HTTP Header를 이용한 실시간 웹 공격 탐지 시각화 도구의 설계 및 구현

구본현, 조규형, 조상현, 문종섭

고려대학교 정보보호대학원/정보보호기술 연구센터

Real-Time Web Attack Detection Visualization Tool Design and Implementation using HTTP Header Information

Bonhyun Koo, Kyuhyung Cho, Sanghyun Cho, Jongsub Moon

GSIS/CIST, Korea University

요 약

본 논문에서는 HTTP 요청, 응답 헤더정보 분석을 통해, 실시간으로 웹 공격을 탐지하는 시각화도구를 제안한다. 공격 탐지기법은 이상, 오용 탐지 기법을 통합한 방식이다. 이상 탐지는 헤더정보의 Refer와 Uri 필드를 이용한 베이지언 분포를 통한 확률 값을 이용하였으며, 오용탐지는 Snort의 공격 시그너처의 웹 공격부분을 사용하였다. 공격 탐지 정보의 효율적인 전달을 위해, 시각화를 GUI로 구현하였다. 본 논문에서는 사용자 에이전트의 비정상 행위 감시, 빈도 분석, 공격 에이전트 위치추적을 실시간으로 시각화하여 표현하는 기법을 제안한다.

I. 서론

DoS/DDoS와 같은 네트워크를 파괴시키는 공격은 크게 줄어들고 있는 추세다. 하지만 OWASP 10대 취약점 공격기법상의 SQLInjection, Cross-site-scripting(XSS) 등을 통해 웹 애플리케이션을 노리는 다양한 공격은 급증하고 있다. 웹어플리케이션의 해킹을 통해 사용자 로그인이나 결제, 계좌 체크 정보 등 웹 애플리케이션에 저장돼 있는 데이터를 훔쳐내 금전적인 이득을 노리는 경우가 많다. 이는 개인이나 기업에게 치명적인 피해를 초래한다. 이러한 웹 어플리케이션에 대한 공격을 방어하기 위해, 본 논문에서는 실시간으로 공격을 탐지하고, 시각화 구현을 통해 효율적인 탐지 인터페이스를 제공하는 도구(WADS : Web Anomaly Detection Station 이하 WADS)를 소개한다.

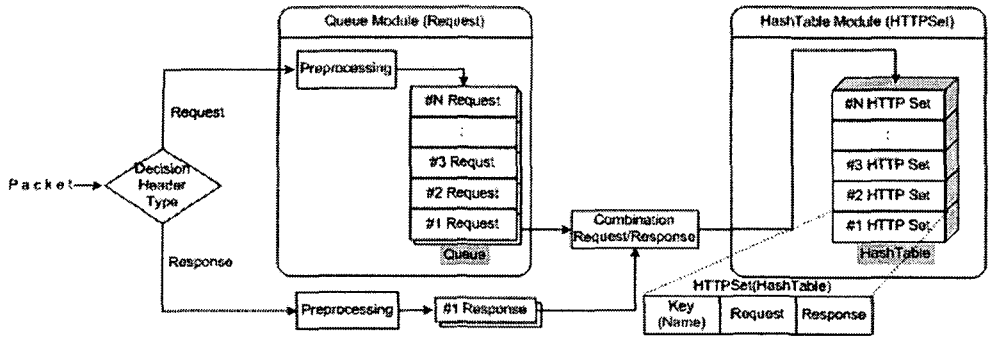
* 주저자 : koo191@korea.ac.kr
** 교신저자 : jsmoon@korea.ac.kr
본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터(ITRC) 육성·지원사업의 연구결과로 수행되었음.

II. 관련 연구

그림1은 CodeRedV2 웜의 명령어 실행파일로의 접근에 대한 Snort 탐지 시그너처이다. Snort와 같은 시그너처의 일치여부를 판단하여 공격을 탐지하는 오용탐지기반 침입탐지 시스템은 다음과 같은 문제점이 있다. 첫째, 새로운 공격 패턴이 발생한 경우 이를 탐지해내지 못한다. 둘째, 공격 패턴의 정의에 따른 저장 공간이 지속적으로 늘어난다. 여기에 공격DB로부터 공격 패턴을 불러오는 신속한 알고리즘의 구현 역시 필수적으로 구현되어야 한다. 이와 같은 오용 탐지 기반침입탐지시스템의 문제점

