디지털 포렌식 관점의 물리 메모리 영역 수집과 분석

*방제완 *김권엽 *이상진 *임종인

고려대학교 정보경영공학전문대학원

*{jwbang, kkyoup, sangjin, jilim}@korea.ac.kr

The Acquisition and Analysis of Physical Memory in a view of Digital Forensic

*Jewan Bang *Kwonyoup Kim *Sangjin Lee *Jongin Lim

*Graduate School of Information Management and Security, Korea University

요약

물리 메모리 영역에는 증거로 활용될 수 있는 프로세스 정보와 이름, ID, 비밀 번호, 전자 메일 주소 등의 정보를 담고 있다. 또 용의자가 행위를 감추기 위해 안티 포렌식 기법을 사용하여 저장매체 상에서 완전 삭제한 파일의 잔여 데이터를 취득할 수 있는 가능성이 있다. 하드 디스크와 같은 저장 매체의 경우 증거 수집 절차시 Hash와 같은 무결성 보장 과정을 거쳐 복사본의 유효성 확인이 가능하지만 물리 메모리 영역의 경우 운용 중인 시스템에서 발생하는 운영체제와 응용 프로그램의 동작에 의한 지속적인 데이터의 변화로 무결성 및 동일한 대상에서 수집되었다는 것을 확인하기 어렵고 소프트웨어 기반의 수집은 시스템의 상태를 변화 시킨다. 본 논문에서는 물리 메모리 영역 수집 기법을 알아보고 IEEE1394의 특성을 이용한 하드웨어 기반 물리 메모리 영역 수집 도구를 구현하였다. 또 수집된 물리 메모리 덤프를 이용하여 물리 메모리에서 얻을 수 있는 정보를 확인하고 동일 대상의 메모리와 다른 대상의 메모리를 비교하여 그 차이를 확인한다.

1. 서론

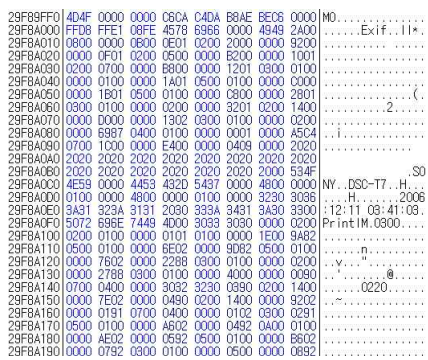
디지털 포렌식[1]은 컴퓨터나 PDA, 핸드폰과 같은 정보처리 기기에 저장되어 있는 데이터를 통해 각종 행위에 대한 사실 관계를 확증하거나 증명하기 위해 행하는 각종 절차와 방법을 말한다. 포렌식 절차는 준비, 증거물 획득, 증거물 보관 및 이송, 증거 분석, 보고서 작성 단계로 구성[2] 되어있다. 이 중 사고 발생 현장에서 디지털 증거를 수집하고, 증거의 무결성을 확보하는 증거물 획득 단계의 수집 사항 중 권고되는 것이 휘발성 데이터[3]이며 주로 메모리 또는 하드 디스크의 임시파일에 저장 되어 있다. 휘발성 데이터는 부팅된 후 운영체제에 의해 동작되고 있는 온라인 상태의 시스템에서의 시스템이 종료되면 사라지는 레지스터, 캐쉬, 실행 중인 프로세스 내역, 네트워크 연결 상태, 실행 중인 서비스 내역과 같은 기록을 말하며 대부분의 휘발성 데이터는 하드 디스크가 아닌 물리 메모리 영역에 존재한다[4].

물리 메모리 영역에는 증거로 활용될 수 있는 프로세스 정보와 이름, ID, 비밀 번호, 전자 메일 주소 등의 정보를 담고 있으며 사용자의 행위를 감추기 위해 안티 포렌식 기법[5]을 사용하여 저장매체에서 완전 삭제한 파일의 잔여 데이터를 취득할 수 있는 가능성을 가지고 있다. 본 논문에서는 소프트웨어 기반의 물리 메모리 수집 기법과 하드웨어 기반의 물리 메모리 수집 기법을 알아보고 IEEE1394[6]의 특성을 이용하여 하드웨어 기반 물리 메모리 영역 수집 도구를 구현하였다. 또 도구를 통해 수집된 물리 메모리 덤프 분석을 통해 얻을 수 있는 정보를 확인하고 동일 대상의 물리 메모리와 다른 대상의 물리 메모리를 비교하여 그 차이를 확인하였다.

