한국정보보호학회
Korea Institute of Information Security & Cryptology

2023년 한국정보보호학회 하계학술대회

CISC-S'23

Conference on Information Security and Cryptography Summer 2023

2023년 6월 22일(목)~23일(금)

강원대학교 춘천캠퍼스 60주년 기념관

주최 한국정보보호학회 Korea Institute of Information Security & Cryptology
주관 KNU 강원대학교 KANGWON NATIONAL UNIVERSITY
후원 국가정보원 NATIONAL INTELLIGENCE SERVICE, 과학기술정보통신부, 행정안전부, KISA 한국인터넷진흥원, ETRI 한국전자통신연구원 Electronics and Telecommunications Research Institute, NSR 국가보안기술연구소 National Security Research Institute, Google

2023년 한국정보보호학회 하계학술대회 CISC-S'23

Conference on Information Security and Cryptography Summer 2023

안티 디버깅 기술/무력화 연계분석을 통한 우회 프레임워크 설계

최기상*, 최상훈[1], 박기웅†

세종대학교 정보보호학과 (학부생,연구원[1], 교수†)

Design of bypass framework with Anti-Debugging technology/countermeasures linkage analysis

Ki-Sang Choi*, Sang-Hoon Choi[1], Ki-Woong Park†

*Dept. of Computer and Information Security, Sejong University

(Undergraduate Student*, Postdoctoral Researcher[1], Professor†)

요 약

악성코드는 금전적인 목적에 의하여 서비스의 한 형태로 블랙마켓에 판매되고 있다. 판매에 따른 수요가 증가함에 따라 악성코드를 통한 공격이 확장되었다. 이를 대응하기 위해 인공지능을 활용한 탐지 및 분류 연구들이 등장하였지만, 공격자들은 분석을 방지하고자 안티 디버깅 기술을 활용하고 있다. 본 논문에서는 고도화된 악성코드에 대응하고자 안티 디버깅 요소들을 기술별 목적을 통해 분류하고 패치를 통한 우회 기법을 설명한다. 더 나아가, 가상화 기술과 베어 메탈 환경을 기반으로 하는 안티 디버깅 우회 프레임워크를 제안한다. 우리가 제안하는 프레임워크는 안티 디버깅이 적용된 다량의 악성코드에 대하여 자동적으로 처리할 수 있는 방안을 제시한다.

1. 서론

PC기반의 악성코드는 금전적인 목적으로 제작되었고 서비스의 한 형태로 블랙마켓에 판매되고 있다 [1]. 타겟에 따라 사용자의 범위가 전문가부터 일반인들까지 사용할 수 있는 형태로 퍼져나가고 있으며 피해 사례 또한 급증하고 있다. 이처럼, 고도화되고 다양화되고 있는 악성코드들에 대응하기 위해 악성코드 분석 연구자들은 인공지능 도구를 활용한 다양한 연구들이 제안됐다 [2, 3]. 특히, 악성코드 바이너리 분석을 통해 데이터 셋을 생성하고, 머신러닝을 활용하여 탐지하고 분류하기 위한 연구들이 제안되었다. 일반적으로 머신러닝 기반의 악성코드 탐지/분류 연구에서 사용되는 데이터셋은 코드 단위에서 추출할 수 있는 특성정보와 런타임에 추출할 수 있는 특성정보로 분류된다 [4].

하지만 최근 악성코드 개발자는 이를 인지하고 정적 및 동적 분석에서의 특성 추출을 방해하고자 악성코드에 안티 디버깅 기술을 다수 적용하였다 [5]. 따라서, 잘못된 데이터셋 구성에 따른 부정확한 지표를 예방하기 위해서는 안티 디버깅 기술을 무력화할 방안이 필요하다. 본 논문은 해당 방안을 찾기 위한 목적으로 안티 디버깅 관련 연구에 관해 설명하고, 안티 디버깅을 회피할 수 있는 하이브리드 환경기반의 안티디버깅 우회 프레임워크를 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 안티 디버깅 기술 항목에 대하여 정적 분석 관련한 기술과 동적 분석 관련한 기술에 대하여 나열한다. 3장에서는 코드 패치를 기반으로 수행된 안티 디버깅 우회 관련 연구에 관하여 기술한다. 4장에서는 안티 디버깅을 우회하기 위한 샌드박스 환경, 베어 메탈 환경 그리고 패치 기반의 프레임워크를 제안한다. 마지막, 5장에

† 교신저자

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 정보통신방송기술국제공동연구사업(Project No. RS-2022-00165794, 40%), 국방ICT융합사업(Project No. Project No.2022-0-00701, 10%), 실감콘텐츠핵심기술개발사업(Project No. RS-2023-00228996, 10%), 대학ICT연구센터 육성지원사업(Project No. 2023-2021-0-01816, 10%) 및 한국연구재단 개인기초연구과제(Project No. RS-2023-00208460, 30%)의 지원을 받아 수행된 연구임.

