https://doi.org/10.33778/kcsa.2019.19.3.037

동적 코드 분석을 위한 전처리부 설계 및 구현★

김 현 철*

요 약

최근 다양한 형태의 악성코드 등장으로 인해 기존의 정적 분석은 많은 한계를 노출하고 있다. 정적분석은 (악성)코드를 실제로 실행하지 않고 원시 코드나 목적 코드를 가지고 코드나 프로그램의 구조를 분석하는 것을 의미한다. 한편 정보보안 분야에서의 동적 분석이란 일반적으로 (악성)코드를 직접 실행하여 분석하는 형태로 프로그램의 실행 플로우를 파악하기 위해 (악성)코드의 실행 전후 상태를 비교·조사하여 분석하는 형태를 의미한다. 그러나 동적 분석을 위해서는 막대한 양의 데이터와 로그를 분석해야 하며 모든 실행 플로우를 실제로 저장하기도 어려웠다. 본 논문에서는 윈도우 환경(윈도우 10 R5 이상)에서 2세대 PT를 기반으로 악성코드 탐지 및 실시간 다중 동적 분석을 수행하는 시스템의 전처리기 구조를 제안하였고 이를 구현하였다.

Design and Implementation of Preprocessing Part for Dynamic Code Analysis

Hyuncheol Kim*

ABSTRACT

Recently, due to the appearance of various types of malware, the existing static analysis expose s many limitations. Static analysis means analyzing the structure of a code or program with sourc e code or object code without actually executing the (malicious) code. On the other hand, dynamic analysis in the field of information security generally refers to a form that directly executes and a nalyzes (malware) code, and compares and examines and analyzes the state before and after execu tion of (malware) code to grasp the execution flow of the program. However, dynamic analysis req uired analyzing huge amounts of data and logs, and it was difficult to actually store all execution flows. In this paper, we propose and implement a preprocessor architecture of a system that performs malware detection and real-time multi-dynamic analysis based on 2nd generation PT in Windows environment (Windows 10 R5 and above).

Key words : Tracing, Processor Trace, Flow Reconstruction, Dynamic Code Analysis.

접수일(2019년 9월 6일), 게재확정일(2019년 9월 21일)

* 남서울대학교 컴퓨터소프트웨어학과 교수

★ 이 논문은 2019년도 남서울대학교 학술연구비 지원에 의해 연구되었음.

1. 서 론

최근 다양한 형태의 악성코드 등장으로 인해 기존의 정적 분석은 많은 한계를 노출하고 있다. 정적 분석은 (악성)코드를 실제로 실행하지 않고 원시 코드나 목적 코드를 가지고 코드나 프로그램의 구조를 분석하는 것을 의미한다. 일반적으로 정적 분석에서는 파일의 문자열, 함수, 헤더에서 제한적인 분석을 진행한다.

한편 정보보안 분야에서의 동적 분석이란 일반적으로 (악성)코드를 직접 실행하여 분석하는 형태로 프로그램의 실행 플로우를 파악하기 위해 (악성)코드의 실행 전후 상태를 비교·조사하여 분석하는 형태를 의미한다. 이를 통하여 파일, 프로그램 실행, 레지스트리, 서비스, 네트워크 및 자원 접근 등 시스템 전반적인 내용에 대한 변경사항들을 분석한다. 그러나 동적 분석을 위해서는 막대한 양의 데이터와 로그를 분석해야 하며 모든 실행 플로우를 실제로 저장하기도 어려웠다. 인텔 프로세서 트레이스(PT)는 CPU에서 실행되는 분기를 추적하는 새로운 하드웨어 기반 추적 기능으로 최소한의 노력으로 모든 실행 코드의 제어 흐름을 재구성할 수 있다. 이러한 하드웨어 트레이스 기능들은 운영 체제에 통합되어 프로파일 링 및 디버깅 메커니즘과의 긴밀한 통합이 가능하게 되었다.

