제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

시스템 모니터링을 통한 악성 행위 패턴 분석에 관한 연구

김은영*, 오형근*, 배병철*, 박중길*
*국가보안기술연구소
e-mail : {eykim, hgoh, bcbae, jgpark}@etri.re.kr

A Study on Malicious Behavior Pattern Analysis Using System Monitoring

EunYoung Kim*, HyungGeun Oh*, ByungChul Bae*, JoongGil Park
*National Security Research Institute

요 약

기존의 바이러스 및 악성 코드 백신의 탐지 기법은 대부분 시그너쳐 기반의 패턴 매칭 기법을 사용하고 있다. 이러한 기법의 단점은 새로운 악성 코드가 발생하면 사용자가 매번 시그너쳐를 업데이트를 해야 탐지가 가능하며, 시그너쳐의 업데이트 없이는 알려지지 않은 바이러스 및 악성 코드를 탐지할 수 없다는 것이다. 따라서 이와 같은 패턴 매칭 기법의 단점을 보완하고자 각각의 악성 코드 종류에 따른 시그너쳐를 이용한 탐지 기법이 아닌 악성 행위별 패턴를 이용하여 탐지를 한다면 기존의 기능을 포함한 알려지지 않은 바이러스 및 악성 코드 등을 탐지할 수 있을 것이다. 본 논문에서는 시스템 모니터링을 통하여 악성 행위별 패턴 분석 및 결과에 대해 기술한다.

1. 서론

최근 악성 코드로 인한 피해 사례는 일일히 열거하지 않아도 거의 매일 뉴스 및 신문 등을 통하여 피해 사례가 보도되고 있다. 이러한 피해 사례를 접하고 있는 일반 사용자들은 새로운 악성 코드 및 바이러스 등이 유포가 되면 불편함을 감수하고서라도 매번 자신이 설치한 백신의 데이터 베이스를 업데이트해야 할 것이다. 그러나 만약 여러가지 이유로 군 및 관공서에서 몇몇 사람이 백신을 업데이트 하지 않았다면, 작게는 자신이 사용한 컴퓨터의 자료 유출 및 파괴로 인한 피해를 입을 것이며, 백신을 업데이트 하지 않은 컴퓨터가 중앙 서버라 가정한다면 그 피해는 실로 엄청난 결과를 낳을 수 있을 것이다. 따라서 사용자는 하루에도 수십개의 악성 코드가 발견되는 현실에서 잦은 백신의 데이터 베이스 업데이트로 인한 불편함을 감내해야할 것이다. 이러한 현실은 다수의 백신업체에서 탐지 기법으로 채택하고 있는 시그너쳐 기반의 패턴 매칭 방식을 계속해서 사용한다면 앞으로도 사용자의 백신 업데이트에 따른 불편함은 지속될 수 밖에 없다.

따라서 본 논문에서는 기존의 시그너쳐 기반의 패턴 매칭 기법의 단점 및 한계점을 인지하고 기존의 악성 프로그램을 분석하여 시스템 모니터링을 통해 각각의 악성 행위에서 발생되는 시스템 이벤트들을 모니터링 후 이를 분석한 결과에 대해 기술한다.

2. 기존의 악성 코드 탐지 기법

이 장에서는 기존 백신 제품들의 탐지 기법에 대해 소개하고, 새로 제시되고 있는 휴리스틱 스캐닝 기법(Heuristic Scanning Method)[1]과 행위 제한 기법(Behavior Blocking Method)[2]에 대해서 기술한다.

2.1 시그너쳐 기반의 스캐닝 기법

시그너쳐 기반의 스캐닝 기법은 현재 바이러스 및 트로이 목마 탐지 제품등에서 많이 사용하고 있는 기법으로 해당 악성 프로그램의 특정 스트링을 백신 시스템에 포함한다. 그리고 해당 컴퓨터에 검사하고자하는 파일을 백신에 포함되어있는 악성 프로그램의 시그너쳐와 비교하여 해당 파일이 악성 프로그램인지 판단하게 된다. 따라서 이러한 스캐닝 기법은 탐지 기

법중에서 가장 접근하기 쉽고 탐지 결과 또한 확실한 방법일 수 있지만, 알려지지 않은 악성 코드나 변종 등은 해당 백신 프로그램에서 시그너쳐를 가지고 있지 않는다면 탐지는 불가능하다.

