한국정보보호학회
Korea Institute of Information Security & Cryptology

2019 한국정보보호학회 하계학술대회

CISC-S'19

Conference on Information Security and Cryptography-Summer 2019

2019. 6. 20(목)~22(토)

부산 동명대학교 (제1정보통신관, 중앙도서관)

Proceedings

주최 한국정보보호학회 Korea Institute of Information Security & Cryptology

주관 동명대학교 TONGMYONG UNIVERSITY

후원 국가정보원 National Intelligence Service Korea, 과학기술정보통신부, 행정안전부, KISA 한국인터넷진흥원, ETRI 한국전자통신연구원 Electronics and Telecommunications Research Institute, NSR 국가보안기술연구소 National Security Research Institute, IITP 정보통신기획평가원, AhnLab

윈도우 환경에서 보안 소프트웨어의 보안 커버리지 분석 한계점 및 대안 제시

이병용*, 박기웅†

*세종대학교 시스템보안연구실, † 세종대학교 정보보호학과

Security Software Coverage Analysis Limits in Windows Environments and Alternative Proposal

Byeong-Yong Yi*, Ki-Woong Park†

*SysCore Lab., Sejong University.

† Department of Computer and Information Security, Sejong University

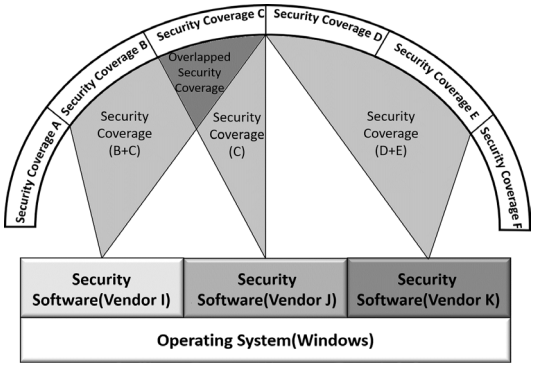
요 약

악성코드로 인한 보안 위협의 증가에 따라 이에 대응하기 위한 보안 소프트웨어 제조사의 숫자도 함께 증가하고 있다. 보안 소프트웨어 제조사 간에 사전 협의가 없다면, 서로 다른 보안 소프트웨어를 사용하는 경우에 보안 커버리지 중복이 발생할 수 있다. 이를 방지하기 위해선, 보안 소프트웨어 분석을 수행하여 보안 커버리지의 중복을 최소화할 필요가 있다. 하지만 윈도우 환경에서 실행되는 상용 보안 소프트웨어는 일반적으로 안티 디버깅 기술로 보호되고 있어 분석에 제한이 있다. 따라서 소프트웨어의 특정 시점 간 고수준 행위 비교 분석을 수행하는 보안 커버리지 분석 프레임워크 구조를 제시한다. 활용할 요소는 윈도우 환경에서 활용 가능한 윈도우 레지스트리와 이벤트 로그 및 패킷 스니핑 도구이며 이를 통해 기존 분석 방법의 한계점을 해결할 대안을 제시한다.

I. 서론

악성코드로 인한 보안 위협의 증가에 따라서, 이에 대응하기 위한 보안 소프트웨어 제조사의 숫자도 함께 증가하고 있다[1]. 포화상태에 이른 보안 소프트웨어 시장에서 제조사 간의 사전 협의가 없다면, 사용자 시스템에서 서로 다른 보안 소프트웨어를 사용하는 경우에 [그림 1]과 같이 보안 커버리지 중복으로 인한 불필요한 오버헤드가 발생할 수 있다. 사용자에게 현재 사용 중인 보안 소프트웨어의 보안 커버리지 정보를 제공할 수 있다면, 각 보안 소프트웨어 기능을 선택적으로 제어함으로써 불필요한 보안 커버리지 중복을 최소화하고 시스템 자원을 효율적으로 사용할 수 있을 것이다.

