프로파일기반 웹 어플리케이션 공격탐지 및 필터링 기법

윤 영 태[†] · 류 재 철[†††] · 박 상 서[†] · 박 종 욱[††]

요 약

최근 웹서버 해킹은 전통적인 해킹기법에 비해 상대적으로 취약한 오픈소스 기반 웹 어플리케이션의 취약점을 이용한 어플리케이션 해킹으로 가고 있는 추세에 있다. 또한, 최근 웹서버는 데이터베이스와 연결을 통해 사용자정보 등을 저장하고 있어, 웹 인터페이스를 통한 데이터베이스 해킹으로 이어지는 문제점을 가지고 있다. 웹 어플리케이션에 대한 공격은 웹서버 자체의 취약점을 이용하는 것이 아니라 특정목적으로 작성된 웹 어플리케이션의 구조, 논리, 코딩상의 취약점을 이용하는 것으로, 공격을 방어하기 위해 패턴매칭을 이용한 필터링을 수행하거나 코드를 수정하는 방법이 있을 수 있지만 새로운 공격에 대해서는 탐지 및 방어가 어렵다. 본 논문에서는 다양한 웹 어플리케이션에 존재하는 취약점을 방어하기 위해 웹 어플리케이션의 구조와 특징 값을 추출하는 프로파일링 기법을 이용하여 비정상 요청을 탐지하는 방법을 제시하고, 웹 프락시 형태로 비정상적인 웹 어플리케이션 공격을 탐지 및 필터링 하도록 구현하였다.

키워드 : 웹 어플리케이션 보안, 프로파일링, 웹 해킹

Profile based Web Application Attack Detection and Filtering Method

Young-tae Yun[†] · Jae-cheol Ryou[†††] · Sangseo Park[†] · Jong-wook Park[††]

ABSTRACT

Recently, web server hacking is trending toward web application hacking which uses comparatively vulnerable web applications based on open sources. And, it is possible to hack databases using web interfaces because web servers are usually connected databases. Web application attacks use vulnerabilities not in web server itself, but in web application structure, logical error and code error. It is difficult to defend web applications from various attacks by only using pattern matching detection method and code modification. In this paper, we propose a method to secure the web applications based on profiling which can detect and filter out abnormal web application requests.

Key Words : Web Application Security, Profiling, Web Hacking

1. 서 론

최근 웹 서버 해킹은 전통적인 시스템 해킹기법에 비해 상대적으로 공략하기 쉬운 오픈소스 기반 웹 어플리케이션의 취약점을 이용한 어플리케이션 해킹으로 가고 있는 추세에 있다. 웹 어플리케이션 해킹은 운영체제나 웹서버 자체에 대한 공격을 의미하는 것이 아니라, 웹 서버에서 동작하는 어플리케이션의 구조, 논리, 코딩상의 문제점을 해킹하는 것이다.

일반적으로 웹 어플리케이션은 운영체제 및 데이터베이스 시스템과 상호 작용을 통해 동작하기 때문에 의도하지 않은 운영체제 명령의 수행 및 사용자 및 트랜젝션 정보와 같은 중요정보의 노출 등이 발생할 수 있다. 특히, SQL 인젝션 공격 등은 웹 인터페이스를 통해 데이터베이스의 중요 정보를 노출할 수 있는 위협으로 많이 알려져 있다. 또한, 2004년 12월에 발생된 샌티(Santy) 웜[1]의 경우와 2005년 11월에 발생된 루퍼(Lupper) 웜[2]의 경우 웹 어플리케이션의 취약점이 웜 공격에서 전파를 위한 수단으로 사용된 사례로 웹 어플리케이션에 대한 보안은 계속해서 중요성이 강조되고 있다.

웹 어플리케이션에 대한 보안 취약점과 관련하여 OWASP (Open Web Application Security Project)에서는 2004년 웹 어플리케이션 취약점에 대해 10가지로 분류하여 발표하였다[3]. OWASP 분류에서 첫 번째 문제로 사용자의 요청에 대해서 입력 값을 검증하지 않는 것을 지적하고 있다. 샌티 웜의 경우에서도 phpBB의 취약점을 이용한 것으로 웹 페이지를 변조하는 공격을 수행하게 되는데, phpBB 코드 내에서 입력 값에 대한 검증이 없었기 때문에 발생한 문제이다. 입력 값을 검증하기 위해서는 데이터 타입, 허용 가능한 문자 셋, 입력 값의 길이, 변수 중복의 허용여부, 숫자의 범위,

※ 제 2저자는 정통부의 ITRC 프로그램의 지원을 받았음.
† 정 회 원 : 국가보안기술연구소 선임연구원
†† 정 회 원 : 국가보안기술연구소 책임연구원
††† 종신회원 : 충남대학교 정보통신공학부 교수
논문접수 : 2005년 9월 9일, 심사완료 : 2006년 1월 19일

지정된 데이터 목록, 지정된 패턴 등을 이용한 검증방법 들이 고려될 수 있다. 웹 어플리케이션은 구조상 지정된 변수들을 사용하여 내부 로직을 구성하고 있으며, 입력받은 값들을 각각의 변수에 대응시켜 전달하는 구조를 가지므로 각 변수들에 대한 프로파일링을 통해 탐지를 수행하는 것이 효율적이다.

본 논문에서는 부적절한 입력 값을 검증하기 위한 방법으로 웹 어플리케이션의 구조와 파라미터 입력 값에 대한 타입과 길이와 같은 특성 값을 자동으로 추출하는 프로파일링 기법을 사용한다. 시험을 통해 분석한 결과 본 논문에서 제시한 프로파일 식별자에 의한 탐지가 매우 효과적이었으며, 파라미터 타입과 길이에 의한 탐지방법을 통해 실시간으로 웹 어플리케이션에 대한 공격을 탐지 할 수 있었다.