따라서 알려지지 않은 악성 코드등을 탐지하기 위한 시도로 휴리스틱 스캐닝 기법 및 행위 제한 기법 등이 소개되고 있다.

2.2 휴리스틱 스캐닝 기법 및 행위 제한 기법

휴리스틱 스캐닝 기법은 시그너쳐 기반의 스캐닝 기법과 유사하다. 그러나 휴리스틱 스캐닝 기법은 시그너쳐 기반의 스캐닝 기법과 같이 어떤 특정 시그너쳐를 찾는 대신에 보통의 응용 프로그램에서 발견할 수 없는 전문가들의 분석에 따른 경험적 규칙에 기반한 어떤 특별한 명령이나 시그너쳐를 찾게 된다. 따라서 이러한 휴리스틱 기법을 이용하여 구현된 탐지 엔진은 바이러스, 웜 또는 트로이 목마 등 알려지지 않은 새로운 악성행위를 탐지할 수 있다.

휴리스틱 스캐닝 기법은 두가지로 구분할 수 있다. 가중치 기반 시스템(Weight-based systems)와 규칙 기반 시스템(Rule-based systems)이다. 가중치 기반 휴리스틱 엔진은 오래전부터 개발 및 발전된 기법으로 바이러스 행위에 따라 차등적으로 점수를 두어 위험 점수의 합이 특정 점수 이상이면 바이러스라 판단하는 기법이다. 휴리스틱 스캐닝 기법에서의 규칙 기반 시스템은 파일에서부터 어떠한 행위를 분석하고, 그 행위에 관한 규칙을 산출한다. 즉, 바이러스 행위에 해당하는 규칙이 생성되면 해당 규칙을 기반으로 바이러스를 탐지하게 된다. 이 기법은 위에서 설명한 가중치 기반 시스템보다 구현이 복잡하지만 바이러스 행위에 대한 규칙이 생성되어 규칙 기법으로 탐지가 되므로, 알려지지 않은 바이러스 행위 및 바이러스 등을 탐지하기에 용의하다. 하지만 일반적인 문서에서도 발생이 가능한 " normal.dot" 등이 악성 코드 등에서 발견되었다고 해당 함수에 대한 위험도를 높게 책정하였을 경우, 휴리스틱 스캐닝 기법은 ' False-Positive' 가 상당히 높아지게 될 것이다.

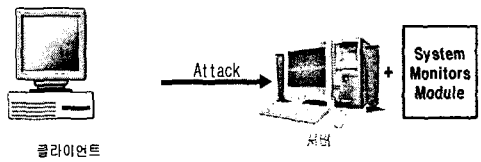
행위 제한 기법은 두가지 시스템으로 분류할 수 있다. 정책 기반 제한 시스템(Policy-Based Systems)은 특정 행위를 모니터링하여 그 행위를 허용할 것인지 제한할 것인지를 결정하는 것이다. 만약 어떤 프로그램이 운영체제에 어떠한 행위를 요청하였을 경우, 정책 기반 시스템은 정책 데이터베이스에 비교하여 해당 행위를 허용할 것인지 제한할 것인지 결정하는 것이다. 이와는 대조적으로 전문가 기반 시스템(Expert-Based Systems)은 전문가가 전체 바이러스를 분석하여 그들이 시스템에 끼치는 바이러스 행위를 분석하고 의심이 가는 행위에 대해 직접 제한을 가할 수 있게 구성한다. 또한 이러한 전문가 기반 시스템은 이미 알려진 바이러스 코드의 80%가 시작 프로그램에 해당하는 시스템 파일을 접근하기 전에 레지스트리를 먼저 접근하기에 이에 준하는 정책을 설정할 수 있다. 따라서 행위 제한 기법에서는 시작 프로그램의 레지스트리 영역을 접근하여 수정을 하게되면 이 행위에 대해 제한을 하게 된다. 그러나 이러한 전문가 시스템 또한 여전히 'False Positive' 요소를 내제하고 있다.

3. 시스템 모니터링을 통한 악성 행위 실험

이 장에서는 시스템 모니터링(*FileMon, TDIMon*)을 이용하여 악성 행위별 패턴을 추출한 실험에 대해 기술한다.

3.1 실험 환경

시스템 이벤트 모니터링을 통한 악성 행위 실험을 위한 실험 환경은 다음과 같다. 실험 시스템 운영체제는 윈도우 NT 4.0 이며, 사용되는 트로이 목마는 백오리피스, 넷버스, Cafeini, Kuang 을 이용하였다.