서는 제안한 프레임워크 구성에 따른 의의와 한계점에 대해 설명한다.

2. 안티 디버깅 기술

본 논문은 바이너리를 분석하는 과정 중에 분석가들을 방해하는 모든 요소에 대하여 안티 디버깅이라고 정의한다. 이에 안티 디버깅의 적용 범위를 정적 분석과 동적 분석으로 분류한다.

2.1 안티 디버깅 관련 정적 분석연구

정적 분석에서의 안티 디버깅은 패킹 혹은 코드 난독화로 불리며 IDA, Ghidra와 같은 코드 디컴파일러 도구를 활용하여 분석할 때, 악성 행위 외의 코드 목록을 보여주거나 불분명한 코드에 대한 결과값을 반환하는 기법이다. 이를 수행해주는 대표적인 도구로는 Themida, Enigma 그리고 VMprotect 등이 있으며 관련 기법은 <표 1>과 같다 [5].

<표 1> 코드 난독화 기법

코드 난독화	설명
Compressors	암호화 없이 파일을 크기를 줄이는 목적으로 사용
Cryptors	코드 분석에 있어 로직을 분석을 어렵게 하기 위해 코드에 대해서 난수화 및 암호화 적용
Protectors	압축과 암호 및 난수를 동시에 적용
Bundle	실행 파일과 필요한 파일을 패키지 형태로 첨부 및 실행

2.2 안티 디버깅 관련 동적 분석연구

동적 분석에서의 안티 디버깅은 크게 두 가지로 분류한다. 프로세스에 접근하여 실행 코드에 대한 디버깅을 방해하는 목적으로 디버거 대상의 안티 디버깅 기술과 독립적인 샌드박스 환경 내에서의 악성 행위분석을 방해하는 목적으로의 개발된 샌드박스 대상의 안티 디버깅 기술이 있다[6].

디버거 대상의 안티 디버깅 기술은 실행 중인 프로세스 혹은 프로세스 실행 과정에 있어서 Windbg나 Ollydbg와 같은 디버거 도구를 기반으로 실행 코드에 대한 목록 확인 및 수정하는 행위를 방해하는 기술이다. 디버거 대상의 안티 디버깅 기술을 적용함에 따라 특정 API 호출의 인자 상태나 내용을 쉽게 볼 수 없으며 전반적인 코드 흐름에 대한 인식을 기존보다 느리게 한다. 이와 관련된 기술로는 <표 2>와 같다.

<표 2> 디버거 대상의 안티 디버깅 기법

디버거 대상의 안티 디버깅	설명
Checking PEB Structure	PEB 구조에서 디버깅 요소 확인
Checking Break Point	코드 상의 Break Point 요소가 첨부 확인
Checking Window Artifact	파일 시스템, 레지스트리, 프로세스 명에서 디버깅 요소 확인
Checking Time-Interval	코드 간의 실행 시간을 측정하여 예외 사항 확인
Thread Handling	Thread 생성, 중지 및 숨김
Debugger Attached	스스로 디버거 접근 및 디버거 접근 불가

샌드박스 대상의 안티 디버깅 기술은 가상화 환경 내에서 악성 행위를 실행하고, 그에 따른 결과 보고서를 받는 행위를 방해하는 방법이다. 해당 기술을 적용하면 악성 행위와 다른 행위의 데이터가 결과 보고서에 쌓이거나, 어떠한 행위 기반의 정보도 포함되지 않은 결과 보고서를 얻을 수 있다. 이와 관련된 안티 디버깅 기술 및 설명은 <표 3>과 같다.