* 본 연구는 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술개발사업의 일환으로 수행하였음. [2007-S019-01, 정보투명성 보장형 디지털 포렌식 시스템 개발]

2. 물리 메모리의 잔여 정보

시스템 설계시 보안보다는 성능을 우선하기 때문에 운영체제는 작업이 완료된 메모리 영역을 초기화 하지 않는다[7]. 그렇기 때문에 다른 작업으로 덮어 쓰이기 전까지 기존 데이터가 남아있게 된다.



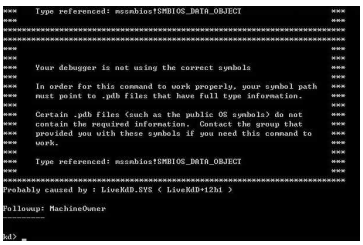
[그림 1] 삭제 이미지의 메모리 내 잔여 데이터

악성 코드의 경우 Obfuscation 과정을 거치거나 실행 압축이 되어 역공학 분석 과정을 어렵게 한다. 하지만 메모리 상에는 압축이 풀린 상태로 상주하므로 메모리 영역 취득을 통해 분석을 용이하게 할 수 있다. 보조 기억 장치를 마치 주기억 장치인 것처럼 이용하는 가상 메모리를 위한 pagefile.sys의 경우도 기본적으로 시스템이 종료될 때 삭제하지 않게 되어있어 증거가 될 수 있는 데이터 획득을 위해 지속적으로 연구[8] 되고 있다. 또 부팅이나 짧은 시간의 전원 차단 후에도 물리 메모리 영역의 데이터의 잔류가 확인[9]됨에 따라 물리 메모리 영역은 더 많은 정보를 확인할 수 있는 가능성을 가지고 있다.

3. 물리 메모리 수집 기법

가. 소프트웨어 기반 물리 메모리 수집 기법

저장 매체나 메모리 영역 Raw Imaging 도구인 dd[10]를 이용하여 *dd if=\\.\Device\PhysicalMemory of=e:\memoryimage.dd bs=4096* 와 같은 명령어로 메모리 영역을 수집할 수 있다. 하지만 커널 영역의 접근 권한 제한으로 시스템의 전체 메모리를 얻을 수 없으며 dd 동작을 위한 메모리 영역을 덮어 사용하게 된다. 다른 방법으로 Crash Dump[11]를 이용한 메모리 영역 수집 기법이 있다. Crash Dump는 시스템이나 응용 프로그램의 비정상 종료시 디버깅 정보를 저장하는 파일로 블루 스크린이나 프로그램이 응답하지 않는 오류 발생시 생성된다. Crash Dump에는 커널과 유저 영역 모두를 포함하는 Complete Memory Dump와 커널 영역만을 포함하는 Kernel Memory Dump, 오류가 발생한 응용 프로그램의 최소 정보만 담고 있는 Small Memory Dump가 있다. 이를 이용해 CrashOnCtrlScroll[12]와 NotMyFault[13] 등의 방법으로 의도적인 블루 스크린을 발생시켜 취득할 수 있는 기법이 있지만 재부팅 과정이 필요하며 pagefile.sys를 손상시킬 위험이 있다.

