악성코드 동적분석 동향

황호*, 문대성*,**, 김익균**
*과학기술연합대학원대학교(UST)
**한국전자통신연구원
e-mail:{kcats, daesung, ikkim21}@etri.re.kr

Trend and Issue Dynamic Analysis for Malware

Ho Hwang*, Daesung Moon*,**, Ikkun Kim**
*Dept of Information Security, Korea University of Science and Technology(UST)
**Network Secuirty Research Laboratory, Electronics and Communications Research Institute(ETRI)

요 약

인터넷이 발전하면서 사이버 공격이 증가하고 있으며, 사이버 공격에 사용되는 악성코드도 점차 지능화 되고 있다. 악성코드 탐지에는 정적분석을 통해 악성코드의 특정패턴을 비교하는 시그니처 기반 접근법이 널리 사용되고 있으며, 높은 탐지율과 빠른 탐지속도를 보인다. 그러나 알려지지 않은 신종 악성코드(0-day)와 실행압축, 난독화 등에 의한 변종 악성코드를 분석하기에 한계가 있다. 더욱이 악성코드의 정적분석은 많은 노력과 시간이 소모되는 작업이며, 최근 악성코드들은 대부분 정적분석 우회기술이 적용되어 대량으로 유포되는 실정이다. 그러므로 악성코드를 직접 실행시켜 발생되는 이벤트들을 수집하여 의미를 분석하는 동적분석이 활발하게 연구되고 있다. 본 논문에서는 악성코드에 적용된 동적분석 우회기술에 관하여 기술하고 나아가 동적분석 우회기술이 적용된 악성코드를 탐지하기 위한 방법에 관한 기술동향을 소개한다.

1. 서론

인터넷 등 네트워크 기술이 발전하면서 악성코드가 접근할 수 있는 영역이 확대되었다. 악성코드의 제작목적이 해커의 지식과시에서 금전적인 이득으로 바뀌면서 더욱 지능화되고 있다. 이에 대응하기 위해 정적분석을 통해 악성코드의 특정패턴을 비교하는 시그니처 기반 접근법이 널리 사용되고 있다. 이 방법은 오탐 및 미탐을 최소화하고 빠른 탐지가 가능하며 모든 실행경로를 파악할 수 있는 장점을 가진다. 하지만 알려지지 않은 악성코드와 실행압축, 난독화를 적용한 변종 악성코드를 분석하는데 한계가 있다[1]. 또한, 최근 정적분석 우회기술이 적용된 악성코드들이 다량으로 유포되고 있고, 2014년 12월에 안랩에서 한달동안 수집한 악성코드 샘플이 607만 9,293건으로 확인되었다[2].

이처럼 정적분석 우회기술이 적용된 악성코드의 대량유포에 대응하기 위해 자동화 가능한 동적분석 방법들이 연구되고 있다. 본 논문에서는 동적분석 우회기술에 대해 알아보고 이를 탐지하기 위한 노력을 소개한다. 2장에서는 동적분석 환경에 대해 소개하고, 3장에서는 동적분석 우회기술을 이용한 악성코드를 소개한다. 4장에서 동적분석 우회기술 탐지법에 대해 소개하고, 5장에서 결론을 맺는다.

2. 동적분석 환경

소스코드를 역공학으로 직접 분석하는 정적분석과는 달리 동적분석은 악성코드가 실행하는 과정에서 발생되는 행위들의 의미를 분석하는 접근법이다. 그림 1처럼 분석환경에서 악성코드에 의해 발생되는 이벤트들을 수집하고 이를 토대로 악성코드 분석가가 악성여부를 판단한다. 동적분석 진행과정은 다음과 같다. ①동적분석 환경에서 의심가는 악성파일을 실행한다. ②악성코드가 OS나 DLL과 상호작용한 이벤트들(API Call, System Call 등)을 수집한다. ③수집된 악성코드의 이벤트들을 분석하여 악성 여부를 확인한다.

(그림 1) 악성코드 동적분석 환경

대량으로 유포되는 악성코드에 대응하기 위해서 동적분석 환경의 자동화가 필요하다. 이를 위해 실제로 사용하는 PC보다는 특정시점으로 복구 가능한 환경이 선호되며, 표 1과 같이 3개로 나눌 수 있다[3]. 가상머신과 에뮬레이터를 이용하는 방법은 실제 하드웨어가 아닌 특정 소프트웨어에 의한 가상환경이므로 특정 하드디스크나 프로세스

등을 사용하는 특징이 있다. 표 1에서 다중실행은 한대의 PC에 다수의 분석환경을 구성하고 각 환경마다 악성코드를 하나씩 실행하는 것을 의미한다.

<표 1> 동적분석 자동화를 위한 가상환경

	다중실행	환경특성	프로그램
가상 머신	가능	가상	VMware
에뮬레이터	가능	가상	QEMU
복구 프로그램	불가능	실제	Ghost

3. 동적분석 우회기술을 이용한 악성코드

악성코드 제작가는 디버깅을 통한 정적분석, 가상환경을 이용한 동적분석과 같이 분석여부를 악성코드가 스스로 판단하도록 만든다. 만약, 악성코드가 자신이 분석된다고 판단하면 악성행위를 숨긴 채 정상 행위만을 실행하거나 스스로 종료한다. 악성코드의 분석환경 판단 기준을 Chen[4] 등이 표 2와 같이 4가지로 분류했다.

