2024년 한국정보보호학회 하계학술대회

CISC-S'24

Conference on Information Security and Cryptography Summer 2024

일자 2024년 6월 20일(목) ~ 21일(금)

장소 세종시 정부세종컨벤션센터

주최 한국정보보호학회 Korea Institute of Information Security & Cryptology

주관 고려대학교 세종캠퍼스 KOREA UNIVERSITY SEJONG CAMPUS

후원 국가정보원 NATIONAL INTELLIGENCE SERVICE, 과학기술정보통신부, 행정안전부, 세종특별자치시, KISA 한국인터넷진흥원, ETRI 한국전자통신연구원 Electronics and Telecommunications Research Institute, NSR 국가보안기술연구소 National Security Research Institute, KISTI 한국과학기술정보연구원 Korea Institute of Science and Technology Information www.kisti.re.kr

한국정보보호학회
Korea Institute of Information Security & Cryptology

Illusion-Injection: 고도화 된 악성코드 저지를 위한 디버깅 기술 연구

이재욱*, 최상훈[1], 박기웅†

세종대학교 정보보호학과 (대학원생, 연구교수[1], 교수†)

Illusion-Injection: A Study on Debugging Techniques for Thwarting Advanced Malware

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

*Dept. of Computer and Information Security, Sejong University

(Graduate student*, Research Professor[1], Professor†)

요 약

악성코드는 점점 더 정교해지면서 분석을 회피하기 위해 안티 디버깅 기술을 사용한다. 해당 기술은 악성코드가 실행되는 동안 디버거의 존재를 감지하고, 디버거가 감지될 경우 악성코드의 동작을 변경하거나 종료함으로써 분석을 회피한다.본 연구에서는 이러한 안티 디버깅 기술을 역이용하여 악성코드가 디버깅 환경에 있는 것으로 속여 정상적인 동작을 방해하는 방법을 제안한다. 우리는 오픈소스 라이브러리를 활용해 NtQuerySystemInformation 함수를 후킹하여 강제로 디버그 모드로 인식하게 만든다. 실험 결과, 악성코드는 디버깅 환경으로 오인하여 실행을 중단한다. 본 연구는 기존의 가상환경 아티팩트 생성 방법보다 효과적으로 악성코드를 저지할 수 있음을 보여준다. 향후 연구에서는 상용 프로그램에 미치는 영향을 최소화하는 방향으로 진행할 예정이다.

I. 서론

악성코드가 점차 발전함에 따라 많은 분석가들이 악성코드를 분석하는데 시간을 많이 사용하고 있다. FireEye의 M-Trend 보고서에 따르면 2020년을 기준으로 매일 평균 110만개의 악성코드 샘플들을 분석하며 분석된 악성코드의 95%가 Windows환경에서 실행되고 있다고 한다[1]. 이러한 악성코드를 처리하기 위해 자동화된 샌드박스 기반 분석기술을 사용한다[2, 3].

최근 발전한 악성코드들은 Anti-Analysis를 통해 분석을 방해하거나 회피하고 있다. 분석환경이라고 판단하는 경우에는 프로그램을 종료하거나 전혀 다른 행위를 하며 분석을 회피하고 있다. 대표적인 Anti-Analysis 기술인 안티 디버깅은 악성코드 분석기술인 역공학으로부터 소프트웨어를 보호하는 기술이다[4]. 분석가들은 Ollydbg, IDA Pro와 같은 도구를 이용하여 악성코드를 분석을 한다. 기존에 존재하는 저지 방법은 안티 VM기술을 활용한 방법으로 가상머신환경에서만 사용하는 아티팩트를 생성해 악성코드를 저지하는 방법[12]이 있지만 실험한 샘플에 대해 모두 저지를 하지 못하였다.

본 연구에서는 악성코드가 분석을 회피하기

† 교신저자: 박기웅 (세종대학교 정보보호학과 교수)

본 논문은 과학기술정보통신부의 재원으로 정보통신기획평가원(IITP)의 정보통신방송기술 국제공동연구(Project No. RS-2022-00165794, 40%), 실감콘텐츠핵심기술개발(ProjectNo. RS-2023-00228996, 10%), 정보통신방송혁신인재양성사업(Project No. 2021-0-01816, 10%) 및 한국연구재단(NRF) 중견후속연구사업(Project No. RS-2023-00208460, 40%)의 지원을 받아 수행된 연구임.

