Intermediate 드라이버를 이용한 변종 악성코드 탐지 및 차단 모델

허주승*, 김기천**
건국대학교 컴퓨터공학과
*e-mail:redguru@konkuk.ac.kr
**e-mail:kckim@konkuk.ac.kr

Model for detecting and blocking metamorphic malware using the Intermediate driver

Ju-Seung Heo*, Kee-Cheon Kim**
Dept of Computer Science, KonKuk University

요 약

인터넷의 급격한 성장과 함께 컴퓨터 통신 이용률이 폭발적으로 증가함에 따라 여러 악성코드가 등장하게 되었다. 이러한 악성코드는 시스템의 비정상 동작 유발, 네트워크 성능 저하, 개인정보유출의 문제를 발생시킨다. 현재의 악성코드 분석은 Signature 분석이 대부분이며, Signature 분석은 특정 패턴의 악성코드는 빠르게 탐지하나, 변조된 코드는 탐지하지 못하며, 이미 피해가 널리 퍼진 뒤 분석 및 차단이 가능하다는 단점을 가진다. 따라서 본 논문은 NDIS(Network Driver Interface Specification)를 이용하여 악성코드에 대해 수동적인 Signature 분석의 단점을 보완 하는 시스템 및 네트워크 상태 분석 모델을 제시 하여 보다 능동적인 탐지 및 차단 프로세스를 정의하고, 모델 구현을 위한 방법을 제시한다.

1. 서론

정보통신기술분야의 발전으로 인터넷이 급격히 발전함에 따라 다양한 악성코드들이 나타나게 되었다. 이러한 악성코드는 시스템의 비정상 동작유발, 네트워크 성능저하, 개인 정보유출의 문제를 발생시키며, 각종 범죄와 연관되어 보다 체계적이고 복잡한 공격으로 변화하고 있다.

현재 대부분의 악성코드 탐지 및 차단은 Signature 분석을 통해 이루어지며, 이는 특정 패턴을 가지는 악성코드만 차단하고, 불규칙한 패턴의 악성코드는 탐지하지 못하는 문제점이 있다.[4] 따라서 이를 보완하는 보다 능동적인 악성코드 분석 모델의 설계와 구현이 필요하다.

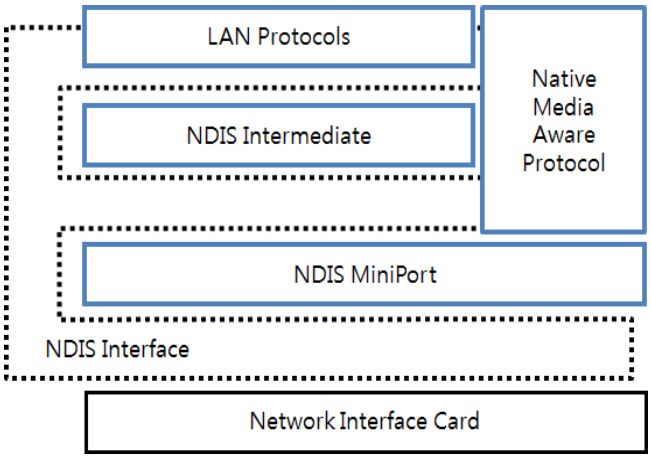
본 논문에서는 NDIS(Network Driver Interface Specification) Intermediate 드라이버를 이용하여 시스템 및 네트워크 상태 변화를 분석하고 커널모드에서 불규칙한 패턴의 악성코드를 탐지 및 차단하는 모델을 제안하여 기존 Signature 분석의 단점을 보완하고자 한다.

각 장의 구성은 2장에서 NDIS를 소개를 하고, 3장에서 기존 Signature 분석을 설명 하며, 4장에서는 상태 변화에 따른 차단 모델을 제시할 것이다. 마지막 5장에서는 결론 및 향후 연구 방향을 통하여 논문을 마무리 하고자 한다.

2. NDIS(Network Driver Interface Specification)

NDIS는 3COM과 Microsoft가 공동으로 개발한 계층형태의 네트워크 구조이다. 여러 가지 네트워크 프로토콜 스택을 사용할 수 있게 운영체제와 NIC(Network Interface Card)가 여러 프로토콜을 사용하기 위해 만든 하나의 개념적인 계층이다.

