DB Application Firewall과 Web Application Firewall의 연동을 통한 불법적인 SQL 질의 차단기법

김수용, 남건우, 김상천*

국가보안기술연구소*

Filtering Unauthorized SQL Query By uniting DB Application Firewall with Web Application Firewall

Su Yong Kim, Geon Wu Nam, Sang Cheon Kim*

National Security Research Institute*

요 약

웹 응용프로그램에 대한 위협이 점차 확산되면서 오늘날 많은 Web Application Firewall들이 등장하고 있다. 하지만, 대부분의 기관에서 웹 서버 자체의 변조는 기관의 미지 실추를 제외하면 업무상 큰 문제를 유발하지 않는다. 웹 서버에 대한 보안을 고려하는 이유는 웹 서버가 침입을 당할 경우 DB 서버의 내용에 손상이 가해질 수 있기 때문이다. 본 고에서는 Web Application Firewall과 연동하여 허용되는 SQL 질의패턴을 자동으로 생성하여 불법적인 SQL 질의를 차단하는 DB Application Firewall을 제안한다. 이를 통해 웹 응용프로그램의 취약점으로 인해 SQL 질의가 변조되더라도 DB 서버에 해당 SQL 질의가 전달되는 것을 차단할 수 있다.

I. 서론

일반 민간 업체들의 업무뿐만 아니라 많은 공공기관의 업무들도 웹 서비스를 통해 이루어질 정도로 웹의 활용은 광범위해 지고 있다. 이로 인해 많은 웹 서버에 대한 침해사고가 빈번히 발생하고 이에 대한 대책도 점차 발전하고 있다. 특히 요즘 문제가 되고 있는 웹 응용프로그램에 대한 침해사고를 예방하기 위한 대책으로 웹 응용프로그램의 보안만을 담당하는 Web Application Firewall까지 등장하고 있다.

많은 기관에서 웹 응용프로그램이나 웹 서버에 대한 침해사고 자체는 기관의 이미지에 좋지 않은 영향을 줄 뿐 업무 자체에 큰 영향을 주지는 않는다. 하지만, 웹 응용프로그램과 연동되어 동작하는 DB 서버가 변조되면 기관의 업무에 큰 문제를 발생시킬 수 있다. DB 서버에는 회원에 대한 중요정보가 포함되어 있고, 심지어 돈과 직결된 정보가 포함되어 있을 수도 있다. 또한, 웹 응용프로그램에는 DB 서버에 접근하기 위해 DB 계정에 대한 정보가 포함되어있다. 웹 서버에 대한 침입만으로 DB 서버의 변조가 가능할 뿐만 아니라 DB 서버에 따라서는 DB 계정만으로 시스템 접근이 가능하여 DB 서버가 존재하는 내부망으로의 침입도 가능하게 된다.

본 논문에서는 웹 응용프로그램의 취약점에 의해 SQL 질의가 변조되더라도 DB 서버의 내용에 대한 불법적인 접근을 차단할 수 있는 DB Application Firewall을 제안한다.

II. 본문

1. 관련 연구

DB Application Firewall에 대한 연구는 현재 거의 이루어지지 않고 있으며, 정의조차 명확하지 않다. 최초의 단일 응용프로그램에 대한 보안장치인 Web Application Firewall에 대해 알아보고, Web Application Firewall과 DB Application Firewall의 연동을 통해 DB에 대한 불법적인 접근을 차단할 수 있는 방안을 제시하고자 한다.

1) Web Application Firewall

Web Application Firewall은 일반적으로 웹 서버와 방화벽 사이에 설치되어 웹 프락시 서버처럼 작동한다. 즉, 사용자와 웹 서버 사이에서 사용자의 요청을 받아 이를 웹 서버에 전달한다. 반대로 웹 서버로부터 웹 페이지를 받아 사용자에게 전달하는 역할도 한다. 두 과정에서 Web Application Firewall은 사용자의 요청이 정상적인지를 판단하여 웹 응용프로그램에 대한 침입시도를 차단한다.

Web Application Firewall 은 침입 여부를 판단하여 필터링하는 방식에 따라 크게 Negative-logic에 의한 필터링 방식과 Positive-logic에 의한 필터링, 그리고 Dynamic-rule에 근거한 필터링 기법으로 나눌 수 있다.[1] 3 가지 필터링 방식 중 Dynamic-rule에 의한 필터링 기법으로 가장 활발하게 개발되고 있는 Sactum 사의 AppShield과 Positive-logic에 의한 필터링 기법을 채택하여 연구되고 있는 사용자 관점의 보안정책 자동 학습 능력을 가진 Web Application Firewall에 대해 소개한다.

