악성 봇 전염 행동 API 및 파라미터 수집 프로그램 구현

황유동*, 유승엽**, 박동규***
*순천향대학교 정보보호학과
**(주)지엔텔
***순천향대학교 정보통신공학과
e-mail:hwangyudong@gmail.com

Malicious Bot API and Parameter Acquisition program Implementation

Seung-Yeop Yoo*, Yu-Dong Hwang**, Dong-Gue Park***
*Dept of Information Security Engineering, SoonChunHyang University
**GNTEL CO. LTD
***Dept of Infomation and communication Engineering, SoonChunHyang University

요 약

본 논문에서는 커널 모드에서 악성 봇이 호스트를 전염 시키는 순간 나타나는 일반적인 행동 특성들을 기반으로 효과적인 악성 봇 탐지가 가능한 프로그램을 구현하였다. 구현된 프로그램은 false-positive(오탐지)를 줄이기 위해서 악성 봇의 전염 과정에서 발생하는 복제 행동, 레지스트리 등록, uninstall 등록, 복제된 파일의 경로 정보 그리고 사용할 API 임포트 정보 등과 같은 악성 행위 탐지 기준 6가지를 고려한다.

1. 서론

최근 악성 봇은 다양한 악성 행동을 하는 상당히 발전된 바이러스의 특성을 보여 주고 있다. 특히 안티 바이러스들의 악성프로그램 분석을 방해하기 위해 분석 환경에서 비활성화 되거나 분석을 지연시키고 실제 사용자 환경에서는 자신을 은닉하며 원본을 제거하거나 레지스트리, 파일 생성 및 수정행동을 숨겨 일반 프로그램과 잘 구별이 되지 않도록 하는 노력들을 볼 수 있다. 기존 악성 프로그램의 탐지 기법에는 크게 정적 분석, 동적 분석 탐지 기법이 있다. 악성 프로그램이 실행이 되기 전에 사용될 행동들의 시그니처를 통해 탐지하는 정적 분석기법은 코드 난독화 회피기법을 이용하여 쉽게 회피되며 프로그램 행동을 실시간으로 감시하여 악의적 행동을 판단하는 동적 분석기법은 불필요한 행동들을 보여 일반 프로그램과 구별을 어렵게 만드는 회피전략들을 갖고 있다. 동적 분석 기법을 통한 탐지 기법은 탐지 기법 자체가 복잡하며 false-positive가 높은 비효율적인 알고리즘을 사용하고 있어 일반프로그램의 정상적인 운영을 저해하는 단점을 가지고 있다[1].

본 논문에서는 최대한 false-positive를 줄이며 효율적으로 악성 프로그램이 호스트 컴퓨터를 전염시키고 악성 행위를 하기 전에 호스트 컴퓨터를 전염하는 행동을 탐지하여 전염에 뒤 따르는 피해를 미리 막기 위해 대부분의 악성 프로그램의 전염 과정에서 볼 수 있는 복제 행동, 레지스트리 등록, 복제된 파일의 경로, uninstall 정보의 등록, 사용자 환경에서의 루트킷 그리고 API 임포트 정보 등을 고려한 악성 봇의 6가지 행위[13]에 대한 API를 커널모드에서 수집할 수 있는 프로그램을 구현하였다.

구현된 프로그램은 6가지 악성 행위를 모니터링하여 실행된 프로그램이 악성 봇인지 아닌지를 판단한다.

본 논문의 2장에서는 악성 봇의 전염 행동기반 탐지 알고리즘[13]을 설명하였고, 3장에서는 2장에서 설명한 알고리즘을 적용하여 구현된 커널모드에서의 API 수집 프로그램을 설명하고 4장에서 결론을 맺는다.

2. 악성 봇의 전염 행동기반 탐지 알고리즘[13]

2.1 악성 봇 전염 행동 모델

본 장에서는 악성 봇이 호스트 컴퓨터를 전염시키는 동안 발생 할 수 있는 행동들을 특성화 하여 일반 프로그램의 설치와 구별하여 탐지하는 기법을 제안한다. 프로그램의 설치 행동을 분석하기 위해서 프로세스의 행동들을 감시하는 ProcessMonitor, 파일 시스템의 변화를 감시하는 Filemon, 그리고 레지스트리관련 정보를 감시하는 Regmon을 사용하였으며, 가상환경이 아닌 실제 사용자 환경에서 프로그램의 설치 행동을 분석하기 위해 detour 라이브러리를 가공하여 만든 API 추출 DLL 및 분석 도구를 사용하였다. 다양한 봇들의 행동을 분석한 결과 그림 1과 같이 봇이 호스트를 전염시키는 과정을 얻을 수 있었다.

