웹 로그 마이닝을 통한 실시간 웹 서버 침입 탐지

진홍태*, 박종서*
한국항공대학교 컴퓨터공학과
e-mail: {jhongtae, jspark}@mail.hangkong.ac.kr

Real-time Web-Server Intrusion Detection Using Web-Log Mining

Hong Tae Jin*, Jong Sou Park*
Dept. of Computer Engineering, Hankuk Aviation Univ.

요 약

인터넷 사용이 보편화됨에 따라 기존의 방화벽만으로는 탐지가 불가능한 웹 서비스의 취약점을 이용한 공격이 증가하고 있다. 그 중에서도 특히 웹 어플리케이션의 프로그래밍 오류를 이용한 침입이 공격 수단의 대부분을 차지하고 있다. 본 논문에서는 웹 어플리케이션의 동작을 분석한 후 취약점 발생 부분에 대해 웹 로그 마이닝 기법을 사용하여 실시간으로 로그를 분석함으로서 공격 패턴을 비교·분석한다. 또한 프로세스 분석기를 통한 결정(decision) 과정을 통해 침입으로 판단되면 해당 접속 프로세스(pid)를 제거한 후 공격 아이피를 차단함으로서 침입을 탐지하는 메커니즘을 제시한다.

Ⅰ. 서론

인터넷이 급속히 확산되고 보편화됨에 따라 인터넷을 이용하는 공격의 유형도 변화하고 있다. 전통적인 공격(traditional attack) 방식은 주로 OS와 네트워크 서비스에 존재하는 취약점(vulnerability)을 목표로 했고 또한 버퍼 오버플로우와 같은 공격을 위해서는 어셈블러로 직접 공격 프로그램을 작성해야 했었다. 그러나 방화벽(firewall)의 도입이 보편화됨에 따라 이런 방식의 공격이 거의 불가능해 졌고 또한 OS 및 네트워크 서비스 개발업체들의 꾸준한 패치로 인해 전통적인 공격 방식으로는 침입이 거의 불가능하게 되었다.

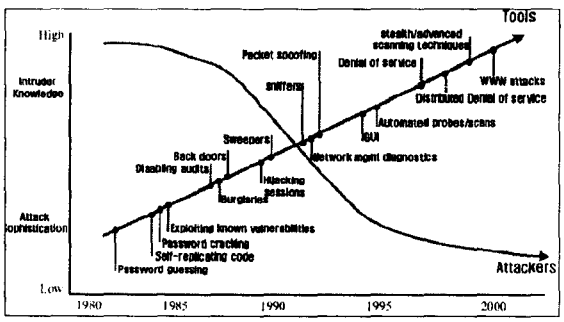

그림 1: 최근 해킹 동향

그렇지만 80번 포트를 통해 침입하는 웹 공격의 경우에는 웹 서비스를 위해 공식적으로 허가된 합법적인 트래픽이므로 기존의 방화벽으로는 탐지할 수 없다. 또한 침입탐지시스템(IDS) 우회 기법을 통한 웹 서비스 공격이 고도화, 다양화, 대중화되어 감에 따라 IDS만으로는 그러한 웹 서버 공격 탐지가 어려운 것이 사실이다. 이 같은 이유로 오늘날의 공격 방식은 그림 1에서 알 수 있듯이 주로 80번 포트를 이용하여 웹 어플리케이션 내부로의 침입을 시도하는 웹 해킹으로 그 형태가 변화되고 있다[1,2].

Ⅱ. 웹 어플리케이션의 취약점 분석

1. 웹 어플리케이션 동작 및 취약점들

전자상거래와 같은 일반적인 웹 어플리케이션 동작 과정을 살펴보면 다음과 같다. 일반적으로 클라이언트에서 웹 서버로 요청(request)을 보내게 되면 웹 서버는 요청받은 쿼리를 웹 어플리케이션(C/C++, Perl, JSP, ASP, etc...)을 통해 해석하게 되고 DB서버로의 커넥션을 시도하여 그 결과를 클라이언트로 응답(reply)하게 된다. 그러나 그 과정에서 다음과 같은 몇 가지 취약점들이 발생할 수 있다.

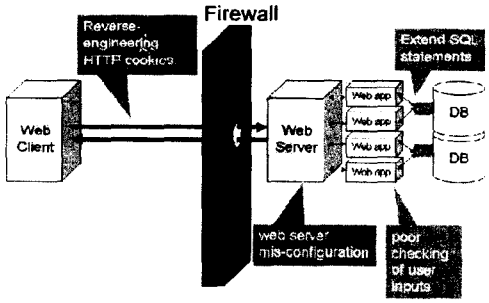

그림 2: 웹 어플리케이션 취약 포인트

현재 웹상에서 발생할 수 있는 취약점들을 살펴보면 다음과 같이 크게 4가지로 나눌 수 있다[3,4,5].