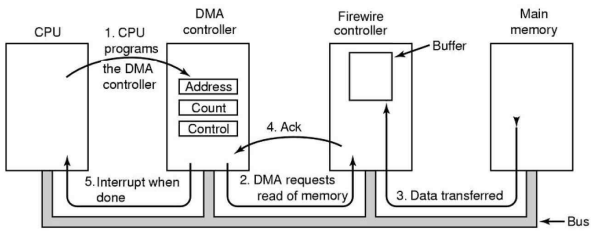


[그림 2] Microsoft LiveKD[17]

시스템 프로그래나 디바이스 드라이버 디버깅을 위한 커널 디버거 Microsoft LiveKD[14]를 사용하여 *.dump /f e:\memory.dmp* 명령어를 통해 Complete Memory Dump를 수집[15]할 수 있다. 또 X-Ways Capture[16]와 같은 도구를 통해 메모리 영역 수집이 가능하다.

나. 하드웨어 기반 물리 메모리 수집 기법

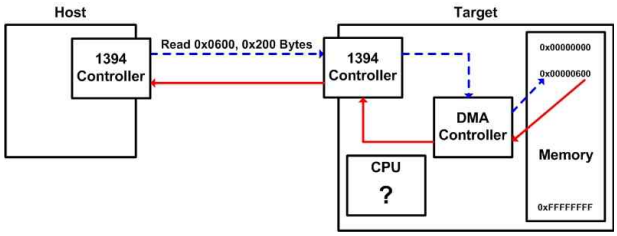
하드웨어 기반 수집 기법으로는 IEEE1394[6]의 CSR 변경을 통한 직접적인 물리 메모리 접근 기법[17]이 있다. IEEE1394는 디지털 캠코더, 디지털 TV, PC를 포함하는 멀티미디어 데이터 전송에 사용되는 인터페이스로 FireWire라고도 불린다. IEEE1394는 속도가 서로 다른 노드들 사이에서도 중재를 통해 하나의 버스를 통하여 전송이 가능하다. 버스는 IEEE1212[18]가 권고하는 Control and Status Register(CSR) Architecture를 따르며 64Bits의 주소 구조를 가진다.



[그림 3] DMA Controller Input/Output[20]

DMA(Direct Memory Access)는 CPU가 다른 태스크를 수행하는 동안 장치와 프로세스들 사이에서 메모리 전송을 허용 한다[19]. [그림 3]와 같이 CPU는 IEEE1394 장치가 특정 메모리 영역을 읽도록 DMA 컨트롤러에 명령을 내린 순간 다른 태스크를 수행할 수 있는 자유로운 상태가 된다. 명령이 완료된 후에도 CPU는 수행 완료에 대한 보고를 받을 필요가 없다. IEEE1394와 같은 DMA 버스 마스터 장치들은 CPU와 독립적으로 동작한다[20]. IEEE1394 CSR 영역에는 시스템에서의 노드 ID, 링크 상태 여부, 전송 속도, Vender ID, Model ID, Unit Software Version 등과 같은 장치 고유의 정보를 포함하고 있다.

1394 OHCI Specification[21]에 따르면 Control and Status Register(CSR) register 변경을 통해 시스템에 상관없이 Host Controller에 의해 직접적으로 읽기, 쓰기가 가능하다고 명시되어 있다.

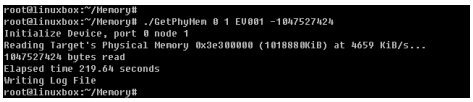


[그림 4] IEEE1394를 이용한 물리 메모리 접근

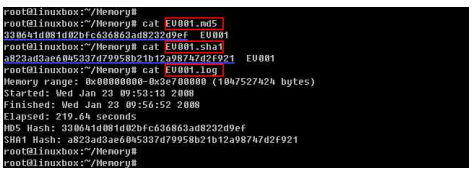
이러한 특성을 이용하여 IEEE1394 장치는 CPU의 제어를 받지 않고 DMA 컨트롤러에 물리 메모리 영역에 대한 읽기, 쓰기 명령을 수행할 수 있다.

4. 하드웨어 기반 물리 메모리 수집 도구

위의 연구를 통해 대상 시스템의 물리 메모리 영역을 수집할 수 있는 리눅스 기반의 도구를 구성하였다. IEEE1394의 64Bits의 주소 공간 중 하위 48bits 영역에 직접 명령을 내리기 위해 linux1394의 raw1394 드라이버[22]를 사용하였다.