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (
        msg:"WEB-IIS CodeRed v2 root.exe access";
        flow:to_server,established; uricontent:"/root.exe"; nocase;
        classtype:web-application-attack; sid:1256; rev:8;)
```

<그림 1> CodeRedV2웜 취약점 관련 Snort 시그너처



<그림 5> HTTP헤더 해시테이블 설계구조

을 보완하기 위해, 본 논문에서 제안하는 베이지언 추론과 같은 알고리즘을 통해, 다양한 이상 탐지 알고리즘이 연구되어지고 있다[2].

III. 실시간 공격탐지 기법

본 절에서는 공격탐지를 실시간을 처리하기 위해 설계한 시스템의 구조에 대해 설명한다.

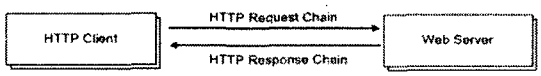
1. HTTP 프로토콜 구조

HTTP 프로토콜은 그림2와같이 Request / Response 방식을 이용하여 동작한다. 즉, GET, HEAD, POST 와 같이 원하는 프로토콜 기능에 대해 서비스 요구를 하면 데이터 송수신을 위한 TCP 연결이 만들어지고, 서버가 응답을 보내어 데이터 전송을 끝내면 자동적으로 연결이 끊어지게 되는 것이다.

HTTP의 헤더의 세부 내용은 아래 그림3과 같다. 그림3은 요청헤더에 대한 정보를 보여주고 있다. Method, Version, URI, Reffer 등을 얻을 수 있다. 반면 응답헤더는 요청에 대한 Status Code, Server Type 등을 얻을 수 있다.

2. 실시간 HTTP 패킷 헤더 분석

HTTP 프로토콜의 실시간 분석을 위해서는 다음과 같은 세 가지 문제점을 해결할 수 있어야 한다.



<그림 2> HTTP 연결 설정 과정

① HTTP 헤더의 어떠한 형태로 사용할 것인가?

② TCP/IP 프로토콜상의 지연으로 인한 결합문제

③ 복수의 클라이언트 구조를 어떻게 처리할 것인가?

문제①을 해결하기 위해, W3C의 웹서버 로그 포맷 구성 형태를 구성하였다. 즉, 패킷의 아스키코드를 파싱하여, 그림4와 같은 자료구조(Log) 형태를 만들어 분석에 이용하였다. 이를 통해 HTTP헤더의 정보를 로컬상의 일반적인 웹서버 로그 파일을 읽어오는 것과 동일한 결과를 가져오도록 하였다. 문제②를 해결하기 위해서 다음과 같은 방식을 설계하였다. HTTP 프로토콜은 요청과 이에 대한 응답 구조를 가지고 있다. TCP 프로토콜의 특성상 요청에 대한 해당 응답을 순서대로 주고받지 못한다. 이를 위해 그림5와 같이 요청 패킷에 대해서는 큐를 생성하여, 패킷을 버퍼에 저장하도록 한다. 서버로부터 응답패킷이 발생하면(그림5의 HTTP Response), 요청 큐로부터 순서대로 결합하여 해시테이블 구조로 데이터를 만들게 된다. 그림4의 Log의 데이터 구성은 HTTP의 요청과 응답 2개의 배열리스트를 결합하는 구조를 통해 생성되어진다.

