2세대 PT(Processor Trace)를 이용한 동적 코드분석 방법 연구★

김 현 철*

요 약

운영 체제의 코어에 Intel PT가 포함된 경우, 크래시 발생 시 디버거는 프로그램 상태를 검사할 수 있을 뿐만 아니라 크래시를 발생시킨 제어 플로우를 재구성할 수 있다. 또한, 커널 패닉 및 기타 시스템 정지와 같은 상황을 디버그하기 위해 실행 트레이스 범위를 전체 시스템으로 확장할 수도 있다. 2세대 PT인 WinIPT 라이브러리는 Windows 10 (버전 1809/Redstone 5)에서 제공하는 IOCTL 및 레지스트리 메커니즘을 통해 프로세스 별 및 코어 별 트레이스를 실행할 수 있는 추가 코드가 포함된 Intel PT 드라이버를 포함하고 있다. 즉 기존 1세대 PT에서 비정규화된 방식으로만 제한적인 접근이 가능했던 PT 트레이스 정보를 2세대 PT에서는 운영 체제에서 제공하는 IOCTL 및 레지스트리 메커니즘을 통해 프로세스 별 및 코어 별 트레이스를 실행할 수 있게 되었다. 본 논문에서는 1/2세대 PT를 이용하여 윈도우 환경에서 PT 데이터 패킷의 수집·저장·디코딩 및 악성코드 검출을 위한 방법을 비교·설명하였다.

A Study on Dynamic Code Analysis Method using 2nd Generation PT(Processor Trace)

Hyuncheol Kim*

ABSTRACT

If the operating system's core file contains an Intel PT, the debugger can not only check the program state at the time of the crash, but can also reconfigure the control flow that caused the crash. We can also extend the execution trace scope to the entire system to debug kernel panics and other system hangs. The second-generation PT, the WinIPT library, includes an Intel PT driver with additional code to run process and core-specific traces through the IOCTL and registry mechanisms provided by Windows 10 (RS5). In other words, the PT trace information, which was limited access only by the first generation PT, can be executed by process and core by the IOCTL and registry mechanism provided by the operating system in the second generation PT. In this paper, we compare and describe methods for collecting, storing, decoding and detecting malicious codes of data packets in a window environment using 1/2 generation PT.

Key words : Tracing, Processor Trace, Flow Reconstruction, Malicious Code Detection, Flow Detection.

접수일(2019년 3월 3일), 수정일(1차: 2019년 3월 21일), 게재확정일(2019년 3월 29일)

★ 이 논문은 2018년도 남서울대학교 학술연구비 지원에 의해 연구되었음.

* 남서울대학교 컴퓨터소프트웨어학과 교수

1. 서 론

인텔 프로세서 트레이스(Intel Processor Trace (PT))는 시스템 오버헤드를 최소화한 디버깅과 실행 트레이스(execution trace)를 수행할 수 있는 기능이다. PT에서는 특정 프로그램의 실행 후 캡처된 트레이스 데이터를 소프트웨어로 처리하여 정확한 프로그램 흐름을 재구성할 수 있도록 전용 하드웨어 기능을 사용한다. 즉, 각 하드웨어 스레드에서 소프트웨어 실행에 대한 정보를 캡처하여 필요한 용도에 맞게 재가공할 수 있다.

Intel PT 하드웨어에 의해 캡처된 정보는 간결성과 신속성 보장을 위해 압축된 데이터 패킷 형태로 수집·저장된다. 기타 압축 코드에서처럼 코드 또는 이전 트레이스를 통하여 직접 추론할 수 있는 데이터는 저장되지 않는다. 데이터 패킷들은 패킷 타이밍, 프로그램 흐름 정보 및 프로그램 유도 모드 관련 정보 등을 담고 있다. 이러한 패킷은 시스템의 하위 메모리로 전송되기 전에 내부적으로 버퍼링 될 수 있다 [1][2][3].

이처럼 Intel PT는 모든 종류의 시스템 이벤트에 대한 컨텍스트를 제공한다. 예를 들어 성능 프로파일러는 PT를 사용하여 실행 품질에 영향을 미치는 '응답 시간'과 같은 근본 원인을 파악할 수 있다. 또한, 영상처리 프로그램 개발자는 문제가 있는 개별 프레임의 실행을 매우 세부적으로 검증할 수 있다. 따라서 Intel PT가 제공하는 완벽한 트레이싱 기능을 통해 이전까지 일반적인 트레이서가 제공하였던 수준보다 훨씬 더 자세한 실행 흐름을 검사할 수 있다.