[그림 5] 물리 메모리 영역 수집



[그림 6] 수집 결과 로그

인터페이스의 포트와 노드, 덤프 파일명과 수집할 영역 지정에 따라 수집을 수행하고 수집 내용에 대한 로그 파일을 기록하도록 하였다. 이를 통해 수집 대상 시스템의 메모리 영역의 변화를 주지 않으며 메모리 영역을 수집할 수 있는 환경을 마련하였다. 1,024MB의 물리 메모리를 가진 대상의 메모리 영역 수집에 약 220초의 시간 소요를 확인하였다.

5. 물리 메모리 분석

가. 물리 메모리 분석 기법

물리 메모리 영역 분석은 문자열 검색과 프로세스 시그니처 전수 조사를 통한 프로세스 구조체 추출로 나누어 볼 수 있다[23]. 물리 메모리 영역에서의 프로세스 정보 추출은 침입 시스템에서 악의적 프로세스 탐지를 위해 사용된다[21].win32 기반의 운영체제에서는 프로세스 정보를 EPROCESS 구조체로 메모리 상에 관리[7]하며 프로세스 카빙 도구는 이를 이용하여 각 구조체가 가지는 특성을 기반으로 프로세스 정보를 추출하게 된다. 이 구조체는 운영체제 버전에 따라 다른 시그니처[15]를 가진다.

No.	Type	PID	Time created	Offset	CR3	Remarks
1	Thrd	0		0x00562ca0		
2	Proc	0		0x00562f00	0x00039000	Idle
3	Thrd	4	2008-01-16 19:55:27	0x050d69a0		
54	Thrd	4	2008-01-16 19:51:48	0x051aada8		
64	Proc	604	2008-01-16 14:33:21	0x051dada0	0x31bfd000	AszMon.exe
75	Thrd	1252	2008-01-16 14:34:05	0x051e5890		
76	Thrd	1252	2008-01-16 14:34:05	0x051e5b08		
77	Proc	624	2008-01-16 14:33:21	0x051e66d8	0x31d83000	TPHDEXLG.exe
81	Thrd	352	2008-01-16 14:33:20	0x051f1460		
82	Proc	352	2008-01-16 14:33:20	0x051f16d8	0x315df000	mdm.exe
83	Thrd	480	2008-01-16 14:33:21	0x051f1b30		
134	Thrd	4	2008-01-16 19:51:48	0x06006da8		
137	Proc	952	2008-01-16 14:33:02	0x06017020	0x2305a000	csrss.exe
166	Thrd	1672	2008-01-16 14:33:18	0x06033730		
167	Proc	300	2008-01-16 14:33:20	0x06033b78	0x30b48000	EvtEng.exe
168	Thrd	1020	2008-01-16 14:33:05	0x06034020		
176	Thrd	4	2008-01-16 19:56:17	0x06047020		
177	Proc	480	2008-01-16 14:33:21	0x06049500	0x3192f000	RegSrvc.exe

[그림 7] PTFinder[24]을 통한 프로세스 정보 추출 결과

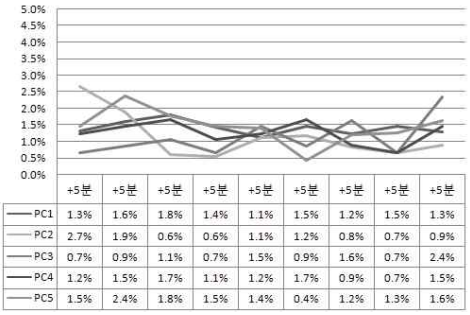
물리 메모리 덤프 영역에서 프로세스 카빙을 통해 프로세스 정보를 추출하는 도구로는 PTFinder[24], Window memory forensic toolkit[25], Window IR tool[26], Memparser[27] 등이 있다. 하지만 각 운영체제 버전에 따라 EPROCESS 구조체가 다르기 때문에 각 도구마다 지원하는 운영체제가 다르며 /PAE, /3GB[28][29]와 같은 옵션을 지원하지 않아 더 정확한 추출을 위한 연구가 진행이 필요하다.