```
Hypertext Transfer Protocol
⊟ GET /Sa      /index.htm HTTP/1.1\r\n
    Request Method: GET
    Request URI: /Sa      /index.htm
    Request Version: HTTP/1.1
  Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, appl
  Referer: http://netsec.        .kr/      ng/Features/index.htm\r\n
  Accept-Language: ko\r\n
  UA-CPU: x86\r\n
  Accept-Encoding: gzip, deflate\r\n
  If-Modified-Since: Mon, 23 Jan 2006 18:58:27 GMT\r\n
  If-None-Match: "ea8b86fe4e20c61:283"\r\n
  User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.2; SV
  Host: netsec.korea.ac.kr\r\n
  Connection: Keep-Alive\r\n
  \r\n
```

<그림 3> HTTP 요청 헤더 정보

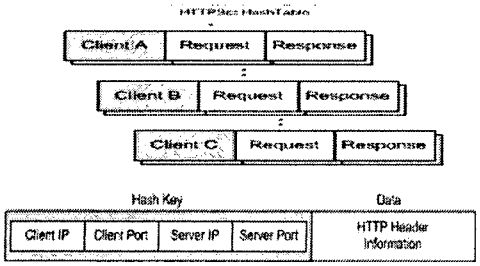
```
Struct Log                                  Enum HttpMethod
{                                           {
        DateTime        time,                       GET,
        IPAddress       clientIp,                   HEAD,
        int             clientPort,                 POST,
        HttpMethod      method,                     OPTIONS,
        String          host,                       PUT,
        String          uriStem,                    DELETE,
        String          uriQuery,                   TRACE,
        String          postQuery,                  ETC
        int             status,             }
        String          referer,
        String          contentType,
}
```

<그림 4> Log 자료구조

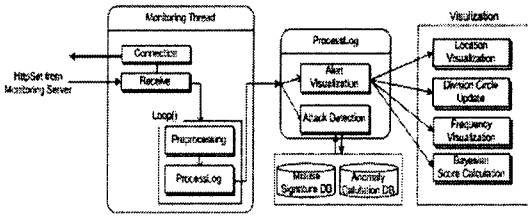
문제 ②를 처리함으로써, Log의 데이터를 완전하게 생성할 수 있게 된다. 문제③은 웹서버 한 대에 다수의 클라이언트가 접속할 수 있는 구조로 인해, 웹서버로의 요청 패킷이 중복되어지는 문제가 발생하기 때문이다. 이 문제를 해결하기위해 해시테이블의 구조를 이용한다. 그림6의 HTTPSet과 같은 해시테이블은 키와 데이터의 결합구조를 가진다. 해쉬키의 설정을 위해 Name 클래스를 이용하여, 그림6과같이 고유키 값을 클라이언트의 IP주소와 포트번호, 서버의 IP주소와 포트번호를 이용하여 생성하였다. 문제점①,②,③을 처리하고, 감시 시스템으로 정보를 전송해주는 프로세스의 구조는 2개의 쓰레드를 통해 작동한다. 웹서버의 이더넷 인터페이스로부터 발생하는 패킷을 분석하는 쓰레드와 분석을 통해 로그형태로 생성된 HTTPSet 정보를 공격감시시스템으로 전송하는 쓰레드가 멀티쓰레드로 작동하는 프로세스를 가지고 있다. 이 프로세스에 의해 공격 감시시스템은 로그파일의 처리와 동일한 과정을 수행하게 된다.

IV. WADS (Web Anomaly Detection Station)

이번 절에서는 본 논문에서 제안하는 WADS 시스템의 설계 내용에 대해 설명한다.



<그림 6> 클라이언트 식별을 위한 해쉬키 설정



<그림 7> WADS 시스템 구조

1. 시스템 구성

WADS의 시스템은 크게 2가지 프로세스로 구성되어진다. 4절에서 설명한 HTTP 헤더를 파싱하고 이를 분석 시스템으로 전송해주는 프로세스와 이를 받아 이상 값을 계산하고 시각화를 하는 프로세스로 구성되어진다. 세부적인 시스템 구성사항은 그림7과 같다. 분석 엔진은 빈도를 분석하고 이상 값을 계산하여 처리하며, 공격 탐지를 위해 기존의 공격패턴을 탐지해낸다. 공격 발생 시에는 이를 시각화로 나타내는 프로세스가 작동하여, 공격자의 역추적 정보 표현, 그래프를 통한 빈도 분석, 사용자의 행위를 GUI로 표현한다.