한편 디버거에서는 Intel PT를 사용하여 현재의 프로그램 실행 위치까지 이르게 만든 코드 흐름을 재구성할 수 있다. 즉 시스템 크래쉬 지점, 브레이크 포인트, 와치 포인트 또는 단순히 함수 호출 다음의 명령인지 여부와 관계없이 디버깅을 지속할 수 있다. 또한, 역방향 스테핑 명령을 통해 저장된 실행 히스토리를 탐색할 수도 있다 [4].

1세대 PT는 윈도우 운영 체제에서 직접적으로 해당 기능을 지원하지 않았기 때문에 비정규화된 ETW(Event Tracing for Windows) 인터널에 의존할 수밖에 없었다. 그러나 2세대 PT에서는 운영체제에서 제공하는 IOCTL 및 레지스트리 메커니즘을 통해 프로세스 별 및 코어 별 트레이스를 실행할 수 있게 되었다 [5][6][7][8]. 본 논문에서는 1/2세대 PT를 이용하여 윈도우 환경에서 데이터 패킷 형태의 수집·저장·디코딩 및 악성코드 검출을 위한 방법을 비교·설명하였다.

본 논문의 구성은 다음과 같다. 2장에서는 PT를 이용하여 윈도우 환경에서 데이터 패킷 형태의 수집·저장·디코딩 및 악성코드 검출을 위한 방법을 설명하였다. 3장에서는 2세대 Intel PT를 이용하여 윈도우 환경에서 데이터 패킷 형태의 수집·저장·디코딩 및 악성코드 검출을 위한 방법을 설명하였다. 마지막으로 4장에서는 2세대 PT를 이용하여 악성코드 트레이스를 수행하는 방법과 관련된 결론과 향후 추가적인 연구 내용을 기술하였다.

2. 1세대 PT와 활용

2.1 1세대 PT 개요

PT는 사용자 및 커널 레벨 코드의 명령어들을 볼 수 있으며, 명령 분기의 세부 단계까지 사이클 정보를 수집할 수 있다. 기존의 트레이스 정보와 비교하여 PT는 기록할 수 있는 트레이스 정보의 유형과 양에 대해 훨씬 빠르고 유연성이 있다. 이를 통해 매우 상세한 실행 흐름을 구성할 수 있으며 최대 정밀도 수준에서 성능 및 정확성 디버깅을 쉽게 할 수 있다 [5][6][7]. PT가 제공하는 장점 중의 하나로는 소스 코드의 변경이 필요하지 않다는 점이며 컨텍스트 스위칭 (Context switches), 주소 스페이스 변경 (address space modification) 등과 같은 운영체제의 사이드밴드 (sideband) 정보가 필요하지 않고 단지 디코드 트레이스를 위해 오브젝트 코드만을 필요로 한다 [4].

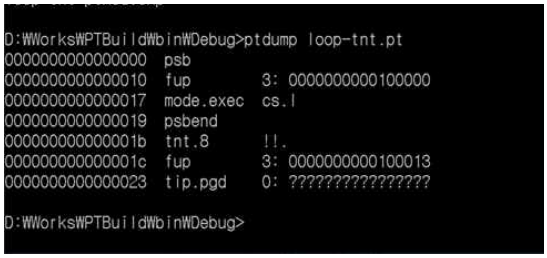
그러나 1세대 PT는 윈도우 운영 체제에서 해당 기능을 공식적으로 지원하지 않았기 때문에 비정규화된 ETW 인터널에 의존할 수밖에 없었다. 즉 윈도우 시스템에 직접적인 접근이 어려웠고, 실행

중인 프로세스 또한 직접 접근이 어려워 pt 명령어를 통하여 트레이스 하고자 하는 프로세스를 직접 실행시키고 그 이후에 트레이스를 수행할 수 있었다. 즉 완전한 동적 분석을 실행할 수 없다는 단점이 있다.

2.2 1세대 PT 활용

(그림 1)은 pttc 명령을 이용하여 임의의 Yasm 어셈블러로 작성한 코드에 대한 pt 파일을 생성하는 과정을 나타내고 있다.

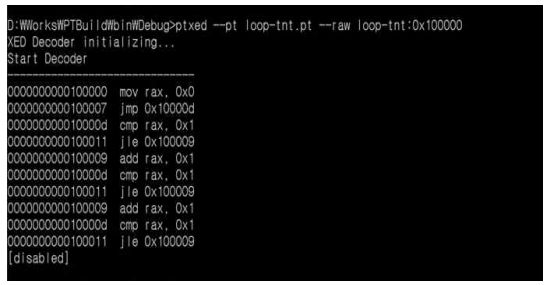
(그림 1) pttc 명령을 이용하여 pt 파일 생성



(그림 2) ptdump 명령을 이용하여 생성된 pt 파일 검증

(그림 2)는 생성된 pt 파일의 검증을 위해 ptdump 명령을 이용하여 pt 파일을 pt 데이터 포맷으로 출력하는 과정을 나타내고 있다.