나. 물리 메모리 검증

하드 디스크와 같은 저장 매체의 증거 수집의 경우 전원 차단, 분리, 쓰기 방지 장치 부착, 복사, Hash와 같은 과정을 통해 증거의 무결성 보장 과정을 거쳐 법정에서 증거로 인정받을 수 있다. 물리 메모리 영역 수집의 경우 휘발성 때문에 구동중인 활성 상태에서 수집이 이루어진다. 하지만 활성 상태의 시스템은 메모리 영역의 데이터가 끊임없이 변하기 때문에 연속적으로 메모리 영역을 수집하여도 다른 Hash 값을 보이며 기존 저장 매체와 같은 무결성 보장 과정을 가지기 어렵다. 이는 많은 증거를 담고 있을 수 있는 물리 메모리 영역 데이터의 법적 효력을 약화 시킨다. 이와 같은 물리 메모리의 특성을 확인하기 위해 수집한 물리 메모리 덤프 파일을 대상으로 Page(4,096 Bytes)단위의 Hash를 통해 동일한 대상의 메모리 영역의 변화를 시간차 별로 확인하고 다른 대상의 시스템과의 차이를 확인하였다.

· 동일 대상의 물리 메모리 변화

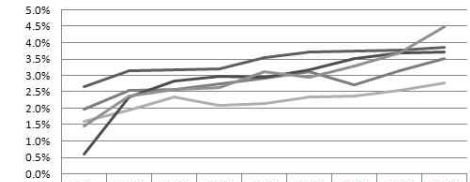
동일 대상의 물리 메모리의 수집 시간 별 변화량을 측정하기 위해 2,048MB의 메모리가 장착된 5개의 시스템을 대상으로 5시간 이상 운용 후 메모리 영역 수집 소프트웨어를 이용하여 5분 간격으로 물리 메모리 영역을 10번 수집하였다. 정확한 측정을 위해 Page(4,096 Bytes) 단위의 메모리 영역을 MD5로 Hash하여 그 차이를 통해 변화율을 계산할 수 있는 스크립트[30]를 구성해 변화율을 측정하였다. 아래 [그림 8]은 5개의 각 수집 대상의 5분 간격 메모리 변화율을 나타낸 것이다.



	+5분	+5분	+5분	+5분	+5분	+5분	+5분	+5분	+5분
PC1	1.3%	1.6%	1.8%	1.4%	1.1%	1.5%	1.2%	1.5%	1.3%
PC2	2.7%	1.9%	0.6%	0.6%	1.1%	1.2%	0.8%	0.7%	0.9%
PC3	0.7%	0.9%	1.1%	0.7%	1.5%	0.9%	1.6%	0.7%	2.4%
PC4	1.2%	1.5%	1.7%	1.1%	1.2%	1.7%	0.9%	0.7%	1.5%
PC5	1.5%	2.4%	1.8%	1.5%	1.4%	0.4%	1.2%	1.3%	1.6%

[그림 8] 5분 간격의 물리 메모리 데이터 변화율

수집된 2,147,483,648Bytes의 메모리 덤프에서 얻은 524,288개의 Hash 값들의 비교를 통해 위와 같은 결과를 얻을 수 있었다. 위의 결과를 통해 동일 시스템의 물리 메모리의 경우 5분당 약 3% 미만의 변화율을 가지는 것을 확인할 수 있었다.



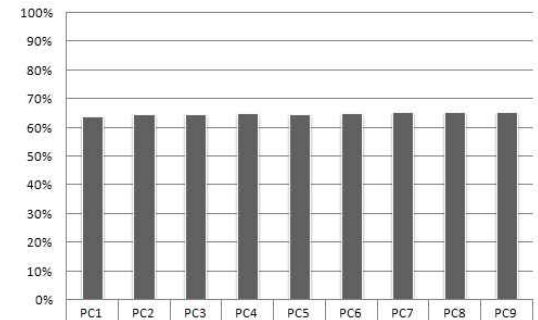
	5분	10분	15분	20분	25분	30분	35분	40분	45분
PC1	2.7%	3.2%	3.2%	3.2%	3.6%	3.7%	3.8%	3.8%	3.9%
PC2	1.6%	1.9%	2.3%	2.1%	2.2%	2.4%	2.4%	2.5%	2.8%
PC3	2.0%	2.5%	2.6%	2.8%	2.9%	3.1%	2.7%	3.2%	3.5%
PC4	0.6%	2.4%	2.8%	3.0%	3.0%	3.2%	3.5%	3.7%	3.7%
PC5	1.5%	2.4%	2.6%	2.6%	3.1%	2.9%	3.3%	3.7%	4.5%

