[1]리눅스 기반에서 실시간 탐지 기법을 이용한 보안 기술 연구

김미영, 문영성
숭실대학교 컴퓨터학부
e-mail : mizero31@sunnuy.soongsil.ac.kr, mun@computing.ssu.ac.kr

A Study on the Security Technology using Real-Time Intrusion Detection in Linux

Miyoung Kim, Youngsong Mun
Dept. of Computing , Soongsil University

요 약

정보 인프라의 고도화와 인터넷 사용의 폭발적인 증가로 인해 다양한 형태의 정보를 대량으로 교환할 수 있는 환경이 마련되었으며, 정보기술의 보편화를 통해 누구든지 쉽게 기술을 습득하고 이용하게 되었다. 인터넷 사용자는 크게 일반 사용자 및 적대적 사용자로 분류될 수 있으며, 특히 적대적 사용자는 정보의 불법적인 유출, 악용, 파괴할 수 있는 고도의 기술을 지닌 그룹으로서 인터넷의 존재 자체를 위협할 수 있는 수준이며, 이들의 기술은 날로 지능화되고 자동화되는 추세이다. 정보의 가치가 중요해 지면서 고급 정보에 대한 피해 사례가 늘어가고 있으나, 이를 정확하게 발견하고 신속하게 대처하기 위한 기술의 개발은 아직 초보 단계에 머무르고 있다. 대부분의 보안 시스템이 침입에 대한 탐지 및 대응 기술 개발에 역점을 두고 있으나, 알려지지 않은 침입에 대해서는 정확한 탐지 및 신속한 대응이 어렵다. 본 논문에서는 가상 서비스를 통해 침입자를 유도하고, 침입 과정 및 기법을 학습함으로써 새로운 기법에 대한 신속한 대응책을 수립할 수 있도록 해 주는 HoneyPot 의 구현에 관한 방법을 제시한다.

1. 서론

HoneyPot 은 간단히 '해커 잡는 덫' 이란 뜻의 전문용어 이다. 즉 해커를 잡는 유혹의 꿀단지라는 의미다. HoneyPot 은 보통 해커를 유인하기 위한 위장 서버와 추적 탐지용 소프트웨어로 구성되어 있다. 이 중 핵심은 위장서버이며, 해커를 끌어들이는 덫 역할을 한다. 접근 방법으로는 Virtual Network 내에 HoneyPot 시스템을 추가하는 방법과 위장된 응용프로그램을 통해 가상의 서비스를 제공함으로써 HoneyPot 을 구성하는 방법이 있다. 궁극적으로는 호스트 전체를 가상 시스템으로 구축하여야 한다. Honey Pot 시스템에는 침입자에 대한 집중 감시 기능, 증거 수집 기능, 침입자 추적기능이 있어야 하며, 침입자가 HoneyPot 이라는 것을 눈치채지 못하게 해야 한다

위장 서버는 침입자들의 호기심을 끌 만한 가상 정보들을 올려놓고 침입자들의 침입과 동시에 자동 추적 탐지용 소프트웨어가 작동, 침입자들을 추적한다. 침입자들이 키보드에 어떤 글자를 입력했고 어떤 해킹 방법을 사용했는지도 분석할 수 있다. HoneyPot 시스템은 지난 90 년대 중반 미국 MIT 대학의 데이비드 클록 교수가 제안했지만 기술적인 문제로 실용화되지는 못했다. 그러나 최근 들어 미국의 리코스 테크놀로지, 글로벌 인터그리티 등 보안 업체들이 제품을 발표하면서 기업에 도입되고 있다. HoneyPot 은 간단히 "공격 당해 져야 할 자원"으로 정의할 수 있다. HoneyPot 자체는 보안에 대한 해결책이 아니다. 대신 하나의 툴로서 볼 수 있는데, 침입자를 유인해서 되도록 시스템 안에서 오랫동안 머물게 하면서 그들의 작업 내용 및 침입에 대한 자료를 많이 남기도록 함으

[1] "이 논문은 정보통신부 대학정보통신 연구센터 지원사업의 지원 및 한국소프트웨어진흥원의 관리로 수행되었음"