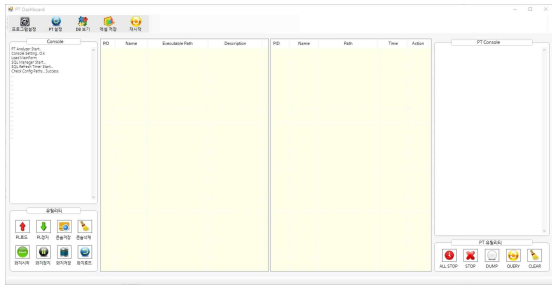
인텔 PT 하드웨어에 의해 수집된 정보는 간결성과 신속성 보장을 위해 압축된 데이터 패킷 형태로 수집·저장된다. 이를 PT 패킷이라고 한다. 기타 압축 코드에서처럼 코드 또는 이전 트레이스를 통하여 직접 추론할 수 있는 데이터는 저장되지 않는다. 데이터 패킷들은 패킷 타이밍, 프로그램 흐름 정보 및 프로그램 유도 모드 관련 정보 등을 담고 있다. 이러한 패킷은 시스템의 하위 메모리로 전송되기 전에 내부적으로 버퍼링 될 수 있다 [1][2][3]. 따라서 인텔 PT가 제공하는 완벽한 트레이싱 기능을 통해 이전까지 일반적인 트레이서가 제공하였던 수준보다 훨씬 더 자세한 실행 흐름을 검사할 수 있다. 또한, 2세대 PT에서는 운영 체제에서 제공하는 IOCTL 및 레지스트리 메커니즘을 통해 프로세스 별 및 코어 별 트레이스를 실행할 수 있게 되었다 [5][6][7][8].

본 논문에서는 윈도우 환경(윈도우 10 R5 이상)에서 2세대 PT를 기반으로 악성코드 탐지 및 실시간 다중 동적 분석을 수행하는 시스템의 전처리기 구조를 제안하였고 이를 구현하였다. 본 논문의 구성은 다음과 같다. 2장에서는 2세대 PT를 기반으로 악성코드 탐지 및 실시간 다중 동적 분석을 수행하는 시스템의 전처리기 구조와 구현 내용을 상세히 기술하였다. 3장에서는 악성코드 탐지를 위한 실시간 동적 분석 시스템과 관련된 결론과 향후 추가적인 연구 내용을 기술하였다.

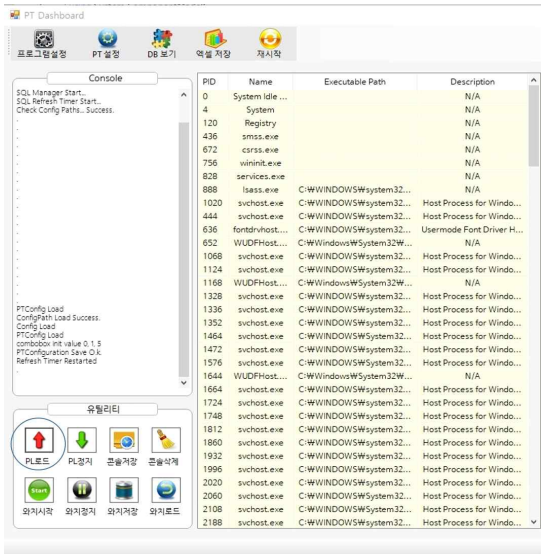
2. 실시간 다중 동적 분석 시스템 전처리기 구조 및 구현

2.1 2세대 PT

2세대 PT로 간주되는 윈도우용 인텔 Process Trace (WinIPT) 라이브러리[8]는 Windows 10 (버전 1809 / Redstone 5)에서 제공하는 새로운 인텔 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. libipt는 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. 또한 ipttool은 libipt 정적 라이브러리를 이용한 샘플 프로그램이며, 주어진 프로세스에 대한 추적을 시작, 중지 및 조회하는 간단한 CLI 유틸리티를 제공한다.