(그림 1) 실험 환경

클라이언트는 서버에 설치되어 있는 트로이 목마를 조정하는 프로그램이 설치가 되고, 서버에서는 클라이언트의 조정을 받는 트로이 목마 서버 프로그램이 설치가 된다. 서버에는 악성 행위에 따른 패턴을 추출하기 위해 시스템 모니터링 모듈이 설치가 된다. 시스템 모니터링 모듈로는 시스인터널스(SYSinternals)에서 만든 FileMon 과 TDIMon 을 이용한다.

3.2 백오리피스를 이용한 악성 행위 실험

클라이언트가 서버에 악성 행위를 하기 위한 초기 작업으로 연결 요청시 보여지는 행위에 대해 실험한 결과이다.

```
FileMon 결과
공통점이 없다.

TDIMon 결과
1       오전 10:10:43          Backdoor.Win32.:      807EF028
        IRP_MJ_CREATE          TCP:Connection obj
        SUCCESS    Context:0x807EBE68
2       오전 10:10:43          Backdoor.Win32.:      807EF028
        TDI_ASSOCIATE_ADDRESSTCP:Connection obj
        SUCCESS    TCP:0.0.0.0:20000
3       오전 10:10:43          Backdoor.Win32.:      80A12DA8
        IRP_MJ_DEVICE_CONTROLTCP:<none>              SUCCESS
        IOCTL_TCP_SET_INFORMATION_EX
```

(그림 2) 연결 설정 행위에 대한 로그 분석 결과

실험 결과 FileMon 의 로그에서는 반복된 행위에 따른 공통점을 발견할 수 없었으며, TDIMon 에서는 *'IRP_MJ_CREATE -> TDI_ASSOCIATE_ADDRESS -> IRP_MJ_DEVICE_CONTROL'* 세가지 요청 메시지가 반복하여 사용함을 알 수가 있다.

다음은 파일 삭제 행위에 대한 시스템 이벤트 분석 결과이다.

```
FileMon 결과
1       오전 10:31:34          Backdoor.Win32.:101
        FSCTL_IS_VOLUME_MOUNTED          D:₩WINNT₩system32
        SUCCESS
2       오전 10:31:34          Backdoor.Win32.:101
        IRP_MJ_CREATE          D:₩util₩monitor₩delete1₩1.exe
        SUCCESS    Attributes: Any Options: Open
3       오전 10:31:34          Backdoor.Win32.:101
        IRP_MJ_SET_INFORMATION
        D:₩util₩monitor₩delete1₩1.exe   SUCCESS
        FileDispositionInformation
4       오전 10:31:34          Backdoor.Win32.:101
        IRP_MJ_CLEANUP         D:₩util₩monitor₩delete1₩1.exe
        SUCCESS
5       오전 10:31:34          Backdoor.Win32.:101
        IRP_MJ_CLEANUP         D:          SUCCESS
6       오전 10:31:34          Backdoor.Win32.:101
        IRP_MJ_CLOSE           D:₩util₩monitor₩delete1₩1.exe
        SUCCESS

TDIMon 결과
4       오전 10:31:48          Backdoor.Win32.:      80AB4A68
        TDI_SEND   TCP:129.254.163.105:20000
        SUCCESS    Length:59
5       오전 10:31:51          Backdoor.Win32.:      80AB4A68
        TDI_SEND   TCP:129.254.163.105:20000
        SUCCESS    Length:59
```