NDIS는 다음 그림과 같은 구조를 가진다.



(그림 1) NDIS의 구조

** 건국대학교 교수, 교신저자

2-1. MiniPort

MiniPort 드라이버는 NDIS 구조에서 가장 최하위에 위치하는 일반적인 이더넷 드라이버로써, NIC를 통해 데이터를 보내고 받는 것을 포함해 NIC를 직접 제어하는 역할을 한다. 상위 계층의 Intermediate 드라이버나 프로토콜 드라이버와 인터페이스를 수행하기도 하는데, 특히 이더넷 드라이버이기 때문에 하드웨어 수준에서 지원하지 않는 이상 TCP/IP 같은 것을 알 수가 없다. MiniPort 드라이버의 주요 기능은 다음과 같다.

- NIC의 초기화 및 등록 기능
- NIC의 설정변경, 종료, 리셋 등의 제어 기능
- 전송할 패킷 전달 및 수신되는 패킷을 상위 계층으로 전달 기능

2-2. Intermediate

Intermediate 드라이버는 MiniPort드라이버와 Protocol 드라이버 중간에 위치하며, 하위 계층과 상위 계층사이에 통신을 한다. Intermediate 드라이버의 주요 기능은 다음과 같다.

- 이더넷, ATM과 같은 서로 다른 전송계층의 패킷을 매핑하는 변환 기능.
- 수신되는 패킷의 정보와 송신시 전달되는 패킷을 우선적으로 읽고, 관리하는 패킷 필터링이나 블록킹 기능.
- 하나의 가상 어댑터를 상위레벨에 노출시켜 하나 이상의 NIC로 전송 패킷을 분산시키는 Road Balancing 기능.

2-3. Protocol

Protocol 드라이버는 NDIS 구조에서 가장 위에 있는 드라이버로써 TCP/IP 또는 IPX/SPX 스택과 같은 프로토콜 전송을 실행하는 전송 계층 드라이버 내에서 가장 낮은 드라이버로 사용된다. Protocol 드라이버의 기능은 다음과 같다.

- 어플리케이션에 전송되는 데이터를 패킷에 복사하는 기능.
- NDIS함수를 호출하여 하위 레벨의 드라이버로 전송하는 기능.
- 하위레벨에서 수신된 패킷을 받기 위해 프로토콜 인터페이스를 제공하고 수신된 데이터를 클라이언트 어플리케이션에 전송하는 기능.

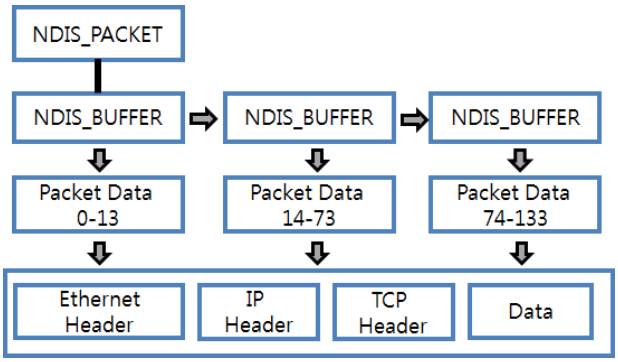
2-4. Intermediate 드라이버를 이용한 패킷 분석

NDIS는 NDIS_PACKET 구조체를 만들어 드라이버간의 통신을 하게 된다.

NDIS_PACKET에서 NDIS_BUFFER 구조체를 얻을 수가 있고, NDIS_BUFFER는 또 다른 NDIS_BUFFER와 연결 된다. 데이터의 용량이 클수록 연결된 NDIS_BUFFER의 개수가 많을 것이고, 처음 NDIS_BUFFER만 14Byte고, 그 다음부터는 60Byte씩 NDIS_BUFFER가 할당된다.

NDIS 상위 레벨에서 NDIS 라이브러리에 접근 할 때는 NdisAllocatePacket API 등을 이용하여 NDIS 패킷을 만들어 통신을 하게 된다.[8]

다음 그림은 NDIS 데이터 구조를 나타낸다.



