Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

State of the art concerning memory acquisition software

A detailed examination of DOS and non-Windows NT memory acquisition

R. Carbone
Certified Hacking Forensic Investigator (EC-Council CHFI)
DRDC Valcartier

Defence R&D Canada – Valcartier
Technical Memorandum
DRDC Valcartier TM 2011-215
October 2011

State of the art concerning memory acquisition software

A detailed examination of DOS and non-Windows NT memory acquisition

R. Carbone
Certified Hacking Forensic Investigator (EC-Council CHFI)
DRDC Valcartier

Defence R&D Canada – Valcartier

Technical Memorandum
DRDC Valcartier TM 2011-215
October 2011

Principal Author

Richard Carbone
Programmer/Analyst

Approved by

Guy Turcotte
Head/System of Systems

Approved for release by

Christian Carrier
Chief Scientist

Abstract

This technical memorandum examines one specific software tool which can be used to carry out a forensic memory acquisition of DOS and Windows 9x systems. This work appears to be the first of its kind as no other comparable work can be found in the publicly available literature. Although DOS and Windows 9x systems are harder to come by today, this should not preclude that investigators may encounter them in the course of their work. By addressing the important issue of DOS and Windows 9x memory acquisition it will be possible for investigators to corroborate disk-based evidence when examining such systems used to commit illicit activities.

Résumé

Ce mémorandum technique décrit un outil logiciel spécifique qui peut être utilisé pour procéder à une acquisition de mémoire inforensique de systèmes DOS et Windows 9x. Cette étude semble être la première du genre puisqu'aucun ouvrage/recherche comparable ne se trouve dans la littérature publique. Bien que les systèmes DOS et Windows 9x ne soient pas très présents de nos jours, il est quand même possible qu'un enquêteur les rencontre dans son travail. En abordant ce problème important de l'acquisition de mémoire DOS et Windows 9x, il sera possible pour les enquêteurs de rassembler les preuves corroborantes du disque lorsqu'ils examineront ces systèmes qui peuvent encore être utilisés aujourd'hui pour commettre des actes illicites.

This page intentionally left blank.

Executive summary

State of the art concerning memory acquisition software: A detailed examination of DOS and non-Windows NT memory acquisition

Carbone, R.; DRDC Valcartier TM 2011-215; Defence R&D Canada – Valcartier; October 2011.

This technical memorandum is the first in a set of three. This memorandum's objective is to examine the technical aspects of Windows 9x operating systems memory acquisition. These systems include MS-DOS and FreeDOS, Windows 95C and Windows 98 SE, all of which are in fact DOS-based operating systems. The second memorandum examines the acquisition of memory under UNIX including Linux (PC-only), Solaris and BSD (OpenBSD, FreeBSD and NetBSD; all PC-based). The final memorandum will focus exclusively on memory acquisition from NT-based 32 and 64-bit operating systems ranging from Windows NT up to Windows 7 and Windows Server 2008 R2.

Although much effort has been made these last couple of years to acquire memory from modern Windows systems, and to a lesser extent from Linux, virtually no effort has been made to consolidate information concerning tools or techniques for acquiring memory from older PC operating systems.

Based on the work carried out herein, the reader should be able to successfully use the same tool and techniques implemented by the author to acquire the memory of the aforementioned systems. In the end, only one software tool, *Memdump,* by APSoft was found to be the only workable solution for these older DOS-based systems. However, in conducting memory acquisition experiments against the aforementioned operating systems important memory-specific limitations were discovered for both the amount of memory supported by these systems and the amount of memory which could be acquired.

Hopefully this memorandum may be of benefit to those who have to acquire the memory of those older systems as part of their forensic acquisition routine.

This work was carried out over a period of two months as part of the Live Computer Forensics project, an agreement between DRDC Valcartier and the RCMP (SRE-09-015, 31XF20). The results of this project will be of great interest to the Canadian Forces Network Operation (CFNOC) in their mission of securing DND networks and investigating computer incidents.

Sommaire

State of the art concerning memory acquisition software: A detailed examination of DOS and non-Windows NT memory acquisition

Carbone, R. ; DRDC Valcartier TM 2011-215 ; R & D pour la défense Canada – Valcartier; octobre 2011.

Ce mémorandum technique est le premier d'une série de trois. L'objectif de ce mémorandum est d'examiner les aspects techniques de l'acquisition de mémoire des systèmes Windows 9x. Ces systèmes incluent MS-DOS, FreeDOS, Windows 95C et Windows 98 SE, qui sont tous en fait des systèmes d'exploitation basés sur DOS. Le second mémorandum étudie l'acquisition de mémoire sous UNIX, incluant Linus (PC seulement), Solaris et BSD (OpenBSD, FreeBSD et NetBSD; tous basés sur un PC). Le mémorandum final se concentrera exclusivement sur l'acquisition de mémoire des systèmes d'exploitation basés sur NT 32 et 64 bits, allant de Windows NT jusqu'à Windows 7 et Windows Server 2008 R2.

Bien que beaucoup d'efforts aient été consacrés ces dernières années pour acquérir la mémoire des systèmes Windows modernes, et un moindre effort pour Linux, virtuellement aucunes énergies n'ont été consacrées à la consolidation de l'information concernant les outils ou techniques relatif à des vieux systèmes d'exploitation pour le PC.

Basé sur le travail décrit dans ce document, le lecteur devrait pouvoir utiliser avec succès les mêmes outils et techniques implémentés par l'auteur pour acquérir la mémoire de ces systèmes. En bout de ligne, un seul outil logiciel, *MemDump* par APSoft, a été découvert comme étant la seule solution pragmatique pour ces vieux systèmes basés sur DOS. Malgré tout, en effectuant ces expériences sur ces plus vieux systèmes d'exploitation, des limites importantes concernant la mémoire ont été trouvées tant pour la quantité de mémoire supportée par ces systèmes que pour la quantité de mémoire pouvant être capturée.

Il est à espérer que ce mémorandum puisse être utile à ceux qui ont à acquérir la mémoire de ces plus vieux systèmes dans le cadre de leur routine d'inforensique.

Ce travail a été accompli sur une période de deux mois dans le cadre du projet « Live Computer Forensics », une entente entre RDDC Valcartier et la GRC (SRE-09-015, 31XF20). Les résultats de ce projet seront d'un grand intérêt pour le Centre d'opérations des réseaux des Forces canadiennes (CORFC) dans leur mission de protection des réseaux du MDN et d'investigation des incidents informatiques.

Table of contents

List of tables

Acknowledgements

The authors would like to thank Mr. Yves van Chestein for peer-reviewing this document and providing many useful comments and insight in order to improve it. The author would also like to thank Mr. Martin Salois for translating various portions of the text.

Disclaimer

The reader should neither construe nor interpret the work described herein by the author as an endorsement of the aforementioned techniques and capacities as suitable for any specific purpose, construed, implied or otherwise.

Furthermore, the aforementioned author of this technical memorandum absolves himself in all ways conceivable with respect to how the reader may use, interpret or construe this technical memorandum. The author assumes absolutely no liability or responsibility, implied or explicit. Moreover, the onus is on the reader to be properly equipped and knowledgeable in the application of digital forensics.

Finally, the author, the Government of Canada, the Minister of National Defence (Canada), the Department of National Defence (Canada) and Defence Research and Development Canada are henceforth absolved of all wrongdoing, whether intentional, unintentional, construed or misunderstood on the part of the reader. If the reader does not agree to these terms then this technical memorandum should be readily returned to the Department of National Defence (Canada). Only if the reader agrees to these terms should he or she continue in reading it beyond this point. It is further assumed by all participants that if the reader has not read said Disclaimer upon reading this technical memorandum and has acted upon its contents, then the reader assumes all responsibility for any repercussions which may result from the information and data contained herein.

Requirements, assumptions, and exclusions

It is assumed that the reader is altogether familiar with digital forensics and the various techniques and methodologies associated therein. This technical memorandum is not an introduction to digital forensics, its techniques or methodologies. However, this technical memorandum will endeavour to present an adequate technically-oriented background to enable the reader to carry out and implement the work and analysis conducted herein.

The present technical memorandum only examines DOS and Windows 9x systems. Other ensuing technical memorandums will examine the acquisition of Linux, UNIX and NT-based operating systems.

The work presented herein has been done primarily using a Linux-based operating system while secondary result validation was carried out using a Windows 7 operating system. As such, regardless of the set-up the reader should arrive at the same overall results as those presented herein, assuming that the guest operating systems are the same and that the same virtualisation technology is used.

The primary Linux system ran Fedora Core 14 64-bit installed atop a Dell Precision 690 workstation with dual-core Xeon processors (with HyperThreading) for a total of 8 logical cores in conjunction with 22 GiB RAM and almost 20 TB of disk storage (see Annex A.1.1 for more details). The validation system was a Windows 7 64-bit system running atop a Dell XPS i7 8-core system with 18 GiB and 1.5 TB disk storage (see Annex A.1.2 for more details).

All DOS-based operating systems were tested under Oracle VirtualBox 4.0.6 (Linux and Windows version) with the appropriate VirtualBox Extension Pack installed. Both Windows 9x systems were tested under VMware Workstation 7.1.4 due to the inability of VirtualBox to adequately support these systems.

Although it was desired to examine and test Windows ME and DOS 8.0 it was not possible as no valid media or serial number could be located.

It is important to highlight that should a memory acquisition of a DOS or Windows 9x system fail, it may be possible acquire that system's memory using the cold boot attack [1] since the technique is not affected by the underlying operating system. However, the success of the cold boot attack is at most limited [2] and as such the investigator must be realistic in his expectations for acquiring memory using this technique.

Forensically capturing a memory acquisition

Physically capturing a memory acquisition under DOS will likely be challenging under the best circumstances and outright impossible in more difficult situations. Consider that in order to avoid contaminating a DOS machine, a CD, DVD or floppy disk can be used to introduce the *Memdump* program to the operating system from where it can be executed. However, storing a memory acquisition against these same devices is not possible. As such, memory acquisitions must either be stored on a remote network drive if one is available (for example, consider that the DOS machine may be connected to a Novell, Windows or other type of network and have a mapped network drive) or to locally attached storage including ZIP, LS-120, JAZ, or other readily removable mass storage device. Of course, this requires that memory acquisition be carried out piecemeal by acquiring only specific chunks of memory at any one time and storing it to a specific media before replacing said media and continuing with the acquisition by acquiring another subsequent chunk of memory. But since USB mass storage support is almost non-existent under DOS the only other option would be to dump memory to a physically connected and accessible hard disk drive partition. Doing so, however, will guarantee that a given memory acquisition will overwrite all previous data and potential evidence therein. Therefore, the computer forensic investigator must closely weigh his options and if necessary decide whether or not a DOS memory acquisition is important enough to acquire prior to jeopardizing potential evidence on a given hard disk drive.