보안 커버리지 정보를 제공하기 위해선 보안 소프트웨어에 대한 심층분석이 필요하다. 분석을 위해 디버깅 기술을 활용하면 API 트레이스를 통해 보안 소프트웨어에 대한 상세한 분석이 가능하다. 하지만, 윈도우 환경에서 사용되는 대부분의 상용 보안 소프트웨어는



[그림 1] 보안 커버리지 중복 발생의 예

† 교신저자: 박기웅 (세종대학교 정보보호학과 교수)

본 연구는 2019년도 과학기술정보통신부의 재원으로 정보통신기획평가원의 지원(No.2018-0-00420, No.2019-0-00273) 및 한국연구재단 연구과제(NRF-2017R1C1B2003957)의 지원을 받아 수행된 연구임.

Themida[2], Enigma Protector[3], ASProtect[4], VMProtect[5] 등의 안티 디버깅 도구로 보호받고 있어 디버깅이 제한된다.

따라서 본 논문에서는 API 트레이스 같은 저수준 행위(Low-level behavior) 기반의 보안 소프트웨어 분석이 아닌, 윈도우 레지스트리, 이벤트 로그, 패킷 스니핑 도구를 통한 고수준(High-level behavior) 행위 기반의 보안 소프트웨어 분석 방안과 분석을 위한 프레임워크 구조를 제시한다. 본 논문의 구성은 다음과 같다. 2장에서는 안티 디버깅 우회기술의 한계점과 보안 커버리지 분석을 위한 분석 요소를 설명한다. 3장에서는 보안 소프트웨어의 보안 커버리지 분석을 위한 분석 프레임워크 구조를 제안한다. 4장에서 결론 및 향후 연구과제를 기술한다.

II. 배경 지식

본 장에서는 현재 사용되는 안티 디버깅 우회기술의 한계점과 보안 커버리지 분석을 위한 분석 요소를 설명한다.

2.1 안티 디버깅 우회기술의 한계점

안티 디버깅 도구는 프로세스 목록에서 디버깅 도구를 탐지하거나 실행시간 비교 등을 통해 현재 디버깅 시도 중으로 판단되면 이를 차단한다. 일반적으로 사용하는 안티 디버깅 기술은 [표 1]과 같이 API, Exception, Hardware and Register, Timing, Direct Process and Thread block detection, Modified code detection으로 분류할 수 있다[6]. 안티 디버깅 도구는 앞서 나열한 디버깅 탐지 기술을 조합하여 사용하고 있으며, 하나의 디버깅 탐지 조건이라도 위반할 경우 디버깅을 차단한다[7]. 따라서 이러한 안티 디버깅 기술을 모두 우회하는 기술 연구에는 기술적 한계가 있다.

2.2 보안 커버리지 파악을 위한 보안 소프트웨어 분석 요소

일반적으로 보안 소프트웨어는 파일에 대한 읽기, 쓰기, 변경, 삭제 과정에서 악성 행위를 탐지하거나, 정해진 규칙 외의 트래픽을 제한하는 방식으로 보안 기능을 수행한다. 따라서 보안 커버리지 분석을 위해선 보안 소프트웨어 프로세스의 특정 파일에 대한 접근 및 수정 여부 정보와 더불어 네트워크 트래픽에 관한 정보에 대한 수집이 필요하다. 이와 관련된 정보는 Process Monitor[8] 등의 프로세스 실시간 모니터링 도구를 통해서 수집할 수 있지만, 이 도구는 안티 디버깅 도구에 의해 실행이 제한되며 획득할 수 있는 트래픽 정보가 전문 네트워크 스니핑 도구에 비해서 비교적 자세하지 않다. 따라서 제안하는 커버리지 분석 프레임워크에서 분석할 요소는 윈도우 레지스트리, 윈도우 이벤트 로그 및 Network Monitor이다. 이 요소들은 [표 2]와 같이 소프트웨어의 실행 정

[표 1] 안티 디버깅 기술 종류 및 설명

종류	설명
API based	디버깅 행위를 탐지하기 위한 API이며 윈도우 운영체제에 내장되어 있다.
Exception based	디버깅 도구가 분석 대상 프로세스의 예외(Exception)를 적절히 처리하지 못할 경우, 이를 디버깅 행위로 판단한다.
Hardware and Register based	하드웨어 레지스터에서 디버깅에 관한 정보를 검사한다.
Timing based	디버깅 도구에 의해 프로그램의 실행 시간 지연이 발생하는지 검사한다.
Direct Process and Thread block detection	프로세스와 스레드 정보에서 디버깅 행위와 관련된 정보 존재 여부를 검사한다.
Modified code detection	프로그램 코드 블록의 CRC 혹은 해시 값 변화 여부를 검사하여 디버깅 여부를 판단한다.

