IDSTa - TCP Stream 분석 기반 침입 탐지 시스템

정해진* 이명선

한국과학기술정보연구원, 슈퍼컴퓨팅센터

email : {hjjung, mslee}@kreonet2.net

IDSTa - Host based IDS through TCP Stream Analysis

Haejin Jung*, Myungsun Lee

*Supercomputing Center, Korea Institution of Science and Technology Informaton(KISTI)

요 약

NIDS 구조의 근본적인 결함을 이용하는 공격기법과 또한 우회하는 공격기법이 많이 개발되고 있다. 이러한 NIDS에 대한 공격기법의 해결로 HIDS가 사용 될 수 있으나 HIDS 또한 서비스 하는 시스템 자체 내에 탑재하기 때문에 시스템에 많은 부하를 준다. 따라서 NIDS의 많은 장점들을 유지하면서도 NIDS의 한계를 극복하고 HIDS로써 시스템에 많은 과부하를 주지 않는 새로운 HIDS 모델을 제시한다. 제안된 HIDS는 특성상 모든 곳에서의 접속을 허용하므로 보안이 취약한 Web 서버의 보안 강화를 목적으로 설계되었다. 또한, Web 서버는 Web Service라는 특정 목적만을 위해 운영되기 때문에 HIDS를 설치하더라도 Web 공격에 대해서만 고려함으로써 HIDS의 부하를 상당히 줄일 수 있다. 본 논문에서 제안하는 HIDS는 Linux 운영체제의 Kernel에서 TCP Stream을 추출하여 이를 감사 자료로써 사용하여 침입탐지를 한다.

I. 서론

인터넷의 증가로 web 서비스의 사용이 급증하고 있다. Web 서비스를 제공하는 Web Server는 그 속성상 모든 곳에서 접속이 가능해야 하기 때문에 일반 서버들에서 쓰이는 보안 툴 예를 들면 TCP Wrapper나 Packet filtering 의 기능을 거의 사용할 수 없다. 이런 특성 때문에 특정 호스트로부터의 접근을 제어할 수 없고, 모든 서비스들을 제공되어야 하기 때문에 서비스 요청에 대한 패킷들을 일일이 거를 수도 없다. 따라서 다른 서버들에 비해 Web Server는 보안이 상당히 취약할 수 밖에 없다. 현재 인터넷과 eBusiness의 급성장에 따라 Web Server에서 보안의 중요성은 더욱 커지고 있다.

따라서 본 논문에서는 Web 서버를 보호하기위한 방법으로 IDS들을 분석하고 보안에 있어 기존 IDS의 한계를 알아보고 이를 대처할 수 있는 방안에 대해 제시했다. 대처 방안으로 기존 IDS의 한계를 극복하면서 Web 서버를 보안 할 수 있는 새로운 IDS 의 모델을 제시한다.

II. 기존 IDS의 한계

현재 Web 서버의 보안은 IDS에 많은 부분을 의존하고 있다. IDS는 감시 대상에 따라 NIDS와 HIDS로 나뉜다. HIDS(Host-based IDS)는 호스트 기반 IDS로 시스템의 내부에 설치되어 시스템 내부 사용자들의 활동을 감시하고 해킹 시도를 탐지해 내는 시스템이다. 반면에 NIDS(Network IDS)는 네트워크의 패킷 캡쳐링에 기반하여 네트워크를 지나다니는 패킷을 분석해서 침입을 탐지해낸다. 이러한 NIDS는 네트워크 단위로 하나만 설치하면 된다. 따라서 NIDS는 하나의 시스템으로 네트워크에 물려있는 모든 시스템에 보안을 제공할 수 있고, 비교적 구현이 쉽다는 장점을 지닌다. 이와 같은 이유로 현재 상용화된 IDS 의 대부분은 NIDS에 해당된다. 현재 웹 서버의 보안도 시스템에 부하를 많이 주는 HIDS보다는 전체적으로 하나의 시스템으로 많은 호스트를 보호할 수 있는 NIDS에 의존하고 있다.[7] 또 HIDS는 모니터링하려는 시스템 마다 하나씩 설치가 되어야하기 때문에 NIDS보다 비용이 많이 든다.