2. 공격 탐지 및 역추적 기법

본 연구에서는 베이지언 추정 기법을 이용하여, 기존에 발생한 이벤트와 발생하지 않은 이벤트에 따라서 새로운 이벤트에 대한 발생 확률을 이전 분포에 근거하여 추정하는 알고리즘을 사용하였다. 기존의 Uri필드의 정보와 Reffer필드 쌍을 이용하여, 새로운 HTTP 패킷 발생 시, 다음과 같은 방식으로 이상 확률 값을 계산한다.

$$P(X=i)=\frac{C(N_i+\alpha)}{(K\alpha+N)} \quad \left(C=\frac{N}{N+L-K} \right)$$

[표 1] 변수 정의

변 수	정 의
X	Reffer 값
N	Reffer필드의 총 빈도수
N_i	Symbol 빈도수
K	Symbol의 경우의 수
L	발생 가능한 전체 Symbol 수
α	의사 결정 값

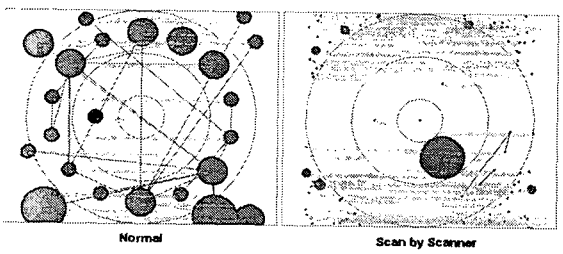
(※ Symbol : {Reffer, Uri} 쌍을 지칭)

WADS에서는 공격 패턴의 일치를 위해 Snort의 탐지 룰의 웹 공격 패턴을 DB화하여 이를 일치하는 방식으로 탐지한다. 감시 프로세스로부터 받은 로그형식(Log)의 데이터 중 PostQuery와 UriQuery필드를 비교한다. 이를 통해 이상탐지와 오용탐지 2가지 탐지기법을 통합하는 탐지기법으로 운용되게 된다.

공격자의 위치를 역추적하기 위해 IP주소의 DNS 값을 이용한다. DNS쿼리 정보를 통해 지역정보를 얻고, 이를 위도와 경도 정보를 반환하여, 구글맵 API에 변수로 입력하여 위성정보를 시각화하여 보여준다.

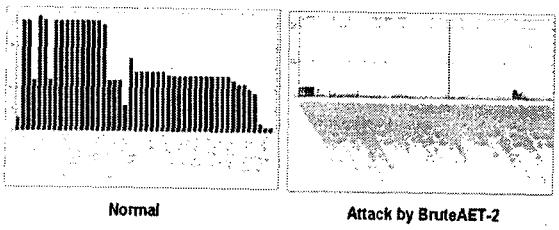
3. 시각화 인터페이스 구현

WADS 시스템의 구현 언어로는 닷넷 기반의 C# 프로그래밍 언어를 이용하였다. Divide Circle은 웹 토폴로지상의 하부 층으로 내려갈수록 분할원을 추가하여 표현하는 시각화 기법이다[3]. 서버에 대한 페이지 요청 시 새로운 원(이하 RC)이 생성되어진다. RC는 요청 페이지에 대한 빈도수를 나타낸다. 웹서버의 처음 접속 시에는 분할원의 중심에 가깝지만, 세부 페이지로 내려갈수록 분할원의 바깥쪽에 위치하는 표현 방식이다. 만약 정상적인 요청이 아니라면, 분할원의 바깥쪽에 위치하게 된다. 페이지에 대한 요청 빈도가 늘어날수록 해당 RC의 반경은 커지게 된다. 헤더 정보가 들어오면 Reffer 필드는 녹색RC, Uri필드는 붉은색RC로 색상을 표현하며, 붉은색→초록색→파란색의 링크 구조를 나타낸다. Nikto 웹취약점스캐너를 이용한 공격 테스트 결과는 다음 그림8과 같다.

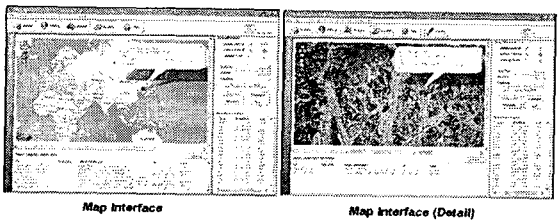


<그림 8> Nikto 웹취약점 스캔공격 시각화

좌측의 정상적인 패턴에 비해 우측의 공격패턴은 분할원 외부의 존재하지 않는 페이지 요청을 확인할 수 있다. Brute-AET2를 이용한 공격 테스트 화면은 그림9의 우측 그림과 같이 특정페이지가 급격하게 높은 비율을 차지하는 것을 확인할 수 있다. 그림10은 공격자의 위치를 역추적하여, 시각화한 인터페이스 화면이다.



<그림 9> 패스워드 무차별 대입공격 시각화



<그림 10> 공격자 역추적 시각화

V.결 론

본 논문에서는 웹서버로의 요청과 응답 구조를 가지는 HTTP 패킷의 헤더를 실시간으로 분석하고, 이를 통해 웹 공격을 탐지하고, 시각화하는 기법을 제안하였다. 해시테이블 구조를 헤더의 분석과 이상 값의 계산은 DB서버를 이용한 기존 시스템에 비해 속도와 성능 면에서 보다 좋은 결과가 기대된다.

[참고문헌]

[1] David Moore, Vern Paxson, Stefan Savage, Colleen Shannon, Stuart Staniford,& Nicholas Weaver. : The Spread of the Sapphire/ Slammer Worm. http://www.cs.berkeley.edu/~nweaver/sapphire/

[2]Chen,Z., Gao, L., Kwiat, K.: Modeling the spread of active worms. In: Proceedings of IEEE INFOCOM 2003.

[3] Giuseppe Serazzi and Stefano Zanero : Computer