SYSTEM\MountedDevices-> Perform Search for Parent Prefix ID in the GUIDs	{GUID} =
7. Find User That Used The Specific USB Device	
NTUSER.DAT\Software\Microsoft\Windows\ CurrentVersion\Explorer\MountPoints2-> Search for Device GUID	User =
8. Discover First Time Device Connected	
C:\Windows\setupapi.log -> Perform search for Serial Number	Time/Timezone =
9. Determine First Time Device Connected After Last Reboot	
SYSTEM\CurrentControlSet\Control\Devic eClasses\{53f56307-b6bf-11d0-94f2- 00a0c91efb8b}-> Perform search for S/N or SYSTEM\CurrentControlSet\Enum\USB\ VID_XXXX&PID_YYYY -> Perform search for Serial Number (Last Written Time of Serial Number Key)	Time/Timezone =
10. Determine Last Time Device Connected	
NTUSER//Software/Microsoft/Windows/Cur rentVersion/Explorer/MountPoints2/{GUI D} -> Perform search for Device {GUID}	Time/Timezone =

<http://forensics.sans.org>
<http://twitter.com/sansforensics>





Profile VISTA USB Key/Thumbdrives



VISTA USB KEY/Thumbdrive	
1. Write Down Vendor, Product, Version	
SYSTEM\CurrentControlSet\Enum\USBSTOR	Vendor = Product = Version =
2. Write Down Serial Numbers	
SYSTEM\CurrentControlSet\Enum\USBSTOR	Serial Number =
3. Determine Vendor-ID (VID) and Product-(PID)	
SYSTEM\CurrentControlSet\Enum\USB -> Perform search for S/N	VID_XXXX = PID_YYYY =
4. Write Down Volume GUIDs	
SYSTEM\MountedDevices-> Perform Search for Serial Number	GUID =
5. Determine Drive Letter and Volume Name Device Mapped To	
SOFTWARE\Microsoft\Windows Portable Devices\Devices-> Perform Search for Serial Number and Match with Volume Name	Drive Letter = Volume Name=
6. Find User That Used The Specific USB Device	
NTUSER.DAT\Software\Microsoft\Windows\C urrentVersion\Explorer\MountPoints2-> Search for Device GUID	User =
7. Discover First Time Device Connected	
C:\Windows\inf\setupapi.dev.log -> Perform search for Serial Number	Time/Timezone =
8. Determine First Time Device Connected After Last Reboot	
SYSTEM\CurrentControlSet\Enum\USBSTOR\ Vendor_Product_Version -> Perform search for Serial Number (Last Written Time of Serial Number Key) or SYSTEM\CurrentControlSet\Control\Device Classes\{53f56307-b6bf-11d0-94f2- 00a0c91efb8b}-> Perform search for S/N (Last Written Time of Key that has Serial Number and Vendor/Product/Revision)	Time/Timezone =
9. Determine Last Time Device Connected	
SYSTEM\CurrentControlSet\Enum\USB\ VID_XXXX&PID_YYYY -> Perform search for Serial Number (Last Written Time of Serial Number Key) or NTUSER//Software/Microsoft/Windows/Curr entVersion/Explorer/MountPoints2/{GUID} -> Perform search for Device {GUID}	Time/Timezone =

<http://forensics.sans.org>
<http://twitter.com/sansforensics>





Profile Windows 7 USB Keys/Thumbdrives

Win7 USB Key/Thumbdrive	
1. Write Down Vendor, Product, Version	
SYSTEM\CurrentControlSet\Enum\USBSTOR	Vendor = Product = Version =
2. Write Down Serial Numbers	
SYSTEM\CurrentControlSet\Enum\USBSTOR	Serial Number =
3. Determine Vendor-ID (VID) and Product-(PID)	
SYSTEM\CurrentControlSet\Enum\USB -> Perform search for S/N	VID_XXXX = PID_YYYY =
4. Determine Drive Letter Device Mapped To	
SYSTEM\MountedDevices-> Perform search for Serial Number in the Drive Letters	Drive =
5. Write Down Volume GUIDs	
SYSTEM\MountedDevices-> Perform Search for Serial Number in the GUIDs	GUID =
6. Find User That Used The Specific USB Device	
NTUSER.DAT\Software\Microsoft\Windows\C urrentVersion\Explorer\MountPoints2-> Search for Device GUID	User =
7. Discover First Time Device Connected	
C:\Windows\inf\setupapi.dev.log -> Perform search for Serial Number	Time/Timezone =
8. Determine First Time Device Connected After Last Reboot	
SYSTEM\CurrentControlSet\Enum\USBSTOR\ Vendor_Product_Version -> Perform search for Serial Number (Last Written Time of Serial Number Key) or SYSTEM\CurrentControlSet\Control\Device Classes\{53f56307-b6bf-11d0-94f2- 00a0c91efb8b}-> Perform search for S/N (Last Written Time of Key that has Serial Number and Vendor/Product/Revision)	Time/Timezone =
9. Determine Last Time Device Connected	
SYSTEM\CurrentControlSet\Enum\USB\ VID_XXXX&PID_YYYY -> Perform search for Serial Number (Last Written Time of Serial Number Key) or NTUSER//Software/Microsoft/Windows/Curr entVersion/Explorer/MountPoints2/{GUID} -> Perform search for Device {GUID}	Time/Timezone =

<http://forensics.sans.org>
<http://twitter.com/sansforensics>



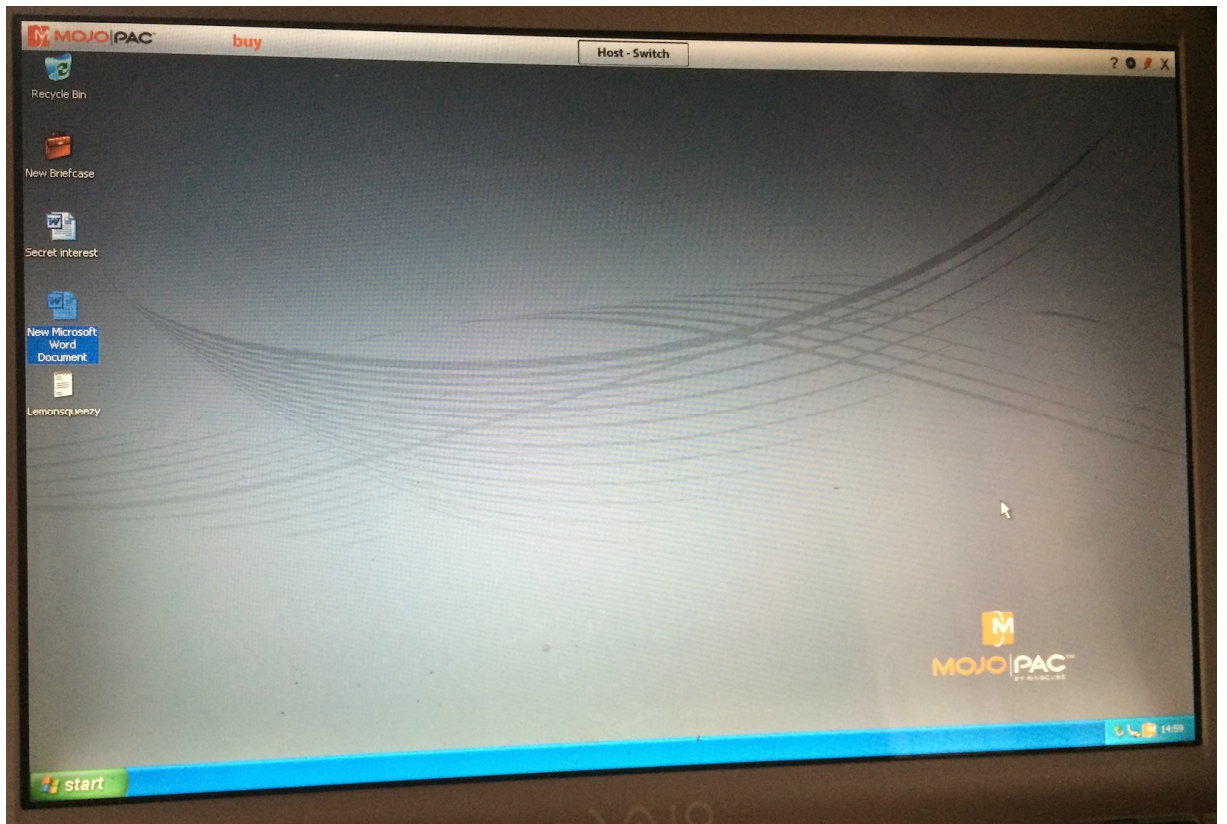
APPENDIX II b)
Arsenal Recon Guide

APPENDIX III

vPC environment presentation on Windows host desktop

APPENDIX III a:

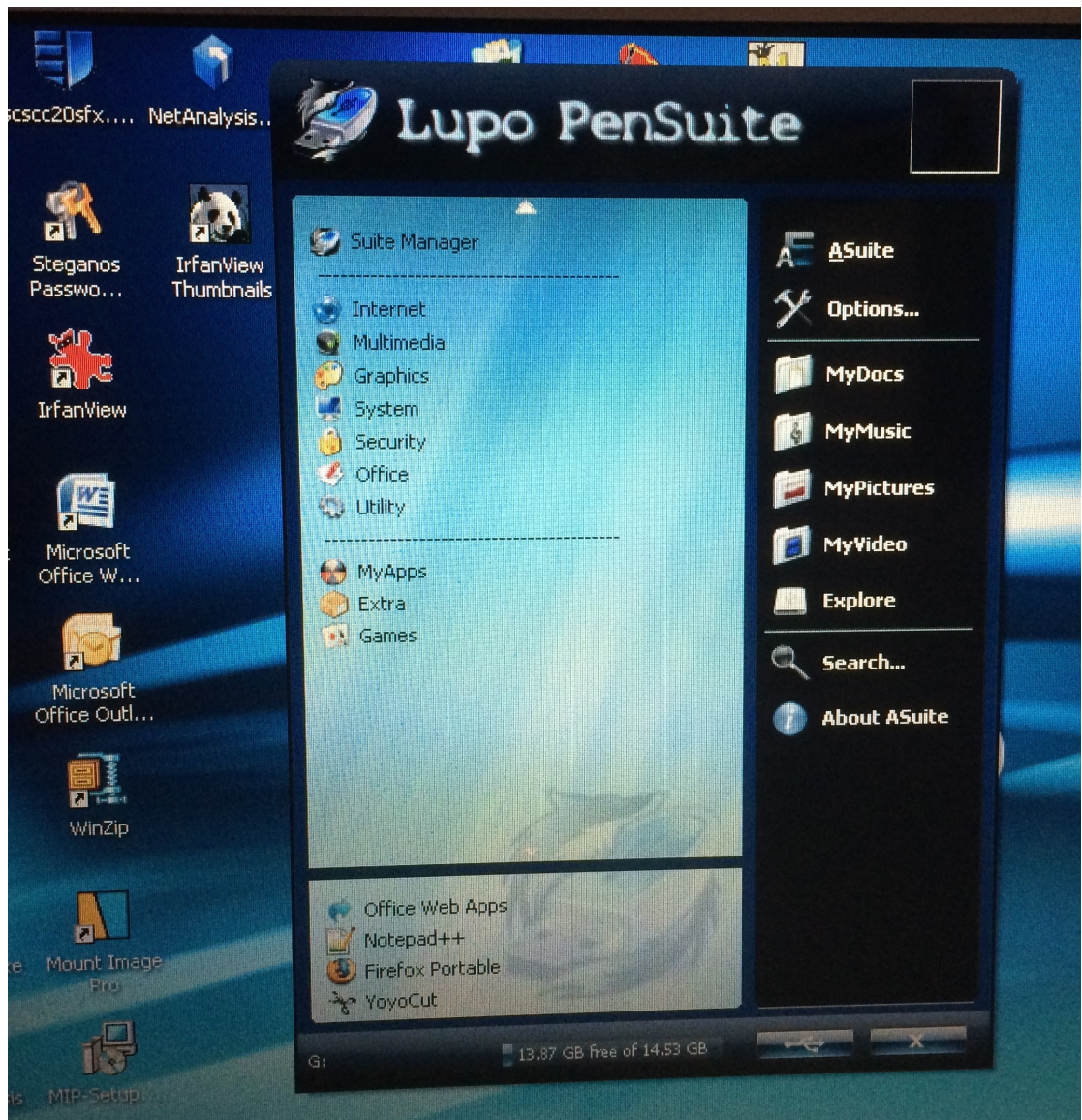
MojoPac Environment run from USB drive on Windows XP



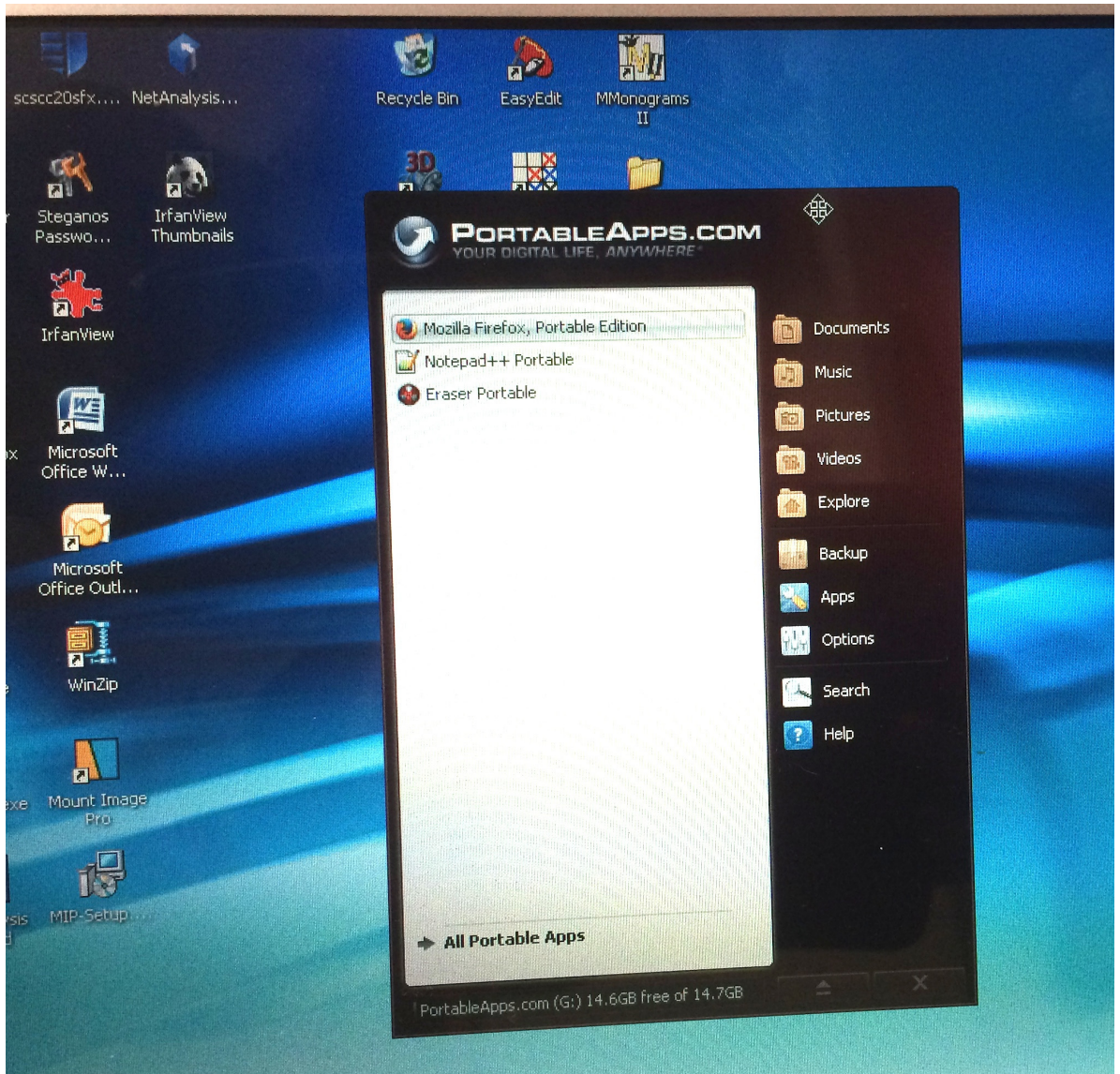
**APPENDIX III b):
Ceedo Personal Environment run from USB drive on Windows XP**



**APPENDIX III c):
Lupo PenSuite Environment run from USB drive on Windows XP**



**APPENDIX III d):
Portable Apps Environment run from USB drive on Windows XP**



APPENDIX IV
Published Paper: the Windows IconCache.db

The Windows IconCache.db: a resource for Forensic Artifacts from USB Connectable Devices

Author: Jan Collie

Address: Discovery Forensics Ltd, 23 Austin Friars, London EC2N 2QP.