[그림 9] 시간의 흐름에 따른 메모리 영역 변화율

수집된 데이터들의 시간의 흐름에 따른 변화율을 확인하기 위해 첫 번째 수집 데이터와 비교하였다. 시간의 흐름에 따라 변화량이 증가하는 것을 확인할 수 있었으며 50분 후 수집된 메모리 영역의 데이터도 5% 이내의 변화율을 가지는 것을 확인하였다.

· 다른 대상의 물리 메모리 비교

다른 대상에서 수집한 메모리의 차이를 확인하기 위해 2,046MB 메모리의 Windows XP SP2가 설치된 10대의 시스템에서 물리 메모리를 수집하였다. 위와 마찬가지로 Page(4,096 Bytes) 단위의 메모리 영역을 Hash하여 차이율을 확인하였으며 그 결과는 [그림 10]과 같다.



	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
차이율	64.1%	64.8%	64.8%	65.0%	64.8%	65.1%	65.3%	65.3%	65.3%

[그림 10] 다른 시스템과의 메모리 영역 데이터 차이율

확인 결과 Windows XP SP2의 경우 각 시스템은 63% 전후의 차이율을 가지고 있었으며 이는 운영체제 영역을 제외한 나머지 유저 영역의 구동중인 응용 프로그램에서 발생하는 차이인 것을 확인할 수 있었다. 동일한 운영체제를 대상으로 약 35% 가량의 동일한 영역을 보이는 것으로 다른 운영체제의 경우 더 많은 차이를 보일 수 있다는 것을 유추할 수 있다.

6. 결론 및 향후 연구 방향

물리 메모리 영역에는 증거로 활용될 수 있는 프로세스 정보와 이름, ID, 비밀 번호, 전자 메일 주소 등의 정보를 담고 있다. 하지만 운용중인 시스템에서 발생하는 운영체제와 응용 프로그램의 동작에 의한 지속적인 데이터의 변화로 무결성 및 동일한 대상에서 수집되었다는 것을 확인하기 어렵다. 이에 본 논문에서는 물리 메모리 영역 수집 기법과 분석 기법을 알아보고 하드웨어 기반 수집 도구를 구현하여 물리 메모리의 변화량과 다른 대상과의 차이를 확인하였다.

또 수사 대상의 물리 메모리 영역 접근 가능성을 확인하였음으로 이를 통해 직접적으로 특정 프로세스 영역 추출, 시그니처 조사를 통한 사용자 계정 정보와 메신저 대화 내역 정보 추출 등의 연구를 진행하고 있으며 리눅스 기반이 아닌 win32 기반의 수집 시스템을 구축하기 위한 구현을 진행 중에 있다. 국내의 경우 수사 시 전원 차단 후 하드디스크와 같은 저장 매체 수집 절차를 수행하기 때문에 일반적으로 물리 메모리 수집은 이루어지지 않는다. 국내의 경우 법적 분쟁에서 디지털 증거의 유효성에 대한 법제가 마련되어있지 않아 증거로 인정받기 힘들다. 국내에서도 디지털 포렌식 기술이나 절차의 표준화 과정이 필요하다.

참고 문헌

[1] Casey E. Digital evidence and computer crime: forensic science, computer and the Internet. Boston: Academic Press; 2000.

[2] G. Palmer, "A Road Map for Digital Forensic Research", Utica, New York, technical report DTR-T001-0, 2001.

