Journal of The Korea Institute of Information Security & Cryptology(JKIISC) ISSN 1598-3986(Print)
VOL.23, NO.5, October 2013 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2013.23.5.931

디스크 암호화 키의 효율적인 탐색을 위한 커널 메모리 수집 방법

강 영 복,[1†] 황 현 욱,[2‡] 김 기 범,[2] 이 경 호,[1] 김 민 수,[3] 노 봉 남[1]
[1]전남대학교 시스템보안연구센터, [2]ETRI 부설연구소, [3]목포대학교

A kernel memory collecting method for efficent disk encryption key search

Youngbok Kang,[1†] Hyunuk Hwang,[2‡] Kibom Kim,[2] Kyoungho Lee,[1] Minsu Kim,[3] Bongnam Noh[1]
[1]Chonnam National University System Security Research Center, [2]The Attached Institute of ETRI, [3]Mokpo National University

요 약

디스크 암호화 소프트웨어로 데이터를 암호화 하는 경우 패스워드를 획득하기 전까지 암호화 데이터의 원본 데이터를 추출하기 위해서는 많은 어려움이 있다. 이러한 디스크 암호화 소프트웨어의 암호화 키는 물리 메모리 분석을 이용하여 암호화 키를 추출할 수 있다. 물리 메모리에서 암호화 키 탐색을 수행하는 경우 일반적으로 메모리 전체를 대상으로 탐색을 수행하기 때문에 메모리 크기에 비례하여 많은 시간이 요구된다. 하지만 물리 메모리 데이터에는 시스템 커널 오브젝트, 파일 데이터와 같이 암호화 키와 관련이 없는 많은 데이터가 포함되어 있음으로, 이를 분석하여 키 탐색에 필요한 유효한 데이터를 추출하는 방법이 요구된다.

본 논문에서는 윈도우즈 커널 가상 주소 공간 분석을 통해 물리 메모리에서 디스크 암호화 키가 저장되는 메모리 영역만 수집하는 효율적인 방법을 제시하고자 한다. 실험을 통해 제안된 방법이 기존 방법보다 암호화 키 탐색 공간을 효율적으로 줄임으로써 우수함을 증명한다.

ABSTRACT

It is hard to extract original data from encrypted data before getting the password in encrypted data with disk encryption software. This encryption key of disk encryption software can be extract by using physical memory analysis. Searching encryption key time in the physical memory increases with the size of memory because it is intended for whole memory. But physical memory data includes a lot of data that is unrelated to encryption keys like system kernel objects and file data. Therefore, it needs the method that extracts valid data for searching keys by analysis. We provide a method that collect only saved memory parts of disk encrypting keys in physical memory by analyzing Windows kernel virtual address space. We demonstrate superiority because the suggested method experimentally reduces more of the encryption key searching space than the existing method.

Keywords: Physical Memory, Disk Encryption, Kernel Memory

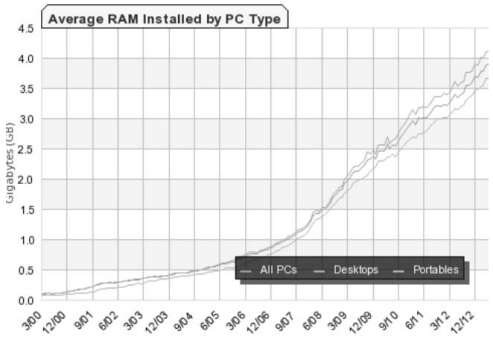
I. 서 론

최근 개인 정보나 기업의 주요 정보를 보호하기 위해 디스크 암호화 도구 사용이 증가하고 있다. 암호화 도구의 수요 증가로 디스크 보안 소프트웨어 시장에서

접수일(2013년 8월 7일),수정일(2013년 9월 17일), 게재확정일(2013년 10월 7일)
† 주저자, k-dupe@nate.com
‡ 교신저자,hhu@ensec.re.kr(Corresponding author)

는 기업을 대상으로 하는 엔터프라이즈 상품 및 오픈 소스로 개발되는 무료 소프트웨어의 사용이 증가하고 있다. 암호화 디스크 도구는 개인 정보 보호 목적뿐만 아니라 범죄 은닉 목적[1]으로도 사용하고 있어 사이버 범죄 수사에 있어 많은 어려움이 발생하고 있다.

