Journal of The Korea Society of Computer and Information
Vol. 21 No. 7, pp. 1-8, July 2016

www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2016.21.7.001

Detecting the HTTP-GET Flood Attacks Based on the Access Behavior of Inline Objects in a Web-page Using NetFlow Data

Koo-Hong Kang*

Abstract

Nowadays, distributed denial of service (DDoS) attacks on web sites reward attackers financially or politically because our daily lifes tightly depends on web services such as on-line banking, e-mail, and e-commerce. One of DDoS attacks to web servers is called HTTP-GET flood attack which is becoming more serious. Most existing techniques are running on the application layer because these attack packets use legitimate network protocols and HTTP payloads; that is, network-level intrusion detection systems cannot distinguish legitimate HTTP-GET requests and malicious requests. In this paper, we propose a practical detection technique against HTTP-GET flood attacks, based on the access behavior of inline objects in a webpage using NetFlow data. In particular, our proposed scheme is working on the network layer without any application-specific deep packet inspections. We implement the proposed detection technique and evaluate the ability of attack detection on a simple test environment using NetBot attacker. Moreover, we also show that our approach must be applicable to real field by showing the test profile captured on a well-known e-commerce site. The results show that our technique can detect the HTTP-GET flood attack effectively.

▸ Keyword : HTTP-GET flood attack, Internet security, NetFlow

I. Introduction

인터넷이 현대 사회에 미치는 영향은 가히 상상을 초월한다. 이러한 영향력은 국가, 기업, 그리고 개인의 생활 깊숙이 파고들어 있다. 즉 국가와 기업은 인터넷을 이용한 정보 인프라를 기반으로 다양한 서비스들을 국민들에게 제공하고 개인들은 언제 어디에서나 자신의 컴퓨터 혹은 모바일 단말기를 통해 해당 서비스를 이용하고자 한다. 따라서 인터넷 중단 사태는 과거와는 달리 단순한 일시적 사건 사고가 아니라 국가 안보 차원에서 다루어지고 있는 실정이다.

해커들이 사이버 공격을 감행하는 이유는 매우 다양하다. 단순한 호기심에서부터 금전적 이득, 그리고 사회적 혼란을 초래하는 사이버 테러까지 매우 다양한 양상을 보인다. 해커들은 자신의 공격 효과를 극대화하기 위해 서비스 이용 빈도가 높은 주요 공공 서비스를 제공하는 웹 서버를 1차 공격목표로 삼고 있다. 즉 국가 주요 기관, 방송사, 대형 포털, 그리고 금융기관들이 운영하는 주요 서버들이 해커의 1차 공격 목표(target)가 되고 있다. 따라서 이러한 중요 서버들이 해커들의 공격으로부터 안전하게 유지되고 서비스된다면 적어도 사회적 혼란만큼은 일어나지 않을 것으로 판단된다.

최근 특정 웹서버를 대상으로 다량의 HTTP-GET 요구(request) 패킷을 발생시켜 웹서버에 과부하를 유발시킴으로서 일반 사용자들이 정상적으로 해당 웹서버에 접근하는 것을 막아 버리는 분산서비스거부(Distributed Denial of Service: DDoS) 공격이 빈번하게 발생되고 있다. HTTP-GET flood 공격은 바이러스에 감염된 클라이언트 컴퓨터에 의해 발생되거나 혹은 봇(Bot) 마스트에 의해 다이나믹하게 조정되는 좀비(zombie) 컴퓨터에 의해 발생되며 그 피해수준은 매우 심각한

• First Author: Koo-Hong Kang, Corresponding Author: Koo-Hong Kang
*Koo-Hong Kang(khkang@seowon.ac.kr), Dept. of Information and Communications Engineering, Seowon University
• Received: 2016. 05. 09, Revised: 2016. 05. 30, Accepted: 2016. 07. 12.

수준에 이르고 있다[1,2]. 그러나 해커들은 정상적인 TCP 연결과정을 거친 후 정상적인 HTTP 페이로드(payload)를 사용하여 이러한 공격을 감행하기 때문에 침입탐지시스템(Intrusion Detection System: IDS)들은 일반 사용자에 의한 HTTP-GET 요구와 이들 해커들이 발생시키는 악의적인 HTTP-GET 요구를 구별해 내기가 쉽지 않다. 즉 단순히 시그너처(공격 패턴)를 이용해 검출을 시도하는 IDS는 사실상 HTTP-GET flood 공격을 검출하기 불가능하다[3]. 그러나 국내외 많은 보안장비들은 HTTP GET 요청 횟수와 임계치(threshold) 값을 모니터링하여 비정상적으로 많은 트래픽을 발생시키는 컴퓨터를 차단하는 방법을 여전히 사용하고 있다. 불행하게도 많은 좀비 PC를 이용하는 DDoS 공격 유형에서는 하나의 좀비 PC에서 발생시키는 HTTP GET 요청 트래픽이 설정된 임계치 값보다 작게 가져갈 수 있어 사실상 이들 slow rate 공격에 대한 검출이 불가능해 질 수 있다. 따라서 HTTP 웹 트랜젝션(transaction)을 일일이 검증할 수 있는 기능을 포함하는 장비들이 속속 출시되고 있다[4,5]. 그러나 이들 보안장비들이 HTTP 웹 트랜젝션을 어떻게 구체적으로 검증하는지는 자세히 밝히지 않고 있다.