(그림 2) NDIS 데이터 구조

패킷에 대한 분석은 (그림 2)와 같이 NdisQueryPacket API를 이용하여 패킷을 복사해온 후 NdisQueryBuffer API로 이더넷헤더를 읽는다. 그리고 이더넷헤더에서 패킷 정보를 분석하고, NdisGetNextBuffer API로 다음 버퍼를 복사해온 후 다시 패킷 정보를 분석하는 방법으로 특정 패킷 및 전체 패킷에 대한 정보를 분석 할 수 있다.

다음 그림은 NDIS의 Intermediate 드라이버를 이용한 수신된 패킷 정보를 분석하는 코드이다.

```
1  GetInterfaceHeader(&pBuffer, &pData, &DataLength, &pIfHeader);
2  GetArpHeader(pIfHeader, &pBuffer, &pData, &DataLength, &pArpHeader)
3
4  DbgPrint("op:%.4X Target ip:%.2X%.2X%.2X%.2X Source ip:%.2X%.2X%.2X%.2X
5      Sendermac:%.2X%.2X%.2X%.2X%.2X%.2X Targetmac:%.2X%.2X%.2X%.2X%.2X%.2X",
6      ntohs(pArpHeader->operation),
7      pArpHeader->target_ip[3],
8      pArpHeader->target_ip[2],
9      pArpHeader->target_ip[1],
10     pArpHeader->target_ip[0],
11     pArpHeader->sender_ip[3],
12     pArpHeader->sender_ip[2],
13     pArpHeader->sender_ip[1],
14     pArpHeader->sender_ip[0],
15     pArpHeader->sender_mac[0], pArpHeader->sender_mac[1], pArpHeader->sender_mac[2],
16     pArpHeader->sender_mac[3], pArpHeader->sender_mac[4], pArpHeader->sender_mac[4],
17     pArpHeader->target_mac[0], pArpHeader->target_mac[1], pArpHeader->target_mac[2],
18     pArpHeader->target_mac[3], pArpHeader->target_mac[4], pArpHeader->target_mac[5]);
19     DbgPrint("\n");
20     DbgPrint("hard type: %.4X    proto type: %.4X   hard len: %.2X   proto len: %.2X",
21     ntohs(pArpHeader->hardware_type), ntohs(pArpHeader->protocol_type),
22     pArpHeader->hardware_len, pArpHeader->protocol_len);
```

(그림 3) NDIS에서 구현된 ARP 헤더 분석 코드

3. Signature 분석

Signature 분석은 대부분의 안티바이러스 소프트웨어에서 사용되는 방법으로, 악성코드의 특정 패턴을 추출하여 해당 패턴을 데이터베이스화하고, 악성 코드의 검사 대상이 되는 파일 내에 해당 패턴의 존재 유무를 판단하는 방법이다. 이는 오탐지율과 미탐지율을 최소화 할 수 있고 특정 패턴만 비교하기 때문에 빠르게 동작한다. 그러나 최신 악성코드나 유사 변종코드는 판별할 수 없는 문제점이 있으며, 이미 피해가 널리 퍼진 후 탐지가 가능하다는 단점이 있다.[4]

다음 그림은 Code Red Worm의 Signature[7] 이다.

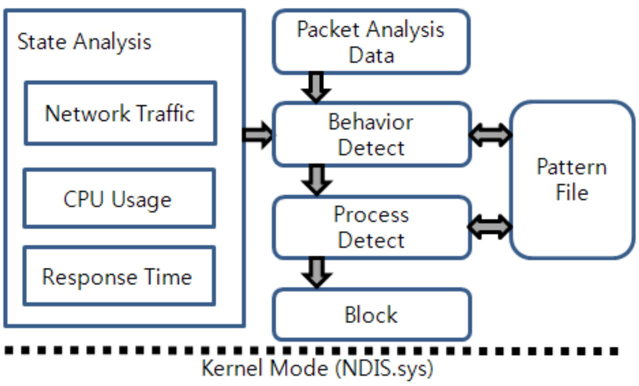
```
/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNN%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%
u7801%u9090%u6858%ucbd3%u7801%u9090%u9090%u8190%u00c3%u0003%u8b00%u531
b%u53ff%u0078%u0000%u00=a
```

(그림 4) Code Red Worm의 Signature

4. 제안하는 모델의 상태분석 과정

본 논문에서 제안하는 모델의 상태분석은 패킷 정보에서 추출한 프로토콜 타입, IP 어드레스, MAC, 사용포트, 길이, 시간정보를 이용하여 IP 스푸핑 패킷발송시도, 과다한 TCP 커넥션 연결 시도, Listen 모드로 연결 대기 시도, 다중 IP 패킷 발송시도, 프로세서 사용량 변화 등의 상태 행동을 감지하고, 해당 행동을 하는 프로세스 및 포트를 차단 한다. 이는 시스템의 상태를 분석하여 Signature 분석 모델이 가지는 수동적 탐지의 단점을 보완 하는 능동적 탐지의 장점을 가진다.