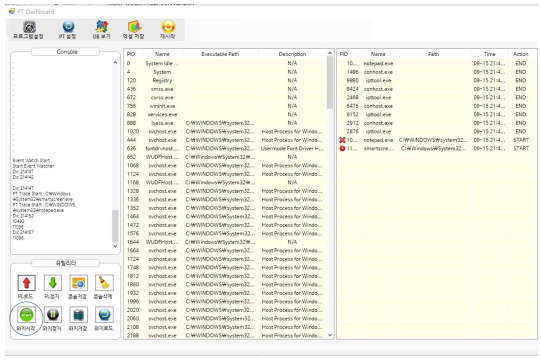


(그림 1) 동적 분석 전처리기 메인화면



(그림 2) 프로세스 리스트(PL) 관리부

(그림 3)은 신규 프로세스 캡처 및 트레이스 제어부를 나타내고 있다. (그림 3)에서와 같이 와치 시작 버튼을 클릭하면, 그 시점부터 새롭게 생성되거나 종료되는 모든 프로세스 리스트를 (그림 3)의 프로세스 트레이스 화면에 표시한다.



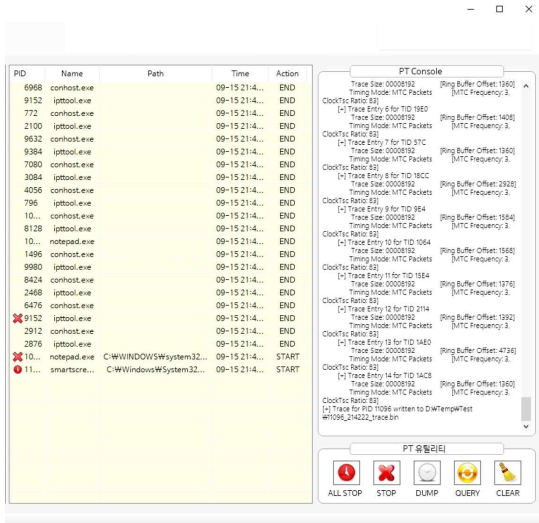
(그림 3) 신규 프로세스 캡처 및 트레이스 제어부

2.2 동적 분석 전처리 구조

그림 1)은 본 논문에서 제안하고 있는 동적 분석 전처리기 메인화면을 나타내고 있다. 동적 분석 전처리기는 크게 (그림 2), (그림 3), (그림 4)에서와 같이 프로세스 리스트(PL) 관리부, 신규 프로세스 캡처 및 트레이스 제어부, PT 덤프 제어부로 구성되어 있다. 또한 (그림 5), (그림 6)에서 나타내고 있는 바와 같이 동적 분석 전처리기 구성을 제어하는 동적 분석 전처리기 경로 설정창과 동적 분석 전처리기 PT 설정창으로 구성된다.

(그림 2)는 프로세스 리스트(PL) 관리부와 콘솔창을 나타내고 있다. 콘솔창에는 시스템 동작과 관련된 모든 내용이 실시간으로 출력되면 콘솔저장 버튼과 콘솔 삭제 버튼을 이용하여 콘솔창 내용을 저장하거나 삭제할 수 있다.

(그림 2) 프로세스 리스트(PL) 관리부에서 PL로드 버튼을 클릭하면 동적 분석 전처리기에서는 주기적으로 현재 실행 중인 프로세스 리스트를 화면에 표시한다. 프로세스 리스트 갱신주기는 (그림 6) 동적 분석 전처리기 PT 설정에서 선택할 수 있다. (그림 2)에서 PL정지 버튼을 클릭하면 프로세스 리스트 자동갱신을 수행하지 않는다.