[표 2] 분석 대상에 따른 분석 요소의 분류

	분석 대상	
분석 요소	파일 접근 로그	네트워크 트래픽 로그
레지스트리	○	△
이벤트 로그	○	△
Network Monitor	×	○

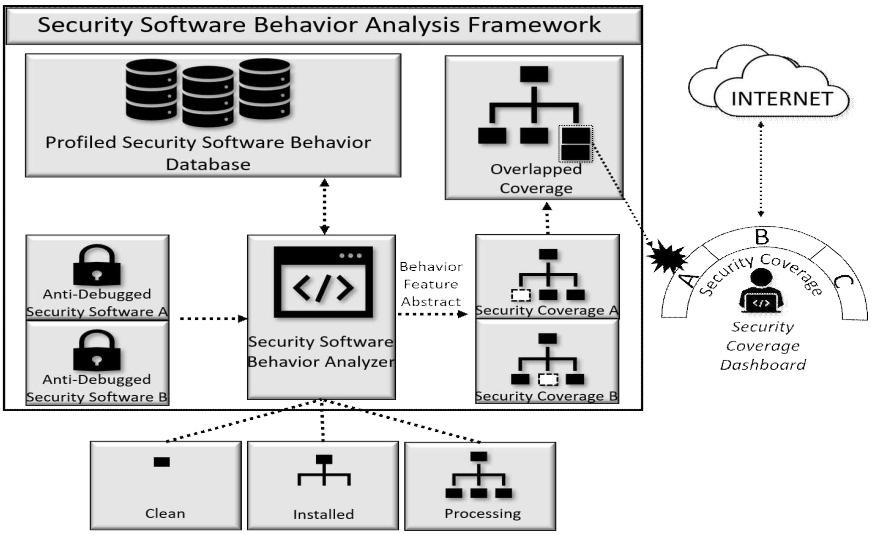
보 및 시스템 변화를 기록할 수 있으며, 획득할 수 있는 정보가 보안 소프트웨어 분석에 있어 상호 보완적이다. 이와 같은 요소들을 통해 보안 소프트웨어의 설치 전/후, 실행 중 시스템 변화를 통계적으로 분석하고, 이를 기반으로 보안 커버리지 분석에 활용할 것이다.

- 윈도우 레지스트리[9] - 윈도우 레지스트리는 운영체제 설정과 설치된 프로그램 및 작동하는 서비스에 대한 정보를 데이터베이스 형태로 저장한다. 윈도우 레지스트리는 윈도우 환경이라면 어디서나 사용할 수 있으며, RegShot[10], RegRipper[11] 등의 레지스트리 분석 도구를 통해 특정 레지스트리의 추가 및 삭제, 변경 등의 정보를 파악할 수 있다. 레지스트리 정보는 단순한 파일 삭제 전/후에서도 레지스트리의 변화 폭이 크며, 이는 다른 프로세스의 작동 데이터를 포함할 수 있다. 따라서 특정 프로세스의 레지스트리 변화 데이터만을 추출할 수 있는 기준이 필요하다. 레지스트리는 하이브(Hive)라는 논리적 그룹으로 분류할 수 있으며, 프로세스 실행 시 주로 변하는 레지스트리는 HKCU와 HKLM이다.
- 윈도우 이벤트 로그[12] - 윈도우 이벤트 로그는 윈도우 실행 중 발생하는 로그를 evtx 파일 형태로 저장하는 이벤트 로깅을 수행한다. 기본적으로 운영체제에 포함되어있는 이벤트 로그 외에도, FullEventLogView[13], EvLog[14] 등의 도구를 통해서도 이벤트 로그 분석이 가능하다. 이벤트 로그는 이벤트 ID마다 의미하는 행위가 다르기 때문에 이에 대해 분석할 필요가 있다. 또한, 방대한 로그가 저장되기 때문에 보안 소프트웨어 분석에 적절한 조건 설정을 통해 데이터를 선별하여 사용해야 한다. 특정 프로세스에 대한 행위 분석 시 대표적으로 사용되는 이벤트 ID에는 [표 3]과 같은 항목들이 있다.
- Network Monitor[15] - 네트워크 설정 정보 및 관한 특정 IP 및 Port로의 단순 연결/종료 정보는 레지스트리와 이벤트 로그에서도 얻을 수 있지만, 상세한 패킷 정보 획득에는 한계가 있다. 따라서 보안 소프트웨어 분석을 위해선 실시간 네트워크 스니핑 도구가 추가로 필요하다. 안티 디버깅 도구에 의해 특정 보안 소프트웨어 프로세스의 직접적인 네트워크 패킷 정보 획득은 제한될 가능성이 있다. 하지만, 시스템 전반의 트래픽 변화 정보를 획득하기 위해 네트워크 스니핑 도구를 사용한다. 여러 네트워크 스니핑 도구 중 마이크로소프트에서 직접 제공하며 사용이 비교적 쉬운 Network Monitor로 선정하였다.