본 논문에서는 NetFlow가 제공하는 정보를 기반으로 HTTP GET flood 공격을 검출하는 방법을 제안한다. NetFlow[6]는 네트워크 설계 및 과금(accounting) 등을 목적으로 C사에 의해 최초 제안되었지만 최근에는 인터넷 보안과 관련된 많은 연구들이 진행되고 있다. 특히 NetFlow가 표준화[7] 과정을 거치고 있으며 C사 뿐만 아니라 대부분의 네트워크 장비회사가 판매하는 라우터 및 스위치들이 NetFlow 정보를 제공하고 있기 때문에 NetFlow 정보를 이용한 보안장비는 매우 큰 장점을 가진다. 즉 실제 네트워크에 트래픽 모니터링을 위한 새로운 장비를 설치하지 않아도 된다. 본 논문에서는 단순히 임계치를 기반으로 HTTP GET flood 공격을 검출할 수 없는 공격 시나리오에서도 NetFlow 정보를 기반으로 HTTP 웹 트랜젝션의 유효성을 검사할 수 있는 네트워크 기반 검출 알고리즘을 제안하였다. 특히 실제 대형 웹서버를 대상으로 관련 NetFlow 정보를 획득하고 이를 기반으로 실험실 수준의 공격과 방어를 시도함으로써 제안 알고리즘을 검증하였다.

서론에 이어, 제2장에서는 HTTP-GET flood 공격 검출을 위한 기존 방법에 대해 간략히 언급하고, 제3장에서는 최근 NetFlow 정보를 활용한 인터넷 보안 솔루션 및 NetFlow 기술에 대해 설명하였다. 제4장에서는 HTTP-GET flood 공격 검출을 위한 새로운 알고리즘을 제안하고 잘 알려진 해외 온라인 쇼핑몰을 대상으로 알고리즘 검증을 위한 기본적인 근거 자료를 제시하였다. 제5장에서는 NetBot Attacker 툴을 이용해 간단한 테스트 베드를 구성하여 제안된 알고리즘의 타당성을 실험을 통해 검증하였다. 마지막으로 제6장에서 결론 및 제안된 기법의 장점과 단점을 기술하였다.

II. Related Works

지난 수 년 간 HTTP GET flood 공격을 검출하기 위해 매우 다양한 방법들이 제안되었다. 그러나 서론에서 언급한 바와 같이 단순히 HTTP GET 요청 횟수의 임계치를 기반으로 공격을 검출하는 방법들은 더 이상 유효하지 않을 수 있다. 따라서 본 절에서는 단순 임계치 기반 방법을 제외한 기존 HTTP GET flood 공격 검출 방법들에 대해서만 간략히 언급하도록 한다.

바이러스나 봇(bot)에 감염된 클라이언트들은 목표 서버에서 동일한 순서로 웹 페이지를 브라우징하는 경향이 있다. Yatagai et al[1]는 이와 같이 계속적으로 동일한 순서로 웹 페이지들을 브라우징하는 발신지 IP 주소를 검출하였다. 또한 Yatagai et al[1]는 일반 사용자의 경우 웹 페이지 내에 포함된 문자, 이미지, 그리고 링크들과 같은 정보의 양에 비례하여 브라우징 시간이 증가한다고 판단하였다. 반면 HTTP GET flood 공격은 웹페이지 내 정보의 양과 무관하게 일정한 브라우징 시간을 유지했다. 따라서 이들은 웹페이지 내 정보의 양과 브라우징 시간의 상관관계를 이용하여 공격을 검출하였다.

한편 오늘날 웹 페이지들은 대부분 로고와 이미지 등과 같은 인라인 오브젝트(inline objects)들을 포함하는 멀티미디어 문서로 구성된다. 따라서 사용자들이 하나의 웹 문서를 요청하면 브라우저는 해당 메인페이지와 함께 자동적으로 인라인 오브젝트를 요청하게 된다. 즉 브라우저는 메인페이지를 수신하면 파싱(parsing)을 통해 페이지 내에 포함된 인라인 오브젝트를 받기 위해 요청을 하게 된다[8]. Lu와 Yu[9]는 일반 사용자들이 요청하는 이들 두 종류의 요청(웹페이지 및 인라인 오브젝트)이 HTTP GET flood 공격이 발생 시키는 요청 이벤트와는 전혀 다른 패턴을 보이는 것을 이용하여 공격을 검출하였다. 그러나 Lu와 Yu[9]는 HTTP GET 요청을 사용자별로 구분하고, 요청 타입별로 구별하며, 그리고 이들 요청들이 정상분포를 따르는지 판단하는 과정이 지나치게 많은 시간이 소모되는 것을 해결하기 위해 단순히 각 사용자별 요청 시간 간격만을 고려하여 공격을 검출하였다. 한편 Choi et al[10]는 사용자들이 메인 페이지를 요청하는 정확한 횟수를 카운트하기 위해 이들 인라인 오브젝트를 요청을 제외하였으며, 따라서 사용자별 정확한 메인 웹페이지 요청 시간 간격을 기준으로 HTTP GET flood 공격을 검출하였다.

Chwalinkski et al[3]는 일반 사용자들의 경우 웹 사이트에 저장된 전체 오브젝트들 중 약 10% 수준의 오브젝트들만 자주 사용한다는 사실을 이용하였다. 즉 해당 웹사이트의 최근 관심 오브젝트를 클러스터링 방법으로 그룹화하고 각 클러스터에 대한 likelihood 값을 학습 행렬에 저장하였다. 이후, 임의의 호스트로부터 해당 웹사이트의 임의의 오브젝트에 대한 요청이 있을 때, 이 세션에 해당하는 클러스터를 할당하고 likelihood 값을 계산하여 해당 클러스터 행렬의 엘리먼트 값보다 작으면 공

격으로 판단한다. Srivatsa et al[11]은 웹 서버에 동시에 서비스되는 클라이언트 수를 제한하는 연결수락제어(admission control) 기법을 제안하고 허락된 클라이언트들에게 우선순위를 부여하여 높은 우선순위를 가진 클라이언트들에게는 더욱 많은 자원을 할당하는 혼잡제어(congestion control) 기법을 제안하였다. 그러나 이들이 제안한 기법은 매우 복잡한 기능을 수행하는 웹 서버를 요구하고 있다.