Fortunately, many USB mass storage devices can be connected to Windows 95 and 98 systems while they are running, although specific support will vary by operating system, mass storage device (and its device drivers, if appropriate) and potential reboots required to access said device. In other cases, memory acquisition can be carried out by saving it to a remote network drive assuming that the underlying system is connected to a file-serving remote computer system which can be mapped as a local drive. It is unlikely that tools such as *Netcat* will be able to successfully stream a memory dump over the network since *Memdump* requires providing the name and location of a physical file for saving a given memory dump. In all other cases, memory dumps will have to be stored to a locally attached device be it removable (i.e. ZIP, LS-120, JAZ, etc.) or a physically accessible hard drive disk partition. However, storing a memory acquisition to a hard disk drive partition again raises the problem of crushing over potential evidence with said memory dump. As such, if removable media is available then this should instead be used. Where no better solution is available then the memory acquisition will have to be saved to a physically available hard disk drive partition. Unfortunately, attempting to save a memory acquisition as it is being produced to optical media will not work as disc burning software requires fixed data files to burn.

Computer memory volatility

This technical memorandum has been written for the computer forensic investigator who may have to perform a DOS or Windows 95/98 memory acquisition at one time or another in the function of his or her duties. However, since little public information or literature is currently available to the computer forensic community the author has made this information available to the reader. Moreover, this technical memorandum brings together important information for the reader by presenting this material in a comprehensible format.

This technical memorandum is not, however, an examination of computer memory analysis. This specific vein of research is outside the scope of this research and warrants an altogether separate technical discussion to sufficiently examine the subject matter.

It is important to consider the volatility of computer memory when attempting to acquire its memory. It does not matter if the computer system is running DOS, UNIX, Windows, or any other operating system. The fact that an individual, in this case a computer forensic investigator, runs a memory acquisition program atop the computer system changes the state of the underlying system. This is a universal principle, commonly known as the Observer Effect, which cannot be undone and follows through for all cases where a physical observation is made of a system.

In the case of computer memory acquisition, in order to obtain a copy of the system's memory, the investigator must interact with the system (in order to observe it) and then run some program, command, or utility in order to acquire said memory. This process irreversibly changes the running state of the computer system and as such, certain bytes of information which may contain evidence would be permanently lost. However, it is logical to conclude that the more memory a given system has the less likely this is to occur as evidence is more apt to be spread out across said memory. Thus, no matter the care and consistent steps followed through by the investigator, some data will be lost with no way of discerning what it was.

1 Introduction

1.1 Objective

The objective of this technical memorandum is to determine which memory acquisition software tools work under MS-DOS, FreeDOS, Windows 95C and Windows 98 SE. Moreover, it will gauge the amount of memory that each operating system supports including how much of that memory can be acquired. Any discovered limitations or caveats will be presented.

1.2 Background information

In this section, the following operating systems' technical and historical details are briefly examined including FreeDOS (version 1.0), MS-DOS (versions 6.22 and 7.1), Windows 95C and 98 SE.

1.2.1 Early PC computer memory and architecture as it relates to DOS

Although ubiquitous throughout the 1980s and much of the 1990s, DOS has been superseded due to its inherent limitations. Chief among its technical limitations was that DOS was a Real-Mode[1] operating system, due entirely to the nature of the 8086 processor. Although this processor could support more than the standard DOS memory limitation of 640 KiB memory the lack of foresight by hardware designers at the time considered 640 KiB to be more than enough memory that would ever be needed by DOS users.

Even though it was possible to purchase systems at that time equipped with more than 640 KiB memory, it was nonetheless prohibitively expensive. In time, PC computer systems required more memory in order to keep up with the needs of popular memory-hungry software which was already pushing the hardware to its limits.

Although the 8088[2] processor could support upwards of 1 MiB memory most systems did not come with this much. However, some 8086[3] systems shipped with as much as 1 MiB memory. Both the 8088 and 8086 processor designs made memory beyond the first 640 KiB essentially invisible to the operating system and user-based software due to design limitations in the hardware which required this space for specific hardware addressing features (e.g. reserved video memory and ROM BIOS). This 640 KiB limitation became better known as Conventional Memory.

Although the 80286 architecture offered some Protected-Mode features and could support upwards of 16 MiB RAM[4], these features were not widely exploited by the various DOS

[1] Real-Mode (or equivalent depending on the processor – 8088 and 8086 run in a mode very similar to Real-Mode) operating systems provide no multi-tasking, memory sharing or memory protection, unlike Protected-Mode (286 and 386) which do offer these capabilities.

[2] The 8088 architecture supported 20-bit addressing.

[3] The 8086 architecture also supported 20-bit addressing.

[4] The 80286 architecture supported 24-bit addressing.

operating systems of the time. Moreover, DOS systems at the time perpetuated Real-Mode functionality and did nothing to change the use or amount of reserved memory atop Conventional Memory.

It was only with the advent of the 80386[5] processor that improved Protected-Mode features were added to the architecture, at which time DOS finally began supporting not only specific Protected-Mode features but actually enabled the use of memory over the first 1 MiB of RAM.

Some of the limitations which remained for years in the DOS operating system are in part due to the fact that the 8088 processor, meant as a low cost alternative to the 8086 processor, introduced many hardware restrictions. The 8086 processor was developed before the 8088 processor by Intel and released in 1978. The 8088 processor, released July 1979, was targeted as a cheaper replacement for the more expensive 8086 and it became very popular with computer users and enthusiasts due primarily to its affordability. However, the 8088's limitations required that DOS at that time be specifically tuned to support its hardware requirements and limitations which remained omnipresent for many years. In fact, the very first version of MS-DOS which Microsoft provided to IBM for its personal computer line ran atop an 8088 processor.

Prior to the introduction of 80386 architecture expanded computer memory hardware add-on cards had being introduced into the marketplace allowing 80286 computers users to take advantage of the 80286 additional memory addressing capabilities. The use of these add-on cards required specialized DOS device drivers which became known as Expanded Memory Managers (EMM). These device drivers were typically provided by the hardware device's manufacturer but eventually a standard came to be set up for EMM and was standardized as the LIM EMS 4.0 standard. However, due to the very high cost of these expanded memory boards their use did not really catch on with the average computer user of the time.

With the introduction of the 80386 computer systems which were coming equipped with more than the typical 1 MiB memory and thanks to the underlying system's improved hardware memory addressing scheme, users could take advantage of more memory than before without the need for costly expanded memory add-on cards. The 80386's design allowed for the emulation of expanded memory over 1 MiB, which spawned the need for EMM emulation software, most notably EMM386.EXE from Microsoft and QEMM from Quarterdeck. Both allowed the user to emulate expanded memory in the range over 1 MiB so that it could be used by EMM-capable[6] software.

However, as some 80286 and 80386 computers were being sold with more than 1 MiB memory, which was found in memory slots rather than as add-on expansion boards, a new form of memory management was needed in order to access and use this increased memory capacity. Extended Memory Managers (XMM) allowed the user to use memory beyond the first 1 MiB RAM. However, XMM management required the use of Protected-Mode and as such programs wanting to use the memory above the first 1 MiB RAM had to support Protected-Mode execution. Under DOS, XMM management was accomplished using the HIMEM.SYS device driver. Several

[5] The 80386 supports 32-bit addressing, as do modern operating systems.

[6] DOS games at the time were often EMM-capable and expected at least several megabytes of expanded memory in order to run.

XMM standards were devised, the most notable of which being XMS 2.0 which supported 64 MiB RAM and XMS 4.0, which supported a maximum of 4 GiB RAM.

Through the use of various techniques which employ EMM, XMM or both, it is possible to access both the memory area between 640 KiB and 1 MiB and the area just above the first 1 MiB to load device drivers and TSRs at boot time. The former memory area is commonly referred to as the Upper Memory Area while the latter is referred to as the High Memory Area. The loading of device drivers and TSRs was carried out using either the *CONFIG.SYS* or *AUTOEXEC.BAT* boot-up configuration files or both.

The final memory management solution to examine is DOS extender technology. This technology made it possible to run supported software in Protected-Mode under both 80286 and 80386 systems. Under 80286-based systems, memory access was limited to 16-bit addressing while on the 80386, 32-bit addressing could be taken advantage of. Complex business software regularly took advantage of DOS extender technology but it was DOS-based games such as *DOOM* and *DOOM2* that brought the technology to the masses.

1.2.2 DOS background

This section briefly examines both FreeDOS and some of the different MS-DOS operating systems throughout the years.

1.2.2.1 FreeDOS background

FreeDOS is a DOS compatible operating systems for x86 computers. It will run on 32 and 64-bit PCs. It is a DOS system developed after the fall of DOS as the dominant player in PC operating systems brought on by the switch to full Protected-Mode 32-bit GUI-driven Windows-based operating systems, beginning with Windows NT. As such, the history of FreeDOS is considerably different from that of MS-DOS and other third-party DOS operating systems including PC-DOS and DR-DOS, which are in little use today. FreeDOS needs to be examined not only because it is open source and therefore freely available to everyone but also due to the fact that it bundles many additional third-party software tools, utilities and games, which make it an ideal DOS replacement.

The FreeDOS project was started by Jim Hall in June 1994. He was joined shortly thereafter by Pat Villani and Tim Norman. The latest release of FreeDOS is version 1.0, released September 2006. However, there are occasional bug fixes and enhancements for specifically bundled software and tools.

FreeDOS is a command line operating system, as are all DOS-based systems. There are various GUIs which are available for FreeDOS and it is even possible to get it working with Windows 3.x and other previous versions of Windows. It cannot, however, replace MS-DOS version 7.x or higher for supporting Windows 95 or 98. Microsoft has made many efforts in their Windows 9x operating systems to ensure that they will only run atop MS-DOS 7.x.

FreeDOS is not a multi-threaded multi-tasking operating system but a single task-oriented monolithic operating system. Therefore, it does not benefit from additional processor cores.

FreeDOS does have many advantages over its MS-DOS predecessor including built-in CD and DVD support, FAT32 and LFN support. It also has an integrated boot manager which can be installed to the MBR to multi-boot between multiple operating systems or it can be integrated (a manual configuration process) with existing disk boot managers. However, unlike previous DOS operating systems, FreeDOS comes bundled with a great many tools and utilities that greatly enhance DOS functionality.

Accessing NTFS is possible due to the integration of the *Ntfsprogs* open source software tool, which allows read/write access to an NTFS volume without mounting it. Although accessing Linux-based Extended partitions is possible, this capability is not integrated directly into FreeDOS but made possible by using the *Ltools* open source software tool.

As with most DOS operating systems, FreeDOS does not support USB, although several third-party add-ons are available for this purpose.

Unlike standard DOS systems, FreeDOS provides basic TCP/IP networking and client software and natively supports several types of network adapters. The operating system also comes bundled with the Watcom C/C++ compiler and DOS Extender technology enabling programs to access memory far beyond the standard 640 KiB limit. FreeDOS also provides fully functional extended and expanded memory managers.

1.2.2.2 MS-DOS background

MS-DOS is based on 86-DOS, an operating system written by Tim Patterson of Seattle Computer Products, who modelled his operating system on an earlier operating system named CP/M. Microsoft purchased 86-DOS in order to cut their development time for constructing and implementing an operating system for IBM to run on the new 8086 processor. 86-DOS was renamed MS-DOS and after some development in 1981, was released for use on IBM 8086 processor-based machines as MS-DOS 1.0.