본 논문의 구성은 2장에서 웹 어플리케이션 보안방법에 대한 관련연구를 분석하고, 3장에서는 제안된 시스템 구조를 설명하고, 4장에서 제안된 탐지 및 필터링 방법에 대해 기술하고, 5장에서는 제안된 시스템에 대한 시험 및 분석을 하고, 6장에서 결론을 맺는다.

2. 관련 연구

웹 어플리케이션 공격 탐지 및 필터링을 위한 연구로는 웹 어플리케이션으로의 입력 값을 규칙에 의해 접근을 통제하는 연구와 문제를 발생시킬 가능성이 높은 메타 문자에 대한 필터링 코드를 삽입하는 연구들이 진행되고 있다.

Krugel은 [4]에서 웹 어플리케이션 침입탐지를 위한 방법으로 파라미터의 길이, 입력 문자의 분포 등을 이용한 확률적 이상상태 점수를 계산함으로써 공격을 탐지하는 모델을 제시하였다. 파라미터 길이에 대해서는 입력 값들에 대한 학습을 통해 평균 및 표준편차를 이용하여 확률적으로 임계치를 계산하게 되는데, 이 방법은 학습 데이터에 매우 의존적으로 오탐율이 높게 나올 수 있다. 특히, 정수형을 가지는 파라미터의 경우 데이터 타입에 따라 임계치를 예측하게 되는 경우 오탐율을 줄일 수 있는데, 본 논문에서는 파라미터 입력 값에 대한 최대길이와 함께 데이터 타입에 따라 정수형의 경우 지정된 길이를 허용할 수 있어 오탐율을 줄일 수 있게 된다. 본 논문에서는 실시간으로 이상상태를 탐지하여 필터링하는 방법을 제안하는 것으로 프로파일 레코드의 키로 이용되는 프로파일 식별자를 이용한 탐지방법에서 좋은 탐지효과를 가지게 된다.

Scott은 [5]에서 어플리케이션 레벨에서 웹 보안을 위한 보안정책을 기술하는 SPDL(Security Policy Description Language)를 제안하였다. SPDL은 XML 기반으로 구성되며 요청 메쏘드와 파라미터의 이름을 키로 하여 파라미터 값의 최소 및 최대 길이를 정의하고, 파라미터의 타입을 정수, 실수, 불린(boolean), 문자열 타입으로 구분하여 해당 페이지 요청에 대해 접근을 통제하는 규칙을 정규화하게 된다. 하지만, 웹 어플리케이션의 모든 링크에 대해 수동으로 접근 규칙을 생성하는 것은 매우 어려운 문제이다. 본 논문에서도 접근을 제어하는 방법으로 파라미터의 길이와 타입을 이용하게 되는데, 접근 규칙을 생성하는 방법으로 웹 프로파일링 기법을 이용하여 정상적인 요청에 대한 학습과정을 통해 자동화된 규칙을 생성함으로써 접근규칙을 효율적으로 생성 할 수 있도록 구성하였다. [6]에서는 소스코드에 대한 타입기반 정적분석을 수행하여, 실행 시에 취약할 수 있는 부분에 변수 입력 값을 검사하고 필터링하는 코드를 삽입함으로써 웹 어플리케이션을 소스코드 레벨에서 보호하는 방법을 제시하였으며, WebSSARI라는 코드 분석도구를 개발하였다. WebSSARI는 PHP로 구현된 웹 어플리케이션에 대한 코드분석 기능 및 필터링 코드 삽입 기능을 제공하게 된다. 하지만, 소스코드 기반에서 분석을 해야 하기 때문에 공개되지 않은 어플리케이션에 대해서는 보호기능을 제공할 수 없는 단점이 있다. 본 논문에서 제시하는 방법은 소스코드에 대한 변경 없이 웹 어플리케이션에 전달되는 사용자 입력 값을 검사하기 때문에 공개되지 않은 어플리케이션에 대해서도 필터링이 가능한 장점을 갖게 된다.

웹 어플리케이션에 대한 특성 값을 추출하기 위해서는 웹 어플리케이션 구조에 대한 학습이 필요하게 된다. 웹 어플리케이션을 학습하기 위한 방법으로 웹 로그를 이용하는 방법, 웹 링크를 추적하는 방법이 있을 수 있다. VeriWeb은 웹 사이트의 링크를 자동으로 추적하는 방법을 제시하고 있다. 기존 웹 수집(Crawling) 기법에서는 웹 어플리케이션에 대한 폼 필드를 이용하는 링크와 클라이언트 스크립트에서 발생하는 링크를 추적하는데 한계가 있었으나, Veriweb에서는 폼 필드를 자동으로 입력하고 이에 대한 링크를 추적하여 요청함으로써 보다 정교한 웹 사이트 분석이 가능하도록 하였다[7]. 그러나, 프로파일링에서는 웹 어플리케이션으로의 입력 패턴에 대한 학습이 이루어져야 하므로, 웹 프로파일러를 웹 서버 전단에 프락시 형태로 구성함으로써 프락시를 경유하여 전송되는 입력 패턴을 학습하도록 구성하고 있다.