(그림 3) 파일 삭제 행위에 대한 로그 분석 결과

FileMon 에서는 *'FSCTL_IS_VOLUME_MOUNTED -> IRP_MJ_CREATE -> IRP_MJ_SET_INFORMATION -> IRP_MJ_CLEANUP -> IRP_MJ_CLEANUP -> IRP_MJ_CLOSE'* 의 파라미터 여섯개가 반복됨을 알 수 있다. 또한 TDIMon 에서는 *'TDI_SEND'* 라는 Request 메시지 로그만 남는다. 다음은 파일 보기에 대한 시스템 이벤트 로그 분석 결과이다.

```
FileMon 결과
166     오전 10:26:10          Backdoor.Win32.:101
        FSCTL_IS_VOLUME_MOUNTED          D:₩WINNT₩system32
        SUCCESS
167     오전 10:26:10          Backdoor.Win32.:101
        IRP_MJ_CREATE
        D:₩WINNT₩System32₩drivers₩etc₩hosts        SUCCESS
        Attributes: N Options: Open
168     오전 10:26:10          Backdoor.Win32.:101
        IRP_MJ_CLEANUP         D:          SUCCESS
169     오전 10:26:10          Backdoor.Win32.:101
        IRP_MJ_READ
        D:₩WINNT₩System32₩drivers₩etc₩hosts        SUCCESS
        Offset: 0 Length: 4096
170     오전 10:26:10          Backdoor.Win32.:101
        FASTIO_READ
        D:₩WINNT₩System32₩drivers₩etc₩hosts        END    OF
FILE    Offset: 737 Length: 4096
171     오전 10:26:10          Backdoor.Win32.:101
        IRP_MJ_CLEANUP
        D:₩WINNT₩System32₩drivers₩etc₩hosts        SUCCESS

TDIMon 결과
공통점이 없다.
```

(그림 4) 파일 보기 행위에 대한 로그 분석 결과

FileMon 에서는 *'FSCTL_IS_VOLUME_MOUNTED -> IRP_MJ_CREATE -> IRP_MJ_READ-> FASTIO_READ-> IRP_MJ_CLEANUP'* 위의 5 가지 요청 메시지가 주기적으로 나타내며 파일 크기에 따라 *'FASTIO_READ'* 메시지의 수가 달라진다. TDIMon 의 모니터링에서는 어떠한 이벤트들의 공통점을 찾을 수 없다.

백오리피스 트로이 목마를 이용하여 테스트한 결과 각각의 행위에 대한 시스템 이벤트들의 주기적인 특성을 나타내고 있음을 보여주고 있다.

3.3 넷버스, Cafeini, Kuang 를 이용한 악성 행위 실험

백오리피스 의 로그를 토대로 넷버스, Cafeini, Kuang 를 위의 백오리피스에서 수행했던 행위에 대해 시스템 이벤트 로그를 분석하였다. 먼저 연결 설정 요청에 대한 시스템 이벤트 분석 결과이다.

```
FileMon 결과
공통점이 없다.

TDIMon 결과

1       오전 10:10:43          Backdoor.Win32.:      807EF028
        IRP_MJ_CREATE          TCP:Connection obj
        SUCCESS    Context:0x807EBE68
2       오전 10:10:43          Backdoor.Win32.:      807EF028
        TDI_ASSOCIATE_ADDRESSTCP:Connection obj
        SUCCESS    TCP:0.0.0.0:20000
3       오전 10:10:43          Backdoor.Win32.:      80A12DA8
        IRP_MJ_DEVICE_CONTROLTCP:<none>              SUCCESS
        IOCTL_TCP_SET_INFORMATION_EX
```

(그림 5) 연결 설정 행위에 대한 로그 분석 결과

백오리피스, 넷버스, Cafeini, Kuang 을 가지고 연결 요청에 대한 로그 분석 결과 File Mon 에서는 어떠한 시스템 이벤트에서는 공통점을 발견할 수 없었고, TDIMon 에서는 넷버스, Cafeini, Kung 는 *'IRP_MJ_CREATE -> TDI_ASSOCIATE_ADDRESS'* 이벤트가 호출되었고, 백오리피스에서는 *'IRP_MJ_CREATE -> TDI_ASSOCIATE_ADDRESS -> IRP_MJ_DEVICE_CONTROL'*이 벤트가 1 번의 행위에 따라 주기적으로 발견되었다. 다음은 파일 삭제에 대한 분석 결과이다.

```
FileMon 결과
1       오전 10:31:34          Backdoor.Win32.:101
        FSCTL_IS_VOLUME_MOUNTED          D:₩WINNT₩system32
        SUCCESS
2       오전 10:31:34          Backdoor.Win32.:101
        IRP_MJ_CREATE          D:₩util₩monitor₩delete1₩1.exe
        SUCCESS    Attributes: Any Options: Open
3       오전 10:31:34          Backdoor.Win32.:101
        IRP_MJ_SET_INFORMATION
        D:₩util₩monitor₩delete1₩1.exe   SUCCESS
        FileDispositionInformation
4       오전 10:31:34          Backdoor.Win32.:101
        IRP_MJ_CLEANUP         D:₩util₩monitor₩delete1₩1.exe
        SUCCESS
5       오전 10:31:34          Backdoor.Win32.:101
        IRP_MJ_CLEANUP         D:          SUCCESS
  6     오전 10:31:34          Backdoor.Win32.:101
        IRP_MJ_CLOSE           D:₩util₩monitor₩delete1₩1.exe
        SUCCESS

TDIMon 결과
1       오전 10:31:48          Backdoor.Win32.:      80AB4A68
        TDI_SEND   TCP:129.254.163.105:20000
        SUCCESS    Length:59
```