Many DOS-based clones were marketed and some have survived up until today in niche markets. Some of the more popular DOS clones include IBM's PC-DOS, DR-DOS and FreeDOS among many others. PC-DOS was originally developed in parallel to MS-DOS at IBM but after some time the two operating systems forked and spawned similar yet distinct operating systems.

In time, MS-DOS found itself primarily used on PC clones, systems not made or sold by IBM and that used PC-DOS or other competitors such as Compaq, which marketed Compaq DOS. It was at this time that there was a push to consolidate the disparate 8086 systems in the marketplace and bring them under an IBM compatible architecture so that PC clones could have enough hardware similarity that the same software and operating systems could run across them all.

Many versions of MS-DOS have been released over the years and an in-depth accounting is beyond the scope of this technical memorandum. For a more in-depth timeline analysis of the various DOS operating systems, the reader is invited to read [3].

This technical memorandum examines three specific versions of MS-DOS including 6.22, 7.0 and 7.1. Unlike MS-DOS, FreeDOS was written to be a fully functional MS-DOS clone that came bundled with all of the tools, utilities and third-party programs which took years for Microsoft and various third parties to develop.

MS-DOS 6.22 was a highly functional and stable DOS system and is considered by some to be the last true MS-DOS operating system. All subsequent versions of MS-DOS were designed specifically for use with Windows 95, 98 or ME. MS-DOS 6.22 supported both EMS and XMS memory management, hard disk drive compression software, anti-virus, basic network support (which has been incorporated since MS-DOS 3.1), 2 GiB FAT16 filesystems and CD drives. However, many of these features were included in previous versions of MS-DOS and as such they should not necessarily be considered as unique to MS-DOS 6.22.

Both MS-DOS 7.0 and 7.1 are very similar except that DOS 7.1 fully supports FAT32 whereas DOS 7.0 and all previous versions of MS-DOS only support FAT12/16 filesystems. MS-DOS 7.x has never been available as a retail version of DOS and was instead bundled directly into its respective Windows operating system.

Interestingly, MS-DOS 7.0 and 7.1 never directly supported long filenames, which instead had to be supported using third-party software. MS-DOS 7.0 is in all respects very similar to MS-DOS 6.22 except that version 7.0 was required to run Windows 95 and 95A. MS-DOS 7.1 was used to boot up Windows 95B, 95C and Windows 98.

1.2.3 Windows background

This section briefly examines the various Windows 9x operating systems, which include Windows 95 and 98.

Experimentation has revealed that the various versions of MS-DOS (see Section 1.2.2) examined above support more memory than their respective Windows counterparts. Although in theory Windows 95 and 98 are capable of supporting more than 512 MiB memory, in practice these systems are known to be problematic and placing too much memory inside a Windows 9x machine can precipitate a variety of issues that can threaten system stability. Specific configuration workarounds can help mitigate some of the memory-related issues but not all of them. Even the configuration changes Microsoft suggests to correct large memory support for Windows 9x did nothing to correct the problems as these changes were made to the Windows 9x virtual machines running inside VMware Workstation 7.1.4 in an attempt at getting them to support larger amounts of memory. Unfortunately, the only Windows systems capable of supporting larger amounts of memory are the NT-based operating systems, including Windows NT 4.0 and higher.

1.2.3.1 Windows 95 background

Windows 95 is considered not only one of the most successful commercial operating systems of all time but it represented a major leap forward in evolution for the PC. Windows 95 provided a more seamless multitasking operating system far superior to Windows 3.x, which it was designed to replace. It supported a Plug'n'Play architecture although USB support was often considered unstable at best. It allowed DOS users to continue to run their favourite DOS programs including games and commercial software packages either directly within Windows or from a dedicated DOS prompt[7].

Windows 95 is a 32-bit Protected-Mode operating system which uses a 16-bit Real-Mode DOS system for booting. In Protected-Mode, Windows does not require DOS to manage its memory and has its own 32-bit memory management system. However, its memory management system has significant problems and does not allow for the support of large amounts of computer memory. Because Windows 95 is a Protected-Mode operating system it requires at least 80386 or better hardware.

All versions of Windows 95 fully supported LFN support, 32-bit disk access and 32-bit memory management. Versions of Windows 95 supporting FAT32 included Windows 95B and 95C while their predecessors did not. FAT32 was first supported under MS-DOS 7.1 and higher. LFN has never been supported by MS-DOS without third-party software but is supported at a DOS prompt while running under Windows terminal window. Windows 95 also supported TCP/IP and even provides a web browser in most versions for navigating the web. Network support under Windows was superior to networking provided by both DOS and Windows 3.x and although Windows supported CD-ROM, CD-R and CD-RW devices, it could only support DVD UDF version 1.02. DVD authoring was functional but always problematic under all versions of Windows 95.

Windows 95 was released in four specific versions. The first release was an OEM retail release which was targeted at the mass market looking to upgrade their DOS or Windows 3.x systems to Windows 95 and was distributed on diskette and CD-ROM. This version was Windows 95 4.00.950 and shipped with MS-DOS 7.0. The original OEM version of Windows 95 did not come with any web browser software although TCP/IP and group networking was fully functional. This version of Windows only supported FAT12/16 but did fully support Windows LFN. Customers who purchased the Microsoft Plus! add-on for Windows 95 could install Internet Explorer 1.0.

The next version of Windows 95 Service Pack 1/OEM Service Release 1 which was set at version 4.00.950 A came to market February 1996. It was released with the same version of MS-DOS as the original version of Windows 95. This version of Windows is generally referred to as Windows 95A and came bundled with Internet Explorer 2.0

Windows 95 OEM Service Release 2, commonly referred to as Windows 95B, was released with MS-DOS 7.1 and was set at version 4.00.950 B. This version of Windows finally supported FAT32, included Internet Explorer 3.0 and was released August 1996. Additional fixes which included the USB Supplement for OEM Service Release 2 and OEM Service Release 2.1 came out one year later in August 1997.

The final version of Windows 95 to be released was Windows 95 OEM Service Release 2.5, which was set at version 4.00.950 C. It came to market November 1997 and came bundled with Internet Explorer 4.0.

[7] A DOS prompt is obtained by exiting Windows or booting the system directly into DOS.

1.2.3.2 Windows 98 background

Windows 98, the successor to Windows 95, is a more stable 32-bit Protected-Mode operating system. Its graphical interface was marginally improved from that of Windows 95 but not so much so that a Windows 95 user could not find his way around.

Windows 98 provided many enhancements over Windows 95. The most important improvements include vastly improved system stability, more robust USB support and enhanced Plug'n'Play functionality. It also provided over 1,000 new native device drivers for a vast array of hardware. It also included Internet Explorer 4.01 and web-based technical support functionality. Multiple monitor support for up to 8 monitors is available directly from the operating system for power users. Improved DirectX[8] and device driver development model were also important features of Windows 98. ACPI power management finally found its way into the Windows operating system enabling laptop and energy conscious users' ability to reduce their environmental impact and preserve battery longevity. Moreover, Windows 98 provided Microsoft's first support initiative for Firewire.

Windows 98 also provided a lightweight web server for sharing web data with others in a workgroup or online and sported shortened system shutdown time. Improved DVD support was provided in Windows 98 as was a FAT32 conversion tool allowing Windows users to convert their FAT16 filesystems to FAT32. Improved networking capabilities such as a faster TCP/IP stack, PPTP tunnelling, multi-homing device configuration and support for additional networking hardware such as FDDI, ISDN and ATM enabled Windows 98 to connect individuals in more ways than ever before to the corporate network or the web.

An improved Dr. Watson utility helped Microsoft technical support better pinpoint the origins of various GPF errors which had plagued Windows 95 and were better controlled under Windows 98. The System Configuration Utility (e.g. *MSCONFIG.EXE*) was a useful tool that could be used to examine technical details about the operating system and the computer system's hardware including disabling unneeded boot-time services and programs. Finally, the System File Checker could be used to verify if critical system files had been corrupted and could replace them if necessary.

Of course, the above list of Windows 98 features is not all-inclusive. The first version of Windows 98 released to the mass market for retail arrived June 1998 and was set at version 4.10.1998. The next and final version of Windows 98 was the Second Edition (SE) implementation and was released May 1999 and set at version 4.10.1998A. Among the improvements found in Windows 98 SE were various bug fixes and the inclusion of Internet Explorer 5.0 and Internet Connection Sharing, which could take advantage of NAT. It also included improved ATM networking support and a more comprehensive Windows Media Player, which was now at version 6.2.

[8] Windows 95 never shipped with DirectX as it had to be installed separately.

2 Memory acquisition

2.1 Tools used for memory acquisition

2.1.1 Memdump

The *Memdump* memory acquisition tool was developed by APSoft of Germany (http://www.tssc.de/products/tools/memdump/default.htm). The tool can be used to linearly dump computer operating system memory from both DOS-compatible and 16/32-bit Windows operating systems including Windows 95 and 98. The tool was found to work under FreeDOS, MS-DOS, Windows 95C and 98 SE. The tool is available as freeware and does not require a commercially purchased license or registration in order to download or use it. The program is currently at version 2.00 and was released June 2005. No software source code is available for the tool.

The tool cannot acquire memory beyond the first 4 GiB of computer memory. Since neither DOS nor Windows 95 and 98 support large memory systems, this is not an issue. Furthermore, none of the aforementioned systems supports Intel PAE.

Memory acquisition under Windows 16/32-bit and DOS systems is different from memory acquisition under true 32 or 64-bit Windows operating systems. Neither Windows 95 nor 98 are true 32-bit operating systems as they run atop DOS in Protected-Mode, which is necessary in order to directly access computer memory beyond the first 1 MiB of RAM. Running in 32-bit Protected-Mode enables these Windows systems to bypass many of the limitations inherent to DOS.

Although one of the original intentions of developing and marketing Windows 95 and 98 was to provide additional memory capabilities to 32-bit applications, it in fact turns out that recent versions of DOS can support more memory than either of these Windows systems ever could. This will be clearly demonstrated in the experiments described later on in this section.

The *Memdump* memory acquisition tool is efficient and fast considering that it is only a 16-bit DOS program. The tool can be used to carry out two specific types of memory dumps including a raw binary memory dump (useful for forensic analysis) and a hexadecimal-based memory dump. The latter dump type should altogether be avoided as it is extremely slow and consumes far more disk space than a binary dump. The *Memdump* tool can be instructed to perform both dump types at the same time although this in no way speeds up its operation as the hexadecimal type dump is the tool's limiting factor. Raw binary memory dumping even on systems with 3 GiB RAM generally requires only a few minutes to complete. However, the tool does not perform checksumming of the memory dump file.

Furthermore, the *Memdump* tool expects the user to specify an address range for acquiring the computer system's memory and without a specified range the tool will not work. The range is specified as a start and stop address. Normally the start address is the offset for the desired first byte of computer memory which typically is at position 0 (zero), representing the very first byte of computer memory. Typically, the stop address will be the last available byte of memory. Both the start and stop address are specified in hexadecimal notation. For example, assuming a system with exactly 1 GiB RAM, the start address will be specified as 0 while the stop address will be specified as 0x40000000.

The tool will work even if a memory manager (e.g. EMS or XMS memory manager) is not installed. However, the tool must have access to sufficient Conventional Memory in order to run. In order to demonstrate that each examined DOS system was functioning correctly, the DOS *MEM* command was used to determine how much computer memory was actually available to the system which was used for specifying the stop address.