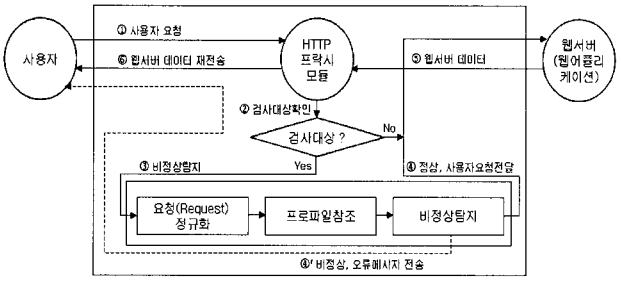
3. 프로파일 기반 웹 어플리케이션 보안

웹 어플리케이션 프로파일은 사전에 웹서버에서의 어플리케이션 구조를 파악하여 이를 데이터베이스화하는 것이다. 웹 어플리케이션은 이미 지정된 변수들을 사용하여 내부 로직을 구성하고 있으며 변수들이 사용자로부터 입력받은 값들을 각각의 변수에 대응시켜 전달하는 구조를 가지므로, 각 변수들에 대한 특성값을 프로파일로 구성함으로써 웹 어플리케이션의 이상상태 탐지를 위한 방법으로 매우 적합하다.

3.1 제안된 시스템 구조

웹 어플리케이션에 대한 프로파일링을 수행하기 위해서는 사용자 브라우저와 웹서버 사이에 전송되는 데이터에 대한 접근이 가능해야 한다. 이러한 요구사항을 만족하기 위한 시스템 구조로는 웹 서버 자체를 변경하는 방법, 웹 서버 전단에서 프락시 형태로 구성하는 방법, Mod_Security[8]와

같이 웹 서버의 모듈로 동작하는 방법이 있다. 본 논문에서는 프로파일 기반의 이상상태 탐지를 제공하는 부분은 앞에서 언급한 각각의 구조에 병합될 수 있으나, 구현 및 실험의 편의성을 위해 프락시 형태로 구성하는 방법을 제시하고 이를 통해 실제 시험 및 분석을 수행하였다.



(그림 1) 제안된 시스템 구조

(그림 1)에서와 같이 사용자와 웹서버의 데이터는 HTTP 프락시 모듈을 통해 재전송된다. 먼저, 사용자의 HTTP 요청은 프락시 모듈에 전달되고, 전달된 URL이 미리 설정해 놓은 웹 어플리케이션 디렉토리에 해당되면 비정상 탐지를 수행하게 된다. 만약, 요청 URL이 웹 어플리케이션 디렉토리에 해당되지 않는다면 웹서버로 사용자 요청을 전달하여 웹서버에서는 정상적인 데이터를 프락시 모듈에 전달하고, 전달된 데이터는 다시 사용자에게 전달되도록 한다.

검사대상에 해당하는 경우 요청 URL을 파싱하여 프로파일 자료구조와 매칭 시킬 수 있도록 URL 디코딩, Query 파싱 과정을 거치면서 요청내의 인코딩된 문자들을 파싱 가능한 문자로 변경하고, 파라미터의 분리 및 특성 추출을 수행하게 된다. URL 디코딩 및 Query 파싱은 다음 절에서 자세히 소개된다. 요청에 대한 파싱이 완료되면 프로파일 자료구조를 참조하여 매칭되는 것이 있으면 비정상탐지를 통해 파라미터에 대한 특성검사를 수행하게 되고, 만약 매칭이 되지 않으면 비정상 행위로 판단하여 오류메시지를 전송하게 된다. 비정상탐지에서는 파라미터의 길이 및 타입을 검사하게 된다. 비정상탐지를 통해 정상으로 판단되는 요청에 대해서는 웹서버에 사용자의 요청을 전달하게 되고 웹서버의 응답을 클라이언트에 전송하게 된다.

3.2 웹 어플리케이션 프로파일링

웹 어플리케이션 프로파일은 사전에 웹서버에서의 어플리케이션 구조를 파악하여 이를 데이터베이스화 하게 된다. 웹 프로파일러는 (그림 1)에서의 프락시 모듈 형태와 유사하며, 사용자의 요청 데이터를 분석하여 (그림 2)에서와 같이 프로파일 레코드를 생성하고, 생성된 레코드를 파일에 저장함으로써 정상프로파일 데이터를 생성하게 된다. [7]에서는 능동적으로 웹 사이트에 접속하여 사용자 입력부분을 시험하고 통계하는 연구가 있었으나, 본 논문에서는 프락시 기반에 수동적으로 프로파일링을 수행하게 된다. 구현에서는 프로파일링 데이터 수집을 위해 웹서버 로그로부터 프로파일 데이터를 생성하는 모듈도 함께 사용이 되었다. 사용자의 요청은 '?' 문자로 구분되며, 뒤쪽에 파라미터 변수와 값이 '=' 문자를 통해 각각 대응되게 된다. 웹 프로파일러에서는 레코드 구성에서 각 파라미터 변수를 연결하여 프로파일 레코드의 구분자인 키(key) 값을 구성하게 된다. 이때, 요청 URL내에 유니코드 지원을 위해 인코딩된 데이터의 경우 URL 디코딩 과정을 통해 변환하고, 공백(Space) 문자를 모두 제거함으로써 프로파일 레코드를 구성하는데 문제가 없도록 한다.

Profile 테이블

index	Profile_id	parameter_link
0	http://192.168.1.36/zboard/view.php-idno	
1	http://129.168.1.36/zboard/zboard.php-id	
2	http://129.168.1.36/zboard/zboard.php-logout.php-s_url	
3	http://129.168.1.36/zboard/view_info2.php-member_no	
.....	.....	.....

index	name
0	id
1	no

Parameter 테이블

index	name	type	len
0	id	Alpha	9
1	no	int	5
2	sn1	int	1
3	divpag	Meta	3
.....			