위한 방법으로 기존 방법인 안티 VM이 아닌 안티 디버깅 기술을 역으로 이용하여 실제 환경에서도 악성코드에게 디버깅 환경인것처럼 속여 원래의 동작을 못하도록하고자 한다.

본 논문의 구성은 2장에서 안티 디버깅 기술에 대한 관련연구를 서술한다. 3장에서는 API 후킹을 통해 강제로 디버거모드를 활성화 시키는 방법에 대해 서술하고, 4장에서는 디버그 모드의 저지 성능을 평가한다. 마지막으로 5장에서는 결론과 향후 연구 방향을 제시한다.

II. 관련연구

2장에서는 실제 악성코드가 사용하는 안티 디버깅 기술과 다른 방식의 악성코드 저지에 관련된 연구에 대해 설명한다.

2.1 안티 디버깅 기술

2.1.1 Win32 API

프로세스 메모리에 상주하고 운영체제가 설정하는 시스템 테이블의 특수 플래그를 사용하여 프로세스가 디버깅 되고있음을 확인한다. 안티 디버깅에 사용하는 함수들로는 IsDebuggerPresnet(), NtQueryInformationProcess(), NtQuerySystemInformation()등등 여러함수가 있다. 본 논문에서 사용되는 함수는 NtQuerySystemInformation()이며 3장에서 더 자세하게 다룰 것이다.

2.1.2 IsDebuggerPresent함수

프로세스 환경 블록(PEB)에서 BegingDebugged 플래그를 확인하여 프로세스가 사용자 모드 기버거에 있는지 식별한다. 안티 디버깅 우회 기술로는 . 프로세스 환경 블록(PEB)의 BeingDebugged 플래그 값을 0x00으로 수정하여 쉽게 우회 할 수 있다[9].

2.1.3 CheckRemoteDebuggerPresent함수

특정 프로세스의 식별자(PID)를 받으면 이 함수는 해당 프로세스가 디버깅되고 있는지 여부를 확인한다. 안티 디버깅 우회 기술로는 내부적으로 호출되는 NtQueryInformationProcess 함수를 비활성화 하는 방법과 다른 함수를 호출하기 전에 EAX 레지스터 값에 0x00을 저장하고 함수를 종료한다. 이 두가지 방법으로 안티 디버깅을 우회 할 수 있다[10].

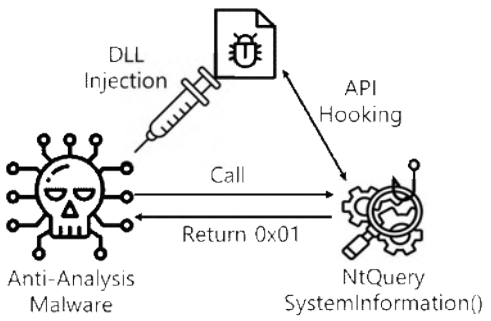
2.2 악성코드 저지 연구

기존 악석코드 저지 방법에는 서론에서 언급한 것 처럼 안티 VM을 저지하기 위한 Fake Sandbox Artifacts(FSA)가 있다. 안티 VM의 기술중에는 가상환경에서만 존재하는 레지스트리, 텍스트파일 및 폴더, 파이프 서버, 프로세스, 서비스와 같은 아티팩트를 검사하여 자기 자신이 가상환경인지 실제 환경인지 확인하여 동작을 회피한다[11]. Fake Sandbox Artifacts(FSA)에서는 앞서 설명한 안티 VM 기술을 가진 악성코드를 저지하기 위한 목적으로 가상환경에서 사용되는 아티팩트를 생성하여 악성코드를 저지한다[12]. 하지만 Fake Sandbox Artifacts(FSA)는 특정 악성코드 들에 대해 저지가 어렵다는 한계점이 존재한다.