그러나 많은 NIDS가 개발되어 사용되고 있지만 NIDS의 근본적인 한계에서 기인하는 보안의 사각지대가 발생하고 있다. NIDS의 한계는 크게 다음과 같다.

a. 우회공격이 가능

보호하고자 하는 서버와 NIDS의 TCP/IP Stack의 구현이 달라 똑 같은 패킷들을 받는다 할지라도 패킷들을 reassemble 하여 만든 Stream들은 달라질 수 있다. 이를 이용한 Insertion, Evasion Attack으로 인해 False Negative Alarm Rate와 False Positive Alarm Rate가 높고, 공격자의 의도에 의해 급격히 높아 질 수 있다.[3]

b. Fail Open

일반적으로 NIDS가 DoS(Denial of Service) 혹은 DDoS(Distributed Denial of service) 공격에 의해 작동이 멈춘다 하더라도, 실제 서비스를 제공하는 서버들은 서비스를 계속적으로 제공한다. 이러한 NIDS의 특성을 이용하여 먼저 NIDS를 공격하여 NIDS를 다운시킨 후에 실제 서비스를 제공하는 서버를 공격하는 것이 가능하다.

c. 패킷 손실

네트워크 장비의 발전으로 인해 네트워크의 속도는 급속히 증가하고 있으나 NIDS의 처리율은 이에 미치지 못하고 있다. 이로 인해 NIDS의 패킷 손실률은 계속적으로 증가하고 있으며, 큰 문제로 떠오르고 있다. 실제로 한 두 패킷만 놓친다 하더라도 해킹을 전혀 탐지할 수 없기 때문에 패킷 손실은 심각한 문제일 수 밖에 없다.

NIDS는 이와 같은 한계로 인해 False Negative 와 False Positive Alarm이 많기 때문에 적극적인 Response를 하기 어렵다.[3] 또한 정확한 침입 탐지를 하기 위해서는 해당 Protocol에 대한 정확한 정보가 필요하다. 하지만 대부분의 IDS들은 범용 IDS로써 특정 프로토콜에 기반한 정밀 검사가 불가능하며, 각 Segment와 공격 Signature의 Pattern Matching 기법만 적용한다는 문제점을 지닌다.

III. 웹 서버의 HIDS

1. 관련 연구

NIDS의 근본적인 한계에 기인하는 보안의 사각지대를 극복할 수 있는 방안을 연구한 논문들을 참고했다.

a. Packet Filtering

패킷 필터는 지나가는 패킷의 헤더를 살펴보고 그 전체 패킷을 시스템으로 보낼 것인지 버릴 것인지를 결정하는 소프트웨어이다. 하지만 패킷의 헤더부분만을 점검하기 때문에 TCP이상의 프로토콜에서 침입이 있을 경우에는 탐지 할 수 없다. [5]

b. Socket Filtering

Socket Filter는 사용자 프로그램의 소켓에 filter를 붙여 소켓을 통해 교환되는 데이터를 제한하기 위해 사용된다. 하지만, TCP 레벨을 지원하지 않으며, 사용자 프로그램에 직접 코드를 첨가하여야 하기 때문에 IDS와 같은 관리 프로그램에서 사용자 프로그램의 이런 기능을 사용할 방법이 없다.[5]

c. Transport and Application Protocol Scrubbing

TCP/IP Stack의 차이로 인해 NIDS와 End System이 똑 같은 패킷들을 다른 Stream으로 조합할 수 있는 문제를 해결하기 위한 논문이다. Firewall의 위치에 Scrubber를 두고, Scrubber 가 모든 시스템들이 같은 Stream으로 조합할 수 있는 형태로 바꾸어 패킷을 보내준다. 이 방식은 Scrubber가 Bottle Neck이 될 수 있기 때문에 Performance가 좋지 않고, Scrubber뒤의 모든 시스템들이 같은 Stream으로 조합하도록 할 수 있는 rule의 개발이 어렵고, 새로운 기법이나 시스템들의 TCP/IP Stack이 변하게 되면 rule 전체가 수정되어야 한다. [1]

d. Identification of Host Audit Data to Detect Attacks on Low-level IP Vulnerabilities