(그림 3)은 ptxed 명령을 이용하여 생성된 pt 파일을 Yasm 디어셈블러를 이용하여 디어셈블 하는 과정을 나타내고 있다. 이를 통하여 디어셈블된 내용이 원래의 어셈블러와 같음을 확인할 수 있다.



(그림 3) ptxed 명령을 이용하여 생성된 pt 파일의 디어셈블

3. 2세대 PT와 활용

3.1 2세대 PT 개요

리눅스에서와같이 운영 체제의 코어 파일에 Intel PT가 포함될 수 있다. 이 경우에는 크래시 발생 시 디버거가 프로그램 상태를 검사할 수 있을 뿐만 아니라 크래시를 초래한 제어 플로우를 재구성할 수 있다. 또한, 커널 패닉 및 기타 시스템 정지를 디버그하기 위해 실행 트레이스 범위를 전체 시스템으로 확장할 수도 있다. 이때 Intel PT는 운영 체제 크래시가 발생했을 때 운영 체제 크래시 덤프 메커니즘의 일부로 트레이스 정보를 저장한 다음 이후에 오류를 재구성할 수 있다.

2세대 PT로 간주하는 Windows 용 Intel Process Trace (WinIPT) 라이브러리[8]는 Windows 10 (버전 1809 / Redstone 5)에서 제공하는 새로운 Intel PT 기능을 활용하는 프로젝트이다. 2세대 PT에서는 운영 체제에서 제공하는 IOCTL 및 레지스트리 메커니즘을 통해 프로세스 별 및 코어 별 트레이스를 실행할 수 있는 추가 코드가 포함된 Intel PT 드라이버(ipt.sys)를 포함하고 있다. WinIPT 저장소에는 다음과 같은 3가지 프로젝트가 진행되고 있다.

libipt: 2세대 PT를 위해 Windows에서 공개한 새로운 기능으로 코어 단위 및 프로세스 단위 트레이스를 가능하게 하는 IPT 드라이버와 서비스 IOCTL에 대한 액세스 권한을 부여하는 라이브러리의 Win32 API 버전이다. 이 라이브러리는 Dbg

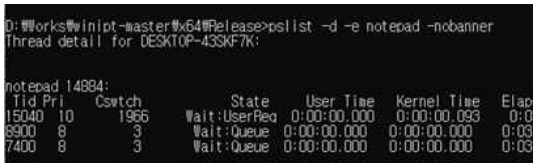
help.dll, Dbgcore.dll, TTDRecordCPU.dll, Faultrep.dll 및 Ntdll.dll의 일부 기능을 다시 구현한 것이다.

libiptnt: libipt와 동일한 라이브러리로 Ntdll.dll 함수만 사용하는 네이티브/비-Win32 응용 프로그램용 네이티브 NT API 버전이다.

ipttool: libipt 정적 라이브러리를 이용한 샘플 프로그램이며, 주어진 프로세스에 대한 추적을 시작, 중지 및 조회하는 간단한 CLI 유틸리티를 제공한다. 코어 트레이스와 디코딩 기능을 지원하지는 않는다.

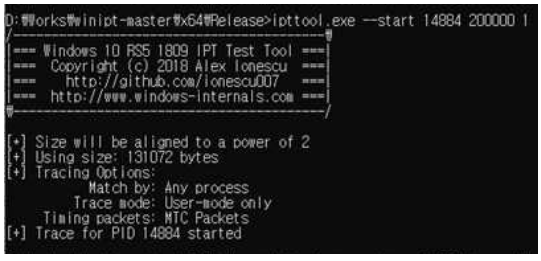
3.2 2세대 PT 활용

(그림 4)에서와 같이 윈도우에서 제공하는 프로세스 관련 툴인 pslist 명령을 사용하여 트레이스를 수행하고자 하는 “notepad” 프로세스의 PID(Process ID) 값을 확인한다.