앞서 나열한 요소들을 통해 보안 소프트웨어의 보안 행위로 발생한 고수준 행위 로그를 획득할 수 있다. 이 로그를 활용하면 디버깅이 제한된 환경에서도 특정 프로세스의 분류에 활용이 가능하다. Han[16] 등은 멀웨어 프로파일링 과정에서 기존의 코드 분석, API 사용 추이 분석과 함께 파일 및 레지스트리에 대한 접근 패턴과 빈도, 네트워크 패킷 분석 등의 과정을 추가하여 멀웨어 프로파일링의 정확도를 높였다. 또한, Deng[17] 등은 네트워크 트래픽 기록에서 특정 패턴을 추출하는 것을 통해, 안티 디버깅 도구로 분석이 제한된 멀웨어의 악성 행위 분

[표 3] 소프트웨어 고수준 행위 관련 이벤트 ID

이벤트 ID	설명
4656	변경 된 레지스트리 값
4657	4656 이벤트를 발생시킨 프로세스 이름과 변경된 레지스트리의 변경 전/후 값
4660	특정 오브젝트가 삭제됨
4663	특정 오브젝트에 접근 시도
4688	새로운 프로세스가 생성됨
4689	프로세스가 종료됨



[그림 2] 보안 커버리지 분석 프레임워크 구조

석에 관한 연구를 수행하였다. 이와 같은 연구들에 의하면, 안티 디버깅 도구로 보호되고 있는 프로그램도 특정 기능을 수행하기 위해서는 파일시스템 및 네트워크 트래픽에서 특정 로그 패턴이 생긴다는 것을 도출할 수 있다. 따라서 앞서 설명한 요소들을 보안 커버리지 분석 프레임워크에서 활용하는 방법은 다음과 같다. 먼저 잘 알려진 보안 소프트웨어의 설치 전/후, 실행 중 로그 변화의 특징을 통계적으로 분석하여 이를 데이터베이스화 한다. 통계적으로 분석할 대상은 보안 소프트웨어 프로세스가 자주 접근하거나, 특정 형태로 값을 변경하거나 특정 데이터를 추가 및 삭제하는 패턴 양상 보이는 데이터이다. 이후, 보안 커버리지 분석이 필요한 보안 소프트웨어는 설치 전/후, 실행 중 레지스트리 및 이벤트 로그와 네트워크 트래픽의 변화에서 특징을 추출하여 보안 커버리지 데이터베이스와 유사도를 비교 후, 보안 커버리지를 도출할 것이다.

III. 보안 커버리지 분석 프레임워크

본 장에서는 앞에서 분석한 내용을 기반으로 윈도우 환경에서 보안 커버리지 분석을 위한 프레임워크 구조를 제안한다. 제안하는 보안 커버리지 분석 프레임워크 구조는 [그림 2]와 같으며, 각 모듈이 수행하는 역할은 다음과 같다.

