Anti-VM기법을 활용한 프로그램동작환경 탐지연구

김경민*, 박용수**
*한양대학교 컴퓨터공학과
**한양대학교 컴퓨터공학과
e-mail: casualab@hanyang.ac.kr

A detection of program operating environment using Anti-VM method

Gyeong-Min Kim*, Yong-su Park**
*Division of Computer Science & Engineering, Hanyang University
**Division of Computer Science & Engineering, Hanyang University

요 약

가상머신을 탐지하는 기법을 이용하여 현재 프로그램이 동작하는 플랫폼과 운영환경을 알아내는 방법을 제안한다. 가상머신 탐지기법으로 데렉 소더(Derek Soeder, eEye Digital Security)를 사용한다. 이 기법은 예외 명령어를 이용해 프로그램이 가상머신과 네이티브머신 환경에서 작동중인지 알아낸다. 가상머신의 코드 세그먼트 레지스트 영역은 네이티브머신보다 불명확 하다. 이런 메모리 주소체계의 차이를 이용한 가상머신탐지 기법이다. 더 응용하여 프로그램의 작동환경이 디버거(인텔 Pin, Olly dbg)인지 가상머신(VMware, QEMU)인지 혹은 네이티브머신(os: windows7 sp3)인지 구분하는 방안을 제안한다.

1. 서론

1.1 연구 개발의 목적 및 중요성

최근 악성코드에 개인정보 유출 및 침해, DDoS 공격, 북한의 해킹 등 경제/사회적인 문제는 급격하게 증가하고 있다. 악성코드에 의한 피해를 최소화 및 예방을 위해서는 신속한 분석과 대응이 필요하다. 하지만 대다수의 악성코드들은 자신을 보호하기 위해 실행 압축, 난독화, 안티 가상머신(이하, Anti-VM), 안티디버깅(Anti-Debugging), 코드 가상화 등의 기술을 사용하며, 아직 그 수는 적지만 악성코드가 구동되는 플랫폼에 따라서 다르게 작동하는 기법도 존재한다.[1] Anti-VM 기법의 연구는 점점 중요해지고 있다.

1.2 연구 개발의 내용 및 범위

본 논문에서는 가상머신인의 특징을 탐색하는 기법 중에 하나인 데렉 소더(Derek Soeder, eEye Digital Security)를 활용해서 프로그램이 작동되는 플랫폼이 디버거(Pin, Olly dbg)인지 가상머신(VMware, QEMU)인지 혹은 네이티브머신의 운영체제(Windows7 sp3)인지 구분하는 방안을 제안하고 있다.

1.3 기대효과

악성코드가 행하는 행동을 빠르고 정확하게 동적 분석하여 대응할 수 있으며, 악성코드의 분석시간을 단축시킬 수 있다.

2. 본론

2.1 연구 개발의 목적 및 중요성

Anti-VM은 악성코드 분석 기법으로 가상환경에서의 악성코드 분석 기술을 우회하는 방법이다. 이를 위해선 가상머신의 종류와 특징 그리고 환경설정을 알아내야 한다. 이를 효과적으로 찾는 방안이 될 것이다.

2.2 연구내용 및 방법

가상머신 VMware를 감지하는 '데렉 소더(Derek Soeder, eEye Digital Security)'라는 기법은 2005년에 소개되었다.[2] 가상머신 VMware 에뮬레이션 모드를 탐지하는 이 기법은 코드영역(Code Segments) 밖으로 형태의 RETF 인스트럭션 명령어를 실행해서 예외를 발생시킨다. 그리고 가상머신과 네이티브머신의 레지스트리 EIP주소 차이를 확인한다.[3]

RET명령어와 RETF의 차이로, RET명령어는 레

지스트리의 EIP에 새로운 주소를 꺼내고 스택 포인터를 증가시킨다. 그리고 주소로 이동한다. RETF는 레지스트리의 EIP에 새로운 주소를 꺼내고 레지스트리의 CS값을 같이 꺼낸다. 그리고 스택 포인터를 증가시킨다.

<표 1> 데렉 소더 기법 주요부분 소스코드

```
...
void __declspec(naked) switchcs()
{
        __asm
        {
            pop eax
            push 0x000F
            push eax
            retf
        } //cs역역을 바꾸기 위해
            0x000F 주소를 할당하며
            RETF명령어를 실행
}

int main()
{
...
/*임의의 DLL 파일을 불러와서 CS 레지스트리 영역을 변경한다.*/
...
    __try
    {
        switchcs();
        __asm
        {
            or eax, -1
            jmp eax
        }
    }

__except(detect(GetExceptionInformation()))
{ }
...
/*__except으로 실행시킨 함수를 통해 RIP주소를 확인한다.*/
...
    return 0;
}
```

가상머신에서는 레지스트리 CS영역이 네이티브머신 보다 불명확한 점을 이용한다. 네이티브머신은 레지스트리 CS영역 밖에 있는 RETF(return far)명령어에 있어서 코드를 실행하지 않고 바로 예외(exception)를 발생한다. 하지만 가상머신에서는 일단 코드에 진입하고 EIP 주소를 불러오고 그 다음 CS영역을 불러온다.

2.2 연구결과

여러 플랫폼(네이티브머신 Windos7 sp3, 가상머신 Vmware/QEMU, 디버깅툴 OllyDbg/인텔Pin)에서 각기 다른 결과를 확인할 수 있었다. 코드가 동작하는 환경이 가상머신인지 특징을 탐색하는 기법을 활용해서 악성코드가 동작하는 플랫폼이 어떤 가상머신 제품 환경인지 판단 할 수 있다고 예상한다.

<표 2> 데릭 소더 기법을 여러 가상환경에서 실행한 결과

구분	설명	결과
네이티브머신 (Win7, SP3)	기법의 의도에 부합하게 예외 발생	○
VMware 에뮬레이션 모드	기법의 의도에 부합하게 예외를 발생시키지 않고 함정경로로 진행 [4][5]	×
QEMU		
Olly 디버거	영역 접근 오류 후 예외 경로로 정상실행	△
인텔 Pin	기능 미지원 실행불가	-

○= 정상실행, ×=함정경로, △=부분오류, - = 기타

EIP와 CS영역의 차이가 나와 정상실행이 되지 않으면 가상머신이나 디버거다. 그 중에서 코드실행이 불가한 경우에는 인텔의 Pin디버거다.

3. 결론

본 논문에서는 가상머신을 탐지하는 기법을 적용시켜 플랫폼별 실행결과차이를 알아보았다. 본 논문에서 소개하는 기법이 실제 분석에 적용되기 위해서는 더 많은 테스트가 필요하고 지속 발전시켜야 한다.

참고문헌

[1] Chen, Xu, et al. "Towards an understanding of anti-virtualization and anti-debugging behavior in modern malware." 2008 IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN). IEEE, 2008.

[2] Soeder, Derek, and Ryan Permeh. "eEye BootRoot." BlackHat USA (2005).

[3] Soeder, Derek, and Ryan Permeh. "eEye BootRoot: A Basis for Bootstrap-Based windows kernel code." BlackHat USA (2005): 11.

[4] Ugurlu, Omer Sezgin. Stealth sandbox analysis of malware. Diss. bilkent university, 2009.

[5] Klein, Tobias. "ScoopyNG - The VMware detection tool." (2008).