정상적인 네트워크 계층 프로토콜을 사용하는 대부분의 응용계층 DDoS 공격이 그렇듯이, HTTP GET flood 공격 역시 쉽게 검출할 수 있는 수준의 공격 유형이 아니다. 다시 말해, 정상적인 사용자의 행위와 매우 느린 속도로 지속적으로 공격을 시도하는 대량의 좀비 호스트를 사용하는 HTTP GET flood 공격을 구별해 내는 작업은 현재 우리가 직면하고 있는 해결하기 가장 어려운 문제들 중 하나임에 틀림없다. 본 절에서 설명한 다섯 가지 방법 이외에도 지난 수년간 많은 방법들이 제안되었지만 실제 시스템에 적용되어 활용되는 예를 찾아보기는 쉽지 않다. 그 이유는 여러 가지 있겠지만, 응용계층에서 이루어지는 방어 체계인 만큼 시나리오별 case-by-case 로 연구가 이루어지고 있기 때문이다. 본 논문에서는 비록 HTTP GET flood 공격이 응용계층에서 이루어지는 DDoS 공격이지만 새로이 제안될 검출 알고리즘이 보다 현실적으로 실제 인터넷 망에 쉽게 구현 가능하도록 네트워크 레벨로 가져왔다. 즉 대부분의 라우터 및 스위치가 제공하는 NetFlow 정보를 활용하여 HTTP GET flood 공격을 네트워크 레벨에서 검출할 수 있는 방법을 제안한다. 뿐만 아니라, 대부분의 네트워크 레벨 방어체계가 지향하는 단순 임계치 기반 기법이 아닌, 사용자별 웹 서버 접근 플로우 행위 기반 방법을 제안함으로써 지속적으로 느린 속도로 공격을 감행하는 좀비 호스트를 검출하고자 한다.

III. Uses of NetFlow Data for Security

트래픽 모니터링을 통한 네트워크 설계 및 사용자 과금 등에 사용되었던 NetFlow 정보는 최근 인터넷 보안과 관련된 많은 연구 논문 및 상용 제품들이 발표되고 있다. 이러한 추세는 네트워크 장비(라우터 및 스위치)들이 NetFlow 혹은 NetFlow와 유사한 플로우 정보를 제공함으로써 이들 정보를 활용한 네트워크 모니터링 수단들이 실제 네트워크에서 활발히 사용되고 있기 때문이다. 특히 NetFlow는 사용자들의 콘텐츠를 직접 보지 않으면서 사용자의 제한적 정보만을 모니터링하기 때문에 개인정보에 매우 애민한 상황들을 일부 극복할 수 있는 이점이 있다. 특히 IETF IPFIX 워킹 그룹은 NetFlow 버전9를 기반으로 IP 플로우 정보 제공을 위한 표준화 작업을 완료하였다. 따라서 NetFlow 정보를 활용한 인터넷 보안에 대한 보다 다양한 솔루션들이 활발하게 개발될 것으로 예상된다.

Chen[12]은 플로우 레벨 정보를 활용하여 DoS 공격를 시도하는 공격 근원지를 검출하는 방법을 제안하였다. Yin[13]은 NetFlow 정보를 시각화하는 방법을 사용하여 상황인지(situational awareness)를 제공하였다. 한편 Huistra[14]는 공격의 요청 및 응답 플로우 정보를 사용하여 DNS reflection 공격을 검출하는 방법을 제안하였으며, Bilge[15]는 플로우 크기, 클라이언트 액세스 패턴, 그리고 temporal 행위 등을 NetFlow 분석을 통해 봇넷(botnet)을 검출하는 방법을 제시하였다. 이와 같은 NetFlow 정보를 이용한 인터넷 보안 분야의 연구는 정상적인 트래픽으로부터 악의적인 트래픽을 구별하기 위한 네트워크 어노멀리(anomaly) 검출 분야에 있어서도 다양한 연구 논문들이 발표되었다.

NetFlow 레코드(record) 정보는 두 호스트 사이 단방향 점-대-점 트랜잭션 정보를 제공하며 패킷 레벨보다 한 단계 높은 플로우 정보를 제공한다. NetFlow는 하나의 플로우가 종료되기 위해 다음과 같은 네 가지 규칙을 적용한다. (i) 비활성 타이머(inactive timeout) - 플로우의 패킷 사이 시간(inter-packet time)이 설정된 NetFlow 비활성 시간을 초과하는 경우, (ii) 활성 타이머(active timeout) - 플로우의 최초 패킷이 도착한 이후 플로우가 설정된 활성 시간을 초과하는 경우, (iii) TCP 연결이 종료 - TCP 플래그가 FIN 혹은 RST인 경우, (iv) 메모리 관리 - 플로우 캐시(cache)가 가득 차는 경우. NetFlow가 활성화된 라우터(혹은 스위치)는 종료된 플로우 정보(NetFlow records)를 UDP 데이터그램을 사용하여 NetFlow 수집기로 전송한다. 표 1은 NetFlow 레코드의 주요 필드 정보를 보여준다. 따라서 NetFlow 레코드 정보를 통해 해당 플로우의 5-tuple 정보(발신지 및 수신지 IP 주소, 발신지 및 수신지 포트 번호, 그리고 IP 프로토콜 타입)와 해당 플로우의 시작 및 종료 시간, 그리고 해당 플로우 내 포함된 패킷 수 및 전체 바이트 수를 확인할 수 있다[6].