- Anti-Debugged Security Software - 분석 대상 보안 소프트웨어이다. 안티 디버깅 기술로 보호받고 있어 디버깅을 통한 소프트웨어 분석이 제한된다.
- Security Software Behavior Analyzer - 윈도우 레지스트리와 이벤트 로그 및 네트워크 트래픽을 기반으로 하는 보안 커버리지 분석 도구이다. 보안 소프트웨어의 설치 전/후 및 작동 중으로 나누어 앞서 나열한 분석 요소들을 통해 특징을 추출한다. 특정 레지스트리 및 이벤트 로그에 대한 접근, 기록 빈도와 네트워크 트래픽 변화 패턴을 통계적으로 분석한다.
- Profiled Security Software Behavior Database - 분석한 보안 소프트웨어의 프로파일링 데이터를 저장하는 데이터베이스이다. Security Software Behavior Analyzer를 통해 도출된 분석결과와 유사도를 비교하여 보안 커버리지를 도출한다.
- Security Coverage - Security Software Behavior Analyzer와 Profiled Security Software Behavior Database와의 유사도 비교를 통해 도출된 보안 커버리지이다.
- Overlapped Coverage - 보안 커버리지 중복 발생 지점이며, 불필요한 오버헤드 제거를 위해 보안 소프트웨어 기능의 선택적 제어가 필요한 부분이다.
- Security Coverage Dashboard - 사용자가 현재 보안 커버리지의 중복된 부분을 파악할 수 있도록, 보안 커버리지를 시각화한다.

이를 통해 보안 기능을 선택적으로 제어하거나, 추가 및 제거가 필요한 보안 소프트웨어에 대한 정보를 획득할 수 있다.

IV. 결론 및 향후 연구과제

본 논문에서는 디버깅이 제한된 보안 소프트웨어 분석 방안을 연구하고 보안 커버리지 도출을 위한 프레임워크 구조를 제안하였다. 본 연구를 통해, 보안 소프트웨어를 사용하는 시스템의 효율적인 컴퓨팅 자원 사용에 관한 연구에 활용될 수 있을 것이다. 향후 연구에서는 분석된 보안 커버리지 간의 유사도를 측정하는 방식을 정의하여 사용자에게 보안 커버리지 중복에 대한 정보를 정량적으로 제공할 수 있도록 연구할 예정이다. 또한, 보안 커버리지 분석을 위해 활용할 수 있는 추가 요소를 확보하여 도출된 보안 커버리지의 정확도를 높이기 위한 연구를 진행할 예정이다.

[참고문헌]

[1] Security landscape plagued by too many vendors: Cisco. https://www.zdnet.com/article/security-landscape-plagued-by-too-many-vendors-cisco/

[2] Themida. https://www.oreans.com/themida.php

[3] The Enigma Protector. https://enigmaprotector.com/en/about.html

[4] ASProtect 64. http://www.aspack.com/asprotect64.html

[5] VMProtect. https://vmpsoft.com/

[6] Shields, Tyler. "Anti-debugging - a developers view." Veracode Inc., USA (2010).

[7] 홍수화, 박용수. "Pin을 이용한 안티디버깅 우회 설계 및 구현." 17.5 (2016): 33-42.

[8] Process Monitor. https://docs.microsoft.com/en-us/sysinternals/downloads/procmon

[9] Registry. https://docs.microsoft.com/ko-kr/windows/desktop/SysInfo/registry

[10] RegShot. https://sourceforge.net/projects/regshot/

[11] RegRipper. https://github.com/keydet89/RegRipper2.8

[12] Windows Event Log. https://docs.microsoft.com/en-us/windows/desktop/wes/windows-event-log

[13] FullEventLogView. https://www.nirsoft.net/utils/full_event_log_view.html

[14] EvLog. http://eventid.net/evlog/

[15] Microsoft Network Monitor 3.4. https://www.microsoft.com/en-us/download/details.aspx?id=4865

[16] Han, Weijie, et al. "MalInsight: A systematic profiling based malware detection framework." Journal of Network and Computer Applications 125 (2019): 236-250.

[17] Deng, Xiyue, and Jelena Mirkovic. "Malware analysis through High-level behavior." 11th {USENIX} Workshop on Cyber Security Experimentation and Test ({CSET} 18). 2018.