It was found through experimentation that attempting to acquire memory beyond the system's physical limit resulted in a duplication of the memory dump. For example, if a given DOS system has 512 MiB RAM (536,870,912 bytes) but is instead given a much larger address with a start address of 0 and stop address of 0x4D200000 (1,293,942,784 bytes) then the memory dump will have acquired the memory two times over.

Experiments have shown that while running under a Microsoft operating system (e.g. MS-DOS 6.22 or 7.x and Windows 95C or 98 SE) the *Memdump* tool could not dump more than 2 GiB RAM at any given time. Under FreeDOS, dump files over 3 GiB in size could be readily generated. The exact cause for this difference cannot be ascertained at this time.

Experimentation has revealed that *Memdump* will not work correctly when attempting to dump computer memory over the first 2 GiB RAM for any of the aforementioned systems. In order to acquire memory over the first 2 GiB RAM it is necessary to specify both a start and stop address over the first 2 GiB RAM. However, this will not work and will only cause the first 2 GiB RAM to be reacquired regardless of the new memory range (so long as the new start and stop address are over the first 2 GiB RAM). Only FreeDOS is not plagued by this problem.

2.1.2 Other possible tools

Although *Memdump* is not the only program or tool which can be used by investigators, it is the only one which is currently available publicly and very easy to use. Other potential programs and tools which could be used in lieu include software debuggers and specialized device drivers.

Software debuggers can be used as powerful memory acquisition and analysis tools but their use is complex and highly unintuitive, especially when considering the use of debuggers which were available during the era of DOS and Windows 9x. Some debuggers can be instructed to dump memory from specific offsets. It is these types of debuggers that an investigator could use as an alternative to *Memdump*, where he would have to specify the appropriate memory offsets for the dumping of memory. However, since this approach is largely related to software development, it is not appropriate to examine it here.

The investigator may be able to find specialized device drivers which can be used to dump memory on the Internet but these tools tend to use malware-based technology to accomplish their goal and as such are not appropriate for review in this technical memorandum. Certainly other memory acquisition software exists for DOS and Windows 9x but those that were found were either posted on illicit web sites or were themselves illicit in nature and as such are not suitable for examination. After several web searches no other suitable non-illicit program or tool could be found.

2.2 Memory acquisition experimentation

This section describes the experimentation relating to memory acquisition.

2.2.1 Experimentation background

The experiments were conducted in as straightforward a manner as possible. Under DOS, in order to appropriately populate system memory, various applications were run, depending on the type of memory management required[9]. An attempt was made to continue using the same programs and applications across all three DOS systems whenever possible. However, since each DOS operating system is different, it is not surprising that the ability to run all these programs and applications would have different outcomes under each specific system. Under each of the DOS systems the following commands were always run as they were available under each system:

MEM /C

DIR /A /W /S

The command *SMARTDRV* could only be used under MS-DOS as no FreeDOS equivalent was readily available. The *SMARTDRV* command was run only when XMS or XMS and EMS memory management was available in order to run the program in Upper Memory so as to minimize the impact on Conventional Memory. If loaded, *SMARTDRV* can use the High Memory Area to store its disk cache and therefore be of immense assistance in populating system memory.

Since *SMARTDRV* is not available under FreeDOS and could not be run under all circumstances under MS-DOS, running another program which could take advantage of High Memory was considered a must in order to adequately populate system memory. As such, the use of the DOS era game *DOOM2* was considered due to its very widespread use at that time. The *DOOM2* program was run under MS-DOS while FreeDOS came bundled with *FreeDOOM*.

It was decided from the outset that under DOS, three specific memory acquisition experiments would be conducted. The first would consist of memory acquisition without any memory manager. The second experiment would determine the ability to acquire memory from a system running under an XMS memory manager while the third would determine the ability to dump memory from a system running both XMS and EMS memory managers. Although some DOS systems may run only with an EMS memory manager, the use of EMS by itself does not enable the use of the DOS Upper Memory Area, which is often used to load important drivers and TSRs into non-Conventional Memory. For this reason, the testing of an EMS-only system was discounted as EMS-only systems are generally used exclusively with systems using specialty add-

[9] XMS or EMS memory management.

on memory boards or for use with specific DOS games, which only worked under EMS. All these various configurations were made from the appropriate DOS start-up files[10].

On the other hand, the Windows experimentation was simpler. Since Windows is not dependent on the two aforementioned DOS start-up files, they were left empty. Under Windows 95C and 98 SE, the only programs run in order to adequately populate system memory were the *Control Panel*, the *Control Panel*'s *System* applet, *Wordpad*, *Notepad*, *Internet Explorer* and *MS-DOS Prompt*.

The purpose of populating memory is to run some program or application that will likely execute in memory above the DOS 1 MiB memory limit, which leave behind artefacts in memory that can be readily extracted as ASCII-based strings. Using the UNIX *strings* command it is easy to verify if readable text strings do in fact exist at memory offsets above the DOS 1 MiB memory limit. In this manner it was possible to confirm that the memory above the DOS limit was in fact used and that the *Memdump* program was capable of dumping and acquiring real computer system memory.

All experiments were originally run on a Dell Precision 690 Workstation and validated using a Dell XPS Desktop with the former running Fedora Core 14 64-bit Linux and the latter running Windows 7 64-bit SP1 (see tables 1 and 2 in Annex A for more details, respectively).

The actual number of strings and memory offsets transcribed for each of the experiments as found in Annex B are those from the Dell Precision 690. However, each experiment was validated using the Dell XPS in order to ensure that: 1) the amount of acquirable memory was always the same, 2) the length of time required for acquisition was comparable[11], and 3) the number of text strings and byte offsets were comparable[12]. As such, it was possible to validate that the results are not due to differences in running the VirtualBox or VMware virtual machines under either Linux or Windows and that the results obtained are instead a direct manifestation of the various virtualised operating systems.

In the following subsections the various DOS and Windows experiments and results are briefly examined.

2.2.2 DOS memory acquisition

In this subsection, experimentation as it pertains to the various DOS-based operating systems is examined.

2.2.2.1 FreeDOS 1.0

Upon installing FreeDOS, the installation software creates highly customized system start-up files, which was not done for any of the other experiments. FreeDOS is very similar in most

[10] MS-DOS uses AUTOEXEC.BAT and CONFIG.SYS while FreeDOS replaces CONFIG.SYS with FDCONFIG.SYS.

[11] Comparable here indicates that the validation time can be no more than twice as fast or slow.

[12] Comparable here indicates that the validation time can be no more than twice as fast or slow.

regards to MS-DOS, although the location of DOS commands are found under *C:\FDOS\BIN* rather *C:\DOS*.

FreeDOS was the only operating system examined which would allow an investigator to fully acquire all of the memory seen by the operating system. Why this is possible under FreeDOS but is not realizable under MS-DOS or Windows is currently unknown. It was also interesting to observe that even without any memory managers in use FreeDOS was able to see and access all the potential memory the operating system would normally have access to only when running one or more memory managers. This reason for this also cannot be readily explained.

It is noteworthy to mention that memory acquisition under FreeDOS while running under an XMS memory manager required the most time in comparison to the other memory acquisition experiments conducted under FreeDOS. When no memory management software was used the dump was considerably faster than with all other methods. Even when memory acquisition was carried out under EMS and XMS memory management, the dump was faster than with XMS memory management alone but still slower than without any memory management software. It was also discovered that under FreeDOS, *FreeDOOM* could only run if both EMS and XMS memory management was present.

In summary, memory acquisition under FreeDOS was a complete, albeit slow, success. Experimental results for FreeDOS can be found in Annex B.1. These experiments were conducted under Oracle VirtualBox 4.0.6.

2.2.2.2 MS-DOS 6.22

DOS 6.22, the last true DOS from Microsoft, is only capable of seeing approximately 64.5 MiB RAM, even when using memory management software (XMS, EMS or both). However, unlike all the other DOS systems examined, MS-DOS 6.22 is the only system capable of running *DOOM2* under XMS and EMS/XMS memory management. None of the other DOS systems permit this. Another point of interest is the fact that while running under XMS memory management, *SMARTDRV* and *DOOM2* are incompatible. However, while running under EMS and XMS memory management, both could coexist. Finally, it is important to note that it was discovered that while EMS and XMS memory management are in use, the system reported 1 KiB more memory than the amount that would have been reported if only XMS or no memory management would have been in use.

In summary, DOS 6.22 gave the most DOS-like look and feel out of all the DOS-based operating systems examined. However, because of the much smaller amount of RAM detected by MS-DOS 6.22, the 2 GiB memory dump limit experienced by both versions of MS-DOS 7.1 (see Section 2.2.2.3 for more details) was not attained. Experimental results for MS-DOS 6.22 can be found in Annex B.2. These experiments were conducted under Oracle VirtualBox 4.0.6.

2.2.2.3 MS-DOS 7.1

This section actually consists of two set of experiments, one set for MS-DOS 7.1 bundled with Windows 95C and the other for MS-DOS 7.1 bundled with Windows 98 SE. Detailed information about these experiments can be found in annexes B.3 and B.4, respectively. These experiments were conducted under Oracle VirtualBox 4.0.6.

The results for both versions of MS-DOS 7.1 were for the most part identical. The only detected differences between these two versions of MS-DOS was the number of identified strings (and their byte offsets) found for each system while conducting the various sets of experiments. As such, both versions of MS-DOS detected the same amount of memory in each experiment set including experiments conducted without memory management software and with XMS and EMS/XMS memory management software in use.

When run without memory management, both experiments yielded memory dump file sizes mirroring exactly the same size as detected memory, 66,112 KiB RAM. Interestingly, when both systems were run with either XMS or EMS and XMS memory management, both systems detected 3,669,632 KiB RAM. When proceeding with the memory dump, however, it was found that the maximum memory dump size which could be attained was 2,097,120 KiB. Why this occurred for both sets of XMS and EMS/XMS experiments is entirely unknown. Attempting to circumvent the problem by specifying a new start and stop address corresponding to the byte offset immediately preceding the dump file size up to the highest available memory offset did nothing to remedy the acquisition of the memory from both systems. Instead, specifying a start byte offset just one byte larger than dump file size actually resulted in the re-imaging of system memory starting from the very first byte of system memory. As such, attempting to go beyond what appears to be either a hard memory or filesystem imposed limitation was not possible in the experiments concerning MS-DOS 7.1.

Finally, it was found the while running under both MS-DOS 7.1 systems the *DOOM2* and *SMARTDRV* applications could only function while running under XMS memory management. Neither would run if the EMS and XMS memory management was in use or without any memory management software.

2.2.3 Windows memory acquisition

In this subsection, experimentation as it pertains to the various Windows 9x operating systems is examined.

2.2.3.1 Windows 95C

A variety of memory sizes were allocated to the Windows 95C virtual machine under VMware Workstation 7.1.4. It was found that when allocated with as much as 4,096 MiB RAM, Windows 95 would fail and crash with the following error message “Insufficient memory to initialize Windows.” As VMware can only allocate memory in 4 MiB increments it was found through trial and error that 944 MiB (966,565 KiB) RAM was the maximum amount of allocable memory which would not cause Windows to crash. When allocating this much memory and running a DOS Prompt inside of Windows it was found that DOS could support a maximum of 64 MiB (65,184 KiB[13]) RAM regardless of Windows PIF configurations for the DOS command line interpreter COMMAND.COM.