한편, NetFlow를 인터넷 보안에 적용하기 위해서는 매우 심각한 문제점이 존재한다. 즉 라우터 혹은 스위치가 NetFlow를 활성화하면 프로세싱 부하 및 메모리 사용량이 증대되어 실제 이들 장비들의 원래의 역할인 라우팅 및 스위칭 능력에 문제가 발생될 수 있다. 이러한 문제점을 해결하기 위해 샘플링 NetFlow를 동작시킨다. 즉 N 개의 패킷 중 1개를 샘플링하도록 설정하며 현장에서는 라우터의 종류 및 처리하는 트래픽 양을 기준으로 N을 10, 100, 그리고 1,000으로 설정하는 것으로 보고되고 있다. 이들 샘플링 NetFlow에 관한 관련 연구 역시 활발하게 진행되고 있다[16]. 그러나 본 논문에서는 no-sample NetFlow, 즉 전수 조사를 가정한다(이 가정에 대해 제6장 결론에서 다시 언급한다).

Table 1. Some important fields of NetFlow record

Field (no. of bytes)	Description
srcaddr(4), dstaddr(4)	source and destination IP addresses
dPkts (4)	packets in the flow
dOctets (4)	total number of layer 3 bytes in the packets of the flow
first(4), last(4)	SysUptime at start and end of low
srcport(2), dstport(2)	source and destination port numbers
proto(1)	IP protocol type

IV. The Proposed Scheme

1. Basic idea

앞에서 언급한 바와 같이 해커들의 주요 공격 목표가 되고 있는 대부분의 포털(portal)이나 온라인 쇼핑몰 등은 각종 로고와 이미지, 그리고 동영상 등을 제공하기 위한 인라인 오브젝트들을 포함하는 멀티미디어 웹 문서로 구성되어 있다[17]. 따라서 그림 1에서 보듯이, 일반 사용자들이 해당 웹문서에 접근하게 되면 이들 두 종류의 요청, 즉 웹페이지 및 해당 인라인 오브젝트를 액세스하게 된다. 그러나 이들 두 종류의 요청은 전혀 다른 트래픽 성질을 나타낸다. 왜냐하면, 웹페이지 요청은 사용자들에 의해 시작되지만 인라인 오브젝트는 브라우저에 의해 자동적으로 시도되기 때문이다 [8,9].

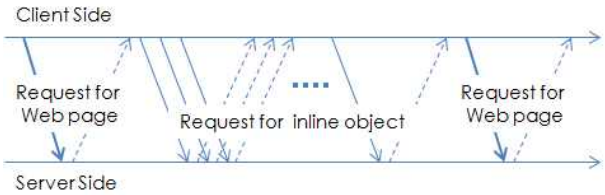

Fig. 1. Process of requesting Web documents[9]

HTTP-GET flood 공격은 응용계층 공격 중 하나로써 수 천 개 혹은 수 만개의 좀비 호스트로 구성된 봇넷(botnet)을 이용하여 공격 목표 웹서버로 정상적인 HTTP GET 요청 메시지를 끊임없이 전송하여 해당 서버의 자원들(CPU 부하, 메모리 사용 등)을 고갈시켜서 일반 사용자들이 정상적으로 웹서버에 접근하지 못하도록 한다. 최근 연구[1,9,18]에 따르면 많은 봇(bot) 들이 정상적인 브라우저와 유사하게 동작함으로써 보다 정교한 HTTP-GET flood 공격을 가능하게 한다. 그러나 실제 인터넷에서 발생되고 있는 많은 HTTP-GET flood 공격들은 여전히 다양한 사이버 공격 툴을 이용해 만들어진 바이너리 파일들을 사용하는 것으로 보고되고 있다[19]. 즉 네트워크 혹은 인터넷에 대한 전문 지식이 전혀 없는 일반인들도 이들 공격 툴들을 사용하여 손쉽게 DDoS를 시도하는 것이 오늘의 현실적 문제로 대두되고 있다. 그러나 일반인들이 쉽게 접할 수 있는 이들 공격 툴들은 최근 연구되고 있는 매우 정교한 형태의 HTTP-GET flood 공격 패턴을 보이지는 않는다. 다시 말하면, 그림 1에서 설명한 바와 같이 일반 사용자들이 사용하는 웹브라우저는 웹페이지를 수신하여 파싱(parsing)과정을 거쳐 해당 인라인 오브젝트를 가져오는 일련의 과정을 진행하게 되지만 기존의 사이버 공격 툴에서 만들어지는 바이너리는 단순히 HTTP-GET 요청 메시지만 생성하여 끊임없이 해당 웹서버로 전송하는 공격 방법을 택하고 있다. 따라서 본 논문에서는 정상적으로 파싱과정과 인라인 오브젝트 액세스를 진행하는 일반 사용자 웹 액세스 트래픽과 단순히 HTTP-GET 요청 메시지만 전송하는 좀비 호스트를 구별함으로써 NetBot과 같은 공격 툴을 이용한 공격을 네트워크 레벨에서 검출하는 것을 목표로 한다.

2. HTTP-GET request and its NetFlow data

먼저, 윈도우즈(Windows 7) 컴퓨터의 인터넷 익스플로러(Internet Explorer) 웹브라우저를 사용하여 세계적인 온라인 쇼핑몰 US 아마존닷컴(www.amazon.com) 웹서버를 일반 사용자가 액세스할 때 생성되는 트래픽들을 수집된 NetFlow 정보를 기반으로 분석한다. 이때, US 아마존닷컴의 잘 알려진 9개 IP 주소 리스트 (54.239.25.200, 54.239.17.6, 54.239.25.192, 54.239.25.208, 54.239.19.0, 54.239.26.128, 54.239.17.7, 54.239.19.16, 그리고 54.239.16.47)를 이용하여 일반 사용자가 웹 메인페이지와 웹페이지 내 하이퍼 링크된 하나의 메뉴를 사용자가 클릭했을 때 발생되는 웹페이지 및 임베디드된(embedded) 인라인 오브젝트들을 가져오기 위한 HTTP GET 요청 세션들을 분석하였다. 그림 2는 이러한 일련을 과정을 분석하기 위해 수집된 NetFlow 정보를 이용하여 클라이언트 컴퓨터에서 발생된 HTTP GET 요청 메시지를 기준으로 얼마나 많은 연결이 발생되었으며 또한 이들 연결들이 얼마동안 지속되었는지 보여준다.