패스워드를 모르는 상황에서 암호화가 적용된 디스크를 복호화하기 위한 방법으로는 물리 메모리 덤프 데이터를 이용한 복호화 방법을 사용한다[2]. 물리 메모리에서 암호화 키를 탐색하는 방법으로는 키 저장 데이터 영역의 구조적 특징을 이용하여 추출하는 방식[3]과 물리 메모리에서 1바이트 단위로 암호화 키를 전수 조사하는 방법이 있다. 위 두 가지 암호화 키 조사 방법에서는 물리 메모리 전체 데이터를 대상으로 검사를 수행하기 때문에 메모리 크기에 따라 탐색 시간이 크게 좌우 된다. 또한 최근 들어 64비트 PC가 보편적으로 사용됨에 따라 그림 1과 같이 메모리 용량도 증가하면서, 물리 메모리에서 암호화 키 탐색 수행 시간이 점차 증가할 전망이다. 따라서 대용량 물리 메모리에서 디스크 암호화 키를 효과적으로 추출하는 방법이 필요하다.



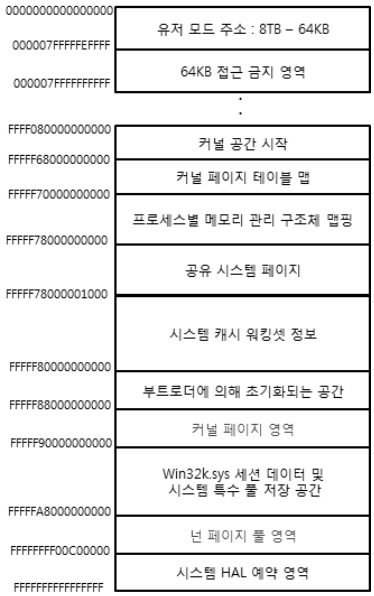
[그림 1] PC에서 사용하는 메모리 크기 추세[4]

본 논문에서는 메모리 사용 용량이 지속적으로 증가하고 있는 윈도우7 64비트 물리 메모리 데이터를 대상으로 커널 페이지 테이블 분석 방법을 이용한 유효한 암호화 키 탐색 메모리 데이터 추출 방법을 제안한다.

II. 관련 연구

2.1 윈도우 64비트 가상 주소 공간 분석

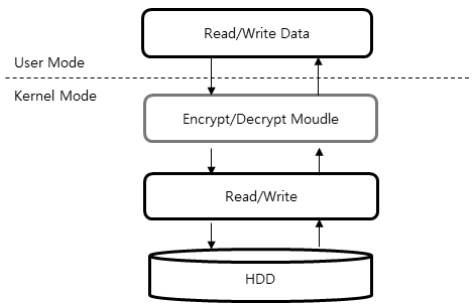
윈도우즈 64비트 가상 주소 공간 배치[5]는 그림 3



[그림 2] 윈도우 64비트 가상 주소 공간 배치

과 같이 유저 영역은 0x0000000000000000로 시작하고 0x00007FFFFFFFFFFF로 끝난다. 커널 영역은 가상 주소 0xFFFF080000000000로 시작하고 0xFFFFFFFFFFFFFFFF로 끝난다.(AMD와 Intel에서 지원하는 x64 프로세서에서는 물리 메모리를 위해 48비트만 사용하고 나머지 16비트는 사용하지 않음)

윈도우즈 커널 가상 주소 공간은 각각의 가상 주소 영역마다 저장될 데이터 영역이 지정되어 있다. 윈도우 커널 드라이버에서 사용하는 메모리 영역에는 커널 스택, 커널 힙 메모리, 페이지 풀, 넌 페이지 풀 영역이 있다. 커널 스택, 커널 힙 메모리, 페이지 풀 데이터는 가상주소 0xFFFFF88000000000~ 0xFFFFF90000000000 커널 페이지 영역에 저장된다. 넌



[그림 3] 드라이버 기반 OTFE 구조

페이지 풀 데이터는 가상 주소 0xFFFFFA8000000000 ~ 0xFFFFFFFF00C00000 넌 페이지 풀 영역에 저장된다.