Abstract - **This paper investigates the evidential potential of the IconCache database file when tracking activity from USB connectable devices on Windows systems. It focuses on the artifacts which are created and retained on a Windows host when executable files are either present on or run from a USB connectable device. Artifacts left in the IconCache database as a result of running executables from a DVD drive or the host itself, are also examined.**

It is shown that the IconCache.db stores numerous artifacts of investigative interest. These are created on system boot and added to, both when using host-based executables and when installing or using executables from other media. Executables present on USB devices, whether invoked or not, will create artifacts in the IconCache.db. file. Findings should therefore be interpreted carefully and corroborated against other evidence.

1. Introduction

The uncontrolled connection of USB devices to computer systems is widely acknowledged as a serious security threat (DiRenzo, 2012; Garrity and Weir, 2010; Pham et. al., 2010; Tetmeyer, 2010). USB connectable memory sticks - also known as thumb drives or keys - present a particular risk, both to sensitive data and the systems used to serve and store them. Small, cheap and easily available, they open up opportunities for both the theft and casual loss of valuable information. Viruses and other malicious software can also be introduced to stand-alone or networked computer systems via USB, whether deliberately or inadvertently (DiRenzo, 2012; Sharma, 2011). From the perspective of digital forensic analysis, there is a danger that anti-forensic programs can be run direct from a USB key, helping wrongdoers to cover up or completely wipe out any traces of their activities (Thomas and Morris, 2008).

For all these reasons, forensic analysis of artifacts left behind by the use of USB connectable devices has become an integral part of computer investigations in corporate and criminal cases. Prior research in this area has concentrated on the Windows Registry, including Carvey (2005 & 2009), Carvey and Altheide (2005), Mee, and Jones (2005) and Mee, et al (2006). Carvey (2005) has shown that the Windows system may also store traces of USB connection activity in Link files, Shortcuts and the PreFetch folder. Such artifacts can help a forensic analyst trace file-related activity, such as opening a document on or copying a picture to a particular device (Roy and Jain, 2012). Other artifacts are created in the Windows Registry when programs are executed. The UserAssist key, for example, stores information about recently used and frequently used programs and the number of times these programs have been launched (Stevens, D. 2010). If a program is run from a USB connectable device, therefore, it is possible that a record of that action will be kept by the system in this Registry key. Using a new application on a Windows system will also populate the MUICache Registry key (NirSoft, 2011), so this key may also contain useful information.

This paper seeks to extend the field of enquiry beyond the Registry by examining the IconCache database file, a hidden file which is created and maintained on computers running Windows operating systems, in terms of its evidential potential. Caching mechanisms, such as the IconCache database, have been discussed by systems specialists in terms of their role in Windows memory management (Russovich and Solomon, 2005). The investigative potential of such mechanisms has yet to be explored. The aim of this work is to help digital forensic analysts to identify and/or corroborate computer user activity, in particular the use of portable USB devices or DVDs.

For the purposes of this research, samples of the three Microsoft Windows operating systems are examined: Windows XP, Windows Vista and Windows 7. These have been chosen because, at the time of writing, the Windows XP and Windows 7 operating systems account for approximately 80% of all operating systems in use, Windows Vista being the next most popular (Statcounter, 2012; w3schools, 2012; Marketshare, 2012). It is likely, therefore, that computer examiners will commonly be analyzing these operating systems. It is assumed that home directories are present on the systems under analysis and have not been moved elsewhere e.g. to a server.

The paper chiefly examines the artifacts produced and retained in the IconCache database file when a USB device containing an executable file is connected to a Windows host. The effects of introducing and running executables from a DVD drive and from the host itself are also considered. An overview of how the IconCache database comes into being and how it accumulates artifacts of potential forensic interest is given, however a full explanation of the workings of the icon caching mechanism in Windows goes beyond this paper's scope. While some interrelated files are referred to, the primary intention is to look at what is stored in the database as a result of user activity and how these artifacts could help inform digital forensic analysis.

2. Illustrative Case Example

A recent investigation performed by the author demonstrates the forensic value of information in the IconCache database. Two rival companies were pitching for a multi-million pound contract to buy and convert a large public building. The process, which was being carried out via closed bids, was overseen by an independent, third company, on behalf of a local council. Competition between the two bidders was intense. As it increased, it was found that an employee of one of the rival companies was in a relationship with an employee of the independent overseer. An urgent investigation was ordered to discover whether any impropriety had taken place. Although documents pertaining to the bidding war were highly sensitive, encryption was not being used by any of the parties. A review of the computer used by one of the employees under suspicion showed that at least three different USB memory sticks had been connected to the machine during the time period of interest, however, no artifacts to suggest the use of any potentially problematic external program were found in the Windows Registry or elsewhere. By comparison, an examination of the IconCache.db showed that an encryption program had been present on one USB drive introduced to the system. This finding sparked a further enquiry involving a search for encrypted files on a number of other computers. At the end of the day, the use of encrypted files to pass sensitive information could not be ruled out. The bidding war was halted and restarted under stricter controls.

3. Synopsis of Findings

On Windows systems, programs, files and folders are represented by icons - small symbolic pictures which provide a visual clue to what may be accessed by clicking on them. However,

the desktop presented to the computer user would be impossibly cluttered if the system displayed every icon that it could. Having to retrieve all possible icon images from disk and render them, time and again, would also consume system resources unnecessarily. As a result, Windows systems save icons already retrieved, such as from programs which run automatically at startup, in a cache in memory (Holdeness, J.,1999).

Since its introduction in Windows XP, the IconCache.db file has provided a mechanism for storing the icons associated with processes and programs that either run in the background or are otherwise initiated on a Windows-based system. This paper demonstrates that IconCache.db files can contain artifacts of a wide range of user-instigated activities which may be relevant to a digital investigation. These include activities carried out on the host system via a removable USB device which could otherwise be hard to trace or may not be traceable at all.

The IconCache.db is a hidden file stored in different locations, depending on the version of the operating system (OS).

- **Windows XP:** C:\Documents and Settings*Username*\Local Settings\Application Data\IconCache.db
- **Windows Vista or Windows 7:** C:\Users*Username*\AppData\Local\IconCache.db

Because a separate IconCache database is maintained for each named user of a computer, artifacts in this file can be attributed to a specific user account.

If the IconCache.db file does not already exist on a Windows system, it is automatically created at system startup and populated with a number of default system icons. The cache of icons exists in memory but will also be written to disk following a Windows shutdown or restart. In this way, the file persists from one session to another (Holdeness, J.,1999). The IconCache.db grows as information is added to it by the processes and activities which occur on the machine. The changes which take place in the IconCache.db include both the addition of new visible icons and the addition of file paths to the programs or processes associated with those icons in the text portions of the data base.

Of particular interest is the information which appears in the IconCache.db when a removable USB device or CD/DVD is connected to a host system and its contents are viewed. If the removable media contains one or more executables at the root of the drive, then any associated icons are added to the database. The file path, including the drive letter associated with the removable concerned, is also retained in the database in the following example format:

E:\Program_Name.exe

It should be noted that this action occurs whether any such executable is run or not, which means that readings obtained from the IconCache.db should be interpreted with care, ideally being corroborated against other findings. Nevertheless, the database can be an useful aid to the digital analyst, especially where attempts have been made to conceal suspect activities, such as the use of an external encryption program or the introduction of malware via some USB device or optical media. Where these types of programs have been used on a computer but never installed, the IconCache.db may retain the only record of that activity.

The IconCache.db can also help digital examiners in cases where a potentially suspect program e.g. Limewire has been installed on a system via removable media and later uninstalled. While few traces of that activity may remain on the host, a record, complete with textual details of both the origin of the installation and the path to its location on the host, will be kept by the database. In an experiment carried out during this research, the Limewire installation package

was downloaded to the root of a USB stick and executed on the subject system, resulting in the following artifacts:

1. Placing the stick into a Windows host and viewing its contents caused two Limewire icons to appear in the IconCache.db. The textual portion of the file showed that the USB stick had been allocated the drive letter F: and that the Limewire executable was on it, as follows:

F:\limewire_setup.exe

2. When the program was installed and run on the host, that path remained in the IconCache database and further information was added. This showed the temporary files and processes needed for the installation being run under a particular user name, as simplified below:

F:\limewire_setup.exe=c:\docume~1\USER \locals~1\temp

The path to the location of the installed program on the C: drive of the computer followed:

\temp\is-trm8s.tmp\limewire_setup.tmp%c:\program files\lime pro\limepro.exe

3. When the program was run, more text was appended to the above information in the IconCache.db:

%c:\program files\lime pro\limepro.exe

4. When the program was uninstalled, further text was appended:

&c:\program files\lime pro\unins000.exe

While this research has not extended beyond this kind of initial result and the full meaning of the data obtained requires further analysis e.g. what, if anything, is indicated by the readings %c:\ and &c:\ in the above examples, the findings could still usefully inform a computer analysis. On that basis, experiments were carried out to ascertain what happens in the IconCache.db as a result of a number of common activities, including:

- opening host files and folders
- placing a link to a host-based program on the desktop
- running a pre-installed program e.g. Notepad on the host
- loading a DVD containing an executable file
- inserting a USB drive containing one or more executables
- running an executable from a USB drive
- opening a file from a USB drive
- writing a file out to a USB drive
- creating a new user account on a system

These effects have been investigated systematically by following defined sequence of actions from clean installs of three different Windows operating systems.

3.1 *File signature*

In some cases, it may be necessary to salvage deleted IconCache.db files by carving for the file signature. The file signature for the IconCache.db appears to be consistent across Windows XP, Windows Vista and Windows 7 at offset 4 for four bytes, where the hexadecimal values: 57 69 6E 34, read as text: Win4, are recorded (Table 1). However, variances in the IconCache.db are found to occur at offset 8 and at offset 12. Examples are given in Table 2.

Table 1: Basic file signature for IconCache.db in Windows XP, Vista and 7

Signature offset	Hex	Text
Offset 4 for 4 bytes	57 69 6E 34	Win4

Table 2: Sample differences in IconCache.db file signature in Windows XP, Vista and 7

Operating System	Offset 0 For 1 byte		Offset 8 For 2 bytes		Offset 12 For 1 byte	
	Hex	Text	Hex	Text	Hex	Text
Windows XP (32 bit)	50	P	05 05	..	54	T
Windows Vista (32 bit)	40	@	06 05	..	71	q
Windows 7 Home Premium (32 & 64 bit)	40	@	06 05	..	B0	°
Windows 7 Professional (64 bit)	40	@	06 05	..	B1	±

4. Research Method

A formal test environment was established to ensure that the same experiment could be repeated consistently using the different versions of the Microsoft Windows operating system. The physical hardware used was a single PC workstation with an Intel Celeron processor (E3400 @ 2.60 Ghz), 4GB of RAM and a standard VGA card. The computer was not connected to any network.

Two 250 GB hard disks were securely wiped and used interchangeably in the following way:

Firstly, the test OS was installed a single user account was set up on the system. Experimentation, data collection and analysis followed. To eliminate potential data contamination or inconsistency during operating system installation and use, the hard disks were wiped before first use and between all experimental phases.

The primary operating systems tested were Windows XP Home (SP2), Windows Vista Home Basic (32 Bit), and Windows 7 Home Premium (32 bit). Preliminary testing was also carried out on Windows 7 Home Premium (64 bit).

In all cases, Windows was set up to run using UK English and with the time set to GMT London. To ensure consistency, a single user name and computer name was used throughout.

The forensic tools used to conduct this research are summarized in Table 3.

Table 3: Forensic tools used during experimentation

Hardware / Software	Purpose
Logitech Talon	1: Hard disk secure data deletion (WipeClean method) 2: Write-blocked hard disk capture
Wiebetech USB Writeblocker	Write-blocked attachment of USB devices to a forensic workstation host.
FTK Imager Lite by Access Data	Imaging test USB devices.
Forensic Tool Kit (FTK) v. 3.1 & 3.2 by Access Data.	Analysis of data.
dec Windows Thumbnail Database Viewer v. 1.8 by Dec Software ¹	Render icons during IconCache.db analysis

For each experiment, data was collected using a USB memory stick, formatted via Windows command line.

5. Experimentation

Based on the common or expected functionality found across all the Windows operating systems, three experiment stages were derived. These were: the clean install of an operating system as a starting point to baseline the experiments, initial interaction with the operating system and finally the extended or repeated user activity. In accordance with these stages and the objectives of this research, the experiments summarized in Table 4 were formulated.

¹Available from: <http://www.thumbnailexpert.com/en/products/commercial-tools/dec-windows-thumbnail-database-viewer/>

Table 4: Details of experiments devised

Set		Purpose	Activities
A: Baseline IconCache.db Research		Establish file provenance.	Ascertain: <ol style="list-style-type: none"> 1. When the IconCache.db file is created. 2. Its contents immediately following creation. 3. Its contents following a system restart. 4. What happens if the IconCache.db file is deleted. 5. What happens when a second user is set up on the host.
	A1	Further research on IconCache.db to corroborate initial state.	Ascertain: <ol style="list-style-type: none"> 1. Assessment of IconCache.db default system icons immediately following creation. 2. Review of default system icons in Shell32.dll
B: Broad empirical observation of the workings of the IconCache.db		To investigate the effects of common user activity on the database.	
	B1	To investigate the effects of basic user activity on the host alone.	Ascertain: <ol style="list-style-type: none"> 3. IconCache.db contents after opening files & folders on the host system. 4. Its contents after placing a link on the desktop from a host-based program. 5. Its contents after running a host-based executable.
	B2	To investigate the effects of user initiated DVD activity.	Ascertain: <ol style="list-style-type: none"> 1. IconCache.db contents after loading a DVD containing an executable file into the host DVD tray. 2. Its contents after installing a program on the host from a DVD.

	B3	To investigate the effects of connecting a USB thumb drive to the host	Ascertain: <ol style="list-style-type: none"> 1. IconCache.db contents following the connection of a clean USB thumb drive on the host. 2. Its contents after connecting a USB drive containing files and folders only. 3. Its contents after opening a file from a connected USB. 4. Its contents after connecting a USB drive containing one or more executables. 5. Its contents after an executable file is run from a thumb drive.
	B4	Further research on IconCache.db to corroborate findings	<ol style="list-style-type: none"> 1. Creating a new user account 2. File size changes for experiments in Set 3. Date and time changes for experiments in Set .

Results from the experiments listed in Table 4 are summarized in the following sections. In the interests of brevity, specific examples of findings recorded in this section relate to Windows XP only since the same basic behaviour patterns were observed during the experiments on Windows Vista and Windows 7 32-bit systems.