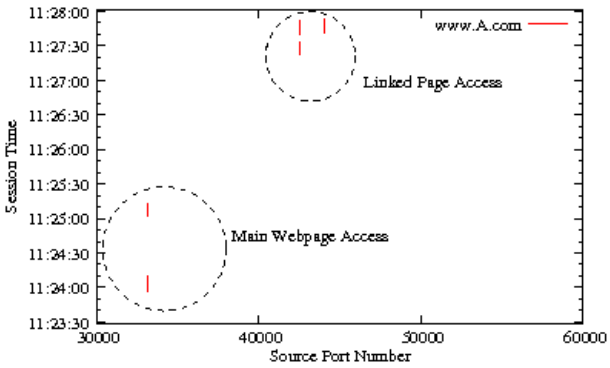

Fig. 2. An example of HTTP GET request sessions for requesting the Web-page and its inline objects

그림 2에서 보듯이, 시간 11:24:00 에 발생된 첫 번째 웹페

이지 요청 및 관련 연결들과 그리고 시간 11:27:30 에 발생된 하이퍼 링크된 메뉴 페이지를 요청하기 위한 연결들을 볼 수 있다.

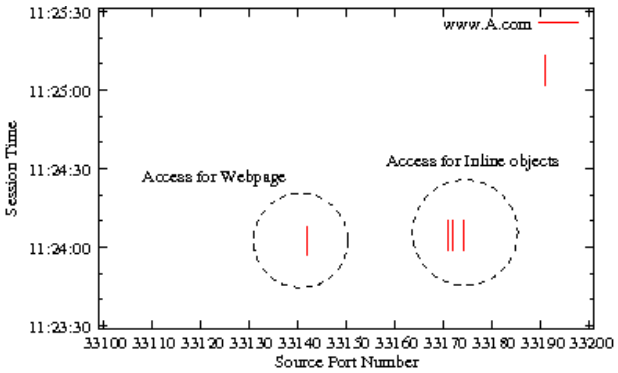

Fig. 3. Zoom in on Fig. 2 around 11:24:00 (An example of HTTP GET request sessions for requesting the Web-page and its inline objects)

한편 그림 2를 보다 자세히 관찰하기 위해 확대한 그림 3과 그림 4를 통해 클라이언트 컴퓨터가 웹 페이지 내 인라인 오브젝트를 가져오기 위한 요청 메시지를 확인할 수 있다. 즉 그림 3에서 클라이언트 발신지 포트번호 33171, 33172, 33174, 그리고 33191를 통한 연결들은 최초 연결 포트번호 33142 연결과 시간 상 병렬로 생성되어 관련 인라인 오브젝트를 가져오고 있음을 확인할 수 있다. 이러한 결과는 그림 4에서 보듯이 두 번째 하이퍼 링크된 웹페이지 액세스에서도 동일하게 확인할 수 있다. 결론적으로, 웹 페이지 수신과 동시에 페이지 내 인라인 오브젝트들을 수신하기 위한 세션들이 동시 다발적으로 이루어지고 있음을 확인할 수 있다.

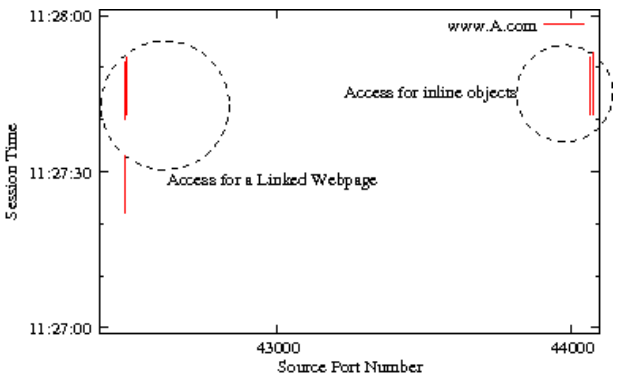

Fig. 4. Zoom in on Fig. 2 around 11:27:30 (An example of HTTP GET request sessions for requesting the Web-page and its inline objects)

3. Detection algorithm

수집된 NetFlow 정보를 기반으로 HTTP GET flood 공격을 검출하기 위해 본 논문에서 제안한 알고리즘을 pseudo code로 그림 5에 나타내었다. 먼저, 보호할 웹서버의 IP 주소 리스트와 해당 웹페이지의 인라인 오브젝트를 제공하는 IP 주소 리스트를 저장한다. 해커들의 주요 공격목표가 되고 있는 대형 포털 및 온라인 쇼핑몰 등은 부하분산(load balancing) 등을 이유로 많은 서버를 운영하고 있기 때문에 이들 IP 주소들을 먼저 확보하여야한다(그림 5의 라인 4와 라인 5). 그림 5에서 보듯이 제안된 검출 알고리즘은 네트워크 레벨에서 동작하기 때문에 매우 단순하다. 즉 보호할 웹서버로 향하는 flow가 검출되면(그림 5의 라인 8), 해당 flow의 세션 기간 동안 인라인 오브젝트를 요청하기 위한 flow의 존재 여부가(그림 5의 라인 11 ~ 라인 17) HTTP GET flood 공격 여부를 판단하게 된다.