2.2 디스크 암호화 도구 구조

윈도우 기반 디스크 암호화 도구에서는 사용자에게 암/복호화 편의성을 제공하기 위해 On-The-Fly Encryption[6](이하 OTFE)를 기반으로 동작한다. 대표적인 오픈소스 암호화 도구인 트루크립트[7]와 BitLocker[8]에서도 드라이버 모듈을 이용한 OTFE 방식을 제공한다. OTFE 암호화 방식에서는 데이터를 디스크 볼륨에서 가져올 때 복호화하고, 저장할 때 암호화를 수행하기 때문에 실시간 암호화 방식(Real-Time Encryption)이라고 한다. 윈도우 시스템에서 OTFE 방식을 지원하기 위해 그림 3과 같이 드라이버를 이용한 디스크 I/O 후킹을 수행한다. 드라이브 후킹 기반 OTFE를 지원하기 위해서는 운영체제에서 항상 암/복호화 드라이브를 시스템에 로드해야 한다.

2.3 디스크 암호화 도구 암호화 키 관리

윈도우즈 드라이버를 이용한 OTFE 암호화 구조에서는 드라이버 모듈에서 암/복호화를 처리하기 때문에 드라이버에서 암호화 키를 관리한다. 드라이버에서 암호화 키를 관리하기 때문에 암호화 키는 항상 커널 메모리 영역에 로드된다. 일반적으로 디스크 암호화 도구에서는 암호화 키의 보안 유지를 위해 커널 메모리에 로드되는 암호화 키가 하드디스크에 페이징이 되지 않도록 설정한다[9]. 윈도우즈 시스템에서는 페이징이 되지 않는 메모리 할당 및 설정을 위해 ExAllocatePoolWithTag[10]과 VirtualLock[11] API를 제공한다. 위 API는 그림 2의 가상 메모리 구조에서 드라이버에서 접근 가능한 커널 페이지 영역, 넌 페이지 풀 메모리 영역에 대해 페이징 옵션을 제공한다. ExAllocatePoolWithTag 함수는 그림 2의 가상 메모리 구조에서 커널 넌 페이지 풀 메모리 또는 커널 페이지 영역에 페이징이 되지 않는 메모리 공간을 할당 한다. VirtualLock 함수는 힙 메모리로 할당된 메모리 영역을 페이징이 되지 않도록 설정한다. 커널 드라이버에서 VirtualLock 함수를 사용하는 경우 그림 2의 가상 메모리 구조에서 커널 페이지 영역에 페이징이 되지 않는 메모리 영역을 설정하게 된다.

2.4 암호화 키 저장 메모리 영역 데이터 수집 방법

윈도우기반 OTFE 방식으로 동작되는 디스크 암호화 도구에서 사용하는 암호화 키를 물리 메모리에서 탐색하기 위해서는 커널 메모리 데이터 영역을 추출해야 한다. 물리 메모리에서 커널 메모리 데이터를 추출하는 방법으로는 풀 헤더 카빙[9]을 이용한 방법과 커널 페이지 테이블 분석[12]을 이용한 추출 방법이 있다. 풀 헤더 카빙을 이용하여 커널 데이터를 수집하는 방법은 커널 풀 헤더 구조체에서 사용하는 풀 태그를 이용하여 추출하는 방식이다. 풀 헤더 카빙을 이용하여 수집된 메모리에는 커널에서 사용한 풀 메모리 공간만 수집되기 때문에 완벽한 커널 메모리 영역 이라고 할 수 없으며, 풀 헤더 카빙수행과정에서 사용하는 풀 태그 값을 알고 있지 않는 상태에서는 카빙을 수행할 수 없다는 문제점이 있다. 또한 암호화 키가 커널 풀 메모리 영역에 저장되지 않는 경우 수집된 풀 메모리에서 암호화 키 탐색을 수행 할 수 없다는 단점이 있다.

페이지 테이블 분석을 이용한 커널 메모리 수집 방법은 커널 프로세스의 페이지 테이블 분석을 이용하여 물리 메모리에서 커널 메모리 영역만을 추출하는 방식이다. 커널 페이지 테이블 분석을 수행하기 위해서 윈도우 커널 프로세스 역할을 하는 System 프로세스의 _EPROCESS 구조체에 저장된 DTB(Directory Table Base)를 이용한다. 페이지 테이블 방식을 이용하여 추출된 커널 메모리에는 커널 시스템 정보, 드라이버 파일 데이터와 같은 모든 커널 메모리 데이터가 포함되어있기 때문에 키 탐색에 유효하지 않는 영역이 많이 포함되어 있다는 단점이 존재한다.