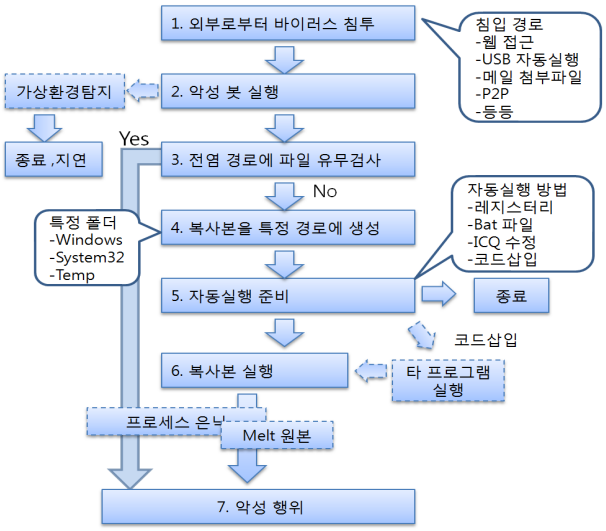

그림 1. 일반적인 악성 봇의 전염과정

봇 제작자 의지에 따라 4번 5번의 순서가 바뀐다거나 약간의 순서 차이는 있으나 행동들의 순서적 연관성을 보고 탐지하는 것이 아니므로 고려될 사항이 아니다. 위 전염과정을 토대로 악성 봇의 일반적인 전염 흐름이나 최근 악성 봇이 갖는 기능을 쉽게 알 수 있다.

<표 1> 6가지 탐지 기준 행동

	6가지 탐지 기준 행동
1B	자동실행준비
2B	uninstall 정보 미등록
3B	자기복제 후 실행
4B	원본 제거 행동
5B	악성 APIs 임포트
6B	DLL 삽입

그림 1에서 주변이 점선으로 이루어진 행동은 악성 봇이 갖는 기능으로 일반 프로그램과 구별 될 수 있으나 반드시 갖추는 기능이 아니다. 일반 프로그램과 구별하기 위해 더 추가되어야 할 행동으로 악성 행위에 필요한 DLL 및 API 임포트와 2번에서 다른 기능을 전혀 다루지 않고 바로 하는 악성행위들이 될 수 있다. 더 많은 행위 들이 있으나 사용되는 빈도와 잘 알려지지 않은 기능으로 본 논문에서는 다루지 않기로 한다. 따라서 대다수의 악성 봇이 호스트 컴퓨터를 전염시키는 행동 특성을 고려하여 악성 봇을 탐지하게 된다. 필요한 탐지 기준으로는 위 표 1과 같은 6가지 행위를 정하였다.
6가지 행위에 대한 설명은 다음과 같다.

1) 자동실행 준비 행동

악성 봇은 호스트 컴퓨터에 전염된 이후로 악성 행위를 지속적으로 해야 함으로 재부팅 후에도 지속적인 봇의 실행이 유지되어야 한다. 일반 프로그램에서도 자동실행 되기 위해 레지스트리 등록을 하기도 하지만 하지 않는 경우가 있다. 악성 봇들 중에 자동 실행되지 않는 일부도 있으나 그러한 프로그램은 지속적인 악성 행위를 하는 봇이라 보기 어렵다.

2) uninstall 정보 등록 및 제거

인스톨러는 보통 설치된 애플리케이션을 uninstall 할 수단을 제공한다. 그리고 그 과정은 일정한 레지스트리에 제거 정보를 등록하는 것으로 진행된다. 다른 한편, uninstall 프로그램은 이 정보가 더 이상 필요하지 않기 때문에 레지스트리로부터 등록된 정보를 삭제한다.

대부분의 일반 프로그램이 이 행동을 하며 악성 봇에서는 볼 수 없는 행동이다.[12] 악성 봇은 uninstall 정보를 등록하거나 제거하는 것이 곧 자신의 존재성을 사용자에게로 알리는 것과 같다고 본다. 따라서 본 행위는 악성 봇과 일반 프로그램을 구별할 수 있는 기준 요소가 될 수 있다.