· **웹 서버 설정상의 오류**: 웹 서버 설정을 올바르게 하지 않음으로서 클라이언트 스크립트 명령을 수행할 수 있고 또한 쿠키 값 설정 실수를 통해서 쿠키 정보를 빼낼 수도 있다. 그리고 마이크로소프트 IIS서버의 경우에는 URL 인코딩 버그와 같은 취약점이 발생할 수 있다.

· **웹 어플리케이션에서의 프로그래밍 오류**: 클라이언트로부터 오는 값에 대한 검증 부분을 고려하지 않았기 때문에 발생할 수 있는 취약성이다. 이것은 현재 발생하고 있는 웹 해킹 공격수단의 대부분을 차지하고 있고 또한 웹 어플리케이션의 대부분이 데이터베이스와 연계하여 구동되고 있기 때문에 그 피해가 심각하다 할 수 있다.

· **SQL 쿼리문 삽입(SQL Injection)**: URL에 대한 입력값(', ", or, etc...) 검증 모듈을 고려하지 않아 발생할 수 있는 취약점이다.

· **Reverse-engineering HTTP cookies**: Sniffing과 Spoofing을 통해 패킷 정보를 수집하거나 변조하는 것을 예로 들수 있다. 이 방법은 실제로 웹 서버에 대한 공격을 하는 것이 아니라 단지 클라이언트와 서버간의 자료(password, cookie, etc...)수집을 통해서 원하는 정보 획득이 목표이다.

이러한 웹 상에서 발생하는 여러 취약점들을 바탕으로 기존의 방화벽이나 IDS에 의존하던 방식에서 벗어나 다음 장에서는 웹 서버 전용의 실시간 Agent를 도입한 효과적인 침입 탐지 방안을 제안하고자 한다.

2. 제안 모델

2.1 연구 배경

기존의 자동화 툴을 통한 소스코드 오류 검출 방법이나 패치를 통해서 알려진 프로그래밍 오류를 어느 정도 보완할 수는 있지만 근본적으로 프로그램 개발자들이 설계 단계에서부터 보안에 대한 인식이 없이 기존의 개발 툴이나 습관에 의존하여 웹 어플리케이션을 제작하기 때문에 취약점이 존재하지 않는 완벽한 웹 어플리케이션을 기대하기는 어려운 현실이다[6]. 일단 공격자가 프로그래밍상의 취약점이 존재하는 웹 서버에 침입했을 경우 시스템 명령어를 통한 대상 서버의 정보수집, 파일 전송 프로그램(wget, tftp)을 이용한 악의적인 코드전송, 리버스 텔넷을 위한 Netcat 프로그램을 사용하는 등의 대략적인 행동 패턴들을 취하게 된다. 이러한 행동 패턴들을 바탕으로 본 논문에서는 실시간으로 웹 로그 마이닝 방법을 통해 기존의 분석된 공격 패턴들과 비교하고, 로그와 프로세스를 분석하는 각각의 Agent들을 사용하여 결정(decision) 과정을 거친 후 침입 여부를 판단해 본다. 그리고 만일 침입으로 판단이 되면 공격자와 연결을 담당하고 있는 접속 프로세스(pid)를 찾아내어 프로세스 kill을 통해 그 프로세스를 강제로 종료하고 공격자의 재접속 시도를 방지하기 위해 해당 아이피를 실시간으로 차단하는 침입 탐지 방식을 제안한다[7]. 아파치 웹 서버의 경우를 살펴보면 80번 포트로의 연결을 담당하는 프로세스(httpd)가 있고 클라이언트와의 접속이 이루어지면 그 클라이언트와의 접속을 유지하기 위한 새로운 프로세스를 생성하게 된다. 그렇지만 연결이 이루어진 후 클라이언트 측에서 웹 서버로 요청을 보내지 않는다면(대략 5~10초 사이) 그 접속 프로세스(ESTABLISHED)는 자동으로 종료되고 또 다른 pid를 갖는 접속 프로세스가 생성되게 된다. 이 경우는 공격자가 어떠한 악의적인 코드를 실행하거나 행동을 하지 않는 것으로 볼 수 있으므로 본 논문에서 이 경우는 고려하지 않았다.

2.2 제안 모델의 구조

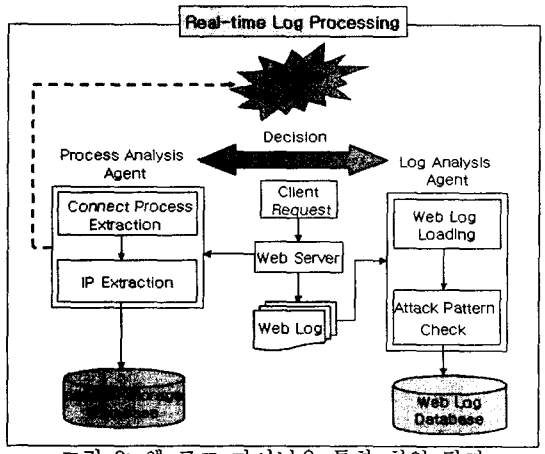

그림 3: 웹 로그 마이닝을 통한 침입 탐지

그림 3에서는 웹 서버의 접속 프로세스를 관리하는 Agent와 로그를 분석하는 Agent를 통해 침입 탐지가 어떻게 이루어지는지에 대한 전체적인 흐름을 보여 주고 있다. 각각의 역할을 살펴보면 다음과 같다.