III. 키 탐색 메모리 영역 수집을 위한 커널 메모리 분석

3.1 커널 메모리 사용 용량

32비트 윈도우 시스템에서는 최대 4GB까지 물리 메모리 사용을 지원하며, 64비트에서는 최대 2,048GB 메모리 사용을 지원한다. 표 1은 커널 메모리 사용 측정 도구[13]를 이용하여 윈도우 7 64비트 환경에서 사용하는 커널 메모리 사용 크기 조사 결과이다.

물리 메모리에서 커널이 사용하는 공간을 측정해본

〔표 1〕 물리 메모리 크기별 커널 사용 공간

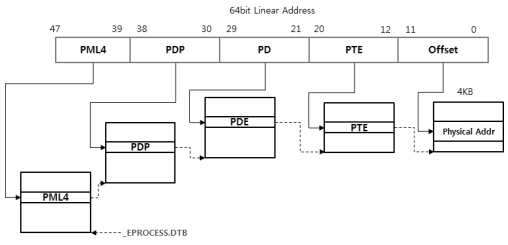
물리 메모리 크기	평균 커널 사용 공간	커널 메모리 비율
1GB	204MB	19.37%
2GB	209MB	10.20%
4GB	538MB	13.14%
8GB	554MB	6.77%
16GB	1120MB	6.83%

결과 전체 물리 메모리 크기에 비해 차지하는 비율이 낮은 것을 알 수 있다. 따라서 물리 메모리 전체를 대상으로 암호화 키 탐색을 수행하는 것은 비효율적이며, 또한 물리 메모리 크기가 증가할수록 커널 공간도 같이 증가하기 때문에 커널 메모리 영역에서 유효한 암호화 키 탐색 메모리 데이터를 추출하는 방법이 필요함을 알 수 있다.

3.2 커널 영역 가상 메모리 주소 복구

물리 메모리에서 그림 2와 같이 커널 가상 주소 영역별로 메모리 데이터를 수집하기 위해서는 페이지 테이블 분석 방법〔12〕을 이용해야한다. 기존 페이지 테이블 분석을 이용한 커널 메모리 공간 추출 방법에서는 페이지 테이블에 명시된 물리 메모리 오프셋 주소를 참조하여 물리 메모리에서 커널 데이터를 추출하는 방식이다. 커널에서 사용하는 가상 메모리 주소를 얻기 위해서는 기존 커널 메모리 영역 수집과 다르게 커널 페이지 테이블에서 참조하는 인덱스를 사용해야 한다. 64비트 가상 주소〔14〕에서는 그림 4와 같이 12비트에서부터 47비트 까지 PTE(Page-Table Offset), PD(Page-Directory Offset), PDP(Page-Directory Pointer Offset), PML4(Page-Map Level4 Offset)이 페이지 테이블 인덱스를 가리키는 용도로 사용된다.

PML4의 베이스 주소는 해당 프로세스의 EPROCESS 구조체 멤버 변수 DTB에 저장되어 있다. PML4, PDP, PDE, PTE 테이블에는 다음 테이블에 대한 베이스 주소가 명시 되어 있다. 테이블 순회 과정에서 사용된 각 테이블 인덱스 값은 가상 주소에서의 비트 영역의 값과 매칭 된다. 커널 가상 메모리 주소를 복구하기 위해서는 DTB에 명시된 PML4 테이블 베이스 주소를 얻어 테이블을 순회해야 한다. 테이블 순회를 수행하면서 각 테이블에서 사용한 인덱스 값을 비트 연산을 통해 64비트 가상 주소로 변환할 수 있다.



〔그림 4〕 64비트 가상 주소 구조(4KB 페이지 기법)