<표 3> 샌드박스 대상의 안티 디버깅 기법

샌드박스 대상의 안티 디버깅	설명
Detecting Environment	샌드박스 환경 탐지
Reverse Turing Test	사용자와 상호작용 파악 및 탐지
Stalling	Sleep 또는 random 요소를 첨부에 따른 파일 분석시간 초과
Trigger-Based	특정 조건(시스템 시간, 네트워크 정보 등등)에 따른 실행
Targeted	특정 환경기반 실행
Fileless	취약한 시스템(OS, Browser, etc)에 악성 스크립트를 첨부 및 실행

3. 안티 디버깅 우회 연구

악성코드 분석 연구자들은 정상 파일과 악성 파일을 분류하고 각각의 파일에 대한 차이점을 알아보고자 역공학적 기법을 활용한 분석을 진행한다. 안티 디버깅이 적용된 경우, 해당 행위를 회피하고자 다양한 연구를 진행했고 실제 상용 도구를 대상으로 패치하는 방안이 진행되어 왔다 [7]. 해당 연구에는 상용 도구인 Themida, Obsidium, Enigma, VMprotect를 목표로 한다. 각 상용 도구별 탐지하는 방안은 크게 인스트럭션, 하드웨어, 프로세스, 레지스트리 부분에서 가상환경임을 탐지 및 패치 방안에 대해 제시한다.

인스트럭션 부문에서 IN 인스트럭션과 CPUID 인스트럭션을 통해 가상환경임을 탐지한다. IN 인스트럭션의 경우, 가상환경임에 따른 통신 관련 추가 정보를 반환하고 CPUID 인스트럭션의 경우, 가상환경임에 따른 관련 정보를 반환한다. 이때 각각의 반환 값을 특정 레지스터에 저장하는데 해당 레지스터의 값 혹은 특정 비트 값을 0으로 세팅하여 해당 부분을 우회한다.

하드웨어 부문에서 디스크 혹은 펌웨어 내부에 가상환경이 사용하는 "VBox_" 혹은 "Virtual_"와 같은 문자열을 통해 가상화 환경임을 탐지할 수 있다. 위의 기법을 우회하기 위해 코드상에서의 비교 대상 문자열의 첫 글자를 다르게 수정하여 우회할 수 있다.

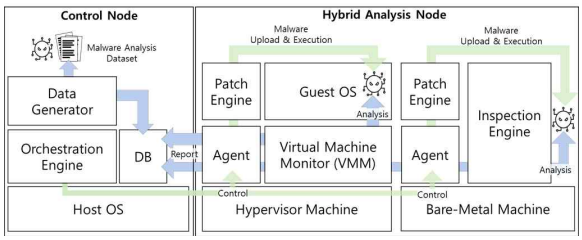
프로세스 부문에서 WideCharToMultiByte 관련한 API를 활용하여 가상머신이 지원하는 "VboxService.exe" 프로세스 실행 여부를 통해 가상화 환경임을 발견한다. 위의 기법을 우회하기 위해 하드웨어 부문과 마찬가지로 코드상에 존재하는 문자열을 수정하여 우회할 수 있다.

레지스트리 부문에서 HKLM안에 있는 HARDWARE를 통해 VBOX_폴더를 조사한다. 해당 폴더에 존재성에 따라 가상환경임을 탐지하는데 이 역시 하드웨어와 프로세스 부문과 동일하게 코드 내에서의 비교 대상 문자열을 수정하여 우회할 수 있다.

코드 패치를 통한 안티 디버깅 우회 방안은 메모리 영역을 사용자가 원하는 값으로 조작할 수 있다는 이점이 있지만, 상용 도구에 의한 안티 디버깅 기술이거나 분석한 악성코드 대상으로 제한된다. 이러한 한계점을 예방하기 위한 추가적인 연구 사례로는 샌드박스 기술을 활용하거나 베어 메탈 환경을 통해 해결할 수 있다 [8]. 하지만 <표 3>의 Reverser Turing Test나 Stalling와 같은 기법을 우회하기에는 많은 어려움이 있다.

4. 안티디버깅 우회 프레임워크

본 논문에서 각각의 기술이 가지는 단점에 대해 보완하기 위한 방안으로 가상화 기술, 베어 메탈 환경을 하이브리드 형태로 활용하며, 패치 기능을 지원하는 안티 디버깅 우회 프레임워크를 제안한다. 우리가 제안하는 프레임워크는 <그림 1>과 같다.



<그림 1> 안티디버깅 우회를 위한 자동화 프레임워크

Host OS에서 특정 악성코드에 대하여 동적 분석을 수행하고자 하였을 때, 오케스트레이션 엔진에 의하여 <표 3>의 Detection Environment가 포함되어 있으면 베어 메탈 환경을, <표 3>의 Trigger-Based나 Targeted가 포함되어 있으면 가상 머신 환경에서 수행하도록 선정하고 해당 환경에 전달한다. 가상화 환경 내에서 수행하게 된다면 가상 머신 내에 있는 Agent가 악성코드를 업로드한다. 업로드된 악성코드를 바탕으로 <표 3>의 Trigger-based 또는 Targeted를 우회하기 위한 가상화 환경 패치, 그 외의 요소를 우회하기 위한 코드 패치를 패치 엔진 하에 갱신하고 가상환 환경 내에서 수행한다. 가상화 환경 내에서 수행된 API, 네트워크와 같은 동적 분석 대상의 목록들은 가상화 환경 모니터링 도구들을 통해 기록한다.