(그림 2) 프로파일 레코드의 구성

$http://host/cgi-bin/vuln.cgi?p_1=v_1\&p_2=v_2\&p_3=v_3\&.....\&p_n=v_n$과 같은 사용자 요청이 오면, 주어진 요청 URL로부터 프로파일 레코드의 키인 $Key(Q) = http://host/cgi-bin/vuln.cgi-p_1p_2p_3.....p_n$로 구성된다. 여기서, p는 변수의 이름이고 v는 변수에 대한 값이다. 키 값을 Key(Q)와 같이 구성하는 이유는 웹 어플리케이션에서 사용되는 URL을 구분할 수 있는 식별자로 이용하기 위한 것이다. 또한, 키 값을 이와 같이 구성함으로써 php와 같이 스크립트 방식의 어플리케이션의 경우에 코드내의 숨겨진 변수를 공격자가 직접 요청 내에 포함하여 전달하는 공격의 경우에, 이미 정상적인 요청만으로 구성된 프로파일 내에서는 존재하지 않기 때문에 비정상으로 판단하는데 단순하면서도 정확한 판단결과를 도출하는데 이용될 수 있다.

프로파일 레코드의 구성은 해당 키 값에 대응되며, 파라미터 변수에 대한 값들의 특성을 추출하게 된다. 이때 각 파라미터 변수에 대한 값들의 특성은 길이와 타입을 추출하게 된다. 웹 프로파일러에서는 동일한 프로파일 키에 해당하는 경우 길이는 입력데이터에서 최대 길이를 저장하도록 구성하고, 타입은 숫자(Numeric), 알파벳(Alphabet), 숫자-문자(Alphanumeric), 메타 문자(Meta Character)로 구분하여 프로파일링된 값의 타입과 사용자 입력 값의 타입을 비교하여 비정상을 탐지하는데 활용된다. 웹 프로파일러를 통해 생성된 레코드는 파일로 저장되며, 파일은 다시 필터링 기능을 수행하는 프락시 모듈이 시작되면서 프로파일 자료구조 형태로 구성하여 사용된다.

3.3 비정상 탐지 방법

본 논문에서 제시하는 비정상 탐지 알고리즘은 세 가지로 구성된다. 첫 번째는 프로파일링 식별자에 의한 탐지 방법, 두 번째는 Query 파라미터 변수에 대한 값의 타입의 변화에 따른 탐지 방법, 세 번째는 변수에 대한 값의 길이의 변화에 따른 탐지방법이다.

3.3.1 프로파일링 식별자에 의한 탐지 방법

프로파일링 식별자에 의한 탐지방법은 사용자의 요청이 프로파일 레코드에 없는 경우 비정상적인 요청으로 판단하는 방법이다. 이 탐지방법은 PHP나 ASP, JSP와 같은 서버 스크립트 방식의 오픈소스 웹 어플리케이션의 경우 소스가 공개되어 코드내의 내부 변수를 사용자가 직접 입력하는 공격에 효과적으로 탐지가 가능하다. 그 이유는 프로파일링을 통해 정상적으로 조합 가능한 파라미터 변수들을 키로 하여 유지하기 때문에, 요청 URL에서 새로운 변수가 사용되는 것을 탐지할 수 있게 된다. 구현된 프로파일 식별자에 의한 탐지방법은 검색의 효율성을 위해 Hashed-Linked 리스트 구조를 이용하였다. 사용자의 요청은 프로파일링에 사용되었던 방식과 마찬가지로 URL 요청을 파싱하여 키 형태로 구성하고, 이를 숫자화 하여 해쉬한 값을 프로파일 테이블의 인덱스로 사용한다. 인덱스가 결정이 되면 링크드 리스트에서 순차적으로 매치되는 키 값을 비교하고, 해당 키 값을 찾지 못하게 되는 경우 비정상으로 판단하게 된다.

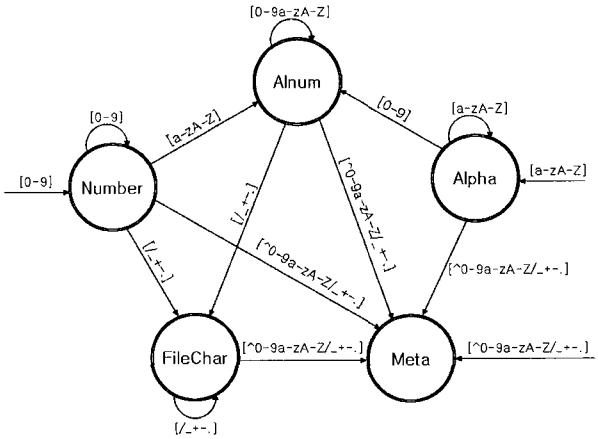
3.3.2 파라미터 변수 값의 길이를 이용한 탐지 방법

버퍼오버플로우나 스크립트 인클루드(include) 공격의 경우 정상적인 인자 값에 비해 긴 값을 가지게 된다. 따라서, 프로파일링을 통해 정상적인 요청에 대한 파라미터 변수에 대한 값의 길이를 분석함으로써 비정상상태를 탐지할 수 있다.