3.3 커널 드라이버 정보 추출

그림 2의 커널 페이지 영역에는 드라이버에서 사용하는 스택, 힙 메모리 데이터뿐만 아니라 커널에 로드된 드라이버 파일 데이터도 저장된다. 커널 드라이버 파일 데이터는 암호화 키 탐색 영역이 아니므로 메모리 수집 과정에서 제외 되어야 한다. 커널 페이지 영역에서 커널 드라이버 파일 데이터를 제외하기 위해서는 먼저 커널 페이지 영역에 로드 된 드라이버 정보를 추출해야 한다. 물리 메모리에서 커널 드라이버 정보를 추출하기 위해서는 드라이버 구조체 카빙 방법〔15〕을 이용해야 한다. 물리 메모리에서 드라이버 구조체 카빙을 수행하면 커널에 로드된 모든 드라이버에 대한 _DRIVER_OBJECT 구조체 정보를 얻을 수 있다. _DRIVER_OBJECT 구조체 멤버에는 그림 5와 같이 커널 페이지 영역에 로드된 드라이버 모듈의 베이스 주소(DriverStart) 및 가상 메모리에 로드된 모듈 크기(DriverSize) 정보가 저장되어 있다.

```
kd> dt _DRIVER_OBJECT
ntdll!_DRIVER_OBJECT
   +0x000 Type             : Int2B
   +0x002 Size             : Int2B
   +0x008 DeviceObject     : Ptr64 _DEVICE_OBJECT
   +0x010 Flags            : Uint4B
   +0x018 DriverStart      : Ptr64 Void
   +0x020 DriverSize       : Uint4B
   +0x028 DriverSection    : Ptr64 Void
   +0x030 DriverExtension  : Ptr64 _DRIVER_EXTENSION
   +0x038 DriverName       : _UNICODE_STRING
   +0x048 HardwareDatabase : Ptr64 _UNICODE_STRING
   +0x050 FastIoDispatch   : Ptr64 _FAST_IO_DISPATCH
   +0x058 DriverInit       : Ptr64     long
   +0x060 DriverStartIo    : Ptr64     void
   +0x068 DriverUnload     : Ptr64     void
   +0x070 MajorFunction    : [28] Ptr64     long
```

〔그림 5〕 DRIVER_OBJECT 구조체 정보

DRIVER_OBJECT 구조체 멤버 DriverSize에 명시된 크기 정보는 윈도우즈 최소 페이지 크기인 4KB 단위로 저장되어 있다. 드라이버 모듈의 베이스 주소 값과 메모리에 로드된 크기 정보를 이용하면 커널 페이지 영역에 로드된 드라이버 파일 데이터 영역 정보를 알 수 있다.

IV. 제안하는 암호화 키 탐색 메모리 수집 방법

4.1 수집 메모리 영역

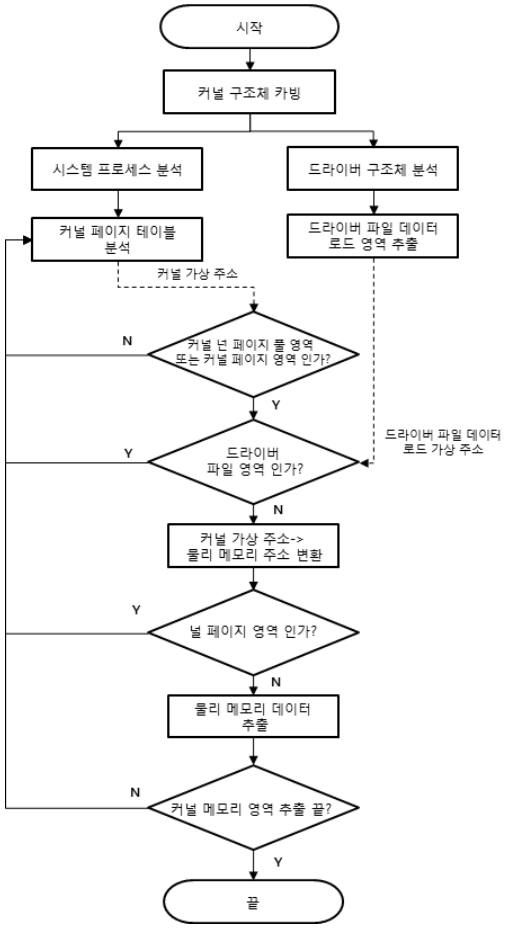
제안하는 방법에서는 커널 가상 주소 분석을 통해 디스크 암호화 도구의 드라이버에서 사용할 수 있는 커널 페이지 영역 및 커널 넌 페이지 영역만을 수집하며, 암호화 키 탐색에 불필요한 드라이버 파일 데이터 및 초기화된 널 메모리 데이터는 수집에서 제외한다. 기존 커널 메모리 수집 방법과 비교하였을 때 제안하는 방법에서는 암호화 드라이버에서 사용할 수 있는 모든 메모리 영역만을 수집하기 때문에 모든 윈도우기반 OTFE 암호화 도구의 암호화 키 탐색 메모리 영역 수집에 사용될 수 있다는 장점이 있다.