네트워크를 통해 교환되는 패킷들을 분석하여 Low-level 네트워크 공격이 일어나는지를 감시하는 HIDS에 관한 연구이다.[2]

2. 웹 서버의 HIDS 설계

NIDS의 한계를 극복하면서 Web Server에 대해 일어나는 공격에 대해 최적의 침입 탐지를 할 수 있는 해결책은 호스트기반 IDS 이다. HIDS는 일반적으로 보호하고자 하는 호스트 내부에서 감사 자료를 수집해서 침입을 탐지한다. 따라서 NIDS가 모니터링 하지 못하는 호스트에서 일어나는 모든 상황을 모니터링할 수 있다. 또한 Switched Network에서도 사용이 가능하고 호스트 내부에서 동작을 하기 때문에 네트워크가 암호화 되어있어도 영향을 받지 않는다.

일반적으로 기존 HIDS는 감사 자료를 내부에서 얻기 때문에 감사 자료로는 시스템이 남기는 로그들과 각 Process에서 일어나는 System Call 만 보고 침입을 탐지하고자 한다. 따라서 네트워크 상에서 Remote로 일어나는 공격에 대해서는 탐지하기가 쉽지 않다.

본 논문에서는 리눅스 Kernel에서 TCP Stream을 확보하여 침입이 일어나고 있는지를 판단하는 HIDS를 제시했다.

구현은 크게 세 가지 부분으로 나누어 진다. 첫째는 커널부터 Stream 정보를 받아오는 부분이고, 둘째는 받은 Stream 정보를 이용하여 침입이 일어나는지 침입탐지를 하는 부분이고, 셋째는 침입탐지를 통해 침입을 발견했을 때, 이에 대한 Response를 책임지는 부분이 필요하다.

1) 감사 자료 확보

리눅스 커널에서 TCP/IP의 Stream(Reassemble된 후를 말함)를 얻을 수 있는 방법은 첫 번째, 리눅스 커널에서의 통신 프로토콜 구조에서 TCP stream을 취급하는 Layer를 수정해서 얻을 수 있다. 이 방법은 커널에서 스트림을 가져올수 있는 부분이 BSD Socket 층과 INET Socket 층인데 이중 프로토콜 패밀리를 선택할 수 있는 BSD 소켓 층을 수정해서 얻어 올 수 있다. 그러나 커널을 수정하여 얻는 방법들은 시스템 호출만해도 bug가 발생할 확률이 대단히 높고, 시스템 호출의 일부분만 수정해도 커널 전체를 다시 컴파일해야 하기 때문에 개발 효율도 떨어지는 방법이다. 또 Proc filesystem을 이용하는 방법 역시 속도면에서는 좋은 편이나, 커널의 많은 부분을 직접 수정하여야 하기 때문에 개발이 용이하지만은 않다. 두번째 방법은 커널에서 시스템 호출로 인한 패킷 캡쳐링을 하는 라이브러리를 사용하여 IP Layer에서 감사 자료를 가지고 올수 있다. 커널의 라이브러리를 통한 TCP 패킷을 캡쳐링하는 옵션이나 명령을 이용하여 코딩한 다음 UDP와 TCP로 구분하여 TCP 정보만 받아온다. TCP 정보를 받아온 후에는 80 port에 해당하는 것만 받고 나머지는 방화벽의 규칙으로 넘긴다. 여기서 주의해야할 사항은 IP 층으로부터 정보를 받오기 때문에 보안상 커널에 방화벽을 올려줘야 우리가 검사하고자 하는 정보외의 공격성 정보들에대해 방어가 가능하고 후에 대응하는 방법에서도 용이하게 설계할 수 있다. 받아온 80 포트 정보는 HTTP 프로토콜을 적용시켜서 정리해서 DB에 저장된 rule 셋에 맞추어 검사하게 된다.

2) 침입 탐지 기법

일반적으로 Stream에서 침입 탐지를 위해 사용되는 기법은 Patten Search와 Protocol analysis 방법이다.

Pattern Search기법은 단순히 독특한 Pattern을 Network Traffic에서 찾아서 기록하는 것 이다. 일종의 Virus scanning 기술을 Network traffic에 적용한 것이라고 생각하면 된다. Pattern searching system의 가장 좋은 예는 open-source로 개발되고 있는 Snort이다.