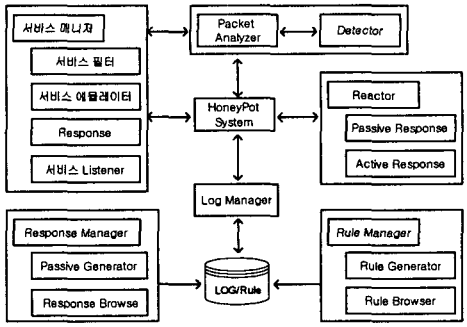
로써, 중요한 분석 자료로 사용된다.
HoneyPot 은 기본적으로 다음과 같은 사항을 만족해야 한다.
1) 반드시 공격 당해야 한다.
공격 당하지 않는다면, HoneyPot 의 가치가 없다. 따라서 침입자를 유인할 수 있는 유인 정보를 많이 가지고 있고, 외부에서 보기에 시스템의 보안이 상당히 취약하다는 판단의 근거를 제공해야 한다. 취약하다고 알려져 있는 서비스 포트를 개방하고 그 포트에 대한 접근을 최대한 허용함으로써 자연스럽게 시스템 내부로 유인한다. 본 논문에서는 침입자 유인을 위한 가상 서비스 구현 방법을 제공한다.
2) 시스템 내에서 가능한 한 오래 머물게 해야 한다.
침입자가 시스템에 오래 머물게 되면, 그만큼 많은 정보를 얻을 수 있게 된다. 유인 정보를 제공하되 HoneyPot 시스템이라는 것을 침입자가 모르게 해야 한다. 이를 위해 본 논문에서는 가상 서비스의 서비스 처리 및 응답 머신의 구현 방법을 제공한다.
3) 로그 및 분석 툴을 제공해야 한다.
침입자가 HoneyPot 에 접근을 한 시점부터 시스템의 사용을 종료한 시점까지 행한 모든 행동이 로그로 기록되어야 한다.

HoneyPot 은 크게 "Research Honeypot"과 "Production HoneyPot"으로 나누는데 "Research HoneyPot"은 침입자에 대한 정보를 획득하고자 설계된 HoneyPot 을 말한다. 이 HoneyPot 은 보안에 직접적인 가치를 더해 주지는 않지만 침입자로부터의 위협을 연구하기 위해 그리고 이러한 위협에 대해 어떻게 더 나은 방어를 해야 할 지를 연구하기 위해 사용된다. HoneyPot 은 침입자로부터의 위협에 대해 연구할 수 있는 플랫폼을 제공한다. 침입자에 대해 더 잘 배울 수 있는 방법은 그들의 행위단위로 관찰하는 것으로서 시스템 공격 및 손상 과정을 단계별로 기록해 두는 것이다. "Production HoenyPot"의 목적은 위협을 완화시켜 주는 데 있다. HoneyPot 은 보호, 탐지, 대응을 통해 보안에 대한 부가적인 기능을 제공해 준다.

본 논문에서 구현하고자 하는 HoneyPot 은 이 둘의 기능을 혼합한 형태의 HoneyPot 으로서 가상 서비스 에뮬레이션을 통해 가상 호스트를 구현하고 공격 패턴에 대한 룰을 기반으로 공격의 시점 및 종류 등을 지능적으로 판단한다. 또한 실시간으로 수집되는 패킷의 내용 및 특정 필드를 기록하고 정의된 대응 정책에 의해 수동적 또는 능동적으로 대응을 한다. 이 둘을 혼합함으로써 침입 방법에 대한 연구가치를 제공하고, 그 자체로서도 하나의 보안 기능을 제공하게 되는 것이다.

2. 시스템 기능 블록

본 시스템은 크게 5 개의 기능 엔티티로 구성된다. 서비스 관리자, 응답 관리자, 룰 관리자 등을 통해 외부의 침입 방법에 대해 배울 수 있으며, 패킷 분석 엔티티와 대응 엔티티를 통해 주어진 정책에 따라 대응을 할 수 있다. 대응 및 분석은 룰을 기반으로 행해진다.
전체적인 엔티티 관계도를 보면 다음과 같다.