파라미터 변수 값의 길이는 각 요청에서 파라미터 길이를 추출하고, 동일 파라미터 변수에 대해 최대 길이를 가지는 값을 프로파일내에 기록하게 된다. 이때, 데이터타입이 정수형을 가지게 되는 경우, 숫자의 범위를 고려하여 길이를 결정하게 된다. 프로파일링을 통해 얻어진 파라미터 값의 길이는 해당 파라미터 변수에 대한 값의 임계치로 사용되는데, 만약 입력 요청이 해당 임계치를 초과하는 경우에 비정상적인 요청으로 처리하게 된다. 물론, 입력변수의 값이 사용자의 입력에 따르기 때문에 오경보(False Alarm)가 발생할 수 있다. 따라서, 파라미터 길이에 대한 임계치는 웹 어플리케이션 내에서 고려되는 최대 임계치를 확인하여 관리자에 의한 보정이 필요하다.

3.3.3 파라미터 타입에 따른 탐지방법

파라미터 타입에 따른 탐지는 프로파일링을 통해 파라미터 변수에 대한 값의 타입을 분석하여 저장하고, 사용자의 요청 값의 타입과 비교하여 변화가 발생했을 때 이를 비정상으로 탐지하는 방법이다. 버퍼오버플로우 공격의 경우 쿼리에 따른 변수 값이 쉘 코드 등을 일반적으로 가지기 때문에 쉘 코드가 담고 있는 특수한 문자에 대해서 타입 유효성 검사를 통해 이상상태를 탐지할 수 있다. 또한, php 스크립트에서 원격 스크립트 인클루드나 악성 php 업로드 공격을 통해 파라미터로 전달되는 경우 특수문자들이 포함될 수 있는데, 싱글쿼트('), 더블쿼트("), 콜른(':'), 세미콜른(;), 백슬래쉬(\)와 같은 메타 문자들이 쿼리내에 포함된 경우 이상상태를 탐지할 수 있다. 본 논문에서는 (그림 3)에서와 같이 파라미터 값의 타입을 숫자(Number), 알파벳(Alpha), 숫자-알파벳(Alnum), 파일문자(FileChar), 메타문자(Meta)와 같이 5가지 타입으로 구분한다.



(그림 3) 파라미터 타입 구별을 위한 상태 다이어그램

파라미터 값이 숫자가 오고, 문자열의 끝까지 숫자가 연속해서 오는 경우에는 Number로 판별하게 되고, 중간에 알파벳 문자가 오는 경우에는 Alnum으로 판별하고, 특수문자가 오는 경우 파일에 사용되는 특수문자인 경우에는 FileChar, 그 외의 문자에 대해서는 Meta로 판별하게 된다. 알파벳 문자로 시작하는 경우에는 문자열의 끝까지 숫자가 연속해서 오는 경우에는 Alpha로 판별하고, 중간에 숫자가 오면 Alnum으로, 파일에 사용되는 특수문자가 오는 경우에는 FileChar로 판별하고 그 외의 문자에 대해서는 Meta로 판별하게 된다. 특수문자에 대해서 FileChar와 Meta 상태로 구분하는 이유는 많은 어플리케이션에서 변수의 입력으로 파일이름을 받아들이도록 구현되어있기 때문이다.

예를 들어, 웹서버 통계를 출력해주는 awstats[17]라는 웹 어플리케이션에서는 configdir이라는 변수를 이용하여 설정 파일의 위치를 지정하게 되어있다. 그런데, configdir 변수는 시스템 명령을 받아들이게 되면 입력 명령을 수행할 수 있는 취약점이 존재하였다. (그림 4)에서는 configdir 변수에 대해 정상적인 로그를 보여주고 있는데 이와 같은 경우 configdir 변수는 FileChar에 해당하게 된다. 하지만, (그림 5)에서의 비정상적인 로그를 보면 ';' 문자로 인해 Meta 상태로 판단이 되게 된다. 이와 같이 정상적인 프로파일에 기록된 파라미터의 값의 타입에 따른 비정상 상태를 판단할 수 있게 된다.

```
xxx.xxx.xxx.xxx - - [16/Apr/2005:06:32:00 +0000] "GET cgi-
bin/awstats/awstats.pl?configdir=/etc&config=linux HTTP/1.1" 200 538 "-" "Mozilla/4.0
(compatible; MSIE 6.0; Windows NT 5.1)"
```

(그림 4) 정상적인 awstats 로그

```
xxx.xxx.xxx.xxx - - [26/Jan/2005:06:32:00 +0000] "GET cgi-
bin/awstats/awstats.pl?configdir=cd%20/tmp;wget%20http://www.nokiacxxxx.cz/dcha0s/cgi;l
s%20-la%20cgi;chmod%20777%20cgi;./cgi;%00 HTTP/1.1" 200 538 "-" "Mozilla/4.0
(compatible; MSIE 6.0; Windows NT 5.1)"
```

(그림 5) 비정상적인 awstats 로그

4. 시험 및 분석

본 논문에서 구현된 웹 어플리케이션 보안 프락시와 프로파일러는 레드햇 리눅스 9.0에서 구현되었으며, Apache 웹 서버를 대상으로 시험이 수행되었다. 프로파일러는 두 가지 형태로 구성이 되는데 첫 번째는 프락시 형태로 사용자의 실시간 연결요청에 대해 프로파일 레코드를 생성하는 방식이고, 다른 한 가지는 웹 로그 파일을 입력으로 하여 프로파일 레코드를 생성하는 형태이다. 시험에 사용한 웹 어플리케이션은 <표 1>에서와 같이 각각 구현방식이 다른 4개의 어플리케이션을 대상으로, <표 2>에서와 같이 버그트랙에 취약점이 보고된 어플리케이션 들이다. 본 논문에서는 두 가지 측면에서 시험 및 분석을 수행하였다. awstats, phpBB[18], CrazyWebBoard[19], zeroboard[20] 어플리케이션에 대해 프로파일링을 수행하여, 논문에서 제시한 프락시 기반의 필터링 시스템에서 유지해야 하는 프로파일 레코드의 수를 확인함으로써 구현된 시스템의 효용성을 확인한다. 두 번째는 zeroboard 웹 보드를 이용하여 서비스하는 웹서버의 로그를 이용하여 정상적인 요청과 비정상적인 요청으로 구분을 수행하여 오탐율(false detection rate)을 시험하였다.