4.2 수집 과정

물리 메모리에서 암호화 키 탐색 대상 메모리 수집은 그림 6과 같이 커널 구조체 카빙을 수행하는 것으로 시작한다. 커널 구조체 카빙 단계에서는 물리 메모리 전체를 대상으로 프로세스 구조체 카빙[16] 및 커널 드라이버 구조체 카빙[15]을 수행한다. 드라이버 구조체 카빙을 통해 물리 메모리에서 사용된 드라이버가 로드 된 가상 주소 영역 값을 추출한다.

시스템 프로세스 분석 단계에서는 시스템 커널 프로세스인 system 프로세스를 대상으로 커널에서 사용하는 페이지 테이블 정보를 추출한다. 커널 페이지 테이블 분석 단계에서는 커널 페이지 테이블 순회를 통해 커널 메모리에서 사용하는 가상 주소 영역을 추출한다. 추출된 커널 가상 주소는 커널 넌 페이지 풀 영역 또는 커널 페이지 영역 여부를 검사한다. 추출된 커널 가상 주소가 검사 대상 영역에 속하는 경우, 드라이버 파일이 로드된 가상 주소 영역인지를 확인한다. 검사 대상 커널 가상 주소가 드라이버 파일 영역이 아닌 경우 해당 커널 가상 주소를 물리 메모리 주소로 변환한다. 변환된 물리 메모리 주소를 이용하여 해당 메모리 데이터 영역에 접근 및 널 페이지 메모리 영역인지를 확인한다. 추출 메모리 영역이 널 영역이 아닌 경우 해당 메모리 데이터를 추출하고, 널 영역인 경우 커널 페이지 테이블에서 다시 커널 가상 주소를 추출한다. 위 작업은 커널 가상 메모리 영역 추출이 완료될 때 까지 반복 추출한다.

제안한 방법을 이용하면 암호화 키 탐색에 불필요한 커널 시스템 메모리 데이터, 드라이버 파일 데이터, 널 메모리 데이터를 포함하지 않으면서 암호화 키 값이 저장될 수 있는 힙 메모리 영역, 스택, 커널 페이지 풀, 넌 페이지 풀 메모리 영역만을 수집할 수 있다.



[그림 6] 디스크 암호화 키 탐색 메모리 영역 수집 방법

V. 시험 및 검증

본 논문에서 제안한 방법을 검증하기 위해 윈도우기반 OTFE 암호화 도구인 트루크립트[7]와 윈도우 시스템에서 제공하는 BitLocker[8] 도구를 대상으로 윈도우 7 64비트 운영체제 환경에서 실험을 수행하였다.

5.1 키 탐색 메모리 영역 수집 실험

암호화 키 탐색 영역 수집 실험을 위해 트루크립트와 BitLocker의 암호화 볼륨이 각각 마운트 되어 있

는 시스템에서 물리 메모리 덤프를 수행한 후, 덤프된 물리 메모리 데이터에서 키 탐색 메모리 영역 수집 실험을 수행하였다. 실험에 사용된 물리 메모리는 크기가 다른 5종의 메모리를 사용하였다. 본 논문에서 제안한 방법의 효율성 테스트를 위해 페이지 테이블을 이용한 커널 전체 메모리 수집 방법[12]과 수집 용량, 수집 시간 비교를 수행하였다. 기존 커널 메모리 데이터 수집 방법 중 하나인 풀 헤더 카빙 방법[9]은 암호화 키 탐색 대상 제품에서 사용하는 풀 헤더 시그니처를 알고 있어야 하는 문제점이 있으므로 비교 실험에서 제외하였다. 제안하는 방법은 기존 전체 커널 메모리 수집방법과 동일하게 커널 구조체 카빙을 수행하였다. 실험 결과 표 2에서 구조체 카빙 시간은 전체 메모리에서 커널 구조체 카빙만을 수행하는 시간을 뜻하며, 수집 시간은 분석 시간과 커널 구조체 카빙시간이 포함된 시간 결과이다. 실험 결과 표 2와 같이 커널 구조체 카빙은 물리 메모리 전체르 대상으로 커널 구조체를 탐색하는 과정이기 때문에 전체 메모리 수집 시간에서 큰 비중을 차지하는 것을 알 수 있다.