가. AppShield[2]

AppShield는 Sanctum 사에서 개발한 Web Application Firewall이다. AppShield에서 불법적인 웹 응용프로그램에 대한 침입시도는 Policy Recognition Engine과 Adaptive Reduction Technology에 의해 차단된다.

사용자의 요청에 의해 웹 서버로부터 사용자에게 웹 페이지가 전송될 때, Policy Recognition Engine 에 의해 자동으로 보안 정책이 설정된다. 사용자가 다시 요청을 보내면, Web Application Firewall은 이를 Reducer에게 보낸다. Reducer는 사용자의 요청을 다른 간단한 형태의 표현으로 변경한다. 변경된 요청을 다시 Expander가 실제 요청으로 변경함으로써 보안 정책에 부합하는 요청만이 웹 서버에 전달되도록 하는 기술이 Adaptive Reduction Technology이다.

예를 들어, 책상, 의자, 펜, 연필, 4가지 물품을 판매하는 웹 응용프로그램이 있다고 가정하면, 웹 서버로부터 사용자에게 웹 페이지가 전달될 때, Policy Recognition Engine은 이 페이지를 분석하여 4가지 품목에 대한 링크를 간단하고 안전한 언어로 변경해 둔다. 가령 책상을 00으로, 의자를 01로, 펜을 10으로 연필을 11로 기술되는 간단하고 안전한 언어로 표현한다. 사용자가 만약 책상에 대한 링크를 클릭하면, Reducer는 그 요청을 00으로 대체하고, 이를 Expander에 보낸다. Expander는 이를 다시 책상 링크에 관한 요청으로 변경하여 웹 서버에 보낸다. 이런 과정을 통해 웹 서버에는 Policy Recognition Engine에 의해 제한된 요청만이 전달된다.

나. 사용자 관점의 보안정책 자동 학습 능력을 가진 Web Application Firewall[3]

기존의 3 가지 필터링 방식 중 Negative-logic에 근거한 필터링 방식은 알려지지 않은 침입에 대해 많은 False Negative 필터링이 발생하는 한계를 지니고 있고, Dynamic-rule에 근거한 필터링 방식은 일부 False Negative 필터링뿐만 아니라 많은 False Positive 필터링이 발생하는 문제점이 있다.

이에 반해, Positive-logic에 의한 필터링은 개념적으로 정상적인 사용을 정의하고 그 외의 모든 접근에 대해 차단하는 것이기 때문에 정상적인 사용에 대한 적절한 정의를 통해 사용자의 정상적인 이용에 불편을 주지 않으면서도 모든 침입을 차단할 수 있다. 따라서, Positive-logic에 의한 필터링이 가장 바람직하지만, 적절한 보안정책을 설정하는 것이 쉽지 않은 단점이 있다.

사용자 관점에서의 자동 보안정책 설정 기법의 기본 개념은 정상적인 사용자의 웹 페이지 사용을 일정 기간 모니터링한 후에 이를 근거로 하여 자동으로 보안정책 설정을 작성하도록 하는 것이다. 이 기법은 정상적인 사용을 제외한 모든 접근을 침입으로 간주하기 때문에 새로운 형태의 침입시도도 쉽게 차단할 수 있다.

사용자 관점에서의 자동 보안정책을 설정하기 위해서는 학습단계를 거쳐야 한다.

학습단계는 웹 응용프로그램이 개발되어 웹 서버에 탑재된 이후, Web Application Firewall을 설치하면서 이루어진다. 이 단계에서는 웹 서비스에 접근할 수 있는 사용자 그룹을 제한한다.

제한된 사용자들은 평소처럼 웹 브라우저를 사용하여 웹 서비스를 이용하는 것으로 충분하다. Web Application Firewall 은 이런 제한된 사용자 그룹의 행위를 모니터링하면서 정상적인 사용에 대한 rule-set을 자동으로 생성한다. 따라서, 학습기간이 길고, 사용자가 많을수록 그 결과는 더 정확해진다. 하지만, 웹 응용프로그램의 경우 매개변수나 쿠키 등의 값들이 크게 변화하지 않는 특성으로 인해 적은 사용자에 의한 적은 학습 기간이라 하더라도 상당히 정확한 결과를 도출할 수 있으며, 적응단계에서 이를 보완할 수 있다.