〈표 1〉 웹 어플리케이션

웹 어플리케이션	설 명	비 고
ZeroBoard 4.1 pl4	게시판 프로그램	PHP 방식
phpBB 2.09	포럼 게시판 프로그램	PHP 방식
CrazyWebBoard 3.01	게시판 프로그램	Cgi 방식
Awstats 6.2	웹 서버 통계 프로그램	Perl 방식

〈표 2〉 버그트랙에 보고된 취약점

웹 어플리케이션	버그트랙 보고
ZeroBoard 4.1 pl4	13823, preg_replace remote command execution vulnerability 12103, zeroboard multiple remote script injection 12257, zeroboard multiple file disclosure vulnerability
phpBB 2.09	GLSA 200501-36, Remote Command Execution GLSA 200507-03, Arbitrary command execution
Awstats 6.2	10950, Rawlog plugin Logfile Parameter Input valication Vulnerability 12298, Remote Command Execution Vulnerability

4.1 프로파일링 시험

정상적인 요청을 위한 프로파일링은 웹 페이지의 링크를 전체 탐색하여 복제할 수 있는 TeleportPro[21] 프로그램을 이용하여 프락시 형태의 프로파일러를 통해 정상적인 프로파일 레코드를 생성하였다. 생성된 레코드의 수는 <표 3>에서와 같은 결과를 얻을 수 있었다. 이 중에서 zeroboard 어플리케이션은 아파치 웹서버를 운영하면서, zeroboard를 운영해온 웹서버의 웹 로그를 이용하여 산출한 결과이다.

프로파일 레코드의 수는 웹 어플리케이션이 생성해낸 웹 페이지의 링크를 따라 산출하게 된 것으로 총 분석된 로그 수에 비해 극히 적음을 알 수 있다. 또한, 요청 수가 늘어나도 프로파일의 수는 크게 증가하지 않게 된다. 그 이유는 프로파일의 식별자를 구성하는 방법에서 요청되는 어플리케이션과 어플리케이션에서 사용되는 파라미터 변수의 이름으로 식별되기 때문이다. 웹 어플리케이션은 이와 같이 코드 내에서 생성해내는 페이지가 정형화 되어 있기 때문에 프로파일 기법이 효과적임을 확인할 수 있다.

〈표 3〉 프로파일 레코드수

웹 어플리케이션	총 요청수	프로파일 레코드수	프로파일 비율(%)
Zeroboard	145,947	152	0.1
phpBB2	485	61	12.1
CrazyWebboard	2,076	6	0.28
awstats	861	13	1.5

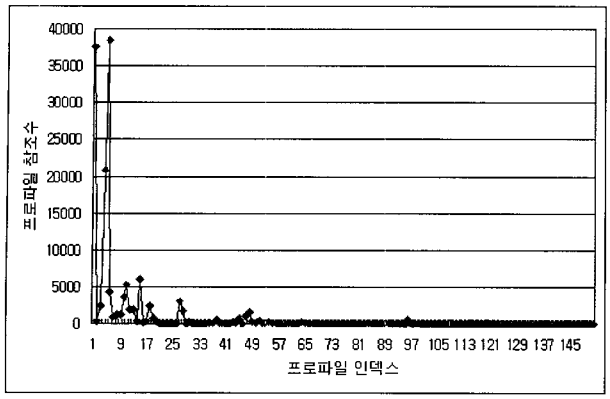
4.2 탐지 알고리즘 시험

시험에 사용된 데이터는 웹서버의 2005년 로그를 대상으로 하였다. 2005년 총 로그를 정상적인 로그와 비정상적인 로그를 분리하였다. 분리방법은 로그에서 원격 PHP 삽입 공격, 파일 노출공격, 내부변수 이용공격, Query 변수조작 공격, 시스템 명령 공격, 페이지가 없는 접속으로 분류할 수 있었다. 정상적인 로그의 크기는 20.7MB이며 요청 수는 145,947개 이고, 비정상 로그의 크기는 265KB이며 요청 수는 1,495개로 구성되었다.

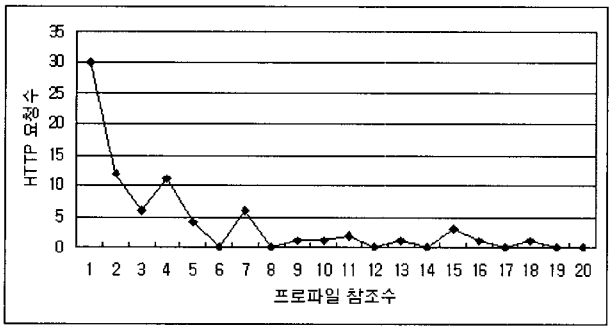
4.2.1 프로파일링 결과 분석

탐지 알고리즘에 대한 시험을 수행하기 위해 먼저 정상적인 로그에 대하여 프로파일링을 수행하여 정상 프로파일 레코드를 생성하였다. (그림 6)에서와 같이 프로파일의 수는 152개로 프로파일 식별자는 1에서부터 152개로 표현될 수 있으며 각 프로파일에 대한 참조 수는 최대 38,543번, 최소 1번 참조된 경우가 발생하였다.