[13] It is unknown why 65,184 KiB RAM was made available rather than 65,536 KiB.

DOS configuration files CONFIG.SYS and AUTOEXEC.BAT were left empty as their use was not required to start-up or define the memory configuration of Windows since it runs in Protected-Mode.

Upon loading various Windows programs to populate memory the *Memdump* program was run directly from the Windows Run dialogue box rather than from the DOS Prompt. Memory acquisition was straightforward and appeared to cause no specific problems until the acquisition had completed at which time the DOS window the program ran in would no longer close. Several minutes later the DOS window emitted an error stating that there was no longer enough memory in which to run this program. Attempting to close the DOS window resulted in a Windows GPF which crashed the virtual machine.

Running the *Memdump* program from the command line resulted in a memory acquisition file of exactly the same size as that run from the Windows Run dialogue box. However, this too caused the virtual machine to once again experience a GPF system crash.

Since the maximum amount of memory Windows could run upon was less than that which DOS 7.1 detected and could dump (approximately 2 GiB of memory) it was not possible to verify if Windows could also have dumped larger amounts of memory without hitting the 2 GiB memory dump file size already encountered under MS-DOS 7.1. More details concerning this experiment can be found in Annex B.5.

2.2.3.2 Windows 98 SE

A variety of memory sizes were allocated to the Windows 98 SE virtual machine under VMware Workstation 7.1.4. It was found that when allocated with as much as 4,096 MiB RAM, Windows 98 would fail and crash with the following error message "Insufficient memory to initialize Windows." As VMware can only allocate memory in 4 MiB increments it was found through trial and error that 1,156 MiB (1,183,744 KiB) RAM was the maximum amount of allocable memory which would not cause Windows to crash. When allocating this amount of memory and running a DOS Prompt inside of Windows it was found that DOS could support a maximum of 64 MiB (65,148 KiB[14]) RAM, 36 KiB less memory than the Windows 95C DOS Prompt experiment, regardless of Windows PIF configurations for the DOS command line interpreter COMMAND.COM.

DOS configuration files CONFIG.SYS and AUTOEXEC.BAT were left empty as their use was not required to start-up or define the memory configuration of Windows since it runs in Protected-Mode.

Upon loading various Windows programs to populate memory the *Memdump* program was run directly from the Windows Run dialogue box rather than from the DOS Prompt. Memory acquisition was straightforward and appeared to cause no specific problems until the acquisition had completed at which the DOS window the program ran in would no longer close. The DOS window was closed manually without resulting in a system GPF crash as occurred with the Windows 95C experiment.

[14] It is unknown why 65,184 KiB RAM was made available rather than 65,536 KiB.

Running the *Memdump* program from the command line resulted in a memory acquisition file of exactly the same size as that run from the Windows Run dialogue box. This time, no system crash was observed from having run the program from inside a DOS Prompt.

Since the maximum amount of memory Windows could run upon was less than that which DOS 7.1 detected and could dump (approximately 2 GiB of memory) it was not possible to verify if Windows could also have dumped larger amounts of memory without hitting the 2 GiB memory dump file size already encountered under DOS 7.1. More details concerning this experiment can be found in Annex B.6.

3 Conclusion

There is unfortunately no ubiquitous solution to acquiring computer memory from out-dated Windows and DOS operating systems. However, using the *Memdump* tool from APSoft it is possible, with certain caveats, to acquire the memory from these systems. Each system has its own particular quirk when it comes to memory acquisition. Detailed analyses and experimental results can be found in Section 2 and Annex B, respectively.

It would appear that based on the results obtained in Annex B from the variously conducted experiments, Windows 95 and 98, when booted directly into graphical mode support far less memory than when booted into DOS 7.1. Windows 95 in graphical mode becomes unstable upon completion of a memory dump while Windows 98 remains stable throughout and after the procedure. Strangely, when both these systems are booted into DOS mode and are allocated upwards of 4 GiB RAM the maximum memory dump size for both systems is only 2 GiB even though the FAT32 filesystems used to record the memory dumps should support maximum file sizes of approximately 4 GiB. Moreover, the two DOS 7.1 operating systems recognized well over 3 GiB allocated RAM when running memory management software. However, without any memory management software both DOS 7.1 systems would only recognize about 64 MiB RAM.

DOS 6.22 could only recognize about 64 MiB RAM even when running memory management software. However, acquiring its memory was fast and successful. FreeDOS was the only operating system which fully supported 32-bit memory allocation both with and without memory management software. Moreover, under all memory management experiments there were no specific issues encountered when acquiring the system's memory which in each case resulted in a memory dump file the same size as the amount of detected RAM. Interestingly, FreeDOS memory acquisition when running XMS memory management software was by far the slowest acquisition across all the experiments.

String and highest byte offset verification ensured that all the experiments conducted were representative of both the amount of memory detected by the operating system and the memory acquired by *Memdump* within the constraints of specific operating system and its memory management software. Furthermore, by populating memory with various applications such as *SMARTDRV* or *DOOM* (or *FreeDOOM*) under DOS and various user applications under Windows) it was possible to verify that these programs, particularly for DOS-based operating systems, do in fact use up memory beyond the first megabyte. Use of system memory beyond this all too common DOS barrier is indicative of two important items. The first is that memory management software does enable the use of memory beyond this limit. The second item is that user-based software can access this memory if designed to (*SMARTDRV*, *DOOM* and *FreeDOOM* can all access memory beyond the first megabyte with appropriate memory management software running on the target system).

One can therefore affirm with some certainty that *Memdump* can be used for forensically acquiring computer memory from DOS and Windows 95 and 98 systems alike. However, investigators are advised to be cognizant of the limitations of these different systems with respect to their ability to undergo memory acquisition. Although no tests could be conducted against Windows ME it is likely that this too would have been successful as its memory management software is similar enough to those employed by Windows 95 and 98.

References

[1] Halderman, J. Alex, Schoen, Seth D., Heninger, Nadia, et al. Lest We Remember: Cold Boot Attacks on Encryption Keys. Research paper. February 2008. Published in Proceedings 2008 USENIX Security Symposium. Princeton University. http://citp.princeton.edu/pub/coldboot.pdf

[2] Carbone, Richard. An in-depth analysis of the cold boot attack: Can it be used for sound forensic memory acquisition? Technical memorandum. TM No.: 2010-296. Defence R&D Canada – Valcartier. January 2011.

[3] Wikipedia. Timeline of x86 DOS operating systems. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. Http://en.wikipedia.org/wiki/Timeline_of_x86_DOS_operating_systems.

[4] Microsoft. "Out of Memory" Error Messages with Large Amounts of RAM Installed. Microsoft Support technical article. Article ID: 253912. Microsoft. http://support.microsoft.com/KiB/253912.

[5] Andrew, Thomas. WinME can't handle more than 512 megs of memory. Online journalistic article. The Register. http://www.theregister.co.uk/2000/11/24/winme_cant_handle_more_than.

This page intentionally left blank.

Annex A Details concerning computer systems used for experimentation

A.1 Systems used for experimentation

The following systems were used throughout this technical memorandum for the purposes of experimentation (see Annex A.1.1) and validation of the former's results (see Annex A.1.2).

A.1.1 Dedicated virtualization experimentation workstation

In order to create various VirtualBox and VMware virtual machines and carry out memory acquisition against said systems a dedicated computer platform was needed. Its specifications are as follows below in Table 1:

Table 1. Dedicated computer system for carrying out memory acquisition against DOS and Windows.

Computer model	Dell Precision 690 Workstation
Processors	Dual Xeon 3.20 GHz w/HyperThreading (8 logical processors)
Physical RAM	22.00 GiB RAM
Swap	None
Operating System	Linux Fedora Core 14, 64-bit
Virtualization Software	1. Oracle VirtualBox 4.0.6 with Extension Pack 2. VMware Workstation 7.1.4
Linux kernel	Kernel 2.6.35.12-90.fc14.x86_64 #1 SMP
Graphics adapter	Nvidia GeForce GTX 460
Graphics driver	Nvidia driver 270.41.06
Monitors	1) Dell E196FP LCD display (19”) 2) BenQ FP992 LCD display (19”)
Floppy	1.44 MiB floppy drive
USB	8 USB ports
Keyboard	USB US English keyboard
Mouse	USB optical mouse
FireWire	2 FireWire ports (no attached devices)
CD Drive	Hitachi CD-RW drive

DVD Drive	Philips CD-RW/DVD-RW drive
Hard drives	1) 1.5 TB Seagate 7,200 RPM SATA drive (system disk) 2) 3x 2 TB Hitachi 7,200 SATA drives in RAID 5 configuration with one spare yielding 4 TB disk space (software RAID connected to Vantec SATA controller) 3) 8x 2 TB Seagate 7,200 RPM SATA drive in RAID 5 configuration with no spare yielding 14 TB disk space (software RAID connected to Vantec SATA controller)
Host adapters	2x Vantec PCI Express E-SATA host adapter
Sound card	Sigma Tel HD sound card
Network cards	1) Broadcom NetXtreme Gigabit Ethernet 2) 1394 Net Adapter

A.1.2 Virtualization validation system

In order to validate the results obtained using the Dell Precision 690 workstation (see Table 1) the following system was used whose configuration can be found in Table 2 below:

Table 2. Result validation system.

Computer model	Dell Studio XPS Desktop 9100
Processors	i7 Quad-core 2.80 GHz w/HyperThreading (8 logical processors)
Physical RAM	18.00 GiB RAM
Swap	36.00 GB on exFAT partition atop PCI Express RevoDrive SSD
Operating System	Windows 7 64-bit Service Pack 1
Virtualization Software	1. Oracle VirtualBox 4.0.6 with Extension Pack 2. VMware Workstation 7.1.4
Graphics adapter	ATI Radeon 5670 HD
Graphics driver	ATI driver version 8.831.2.0
Monitors	1) Philips (19") 2) Philips (19")
Floppy	None
USB	7 USB ports
Keyboard	USB US English keyboard

Mouse	Dell USB optical mouse
FireWire	1 FireWire ports (no attached devices)
Optical Drives	1) LG Blu-Ray (RO)/DVD/CD drive 2) Plextor Blu-Ray RW
Hard drives	1) 2 x 750 MiB Seagate 7,200 RPM SATA drive in RAID 1 connected to Intel ICHR8 SATA RAID controller) 2) PCI Express RevoDrive SSD 100 GB
Host adapters	None
Sound card	Creative Labs PCI Express Sound Blaster X-I sound card
Network cards	1) Broadcom NetXtreme Gigabit Ethernet 2) RealTek RTL8168D Gigabit Ethernet

This page intentionally left blank.

Annex B Memory acquisition results

B.1 Results for FreeDOS 1.0

B.1.1 System information

Table 3. FreeDOS system information.