3) 자기 복제 후 실행

악성 봇의 설치는 대부분 Windows, system32, temp으로 복사본을 만드는 행동을 보인다. 일반 프로그램도 필요한 파일들을 위와 같은 경로에 생성하기는 하나 자기 자신을 그대로 복제하지 않으며 악성 봇은 자신을 위 경로에 복제를 하고 바로 실행 시키거나 재부팅 또는 다른 프로세스에 의해 startup을 걸어 실행되도록 한다. 그러므로 이 특정 폴더들에 복사본을 만들었다고 해서 악성 프로그램이라고는 할 수 없으며 운영체제와 관련된 유용한 프로그램 또는 일반 프로그램도 복사본과 설치에 필요한 파일들을 위 특정 폴더에 생성하기도 하며 악성 봇을 작성할 때 제작자 의지에 따라 어느 폴더에라도 생성이 가능하다. 현재 실행 중인 프로세스의 정보를 얻어와 자신의 파일을 복제하거나 CreateFile()을 이용하여 생성하고 실행하는 행위를 탐지한다.

4) 원본 제거 행동

다음은 일반프로그램이 설치할 때 생성했던 파일들의 제거 행동이다. 생성했던 설치 파일들은 제거하나 설치 프로그램 원본을 제거하는 행동은 일반 프로그램에서는 없는 행동으로 악성 봇에서 프로그램 전염 흔적을 없애기 위하여 하는 "melt" 기능과 확실히 차이가 있다. 악성 봇에서 원본 설치 프로그램을 제거하기 위한 행동 중에 하나로 특정 위치에 bat 파일을 생성하여 타 프로세스로부터 원본이 제거되도록 하는 행동을 보인다. 자기 자신을 제거하기 위해서 자식 프로세스를 생성하거나 다른 프로세스에 코드를 삽입해 자신을 제거하는 경로를 취하고 있다. 그러므로 자식 프로세스와 삽입되는 코드까지 추적을 해서 확인하며 원본을 제거하는 행동을 명확히 판단하기 위해 현재 삽입된 프로세스의 경로를 얻어와 주기적인 검

사로 확인을 시도한다. 또한 시작 프로세스로부터 현재 파일 경로를 유지하여 자식 프로세스에서 제거하거나 dll 및 배치파일에서 제거 행동을 탐지하며 직접 원본 파일이 원본 정보로부터 얻은 경로에 위치하는지를 확인하여 판단하게 된다.

5) 의심스러운 API 임포트

악성 봇은 호스트 컴퓨터를 전염시키는 과정에서 필요한 API를 임포트 하는 행동을 보인다. 임포트된 API들에는 악성 행위를 하기 위한 API들을 볼 수 있다. 악성 행위로는 레지스트리 수정, 암호화, 디버거 탐지, DNS 서버 접근, 강제 종료, 중복 실행 방지, 디버그 탐지 등등이 있으며 해당 DLL로부터 LoadLibrary(), GetProcessAddress()를 이용하여 임포트 한다. 각 DLL과 API들을 보면 어떠한 행동을 할지 유추할 수 으며 임포트된 DLL과 API를 결과 리포트에 생성함과 동시에 악성 행위 중 일부를 포함하는 행동을 탐지한다.

6) 프로세스 은닉 및 DLL 삽입

악성 봇은 실행되고 있는 자신을 숨겨서 사용자 모르게 악성 행위를 하곤 한다. 은닉 방법에는 여러 가지가 있으며 주로 루트킷을 사용한다. 사용자 모드에서의 루트킷은 쉽게 탐지가 되나 커널 루트킷은 탐지하기가 쉽지 않으며 유익한 프로그램들에서 사용하는 루트킷과 구별하기가 쉽지 않아 악성 루트킷을 구별하기가 어렵다. 본 논문에서는 사용자 모드에서 사용되는 은닉 행동을 볼 것이며 발생한 API 뿐만 아니라 ProcessMonitor를 통해 은닉 행동의 발생 유무를 명확하게 판단한다. 루트킷을 이루기 위해 사용자 모드에서 DLL 삽입이 주로 사용된다. 또한 DLL 삽입으로 악성 코드를 시스템의 주요 시스템 실행 파일에 삽입되어 기생하며 악성행위를 하고 있다.
실행 중인 프로세스에 삽입은 CreateRemoteThread 를 이용한 방법을 주로 사용하며 KeServiceDiscriptoTable을 참조하는 방법 등을 탐지한다. 생성되는 프로세스들에는 생성과 동시에 바로 API를 감시하는 DLL을 삽입하여 행동을 지속적으로 감시하며 생성된 파일을 검사하여 자동생성과 관련된 내용을 담고 있는지 또는 설치 파일을 제거하는 행위를 하는 내용이 담겨있는지 확인한다.