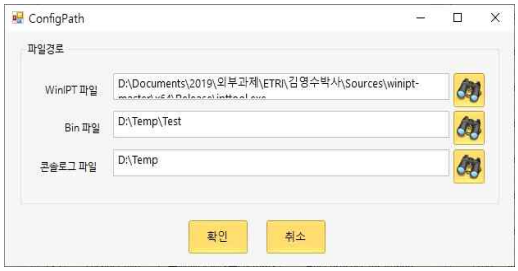


(그림 4) PT 덤프 제어부

프로세스 트레이스 화면에는 프로세스 ID, 프로세스의 이름과 경로, 생성/종료 시각, 그리고 생성인지 종료인지를 나타내는 식별자가 표시된다.

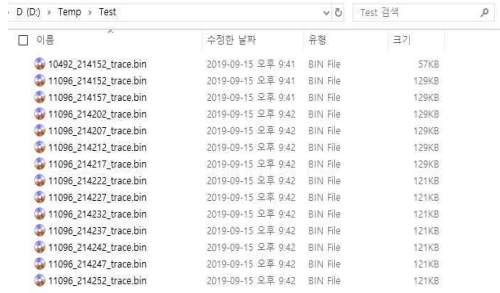
동적 분석 전처리기에서는 (그림 6) 동적 분석 전처리기 PT 설정을 통하여 신규 프로세스가 생성

될 때마다 자동 또는 수동으로 PT 트레이스를 실행할수 있다. 또한, 동적 분석 전처리기 PT 설정을 통하여 자동 또는 수동으로 PT 덤프 파일을 생성할 수 있다. (그림 6)의 DUMP버튼을 클릭하면 현재 트레이스 중인 프로세스들에 대해서 개별적으로 또는 전체적으로 Dump 파일을 생성할 수 있다. 또한 (그림 6)의 ALL STOP 버튼을 클릭하면 모든 PT 트레이스를 중단하며, STOP 버튼을 클릭하면 개별 프로세스에 대해서 트레이스 중단을 실행한다. (그림 7)은 PT 덤프 제어부에서 자동으로 생성한 PT 덤프 파일들의 내용을 나타내고 있다.



(그림 5) 동적 분석 전처리기 경로 설정

(그림 6) 동적 분석 전처리기 PT 설정



(그림 7) PT 덤프 생성 파일

3. 결 론

일반적으로 동적 분석이란 (악성)코드를 직접 실행하여 분석하는 형태로 프로그램의 실행 플로우를 파악하기 위해 (악성)코드의 실행 전후 상태를 비교·조사하여 분석하는 형태를 의미한다. 이를 통하여 파일, 프로그램 실행, 레지스트리, 서비스, 네트워크 및 자원 접근 등 시스템 전반적인 내용에 대한 변경사항들을 분석한다. 그러나 동적 분석을 위해서는 막대한 양의 데이터와 로그를 분석해야 하며 모든 실행 플로우를 실제로 저장하기도 어려웠다. 인텔 프로세서 트레이스(PT)는 CPU에서 실행되는 분기를 추적하는 새로운 하드웨어 기반 추적 기능으로 최소한의 노력으로 모든 실행 코드의 제어 흐름을 재구성할 수 있다.

인텔 PT가 제공하는 완벽한 트레이싱 기능을 통해 이전까지 일반적인 트레이서가 제공하였던 수준보다 훨씬 더 자세한 실행 흐름을 검사할 수 있다. 또한, 2세대 PT에서는 운영 체제에서 제공하는 IOCTL 및 레지스트리 메커니즘을 통해 프로세스 별 및 코어 별 트레이스를 실행할 수 있게 되었다. 본 논문에서는 윈도우 환경(윈도우 10 R5 이상)에서 2세대 PT를 기반으로 악성코드 탐지 및 실시간 다중 동적 분석을 수행하는 시스템의 전처리기 구조를 제안하였고 이를 구현하였다.

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 service s,” 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 de-fined 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

〔저 자 소 개〕

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