정상 로그 중에서 한 번 참조된 프로파일 식별자는 (그림 7)에서와 같이 30개로 전체 요청수에서 0.02%를 차지하고 있다. 한 번 참조된 HTTP 요청을 이용하여 구성된 프로파일은 파라미터 타입과 길이에 의한 탐지방법에서 비교적 오탐율(false positive)을 높이는 결과를 가져오게 된다. 그 이유는 파라미터 타입과 길이에 대해 충분한 학습이 되지 않기 때문이다. 따라서, 이와 같이 참조수가 적은 프로파일 정보에 대해서는 관리자의 보정이 필요하게 된다.



(그림 6) 프로파일 인덱스에 대한 참조수



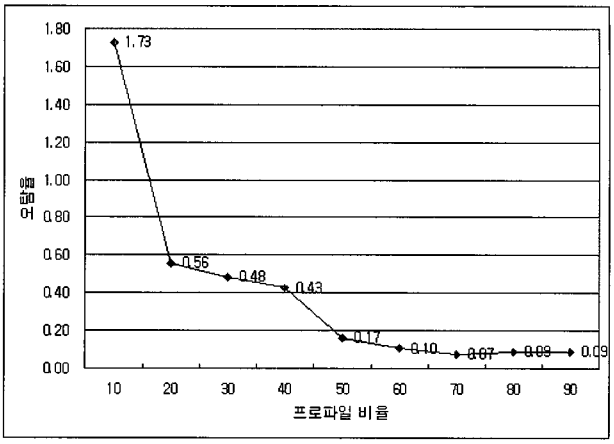
(그림 7) 프로파일 참조수에 대한 요청수

4.2.2 오탐율 시험 및 분석

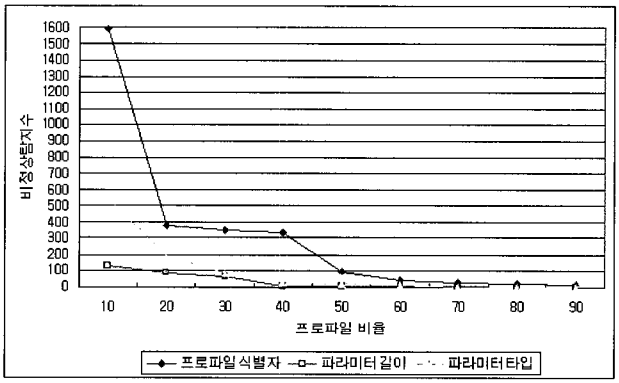
오탐율을 시험하기 위해 145,947개의 정상적인 로그를 이용하여 정상 요청을 비정상으로 판단하는 false positive 오탐율과 1,495개의 비정상적인 로그를 이용하여 비정상 요청을 정상으로 판단하는 false negative 오탐율을 측정하였다. 먼저, false positive 오탐율은 정상적인 로그로부터 전체 로그의 10%에서부터 90%의 크기를 프로파일 레코드 생성에 사용하고, 나머지 크기를 시험데이터 입력으로 사용하였다.

시험 결과 (그림 8)에서와 프로파일 비율이 증가함에 따라 false positive 오탐율이 급격히 감소하다가 50% 크기의 로그를 정상 프로파일로 구성하고 나머지를 테스트 데이터로 이용하는 경우 0.17%로 false positive 오탐율로 크게 감소하였다. (그림 9)에서는 본 논문에서 제시하는 탐지방식에 따른 비정상 탐지수를 보이고 있는데, 구현에서 프로파일 식별자에 따른 탐지, 파라미터 타입, 파라미터 길이에 의한 탐지에 의한 탐지의 우선순위를 가지고 비정상을 탐지하기 때문에 프로파일 식별자에 의한 탐지방법에 의해 비정상으로 분류된 경우가 가장 많았다. false positive 오탐율을 줄이기 위해서는 웹 어플리케이션에 대한 모든 링크에 대한 학습과 함께 길이 및 타입에 대한 관리자의 보정이 필요하다.

false negative 오탐율을 시험하기 위해서 정상 로그를 모두 프로파일링하고, <표 4>와 같은 공격으로 분류된 비정상 로그를 시험데이터로 활용하였다. 시험 결과 <표 5>에서와 같이 프로파일 식별자에 의한 탐지, 파라미터의 길이, 파라미터의 타입에 따라 false negative 오탐이 없이 모두 탐지가 되었다. 그 이유는, 비정상 시험데이터의 대부분이 원격 PHP 삽입을 이용한 복합적인 공격이 다수이기 때문에 대부분 프로파일 식별자에 의한 탐지방법에서 대부분 탐지가 되었기 때문이다.



(그림 8) 프로파일 비율에 따른 false positive 오탐율



(그림 9) 탐지방법별 분포

〈표 4〉 비정상 로그 수

공격유형	공격수
원격 PHP 삽입공격	142
파일노출 공격	15
내부변수 이용공격	656
Query 변수조작	20
시스템 명령공격	3
페이지가 없는 접속	659

〈표 5〉 탐지방법에 따른 분포

공격유형	프로파일 식별자	파라미터 길이	파라미터 타입
원격 PHP 삽입공격	139	0	3
파일노출 공격	14	0	1
내부변수 이용공격	656	0	0
Query 변수조작	0	20	0
시스템 명령공격	0	0	3
페이지가 없는 접속	659	0	0
탐지수	1468	20	7