6.1 *IconCache.db* baseline behaviour

When performing a forensic examination, it is important to know the baseline state of IconCache.db files to distinguish it from artifacts of user activities. Conducting the first experiment from Table 4 (Baseline IconCache.db Research) for the Windows operating systems chosen for research purposes resulted in the following findings:

- The IconCache.db is not created during a clean install of the operating system.
- The IconCache.db is created on system reboot following a clean OS install.
- Where no user action apart from a system restart has taken place, an IconCache.db which contains a number of default system icon images is created as shown in Figure 1.
- The numbers of default system icons present in the IconCache.db at creation may vary depending upon the operating system in use and the configuration settings chosen during installation. The numbers of icons recorded in the IconCache.db as a result of a clean install for the operating systems under review are provided in Table 5.

Figure 1. Icons present in IconCache.db on initial creation (Windows XP)

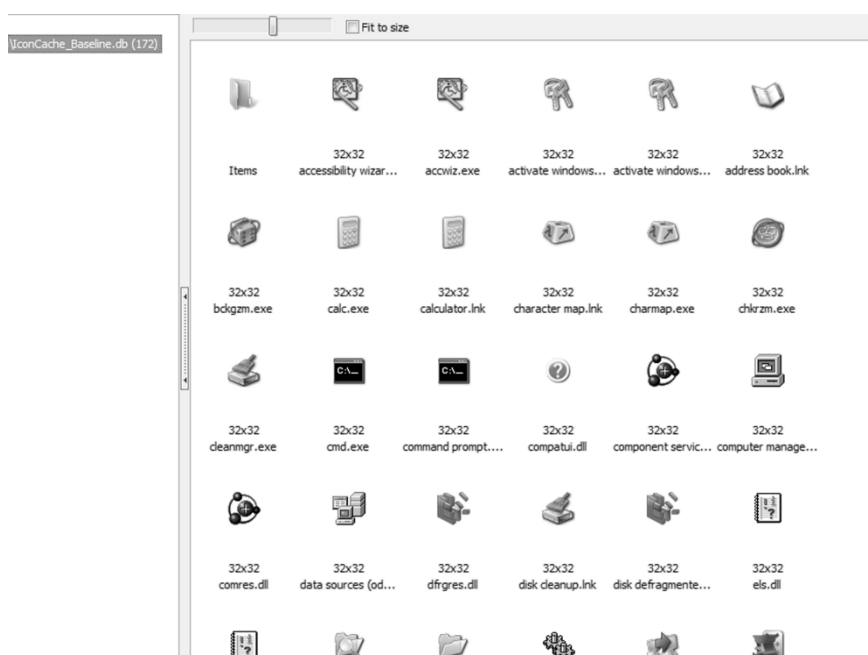


Table 5: Numbers of icons in the IconCache.db on first creation by version of Windows OS

Windows Operating System Version	Numbers of default system icons in IconCach.db on first creation of the file
Windows XP Home Edition SP. 2 (32bit)	172
Windows Vista Home Basic (32 bit)	426
Windows 7 Home Premium (32 bit)	158
Windows 7 Home Premium (64 bit)	183

Further examination of the IconCache.db file using forensic software revealed a listing of programs and processes associated with the default system icons as shown in Fig. 2 . Specifically, Fig. 2 shows a sample start of the IconCache.db file, where references to shell32.dll is first noticeable, closely followed by the loading of Internet Explorer from the program files folder on the host hard drive, allocated letter C:, as is commonly the case. A record of other automatic processes which run on system restart comes next, with those that access the user's *documents and settings* folder clearly shown.

Figure 2. Sample start of file, first generation of IconCache.db

```
shell32.dll shell32.dll shell32.dll shell32.dll shell32.dll shell32.dll shell32.dll shell32.dll shell32.dll  
shell32.dll/c:\program files\internet explorer\iexplore.exeGc:\documents and settings\jan\start  
menu\programs\internet explorer.lnk*c:\program files\outlook  
express\msimn.exeEc:\documents and settings\jan\start menu\programs\outlook  
express.lnk!%systemroot%\system32\rcimlby.exeGc:\documents and settings\jan\start  
menu\programs\remote assistance.lnk/c:\program files\internet  
explorer\iexplore.exe2c:\program files\windows media player\wmplayer.
```

Forensic examination of the end of the IconCache.db file reveals that certain executables were loaded, including the solitaire game which forms part of the Windows install package (Fig. 3). The record: d:\setup.exe, is also visible. This refers to the Windows XP executable, which has been run from the DVD drive, here allocated drive letter D: by the system.

Figure 3. Sample end of file, first generation of IconCache.db

```
c:\windows\system32\spider.exeRc:\documents and settings\all users\start  
menu\programs\games\spider solitaire.lnk c:\windows\system32\mydocs.dll  
shell32.dll c:\windows\explorer.exe c:\windows\explorer.exe d:\setup.exe  
shell32.dll c:\windows\system32\mydocs.dll shell32.dll shell32.dll shell32.dll  
c:\windows\system32\shdocvw.dll
```

A second system restart, immediately following the first, added data to the IconCache.db file as shown in Fig. 4. For instance, the fact that the logon screen had run was recorded by the “logon.scr” entry.

Figure 4. Sample end of file, simple system restart following first generation of IconCache.db

```
c:\windows\explorer.exe c:\windows\explorer.exe d:\setup.exe shell32.dll  
c:\windows\system32\mydocs.dll shell32.dll shell32.dll shell32.dll  
c:\windows\system32\shdocvw.dll c:\windows\system32\logon.scr shell32.dll
```

Two conclusions were drawn from the findings in Fig. 4. First, since this information did not exist in the IconCache.db at inception, it must have been added to the file during use of the system. Second, since the information was not visible in the database whilst the computer was running, it must have been added to the database on system shut down.

6.2 Default System Icons in IconCache.db

Some understanding of the origin of default system icons found in a baseline IconCache.db file can be helpful when explaining forensic findings. Tests carried out during this research indicated that, across the three Windows operating systems under review, the Shell.32.dll

contains a number of default icons at initiation, each of which exist in a number of versions which have differing dimensions, numbers of colours and file sizes, as illustrated in Figures 5 and 6, below:

Fig. 5: Icons stored in Shell32.dll in Windows XP

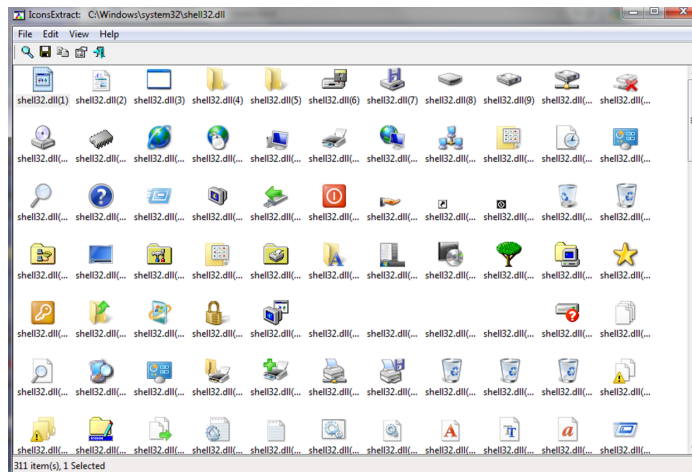
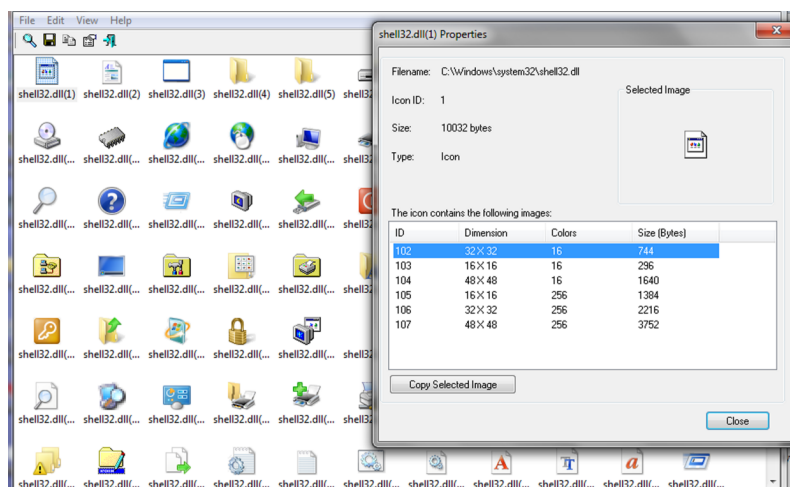


Fig 6: Shell32.dll stored icon properties in Windows XP



According to Microsoft (Hornick, 1995), Windows is aware of four different icon sizes – System Small, System Large, Shell Small and Shell Large. A single icon (.ICO) file or an .EXE or .DLL file can contain multiple images, each with a different size and/or colour depth.

6.3 IconCache Artifacts of Program Usage

From a forensic perspective, artifacts of user activities on a computer system can be useful for reconstructing events relating to an offense. Conducting the second set of experiments described in Table 4 resulted in findings summarized in the following sections, starting with the effects of basic user activity. The listing of programs and processes in the IconCache.db was shown to

grow as they were newly used, as did the number of associated icons retained. This finding was corroborated by further tests to establish file size changes prompted by experimentation. An example is given in Table 6 when a Desktop shortcut was created to the WordPad program.

Table 6: Increase in file size and icons retained in IconCache.db resulting from user action

Action	Number of icons in file	New icons added	File size (Bytes)
Pre-Action IconCache.db	181	n/a	2,149,330
Create link to 'WordPad.exe' on Desktop	188	7	2,149,870
Create 'WordPad' .txt file, save on Desktop	189	1	2,149,994

6.4 IconCache Artifacts of DVD Usage

Artifacts of user initiated DVD activity, including the installation and running of applications, can be very helpful in a digital investigation.

a) Loading a DVD containing an executable file

When a disk containing an application which had an 'autorun' icon was placed into the host's DVD drive, that icon was stored in the IconCache.db . Fig 7 illustrates how, when a DVD containing the forensic software application 'XWays' was loaded, two 'autorun' icons (the same graphic presented in different resolutions) were stored in the IconCache.db. The textual reading from the database shows the presence of both of these icons (Fig. 8). It also shows that the path to the application, including the drive letter associated with the DVD drive was retained.

Figure 7: Icons from application's 'autorun' icon present on DVD in IconCache.db

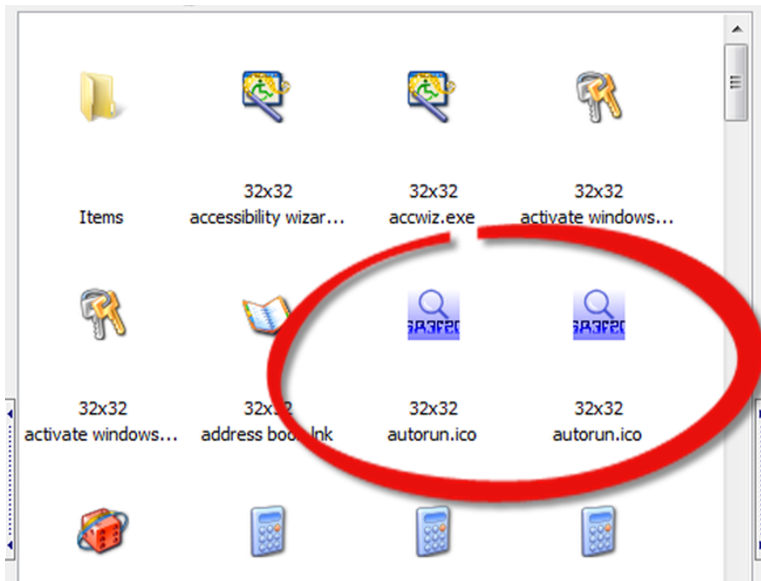


Fig. 8: 'Autorun' icon present in an application on DVD listed in IconCache.db

```
c:\windows\system32\spider.exeRc:\documents and settings\all users\start
menu\programs\games\spider
solitaire.lnk:~1\windows\explorer.exe:~1\windows\system32\mydocs.dll
shell32.dllc:\windows\explorer.exe shell32.dll shell32.dll shell32.dll
shell32.dllc:\windows\system32\shdocvw.dll shell32.dll shell32.dll shell32.dll
d:\autorun.ico shell32.dll d:\autorun.ico
```

b) Installing an executable from a DVD

When an application was installed on the host from an executable file on DVD, the icons for the application were stored in the IconCache.db. The path to the executable, both on the DVD and in the 'program files' folder on the host's C: drive, were also recorded in the file as shown in Fig. 9.

Fig. 9: Path to executable run from DVD in IconCache.db file

```
c:\windows\system32\mspaint.exe:~1\windows\system32\shimgvw.dll"d:\x-
ways forensics 14.9\setup.exe(d:\x-ways forensics 14.9\xwforensics.exe"d:\x-
ways forensics 14.9\setup.exe1c:\program files\x-ways
forensics\xwforensics.exe1
```

c) Running an installed executable

When the newly installed executable was then run on the host, a record this action was retained in the IconCache.db as shown in Fig. 10.

Fig. 10: Record of executable run from host in IconCache.db file

```
c:\windows\system32\mspaint.exec:\windows\system32\shimgvw.dll"d:\x-ways forensics 14.9\setup.exe(d:\x-ways forensics 14.9\xwforensics.exe"d:\x-ways forensics 14.9\setup.exe1c:\program files\x-ways forensics\xwforensics.exe1c:\program files\x-ways forensics\xwforensics.exec:\windows\winhlp32.exe
```

6.5 IconCache Artifacts of USB Device Usage

Artifacts of user initiated USB activity, including opening files and running executables, can be very helpful in a digital investigation.

a) Opening a text file

When a simple text file was opened from a memory stick, the IconCache.db recorded the running of both notepad.exe and wordpad.exe from the C: drive but did not retain the drive letter for the stick (Fig. 11). Icons for both programs were also stored in the database.

Figure 11: Sample end of file after text document run from USB memory stick

```
c:\windows\system32\mydocs.dll shell32.dll shell32.dll shell32.dll  
c:\windows\system32\shdocvw.dll c:\windows\system32\logon.scr shell32.dll  
c:\windows\system32\notepad.exe3c:\program files\windows  
nt\accessories\wordpad.exe
```

b) Viewing a USB containing executable files

When a USB memory stick containing an executable in the root of the drive was connected and viewed on the host, icons associated with that executable were stored in the IconCache.db despite the fact that it had not been run. As illustrated in Fig. 12, in a test in which a USB with the integral encryption program, Flashlokv232.exe, was connected to the host, its icon was stored in the IconCache.db file (first icon, second row down). The path to the associated executable files, together with the drive letter allocated to the USB stick, was also retained (Fig. 13).

Figure 12: Icons from executables present on USB memory stick in IconCache.db file

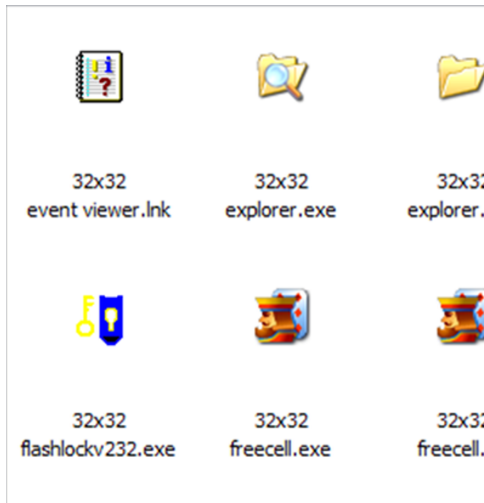


Figure 13: Listing of executables on USB connectable shown in ASCII within IconCache.db

```
shell32.dllc:\windows\system32\logon.scr"%systemroot%\system32\wiaacm
gr.exe shell32.dll shell32.dll shell32.dll
shell32.dll!%systemroot%\system32\shimgvw.dll e:\osf.exe
c:\windows\system32\zipfldr.dll e:\flashlockv232.exe shell32.dll
c:\windows\system32\wiaacmgr.exe
```

c) Running an executable file from USB

When an executable was run from a USB drive, icons associated with it were stored in the IconCache.db. In a test in which the forensic software FTK imager lite was run from a USB, two icons associated with it were shown in the IconCache.db. The textual reading from the database showed that these were both recorded, along with the path to the executable (Fig. 14).