3. 악성 봇 전염 행동 API 및 파라미터 수집 프로그램 구현

프로세스들의 특정 행위 수행 여부를 판단하기 위해 프로세스가 실행하는 API를 수집하는 방법으로 커널 영역에 native API의 주소가 저장된 SSDT(System Service Descriptor Table)라는 테이블에 새로운 API 주소로 대체하는 후킹기법을 사용하여 커널 영역에서 수행되는 프로세스들의 함수 호출을 감시할 수 있다. 이와 같은 방법으로 프로세스의 실행시 호출되는 API들의 정보를 감시할 수 있는 프로그램을 구현하였다.

전염 행동의 수행 여부 판단은 각 행위를 위해 반드시 호출해야하는 API 또는 API의 파라미터로 사용된 변수를 기반으로 판단하며, 구현된 프로그램 외에 충분히 신뢰할 만한 도구로 ProcessMonitor, Filemon, Regmon을 사용하여 행동 수행여부를 판단하였다.

커널 영역에서 호출되는 API 명령어의 정보를 감시하기 위해서 그림 2 사용자 모드 응용프로그램으로부터 커널 모드 native API 호출과정 중에서 실제 native API의 주소를 담고 있는 SSDT를 우리가 만든 새로운 함수의 주소로 대체하는 방식을 사용한다.

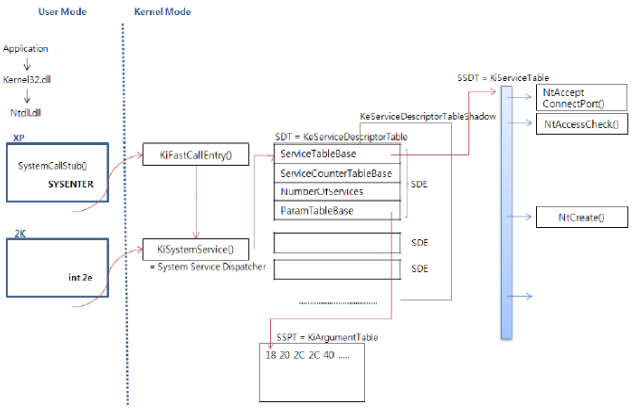

그림 2. 커널 레벨에서 native API 호출 과정

새롭게 만들어진 함수에서 호출된 API에 전달된 정보들을 모두 사용자 모드에 API 모니터링 및 분석 툴로 전송하기위해 그림 3과 같이 구성하여 SSDTHOOK APP에 수집된 정보를 전달한다.

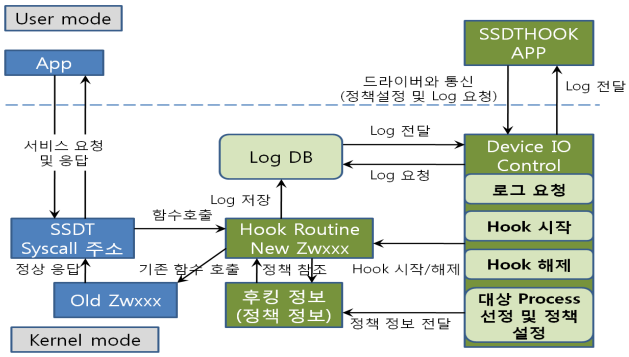

그림 3. 커널 영역에서 분석 및 탐지 프로그램 수행 흐름

그림 3에서 SSDT를 후킹하고 정보를 사용자 영역 SSDTHOOK APP에 전달하는 기능을 갖는 디바이스 드라이버를 작성하고 그림4의 SSDTHOOK APP는 작성된 드라이버를 install하여 모든 프로세스에 대한 native API 감시가 시작된다. 기본 설정은 모든 프로세스에 대한 감시가 이루어지고 특정 프로세스만 감시할 수 있도록 재설정

가능하며 특정 프로세스에 대한 행동 분석은 두 번째 탭 “프로세스 행동 분석”에서 가능하다. 6 가지 행위와 관련된 API를 추출하여 행동여부를 판별한다.