V. Experimental Results

본 실험에서는 NetBot Attacker 버전 5.4를 이용해 에이전트 바이너리 파일을 만들어 윈도우즈 XP 컴퓨터에 감염시켜 좀비(zombie) 호스트를 마련하였다. 좀비 호스트는 C&C 서버를 주기적으로 액세스하여 자신의 존재를 알리게 된다. 이제 공격자들은 C&C 서버를 통해 원격으로 감염된 좀비 호스트를 제어하게 되며 본 연구에서 진행하는 HTTP GET flood 공격을 시도하게 된다. 그림 6은 본 연구에서 HTTP GET flood 공격을 검출하기 위해 제안한 알고리즘을 검증하기 위해 구성한 실험실 수준의 테스트베드이다. 한편, 테스트베드의 중앙에 위치한 라우터(C사 3825)는 좀비 호스트가 연결된 인터페이스를 통해 들어오는 (inbound) 트래픽에 대해 NetFlow 정보를 생성하도록 설정하였다. NetFlow 운영을 위한 관련 파라미터들은 기본 값들을 그대로 사용하였으며 앞에서 언급한 바와 같이 'no sampled NetFlow'를 사용하였다. NetBot Attacker C&C 서버는 HTTP GET flood 공격의 기본 파라미터 (쓰레드 10개 그리고 초당 10개 공격 패킷 생성)를 이용하여 좀비 호스트가 웹서버를 공격하도록 명령하였다. 한편 웹서버의 메인페이지 내에는 인라인 오브젝트 두 개를 내장하도록 하여 그림 6에서 보이는 바와 같이 또 다른 외부 웹서버를 하이퍼링크 하였다. 다음 표 2는 실험에 사용된 각종 파라미터 값들을 요약하였다.

```
1  HTTPGETflood_detection ( )
2  {
3     // Initialization
4     webserverIPlist [ ] = { };
5     inlineObjectIPlist [ ] = { };
6
7     while(true) {
8        sleep until a NetFlow record Ψ which has any destination IP address in
9        webserverIPlist[ ] and destination port=80;
10
11       while (session time interval of Ψ) {
12          if (any NetFlow records ζ which have any destination
13             IP addresses in inlineObjectIPlist[ ] and destination port=80)
14                declare flow Ψ is valid;
15          else
16                declare flow Ψ is a HTTP GET flood attack;
17       } // end of while
18    } // end of outer while
19 } // end HTTPGETflood_detection
```

Fig. 5. A pseudo-code of the proposed detection algorithm for HTTP GET flood attacks using NetFlow data

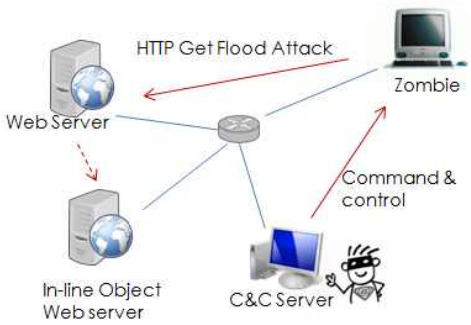

Fig. 6. Testbed for HTTP GET flood Attacks using NetBot Attacker

Table 2. Experimental parameters

Field	Parameters
NetFlow	inactive timer = 15 sec active timer = 30 min
Netbot C&C Server	Number of threads = 10 Number of HTTP GET request = 10/s

실험 시나리오는 좀비 호스트로부터 HTTP GET flood 공격 트래픽 발생과 정상적인 웹서버 접근을 비교하기 위해 다음과 같이 시도하였다.

- 좀비 호스트에 인터넷 익스플로러 웹 브라우저를 실행
- C&C 서버를 원격 제어하여 좀비 호스트가 2분간 HTTP GET flood 공격 시도
- 2분간 공격 중단
- 2분간 재공격 시도
- 2분간 공격 중단 후, 좀비 호스트에서 정상적인 웹서버 접근

그림 7은 실험 시나리오를 통해 테스트 베드에서 수집된 NetFlow 정보를 사용하여 좀비 호스트에서 외부 네트워크로 발생된 모든 플로우를 보여준다. 그림을 통해 알 수 있듯이 좀비 호스트는 C&C 서버로 자신의 존재를 알리기 위해 주기적으로 패킷을 전송한다. 또한 C&C 서버로부터 공격 시도 명령을 수신하게 되면 발신지 포트번호를 계속 증가하면서 해당 웹서버로 지속적인 HTTP GET 요청 패킷을 보내게 된다. 이 후, 공격 시나리오에 준하여 2분 휴식한 다음 다시 공격을 시도하게 된다. 마지막으로, 그림 7에서 시간 89400000 밀리초(하루 시작 0시 기준으로 밀리초로 나타냄)에 좀비 호스트로부터 정상적인 웹서버 접근을 확인할 수 있다.

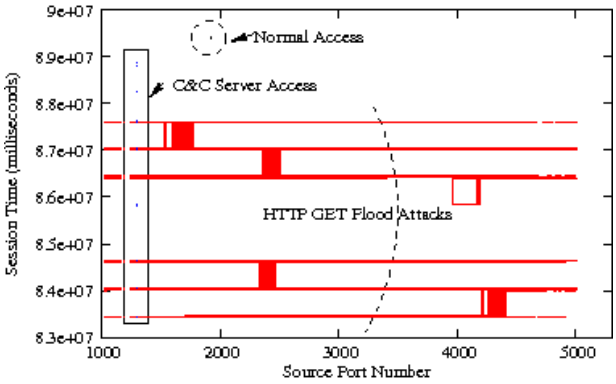

Fig. 7. Generated traffic from the zombie host (plotting using NetFlow data)

그림 8과 그림 9는 HTTP GET flood 공격 트래픽과 정상적인 웹서버 접근 트래픽을 NetFlow 데이터에 기반 하여 비교하기 위해 그림 7을 확대하여 나타내었다. 그림 9의 경우(정상적인 웹서버 접근), 웹서버의 메인 페이지를 가져오기 위한 트래픽과 메인 페이지 내에 존재하는 인라인 오브젝트를 가져오기 위한 트래픽을 확인할 수 있다. 즉 웹브라우저는 메인 페이지를 수신하여 파싱한 결과 인라인 오브젝트를 정상적으로 가져오게 된다. 이에 반하여, 그림8의 경우(HTTP GET flood 공격), 파싱과정을 생략한 무차별적인 request 트래픽만이 존재하는 것을 확인할 수 있다.