Web Application Firewall이 보안정책을 자동으로 생성하는 원리는 다음과 같다. 학습기간 동안 정상적인 사용자에 의해 방문되는 모든 페이지를

일반 사용자들에게 접근이 허용되는 페이지로 학습한다. 특정 IP 주소에 대해서 관리자 권한을 설정해 주면, 학습 과정에서 그 IP 주소로부터 접근되는 페이지들은 관리자만이 접근할 수 있는 페이지로 인식되어 일반 사용자들의 접근은 금지된다. 이와 같이 할 경우 학습 과정에서 인식되지 못한 테스트 페이지들과 개발중인 페이지들에 대해서는 어느 누구도 접근할 수 없는 페이지들이 되어 취약한 페이지들의 노출을 막을 수 있다.

쿠키, 매개변수, Hidden Field 등에 허용되는 값을 학습하는 방법은 다음과 같다. 사용자들이 특정 매개변수에 대해서 계속해서 숫자 값을 제공한다면 그 매개변수는 숫자만 허용하는 것으로 학습된다. 사용자의 입력 값이 문자열인 경우는 최대 길이와 특수 문자의 사용에 대해 조사한다. 특수 문자들이 사용될 경우 허용되는 특수 문자로써 인식하고, 그 이외의 모든 특수 문자들을 허용하지 않도록 학습한다. 특히 "." 문자가 허용되는 경우는 "." 문자 뒤에 오는 문자열에 대한 패턴도 학습한다. "." 문자 뒤에 오는 문자열이 확장자로 사용될 수 있기 때문이다. 이를 통해 허용되는 않는 확장자를 가진 파일의 업로드를 막을 수 있다.

2) DB 서버에 대한 침입 유형

DB 서버에 대한 침입시도는 크게 두 가지로 분류할 수 있다. 하나는 DB 서버 운영체제 등의 취약점을 악용하여 불법적인 시스템 권한을 획득하는 것이고, 다른 하나는 DB 서버 자체에 취약점은 없지만, 불법적인 사용자가 SQL 문을 조작하여 악의적인 행위를 수행하는 것이다.

본 고에서 전자의 경우는 고려하지 않는다. 많은 DB 서버들이 방화벽 등에 의해 외부로부터의 침입에 안전하게 보호되고 있기 때문이다. 본 고에서는 후자의 경우를 막기 위한 기법에 대해 살펴본다. 후자의 경우는 두 가지 경우에 가능하다. 웹 응용프로그램의 취약점을 통해 SQL 명령을 조작하는 경우와 웹 데몬의 취약점이나 웹 응용프로그램의 취약점을 이용하여 웹 서버에서 불법적인 권한을 먼저 획득한 후, 웹 응용프로그램에 포함되어 있는 DB 계정을 알아낸 뒤, 이 계정을 이용해 DB 서버에 접근하는 경우이다. 전자의 경우는 Web Application Firewall 에서도 막고자 하지만, 아직까지 모든 침입을 막기에는 부족한 실정이다. 후자의 경우에는 현존하는 어떤 기술로도 막기 어려운 것이 사실이다. 웹 서버 내에는 어떤 형태로든 DB 계정에 대한 정보가 존재해야 하고, 이를 이용하여 정상적인 DB 접근을 차단하기는 어렵기 때문이다. 또한, 악의의 사용자는 웹 서버에서 DB로 접근하려 하기 때문에 DB 서버에 대한 IP 접근을 제한하는 것도 역시 불가능하다.

2. DB Application Firewall

1) 정의

DB Application Firewall은 DB 서버들에 대해서 허용되지 않는 SQL 질의를 제한하기 위해 사용자와 DB 서버 사이에 존재한다. DB Application Firewall은 사용자와 DB 서버 사이에 오가는 모든 통신을 분석하여 침입으로 판단되는 통신을 차단한다. 본 고에서 제안하는 DB Application Firewall은 Positive-logic 에 의한 필터링을 사용하여 웹 응용프로그램에서 사용하는 SQL 질의 패턴과 일치하지 않는 모든 SQL 질의를 침입으로 간주한다.

DB Application Firewall 과 Web Application Firewall은 다음과 같은 위치에 배치된다.