(그림 6) 파일 삭제 행위에 대한 로그 분석 결과

파일 삭제인 경우 FileMon 의 결과는 두가지로 나눌 수 있다. 먼저 *'IRP_MJ_CREATE -> IRP_MJ_SET_INFORMATION -> IRP_MJ_SET_INFORMATION -> IRP_MJ_CLEANUP -> IRP_MJ_CLEANUP -> IRP_MJ_CLOSE'*와 같은 행위 패턴이 보이는 것은 Kuang, 넷버스 이고, *'FSCTL_IS_VOLUME_MOUNTED -> IRP_MJ_CREATE -> IRP_MJ_SET_INFORMATION -> IRP_MJ_SET_INFORMATION -> IRP_MJ_CLEANUP -> IRP_MJ_CLEANUP ->*

*IRP_MJ_CLOSE'*와 같은 행위 패턴 형태를 보이는 것은 백오리피스, Cafeini 이다. TDIMon 은 백오리피스, Cafeini, Kuang 가 *'TDI_SEND'* 메시지 하나만 보인다. 파일 리스트 보기 행위에 대한 로그 분석 결과는 다음과 같다.

```
FileMon 결과
1       FSCTL_IS_VOLUME_MOUNTED         D:₩WINNT₩system32
        SUCCESS
2       IRP_MJ_CREATE       D:₩                 SUCCESS
        Attributes: Any Options: Open Directory
3       IRP_MJ_DIRECTORY_CONTROL        D:₩         SUCCESS
        FileBothDirectoryInformation: 4 IRP_MJ_CLEANUP
        D:₩         SUCCESS
5       IRP_MJ_CLOSE        D:₩         SUCCESS

TDIMon 결과
1       TDI_EVENT_RECEIVE    TCP:0.0.0.0:17300
        129.254.163.73:1085  SUCCESS    Length:1024   Flags:
ENTIRE_MESSAGE LOOKAHEAD DISPATCH
2       TDI_SEND   TCP:0.0.0.0:17300    129.254.163.73:1085
        SUCCESS-4  Length:1024
```

(그림 7) 파일 리스트 보기 행위에 대한 로그 분석 결과

먼저 FileMon 인 경우 두가지로 나눌 수 있다. 백오리피스, 넷버스, Kuang 은 *'FSCTL_IS_VOLUME_MOUNTED -> IRP_MJ_CREATE -> IRP_MJ_DIRECTORY_CONTROL -> IRP_MJ_CLEANUP -> IRP_MJ_CLOSE'*의 형태를 보이고 있으며, Cafeini 는 *'FSCTL_IS_VOLUME_MOUNTED'* 의 이벤트 만이 보이고 있다. TDIMon 의 결과는 세가지로 나눌 수 있다. 먼저 넷버스 는 *'TDI_EVENT_RECEIVE'* 이벤트 패턴을 보이고 있고, Kuang 는 *'TDI_EVENT_RECEIVE -> TDI_SEND'* 이벤트 패턴을 보이고 있다. Cafeini, 백오리피스는 *'TDI_SEND'* 이벤트 패턴만이 반복적으로 보이고 있다. 파일 업로드 행위에 대한 분석 결과이다.

```
FileMon 결과
1       오후 7:02:25.101     Patch.exe:251
        FSCTL_IS_VOLUME_MOUNTED         C:₩해킹서버₩netbus
        SUCCESS
2       오후 7:02:25.101     Patch.exe:251
        IRP_MJ_CREATE       C:₩Cpusys.sys          SUCCESS
        Attributes: N Options: Overwritelf
3       오후 7:02:25.101     Patch.exe:251
        IRP_MJ_WRITE        C:₩Cpusys.sys          SUCCESS
        Offset: 0 Length: 4144
4       오후 7:02:25.101     Patch.exe:251
        IRP_MJ_CLEANUP      C:₩Cpusys.sys          SUCCESS
TDIMon 결과
1       오후 7:02:24         Patch.exe:251          80A22CC8
        TDI_EVENT_RECEIVE    TCP:0.0.0.0:12345
        129.254.163.73:1290  SUCCESS    Length:31     Flags:
ENTIRE_MESSAGE LOOKAHEAD DISPATCH
2       오후 7:02:24         Patch.exe:251          80A04CC8
        TDI_SEND   TCP:0.0.0.0:12345    129.254.163.73:1290
        SUCCESS-22 Length:12
```