Figure 14: Icons of Executable run from USB connectable shown in IconCache.db

```
\system32\logon.scr:c:\windows\explorer.exec:\windows\system32\mydocs.d
ll
shell32.dllc:\windows\explorer.exec:\windows\system32\shdocvw.dllshell32.
dllshell32.dllshell32.dllc:\windows\system32\mshta.exec:\windows\system32
\cryptui.dll!f:\imager lite 2-1\ftk imager.exe!f:\imager lite 2-1\ftk imager.exe
```

d) Installing an executable from a USB

When an executable was installed on a host from a USB drive, the icons associated with it were stored in the IconCache.db. In a test involving the once popular peer-to-peer application, Limewire, two instances of its executable icon and two associated setup icons were retained in the IconCache.db, (Fig. 15). The path to the executable on the USB was also revealed in the ASCII (Fig. 16).

Figure 15: Icons from Limewire application, installed from USB connectable & run on host shown within IconCache.db file

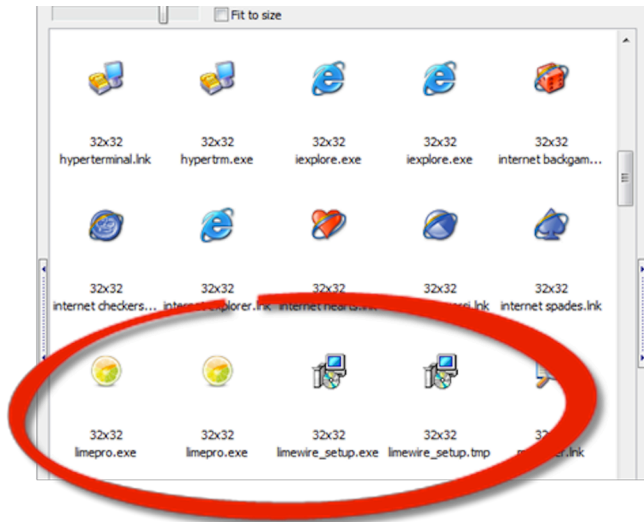


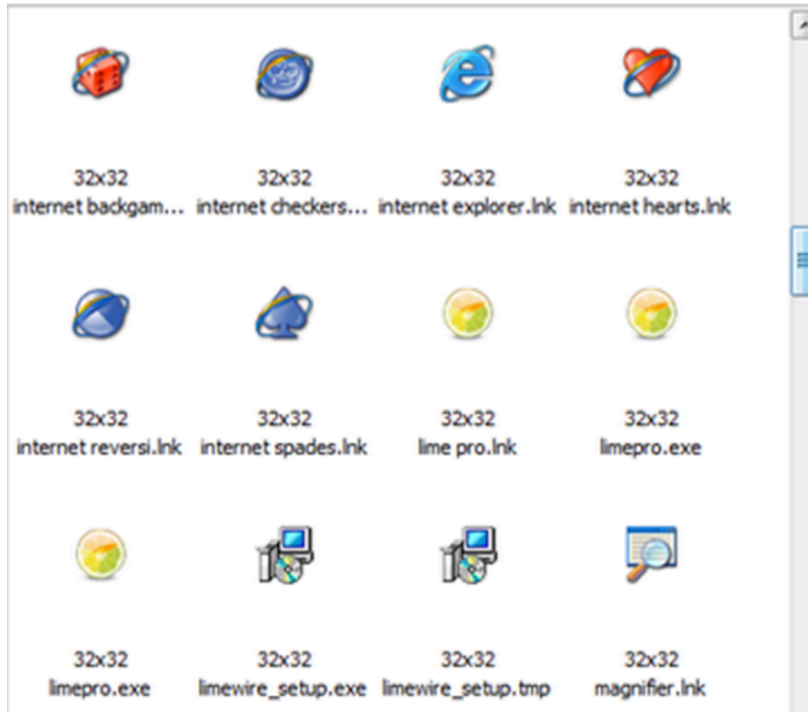
Figure 16: Executable installed from USB connectable & run on host shown within IconCache.db file

```
shell32.dllc:\windows\system32\shdocvw.dllshell32.dllc:\windows\system32\syncui.dll
shimgvw.dllc:\programfiles\windowsnt\accessories\wordpad.exe!c:\progra~1>window~2\w
mplayer.exec:\windows\system32\zipfldr.dllf:\limewire_setup.exe=c:\docume~1\max\locals
~1\temp\is-trm8s.tmp\limewire_setup.tmp%c:\program files\lime
pro\limepro.exe%c:\program files\lime pro\limepro.exe&c:\program files\lime
pro\unins000.exe
```

e) Uninstalling an executable placed on the host via USB

When the Limewire executable was subsequently deleted from the Windows host, using the program’s own uninstaller, five icons related to it nevertheless remained in the IconCache.db, as shown in Fig. 17 (icons 3 & 4 second row and 1, 2 & 3, third row).

Figure 17: Icons retained in IconCache.db following uninstall of Limewire program



6.6 Creating a New User Account

When a new user account was set up on the host machine, an IconCache.db for that user was created in their profile, separate to and different from the IconCache.db for the original user, as shown in the ASCII output in Fig. 18. The creation of the new user did not update the original user's IconCache.db with any information about new user activity. The original user's IconCache.db was merely updated with a call to "System 32\mshta.exe," which is a system file associated with the Microsoft HTML application host.

Figure 18: First instance of IconCache.db for new user 'Max'

```
shell32.dll/c:\program files\internet explorer\iexplore.exeGc:\documents
and settings\max\start menu\programs\internet
explorer.lnk*c:\program files\outlook express\msimn.exeEc:\documents
and settings\max\start menu\programs\outlook express.lnk/c:\program
files\internet
```

6.7 IconCache.db File System Information

Changes in date and time stamps for the IconCache.db were recorded during experimentation. It was observed that, for the 32-bit operating systems tested during this research:

- When the IconCache.db was deleted, it was recreated on system reboot with its original created date/time and with a modified time which reflected the time of its deletion. This finding supports the supposition that the database is written to disk at system shut down or reboot.
- The IconCache.db file modification times were updated following actions which caused new information to be retained in it – the example chosen here was creating a text file and

saving it to a USB memory stick.

Table 7. Date and Time changes to IconCache.db in Windows XP

Experiment	Action	Created		Modified	
A. 1.1	First instance IconCache.db	28/02/12	21.07	28/02/12	21.07
A. 1.1	Delete IconCache.db@ 21.33, restart	28/02/12	21.07	28/02/12	21.33
B2. 1.3	Create .txt file, save to USB, restart	28/02/12	21.07	28/02/12	21.12

Preliminary testing on a Windows 7 Home Basic 64-bit installation produced a variation in results, as follows:

- When the IconCache.db was deleted, it was not immediately recreated on system re-boot. In fact, no IconCache.db was apparent.
- A new IconCache.db was created following a second system reboot. It was smaller in size than the original (804 kb as opposed to 1.08 mb following a fresh install).

In common with the 32-bit systems examined, however, modification times for the IconCache.db were updated following actions which caused new information to be retained in it.

8. Conclusion

The IconCache.db can contain artifacts of user activities that could be useful in digital investigation. Artifacts retained in the IconCache.db can be especially useful in identifying executable files that were run or installed from external media such as DVDs and USB connectable devices.

The ability to trace the installation or use of a foreign program to a USB connectable or other media can help to narrow down the search for a culprit, can reveal usage of external encryption programs, and can suggest where other evidence might exist. Care should be taken, however, in interpreting findings since the IconCache.db may contain the icons of executables which were merely stored on a USB connectable, not actually run from it. Opening and viewing the contents of a USB device which contains a program executable file will cause any related icon to be stored in the database.

The IconCache.db contains a wealth of information including file paths to the programs and processes which have been invoked on fixed and attached drives. Since an IconCache.db exists for each named user of a computer, the file paths also reveal which activities occurred under which user name.

Further research is needed in order to gain a better understanding of the IconCache.db in terms of its structure and functionality. This would greatly aid the interpretation of information to be found in the file. An opportunity also exists to develop software capable of rendering icons stored in the data base together with properly parsed date and time data.

Further research is needed in order to clarify how the IconCache.db behaves in 64 bit versions of the Windows operating system. Initial experimentation with Windows 7 Home Premium 64 bit has indicated that, once created, the file stores artifacts associated with user-based computer activity in a similar way to the 32-bit operating systems which were tested during this research. However, how the file is created or recreated following deletion is a subject for future enquiry.

REFERENCES

Carvey, H, (2005) The Windows Registry as a forensic resource. *Digital Investigations* 2005; 2: 201-5.

Carvey, H. and Altheide, C.,(2005) Tracking USB storage: Analysis of windows artifacts generated by USB storage devices. *Digital Investigation* 2005; 2: 94 - 100.

Carvey, H, (2009). *Windows Forensic Analysis DVD Toolkit 2E*. Chapter 4, P206- 213. Syngress/Elsevier, USA.

Dec. Windows shell icon cache

Available from: <http://www.thumbnailexpert.com/en/formats/windows-shell-icon-cache/>; September, 2011.

DiRenzo, J. (2012), The Risk of Disruption or Destruction of Critical U.S. Infrastructure by an Offensive Cyber Attack, research paper

Available from: <http://blog.havagan.com/wp-content/uploads/2012/05/The-Risk-of-Disruption-or-Destruction-of-Critical-U.S.-Infrastructure-by-an-Offensive-Cyber-Attack.pdf>; July, 2012.

Garrity S and Weir G. (2010), Balancing the threat of personal technology in the workplace *Int. J. Electronic Security and Digital Forensics, Vol. 3, No. 1, 2010 pp.73-81*.

Holderness, James (1999) Undocumented Windows 95, 'The Shell Icon Cache'.

Available from: <http://reocities.com/SiliconValley/4942/iconcache.html>; September, 2011.

Hornick, John (1995) MSDN Library, 'Icons'.

Available from: <http://msdn.microsoft.com/en-us/library/ms997538.aspx>; September, 2011.

Marketshare, (2012), Operating System market share.

Available from: <http://marketshare.hitslink.com/operating-system-market-share.aspx?qprid=8>; July, 2012.

Mee, V and Jones, A. (2005) The Windows operating system registry - a central repository of evidence. In: *Proceedings from e-crime and computer evidence conference 2005*.

Mee, V, Tryfonas, T, and Sutherland, I, (2006), The Windows Registry as a forensic artifact: Illustrating evidence collection for Internet usage. *Digital Investigation* 3 (2006) P. 166-173.

NirSoft, (n.d). MUICacheView v1.01

Available from: http://www.nirsoft.net/utills/muicache_view.html; September, 2011.

Pham, D. V., Halgamuge, M. N. and Mendis P. (2010), Optimizing Windows Security Features to Block Malware and Hack Tools on USB Storage Devices, PIERS Proceedings, Cambridge, USA, July 5-8, 2010 .

Roy, Tanushree, Jain, Aruna (2012) Windows Registry Forensics: An Imperative Step in Tracking Data Theft via USB Devices. International Journal of Computer Science and Information Technologies (IJCSIT), Vol 3(3), 2012, P. 4427-4433.

Russinovich, M & Solomon, D (2005), Microsoft Windows Internals, Fourth Edition: Microsoft Windows Server 2003, Windows XP and Windows 2000. Microsoft Press, Washington. Chapter 6, Pp 311 - 313.

Sharma, V. (2011), An Analytical Survey of Recent Worm Attacks, IJCSNS International Journal of Computer Science and Network Security, VOL.11 No.11, November 2011, pp.99-103.

StatCounter, (2012), Top 5 operating systems.

Available from: <http://gs.statcounter.com/#os-ww-monthly-201003-201105>; July, 2012.

Stevens, D, (n.d) UserAssist

Available from:<http://blog.didierstevens.com/programs/userassist/>; December, 2010.

Tetmeyer, A. (2010), Security Threats and Mitigating Risk for USB Devices, Technology and Society Magazine, IEEE, Vol. 29, Issue 4, Winter 2010, pp.44-49.

Thomas, P, and Morris, A, (2008) , 'An investigation into the development of an anti-forensic tool to obscure USB flash drive device information on a windows XP platform.' ADFIA'08. Third International Annual Workshop on Digital Forensics and Incident Analysis, 2008. P. 60 - 66.

W3schools, 2012, Web statistics and trends, OS platform statistics

Available from: http://www.w3schools.com/browsers/browsers_os.asp; July, 2012.

APPENDIX V
IconCache.db research work cited and extended
[Pages removed for Copyright reasons]

APPENDIX VI

Form for identifying vPCs and Windows OSs and monitoring experiments carried out

vPC name	vPC Designation:	Windows OS:	OS Designation:
MojoPac	A	XP Pro (32)	A
Ceedo	B	XP Pro (64)	B
Lupo PenSuite	C	7 Pro (32)	C
Portable Apps	D	7 Pro (64)	D

vPC:		OS:		
Experiment:		Details	Artefacts?	Record sheet
1	Copy a text file ; vPC to host			
2	Copy a text file ; host to vPC			
3	Copy a picture file ; vPC to host			
4	Copy a picture file; host to vPC			
5	Write and save a text file on the vPC			
6	Run a program executable on the vPC			
7	Launch a browser on the vPC			
8	Conduct named search from vPC browser			

APPENDIX VII
Form devised for collecting experimental results

Test OS:	Test vPC:	Record sheet:
Experiment:		
Area of Operating System		
Registry HKey		
SYSTEM\CurrentControlSet	\EnumUSB	Artefacts? Y/N
	\EnumUSBSTOR	
SYSTEM	\Control\DeviceClasses\{a\phanumeric}	Example:
SYSTEM	\MountedDevices	
SOFTWARE\Microsoft	\Windows Portable Devices\Devices	
	\Windows\CurrentVersion\Explorer\MountPoints2	
	\Windows\CurrentVersion\Explorer\UserAssist	
	Windows\Shell	
	Windows\ShellNoRoam	
	Windows\ShellNoRoam\MUICache (Win XP)	
	Windows\StreamMRU	
SOFTWARE\Classes	\LocalSettings\Software\Microsoft\Windows\Shell\MuiCache (WinVista, Win7)	
UsrClass.dat\Local Settings	\Software\Microsoft\Windows\Shell	
Prefetch		
Lnk files		
IconCache.db		
Pagefile (s)		

Artefacts in Live Memory:

Further:

C:\Windows\inf\setupapi.dev.log – Time/Timezone (First Connect)

APPENDIX VIII

**‘Quick reference’ chart for vPC artefacts
in Windows XP and 7 locations**

Location		vPC				
		MojoPac	Ceedo Personal		Portable Apps	Lupo PenSuite
		Win XP	XP	Win 7	Win 7	Win7
NTUser.dat	UserAssist	N	Y	Y	Y	Y
	ComDlg32	N	N	N	Y	Y
	MountPoints2	Y	Y	Y	Y	Y
MuiCache		Y	Y	N	N	N
Prefetch		Y	Y	Y	Y	Y
Lnk files		Y	N	N	N	N
IconCache.db		Y	Y	Y	Y	Y

APPENDIX IX

IconCache.db sample file size changes during experiments

NB: ‘Baseline IconCache.db’ indicates properties of .db file before the listed experiment was run, not a virgin IconCache.db file.