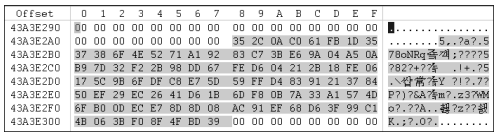
표 2에서 제안하는 방법은 물리 메모리에서 암호화 키가 저장될 수 있는 영역만을 수집하기 때문에 커널 전체 메모리를 수집한 결과 보다 수집된 메모리 크기가 크게 감소하는 것을 알 수 있다. 또한 물리 메모리 크기가 커질수록 전체 커널 메모리 크기와 제안하는 방법의 수집된 메모리 크기의 차이가 더 커지며, 수집 시간은 더 줄어드는 것을 알 수 있다. 또한 전체 물리 메모리 크기가 커질수록 제안하는 방법과 기존 전체 메모리를 수집한 결과의 차이가 증가하며 수집 시간은 더 줄어드는 것을 알 수 있다. 실험 결과 전체 물리 메모리 크기가 커질수록 본 논문에서 제안하는 방법이 시간과 공간적인 측면에서 기존 방법보다 더 우수한 것을 알 수 있다.

[표 2] 암호화 키 탐색 메모리 영역 수집 결과 비교

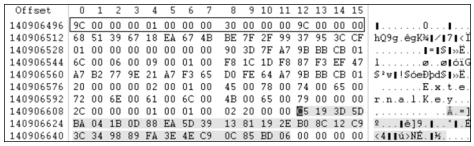
물리 메모리 크기	커널 구조체 카빙 시간	커널 페이지 테이블 분석 기반 수집 크기 (전체 커널 메모리)		제안한 방법을 이용한 수집 크기	
		수집 메모리 크기	수집 시간	수집 메모리 크기	수집 시간
1GB	35s	211MB	42s	124MB	45s
2GB	60s	230MB	76s	139MB	77s
4GB	121s	461MB	144s	117MB	141s
8GB	255s	635MB	288s	121MB	280s
16GB	598s	1493MB	1449s	561MB	1236

5.2 수집된 메모리를 이용한 암호화 키 추출 검증

제안한 방법으로 수집된 메모리 데이터의 암호화 키 저장 유무를 확인하기 위해 Password Recovery Kit[17]를 이용하였다. Password Recovery Kit 도구를 이용하여 그림 7과 그림 8과 같이 수집된 메모리 영역에 저장된 트루크립트, BitLocker 암호화 키를 성공적으로 추출 하였다.



[그림 7] 수집 메모리에 저장된 트루크립트 암호화 키 데이터



[그림 8] 수집 메모리에 저장된 BitLocker 암호화 키 데이터

암호화 키 추출 검증 실험 결과 물리 메모리 크기에 따라 수집된 모든 메모리 데이터에서 디스크 암호화 암호화 키를 성공적으로 추출할 수 있었다.

VI. 결 론

본 논문에서는 물리 메모리 데이터에서 효율적인 디스크 암호화 키 탐색을 위한 유효한 메모리 영역 수집 방법을 제안을 하였다. 또한 트루크립트[7], BitLocker[8] 도구를 대상으로 암호화 키 탐색 메모리 영역 수집 실험을 하였다. 실험 결과 물리 메모리가 커질수록 기존 수집 방법보다 시간과 공간적인 측면에서 더 우수한 것을 알 수 있었다.

제안하는 방법을 이용하면 커널 드라이버에서 사용가능한 모든 영역을 추출하기 때문에 모든 윈도우즈 드라이버 기반 OTFE 암호화 도구를 대상으로 암호화 키 메모리 영역 수집을 수행할 수 있는 장점이 있다. 하지만 제안한 방법은 기존 커널 정보 수집 방법과 동일하게 물리 메모리 전체를 대상으로 커널 구조체 카빙이 선행되어야 하는 단점이 있다. 향후 연구에서는 물리 메모리 분석에서 수행되는 카빙 문제점을

해결하기 위한 빠른 메모리 정보 수집 방법에 대한 연구를 수행할 예정이다.

참고문헌

〔1〕 Niedermeier, "In Re Boucher", United States District Court No. 2:06-mj-91 2009 WL 424718, Nov. 2009.

〔2〕 Sasa Mrdovic, Alvin Huseinovic "Forensic Analysis of Encrypted Volumes Using Hibernation File," Telecommunications Forum, pp. 1277 - 1280, Nov. 2011.

〔3〕 Christopher Hargreaves, Howard Chivers "Recovery of Encryption Keys from Memory Using a Linear Scan March," ARES 08. Third International Conference on, pp. 1369-1376, Mar. 2008.