(그림 8) 파일 업로드 행위에 대한 로그 분석 결과

FileMon 은 *'FSCTL_IS_VOLUME_MOUNTED -> IRP_MJ_CREATE -> IRP_MJ_WRITE -> IRP_MJ_CLEANUP'* 의 이벤트가 반복적으로 나타난다. TDIMon 은 넷버스 와 Kuang 은 *'TDI_EVENT_RECEIVE -> TDI_SEND'*의 이벤트가 반복적으로 나타나며, Cafeini 는 *'TDI_SEND'* 이벤트만이 반복적으로 나타난다. 특이한 점은 백오리피스의 로그는 FileMon, TDIMon 둘다 나타나지 않았다.

3.4 시스템 모니터링을 통한 악성 행위 실험 결과

시스템 모니터링을 기반으로 악성 행위에 대한 패턴을 찾기 위한 실험을 시도하였을 때는 악성 행위에 대해 모든 트로이안이 적용될 수 있는 시스템 이벤트 패턴을 찾을 수 있을 거라는 가정하에 실험을 착수하였다. 그래서 첫 시도로 백오리피스만을 가지고 악성 행위에 대한 실험을 하였을때 어떤 악성 행위에 대해 시스템 패턴이 도출되는듯 하였으나, 넷버스 등의 다른 트로이안을 가지고 같은 악성 행위에 대한 시스템 이벤트 모니터링 결과 새로운 트로이 목마를 추가할 때마다 서로 상이한 시스템 이벤트의 패턴 양상을 보이고 있다. 또한 어떤 악성 행위에 대한 패턴이 있더라도 특정 악성 행위에 대한 패턴이 아닌 모습을 보이고 있다. 예를 들어 TDIMon 의 패턴에서보면 파일 삭제, 파일 보기, 파일 업로드 모두가 *'TDI_SEND'* 이벤트 메시지를 각자 자신의 악성 행위에 대한 결과로 나타내주고 있다. 따라서 본 논문에서 처음 의도하였던 파일 삭제 행위라면 다른 악성 행위에서는 발견될 수 없는 패턴을 찾을 수 있을꺼라는 가정은 잘못되었음을 나타내고 있다.

4. 결론

본 논문에서는 시스템 모니터링 툴 두가지를 가지고 악성 행위별 패턴을 추출하고자 실험을 하였다. 만약 악성 행위별 패턴 추출이 가능하다면 기존의 시그너쳐 기반의 스캐닝 기법의 단점인 알려지지 않은 악성 코드를 탐지할 수 있을뿐 아니라 새로운 악성 코드 탐지 기법으로 제안되고 있는 행위 제한 기법등의 'False-Positive'를 줄일 수 있는 기초 정보로 사용할 수 있을 것이다. 그러나 실제 실험한 결과 특정 악성 행위별 패턴은 실험을 거듭할수록 서로 다른 패턴을 나타내주고 있어서 본 논문에서 가정하였던 악성 행위별 패턴은 FileMon 과 TDIMon 을 이용해서는 추출할 수 없다라는 결론을 얻게 되었다. 따라서 악성 행위별 패턴을 추출하기 위해서는 본 논문에서 실시하였던 시스템 모니터링 방법 이외에 다른 방법을 모색해야 할 것이다.

참고문헌

[1] M. Schmall. "Heuristic Techniques in AV Solutions: An Overview ", http://online.securityfocus.com/infocus/1542, Feb, 2002.

[2] C. Nachenberg. "Behavior Blocking : The Next Step in Anti-Virus Protection", http://online.securityfocus.com/infocus/1557, March, 2002.

[3] 김은영, 오형근, 배병철 "시스템 모니터링을 통한 트랩도어 탐지에 관한 연구", WISC 2002, pp.312~327

[4] 김은영, 강인곤, 고재영, "판단-대응 호스트 기반 분산 침입 탐지 시스템", WISC 2001, pp.215~223

[5] SYSinternals, http://www.sysinternals.com/