<그림 1> HoneyPot System 구성도

각 엔티티 별로 기능을 살펴 보면 다음과 같다.

1) 서비스 관리자: 가상 서비스를 에뮬레이션 하는 엔티티로서 기존의 표준 서비스 포트를 대체하고, 외부로부터의 서비스 요청을 받아 서비스 별로 구분되는 내부의 각 처리 모듈로 전달하고, 처리 모듈에 의해 생성된 결과를 외부의 클라이언트로 전달하는 기능을 가진다. 또한 서비스 감시자의 서브 모듈로서 실시간 패킷 모니터링 모듈이 존재하고, 이 모듈을 통해 저 수준의 패킷을 실시간으로 수집할 수 있다. 서비스 감시자가 구동 및 관리하는 서비스는 "서비스 필터"를 통해 결정된다. 외부에서의 서비스 요청이 "서비스 감시자"를 통해 서비스 응답 모듈로 전송되면, 현재 지정된 룰을 검색하고, 명령의 종류를 판별한 뒤, 상태 머신을 기반으로 응답 결과를 생성해서 전달해 준다. 응답 처리절차와 결과는 지정된 룰을 따르므로, 만일 새로운 명령을 처리하거나 감시할 필요가 있는 경우 기존 룰에 새로 추가할 룰을 기술하면 된다.

2) 응답 관리자: 서비스 응답 룰의 서브 룰인 응답 패턴을 생성, 저장 및 검색할 수 있도록 하는 기능을 제공한다. 현재는 일반 텍스트 기반의 응답 룰을 지정하도록 구현되어 있으며, 향후 명령 상호간의 관계성을 부여해서 명령의 결과에 대한 영향이 다른 명령에도 파급되어 명령 간에 동일 정보에 대한 일치성을 유지할 수 있도록 정형화되어야 한다.

3) 룰 관리자: 실시간 패킷 모니터링 기능에 의해 수집된 패킷이 패킷 분석기로 보내지고, 패킷 분석기는 수신된 패킷의 관심 필드 및 공격 패턴을 검출해서 적절할 대응을 취하게 된다. 이때, 룰 관리자를 통해 패킷 분석기가 참조하는 정보를 룰 형태로 생성, 저장, 삭제할 수 있다.

4) 대응 관리자: 패킷 분석기와 탐지기에 의해 침입이 탐지된 후 취해질 대응 방법에 대한 정책을 관리한다. 대응 방법에는 '수동적 대응'과 '적극적 대응'이 있다. 수동적 대응은 침입이 탐지된 후 침입에 대해 어떤 행위를 즉각 취하지 않고 침입에 대한 정보를 로그에 저장하거나 침입 사실을 관리자에게 알려주게 된다. 적극적인 대응은 침입이 발견되었을 때 즉각 대응에 나서는 방법으로 침입에 대한 대응 룰을 기반으로 동작한다.

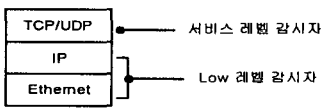
2.1 서비스 관리자

서비스 관리자의 구성 엔티티는 다음과 같다.

1) 서비스 필터: 현재 시스템에서 구성할 가상 서비스에 대한 설정을 한다. 등록할 정보는 "서비스 번호"," 서비스 이름","사용 권한","가상 서비스 룰"등으로 구성된다. "가상 서비스 룰"은 사용자와 대화 형식으로 서비스를 제공하기 위한 룰을 정의한 파일로서 가상 서비스의 동작 및 반응 절차를 기술한다.

2) 서비스 에뮬레이터: 서비스 감시자를 통해 전달된 패킷을 처리하고 응답 엔티티를 통해 룰 기반의 응답을 전송하는 기능을 갖는다.

3) 서비스 감시자: 외부의 접속 요청, 해제 및 명령에 대한 송수신 등에 관한 패킷을 수신하고 서비스를 시작하도록 하는 기능을 한다. 서비스 감시자는 패킷을 수집하는 계층별로 두 가지가 존재한다.