Version	FreeCom version 0.84-prex XMS_Swap [Aug 28 2006 00:29:00]
Allocated memory	4,096 MiB (4,194,304 KiB or 0x100000000)
Disk size	8,197 MiB
Drive name	C:\
Filesystem type	FAT32

B.1.2 Experiment 1

Table 4. Results for Experiment 1 – no memory management.

Memory management status	No memory managers loaded
FDCONFIG.SYS loaded	No
AUTOEXEC.BAT loaded	No
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	None
Conventional Memory (used)	144 KiB
Conventional Memory (free)	495 KiB
Total system memory	3,669,952 KiB (0xDFFF0000)
Command used	C:\memdump /D:0,0xDFFF0000 /F:none /B:mem1.dd
Command output: DPMI interface Used processor Prot mode V86 mode VCPI interface XMS interface Flat interface	 Not detected Pentium class NO NO Not detected Not detected Assumed
Time elapsed	About 10 minutes
Memory dump size	3,669,952 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	3,277
Number of strings (8-bit)	24,340
Number of strings (16-bit)	26
Number of strings (32-bit)	0
Highest byte string offset	1,048,565
Additional notes	None

B.1.3 Experiment 2

Table 5. Results for Experiment 2 – XMS memory management.

Memory management status	XMS (HIMEM) memory manager loaded
FDCONFIG.SYS loaded	No
AUTOEXEC.BAT loaded	No
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	None
Conventional Memory (used)	86 KiB
Conventional Memory (free)	553 KiB
Total system memory	3,669,952 KiB (0xDFFF0000)
Command used	C:\memdump /D:0,0xDFFF0000 /F:none /B:mem2.dd
Command output: DPMI interface Used processor Prot mode V86 mode VCPI interface XMS interface Version	 Not detected Pentium class NO NO Not detected Detected 3.0
Time elapsed	About 95 minutes
Memory dump size	3,669,952 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	8,555
Number of strings (8-bit)	44,126
Number of strings (16-bit)	41
Number of strings (32-bit)	1
Highest byte string offset	32,148,823
Additional notes	SMARTDRV was not available and FreeDOOM would not run.

B.1.4 Experiment 3

Table 6. Results for Experiment 3 – XMS and EMS memory management.

Memory management status	EMS and XMS (EMM386 and HIMEM) memory manager loaded
FDCONFIG.SYS loaded	No
AUTOEXEC.BAT loaded	No
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	FreeDOOM
Conventional Memory (used)	90 KiB
Conventional Memory (free)	649 KiB
Total system memory	3,669,952 KiB (0xDFFF0000)
Command used	C:\memdump /D:0,0xDFFF0000 /F:none /B:mem3.dd
Command output: DPMI interface Used processor Prot mode V86 mode VCPI interface Version	 Not detected Pentium class YES YES Detected 1.0
Time elapsed	About 25 minutes
Memory dump size	3,669,952 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	390,256
Number of strings (8-bit)	663,537
Number of strings (16-bit)	1,201
Number of strings (32-bit)	328
Highest byte string offset	71,806,995
Additional notes	SMARTDRV was not available but FreeDOOM was able to run.

B.1.5 System boot-up configuration files

B.1.5.1 FDCONFIG.SYS

```
!COUNTRY=001,437,C:\FDOS\BIN\COUNTRY.SYS
!SET lang=EN
!LASTDRIVE=Z
!BUFFERS=20
!FILES=40
!DOS=HIGH,UMB
!DOSDATA=UMB
!set dircmd=/ogn /4
!MENUCOLOR=7,0
MENUDEFAULT=2,5
MENU 1 - Load FreeDOS with EMM386, no EMS (most UMBs), max RAM free
MENU 2 - Load FreeDOS with EMM386+EMS and SHARE
MENU 3 - Load FreeDOS including HIMEM XMS-memory driver
MENU 4 - Load FreeDOS without drivers
DOS=HIGH,UMB
123?DEVICE=C:\FDOS\BIN\HIMEM.EXE
1?DEVICE=C:\FDOS\BIN\EMM386.EXE NOEMS X=TEST
2?DEVICE=C:\FDOS\BIN\EMM386.EXE X=TEST
;123?DEVICEHIGH=C:\FDOS\bin\xdma.sys
123?DEVICEHIGH=C:\FDOS\bin\xcdrom.sys /d:FDCD0001
123?DEVICEHIGH=C:\FDOS\bin\cdrcache.sys FDCD0001 CDRCACH0 15000
REM 123?INSTALL=C:\FDOS\BIN\BLACKOUT.EXE
REM 123?INSTALL=C:\FDOS\BIN\BANNER1.COM
123?DEVICEHIGH=C:\FDOS\BIN\MORESYS.SYS
SHELLHIGH=C:\FDOS\bin\command.com C:\FDOS\bin /E:1024 /P=C:\autoexec.bat
123?INSTALLHIGH=C:\FDOS\bin\lbacache.com 15000 TUNS
```

B.15.2 AUTOEXEC.BAT

```
@echo off
SET dosdir=C:\FDOS
C:\FDOS\BIN\BANNER2
C:\FDOS\BIN\BLACKOUT
set PATH=%dosdir%\bin
set NLSPATH=%dosdir%\NLS
set HELPPATH=%dosdir%\HELP
set temp=%dosdir%\temp
set tmp=%dosdir%\temp
SET BLASTER=A220 I5 D1 H5 P330
```

```
REM ShsuCDhd /QQ /F:C:\FDBOOTCD.ISO
if not "%config%"=="4" REM LH VIAUDIO
if not "%config%"=="4" REM LH VIAFMTSR
if not "%config%"=="4" LH FDAPM APMDOS
if "%config%"=="2" LH SHARE
if not "%config%"=="4" ShsuCDX /QQ /~ /D:?FDCD0002 /D:?FDCD0003
/D:?CDRCACH0
SET autofile=C:\autoexec.bat
alias reboot=fdapm warmboot
alias halt=fdapm poweroff
SET CFGFILE=C:\fdconfig.sys
echo type HELP to get support on commands and navigation
echo.
echo Welcome to FreeDOS
echo.
if not "%config%"=="4" mouse
lh doslfn
lh peruse /X8192
set PATH=%PATH%;%DOSDIR%\emacs
set PATH=%PATH%;%DOSDIR%\SETEDIT
SET VIM=C:\FDOS\VIM
SET PATH=%PATH%;%DOSDIR%\vim\vim70
SET PATH=%PATH%;%DOSDIR%\fbc
SET PATH=%DOSDIR%\FPC\BIN\GO32V2;%PATH%
call %DOSDIR%\watcom\setvars.bat
SET PATH=%PATH%;%DOSDIR%\PACIFIC\BIN
SET PATH=%PATH%;%DOSDIR%\xharbour\bin
SET PATH=%PATH%;%DOSDIR%\DOG
BLACKOUT
MODE CO80
```

B.2 Results for MS-DOS 6.22

B.2.1 System information

Table 7. MS-DOS 6.22 system information.

Version	MS-DOS Version 6.22
Allocated memory	4,096 MiB (4,194,304 KiB or 0x100000000)
Disk size	2,047 MiB
Drive name	C:\
Filesystem type	FAT16

B.2.2 Experiment 1

Table 8. Results for Experiment 1 – no memory management.

Memory management status	No memory managers loaded
CONFIG.SYS loaded	No
AUTOEXEC.BAT loaded	No
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	None
Conventional Memory (used)	61 KiB
Conventional Memory (free)	577 KiB
Total system memory	66,111 KiB (0x408FC00)
Command used	C:\memdump /D:0,0x408FC00 /F:none /B:C:\mem1.dd
Command output: DPMI interface Used processor Prot mode V86 mode VCPI interface XMS interface Flat interface	 Not detected Pentium class NO NO Not detected Not detected Assumed
Time elapsed	Less than 1 minute
Memory dump size	66,111 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	2,522
Number of strings (8-bit)	20,726
Number of strings (16-bit)	16
Number of strings (32-bit)	0
Highest byte string offset	1,048,565
Additional notes	None

B.2.3 Experiment 2

Table 9. Results for Experiment 2 – XMS memory management.

Memory management status	XMS (HIMEM) memory manager loaded
CONFIG.SYS loaded	Yes
AUTOEXEC.BAT loaded	Yes
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	DOOM2
Conventional Memory (used)	93 KiB
Conventional Memory (free)	545 KiB
Total system memory	66,111 KiB (0x408FC00)
Command used	C:\memdump /D:0,0x408FC00 /F:none /B:C:\mem2.dd
Command output: DPMI interface Used processor Prot mode V86 mode VCPI interface XMS interface Version	 Not detected Pentium class NO NO Not detected Detected 3.0
Time elapsed	Less than 1 minute
Memory dump size	66,111 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	70,647
Number of strings (8-bit)	133,430
Number of strings (16-bit)	156
Number of strings (32-bit)	156
Highest byte string offset	10,711,032
Additional notes	DOOM2 and SMARTDRV are incompatible so only DOOM2 was run so as not to crash the virtual machine.

B.2.4 Experiment 3

Table 10. Results for Experiment 3 – XMS and EMS memory management.

Memory management status	EMS and XMS (EMM386 and HIMEM) memory managers loaded
CONFIG.SYS loaded	Yes
AUTOEXEC.BAT loaded	Yes
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	SMARTDRV /L 34000 34000 ; DOOM2
Conventional Memory (used)	103 KiB
Conventional Memory (free)	541 KiB
Total system memory	66,112 KiB (0x4090000)
Command used	C:\memdump /D:0,0x4090000 /F:none /B:D:\mem3.dd
Command output: DPMI interface Used processor Prot mode V86 mode VCPI interface Version	 Not detected Pentium class YES YES Detected 1.0
Time elapsed	Less than 1 minute
Memory dump size	66,112 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	239,116
Number of strings (8-bit)	449,366
Number of strings (16-bit)	548
Number of strings (32-bit)	662
Highest byte string offset	45,837,014
Additional notes	Memory grew by 1 KiB in size due to the use of EMS. In this configuration, both SMARTDRV and DOOM2 could be run without causing stability issues.

B.2.5 System boot-up configuration files

B.2.5.1 CONFIG.SYS

```
DEVICE=C:\WINDOWS\HIMEM.SYS
DEVICE=C:\WINDOWS\EMM386.EXE NOEMS
DOS=HIGH,UMB
FILES=40
BUFFERS=20
SHELL=C:\COMMAND.COM /E:1024 /P
DEVICEHIGH=cd1.SYS /D:banana
```

B.2.5.2 AUTOEXEC.BAT

```
LOADHIGH C:\SHSUCDX /D:banana /L:R
LOADHIGH MOUSE.COM
```

B.3 Results for Windows 95C based MS-DOS 7.1

B.3.1 System information

Table 11. Windows 95C/MS-DOS 7.10 system information.

Version	Windows 95 [Version 4.00.1111]
Allocated memory	4,096 MiB
Disk size	8,189 MiB
Drive name	C:\
Filesystem type	FAT32

B.3.2 Experiment 1

Table 12. Results for Experiment 1 – no memory management.