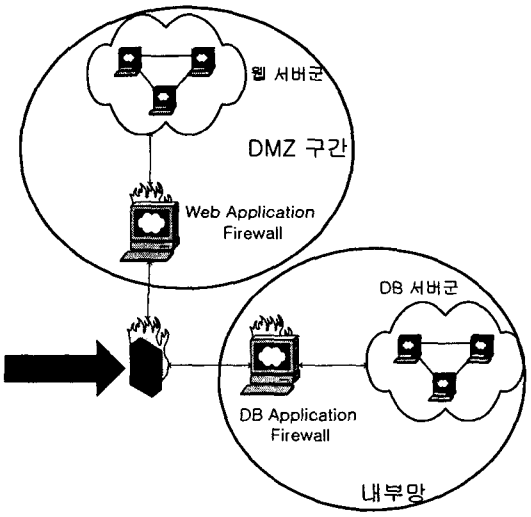

그림 1: DB Application Firewall과 Web Application Firewall의 배치도

DB Application Firewall은 정상적인 SQL 질의에 대한 rule-set을 만들기 위해서 학습과정을 거친다. 학습과정에서는 관리자만이 순차적으로 웹 사이트를 방문하면서, 정상적인 SQL 질의를 DB 서버에 보낸다. 이 과정에서 DB Application Firewall은 특정 SQL 질의가 올 때마다 Web Application Firewall과 통신하여 현재 사용자에 의해 요청되는 웹 응용프로그램의 URL을 확인한다. 이런 과정을 통해 DB Application Firewall은

특정 URL에 해당하는 정상적인 SQL 질의의 패턴을 테이블 형태로 기록할 수 있다.

이와 같은 학습단계를 거친 뒤, DB Application Firewall은 실제 네트워크에서 운영된다.

2) 동작원리

학습단계에서 생성된 정상적인 SQL 질의 패턴 rule-set을 이용하여 운영단계에서 DB 서버에 대한 불법적인 SQL 질의를 막는 방법에 대해서 살펴본다.

웹 페이지에 대한 사용자의 요청이 Web Application Firewall에 도달하면, Web Application Firewall은 요청된 URL을 저장하고, 웹 서버에 해당 URL을 요청한다. 웹 서버는 DB에 대한 SQL 질의가 필요할 경우, SQL 질의를 DB 서버에 보낸다. SQL 질의는 DB 서버에 도착하기 전에 DB Application Firewall에 도착한다. DB Application Firewall은 SQL 질의의 패턴을 분석하여 어떤 URL을 요청했을 때 일어나는 SQL 질의인지를 찾는다. 만약 정상적인 "URL/SQL 질의패턴" 테이블에 존재하지 않는 패턴일 경우 이를 불법적인 SQL 질의로 보고 DB 서버에 대한 접근을 허용하지 않는다. SQL 질의 패턴에 대한 적절한 URL이 존재하면, 해당 URL이 실제로 웹 서버에 요청되었는지를 Web Application Firewall에 확인한다. Web Application Firewall은 자신의 테이블에 해당 URL의 존재여부를 DB Application Firewall에 알려준다. Web Application Firewall로부터 해당 URL이 존재하는 것으로 알려지면, DB Application Firewall은 비로소 SQL 질의를 합법적인 것으로 간주하고, 이를 DB 서버에 전달한다. DB 서버는 SQL 질의 결과를 웹 서버에 전달하고, 웹 서버는 사용자에게 보낼 웹 페이지를 Web Application Firewall에 보낸다. Web Application Firewall은 테이블에서 해당 URL을 삭제하고, 사용자에게 웹 페이지를 보낸다.

3) Web Application Firewall의 기능

DB Application Firewall과 연동되기 위해서는 기존의 Web Application Firewall에 수정이 필요하다. 먼저 DB Application Firewall과 통신하기 위한 메커니즘이 필요하고, 특별한 테이블을 유지해야 한다. 테이블은 "요청URL/요청시간/요청IP"을 유지해야 한다.

학습단계에서 Web Application Firewall은 여러 웹 페이지에 대한 요청이 오면, 하나씩 순차적으로 웹 서버에 요청을 보낸다. 웹 페이지에 대한 요청이 오는 즉시 웹 서버에 모두 전송할 경우 DB Application Firewall에서 SQL 질의를 발생시킨 웹 페이지를 찾을 수 없기 때문이다. 웹 페이지에 대한 요청을 순차적으로 처리함으로써 DB Application Firewall에서 현재 처리되고 있는 웹 페이지를 질의할 때 해당 URL을 알려줄 수 있다.