Experiment A

Exp	Action	Number of icons in file	New icons added	File size (Bytes)
	Clean OS install	No file		No file
A1.3	Restart host	172		2,148,748
A1.4	Delete IconCache.db, restart	181	13	2,149,246

Experiment B1

Exp	Action	Number of icons in file	New icons added	File size (Bytes)
	Baseline IconCache.db	181		2,149,330
1.1	Create link to ‘WordPad.exe’ on Desktop	188	7	2,149,870
1.3	Create ‘WordPad’ .txt file, save on Desktop	189	1	2,149,994

Experiment B2

Exp	Action	Number of icons in file	New icons added	File size (Bytes)
	Baseline IconCache.db	181		2,149,246
2.1	Place DVD containing 1 executable in D: drive	186	5	2,149,566
2.2	Install executable from DVD	192	6	2,150,096
2.3	Run executable from Host	195	3	2,678,404

Experiment B.3

Exp	Action	Number of icons in file	New icons added	File size (Bytes)
	Baseline IconCache.db	182		2,149,330
3.1	Insert clean memory stick	184	2	2,149,454
3.2	Insert memory stick containing files only	184	0	2,149,454
3.4	Insert memory stick containing program executable	183	2	2,149,374
3.5	Run executable from memory stick	192	9	2,150,104

APPENDIX X
Process Monitor sample readings for MojoPac

Appendix X a)
Sample Outputs Baseline Test a) MojoPac

17:27.9 services.exe
17:27.9 services.exe
17:27.9 services.exe
17:27.9 services.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 services.exe
17:27.9 services.exe
17:27.9 services.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe

17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 csrss.exe
17:27.9 csrss.exe
17:27.9 csrss.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:27.9 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe

17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 services.exe
17:28.3 services.exe
17:28.3 services.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 svchost.exe
17:28.3 services.exe

HKLM\System\CurrentControlSet\Control\IDConfigDB
HKLM\System\CurrentControlSet\Control\IDConfigDB\CurrentConfig
HKLM\System\CurrentControlSet\Control\IDConfigDB\Hardware Profiles\0001
HKLM\System\CurrentControlSet\Control\IDConfigDB\CurrentDockInfo
HKLM\System\CurrentControlSet\Control\IDConfigDB\DockingState
HKLM\System\CurrentControlSet\Control\IDConfigDB\Hardware Profiles\0001\DockState
HKLM\System\CurrentControlSet\Control\IDConfigDB\Hardware Profiles\0001\HwProfileGuid
HKLM\System\CurrentControlSet\Control\IDConfigDB\Hardware Profiles\0001\FriendlyName
HKLM\System\CurrentControlSet\Control\IDConfigDB
HKLM\System\CurrentControlSet\Control\IDConfigDB\Hardware Profiles\0001
HKLM\System\CurrentControlSet\Control\IDConfigDB\CurrentDockInfo
HKLM\System\CurrentControlSet\Enum\USB\VID_0951&PID_1643\0013728EE05CEAA1F51E0002
HKLM\System\CurrentControlSet\Enum\USB\Vid_1643\0013728EE05CEAA1F51E0002\Capabilities
HKLM\System\CurrentControlSet\Enum\USB\Vid_1643\0013728EE05CEAA1F51E0002
HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\AutoplayHandlers\Files
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\AutoplayHandlers\CancelAutoplay\Files
E:\
E:\ *setup* .exe
E:\
E:\
E:\
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\AutoplayHandlers\CancelAutoplay\Files
E:\
E:\ *instal* .exe
E:\
E:\
E:\
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\AutoplayHandlers\CancelAutoplay\Files
HKLM\System\CurrentControlSet\Enum\USB\VID_0951&PID_1643\0013728EE05CEAA1F51E0002

E:\HVDVD_TS
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\AutoplayHandlers
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\AutoplayHandlers\IsAutorunForCDROMOnly
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\AutoplayHandlers
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\IniFileMapping
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\IniFileMapping\Autorun.inf
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\IniFileMapping
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\
E:\
E:\
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\autorun.inf
E:\
E:

Desired Access: Read
Desired Access: Delete

Desired Access: Maximum Allowed
Length: 144
c91efb8b}

Desired Access: Read
Type: REG_DWORD, Length: 4, Data: 20

Control: IOCTL_STORAGE_CHECK_VERIFY
Control: IOCTL_STORAGE_CHECK_VERIFY

Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Attributes: n/a, ShareMode: Read, Write, All Name: \

VolumeCreationTime: 01/01/1601 01:00:00, VolumeSerialNumber: 6420-3808, SupportsObjects: False, VolumeLabel: MOJO

FileSystemAttributes: Case Preserved, Unicode, MaximumComponentNameLength: 255, FileSystemName: FAT32

Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationSize: n/a, ImpersonationCreationTime: 01/01/1601 01:00:00, LastAccessTime: 01/01/1601 01:00:00, LastWriteTime: 01/01/1601 01:00:00, ChangeTime: 01/01/1601 01:00:00, FileAttributes

Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationSize: n/a, ImpersonationCreationTime: 15/10/2008 03:51:54, LastAccessTime: 22/09/2014 00:00:00, LastWriteTime: 15/10/2008 03:51:54, ChangeTime: 01/01/1601 01:00:00, FileAttributes

Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationSize: n/a, Impersonation

Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationSize: n/a, Impersonation

Desired Access: Read Attributes, Disposition: Open, Options: Open Reparse Point, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationSize: n/a, ImpersonationType: REG_DWORD, Length: 4, Data: 1

Desired Access: Read
Desired Access: Read

Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-Alert, Non-Directory File, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationExclusive: False, Offset: 0, Length: 4,294,967,295, Fail Immediately: False
AllocationSize: 4,096, EndOfFile: 113, NumberOfLinks: 1, DeletePending: False, Directory: False
Offset: 0, Length: 113
Offset: 0, Length: 113, /O Flags: Non-cached, Paging I/O, Synchronous Paging I/O
Offset: 0, Length: 4,294,967,295

Offset: 0, Length: 4,096, /O Flags: Non-cached, Paging I/O, Synchronous Paging I/O

Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-Alert, Non-Directory File, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationExclusive: False, Offset: 0, Length: 4,294,967,295, Fail Immediately: False
AllocationSize: 4,096, EndOfFile: 113, NumberOfLinks: 1, DeletePending: False, Directory: False
Offset: 0, Length: 113
Offset: 0, Length: 4,294,967,295

Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-Alert, Non-Directory File, Attributes: n/a, ShareMode: Read, Write, Delete, AllocationExclusive: False, Offset: 0, Length: 4,294,967,295, Fail Immediately: False
AllocationSize: 4,096, EndOfFile: 113, NumberOfLinks: 1, DeletePending: False, Directory: False
Offset: 0, Length: 113
Offset: 0, Length: 4,294,967,295

Desired Access: Query Value
Type: REG_DWORD, Length: 4, Data: 1
Desired Access: Query Value
Desired Access: Query Value
Type: REG_DWORD, Length: 4, Data: 3
Length: 144
Type: REG_SZ, Length: 78, Data: {96f483c0-4091-11e4-ab80-806d6172696f}
Type: REG_SZ, Length: 20, Data: Profile 1

Desired Access: Read
Type: REG_DWORD, Length: 4, Data: 20

Desired Access: Maximum Allowed
Index: 0, Name: *setup*.exe, Type: REG_SZ, Length: 2, Data:
Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Attributes: n/a, ShareMode: Read, Write, All
Filter: *setup*.exe

Index: 1, Name: *instal*.exe, Type: REG_SZ, Length: 2, Data:
Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Attributes: n/a, ShareMode: Read, Write, All
Filter: *instal*.exe, 1: mojopacinstaller.exe

Desired Access: Read/Write

LocationSize: n/a, Impersonating: NT AUTHORITY\SYSTEM, OpenResult: Opened

NT AUTHORITY\SYSTEM, OpenResult: Opened
.: D

NT AUTHORITY\SYSTEM, OpenResult: Opened
.: R

NT AUTHORITY\SYSTEM
NT AUTHORITY\SYSTEM

Size: n/a, Impersonating: NT AUTHORITY\SYSTEM, OpenResult: Opened

Size: n/a, Impersonating: NT AUTHORITY\SYSTEM, OpenResult: Opened

Size: n/a, Impersonating: NT AUTHORITY\SYSTEM, OpenResult: Opened

ocationSize: n/a, OpenResult: Opened

ocationSize: n/a, OpenResult: Opened

Appendix X b) Sample Outputs Baseline Test b) MojoPac

Operation	Path
CreateFile	G:\mojopacinstaller.exe
QueryBasicInformationFile	G:\mojopacinstaller.exe
CloseFile	G:\mojopacinstaller.exe
CreateFile	G:\mojopacinstaller.exe
QueryBasicInformationFile	G:\mojopacinstaller.exe
CloseFile	G:\mojopacinstaller.exe
CreateFile	G:\mojopacinstaller.exe
QueryBasicInformationFile	G:\mojopacinstaller.exe
SetBasicInformationFile	G:\mojopacinstaller.exe
ReadFile	G:\mojopacinstaller.exe
ReadFile	G:\mojopacinstaller.exe
QueryBasicInformationFile	G:\mojopacinstaller.exe
QueryDirectory	G:\MojoPacIncomplete.txt
QueryDirectory	G:\MojoPrepUpdate.dat
RegLoadKey	HKLM\MojoControl
RegCloseKey	HKLM\MOJOCONTROL
RegOpenKey	HKLM\MojoControl
RegQueryValue	HKLM\MOJOCONTROL\deletelog
RegQueryValue	HKLM\MOJOCONTROL\deletelog
RegQueryValue	HKLM\MOJOCONTROL\sequence
RegQueryValue	HKLM\MOJOCONTROL\cache
RegQueryValue	HKLM\MOJOCONTROL\prefixes
RegQueryValue	HKLM\MOJOCONTROL\prefixes

RegQueryValue	HKLM\MOJOCONTROL\prefixes
RegQueryValue	HKLM\MOJOCONTROL\prefixes
RegQueryValue	HKLM\MOJOCONTROL\Version
RegCloseKey	HKLM\MOJOCONTROL
RegOpenKey	HKLM\MojoControl\active\WINDOWS_PREFETCH\PRECONFIG.EXE-16D7B139.PF
RegOpenKey	KLM\MojoControl\active\WINDOWS_PREFETCH
RegOpenKey	KLM\MojoControl\active\PROGRAM FILES\RINGTHREE\BIN\WINSPOOL.DRV
RegOpenKey	KLM\MojoControl\active\PROGRAM FILES\RINGTHREE\BIN
RegOpenKey	KLM\MojoControl\active\PROGRAM FILES\RINGTHREE
RegOpenKey	KLM\MojoControl\active\PROGRAM FILES
QueryNameInformationFile	G:\WINDOWS
CloseFile	G:\WINDOWS
CreateFile	G:\WINDOWS\WINSXS
QueryDirectory	G:\WINDOWS\WINSXS\POLICIES
QueryNameInformationFile	G:\WINDOWS\WINSXS
QueryNameInformationFile	G:\WINDOWS\WINSXS
CloseFile	G:\WINDOWS\WINSXS
CreateFile	G:\WINDOWS\WINSXS\POLICIES\x86_Policy.6.0.Microsoft.Windows.Common-Controls_6595
CreateFile	US_580a28ff
CreateFile	G:\WINDOWS
QueryDirectory	G:\WINDOWS\ASSEMBLY
QueryNameInformationFile	G:\WINDOWS
QueryNameInformationFile	G:\WINDOWS
CloseFile	G:\WINDOWS
CreateFile	G:\WINDOWS\ASSEMBLY\GAC\Policy.6.0.Microsoft.Windows.Common-Controls
CreateFile	G:\WINDOWS

QueryDirectory	G:\WINDOWS\system32
QueryNameInformationFile	G:\WINDOWS
CreateFile	G:\Documents and Settings\RingCube\Start Menu\Programs\Accessories\Program Compatibility Wizard.
ReadFile	G:\Documents and Settings\RingCube\Start Menu\Programs\Accessories\Program Compatibility Wizard.
ReadFile	G:\Documents and Settings\RingCube\Start Menu\Programs\Accessories\Program Compatibility Wizard.
CloseFile	G:\Documents and Settings\RingCube\Start Menu\Programs\Accessories\Program Compatibility Wizard.
WriteFile	G:\Documents and Settings\RingCube\Start Menu\Programs\Accessories
	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\ShellCom
RegOpenKey	3309D}
	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Classes\CLSID\{871C5380-42A0-1069-A2EA-
RegOpenKey	3309D}\InProcServer32
	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Classes\CLSID\{871C5380-42A0-1069-A2EA-
RegOpenKey	3309D}\InProcServer32
	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Classes\CLSID\{871C5380-42A0-1069-A2EA-
RegQueryValue	3309D}\InProcServer32\{Default}
	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Classes\CLSID\{871C5380-42A0-1069-A2EA-
RegCloseKey	3309D}\InProcServer32
	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Classes\CLSID\{871C5380-42A0-1069-A2EA-
RegOpenKey	3309D}\InProcServer32
	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Classes\CLSID\{871C5380-42A0-1069-A2EA-
RegQueryValue	3309D}\InProcServer32\LoadWithoutCOM
	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Classes\CLSID\{871C5380-42A0-1069-A2EA-
RegCloseKey	3309D}\InProcServer32
	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Classes\CLSID\{871C5380-42A0-1069-A2EA-
RegCloseKey	3309D}\InProcServer32

RegOpenKey	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\Software\Microsoft\Windows\CurrentVersion\Shell
	1s
RegCloseKey	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Shell
	1s
RegCreateKey	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\Software\Microsoft\Windows\CurrentVersion\Shell
	1s\Blocked
RegOpenKey	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Shell
	1s\Blocked
RegQueryValue	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Shell
	0309D}
RegCloseKey	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Shell
	1s\Blocked
RegOpenKey	HKLM\RINGTHREE\VM1\REGISTRY\USER\CURRENT\Software\Microsoft\Windows\CurrentVersion\Shell Extensions
	HKLM\RINGTHREE\VM1\REGISTRY\USER\CURRENT\Software\Microsoft\Windows\CurrentVersion\Shell Extensions
RegCloseKey	HKLM\RINGTHREE\VM1\REGISTRY\USER\CURRENT\Software\Microsoft\Windows\CurrentVersion\Shell Extensions
	HKLM\RINGTHREE\VM1\REGISTRY\USER\CURRENT\Software\Microsoft\Windows\CurrentVersion\Shell Extensions\Blocked
RegCreateKey	HKLM\RINGTHREE\VM1\REGISTRY\USER\CURRENT\Software\Microsoft\Windows\CurrentVersion\Shell Extensions\Blocked
	HKLM\RINGTHREE\VM1\REGISTRY\USER\CURRENT\Software\Microsoft\Windows\CurrentVersion\Shell Extensions\Blocked
RegOpenKey	HKLM\RINGTHREE\VM1\REGISTRY\USER\CURRENT\Software\Microsoft\Windows\CurrentVersion\Shell Extensions\Blocked
	HKLM\RINGTHREE\VM1\REGISTRY\USER\CURRENT\Software\Microsoft\Windows\CurrentVersion\Shell Extensions\Blocked\{871C5
RegQueryValue	0309D}
	HKLM\RINGTHREE\VM1\REGISTRY\USER\CURRENT\Software\Microsoft\Windows\CurrentVersion\Shell Extensions\Blocked
RegCloseKey	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\Software\Microsoft\Windows\CurrentVersion\Shell Extensions\Blocked
	HKLM\RINGTHREE\VM1\REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explo
RegOpenKey	