Memory management status	No memory managers loaded
CONFIG.SYS loaded	No
AUTOEXEC.BAT loaded	No
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	None
Conventional Memory (used)	108 KiB
Conventional Memory (free)	532 KiB
Total system memory	66,112 KiB (0x4090000)
Command used	C:\memdump /D:0,0x4090000 /F:none /B:D:\mem1.dd
Command output: DPMI interface Used processor Prot mode V86 mode VCPI interface XMS interface Flat interface	 Not detected Pentium class NO NO Not detected Not detected Assumed
Time elapsed	Less than 2 minutes
Memory dump size	66,112 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	7,093
Number of strings (8-bit)	28,806
Number of strings (16-bit)	82
Number of strings (32-bit)	1
Highest byte string offset	1,048,565
Additional notes	None

B.3.3 Experiment 2

Table 13. Results for Experiment 2 – XMS memory management.

Memory management status	XMS (HIMEM) memory manager loaded
CONFIG.SYS loaded	Yes
AUTOEXEC.BAT loaded	Yes
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	SMARTDRV /L 34000 34000
Conventional Memory (used)	55 KiB
Conventional Memory (free)	585 KiB
Total system memory	3,669,632 KiB (0xDFFA0000)
Command used	C:\memdump /D:0,0xDFFA0000 /F:none /B:D:\mem2.dd
Command output: DPMI interface Used processor Prot mode V86 mode VCPI interface XMS interface Version	 Not detected Pentium class NO NO Not detected Detected 3.0
Time elapsed	About 55 minutes
Memory dump size	2,097,120 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	19,433
Number of strings (8-bit)	80,524
Number of strings (16-bit)	98
Number of strings (32-bit)	1
Highest byte string offset	35,917,819
Additional notes	SMARTDRV could be loaded but DOOM2 could not be run without crashing the virtual machine. Thus, this experiment was carried out without running DOOM2. Even without running SMARTDRV DOOM2 would not run.

B.3.4 Experiment 3

Table 14. Results for Experiment 3 – XMS and EMS memory management.

Memory management status	EMS and XMS (EMM386 and HIMEM) memory managers loaded
CONFIG.SYS loaded	Yes
AUTOEXEC.BAT loaded	Yes
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	None
Conventional Memory (used)	65 KiB
Conventional Memory (free)	579 KiB
Total system memory	3,669,632 KiB (0xDFFA0000)
Command used	C:\memdump /D:0,0x0xDFFA0000 /F:none /B:D:\mem3.dd
Command output: DPMI interface Used processor Prot mode V86 mode VCPI interface Version	 Not detected Pentium class YES YES Detected 1.0
Time elapsed	About 15 minutes
Memory dump size	2,097,120 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	17,933
Number of strings (8-bit)	66,815
Number of strings (16-bit)	117
Number of strings (32-bit)	0
Highest byte string offset	4,348,352
Additional notes	Neither SMARTDRV nor DOOM2 could be run. Thus, this experiment was carried out without running any additional programs.

B.3.5 System boot-up configuration files

B.3.5.1 CONFIG.SYS

```
DEVICE=C:\WINDOWS\HIMEM.SYS
DEVICE=C:\WINDOWS\EMM386.EXE NOEMS
DOS=HIGH,UMB
FILES=40
BUFFERS=20
SHELL=C:\COMMAND.COM /E:1024 /P
DEVICEHIGH=C:\VIDE-CDD.SYS /D:EIDECD1
```

B.3.5.2 AUTOEXEC.BAT

```
LOADHIGH C:\SHSUCDX /D:EIDECD1 /L:E
LOADHIGH MOUSE.EXE
```

B.4 Results for Windows 98 SE based MS-DOS 7.1

B.4.1 System information

Table 15. Windows 98 SE/MS-DOS 7.10 system information.

Version	Windows 98 [Version 4.10.2222]
Allocated memory	4,096 MiB
Disk size	8,189 MiB
Drive name	C:\
Filesystem type	FAT32

B.4.2 Experiment 1

Table 16. Results for Experiment 1 – no memory management.

Memory management status	No memory managers loaded
CONFIG.SYS loaded	No
AUTOEXEC.BAT loaded	No
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	None
Conventional Memory (used)	96 KiB
Conventional Memory (free)	544 KiB
Total system memory	66,112 KiB (0x4090000)
Command used	C:\memdump /D:0,0x4090000 /F:none /B:D:\mem1.dd
Command output	
DPMI interface	Not detected
Used processor	Pentium class
Prot mode	NO
V86 mode	NO
VCPI interface	Not detected
XMS interface	Not detected
Flat interface	Assumed
Time elapsed	Less than 1 minute
Memory dump size	66,112 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	5,767
Number of strings (8-bit)	35,833
Number of strings (16-bit)	18
Number of strings (32-bit)	4
Highest byte string offset	1,508,358
Additional notes	None

B.4.3 Experiment 2

Table 17. Results for Experiment 2 – XMS memory management.

Memory management status	XMS (HIMEM) memory manager loaded
CONFIG.SYS loaded	Yes
AUTOEXEC.BAT loaded	Yes
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	SMARTDRV /L 34000 34000
Conventional Memory (used)	76 KiB
Conventional Memory (free)	564 KiB
Total system memory	3,669,632 KiB (0xDFFA0000)
Command used	C:\memdump /D:0,0xDFFA0000 /F:none /B:D:\mem2.dd
Command output DPMI interface Used processor Prot mode V86 mode VCPI interface XMS interface Version	 Not detected Pentium class NO NO Not detected Detected 3.0
Time elapsed	About 75 minutes
Memory dump size	2,097,120 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	15,539
Number of strings (8-bit)	98,617
Number of strings (16-bit)	202
Number of strings (32-bit)	28
Highest byte string offset	37,886,448
Additional notes	SMARTDRV could be loaded but DOOM2 could not be run without crashing the virtual machine. Thus, this experiment was carried out without running DOOM2. Even without running SMARTDRV DOOM2 would not run.

B.4.4 Experiment 3

Table 18. Results for Experiment 3 – XMS and EMS memory management.

Memory management status	EMS and XMS (EMM386 and HIMEM) memory managers loaded
CONFIG.SYS loaded	Yes
AUTOEXEC.BAT loaded	Yes
Commands run	MEM/C ; DIR /A/W/S
Other programs loaded	None
Conventional Memory (used)	86 KiB
Conventional Memory (free)	558 KiB
Total system memory	3,669,632 KiB (0xDFFA0000)
Command used	C:\memdump /D:0,0x0xDFFA0000 /F:none /B:D:\mem3.dd
Command output	
DPMI interface	Not detected
Used processor	Pentium class
Prot mode	YES
V86 mode	YES
VCPI interface	Detected
Version	1.0
Time elapsed	About 50 minutes
Memory dump size	2,097,120 KiB
Triggered system crash	No
MEMDUMP application crash	No
Number of strings (7-bit)	15,760
Number of strings (8-bit)	68,934
Number of strings (16-bit)	62
Number of strings (32-bit)	5
Highest byte string offset	4,348,352
Additional notes	Neither SMARTDRV nor DOOM2 could be run. Thus, this experiment was carried out without running any additional programs.

B.4.5 System boot-up configuration files

B.4.5.1 CONFIG.SYS

```
DEVICE=C:\WINDOWS\HIMEM.SYS
DEVICE=C:\WINDOWS\EMM386.EXE NOEMS
DOS=HIGH,UMB
FILES=40
BUFFERS=20
SHELL=C:\COMMAND.COM /E:1024 /P
DEVICEHIGH=C:\XCDROM.SYS /D:CDROM1
```

B.4.5.2 AUTOEXEC.BAT

```
LOADHIGH MSCDEX.EXE /D:CDROM1 /L:E
LOADHIGH MOUSE.EXE
```

B.5 Results for Windows 95C

B.5.1 System information

Table 19. Windows 95C system information.

Version	Windows 95 [Version 4.00.1111]
Maximum allocable memory	944 MiB (966,656 KiB or 0x3B000000)
Disk size	8,189 MiB
Drive name	C:\
Filesystem type	FAT32

B.5.2 Experiment 1

Table 20. Results for Experiment 1 – Windows default memory management.

Memory management status	XMS memory manager (HIMEM.SYS) automatically loaded by Windows
CONFIG.SYS loaded	Yes (file empty)
AUTOEXEC.BAT loaded	Yes (file empty)
Commands run	Control Panel Notepad Wordpad Internet Explorer 1.0 MS-DOS Prompt (COMMAND.COM)
Other programs loaded	None
Used Conventional Memory (from DOS Prompt)	63 KiB
Free Conventional Memory (from DOS Prompt)	573 KiB
Total system memory (Windows detected)	966,656 KiB (0x3B000000)
Total system memory (DOS shell detected)	65,184 KiB (0x3FA8000)
Command used	C:\memdump.exe /D:0,0x3B000000 /F:none /B:C:\mem.dd
Command output DPMI interface Version Processor 32-bit API	 Detect at FB98:2F93 0.90 80486 YES
Time elapsed	About 1 minute
Memory dump size	966,656 KiB
Triggered system crash	Yes
MEMDUMP application crash	No

Number of strings (7-bit)	737
Number of strings (8-bit)	10,806
Number of strings (16-bit)	1
Number of strings (32-bit)	0
Highest byte string offset	989,823,300
Additional notes	The MEMDUMP.EXE command is run from the Start -> Run. This opens a DOS Prompt window where the dump is carried out. Once complete, the window does not automatically close. Several minutes upon the dump terminating a Windows error message stating not enough memory to run this program is presented to the investigator. This message then continues to occur every few moments. Closing the DOS Prompt opened up by running the MEMDUMP.EXE command causes a Windows fatal exception 0E error and then a GPF. At this point, the investigator should forcibly power off the Windows 95 system. Running the program from the command line results in a dump the same size as total physical memory (966,656 KiB) even though the DOS Prompt only recognizes only 65,184 KiB memory.

B.5.3 System boot-up configuration files

B.5.5.1 CONFIG.SYS

EMPTY

B.5.5.2 AUTOEXEC.BAT

EMPTY

B.6 Results for Windows 98 SE

B.6.1 System information

Table 21. Windows 98 SE system information.

Version	Windows 98 Second Edition 4.10.2222 A
Maximum allocable memory	1,156 MiB (1,183,744 KiB or 0x48400000)
Disk size	8,189 MiB
Drive name	C:\
Filesystem type	FAT32

B.6.2 Experiment 1

Table 22. Results for Experiment 1 – Windows default memory management.

Memory management status	XMS memory manager (HIMEM.SYS) automatically loaded by Windows
CONFIG.SYS loaded	Yes (file empty)
AUTOEXEC.BAT loaded	Yes (file empty)
Commands run	Control Panel Notepad Wordpad Internet Explorer 5.0 MS-DOS Prompt (COMMAND.COM)
Other programs loaded	None
Used Conventional Memory (from DOS Prompt)	35 KiB
Free Conventional Memory (from DOS Prompt)	601 KiB
Total system memory (Windows detected)	1,183,744 KiB (0x48400000)
Total system memory (DOS shell detected)	65,148 KiB (0x3F9F000)
Command used	C:\memdump.exe /D:0, 0x48400000 /F:none /B:C:\mem.dd
Command output DPMI interface Version Processor 32-bit API	 Detect at FB98:2F93 0.90 80486 YES
Time elapsed	About 2 minutes
Memory dump size	1,183,744 KiB
Triggered system crash	No
MEMDUMP application crash	No

Number of strings (7-bit)	23,884
Number of strings (8-bit)	96,212
Number of strings (16-bit)	65
Number of strings (32-bit)	2
Highest byte string offset	1,212,125,505
Additional notes	The MEMDUMP.EXE command was run from the Start -> Run. This opens a DOS Prompt window where the dump is carried out. Once complete, the window does not automatically close and was instead closed manually. No crash occurred. Running the program from the command line results in a dump the same size as total physical memory (1,183,744 KiB) even though the DOS Prompt only recognizes only 65,148 KiB memory.