5. 결 론

본 논문에서는 웜 형태로 까지 발전된 웹 어플리케이션에 대한 해킹위협에 대한 방어를 할 수 있는 프로파일 기반의 웹서버 보안 방법을 제시하고, 이에 대한 구현 및 시험을 수행하였다. 제시된 프로파일 기반의 웹서버 보안방식은 웹 서버에 의존적이지 않고, 각 웹 서버 사용자별 어플리케이션을 지원하는 동시에 어플리케이션 게이트웨이 방식으로 보안을 향상시킬 수 있도록 설계 및 구현되었다. 웹 어플리케이션 보호를 위한 방법으로 ModSecurity와 같은 웹서버 임베디드 모듈에서는 패턴 매칭에 의해 필터링을 수행하기 때문에 변형된 공격에 대해서는 탐지하기 어려운 단점을 가지고 있으나, 제시된 프로파일 식별자에 의한 탐지, 파라미터 길이, 파라미터 타입에 의한 탐지방법을 이용하게 되면 구축된 사이트의 특징을 반영할 수 있기 때문에 이상상태 탐지가 가능하게 된다. 또한, 웹 어플리케이션의 경우 정형화된 형식을 통해 지정된 페이지를 요청하도록 구성하고 있기 때문에 프로파일 기반의 탐지 방법이 효과적이라고 할 수 있다.

시험에서는 버그트랙에 보고된 4종의 각각 구현방법이 다른 웹 어플리케이션에 대해 프로파일링을 통해 생성된 레코드의 수를 확인함으로써 많은 사용자 요청을 프로파일링 하는데 문제가 없음을 확인하였다. 또한, 실제 웹서버에서의 로그를 가지고 오탐율을 시험함으로써 제시된 방법이 효과가 있음을 확인하였다.

향후 연구에서는 성능향상을 위해 ModSecurity에서와 같은 웹 서버에 임베디드된 모듈을 구현함으로써 성능을 개선하는 연구를 수행할 것이다.

참 고 문 헌

[1] http://www.theregister.co.uk/2004/12/21/santy_worm/

[2] http://isc.sans.org/diary.php?date=2005-11-05

[3] Jeffry R. Williams et al., "The Ten Most Critical Web Application Security Vulnerabilities", OWASP, 2004.

[4] Christoher Kruegel, Giobanni Vigna, William Robertson, "A multi-model approach to the detection of web-based attacks", Computer Networks : Vol.48, No.5, pp.717~738, August, 2005.

[5] Scott, D., Sharp. R, "Abstracting Application-Level Web Security", Proc. of the World Wide Web Conference, 2002.

[6] Y. W. Huang et al, "Securing Web Application Code by static Analysis and Runtime Protection", Proc. of the World Wide Web Conference, May, 2004.

[7] Michael Benedikt, Juliana Freire, Patri Godeproid, "VeriWeb : Automatically Testing Dynamic Web Sites", Proc. of the World Wide Web Conference. 2002.

[8] Ivan Ristic, "Web Intrusion Detection with Mod_Security", OWASP AppSec Europe, 2005.

[9] Mark Curphey, Joel Scambray, Erik Olson, "Improving Web Application Security : Threats and Countermeasures", Microsoft Corporation, 2003.

[10] Shreeraj Shah, "Defending Web Services using Mod Security(Apache)", NetSquare, 2004.

[11] Mark Curphey, David Endler, "A Guide to Building Secure Web Applications", OWASP, Sep., 2002.

[12] Robert Auger, Ryan Barnett, "Web Application Security Consortium : Threat Classification Version 1.0", Web Application Security Consortium(www.webappsec.org), 2004.

[13] Ory Segal, "Web Application Forensics : The unchatterd Territory", SANCTUM, 2002.

[14] Sverre H. Huseby, "Common Security Problems in the Code of Dynamic Web Applications", Web Application Security Consortium(www.webappsec.org), June, 2005.

[15] Gentoo Linux Security Advisory, http://www.gentoo.org.

[16] BugTraq, http://www.securityfocus.com/archive/1

[17] Awstats, "http://www.awstats.org"

[18] phpBB, "http://www.phpbb.com"

[19] CrazyWebBoard, "http://www.crazywebboard.com"

[20] zeroboard, "http://www.zeroboard.com"

[21] TeleportPro, "http://www.tenmax.com/telport/pro/home.htm"

윤 영 태

e-mail : ytyun@etri.re.kr

1995년 충남대학교 컴퓨터과학과(학사)

1997년 현대전자 정보시스템 사업본부

1999년 충남대학교 컴퓨터과학과(이학석사)

1999년~현재 국가보안기술연구소 선임연구원

관심분야 : 네트워크 정보보호, 보안운영체제

류 재 철

e-mail : jcryou@home.cnu.ac.kr

1985년 한양대학교 산업공학과(학사)

1988년 Iowa State University(전산학 석사)

1990년 Northwestern Univercity(전산학 박사)

1991년~현재 충남대학교 정보통신공학부 교수

관심분야 : 컴퓨터 및 통신망 보안, 전자상거래

박 상 서

e-mail : sangseo@etri.re.kr
1991년 중앙대학교 전자계산학과(공학사)
1993년 중앙대학교 전자계산학과(공학석사)
1996년 중앙대학교 전자계산학과(공학박사)
1996년~1998년 국방정보체계연구소 선임연구원
1999년~1998년 국방과학연구소 선임연구원
2000년~현재 국가보안기술연구소 선임연구원
관심분야 : 정보전

박 종 욱

e-mail : khspjw@etri.re.kr
1986년 경북대학교 전자공학과(공학사)
1988년 경북대학교 전자공학과(공학석사)
2004년 경북대학교 전자공학과(공학박사)
1988년~2000년 국방과학연구소 선임연구원
2000년~현재 국가보안기술연구소 책임연구원
관심분야 : 정보보호, 네트워크 통신