· **Process Analysis Agent**: 클라이언트로부터 접속 요청이 발생할 때마다 실시간으로 모든 접속 프로세스(pid)와 IP를 검출한다. Log Analysis Agent를 통해 공격 패턴과 일치하는 결과를 얻게 되면 즉시 해당 접속 프로세스를 제거하여 공격자와의 접속을 해제한 후 재접속 방지를 위해 IP를 차단한다.

· **Log Analysis Agent**: 웹 서버 로그를 실시간으로 분류(classification)하고 정렬(order)한다. 침입 패턴들을 비교하기 위해 각각의 URL에서 특정 문자열를 검색하는 Token Finder와 같은 방법들을 사용한다. 침입 패턴을 추출하게 되면 Process Analysis Agent로 그 결과를 전송한다.

2.3 제안 모델의 장·단점

본 논문에서 제안한 Agent 기반 모델은 침입이 이루어진 후에 로그 분석을 통해 실시간으로 판단하고 대처

하는 방식이므로 공격자의 패턴 유형만 제대로 분류되고 클러스터링(clustering)되어진다면 즉각적인 의사결정을 통한 해당 프로세스 제거가 가능하기 때문에 침입 탐지율이 높은 웹 서버 보안 구현이 가능하다. 또한 웹 서버로 접속하는 모든 접속 프로세스를 실시간으로 모니터링하고 있기 때문에 프로세스 분석을 통해 DOS와 같이 웹 서버의 자원을 고갈시킴으로서 웹 서버의 서비스를 하지 못하게 하는 공격들을 차단할 수도 있다. 그렇지만 대용량 웹 서버의 경우에는 보통 초당 몇 백 메가바이트(MegaByte)의 로그가 기록되므로 실시간으로 로그 기록들을 분석하기 위해서는 고성능의 하드웨어 사양을 요구하고 공격 패턴들의 분류와 클러스터링을 위한 실시간 모니터링이 필요하므로 이에 따른 프로세스에 과부하 문제가 발생할 수 있다.

3. 사례 연구(Case Study)

그림 4는 임의의 공격자가 프로그래밍 오류의 하나인 php파일 업로드 버그가 있는 게시판에 침입한 후 정보 수집을 위해 몇몇 시스템 명령어를 실행했을 때 웹 서버 로그 파일에 나타난 기록과 그 접속 프로세스의 pid를 포착한 것이다[8]. 이 결과에서 알 수 있듯이 수많은 로그 데이터들 중에서 공격자들의 정확한 행동 패턴이 분류되어진다면 pid 추출을 통해 실시간으로 공격 IP를 차단하는 것이 가능하다.

그림 4a: 접속 프로세스의 pid 정보

그림 4b: 쉘 명령어 탐지

III. 결론 및 향후 연구

본 논문에서는 웹 로그 마이닝을 통해 실시간으로 웹 서버의 침입을 탐지하는 모델을 제안하였다. 기존의 웹 소스코드 오류 검출 자동화 툴이나 방화벽만으로는 80번 포트를 통해 실시간으로 이루어지는 웹 관련 공격에 노출되어 있으므로 이를 해결하기 위해 기존의 웹 어플리케이션의 취약점을 분석하였다. 또한 Agent의 도입으로 웹 관련 로그를 분석하여 실시간으로 판단하고, 해당 접속 프로세스를 제거함으로서 공격을 차단할 수 있는 메커니즘을 제안함으로서 침입 탐지율이 높은 웹 서버 보안이 가능함을 살펴보았다. 본 논문에서 제안한 방법을 확대해서 향후에는 프로그래밍 상의 오류를 이용한 침입 탐지뿐 아니라 SQL Injection, XSS(Cross Site Script), Cookie sniffing/spoofing등의 공격 탐지를 위한 연구가 필요하다.

참고 문헌

[1] http://www.stgsecurity.com/data/WebHacking_Tech.pdf, STG 시큐리티(주), "웹 해킹기술"

[2] http://www.krcert.or.kr, "인터넷침해사고대응지원센터", 인터넷 통계

[3] http://www.ezhack.net, 황순일, "웹 어플리케이션 해킹"

[4] "Web Application Security Secrets & Solutions", 사이버출판사, 2002

[5] "리눅스 웹 서버와 실전 웹 해킹", 세화출판사, 2003

[6] "소스코드를 이용한 웹 응용 취약점 분석에 관한 연구", CISC 2003, pp458-462, 추계학술대회

[7] C. Kruegel and G. Vigna, "Anomaly Detection of Web-based Attacks", ACM CCS, 2003

[8] http://www.null2root.org, "Hacker group[NULL@ROOT]", 해킹강좌