B.6.3 System boot-up configuration files

B.5.5.1 CONFIG.SYS

EMPTY

B.5.5.2 AUTOEXEC.BAT

EMPTY

Bibliography

FreeDOS. FreeDOS wiki FreeDOS Spec. Informational online article. FreeDOS. August 2010. http://sourceforge.net/apps/mediawiki/freedos/index.php?title=FreeDOS_Spec.

FreeDOS. FreeDOS wiki Main Page. Informational online article. FreeDOS. August 2010. http://sourceforge.net/apps/mediawiki/freedos/index.php?title=Main_Page.

Houlden, Marcus. MS-DOS History. Informational online article. Nukesoft.co.uk. 2005. http://www.nukesoft.co.uk/msdos/dosversions.html.

Microsoft Corporation. Microsoft Extensible Firmware Initiative FAT32 File System Specification – FAT: General Overview of On-Disk Format. Hardware white paper. December 2000. http://msdn.microsoft.com/en-us/windows/hardware/gg463084.

Wikipedia. 3 GB barrier. Online encyclopaedic article. Wikimedia Foundation Inc. April 2011. http://en.wikipedia.org/wiki/3_GB_barrier.

Wikipedia. 86-DOS. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/86-DOS.

Wikipedia. Comparison of x86 DOS operating systems. Online encyclopaedic article. Wikimedia Foundation Inc. March 2011. http://en.wikipedia.org/wiki/Comparison_of_x86_DOS_operating_systems.

Wikipedia. COMMAND.COM. Online encyclopaedic article. Wikimedia Foundation Inc. April 2011. http://en.wikipedia.org/wiki/COMMAND.COM.

Wikipedia. Conventional memory. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/Conventional_memory.

Wikipedia. Doom (video game). Online encyclopaedic article. Wikimedia Foundation Inc. June 2011. Wikipedia. http://en.wikipedia.org/wiki/Doom_(video_game).

Wikipedia. Doom II: Hell on Earth. Online encyclopaedic article. Wikimedia Foundation Inc. June 2011. Wikipedia. http://en.wikipedia.org/wiki/Doom_II:_Hell_on_Earth.

Wikipedia. DOS extender. Online encyclopaedic article. Wikimedia Foundation Inc. March 2011. Wikipedia. http://en.wikipedia.org/wiki/DOS_extender.

Wikipedia. EMM386. Online encyclopaedic article. Wikimedia Foundation Inc. March 2011. Wikipedia. http://en.wikipedia.org/wiki/EMM386.

Wikipedia. Expanded memory. Online encyclopaedic article. Wikimedia Foundation Inc. March 2011. Wikipedia. http://en.wikipedia.org/wiki/Expanded_memory.

Wikipedia. Extended memory. Online encyclopaedic article. Wikimedia Foundation Inc. March 2011. Wikipedia. http://en.wikipedia.org/wiki/Extended_memory.

Wikipedia. File Allocation Table. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/File_Allocation_Table.

Wikipedia. FreeDOS. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/FreeDOS.

Wikipedia. High memory area. Online encyclopaedic article. Wikimedia Foundation Inc. March 2011. Wikipedia. http://en.wikipedia.org/wiki/High_memory_area.

Wikipedia. HIMEM.SYS. Online encyclopaedic article. Wikimedia Foundation Inc. March 2011. Wikipedia. http://en.wikipedia.org/wiki/HIMEM.SYS.

Wikipedia. History of Microsoft Windows. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/History_of_Microsoft_Windows.

Wikipedia. Intel 80286. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. Wikipedia. http://en.wikipedia.org/wiki/80286.

Wikipedia. Intel 80386. Online encyclopaedic article. Wikimedia Foundation Inc. April 2011. Wikipedia. http://en.wikipedia.org/wiki/80386.

Wikipedia. Intel 8086. Online encyclopaedic article. Wikimedia Foundation Inc. April 2011. Wikipedia. http://en.wikipedia.org/wiki/8086.

Wikipedia. Intel 8088. Online encyclopaedic article. Wikimedia Foundation Inc. March 2011. Wikipedia. http://en.wikipedia.org/wiki/8088.

Wikipedia. ISO 9660. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. Wikipedia. http://en.wikipedia.org/wiki/ISO_9660.

Wikipedia. Loadhigh. Online encyclopaedic article. Wikimedia Foundation Inc. March 2011. http://en.wikipedia.org/wiki/Loadhigh.

Wikipedia. Long filename. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/Long_filename.

Wikipedia. MSAV. Online encyclopaedic article. Wikimedia Foundation Inc. December 2010. http://en.wikipedia.org/wiki/MSAV.

Wikipedia. MS-DOS. Online encyclopaedic article. Wikimedia Foundation Inc. April 2011. http://en.wikipedia.org/wiki/Ms-dos.

Wikipedia. Physical Address Extension. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/Physical_Address_Extension.

Wikipedia. Program Information File. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/Program_Information_File.

Wikipedia. Protected mode. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/Protected_mode.

Wikipedia. Real mode. Online encyclopaedic article. Wikimedia Foundation Inc. April 2011. http://en.wikipedia.org/wiki/Real_mode.

Wikipedia. Timeline of x86 DOS operating systems. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/Timeline_of_x86_DOS_operating_systems.

Wikipedia. Universal Disk Format. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. Wikipedia. http://en.wikipedia.org/wiki/Universal_Disk_Format.

Wikipedia. Upper memory area. Online encyclopaedic article. Wikimedia Foundation Inc. March 2011. http://en.wikipedia.org/wiki/Upper_memory_area.

Wikipedia. Windows 3.0. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/Windows_3.0.

Wikipedia. Windows 3.1x. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/Windows_3.1x.

Wikipedia. Windows 9x. Online encyclopaedic article. Wikimedia Foundation Inc. April 2011. http://en.wikipedia.org/wiki/Windows_9x.

Wikipedia. Windows 95. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/Windows_95.

Wikipedia. Windows 98. Online encyclopaedic article. Wikimedia Foundation Inc. May 2011. http://en.wikipedia.org/wiki/Windows_98.

List of symbols/abbreviations/acronyms/initialisms

3x	Windows 3.0, 3.1, and 3.11
86-DOS	8086-Disk Operating System
9x	Windows 95A/B/C or Windows 98/98 SE
ACPI	Advanced Configuration and Power Interface
ASCII	American Standard Code for Information Interchange
ATM	Asynchronous Transfer Mode
BIOS	Basic Input/Output System
BSD	Berkeley Software Distribution
CD	Compact Disc
CP/M	Control Program for Microcomputers
DOS	Disk Operating System
DR-DOS	Digital Research-Disk Operating System
DVD	Digital Video Disc / Digital Versatile Disc
EMM	Expanded Memory Manager
EMS	Expanded Memory System
FAT12/16/32	File Allocation Table 12-bit/16-bit/32-bit
FDDI	Fibre Distributed Data Interface
FreeBSD	Free Berkeley Software Distribution
FreeDOS	Free Disk Operating System
GB	Gigabyte (10^9 bytes)
GiB	Gibibyte (2^{30} bytes)
GPF	General Protection Fault
GUI	Graphical User Interface
IBM	International Business Machine Inc.
ISDN	Integrated Services Digital Network
KiB	Kibibyte (2^{10} bytes)
LFN	Long File Name
LIM	Lotus, Microsoft, Intel

MiB	Mebibyte (2^{20} bytes)
MBR	Master Boot Record
ME	Millennium
MS-DOS	Microsoft Disk Operating System
NAT	Network Address Translation
NetBSD	Net Berkeley Software Distribution
NT	New Technology
NTFS	New Technology File System
OEM	Original Equipment Manufacturer
OpenBSD	Open Berkeley Software Distribution
PAE	Physical Address Extension
PC	Personal Computer
PC-DOS	Personal Computer-Disk Operating System
PIF	Program Information File
PPTP	Point-to-Point Tunnelling Protocol
QEMM	Quarterdeck Expanded Memory Manager
RAM	Random Access Memory
ROM	Read-Only Memory
SE	Second Edition
SPARC	Scalable Processor ARChitecture
TB	Terabyte (10^{12} bytes)
TCP/IP	Transmission Control Protocol/Internet Protocol
TSR	Terminate and Stay Resident
USB	Universal Serial Bus
XMM	eXtended Memory Manager
XMS	eXtended Memory System

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Defence R&D Canada – Valcartier
2459 Pie-XI Blvd North
Quebec (Quebec)
G3J 1X5 Canada

2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.)

UNCLASSIFIED

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

State of the art concerning memory acquisition software: A detailed examination of DOS and non-Windows NT memory acquisition

4. AUTHORS (last name, followed by initials – ranks, titles, etc. not to be used)

Carbone, R.

5. DATE OF PUBLICATION (Month and year of publication of document.)

October 2011

6a. NO. OF PAGES (Total containing information, including Annexes, Appendices, etc.)

64

6b. NO. OF REFS (Total cited in document.)

5

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Memorandum

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

Defence R&D Canada – Valcartier
2459 Pie-XI Blvd North
Quebec (Quebec)
G3J 1X5 Canada

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)

31XF20 « MOU RCMP "Live Forensics" »

9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)

DRDC Valcartier TM 2011-215

10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

Unlimited

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.))

Unlimited

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

(U) This technical memorandum examines one specific software tool which can be used to carry out a forensic memory acquisition of DOS and Windows 9x systems. This work appears to be the first of its kind as no other comparable work can be found in the publicly available literature. Although DOS and Windows 9x systems are harder to come by today, this should not preclude that investigators may encounter them in the course of their work. By addressing the important issue of DOS and Windows 9x memory acquisition it will be possible for investigators to corroborate disk-based evidence when examining such systems used to commit illicit activities.

(U) Ce mémorandum technique décrit un outil logiciel spécifique qui peut être utilisé pour procéder à une acquisition de mémoire inforensique de systèmes DOS et Windows 9x. Cette étude semble être la première du genre puisqu’aucun ouvrage/recherche comparable ne se trouve dans la littérature publique. Bien que les systèmes DOS et Windows 9x ne soient pas très présents de nos jours, il est quand même possible qu’un enquêteur les rencontre dans son travail. En abordant ce problème important de l’acquisition de mémoire DOS et Windows 9x, il sera possible pour les enquêteurs de rassembler les preuves corroborantes du disque lorsqu’ils examineront ces systèmes qui peuvent encore être utilisés aujourd’hui pour commettre des actes illicites.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Live forensics□Digital forensics□Computer forensics□Memory forensics□Memory acquisition□Memory dump□DOS□FreeDOS□Windows 95□Windows 98□Memory