악성코드 수행이 베어 메탈 환경 내에서 수행하게 된다면 가상 머신과 유사한 경로로 수행하고 기록된다. 하지만 베어 메탈 환경 내에서의 H/W가상화 어려움과 모니터링의 어려움으로 패치 엔진을 통해 코드 패치만을 지원하고 수행된 목록들에 있어 Dll Injection이나 Hooking과 같은 기법을 통해 기록한다[9]. 기록된 악성코드들은 프로파일링된 정보들을 바탕으로 HostOS의 데이터베이스에 저장한다. 만일 사용자가 분석된 정보들을 요청한다면 Data Generator를 통해 데이터베이스에 접근하여 기록된 정보를 토대로 사용자에게 전달한다. 이와 같은 프레임워크를 지원하게 된다면 각각이 가지는 한계점에 대해 보완하고 회피하면서 악성코드를 근본적으로 접근할 수 있다.

5. 결론 및 향후 연구

본 논문에서는 악성코드에 사용되고 있는 안티 디버깅 기술들을 목적에 맞게 분류하며 코드 패치와 샌드박스 기술을 통한 우회 방안에 대해 기술한다. 안티 디버깅 기술을 크게 코드 난독화, 디버거 대상의 안티 디버깅 그리고 샌드 박스 대상의 안티 디버깅으로 분류하고 안티 디버깅 우회 연구 사례를 분석하여 안티디버깅 우회 프레임 워크를 설계한다. 해당 프레임워크는 기존 분석 프레임워크와 다르게 안티 디버깅 우회 목적인 하이브리드 기반의 패치 프레임워크이며 자동적으로 분석 환경 선정과 동시에 패치한다. 이에 안티디버깅이 적용된 다량의 악성코드를 효율적으로 처리할 수 있을 것이라 사료된다. 하지만 제안된 프레임워크는 동적 분석을 기반으로 수행되기에 시간과 막대한 리소스가 소요되고 같은 항목의 안티디버깅일지라도 패밀리 별로 적용 방식이 상이하여 패치 엔진을 통한 수행 여부가 불분명하다. 위와 같은 사유로 안티디버깅이 첨부된 악성코드에 대하여 역공학적으로 접근하여 각 패밀리에 따른 적용 방식을 목록화하고 패치 방안을 나열할 필요가 있다. 향후 연구에서는 악성코드를 분석하여 코드 목록과 패치 방안을 맵핑할 것이며 코드 커버리지 기반으로 수행 여부를 판단하여 연구를 진행할 것이다.

[참고문헌]

[1] Per Hakon Meland, Yara Fareed Fahmy Bayoumy, Guttorm Sindre, The Ransomware-as-a-Service economy within the darknet, Computers & Security, Volume 92, 2020,

[2] Pfeffer, Avi, et al, Artificial intelligence based malware analysis., arXiv preprint arXiv:1704.08716,Apr,2017

[3] Boydell, B. Mac, M. Scanlon, Deep learning at the shallow end :Malware classi fi cation for non-domain experts, Digit. Investig. 26, 2018

[4] Singh, Jagsir, and JaswinderSingh,A survey on machine learning-based malware detection in executable files, Journal of Systems Architecture 112 2021

[5] Muralidharan, Trivikram, et al, File Packing from the Malware Perspective: Techniques, Analysis Approaches, and Directions for Enhancements, ACM Computing Surveys 55.5, 2022

[6]Afianian, Amir, et al, Malware dynamic analysis evasion techniques: A survey, ACM Computing Surveys (CSUR) 52.6, 2019

[7] Lee, Young Bi, Jae Hyuk Suk, and Dong Hoon Lee, Bypassing anti-analysis of commercial protector methods using DBI tools., IEEE Access 9, 2021

[8] Kirat, Dhilung, Giovanni Vigna, and Christopher Kruegel, Barecloud: Bare-metal analysis-based evasive malware detection, 23rd {USENIX} Security Symposium ({USENIX} Security 14), 2014.

[9] 최상훈, 박기웅, "Malware 분석을 위한 Bare-Metal 환경과 가상화 환경의 성능/기능 비교분석을 통한 개선사항 도출", 한국정보보호학회 추계학술대회, 2014