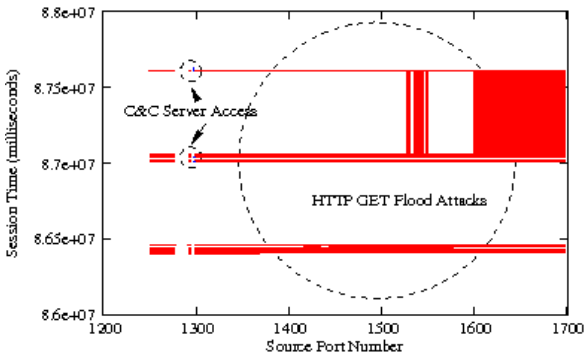

Fig. 8. Zoom in on Fig.7 (HTTP GET flood attacks)

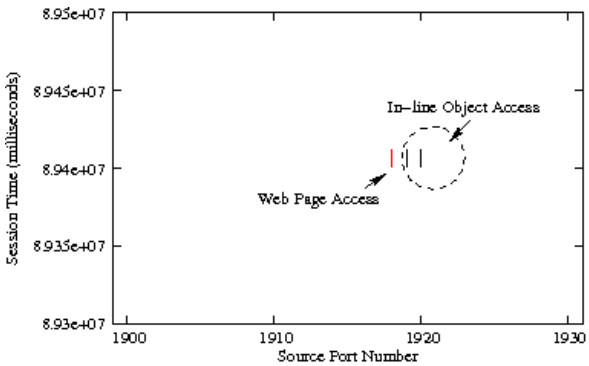

Fig. 9. Zoom in on Fig. 7 (Normal Web access)

VI. Conclusions

본 논문에서는 HTTP GET flood 공격을 NetFlow를 이용한 네트워크 트래픽 플로우 정보를 기반으로 검출하는 방법을 제안하였다. 먼저, 제안된 방법의 장점을 세 가지로 요약할 수 있다. 첫째, 제안한 방법이 사용하는 NetFlow 정보는 오늘날 실제 인터넷에서 운영 중인 대부분의 라우터 및 스위치가 기본적으로 제공하는 정보이기 때문에 공격 트래픽 정보를 추출하기 위해 특별히 새로운 장치를 네트워크에 설치 운영해야하는 부담이 없다. 두 번째의 장점은 다음과 같다. 봇넷을 구성하는 엄청난 수의 좀비 호스트를 이용한 HTTP GET flood 공격과 같은 애플리케이션 레벨 공격들은 정상적인 네트워크 계층 프로토콜을 사용하기 때문에 대부분의 기존 공격 검출 방법들은 애플리케이션 계층에서 deep packet inspection을 통해 처리되고 있다. 그러나 애플리케이션 계층에서 운영되는 공격 검출 기법들은 공격이 이루어지는 근원지보다는 공격에 따른 피해를 입는 희생(victim) 호스트 가까이에서 구현된다. 뿐만 아니라, 애플리케이션 계층에서 이루어지는 검출 방법들은 구현 및 동작에 있어서 매우 복잡한 양상을 보인다. 따라서 본 논문에서 제안하고 있는 네트워크 계층에서 검출할 수 있는 방법은 구현이 비교적 쉬우며 공격의 근원지 가까이 검출 시스템을 위치함으로써 공격 트래픽을 조기에 차단하여 공격에 따른 네트워크의 오용을 방지할 수 있다. 마지막으로, 세 번째 장점은 좀비 호스트로부터 발생되는 트래픽의 양에 의존해 공격을 검출하지 않는다는 것이다. 대부분의 네트워크 레벨에서 flood 공격을 검출하는 기존 방법들은 단순히 공격 패킷 양이 비정상적으로 많아지는 것은 탐지는 어노멀리(anomaly) 기법에 의존한다. 그러나 최근 APT(Advanced Persistent Threat) 공격들은 매우 긴 시간동안 최소의 트래픽을 사용하여 지속적으로 공격하는 경향이 뚜렷하다. 제안된 검출 기법은 단순 임계치 기반이 아니라 웹페이지의 액세스 패턴에 기반하기 때문에 매우 긴 시간동안 느리게 공격패턴을 발생하는 공격들도 검출할 수 있게 된다.

한편, 제안된 기법의 단점은 크게 두 가지이며 다음과 같다. 최근 HTTP GET flood 공격들은 정상 사용자의 액세스 패턴을 흉내 내는 것으로 보고되고 있다. 즉 해당 웹서버의 메인 페이지를 정상적으로 파싱하고 이후 관련 하이퍼링크들을 적절히 액세스하여 일반 사용자들의 접근 패턴과 매우 유사한 행위를 수행한다. 본 논문에서 제시된 검출 기법은 이와 같은 매우 정교한 HTTP GET flood 공격을 검출하는 것은 사실상 불가능하다. 이것은 제안된 검출 기법이 애플리케이션 레벨에서 동작하는 것이 아니라 네트워크 레벨에서 동작하는 근본적인 한계점에 기인한다. 그러나 NetFlow 정보를 활용하여 NetBot과 같은 사이버 공격을 위한 공개 툴들을 이용한 일부 정제되지 않은 HTTP GET flood 공격을 네트워크 레벨에서 효과적으로 차단하는 것이 본 논문의 핵심이다.