〔4〕 Average RAM, http://techtalk.pcpitstop.com/research-charts-memory/, May. 2008.

〔5〕 마크 러시노비치, 데이비드 솔로몬, 알렉스 이오네스쿠, "Windows Internals 5," 에이콘출판사, pp. 901-912, 2010년 7월

〔6〕 Robin Snyder, "Some Security Alternatives for Encrypting Information on Storage Devices," InfoSecCD 06 Proceedings ofthe 3rd annual conference on Informationsecurity curriculum development, pp. 79-84, 2006.

〔7〕 TrueCrypt, http://www.truecrypt.org

〔8〕 BitLocker, http://windows.microsoft.com/ko-kr/windows7/products/features/bitlocker

〔9〕 Brian Kaplan, Advisor Matthew Geiger, "RAM is Key," Master of Science in Information Security Policy and Management, pp. 14-18, May. 2007.

〔10〕 ExAllocatePoolWithTag, http://msdn.microsoft.com/en-us/library/windows/hardware/ff544520(v=vs.85).aspx

〔11〕 VirtualLock, http://msdn.microsoft.com/en-us/library/windows/desktop/aa366895(v=vs.85).aspx

〔12〕 Volatility memdump, http://code.google.com/p/volatility/wiki/CommandReference22#memdump

〔13〕 ProcessExplorer, http://technet.microsoft.com/ko-kr/sysinternals/bb896653.aspx

〔14〕 AMD64 Technology, "AMD64 Architecture Programmer's Manual Volume 2:System Programming," Advanced Micro Device Inc, Publication No. 24593, pp. 127-144, May. 2013.

〔15〕 Volatility driverscan, http://code.google.com/p/volatility/wiki/CommandReference22#driverscan

〔16〕 Volatility psscan, http://code.google.com/p/volatility/wiki/CommandReference22#psscan

〔17〕 Password Recovery, http://www.lostpassword.com/kit-forensic.htm

〈저자소개〉

강 영 복 (Youngbok Kang) 학생회원
2012년 2월: 전남대학교 전자컴퓨터공학부(공학사)
2012년 2월~현재: 전남대학교 정보보안협동과정 석사과정
〈관심분야〉 디지털 포렌식, 악성코드 탐지, 취약점 분석

사 진

황 현 욱 (Hyunuk Hwang) 정회원
2000년 2월: 조선대학교 정보통신공학과 졸업(공학사)
2002년 2월: 조선대학교 전자공학과 졸업(공학석사)
2004년 8월: 전남대학교 정보보호협동과정 졸업(이학박사)
2004년 9월~현재: ETRI 부설연구소 선임연구원
〈관심분야〉 디지털 포렌식, 악성코드, 사이버보안

사 진

김 기 범 (Kibom Kim) 정회원
1994년 2월: 제주대학교 정보공학과 졸업(공학사)
1996년 8월: 고려대학교 전산과학과 졸업(이학석사)
2001년 2월: 고려대학교 전산과학과 졸업(이학박사)
2004년 1월~2004년 7월: (주) 이씨오 개발부장
2004년 8월~현재: ETRI 부설연구소 선임연구원
〈관심분야〉 디지털 포렌식, 사이버보안, 정보보호

사 진

이 경 호 (Kyoungho Lee) 학생회원
2012년 8월~현재: 전남대학교 정보보안협동과정 (학석사 연계과정)
〈관심분야〉 정보보호, 디지털 포렌식, 악성코드

김 민 수 (Minsoo Kim) 종신회원
1993년: 전남대학교 전산통계학과 (이학사)
1995년: 전남대학교 전산통계학과 (이학석사)
2000년: 전남대학교 전산통계학과 (이학박사)
2000년~2001년: 한국인터넷진흥원 선임연구원
2001년~2004년: 전남대학교 연구교수
2005년~현재: 목포대학교 정보보호학과 부교수
〈관심분야〉 침입탐지, 디지털 포렌식, 데이터마이닝, 악성코드 분석

노 봉 남 (Bongnam Noh) 종신회원
1987년: 전남대학교 수학교육과 (이학사)
1982년: KAIST 전산학과 (이학석사)
1994년: 전북대학교 전산과 (이학박사)
1983년~현재: 전남대학교 전자컴퓨터공학부 교수
2000년~현재: 시스템보안연구센터 소장
〈관심분야〉 디지털 포렌식, 시스템 및 네트워크 보안, 정보사회와 사이버 윤리