Appendix X c) Sample findings on Host HD - Baseline Test b) MojoPac

/NONAME [NTFS]/[root]/Documents and Settings/Administrator/NTUSER.DAT",125",1", " this file E:\mojopacinstaller.exe mojopacinstal
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/NTUSER.DAT",125",1", " this file E:\mojopacinstaller.exe mojopacinstal
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/NTUSER.DAT",125",1", " this file E:\mojopacinstaller.exe mojopacinstal
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/NTUSER.DAT",125",1", " this file E:\mojopacinstaller.exe mojopacinstal
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/NTUSER.DAT",125",1", " this file E:\mojopacinstaller.exe mojopacinstal
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/NTUSER.DAT",125",1", " this file E:\mojopacinstaller.exe mojopacinstal
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/NTUSER.DAT",125",1", " this file E:\mojopacinstaller.exe mojopacinstal
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/NTUSER.DAT",125",1", " this file E:\mojopacinstaller.exe mojopacinstal
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/NTUSER.DAT",125",1", " this file E:\mojopacinstaller.exe mojopacinstal

/NONAME [NTFS]/[root]/Documents and
Settings/Administrator/LocalSettings/Temp/RingThreeInstallerHelper.dll",100",1", "ram
Files/RingCube\MojoPac ?* |<>" " %s\Program Fil"
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/Local
Settings/Temp/RingThreeInstallerHelper.dll",100",2", "allation 2.0.0.0 MojoPac Version: Welcome
to M"
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/Local
Settings/Temp/RingThreeInstallerHelper.dll",100",3", "sion: Welcome to MojoPac Installation
ForceRemo"
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/Local


```

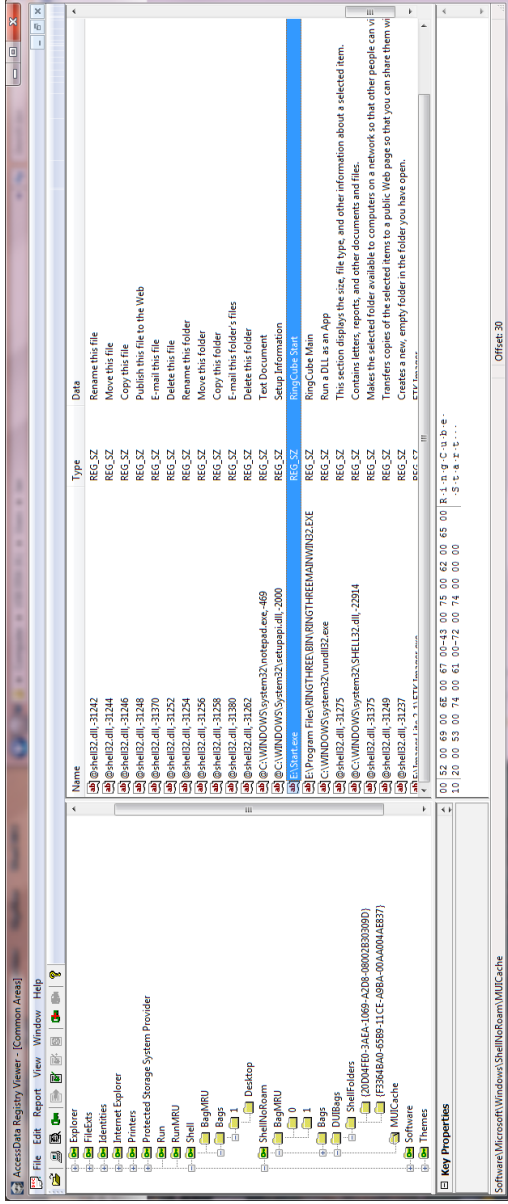
/NO NAME [NTFS]/[root]/$LogFile", "15", "4", " MOJOPA~1.LNKgCu0 MojoPac
(RingCube).lnk MojoPac"
/NO NAME [NTFS]/[root]/$LogFile", "15", "5", "ac (RingCube).lnk MojoPac Website.lnk
MOJOPA~2.L"
/NO NAME [NTFS]/[root]/$LogFile", "15", "6", " MOJOPA~2.LNKite0 MojoPac
Website.lnk http://ww"
/NO NAME [NTFS]/[root]/$LogFile", "15", "7", "e.lnk http://www.mojopac.com
Start.exe Start.e"
/NO NAME [NTFS]/[unallocated space]/00314073/00322682", "14134", "1", ".0 %s.sys
Warning: MojoPac detected a version conf"

/NO NAME [NTFS]/[unallocated space]/00314073/00322682", "14134", "5", "fessional
CISCO_PV MojoPac Registry backup MojoPac"
/NO NAME [NTFS]/[unallocated space]/00314073/00322682", "14134", "6", "ac Registry
backup MojoPac Print Support OPSWAT JU"
/NO NAME [NTFS]/[unallocated space]/00314073/00322682", "14134", "7", "in this
version of MojoPac. The Feature FeatureNam"
space]/00000067/00041905", "11251", "1", ".0 %s.sys Warning: MojoPac detected a
version conf"
/NO NAME [NTFS]/[unallocated space]/00000067/00041905", "11251", "2", "o continue
anyway? MojoPac Core Conflict (%s,%s) 0"
/NO NAME [NTFS]/[unallocated space]/00000067/00041905", "11251", "3", "river
successfully MojoPac ; ncln stalled = %d, run"
/NO NAME [NTFS]/[unallocated space]/00000067/00041905", "11251", "4", "d:
GetLastError %d MojoPac Professional CISCO_PV M"

```

/NONAME [NTFS]/[unallocated space]/00000067/00041905", "11251", "5", "fessional
CISCO_PV MojoPac Registry backup MojoPac"
/NONAME [NTFS]/[unallocated space]/00000067/00041905", "11251", "6", "ac Registry
backup MojoPac Print Support OPSWAT JU"
/NONAME [NTFS]/[root]/\$MFT", "16", "1", "06482.pf RingCube MojoPac
(RingCube).lnk MojoPac "
/NONAME [NTFS]/[root]/\$MFT", "16", "2", "Pac (RingCube).lnk MojoPac Website.lnk
MOJOPA~1.LN"
/NONAME [NTFS]/[root]/\$MFT", "16", "3", "K MOJOPA~1.LNkgCu0 MojoPac
(RingCube).lnk Start.ex"
/NONAME [NTFS]/[root]/\$MFT", "16", "4", "o MOJOPA~2.LNKite0 MojoPac
Website.lnk http://www"
/NONAME [NTFS]/[root]/Documents and Settings/Administrator/Local
Settings/Temp/RingThreeInstallerHelper.dll", "100", "37", "://www.ringcube.com/mojopac/prod/ins/vdesk/index.ht"

Appendix X d) Findings in Host Registry - Baseline Test b) MojoPac



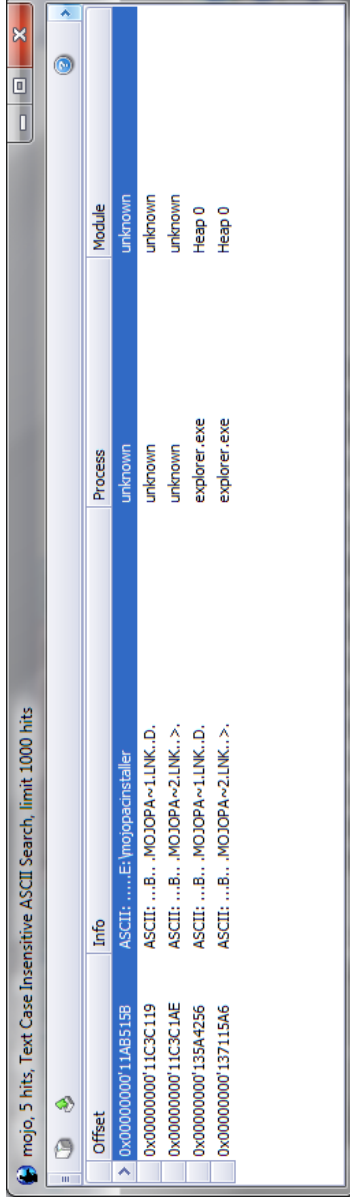
Appendix X e) Sample findings in Host RAM - Baseline Tests a) and b) MojoPac

i) Test a)

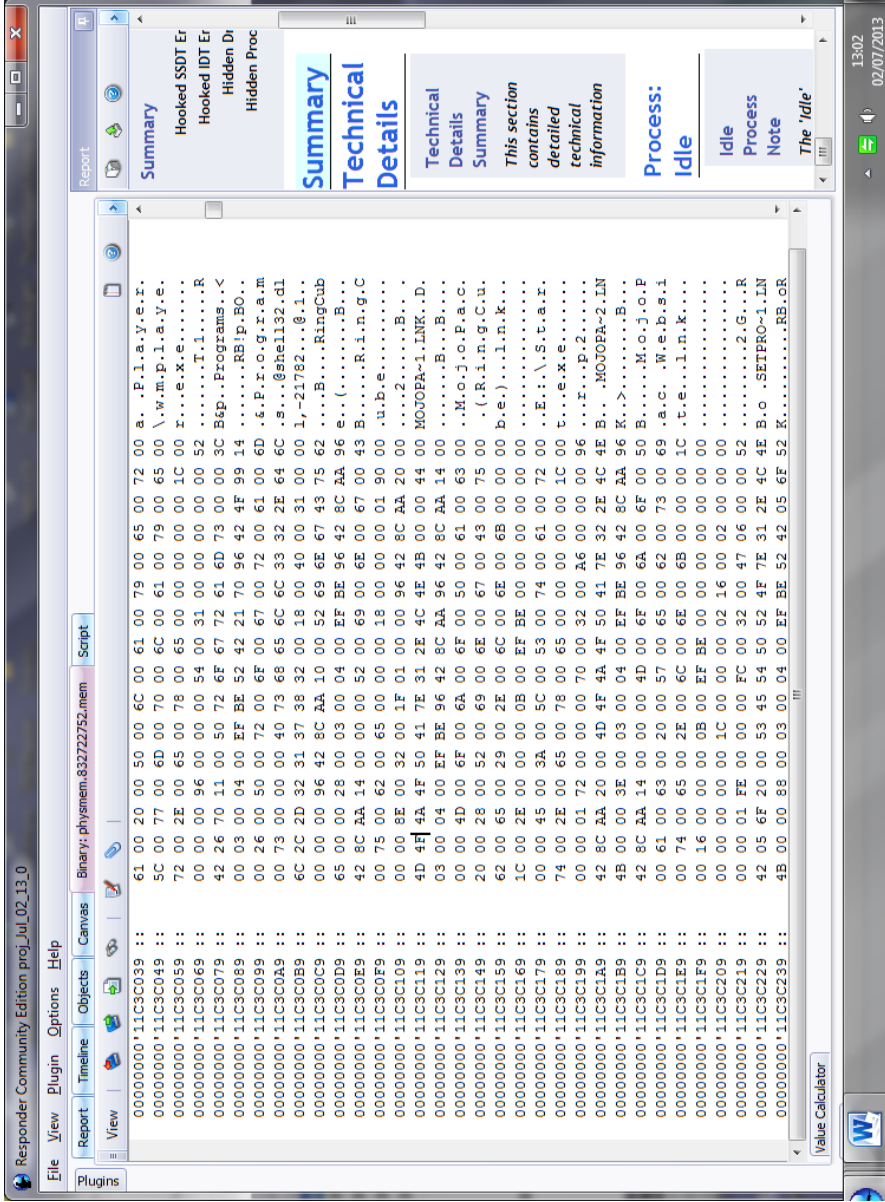
Offset	Info	Process	Module
> 0x00000000'62960000	ASCII:MOJO	unknown	unknown
0x00000000'62960000	ASCII: MOJO	unknown	unknown
0x00000000'62960060	ASCII: t...a..MOJOPA~IEXE .5B.	unknown	unknown
0x00000000'629E1016	ASCII: ion=Run MojoPac..icon=\\	unknown	unknown
0x00000000'629E1066	ASCII: ..label=MojoPac.....	unknown	unknown

Offset	Info	Process	Module
> 0x00000000'629E1034	ASCII: m Files RingThree\bin\RingOS.co...	unknown	unknown

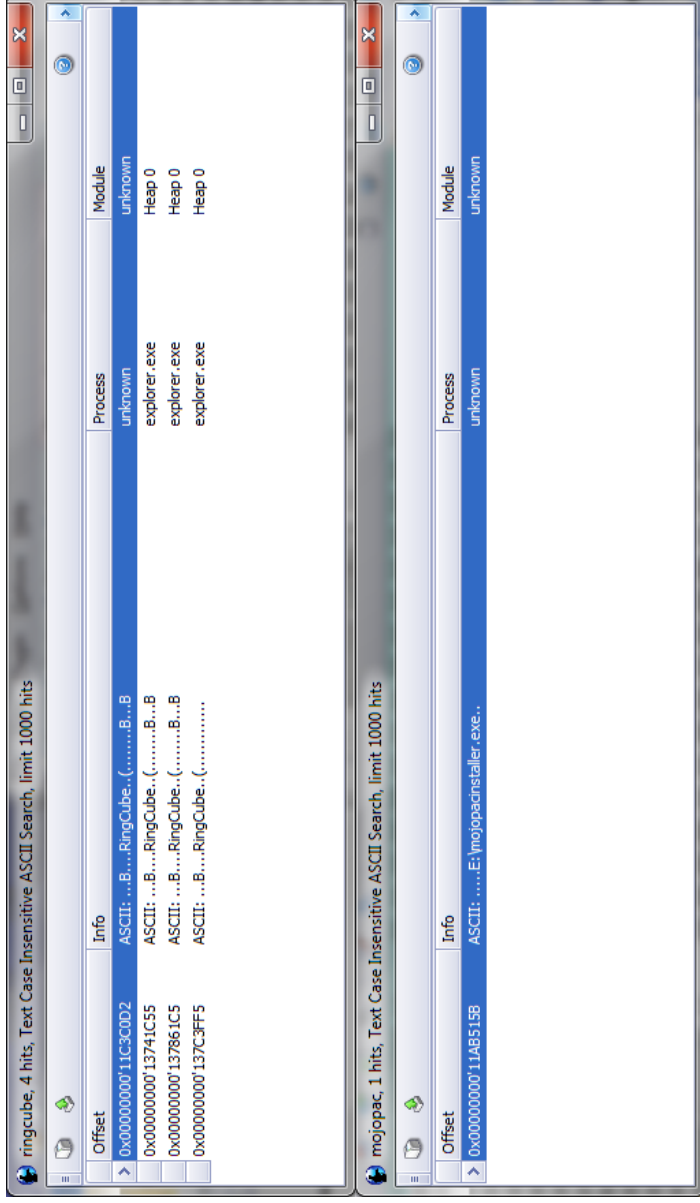
ii) Test b)



Textural record of vPC attachment event within captured RAM



Further results in HBGary report



Appendix X f) Findings in Host Pagefile - Baseline Tests b) MojoPac

C:\Documents and Settings\Administrator
 EGISTRY\USER\S-1-5-21-448539723-2139871995-1801674531-500_Classes\CLSID\{66742402-F9B9-11D1-A202-0000F81FEDEE}\InProcServer32
 :\mojopacinstaller.exe Config
 C:\Documents and Settings\Administrator\Local Settings

egistry\Machine\Software\Classes\Applications\mojopacinstaller.exe
shellex\ContextMenuHandlers\{a2a9545d-a0c2-42b4-9708-a0b2badd77c8}
EGISTRY\USER\S-1-5-21-448539723-2139871995-1801674531-500
ntsize
ntface
mojopacinstaller
ttomright