마지막으로, 본 논문에서는 좀비 호스트에서 발생되는 모든 트래픽들을 관찰하기 위해 no sampled NetFlow 정보를 활용하는 것으로 가정하였다. 본문에서도 언급한 바와 같이, 인터넷에서 운영 중인 대부분의 core 네트워크 장비들은 sampled NetFlow 정보를 제공하고 있어, 이러한 가정에 대한 문제점을 제기하는 것은 당연하다. 그러나 10G급 core 망이 아닌 access 망에 위치한 네트워크 장비들의 경우는 트래픽 볼륨이 그렇게 크지 않기 때문에 no sampled NetFlow가 가능한 것으로 판단된다. 뿐만 아니라, NetFlow 전용 하드웨어 모듈을 탑재한 네트워크 장비가 현재 출시되고 있으며[20], 또한 빠르게 발전하고 있는 CPU 성능 및 메모리 가격 하락을 감안하면 core 망 네트워크 장비 역시 no sampled NetFlow가 가능할 날이 조만간 올 것으로 예상된다. 만약 예산상의 문제만 없다면 라우터 혹은 스위치가 아닌 NetFlow 생성 전용장비[21]를 네트워크에 태핑(tapping)하여 사용하게 되면 이러한 가정에 대한 문제점을 완전히 해결할 수 있다. 한편, 본 연구진은 앞에서 언급한 바와 같이 최근 매우 정교한 형태의 HTTP GET flood 공격을 NetFlow data를 활용해 검출하기 위한 작업들을 향후 연구과제로 진행하고 있다.

REFERENCES

[1] T. Yatagai, T. Isohara, and I. Sasase, "Detection of HTTP-GET flood Attack Based on Analysis of Page Access Behavior," Proceedings of IEEE Pacific Rim Conference, pp. 232-235, 2007.

[2] D. Dittrich and F. Sven, "P2P as botnet command and control: a deeper insight," Proceedings of the 3rd International Conference on Malicious and Unwanted Software, pp. 41-48, 2008.

[3] P. Chwalinkski, R. Belavkin, and X. Cheng, "Detection of Application Layer DDoS Attack with Clustering and Likelihood Analysis," Proceedings of Globecom, 2013.

[4] AhnLab TrusGuard DPX, http://download.ahnlab.com

[5] Peakflow Threat Management System, http://www.arbornetworks.com

[6] Introduction to Cisco IOS NetFlow, http://cisco.com

[7] B. Claise, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," IETF working group, 2013.

[8] B. Mah, "An Empirical Model of HTTP Network Traffic," Proceedings of INFOCOM'97, pp.592-600, 1997.

[9] W. Lu and S. Yu, "An HTTP Flooding Detection

Method Based on Browser Behavior," Proceedings of Computational Intelligence and Security, pp.1151-1154, 2006.

[10] Y. Choi, I. Kim, J. Oh, and J. Jang, "AIGG Threshold Based HTTP GET Flooding Attack Detection," Proceedings of WISA, 2012.

[11] M. Srivatsa, A. Iyengar, J. Yin, and L. Liu, "Mitigating application-level denial of service attacks on Web servers: A client-transparent approach," ACM Trans. on the Web, Vol. 2, No. 3, Article 15, July 2008.

[12] C.M Chen, B.C Jeng, C.R. Yang, and G.H. Lai, "Tracing denial of service origin: Ant colony approach," Applications of Evolutionary Computing, Springer Berlin Heidelberg, pp.286-295, 2006.

[13] X. Yin, W. Yurcik, M. Treaster, Y. Li, and K. Lakkaraju, "VisFlowConnect: netflow visualizations of link relationships for security situational awareness," Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security, pp.26-34, 2004.

[14] D. Huistra, "Detecting Reflection Attacks in DNS Flows," Proceedings of 19th Twente Student Conference on IT, 2013.

[15] L. Bilge, D. Balzarotti, W. Robertson, E. Kirda, and C. Kruegel, "Disclosure: detecting botnet command and control servers through large-scale netflow analysis," Proceedings of the 28th Annual Computer Security Applications Conference, pp.129-138, 2012.

[16] C. Estan, K. Keys, D. Moore, and G. Varghese, "Building a better NetFlow," ACM SIGCOMM Computer Communication Review, Vol. 34, No. 4, pp.245-256, 2004.

[17] H. Choi and J.O. Limb, "A Behavioral Model of Web Traffic," Proceedings of Seventh International Conference on Network Protocols, pp. 327-334, 1999.

[18] S. Yu, G. Zhao, S. Guo, Y. Xiang, and A.V. Vasilakos, "Browsing Behavior Mimicking Attacks on Popular Web Sites for Large Botnets," Proceedings of IEEE INFOCOM WKSHPS, pp.947-951, 2011.

[19] K.S. Han and E.G. Im, "A Study on the Analysis of Netbot and Design of Detection Framework," Proceedings of Joint Workshop on Information Security, pp.1-12, 2009.

[20] Cisco, Cisco Catalyst 3750-X and 3560-X Series Switches Data Sheet, http://www.cisco.com/c/en/us/products/collateral/switches/catalyst-3750-x-series-switches/data_sheet_c78-584733.html

[21] endace, EndaceFlow NetFlow Generator Appliances, http://www.endace.com/endaceflow-high-speed-netflow-generators.html

Authors

Koo Hong Kang received the B.S. and M.S. degrees in Electronics Engineering from Kyungpook National University and Chungnam National University, Korea, in 1985 and 1990, respectively, and then Ph.D. degree in Computer Science and Engineering from POSTECH, Korea, in 1998.

From 1985 to 2000, Dr. Kang was a researcher and a senior researcher at ETRI (a national lab in Korea) participating in various research projects in TDX switching system, ATM networks, and network security. He is currently a Professor in the Department of Information and Communications Engineering, Seowon University. He is interested in computer networks and Internet security.