실머신 기반 악성코드 자동 분석 시스템에서의 메모리 덤프

나재찬, 김현우, 조영훈, 윤종희,
영남대학교 컴퓨터공학과
e-mail : youn@yu.ac.kr

Memory Dump of Automated Malware Analysis System based on Real Machine

Jaechan Na, Hyunwoo Kim, Younghun Jo, Jonghee M. Youn
Dept. of Computer Engineering, Yeungnam University

요 약

쿠쿠 샌드박스(Cuckoo Sandbox)는 가상머신을 이용해 악성코드를 효율적으로 분석할 수 있는 도구이다. 가상머신에서 동작하기 때문에 악성코드에 거상머신 탐지기법(VM Detect)이 있다면, 분석을 하는데 어려움이 있다. 이러한 경우 악성코드를 분석하기 위해 실머신 기반에서 분석이 가능하도록 구현하고, 구현 과정에서 메모리 덤프(Memory Dump)문제가 존재한다. 이전 방식은 가상머신 소프트웨어들이 메모리 덤프 파일을 따로 만들고 해당 파일을 분석하였지만, 실머신에서는 메모리파일을 따로 가지지 않는다. 이러한 문제를 해결하기 위해 실머신에서는 어떻게 메모리덤프 문제를 해결할 수 있는지를 알아보고 덤프를 하였을 때, 가상머신과 실머신에서 어떤 차이점이 나타나는지 알아보고자 한다.

1. 서론

쿠쿠 샌드박스(Cuckoo Sandbox)의 시스템에서는 Vm ware, VirtualBox, KVM 등과 같은 가상머신 소프트웨어를 이용하여 가상머신위에서 악성코드를 동적분석을 한다. 가상머신안에서 악성코드가 동작하였던 내용들을 쿠쿠 샌드박스로 정보를 전송하고, 그 정보로 악성코드의 행위에 대해서 분석할 수 있었다. 하지만 이러한 방법은 분석하고자 하는 악성코드 안에 VM Detect 기술이 적용되어 있다면, 악성코드는 처음 실행하고자 하는 기능이 아닌 다른 정상적인 기능을 수행하게 된다. 이러한 문제를 해결하기 위하여, Cuckoo Sandbox에서 Real 머신기반 분석환경을 추가 하고자 하였다. 이 과정에서 여러 가지 문제가 발생하게 되는데, 그 중 하나가 메모리 덤프와 관련된 문제 이다. 이전 가상머신에서는 suspend기능을 사용하여 메모리 정보를 파일로 저장하고, 해당 파일을 여러 가지 메모리 분석도구를 활용하여 분석을 할 수 있다. 하지만 실머신 기반에서는 이전 가상머신과 같은 기능이 제공되지 않는다. 즉 직접 메모리 덤프를 수행해야 하는데, 이 문제를 해결하기 위한 방법과, 이전 메모리 덤프 방법에 대해서 비교하고, 악성코드를 두고 가상머신에서의 메모리 덤프 결과와 실머신에서의 메모리 덤프 결과의 차이점에 대해서 이야기 하고자 한다.

본 논문은 먼저 현재 문제점에 대해서 소개하고, 가상머신의 메모리 덤프와 실머신 메모리 덤프의 결과의 차이를 분석하여, 최종적으로 이 문제를 해결하기 위한 해결책을 제안하고, 결론으로 끝을 맺는다.

2. 메모리 덤프 문제

현재 실제 Cuckoo Sandbox에서는 메모리 덤프 파일을 [cuckoo root]/storage/[Task ID]/memory.dmp에 저장을 한다. virtualbox는 debugvm 도구의 dumpguestcor 옵션을 이용하여 메모리 덤프를 수행하게 된다. 하지만 실머신에서는 이와 같은 방법으로 불가능 하다. 즉 직접 메모리덤프를 얻어야 한다. 일반적으로 실머신에서 메모리덤프를 얻기 위한 방법으로 크게 3가지 존재한다. 첫 번째 방법으로는 하드웨어를 이용한 메모리 덤프 방법이다. 하드웨어 방식에는 PCI를 이용한 방법과, FireWire(IEEE 1394)를 이용한 방법 2가지가 있다. 하지만 두가지 방법 전부 장비를 구입해야 한다는 단점이 존재하고, 특히 FireWire(IEEE 1394) 같은 경우에는 아직 안정성이 확보되지 않아, Crash가 종종 발생한다고 한다. 다음으로 소프트웨어를 이용한 방법이 있다. 일반적으로 많이 알려진 dd, win(32/64)dd, FastDump Pro, DumpIt 등 다양한 소프트웨어가 존재한다. 하지만 소프트웨어를 이용한 방식에서는 덤프 툴 자체도 메모리에 올라가고 실행되므로 툴에 대한 정보들도 메모리덤프결과에서 나오게 된다. 다른 방법으로는 크래시덤프를 이용한 방법이 존재한다. 이 경우에는 BSD가 발생하였을 때만 덤프가 가능하고, 또 일부

키설정을 통해 강제로 덤프를 만들 수 있지만, 이후 재부팅된 다음 다시 Cuckoo Sandbox로 dump을 전송할지 안할지 확실히 선택할 수 없다. 따라서 이 논문에서는 소프트웨어 방식을 이용한 방법을 채택하여 가상머신과 실머신에서의 덤프 결과의 차이에 대해 비교하고자 한다.