APPENDIX XI
Baseline test results for Ceedo Portable

Appendix XI a) Sample PM Outputs Baseline Test a) Ceedo

```

svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:95679 VolumeSerialNumber: 732D-
160E SupportsObjects: False VolumeLabel: CEEDO"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""VolumeCreationTime: 01/01/1601 00:00:3035" LastAccessTime:
28/08/2014 00:00:00 LastWriteTime: 05/08/2013 21:00:34 ChangeTime: 01/01/1601 00:00:00
svchost.exe 828 CreateFile E: SUCCESS Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-
Alert, Non-Directory File, Attributes: n/a, ShareMode: Read, Write, AllocationSize: n/a, Impersonating: NT AUTHORITY\SYSTEM, OpenResult:
Opened
svchost.exe 829 CreateFile E: SUCCESS Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-
Alert, Non-Directory File, Attributes: n/a, ShareMode: Read, Write, AllocationSize: n/a, Impersonating: NT AUTHORITY\SYSTEM, OpenResult:
Opened
svchost.exe 830 CreateFile E: SUCCESS Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-
Alert, Non-Directory File, Attributes: n/a, ShareMode: Read, Write, AllocationSize: n/a, Impersonating: NT AUTHORITY\SYSTEM, OpenResult:
Opened
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""VolumeCreationTime: 01/01/1601 00:00:3043" 1: .Trashes 2:
.Spotlight-V100 3: _cp_ga.exe 4: Ceedo 5: Autorun.inf 6: FoundMe.exe 7: My Documents 8: AutoDetect.exe 9:
StartCeedo.exe 10: Autorun.exe 11: Secrets.txt 12: Recycled"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""VolumeCreationTime: 01/01/1601 00:00:3046" 1: Ceedo 2:
CeedoPersonal.cab 3: ceedo.nvs 4: Common 5: Program Files 6: Program Files (x64) 7: User 8: Windows 9:
ceedo.cab 10: desktop.ini 11: ceedo.mrf 12: ~ceedo.mrf"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""VolumeCreationTime: 01/01/1601 00:00:3047"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""VolumeCreationTime: 01/01/1601 00:00:3048"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""VolumeCreationTime: 01/01/1601 00:00:3049" 1: Config.inf 2:
Ceedo_eng.dll 3: CeedoSettings.ini 4: SmartPlayer 5: ceedores.dll 6: BasPrefs.dll 7: Config 8: Downloads 9: Eula 10:

```

Languages	11: Themes	12: UserPictures	13: Utilities	14: Ceedo.exe	15: Localprg.ini	16: CeedoPLP.dll	17: Ceedolco.dll
	18: CeedoCMP.dll	19: AppManager.exe	20: Vsflex7N.ocx	21: CeedoSht.exe			
svchost.exe	860	QueryInformationVolume	E:\	SUCCESS	""VolumeCreationTime: 01/01/1601 00:00:3050"		
svchost.exe	860	QueryInformationVolume	E:\	SUCCESS	""VolumeCreationTime: 01/01/1601 00:00:3051"		
svchost.exe	860	QueryInformationVolume	E:\	SUCCESS	""VolumeCreationTime: 01/01/1601 00:00:3052"		

Appendix X b)

Sample findings in RAM - Baseline Test a) Ceedo

Offset	Info	Process	Module
> 0x0000000000000000	ASCII: L...g.CEEDON<-ICSV m,yE	unknown	unknown
0x0000000000000230A0A0	ASCII: d...l.CEEDO ~-IDL .X.v.B	unknown	unknown
0x0000000000000230A100	ASCII: n...g.s.CEEDOS~JINI d.v.B	unknown	unknown
0x0000000000000230A160	ASCII: .B.....CEEDORESULLI...q=	unknown	unknown
0x0000000000000230A3A0	ASCII:CEEDO EXE .B.v.B	unknown	unknown
0x0000000000000230A420	ASCII: l...l...CEEDOPLPOLL =v.B	unknown	unknown
0x0000000000000230A460	ASCII: l...l...CEEDOCODLL ...v.B	unknown	unknown
0x0000000000000230A4A0	ASCII: l...l...CEEDOCMPDLL ...v.B	unknown	unknown
0x0000000000000230A580	ASCII: x...e...CEEDOSHTEKE .3.v.B	unknown	unknown
0x00000000000002980060	ASCII:CEEDO ...v.B	unknown	unknown
0x000000000000029800C0	ASCII: n...a.l.CEEDOP~LCAB ...v.B	unknown	unknown
0x000000000000029800E0	ASCII: .B...Z...CEEDO NIS ...v.B	unknown	unknown
0x00000000000002980260	ASCII: .BV.....CEEDO CAB .s.v.B	unknown	unknown
0x000000000000029802E0	ASCII: .C.....CEEDO MRF# H(x.B	unknown	unknown
0x00000000000002980301	ASCII: E.(...~CEEDO MRF# Q(x.B.	unknown	unknown
0x000000000000049A95B3	ASCII: etlabel: CEEDO".11:30:02.	unknown	unknown
0x00000000000005329046	ASCII: !e",E:\Ceedo\Ceedo",SUCC	unknown	unknown
0x000000000000053290AC	ASCII: !e",E:\Ceedo\Ceedo",SUCCESS,"	unknown	unknown
0x0000000000000532914C	ASCII: !e",E:\Ceedo\Ceedo\Ceedo!	unknown	unknown
0x00000000000005329152	ASCII: !e",E:\Ceedo\Ceedo\Ceedo.dll	unknown	unknown
0x00000000000005329158	ASCII: o\Ceedo\Ceedo.dll",SUC	unknown	unknown
0x000000000000053291E	ASCII: !e",E:\Ceedo\Ceedo\Ceedo!	unknown	unknown

Appendix X c) Sample PM outputs - Baseline Test b) Ceedo

```

svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:53632" Length: 110
Data: ""G:\Program Files\Office\OFFICE11\WINWORD.EXE"" /w ""%1""
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:53633" Length: 120
Data: )|1^v^n-}f{ZXfeAR6.jjWORDFiles>P`os 1@SW=P7v6GPIjXh /w ""%1""
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:53634" Length: 130
Data: ""C:\Program Files\Microsoft Office\Office12\WINWORD.EXE""
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:53635" Length: 120
Data: vUpAV6!!!!!!MKKskWORDFiles>twf{~$4Q]c@5d1` xaTO5 /w ""%1""
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:53636" Length: 110
Data: ""G:\Program Files\Office\OFFICE11\WINWORD.EXE"" /w ""%1""

svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:45478" Length: 28
Data: G:\Ceedo\User"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:45479" Length: 28
Data: E:\Ceedo\User"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:45480" Length: 28
Data: E:\Ceedo\User"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:45481" Length: 28
Data: E:\Ceedo\User"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:45482" Length: 28
Data: E:\Ceedo\User"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:45483" Length: 28
Data: E:\Ceedo\User"
svchost.exe 860 QueryInformationVolume E:\ SUCCESS ""\VolumeCreationTime: 01/01/1601 00:00:45484" Length: 72

```

Data: G:\Ceedo\Program Files\Common Files"
 .4 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45485 Length: 84
 Data: G:\Ceedo\Program Files (x64)\Common Files"
 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45490 Length: 52
 Data: G:\Ceedo\Common\Documents"
 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45491 Length: 52
 Data: G:\Ceedo\Common\Favorites"
 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45492 Length: 72
 Data: G:\Ceedo\Common\Start Menu\Programs"
 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45493 Length: 54
 Data: G:\Ceedo\Common\Start Menu"
 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45494 Length: 88
 Data: G:\Ceedo\Common\Start Menu\Programs\Startup"
 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45495 Length: 52
 Data: G:\Ceedo\Common\Templates"
 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45496 Length: 70
 Data: G:\Ceedo\Common\Documents\My Music"
 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45497 Length: 76
 Data: G:\Ceedo\Common\Documents\My Pictures"
 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45498 Length: 72
 Data: G:\Ceedo\Common\Documents\My Videos"
 svchost.exe 860 QueryInformationVolume E:\ SUCCESS "VolumeCreationTime: 01/01/1601 00:00:45499 Length: 32
 Data: G:\My Documents"

Appendix X d)

Sample findings in RAM - Baseline Test b) Ceedo

Offset	Info	Process	Module
> 0x0000000000000000-KC08	ASCII:Fm!CEEDOR~Jm.T.a.Q...	unknown	unknown
0x0000000000000000-F9C0B	ASCII:E\Ceedo\Ceedo\SmartP	unknown	unknown
0x0000000000000000-F9CE1	ASCII: ;\Ceedo\Ceedo\SmartPlayer\	unknown	unknown
0x0000000000000000-110C38	ASCII:E\Ceedo\Ceedo\Ceedo	unknown	unknown
0x0000000000000000-1110C41	ASCII: ;\Ceedo\Ceedo\Ceedo.exe.v	unknown	unknown
0x0000000000000000-1110C47	ASCII: o\Ceedo\Ceedo\Ceedo.exe.v\!FF	unknown	unknown
0x0000000000000000-103DE768	ASCII:E\Yeedo\Ceedo\Ceedo	unknown	unknown
0x0000000000000000-103DE771	ASCII: \Yeedo\Ceedo\Ceedo.exe...	unknown	unknown
0x0000000000000000-103DE777	ASCII: o\Ceedo\Ceedo.exe....C.e.e	unknown	unknown
0x0000000000000000-10431483	ASCII:E\Ceedo\Ceedo\SmartP	unknown	unknown
0x0000000000000000-10431489	ASCII: ;\Ceedo\Ceedo\SmartPlayer\	unknown	unknown
0x0000000000000000-11354EBB	ASCII:B...CEEDOR~1.*.....	explorer.exe	Heap 0
0x0000000000000000-1335D600	ASCII:Fm!CEEDO2_MEMDUMP_MEM	unknown	unknown
0x0000000000000000-13402688	ASCII:Fm!CEEDO_RES..n.t...	unknown	unknown
0x0000000000000000-1376E236	ASCII:E\Yeedo\Ceedo\Ceedo	unknown	unknown
0x0000000000000000-1376E261	ASCII: \Yeedo\Ceedo\Ceedo.exe...	unknown	unknown
0x0000000000000000-1376E267	ASCII: o\Ceedo\Ceedo.exe....CM!a	unknown	unknown
0x0000000000000000-18FD5188	ASCII: ;.B...CEEDOR~1.*.....	explorer.exe	Heap 0
0x0000000000000000-18D152F8	ASCII:E\Ceedo.....J...>N:h	unknown	unknown
0x0000000000000000-1A172D28	ASCII:Fm!CEEDO2~1_MEM.....	unknown	unknown
0x0000000000000000-D940FC0	ASCII:CEEDOR~1.....B	unknown	unknown
0x0000000000000000-D9487718	ASCII:E\Ceedo\Ceedo\SmartP	unknown	unknown
0x0000000000000000-D9487721	ASCII: ;\Ceedo\Ceedo\SmartPlayer\	unknown	unknown
0x0000000000000000-2072ED63	ASCII:E\Ceedo.....(<....>N:h	unknown	unknown
0x0000000000000000-21E77FBA	ASCII: ;.4_Mem\cedo2_memdump.mem	FTK_Innager.exe	Unidentified
0x0000000000000000-22051E3A	ASCII: A..4_Mem\cedo2_memdump.mem	FTK_Innager.exe	Unidentified
0x0000000000000000-23022080	ASCII: d...um.CEEDO2~jMEM.....B	unknown	unknown

Comparison Sample: nappplay.exe shown in captured RAM file

The screenshot displays the Responder Community Edition interface. The main window shows a comparison of memory samples for the process nappplay.exe. The data is presented in a hex dump format, with columns representing memory addresses and rows representing the hex values. The interface includes a menu bar (File, View, Plugin, Options, Help), a toolbar, and a main window with a report view. On the right side, there is a 'Value Calculator' and a 'Log' button. The system tray shows the date and time as 13:19 on 02/07/2013.

Appendix X e)

Appendix X f)
Sample findings on Host HD - Baseline Test b) Ceedo

[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "1", "APAUTODETECTMUTEX \Ceedo\Ceedo\SmartPlayer\brand."
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "2", "DETECTMUTEX \Ceedo\Ceedo\SmartPlayer\brand.dll /"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "3", "igned correctly!!! Ceedo Repair Software\Microso"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "4", "WndClass Autorun \Ceedo Desktop %AppData%\Micr"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "5", " Launch \Windows \Ceedo\Windows explore StartC"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "6", "aperStyle Software\Ceedo\Ceedo HostWallpaper Co"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "7", "yle Software\Ceedo\Ceedo HostWallpaper Control"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "8", "er folder Removing Ceedo shortcuts /detect= /st"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "9", "pShortcut Software\Ceedo\Apps HKU\Current Ceedo"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "10", "\Apps HKU\Current Ceedo device removed \StartCe"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "11", " \StartCeedo.exe \Ceedo Restarting Ceedo NAPCE"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "12", "\Ceedo Restarting Ceedo NAPCEDOAUTOPLAY Ceedo"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "13", "itScreen Visible \Ceedo\Ceedo Shutdown Settings"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "14", "en Visible \Ceedo\Ceedo Shutdown Settings Soft"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "15", "le for %s (pid %d) Ceedo.exe Ceedo was ejected s"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "16", "pid %d) Ceedo.exe Ceedo was ejected successfully"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "17", "ily Error ejecting Ceedo Ejecting Ceedo Ejectin"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "18", "ing Ceedo Ejecting Ceedo Ejecting true crypt ima"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "19", " computer Software\Ceedo\Apps\Ceedo Software\Cee"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "20", "Software\Ceedo\Apps\Ceedo Software\Ceedo\Apps\Cee"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "21", "pps\Ceedo Software\Ceedo\Apps\Ceedo\%s Deleting"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "22", "Software\Ceedo\Apps\Ceedo\%s Deleting Ceedo regis"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "23", "\Ceedo\%s Deleting Ceedo registry ERROR: MRF in "
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "24", " Unable to wait for Ceedo termination, Virtual Reg"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "25", "\$#B2Q %CeedoDrive%\Ceedo\SmartPlayer\nappla"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "26", "%CeedoDrive%\Ceedo\SmartPlayer\napplay.spc"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "27", ".play.spc %AppData%\Ceedo\SmartPlay\SmartPlayer\na"
[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "326629", "28", ".y.spc %CeedoDrive%\Ceedo\User\Application Data\Cee"

[root]/Documents and Settings/Administrator/Local Settings/Temp/AutoDetect.exe", "29", "er\\Application Data\\Ceedo\\SmartPlay\\SmartPlayer\\na"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "1", "CEEDO.EXE \\DEVICE\\HARDDISK\\VOLUME1\\WINDOWS\\SYSTEM3"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "2", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\SMARTPLAYER\\NAPCOR"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "3", "K1\\DP(1)0-0+5\\CEEDO\\SMARTPLAYER\\NAPCORE.DLL "
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "4", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\CEEDO.EXE \\DEVICE\\"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "5", "K1\\DP(1)0-0+5\\CEEDO\\CEEDO.EXE \\DEVICE\\HARDDI"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "6", "1)0-0+5\\CEEDO\\CEEDO.EXE \\DEVICE\\HARDDISKVOLU"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "7", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\SMARTPLAYER\\CEEDOP"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "8", "K1\\DP(1)0-0+5\\CEEDO\\SMARTPLAYER\\CEEDOPST.DLL"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "9", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\CONFIG.INF \\DEVICE"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "10", "K1\\DP(1)0-0+5\\CEEDO\\CONFIG.INF \\DEVICE\\HARDD"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "11", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\SMARTPLAYER\\BRAND."
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "12", "K1\\DP(1)0-0+5\\CEEDO\\SMARTPLAYER\\BRAND.DLL \\D"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "13", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\CEEDO.NVS \\DEVICE\\HARDDI"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "14", "K1\\DP(1)0-0+5\\CEEDO\\CEEDO.NVS \\DEVICE\\HARDDISK1\\DP"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "15", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\CEEDOSETTINGS.INI "
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "16", "K1\\DP(1)0-0+5\\CEEDO\\CEEDOSETTINGS.INI \\DEVIC"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "17", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\USER\\LOCAL SETTINGS\\TEMP"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "18", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\USER\\LOCAL SETTINGS\\TEMP"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "19", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\USER\\LOCAL SETTINGS\\TEMP"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "20", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\USER\\LOCAL SETTINGS\\TEMP"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "21", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\USER\\LOCAL SETTINGS\\TEMP"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "22", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\USER\\LOCAL SETTINGS\\TEMP"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "23", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\USER\\LOCAL SETTINGS\\TEMP"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "24", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\USER\\COOKIES\\INDEX.DAT \\
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "25", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\USER\\LOCAL SETTINGS\\HIST"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "26", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\USER\\LOCAL SETTINGS\\HIST"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "27", "ARDDISK1\\DP(1)0-0+5\\CEEDO\\CEEDORES.DLL \\DEVI"