[3] RFC3227, "Guidelines for Evidence Collection and Archiving", Available from http://www.faqs.org/rfcs/rfc3227.html, 2002.

[4] S. Stover and M. Dickerson, "Using Memory Dumps in Digital Forensics," Login: Magazine 30, no 6 (Dec 2005): 43 - 48.

[5] Valli, C., & Patak, P. (2005). An investigation into the efficiency of forensic erasure tools for hard disk mechanisms. Paper presented at the 3rd Australian Computer, Information and Network Forensics Conference, Edith Cowan University, Perth, Western Australia.

[6] IEEE1394, Standard for High Performance Serial Bus, 1995.

[7] Mark E Russinovich, David A Solomon, Microsoft Windows Internals, Fourth Edition: Microsoft Windows Server 2003, Windows XP, and Windows. Microsoft Press, 2004

[8] S Lee, A Savoldi, S Lee, J Lim, Windows Pagefile Collection and Analysis for a Live Forensics Context, Future Generation Communication and Networking, 2007.

[9] Jim Chow, Ben Pfaff, Tal Garfinkel, Mendel Rosenblum, "Shredding Your Garbage: Reducing Data Lifetime Through Secure Deallocation."14th USENIX Security Symposium, July/August 2005.

[10] WinDD, Disk Dump for Windows, Available from http://sourceforge.net/projects/windd

[11] Microsoft Crash Dump Analysis, Available from http://msdn2.microsoft.com/en-us/library/bb204861(VS.85).aspx

[12] "KB 244139: Windows Feature Allows a Memory Dump File to Be Generated with the Keyboard." Available from http://support.microsoft.com/kb/244139/en-us. Retrieved February 8, 2007.

[13] Microsoft Sysinternals, Mark Russinovich, NotMyFault.exe, Available from http://www.microsoft.com/technet/sysinternals

[14] Microsoft Sysinternals, LiveKD, Available from http://www.microsoft.com/technet/sysinternals/SystemInformation/LiveKd.mspx

[15] Joanna Rutkowska, Rootkits Detection on Windows Systems, IT Underground Conference, October 12th -13th 2004

[16] X-Ways Software Technology AG, X-Ways Capture, Available from http://www.winhex.com/capture/index-m.html

[17] Maximillian Dornseif, 0wned by an iPod - hacking by Firewire, PacSec conference, Japan, Nov 2004

[18] IEEE1212, Information technology Microprocessor systems, Control and Status Registers (CSR) Architecture for Microcomputer buses, 1994.

[19] N Amini, PM Bland, BF Boury, RG Hofmann, TJ Lohman, System direct memory access (DMA) support logic for PCI based computer system, US Patent 5,450,551, 1995.

[20] BD Carrier et al, A Hardware-Based Memory Acquisition Procedure for Digital Investigations, Digital Investigation, 2004

[21] 1394 Open HCI, "1394 Open Host Controller Interface Specification" Release 1.1, Jan 2000

[22] IEEE1394 for Linux, Available from http://www.linux1394.org

[23] N. Petroni, A.Walters, T. Fraser, andW. Arbaugh. Fatkit: A framework for the extraction and analysis of digital forensic data from volatile system memory. Digital Investigation, 3:197 - 210, 2006.

[24] Andreas Schuster, "PTFinder." Available from http://computer.forensikblog.de/en

[25] Mariusz Burdach, "Windows Memory Forensic Toolkit." Available from http://forensic.secure.net

[26] Harlan Carvey, "Windows IR/CF Tools." Available from http://sourceforge.net/projects/windowsir

[27] Chris Betz, "Memparser." Proceedings of the 14th annual USENIX Security Symposium, pp. 331 - 346. Available from http://sourceforge.net/projects/memparser

[28] "Memory Support and Windows Operating Systems." Available from http://www.microsoft.com/whdc/system/platform/server/PAE/PAEmem.mspx

[29] Intel 64 and IA-32 Architectures Software Developer's Manual: Volume 3A: System Programming Guide Part 1 (Intel Corp., 2006).

[30] Python Programming Language - Official Website, Available from http://www.python.org