[표 1] PHF 공격에 대한 Snort Rule의 예

```
Alert tcp $EXTERNAL_NET any ->
$HTTP_SERVERS 80 (msg:"WEB-CGI phf
access";flags: A+; uricontent:"/phf"; nocase;
reference:bugtraq,629;
reference:arachnids,128;reference:cve,CVE-1999-00
67; classtype:attempted-recon; sid:886; rev:3
```

예를 들면, [표 1]의 PHF 공격에 대한 Snort Rule은 URL의 "/phf"에 의해 만들어지는 HTTP 요구가 일어날 때마다 Snort를 동작하게한다. 만약 hacker 가 PHF를 실행시키는 특별한 URL을 써서 시스템 안에 침입을 해온다면 패턴 서칭 시스템은 '/phf'대한 들어오는 모든 URL을 검사함으로써 침입자를 찾아낼 수 있다. 그러나 이 기법은 최근에 우회공격등에 의해 무력화될 수 있다. 또한 '/phf'패킷은 공격에 전혀 관련되지 않는 곳에서도 일어날 수 있다. 따라서 NIDS의 false positive 가 발생될 수 있다.[4]

Protocol analysis는 pattern - search NIDS의 HTTP signature 보다 더 많은 정보와 header의 각각의 field를 나누어서 그 의미를 두어 분석한다. 따라서 evasion에 대해 좀 더 꼼꼼하게 분석할 수 있다.

[표 2] 전형적인 http 요청 예 [4]

```
GET / index.html HTTP/1.0
Host: www.robertgraham.com
Referer : http://www.robertgram.com/cgi-bin/phf
User-Agent : Mozilla/2.0
```

[표 3] Pattern Search 예[4]

GET / index.html HTTP/1.0Host: www.robertgraham.com Referer : http://www.robertgram.com/cgi-bin/phfUser-Agent : Mozilla/2.0

[표 4] Protocol Analysis 예 [4]

Method = GET URL = / index.html Version = HTTP/1.0 Fieldname = Host HTTP_Host: www.robertgraham.com Fieldname = Referer HTTP_Referer : http://www.robertgram.com/cgi-bin/phf Fieldname = User-Agent HTTP_Agent : Mozilla/2.0

[표 2]과 같은 전형적인 HTTP 요청이 올 경우 Pattern Search 기법을 이용하는 IDS 시스템은 [표 3]와 같은 긴 문자열에서 공격 패턴과 일치하는 문자열이 존재하는지를 검색한다. 이는 실제 공격을 수행하기 위해 이용될 가능성이 전혀 없는 것들에 대해서까지 모두 검색하기 때문에 실제로는 공격이 일어나지 않는 부분인 HTTP_REFERER 부분에서 나오는 '/phf' 문자열로 인해 False Positive alarm를 일으키게 된다. 이에 반해 Protocol Analysis 기법은 [표 4]에서처럼 HTTP의 프로토콜에 따라 TCP Stream을 분석하고, 공격에 이용될 가능성이 전혀 없는 부분에 대해서는 공격 패턴이 존재하는지 찾아보지 않고, 공격이 일어날 수 있는 부분(Method 와 URL 부분)만 검색한다. 따라서, 예제 패킷에 대해서False Positive Alarm을 내지 않고 정상적인 HTTP 요청으로 간주한다.[4]

결과적으로 80포트에 대한 패킷을 HTTP 프로토콜을 적용해서 응용 층까지 패킷을 가져와서 HTTP 프로토콜이 아닌 것은 방화벽의 rule set으로 넘기고 나머지는 각 세그먼트 단위를 스페이스로 구분하여 긴 문자열이 되지 않게 하여 구현이 쉬운 pattern search 기법을 통해서 침입을 탐지하는 솔루션을 사용한다.

3) Alerting

본 논문에서는 IP층으로부터 추출된 Stream을 감사 자료로 이용하고 향상된 패턴 매칭 기법으로 침입탐지를 하고 침입탐지에서 침입으로 판단될 경우는 커널에 방화벽을 올린 상태이기 때문에 방화벽과 연동하여 방화벽 차원에서 rejection을 한다. 이 방식에 대해서는 어느 시점에서 rejection을 하는 것이 가장 적절할지를 고려하고 있다.