3. 실머신에서의 메모리 덤프 방법

소프트웨어 방식의 메모리 덤프를 위해 이 논문에서는 MoonSols사의 DumpIt (http://moonsols.com/resources/) 도구를 활용하고자 한다. 일반적인 DumpIt도구의 메모리 덤프 획득 방법은 해당 도구를 다운받은 후, 실행하여 y를 입력하고 Enter를 입력하게 되면 다음과 같이 메모리 덤프가 수행된다.

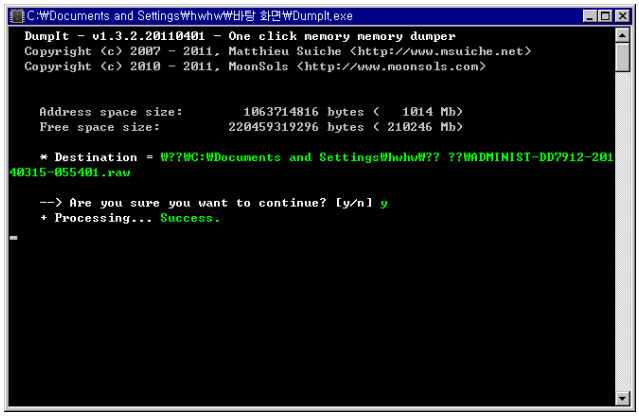

그림 1 MoonSols사의 DumpIt

이 도구를 Ollydbg 도구를 이용하여 Code Patch를 하였다. y를 입력하지 않아도, 실행만 함으로써 메모리가 덤프되도록 수정하였고, Success 메시지가 출력 후에는 키(Enter)를 입력해 주어야 종료되었는데, 이 부분도 패치하여 종료되도록 코드를 수정하였다. 이렇게 메모리 덤프가 수행되면 시스템을 종료하기 전에 메모리 덤프를 수행하고 메모리 덤프 파일을 Cuckoo Sandbox서버로 전송을 해주게 된다.

4. 가상머신의 메모리 덤프와 실머신의 메모리 덤프의 차이

위의 도구를 이용하여 실제머신과 가상머신에서 메모리 덤프를 비교해 보면, 크게 2가지가 차이가 나게 되는데, 첫 번째는 덤프파일 용량과 두 번째는 덤프 프로그램(DumpIt)에 대한 정보가 실머신에서 얻은 메모리 덤프에는 존재한다는 것이다. 용량적인 문제에서는 테스트하기 위해 사용한 가상머신은 192mb를 주었고, 실머신에서는 물리메모리가 1gb가 장착되어있다. 즉 가상머신에서는 192mb가 덤프가 되고, 실 머신에서는 1gb의 크기의 덤프파일이 나오게 된다. 두 번째로는 메모리 분석하는데 있어서 volatility도구를 사용하였는데, 분석 결과 실 머신 기반의 덤프파일에서만 DumpIt 프로세스 정보가 남아 있게 된다. 즉 소프트웨어방식으로 메모리 덤프를 수행하다보니, 메모리 덤프 프로그램에 대한 정보도 덤프 하는 동안 메모리에 올라가게 되고, 해당 정보도 같이 덤프가 되었다. 하지만 실제 악성코드 분석에 있어서는 큰 영향을 주지 않으므로, 메모리 덤프를 이용한 악성코드 분석에는 영향을 주지 않는다.

5. 기대 효과 및 결론

본 논문에서 제안한 실머신 기반의 악성코드 자동 분석 시스템의 직접 메모리덤프를 함으로써, 가상머신 기반의 쿠쿠 샌드박스의 메모리덤프와 동일한 기능을 수행할 수 있게 한다. 더불어 실머신에서의 다른 문제가 되는 기능이 모두 해결된다면, 오늘날 VM Detect기능을 가진 악성코드가 출연하는 것에 민첩하게 대응 할 수 있을 것으로 기대된다.

참고문헌

[1] Vmware, http://www.vmware.com
[2] Virtual Box, http://www.virtualbox.org
[3] KVM, http://www.linux-kvm.org
[4] Cuckoo, http://www.cuckoosandbox.org
[5] Detecting VMWare, http://brundlelab.wordpress.com/2012/10/21/detecting-vmware/
[6] Recovery Magic, http://www.softgate.co.kr/
[7] WOL, http://en.wikipedia.org/wiki/Wake-on-LAN