다음 그림은 제안하는 모델의 상태 분석 과정을 나타낸다.

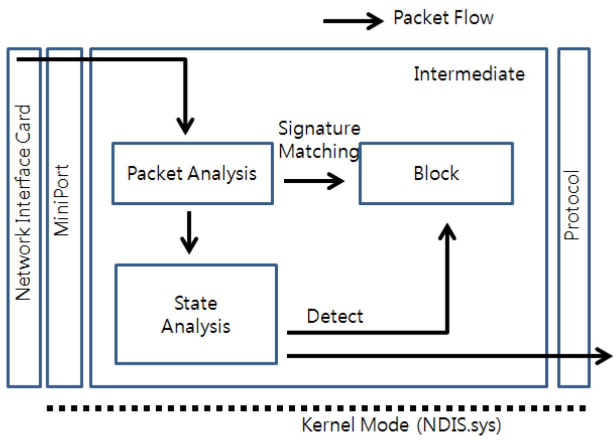


(그림 5) 제안하는 모델의 상태 분석 과정

4-1. 제안하는 모델의 악성코드 차단 과정

제안하는 모델의 악성코드 차단 과정은 먼저 Intermediate 드라이버를 이용하여 (그림 2)와 (그림 3)의 방법으로 해당 악성코드를 분석을 한다. 악성코드에 대한 분석이 끝나면 (그림 4)와 같은 특정 Signature를 찾고, Signature 분석을 마친 패킷은 Protocol 로 전송 된다. 그리고 시스템 및 네트워크 상태를 (그림 5)의 상태 분석 모델로 주기적으로 모니터링 하여, 특정 프로세스가 특정 포트에서 패킷이 과다하게 발생하는 이상 상태가 탐지되면 커널 모드에서 프로세스 및 포트를 차단 한다.

다음 그림은 제안하는 모델의 패킷 흐름과 차단 과정을 나타낸다.



(그림 6) 제안하는 모델의 패킷흐름 및 차단 과정

5. 결론 및 향후 연구 방향

본 논문에서는 특정 패턴의 악성코드만 차단이 가능한 Signature 분석 모델의 단점을 보완하여, 시스템 및 네트워크 상태를 분석 하고, 커널모드에서 악성코드를 탐지, 차단 할 수 있는 모델을 제시하였다. 제시된 모델은 시스템 상태 분석을 통해 Signature 분석과 비교하여 보다 능동적인 탐지 및 차단을 제공하여 줄 것으로 예상되며, 향후 제안된 모델을 구현하여, 악성코드 탐지 및 차단 확률을 분석하고, 다른 모델들과 비교, 분석할 것이다

참고문헌

[1] 이봉석, "윈도우 디바이스 드라이버", 한빛 미디어
[2] 강부중, 한경수, 임을규, "악성코드 현황 및 탐지 기술", 정보과학회지, 2012.01
[3] 한국인터넷진흥원, "악성코드 은닉사이트의 분산적, 동적 탐지를 통한 감염피해 최소화 방안 연구", 2011.06

[4] Lee Ling Chuan, Chan Lee Yee, Mahamod Ismail and Kasmiran Jumari, “Automated Blocking of Malicious Code with NDIS Intermediate Driver”, ICACT 2011, IEEE February 2011.
[5] Zhao Rongcai, Zhang Shuo, "Network traffic generation : A combination of stochastic and self-similar", ICACC2010, p171-175, 2010
[6] Fulvio Risso, Loris Degioanni, "An Architecture for High Performance Network Analysis", ISCC01, p686-693, 2001
[7] http://www.cert.org/advisories/CA-2001-19.html
[8] http://ndis.com