다음은 IDSTa와 기존 IDS들과 의 차이점을 비교해보았다.

[표 5] IDSTa와 기존 IDS들과의 차이

	NIDS	HIDS	IDSTa
Audit Data	Network Packet	system Log	TCP Stream
Insertion, Evasion Attack	Possible	Impossible	Impossible
System Load	None	High	Medium
Packet Loss	Possible	Impossible	Impossible
Fail-open	Yes	No	No

제안된 솔루션 IDSTa는 Host 기반 IDS이므로 HIDS들이 일반적으로 가지는 큰 단점인 시스템의 로드를 줄여야 하는 문제점을 안고 있다. 그러나 TIDS는 논문 제목에서도 알 수 있듯이 웹 서버만을 위한 IDS이다. 따라서 IDSTa의 룰셋과 침입 탐지는 웹 요청 트래픽과 플로우에 대해서만 탐지하기 때문에 시스템에 주는 로드가 적을 것이며 감사 자료가 시스템 로그가 아니기 때문에 자연적으로 로드는 줄어들 수밖에 없다.

IV. Conclusion

새롭게 제안하는 Stream Analysis를 통한 Intrusion Detection은 다음과 같은 것들을 해결할 수 있다.

a. 위에서 언급한 NIDS의 한계를 극복할 수 있는 새로운 Intrusion Detection기법을 제시한다.

b. 기존의 범용 IDS의 한계를 극복할 수 있는 Web Server에 특화된 Intrusion Detection 기법을 제시한다.

c. False Positive와 False Negative를 현저히 줄임으로써 Active Response가 가능한 IDS를 제시한다.

기존의 범용 IDS는 원래의 취지가 Intrusion이 일어나는지 Detection을 하는 시스템이다. 그러나 침입에 대한 탐지를 이미 시스템이 해킹을 당하고 난 뒤에 침입을 탐지했다고 좋은 IDS는 아니다. 앞으로는 IDS가 침입에 대해 어느 정도의 예방을을 가지고 있고 또한 침입이 일어났을 때 얼마나 신속하게 침입을 오탐 없이 판단하고 침입에 어떻게 대응할 수 있는지가 중요하다.

따라서 앞으로는 IDS가 각종 공격에 대한 통계로 예방, 차후 대응 등을 할 수 있는 종합적인 보안 툴이 되어야 한다고 본다. 이미 그러한 Management로써의 IDS가 개발되고 있는 실정이다.[7]

본 논문에서는 새로운 HIDS를 개발하기위한 간단한 이론과 설계를 제시했다. 앞으로는 좀더 많은 IDS를 분석하여 Active한 대응과 오탐이 없는 정확한 탐지 기법을 쓸 수 있는 HIDS로 완벽하게 구현하고자 한다.

참고문헌

[1] *G. Robert Malan, David Watson and Farnam Jahanian "Transport and Application Protocol Scrubbing",IEEE INFOCOM 2000, March 26-31, Tel Aviv, Israel*

[2] *Thomas Daniels, Eugene Spafford "Identification of Host Audit Data to Detect Attacks on Low-level IP Vulnerabilities", July, 1998, Journal of Computer Security*

[3] *Thomas H. Ptacek , Timothy N. Newsh , "Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection", January, 1998*

[4] *Robert Graham, "NIDS-Pattern Serch vs. Protocol Decode", September 2001 Journal of Computer Science*

[5] *Julia Allen, Alan Christie, William Fithen, John McHugh, Jed Pickel, Ed Ston, "State of the Practice of Intrusion Detection Technologies", January 200, Networked Systems Survivability Progr*

[6] *Steven J. Scott, "Threat Management Systems The State of Intrusion Detection" August 9, 2002*

[7] *Rebecca Bace, Peter Mell " NIST Special public On Intrusion Detection System"*

[8] *Marcus J. Ranum " Experiences Benchmarking Intrusion Detection Systems" NFR Security Technical Publications, Decenber, 2001*