<표 2> 악성코드의 분석환경 판단 기준

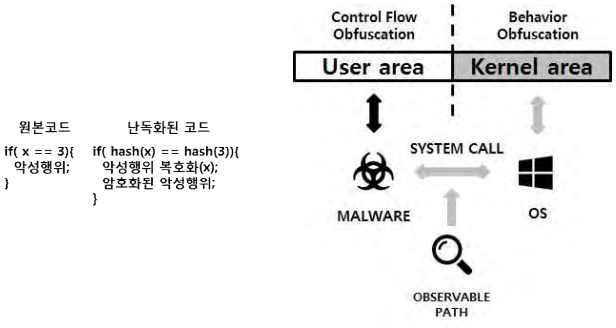
Category	example
하드웨어	VmWare's "pcnet32" QEMU HADRDISK
실행 환경	IsDebuggerPresent() IDT address
외부 프로그램	well-known monitoring application
행위	RDSTC

예를 들어 악성코드는 네트워크 랜카드를 확인하여 pcnet32이 확인되면 Vmware환경에서 분석된다고 판단한다. 또, RDSTC명령을 이용하여 가상 CPU와 실제 CPU의 사이클 차이로 가상환경을 판단한다.

악성코드의 동적분석에 의한 탐지를 어렵게 하기 위한 난독화는 제어흐름 난독화(Control Flow Obfuscation)와 행위 난독화(Behavior Obfuscation)로 나뉜다[5]. 제어흐름 난독화는 그림 2(b)의 검정색 화살표가 나타내는 부분으로 악성코드가 실행되면서 활용하는 사용자 메모리영역(User Memory)의 특정부분을 난독화한다. 행위 난독화는 그림 2(b)의 회색 화살표가 나타내는 부분으로 악성코드가 실행되면서 사용자메모리 영역과 커널영역에 모두 영향을 미치는 시스템콜을 난독화한다.

제어흐름 난독화는 소스코드 레벨에서 진행된다[6]. 그림 2(a)처럼 악성코드는 x가 3인 경우에 악성행위를 실행하는 것을 가정한다. 이때 난독화는 3에 해시함수를 적용하여 정당한 x값을 찾을 수 없게 한다. 또, 3을 확인하지 않고 저장 된 해시값이나 CPU의 FLAG를 이용해서 비교구문을 통과하는 것을 방지하기 위해 암호화된 악성행위가 3으로만 복호화하여 실행되도록 했다.

행위 난독화는 악성코드가 실행하기 위해서 반드시 나타나는 시스템 콜을 난독화한다. Sebastain[5] 등이 시스템콜 삽입(Insertion)과 재정렬(Reordering)으로 난독화한 변형 악성코드를 생성했다.



(a)제어흐름 난독화의 예 (b)동적분석 우회 난독화 분류

(그림 2) 동적분석 우회 난독화

4. 동적분석 우회 탐지기술

동적분석을 우회하는 악성코드를 탐지하기 위해서는 모든 실행 경로를 탐색하여 악성행위를 추출해야 한다. Moser[7] 등은 모든 경로 탐지시 이전 경로로 복귀하는 QEMU에서 모든 분기마다 스냅샷을 찍고 각 분기로 넘어가기 위한 입력 값을 Constraint Solver로 찾는 접근법을 제시했다. Wilhelm[8] 등은 입력 값을 찾지 않고 CPU의 FLAG를 바꿔 모든 실행경로를 찾는 접근법을 제시했다. 이 접근법은 루트킷이 로딩되기 전에 모든 실행경로를 0.5초 내에 찾았다. 앞서 3장에서 보인 제어흐름 난독화는 모든 실행경로 탐색을 어렵게 하는 것이다.

5. 결론

나날이 진화하는 악성코드의 폭발적인 유포에 대응하기 위해서 자동화된 악성코드 탐지 기법이 필요하다. 본 논문은 최근 활발히 연구되고 있는 동적분석 우회 및 탐지기술에 대해 소개했다. 모든 실행경로를 분석하기 위해서는 Moser 등이 접근한 방식보다 Wilhelm 등이 접근한 방식이 실용적이지만 루트킷외의 악성코드 동적분석에 적용하기 위해서는 갈 길이 멀다. 앞으로 악성코드 제작가와 분석가 사이의 전쟁(arm-race)에서 활발한 연구를 통해 분석가가 앞서 갈 수 있는 날을 기대한다.

Acknowledgement

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 정보통신·방송 연구개발 사업의 일환으로 수행하였음.[13-921-06-002, 다중소스 데이터의 Long-term History 분석기반 사이버 표적공격 인지 및 추적 기술개발]

참 고 문 헌

[1] 강부중, 한경수, 임을규. “악성코드 현황 및 탐지 기술”. 정보과학회지, Vol 30, No 1, January 2012, pp.44-53

[2] ASeC RepoRt VOL.60 December, 2014, http://download.ahnlab.com/asecReport/ASEC_Report_Vol.60_Kor.pdf

[3] Egele Manuel, Theodoor Scholte, Engin Kirda, Christopher Kruegel, “A survey on automated dynamic malware-analysis techniques and tools”, ACM Computing Surveys (CSUR), Vol.44, No.2, February 2012, Article 6.

[4] Xu Chen, Jon Andersen, Z. Morley Mao, Michael Bailey, Jose Nazario, “Towards an understanding of anti-virtualization and anti-debugging behavior in modern malware”, Dependable Systems and Networks With FTCS and DCC, June 2008, pp.177-186

[5] Sebastian Banescu, Tobias Wüchner, Marius Guggenmos, Martín Ochoa, Alexander Pretschner. FEEBO: An Empirical Evaluation Framework for Malware Behavior Obfuscation. arXiv preprint arXiv:1502.03245.

[6] Monirul Sharif, Andrea Lanzi, Jonathon Giffin, Wenke Lee, “Impeding Malware Analysis Using Conditional Code Obfuscation”, NDSS, February 2008.

[7] Andreas Moser, Christopher Kruegel, and Engin Kirda. "Exploring multiple execution paths for malware analysis.", Security and Privacy, May 2007, pp. 231-245.