III. Illusion-Injection 설계 및 구현

우리는 Detours를 활용하여 악성코드에서 동작하는 NtQuerySystemInformation함수를 후킹한다. 이후 악성코드가 현재 환경이 디버거 환경에 있는것처럼 속여 본래의 동작을 저지시키는 것을 목표로 한다[5].

3.1 동작 과정



(그림 1) Illusion-Injection 동작 과정

(그림 1)은 악성코드가 사용자 환경에 진입한 후 Detours를 사용해 NtQuerySystemInformation 함수를 후킹하여 악성코드의 동작을 저지시키는 과정이다. 악성

코드가 사용자 환경에 진입하면, Detours를 사용하여 NtQuerySystemInformation 함수를 후킹한다. 후킹된 NtQuerySystemInformation 함수는 악성코드가 디버깅 환경 여부를 확인할 때 디버깅 환경임을 나타내는 0x01 값을 반환한다. 이에 따라, 악성코드는 디버깅 환경으로 인식하여 악성 행위를 중단한다.

3.2 Detours

Detours는 Windows에서 API호출을 모니터링하고 측정하기위한 오픈소스 소프트웨어 패키지이며 DLL에 포함된 함수들을 가로채는 기능을 제공한다 [5]. 메모리 내 함수나 메소드의 진입점을 수정하고 이전 위치에 인터셉팅 루틴으로의 무조건적인 점프를 배치한다. 이전 기능은 메모리의 새로운 위치로 이동되고, 새로운 진입지점이 제공되어 인터셉팅 루틴이 가로챈 루틴을 하위 함수로 호출할 수 있다.

3.3 NtQuerySystemInformation 함수

NtQuerySystemInformation는 Win32API 함수중 하나로 Ntdll.dll라이브러리에 정의되어있고 현재 동작중인 OS 시스템에 대해 다양한 상태 정보를 반환해준다. 해당 함수는 운영체제가 디버그 모드로 부팅되어있는지 판단하여 안티 디버깅을 가능하게 한다. SYSTEM_INFORMATION_CLASS SystemInformationClass 파라미터에 SystemKernelDebuggerInformation(0x23)을 입력하면 현재 시스템이 디버그 모드로 부팅되었는지 알 수 있다. 디버그 모드 일경우 0x01을 반환한다.

IV.실험 및 결과

우리는 Illusion-Injection의 악성코드 저지 성능을 평가하기 위해 Malware bazaar, Malshare로부터 Anti-Analysis 기술이 적용된 악성코드를 수집하였다[7, 8]. 추가적으로, 저지 성능을 비교하기 위해 FSA 에서 사용한 악성코드를 평가에 활용하였다.

4.1 악성코드 저지 평가

해당 실험은 Windows10 환경에서 진행하였다. 우선적으로 해당 악성코드를 x32dbg를 이용해 NtQuerySystemInformation 함수가 존재하는지 확인하였다. 수집한 샘플에서 해당 함수가 존재하는것을 확인하였고 다음으로는 악성코드 샘플에 본 연구에서 제안하는 Illusion-Injection 기술을 적용시킨 후 악성코드를 실행시켜 보니 기존에 하던 동작과는 다른 실행된 직후 바로 종료가 되는 모습을 보였다. 실험에 사용한 악성코드가 Detours를 이용해 후킹 한 결과 원래의 동작을 저지시키는것에 성공하였다. FSA와 비교한 실험결과는 [표1]과 같다. FSA에서는 특정 악성코드들에 대해 저지가 불가능하였다. 반면에 우리가 제안하는 Illusion-Injection는 실험한 악성코드들이 모두 저지가 되는 것을 확인할 수 있다.