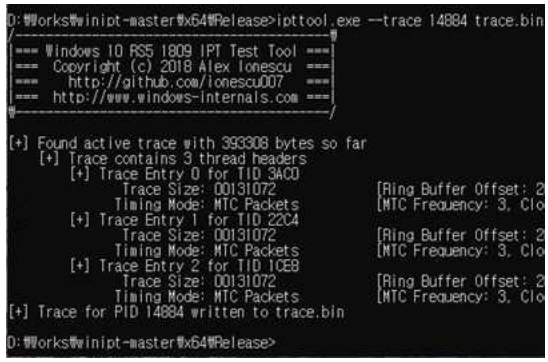
(그림 4) 트레이스 프로세스 선택

(그림 5)에서는 현재 실행 중으로 확인된 “notepad” 프로세스의 PID 값을 기반으로 ipttool 명령을 이용하여 트레이스를 시작하는 과정을 나타내고 있다. (그림 5)에서 20000 값을 트레이스 버퍼 크기를 나타낸다.



(그림 5) 선택 프로세스 트레이스 시작

(그림 6)은 트레이스 과정을 종료하고, 모든 트레이스 정보를 trace.bin 파일로 저장하는 과정을 나타내고 있다. ipttool은 디코딩 기능을 제공하지 않기 때문에 trace.bin 파일은 ptxed나 상용 디어셈블러 툴을 이용하여 디코딩할 수 있다.



(그림 6) 트레이스 정보 저장

4. 결 론

운영 체제의 코어 파일에 Intel PT가 포함될 경우, 크래시 발생 시 디버거가 프로그램 상태를 검사할 수 있을 뿐만 아니라 크래시를 초래한 제어 플로우를 재구성할 수 있다. 또한, 커널 패닉 및 기타 시스템 정지를 디버그하기 위해 실행 트레이스 범위를 전체 시스템으로 확장할 수도 있다.

2세대 PT인 WinIPT 라이브러리[8]는 Windows 10 (RS 5)에서 제공하는 IOCTL 및 레지스트리 메커니즘을 통해 프로세스 별 및 코어 별 트레이스를 실행할 수 있는 추가 코드가 포함된 Intel PT 드라이버(ipt.sys)를 포함하고 있다. 즉 기존 1세대 PT에서 비정규화된 방식으로만 제한적인 접근이 가능했던 PT 트레이스 정보를 2세대 PT에서는 운영 체제에서 제공하는 IOCTL 및 레지스트리 메커니즘을 통해 프로세스 별 및 코어 별 트레이스를 실행할 수 있게 되었다. 본 논문에서는 1/2세대 PT를 이용하여 윈도우 환경에서 데이터 패킷 형태의 수집·저장·디코딩 및 악성코드 검출을 위한 방법을 비교·설명하였다.

참고문헌

[1] Napoleon C. Paxton, “Cloud Security: A Review of Current Issues and Proposed Solutions,” International Conference on Collaboration and Internet Computing (CIC), pp. 452-455, 2016

[2] Tahira Mahboob; Maryam Zahid; Gulnoor Ahmad, “Adopting information security techniques for cloud computing－A survey,” International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE), pp. 7-11, 2016

[3] Jörg Thalheim; Pramod Bhatotia; Christof Fetzer, “INSPECTOR: Data Provenance Using Intel Processor Trace (PT),” International Conference on Distributed Computing Systems (ICDCS), pp. 25-34, 2016

[4] Khalid El Makkaoui; Abdellah Ezzati; Abderrahim Beni-Hssane; Cina Motamed, “Cloud security and privacy model for providing secure cloud services,” 2016 2nd International Conference on Cloud Computing Technologies and Applications (CloudTech), pp. 81-86, 2016

[5] Bob Duncan; Alfred Bratterud; Andreas Happe, “Enhancing cloud security and privacy: Time for a new approach?,” International Conference on Innovative Computing Technology (INTECH), pp. 110-115, 2016

[6] Sin-Fu Lai; Hui-Kai Su; Wen-Hsu Hsiao; Kim-Joan Chen, “Design and implementation of cloud security defense system with software defined networking technologies,” International Conference on Information and Communication Technology Convergence (ICTC), pp. 292-207, 2016

[7] Andi Kleen, “Simple Intel CPU processor tracing on Linux,” https://github.com/andikleen/simple-pt

[8] Alex Ionescu, “The Windows Library for Intel Process Trace (WinIPT)”, https://github.com/ionescu007/winipt

〔저자소개〕

김 현 철 (Hyuncheol Kim)
1990년 2월 성균관대학교 학사
1992년 2월 성균관대학교 석사
2005년 8월 성균관대학교 박사
2006년 9월 ~ 현재 남서울대학교 컴퓨터소프트웨어학과 교수
email : hckim@nsu.ac.kr