<그림 2> 레벨 감시자

Low 레벨 감시자는 실시간으로 도착하는 이더넷과 IP 계층의 패킷 정보를 수집해서 rule 에 맞게 처리한 후 Log 하는 기능을 가진다. 실시간 모니터링 기능은 많은 양의 패킷을 수신하고 처리하므로, 일부 패킷이 drop 될 수도 있다. 그러므로 rule 을 정확히 기술하고 최적화함으로써 Log 에 저장하는 패킷의 양을 줄여야 한다. 서비스 레벨 감시자는 가상 서비스 수준에서 발생하는 명령어 및 응답에 대한 패킷을 감시한다. 또한 서비스에 대한 지정된 rule 과 일치하는 공격 패턴을 추적해서 Log 에 기록한다.

4) 응답 엔티티: 미리 정해진 룰에 따라 침입자의 요청에 능동적으로 응답을 한다. 중요한 것은 응답 엔티티를 잘 구성해야 실제 시스템과의 유사성이 향상된다. 본 구현에서는 이러한 응답을 처리하기 위한 룰기반의 상태 머신을 사용한다. 명령 처리 및 응답이 룰을 기반으로 동작하므로, 새로운 룰을 정의함으로써 새로운 서비스를 쉽게 에뮬레이션 할 수 있다.

2.2 응답 관리자

서비스 응답 룰의 서브 룰인 응답 패턴을 생성, 저장 및 검색할 수 있도록 관련 GUI 와 콘솔 화면을 제공한다. 현재는 일반 텍스트 기반의 응답 룰을 지정하도록 구현되어 있으며, 향후 명령 상호간의 관계성을 부여해서 명령의 결과에 대한 영향이 다른 명령에도 파급되어 명령간에 동일 정보에 대한 일치성을 유지할 수 있도록 정형화되어야 한다.

2.3 룰 관리자

서비스 응답 엔티티의 서브 모듈인 실시간 패킷 모니터링 기능에 의해 수집된 패킷이 패킷 분석기로 보내지고, 패킷 분석기는 수집된 패킷의 관심 필드 및 공격 패턴을 검출해서 적절한 대응 동작을 취하게 되는데 이때 관리자를 통해 패킷 분석기가 참조하는 정보를 룰 형태로 생성, 저장 및 삭제할 수 있다.
룰은 다음과 같은 구문 형식을 가진다.

```
Rule_name :: = [A..Z | 0..9]*
Protocol_Type = ['tcp'|'udp'|'icmp'|'arp']
Response_Action = {'alert' | 'log' }
Port_Number :: = [1..9][0..9]*
Network ::= [IP addr | MACROS]
Filter ::= ['flags' | 'content' ]
DetectLevel = [Warning(0)|Minor(1)|Major(2)| Critical(3)]
Define {
        Source = { Network Port_Number };
        Dest   = {Network Port_Number };
        Action = Response_Action;
        Protocol = Protocol_Type;
        Condition = {Filter : value}*
        Level = Detect_Level;
} ::= Rule_name(Rule_Descriptor);
```

'root' 권한으로 Login 하는 경우에 대한 룰을 간단히 기술하면 다음과 같다.

```
Define {
        Source = {0.0.0.0 any}
        Dest = {192.168.0.1 23};
        Action = alert;
        Protocol_Type = tcp;
        Condition = {content : 'login', content:'root'}
        Level = Critical;
} ::=RootLogin("Telnet Root Loggin Detection");
```

2.4 대응 관리자

Detector 가 룰을 기반으로 침입을 탐지한 후 이에 대한 대응을 시작하는 데, 여기에는 대응 관리자에 지정된 정책에 의존하게 된다. 대응 관리자는 BP(Blocking Profile)이라고 부르는 정책 지식 베이스를 유지하고 있다. Detector 룰 통해 침입이 발견되면 대응 엔티티로 보내지게 되며, 대응 엔티티는 설정된 정책에 따라 각각의 BP 를 선택하고 그에 따르는 처리를 하게 된다. 예를 들어 파일 접근에 대한 BP 룰 구성하면 다음과 같다.