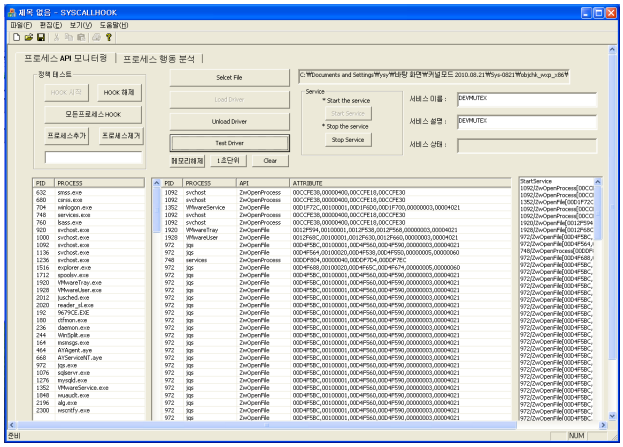

그림 4. 커널 영역에서 분석 및 탐지 프로그램의 실행 예

4. 결론

본 논문에서는 악성 봇 설치 시 갖는 특성을 세분화하여 일반 프로그램과 구별될 수 있는 호스트 레벨의 동적 탐지 기법을 적용한 프로그램을 구현하였다. 구현한 프로그램은 커널모드에서 프로세스의 동작 시 실행되는 API를 수집하고 수집된 API 들은 사용자 모드에서 동작하는 모니터링 프로그램에 전송되고 악성 행위 여부가 판별된다.

본 논문에서 구현된 프로그램은 [13]에서 구현된 사용자 모드에서 동작하는 탐지 프로그램과 동일한 탐지율을 갖는다. 본 논문에서 구현된 디바이스 드라이버를 이용한 커널 모드에서 수집되는 API를 늘리고, 악성 행위 탐지 기법을 좀 더 다양하게 한다면, 커널모드에서 동작하는 악성 봇과 기타 악성행위를 하는 프로그램의 탐지가 가능할 것 으로 사료된다.

참고문헌

[1] Clemens Kolbitsch, Paolo Milani Comparetti, Christopher Kruegel, Engin Kirda§,Xiaoyong Zhou, and XiaoFeng Wang, “Effective and Efficient Malware Detection at the End Host,” usenix security sysposium, 2009.

[2] LI, W, STOLFO, S, STAVROU, A, ANDROULAKI, E, AND KEROMYTIS, A. “A Study of Malcode-Bearing Documents”, In Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA), 2007.

[3] LI, W., WANG, K., STOLFO, S., AND HERZOG, B. “Fileprints: Identifying File Types by N-Gram Analysis” In IEEE Information Assurance Workshop, 2005.

[4] KRUEGEL, C., ROBERTSON, W., AND VIGNA, G. “Detecting Kernel-Level Rootkits Through Binary Analysis“, In Annual Computer Security Applications Conference (ACSAC), 2004.

[5] KINDER, J., KATZENBEISSER, S., SCHALLHART, C., AND VEITH, H. “Detecting Malicious Code by Model Checking” In Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA), 2005.

[6] CHRISTODORESCU, M., JHA, S., SESHIA, S., SONG, D., AND BRYANT, R. “Semantics-Aware Malware Detection”, In IEEE Symposium on Security and Privacy, 2005.

[7] MOSER, A., KRUEGEL, C., AND KIRDA, E. “Limits of Static Analysis for Malware Detection”, In 23rd Annual Computer Security Applications Conference (ACSAC), 2007.

[8] FELT, A., PAUL, N., EVANS, D., AND GURUMURTHI, S. “Disk Level Malware Detection,” In Poster: 15th Usenix Security Symposium, 2006.

[9] KIRDA, E., KRUEGEL, C., BANKS, G., VIGNA, G., AND KEMMERER, R. “Behavior-based Spyware Detection,“ In 15th Usenix Security Symposium, 2006.

[10] YIN, H., SONG, D., EGELE, M., KRUEGEL, C., AND KIRDA, E. “Panorama: Capturing System-wide Information Flow for Malware Detection and Analysis”, In ACM Conference on Computer and Communication Security (CCS), 2007.

[11] Clemens Kolbitsch, Paolo Milani Comparetti, Christopher Kruegel, Engin Kirda§,Xiaoyong Zhou, and XiaoFeng Wang, “Effective and Efficient Malware Detection at the End Host”, usenix security sysposium, 2009.

[12] Yoshiro Fukushima, Akihiro Sakai, Yoshiaki Hori, and Kouichi Sakurai, "Malware Detection Focusing on Behaviors of Process and its Implementation", JWIS, 2009.

[13] “악성 봇의 호스트 전염 특성을 이용한 효과적인 행동기반 탐지기법”, 유승엽, 박동규, 한국정보기술학회, 2010,06