[표 1] Illusion-Injection 저지 성능평가

Category	Malware Signature	SHA256	FSA	Illusion-Injection
Backdoor	unknown	7954...422b	X	O
Backdoor	unknown	17b1...8e32	O	O
Trojan	Formbook	198a...ba9c	X	O
Trojan	PhoenixStealer	1c52...2dd5	O	O
Trojan	GuLoader	350b...bb84	O	O
Trojan	GuLoader	4df4...32ee	O	O
Trojan	MassLogger	7004...177f	X	O
Trojan	CobaltStrike	e9cb...6bac	O	O
Trojan	RedLine	075a...3cd2	O	O
Trojan	MassLogger	7a84...c3bd	O	O
Trojan	Xtrat	7bea...8020	X	O
Trojan	FFDroider	806a...7fb9	X	O
Trojan	unknown	c876...ba49	X	O
Trojan	AgentTesla	737e…22fa	O	O
TrickBot	TrickBot	d5ef…be83	X	O
VirTool	ImminentRAT	71b4...77b3	X	O

V.결론 및 향후연구

본 연구에서는 이러한 안티 디버깅 기술을 역이용하여 악성코드가 디버깅 환경에 있는 것으로 속여 정상적인 동작을 방해하는 Illusion-Injection 기술을 제안한다. 우리가 제안하는 Illusion-Injection 기술은 기존의 악성코드 저지 연구인 FSA와 비교하여 더 효과적임을 실험을 통해 입증했다. FSA는 특정 악성코드에 대한 저지가 실패했지만, Illusion-Injection의 경우 안티 디버깅이 적용된 모든 악성코드 샘플에 대한 저지를 성공하

였다. 하지만, 우리가 제안한 디버깅 기반의 저지 기법은 상용 소프트웨어 및 안티 디버깅이 적용된 일반적인 소프트웨어의 실행을 방해시킨다는 한계점이 있다. 향후 연구에서는 추가적인 검증과정을 통해 악성코드를 식별하여 Illusion-Injection을 적용시키는 방향을 연구할 것이다.

[참고문헌]

[1] Kim Minho, Cho Haehyun, and Yi Jeong Hyun. 2022. Large-scale analysis on anti-analysis techniques in real-world malware. IEEE Access 10 (2022), 75802-75815.

[2] Any.Run. (2021). Interactive Malware Hunting Service. [Online]. Available: https://any.run/

[3] H. Triage. Hatching Triage is a Fully Automated Solution for High-Volume Malware Analysis Using Advanced Sandboxing Technology, 2018–2022. Accessed: Feb. 15, 2021. [Online]. Available: https://tria.ge/

[4] M. N. Gagnon, S. Taylor and A. K. Ghosh, "Software Protection through Anti-Debugging," in IEEE Security & Privacy, vol. 5, no. 3, pp. 82-84, May-June 2007, doi: 10.1109/MSP.2007.71.

[5] Galen Hunt and Doug Brubacher, "Detours: Binary Interception of Win32 Functions", *Proceedings of the 3rd USENIX Windows NT Symposium*, 1999.

[6] Branco, R.R., Barbosa, G.N., & Drimel, P. (2012). Scientific but Not Academical Overview of Malware Anti-Debugging , Anti-Disassembly and Anti-VM Technologies.

[7] abuse.ch. [n. d.]. About MalwareBazaar. https://bazaar.abuse.ch/about/

[8] Cutler, S. Malshare. http://malshare.com/. Accessed: 2019-02-12.

[9] SAAD, Muhammad; TASEER, Muhammad. The study of the anti-debugging techniques and their mitigations. International Journal for Electronic Crime Investigation, 2022, 6.3: 33-44.

[10] J. -W. Kim, J. Bang, Y. -S. Moon and M. -J. Choi, "Disabling Anti-Debugging Techniques for Unpacking System in User-level Debugger," 2019 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea (South), 2019, pp. 954-959, doi: 10.1109/ICTC46691.2019.8939719. keywords: {Malware;Debugging;Registers;Kernel;Computer science;malware;anti-debugging;anti-anti-debugging;packing;portable executable file},

[11] Olaimat, M. N., Maarof, M. A., & Al-rimy, B. A. S. (2021, January). Ransomware anti-analysis and evasion techniques: A survey and research directions. In 2021 3rd international cyber resilience conference (CRC) (pp. 1-6). IEEE.

[12] NavyTitanium, "Fake-Sandbox-Artifacts," GitHub repository, https://github.com/NavyTitanium/Fake-Sandbox-Artifacts. Accessed: May 22, 2024.