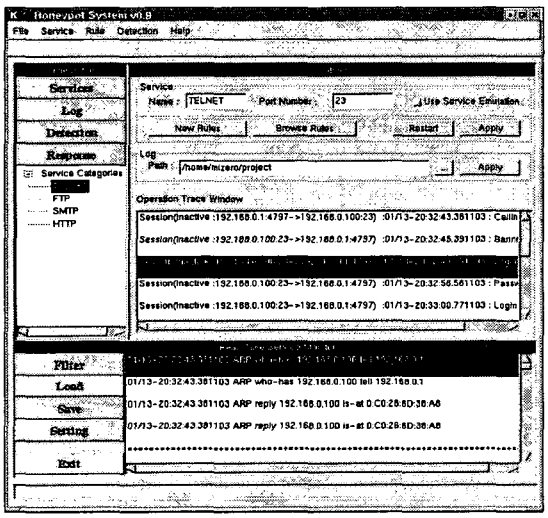
Level0(Warning) = read/write/execution 가능

Level1(Minor) = read/write 가능
Level2(Major) = read 가능
Level3(Critical) = 모두 불가능

각 Level 은 룰 관리자에서 정의한 Level 과 일치해야 한다.

2.4 실시간 서비스 모니터

침입자의 행위 및 수집되는 패킷을 화면에 표시해 준다. 서비스 감사자의 서브 모듈인 "실시간 패킷 모니터링"을 통해 외부로부터 유입되는 패킷이 실시간으로 추출되며 이 패킷은 이후 공격 패턴 분석 및 대응을 위해 참조 된다.



<그림 3> 실시간 서비스 모니터

3. 결론 및 향후 연구과제

본 논문에서 구현한 HoneyPot 시스템은 'Research'와 'Production'의 혼합 형태를 가지는 것으로서, 새로운 침입 기술에 대한 지식의 축적 및 학습이 가능하고, 대응 엔티티를 통한 정책을 통해 실시간 대응이 가능하다. 본 구현에서는 가상 서비스를 에뮬레이션 함으로써 외부로부터의 침입을 유도하였고, 가상 서비스의 완성도를 높이기 위해 룰 기반의 서비스 응답 상태 머신을 구현하였다. 가상 서비스를 통해 접근한 침입자의 행위를 단계별로 기록하고 미리 정해진 룰을 통해 능동적인 대응이 가능하도록 구현하였다. 탐지 및 대응 과정이 모두 룰을 기반으로 동작하며, 쉽게 수정 및 변경될 수 있으므로, 유지 보수 및 업그레이드가 편리하다. 그러나 가상 시스템의 완성도를 높이기 위해 보다 많은 서비스가 에뮬레이션 되어야 하고 각 에뮬레이션 서비스들은 실제의 서비스가 제공하는 대부분의 기능을 제공하도록 구현되어야 한다. 또 다른 접근 방법으로는 Vmware 와 같은 가상 호스트 구축 소프트웨어 등을 이용해서 시스템을 구축하거나 아니면 리눅스와 같이 소스가 공개된 운영체제의 inet 데몬이나, shell, 각 서비스 데몬의 수정을 통해 HoneyPot 시스템의 목적에 맞도록 커널 및 서비스를 재구성하는 것이다. 또한 근래에 들어서 공격의 유형이 호스트를 목표으로 하기보다, 네트워크를 목표으로 진행되고 있으므로, 본 시스템을 확장해서 Firewall 이나 Router 등의 시스템과 함께 연동을 하도록 함으로써 네트워크 차원의 HoneyPot 시스템을 구축하기 위한 연구가 진행되어야 한다.

참고문헌

[1] Snort Users Manual Snort Release: 1.8.1 Martin Roesch 10th August 2001
[2] http://project.honeynet.org/
[3] http://all.net/ "Deception Toolkit"
[4] 김병구,김동성,정태명, "계층적 구조를 갖는 침입탐지 통합 시스템 설계", 정보처리학회지, vol.6, No.2, Jan.,1999
[5] 정훈조, 김병구, 정태명, "침입의 유형과 탐지 시스템의 분류", 정보처리학회지, vol.6, No.2, Jan.,1999