운영단계에서는 속도문제로 인해 웹 페이지 요청을 순차적으로 처리하는 것은 불가능하다. 따라서, 사용자로부터 특정 URL에 대한 요청이 도착할 때마다 테이블에 URL과 요청이 도착한 시간, 요청한 사용자의 IP를 저장하고, 즉시 웹서버에 요청을 전송한다. 웹서버로부터 결과 페이지가 도착하면 해당 요청의 URL에 대한 정보를 삭제하고 사용자에게 결과 페이지를 전송한다.

운영단계에서 DB Application Firewall로부터 시간과 URL에 대한 정보가 전송되어오면, 해당 시간 이전에 URL에 대한 요청이 존재하는지를 확인하여 DB Application Firewall에 알려준다. 이는 특정 URL에 해당하는 SQL 질의는 반드시 URL에 대한 요청 시간보다 더 늦게 DB Application Firewall에 도착해야 하기 때문이다.

4) DB Application Firewall의 기능

DB Application Firewall에서 가장 중요한 것은 "URL/SQL 질의패턴"의 정보를 저장하고 있는 테이블이다. DB Application Firewall은 학습단계에서 SQL 질의가 도착할 때마다 Web Application Firewall에 해당 웹 응용프로그램의 URL을 질의한다. Web Application Firewall로부터 해당 URL을 받아 "URL/SQL질의패턴" 테이블에 추가한다.

이렇게 생성된 테이블을 이용해서 운영단계에서는 웹 서버로부터 SQL 질의 요청이 오면 SQL 질의의 패턴을 분석하여 이 SQL 질의 요청이 일어나기 위해 사용자가 요청했을 URL을 찾아낸다. 이런 URL이 존재하면, SQL 질의가 DB Application Firewall에 도착한 시간 이전에 해당 URL 요청이 있었는지를 Web Application Firewall에 질의한다. Web Application Firewall로부터 해당 URL 요청이 존재하는 것으로 확인되면, SQL 질의를 DB 서버에 전송한다.

만약 도착한 SQL 질의가 "URL/SQL질의패턴" 테이블에 존재하는 않거나 해당 URL이 Web Application Firewall에 존재하지 않으면 불법적인 SQL 질의로 간주하고 DB 서버로의 접근을 차단한다.

5) 침입 시도 차단 예제

본 고에서 제안하는 기법이 예상되는 침입 시도

들을 어떻게 막을 수 있는지 살펴본다. 일반적으로 웹 응용프로그램의 취약점을 이용하는 SQL Injection과 같은 경우는 거의 불가능하다. 왜냐하면 SQL Injection과 같은 경우 SQL 문을 변조하게 되는데, 변조된 SQL 질의는 DB Application Firewall의 "URL/SQL질의패턴" 테이블에 존재하지 않기 때문에 DB로의 접근이 차단되기 때문이다.

웹 서버에 불법적인 침입이 성공한 후에 DB 계정을 획득하여 DB 서버에 불법적인 SQL 질의를 실행하려는 경우는 현존하는 어떤 기술로도 차단하는 것이 불가능하다. 하지만, 본 고에서 제안하는 기법을 이용하면 DB 계정을 알고 있다하더라도 DB Application Firewall에 의해 웹 응용프로그램에서 사용되지 않는 SQL 질의는 허용되지 않는다. 웹 응용프로그램에서 사용하는 SQL 질의를 사용하고자 하는 경우에도 웹 응용프로그램 자체를 수정한 뒤 가능하기 때문에 쉽지 않음을 알 수 있다.

3. 결론

본 고에서 제안하는 기법은 Web Application Firewall과 DB Application Firewall을 연동하여 변조된 SQL 질의를 차단할 수 있다. 악의의 사용자가 웹 서버의 시스템 권한을 불법적으로 획득하는 것은 Web Application Firewall에 의해서 차단되어야 하지만, 모든 SQL 질의에 대한 변조까지 Web Application Firewall에서 차단하는 것은 매우 어렵다. 따라서, 이런 DB 서버에 영향을 줄 수 있는 SQL 질의의 변조에 대해서는 DB 서버들을 보호하는 DB Application Firewall에 의해 이루어져야 한다.

참고문헌

[1] Whale Communications, *e-Gap Application Firewall Appliance*, 2003.

[2] Sactum, Inc, *AppShieldTM White Paper*, 2001.

[3] 김수용, 남건우, 박중길, *사용자 관점의 보안정책 자동 학습 능력을 가진 Web Application FirewallA*. 한국인터넷정보학회 동계학술대회, 2003

[4] SQL Injection FAQ, *http://www.sqlsecurity.com/DesktopDefault.aspx?tabindex=2&tabid=3*