[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "28", "K1\DP(1)0-0+5\CEEDO\CEEDORES.DLL \DEVICE\HAR"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "29", "ARDDISK1\DP(1)0-0+5\CEEDO\CEEDO_ENG.DLL \DEV"
[root]/WINDOWS/Prefetch/CEEDO.EXE-1539FA3E.pf", "331249", "30", "K1\DP(1)0-0+5\CEEDO\CEEDO_ENG.DLL \DEVICE\HA"

/NONAME [NTFS][unallocated space]/00001103/00037476", "337450", "1", "shell\open\command\Ceedo\Program Files (x64)\Inte"

/NONAME [NTFS]/[unallocated space]/00001103/00037476", "337450", "2", "Shell\Open\Command\Ceedo\SmartPlayer\nappla"
/NONAME [NTFS]/[unallocated space]/00001103/00037476", "337450", "3", "Open\Command\Ceedo\SmartPlayer\napplay.exe "
/NONAME [NTFS]/[unallocated space]/00001103/00037476", "337450", "4", "n Enables to launch Ceedo SmartPlayer packages fro"
/NONAME [NTFS]/[unallocated space]/00001103/00037476", "337450", "5", "1 Enables to launch Ceedo SmartPlayer packages fro"
/NONAME [NTFS]/[unallocated space]/00001103/00037476", "337450", "6", "046}\LocalServer32 \Ceedo\Program Files (x64)\Inte"
/NONAME [NTFS]/[unallocated space]/00001103/00037476", "337450", "7", "1 Enables to launch Ceedo SmartPlayer packages fro"
/NONAME [NTFS]/[unallocated space]/00001103/00037476", "337450", "8", "33}\InprocServer32 \Ceedo\Ceedo\SmartPlayer\npceed"
/NONAME [NTFS]/[unallocated space]/00001103/00037476", "337450", "9", "procServer32 \Ceedo\Ceedo\SmartPlayer\npceedo.dll "
/NONAME [NTFS]/[root]/\$LogFile", "326547", "1", "D((1 Mb\ (1 Mb\ ceedo.cab ceedo.cab z41 Mb\R"
/NONAME [NTFS]/[root]/\$LogFile", "326547", "2", " (1 Mb\ ceedo.cab ceedo.cab z41 Mb\RCRD(ceedo"
/NONAME [NTFS]/[root]/\$LogFile", "326547", "3", ".cab z41 Mb\RCRD(ceedo.csd ceedo.csd c\RCRD("
/NONAME [NTFS]/[root]/\$LogFile", "326547", "4", "b\RCRD(ceedo.csd ceedo.csd c\RCRD(Install.in"
/NONAME [NTFS]/[root]/\$LogFile", "326547", "5", "=&d\RCRD(61 Me\ ceedo.csd Product Product "
/NONAME [NTFS]/[root]/\$LogFile", "326547", "6", "tect.exe %f\RCRD(ceedo.cab Install.ini g\FILE"
/NONAME [NTFS]/[root]/\$LogFile", "326547", "7", "N.EXE-0F809CB4.pf CEEDO.EXE-1539FA3E.pf @4 Mn" "
/NONAME [NTFS]/[root]/\$LogFile", "326547", "8", "CRD(CEEDOE~1.PF0 CEEDO.EXE-1539FA3E.pf @4 Mn" "
/NONAME [NTFS]/[unallocated space]/00001103/00038912", "337481", "1", "6A)\InprocServer32 \Ceedo\Ceedo\SmartPlayer\nappla"
/NONAME [NTFS]/[unallocated space]/00001103/00038912", "337481", "2", "procServer32 \Ceedo\Ceedo\SmartPlayer\napplay.dll "
/NONAME [NTFS]/[unallocated space]/00001103/00038912", "337481", "3", "t FriendlyTypeName \ceedo\Windows\SysWOW64\ieframe"
/NONAME [NTFS]/[unallocated space]/00001103/00038912", "337481", "4", "s FriendlyTypeName \ceedo\Windows\System32\ieframe"
/NONAME [NTFS]/[unallocated space]/00001103/00089478", "337794", "1", "ceedo.csd Install.ini PKYY/ 'K6 _ \ HY6ägÜj~RqH"
/NONAME [NTFS]/[unallocated space]/00001103/00089478", "337794", "2", "I1 Rosh Haayin1&0\$ CEEDO TECHNOLOGIES (2005) LTD1"
/NONAME [NTFS]/[unallocated space]/00001103/00089478", "337794", "3", "ION DEVELOPMENT1&0\$ CEEDO TECHNOLOGIES (2005) LTD0"

APPENDIX XII
Sample Baseline test results for
LupoPenSuite & PortableApps

Appendix XII a) Sample PM Outputs Baseline Test b) LupoPenSuite

rundll32.exe	RegCloseKey	HKLMSOFTWARE\Microsoft\Windows NT\CurrentVersion\IMM
rundll32.exe	QueryOpen	C:\WINDOWS\system32\MSCTFIME.IME
Explorer.EXE	QueryDirectory	E:\Lupo_PenSuite_v2013.04_Lite\Apps\7-Zip Portable\App
Explorer.EXE	QueryDirectory	E:\Lupo_PenSuite_v2013.04_Lite\Apps\7-Zip Portable\App
Explorer.EXE	CloseFile	E:\Lupo_PenSuite_v2013.04_Lite\Apps\7-Zip Portable\App
rundll32.exe	CreateFile	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	CreateFileMapping	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	QueryStandardInformationFile	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	CreateFileMapping	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	CloseFile	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	QueryOpen	C:\WINDOWS\system32\MSCTFIME.IME
Explorer.EXE	CreateFile	E:\Lupo_PenSuite_v2013.04_Lite\Apps\7-Zip Portable\Data
rundll32.exe	CreateFile	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	CreateFileMapping	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	QueryStandardInformationFile	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	CreateFileMapping	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	CloseFile	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	QueryOpen	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	CreateFile	C:\WINDOWS\system32\MSCTFIME.IME
rundll32.exe	CreateFileMapping	C:\WINDOWS\system32\MSCTFIME.IME
Explorer.EXE	QueryDirectory	E:\Lupo_PenSuite_v2013.04_Lite\Apps\7-Zip Portable\Data*
Explorer.EXE	ReadFile	E:\Lupo_PenSuite_v2013.04_Lite\Apps\7-Zip Portable\Data

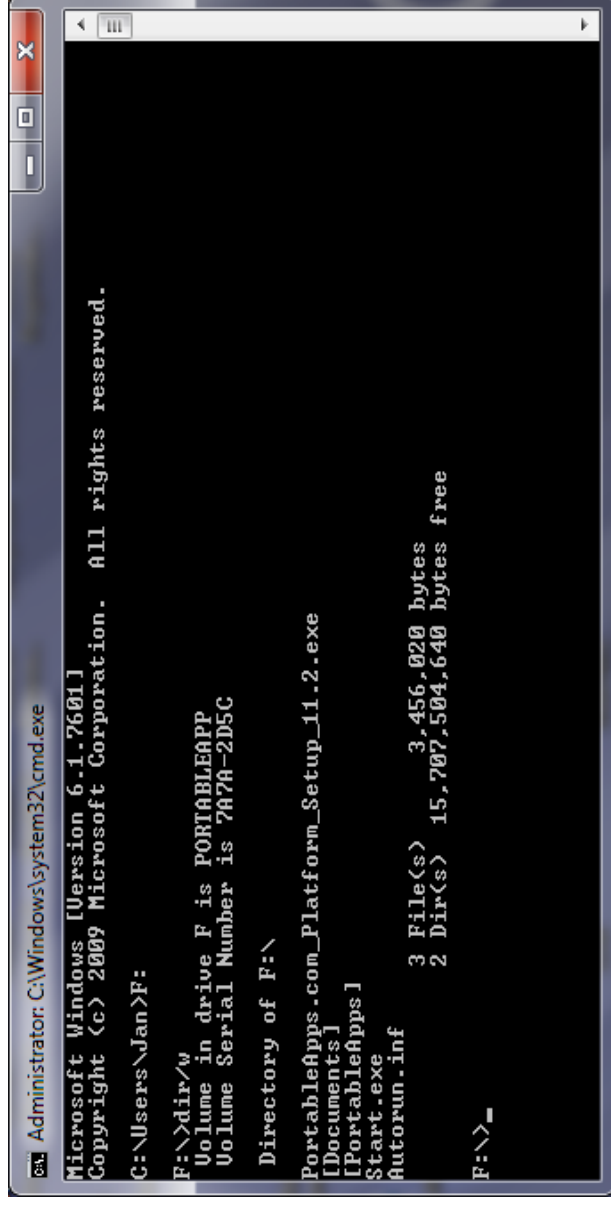
rundll32.exe	Thread Exit	C:\WINDOWS\system32
rundll32.exe	Process Exit	C:\WINDOWS\WinSxS\x86_Microsoft_Windows.Common-Controls_6595b64144ccf1df_6.0.2600.6028_x-ww_61e65202
rundll32.exe	CloseFile	E:\Lupo_PenSuite_v2013.04_Lite\Apps\CCleaner
rundll32.exe	CloseFile	E:\Lupo_PenSuite_v2013.04_Lite\Apps\CCleaner*
Explorer.EXE	CreateFile	E:\Lupo_PenSuite_v2013.04_Lite\Apps\CCleaner
Explorer.EXE	QueryDirectory	E:\Lupo_PenSuite_v2013.04_Lite\Apps\CCleaner
Explorer.EXE	ReadFile	E:\Lupo_PenSuite_v2013.04_Lite\Apps\CCleaner
Explorer.EXE	QueryDirectory	E:\Lupo_PenSuite_v2013.04_Lite\Apps\CCleaner
Explorer.EXE	QueryDirectory	E:\Lupo_PenSuite_v2013.04_Lite\Apps\CCleaner
Explorer.EXE	CloseFile	E:\Lupo_PenSuite_v2013.04_Lite\Apps\CCleaner
svchost.exe	CreateFile	C:\WINDOWS\Prefetch\RUNDLL32.EXE-451FC2C0.pf
Explorer.EXE	CreateFile	E:\Lupo_PenSuite_v2013.04_Lite\Apps\CCleaner\lang
svchost.exe	QueryStandardInformationFile	C:\WINDOWS\Prefetch\RUNDLL32.EXE-451FC2C0.pf
svchost.exe	CreateFileMapping	C:\WINDOWS\Prefetch\RUNDLL32.EXE-451FC2C0.pf
svchost.exe	QueryStandardInformationFile	C:\WINDOWS\Prefetch\RUNDLL32.EXE-451FC2C0.pf

Appendix XII b) Sample PM Outputs Baseline Test b) PortableApps

Explorer.EXE	CreateFile	E:\PortableApps\PortableApps.com\App\Graphics\usb.ico
Explorer.EXE	CreateFileMapping	C:\WINDOWS\system32\shell32.dll
Explorer.EXE	QueryStandardInformationFile	C:\WINDOWS\system32\shell32.dll
Explorer.EXE	CreateFileMapping	C:\WINDOWS\system32\shell32.dll
Explorer.EXE	CloseFile	C:\WINDOWS\system32\shell32.dll
Explorer.EXE	QueryOpen	C:\Documents and Settings\Jan\shell32.dll
Explorer.EXE	QueryOpen	C:\WINDOWS\shell32.dll
Explorer.EXE	QueryOpen	C:\Documents and Settings\Jan\shell32.dll
Explorer.EXE	QueryOpen	C:\WINDOWS\system32\shell32.dll
Explorer.EXE	SetEndOfFileInformationFile	C:\Documents and Settings\Jan\ntuser.dat.LOG
Explorer.EXE	RegCloseKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{72e73510-7168-11e4-ab8b-a6506f279d18}_Autorun
Explorer.EXE	RegCloseKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{72e73510-7168-11e4-ab8b-a6506f279d18}
Explorer.EXE	RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{72e73510-7168-11e4-ab8b-a6506f279d18}_Autorun\DefaultIcon\{Default}
Explorer.EXE	ReadFile	E:\Autorun.inf
Explorer.EXE	UnlockFileSingle	E:\Autorun.inf
Explorer.EXE	CloseFile	E:\Autorun.inf
Explorer.EXE	RegCreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{72e73510-7168-11e4-ab8b-a6506f279d18}
Explorer.EXE	RegCreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{72e73510-7168-11e4-ab8b-a6506f279d18}_Autorun\DefaultLabel
Explorer.EXE	CreateFile	E:\Start.exe
Explorer.EXE	CreateFileMapping	E:\Start.exe
Explorer.EXE	QueryStandardInformationFile	E:\Start.exe
Explorer.EXE	CreateFileMapping	E:\Start.exe
Explorer.EXE	CreateFileMapping	E:\Start.exe
Explorer.EXE	RegOpenKey	HKLM\System\CurrentControlSet\Control\Session Manager\AppCertDlls
Explorer.EXE	RegOpenKey	HKLM\System\CurrentControlSet\Control\Session Manager\AppCompatibility
Explorer.EXE	RegQueryValue	HKLM\System\CurrentControlSet\Control\Session Manager\AppCompatibility\DisableAppCompat
Explorer.EXE	RegCloseKey	HKLM\System\CurrentControlSet\Control\Session Manager\AppCompatibility

Explorer.EXE	ReadFile	C:\WINDOWS\system32\kernel32.dll
Explorer.EXE	QueryOpen	C:\WINDOWS\system32\apphelp.dll
Explorer.EXE	CreateFile	C:\WINDOWS\AppPatch\sysmain.sdb
Start.exe	CloseFile	E:\PortableApps\PortableApps.com\PortableAppsPlatform.exe
Start.exe	FileSystemControl	E:\
Start.exe	FileSystemControl	E:\
Start.exe	CreateFile	E:\PortableApps\PortableApps.com\PortableAppsPlatform.exe
Start.exe	CreateFileMapping	E:\PortableApps\PortableApps.com\PortableAppsPlatform.exe
Start.exe	QueryStandardInformationFile	E:\PortableApps\PortableApps.com\PortableAppsPlatform.exe
Start.exe	CreateFileMapping	E:\PortableApps\PortableApps.com\PortableAppsPlatform.exe
Start.exe	CloseFile	E:\PortableApps\PortableApps.com\PortableAppsPlatform.exe
Start.exe	FileSystemControl	E:\
Start.exe	FileSystemControl	E:\
Start.exe	CreateFile	E:\PortableApps\PortableApps.com\PortableAppsPlatform.exe
Start.exe	QueryBasicInformationFile	E:\PortableApps\PortableApps.com\PortableAppsPlatform.exe
Start.exe	CloseFile	E:\PortableApps\PortableApps.com\PortableAppsPlatform.exe

Appendix XII c) Programs associated with PortableApps



```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Jan>F:
F:\>dir/w
Volume in drive F is PORTABLEAPP
Volume Serial Number is 7A7A-2D5C

Directory of F:\

Portableapps.com_Platform_Setup_11.2.exe
[Documents]
[Portableapps]
Start.exe
Autorun.inf          3 File(s)    3,456,020 bytes
                    2 Dir(s)    15,707,504,640 bytes free

F:\>_
```