TCP 연결의 스테이트풀 인스펙션에 있어서의 보안 약점 최소화 및 성능 향상 방법

(Minimizing Security Hole and Improving Performance in Stateful Inspection for TCP Connections)

김 효 곤 † 강 인 혜 ††
(Hyogon Kim) (Inhye Kang)

요 약 스테이트풀 인스펙션을 수행하는 기기에서는 패킷 플로우에 대한 정보를 유지해야 한다. 이러한 기기는 네트워크 공격 패킷에 대하여도 패킷 플로우 정보를 유지하게 되어 네트워크 공격 하에서 과도한 메모리가 요구되고 이로 인하여 메모리 오버플로우나 성능 저하가 일어난다. 따라서 이 논문은 스테이트풀 패킷 인스펙션 시 공격에 의해 생성되는 불필요한 미완성 엔트리를 제거하기 위해 사용할 수 있는 플로우 엔트리 타임아웃 값에 대한 가이드라인을 제시한다. 대부분의 인터넷 트래픽과 상당수의 네트워크 공격이 TCP 프로토콜을 사용하기 때문에 RFC2988의 TCP 재전송 시간 계산 규약에 기초를 둔 실제 인터넷 트레이스에 대한 분석을 통해 가이드라인을 도출한다. 구체적으로, 미완성 TCP 연결 설정 상태에서 (R+T) 초 이상 경과한 엔트리는 제거하여야하며, 이 때 R은 SYN 재전송 허용 회수에 따라 0,3,9를 선택하고 T는 $1 \leq T \leq 2$ 에서 부가적인 왕복 지연 허용치에 따라 선택하여야 함을 보인다.

키워드 : 패킷 인스펙션, 세션 테이블 관리, TCP 재전송 시간 계산 알고리듬, 타임아웃

Abstract Stateful inspection devices must maintain flow information. These devices create the flow information also for network attack packets, and it can fatally inflate the dynamic memory allocation on stateful inspection devices under network attacks. The memory inflation leads to memory overflow and subsequent performance degradation. In this paper, we present a guideline to set the flow entry timeout for a stateful inspection device to remove harmful embryonic entries created by network attacks. Considering Transmission Control Protocol (TCP) is utilized by most of these attacks as well as legitimate traffic, we propose a parsimonious memory management guideline based on the design of the TCP and the analysis of real-life Internet traces. In particular, we demonstrate that for all practical purposes one should not reserve memory for an embryonic TCP connection with more than $(R+T)$ seconds of inactivity where R=0, 3, 9 and $1 \leq T \leq 2$ depending on the load level.

Key words : packet inspection, session table management, TCP retransmission timeout calculation, timeout

1. 서 론

최근 인터넷이 발전함에 따라 패킷 처리를 위해 특화된 많은 종류의 컴퓨터가 사용되고 있다. 대표적으로는 라우터나 스위치와 같은 장비로부터 방화벽[1], VPN(Virtual Private Network), 네트워크 침입 탐지 시스템, 트래픽 모니터링 장비[2,3], 과금 시스템[4], 로드 밸런싱 장비[5] 등이 있다. 인터넷 트래픽 양이 무어의 법칙[6]을 앞지르게 증가함에 따라, 이러한 컴퓨터에서는 패킷 처리 작업의 부하가 높아져 성능을 위한 패킷 처리의 최적화가 필수적이다. 따라서 라우팅 테이블 검색(routing table lookup)과 패킷 분류(packet classification)와 같은 패킷 처리에 요구되는 기능에 대한 효율을 높이기 위한 많은 연구가 진행되어 왔다[7-9]. 그러나 패킷처리를 위해 동적으로 할당되는 메모리의 구성과 관리를 위한 연구는 상대적으로 부족하다. 따라서 본 논문에서는 패킷 처리에서 동적으로 할당되는 메모리를 구성 및 관리하는 이슈를 다룬다.

스테이트풀 패킷 인스펙션(Stateful Packet Inspec-

· 이 논문은 2003년도 서울시립대학교 학술연구조성비에 의하여 연구되었음
† 비 회 원 : 고려대학교 컴퓨터학과 교수
hyogon@korea.ac.kr
†† 종신회원 : 서울시립대학교 기계정보공학과 교수
inhye@uos.ac.kr
논문접수 : 2004년 4월 8일
심사완료 : 2005년 4월 8일

tion)에서의 패킷 처리는 각 패킷의 내용뿐 아니라 같은 플로우의 이전 패킷에 의해 영향을 받는다. 따라서 같은 플로우 내에 있는 이전 패킷들의 상태에 대한 정보를 유지하는 것이 필요하다. 이를 위하여 패킷 인스펙션 컴퓨터에는 플로우가 생성되고 사라짐에 따라 대응하는 엔트리가 생성되고 삭제된다. 현재 방화벽, VPN, 네트워크 침입 탐지 시스템, 트래픽 모니터링과 사용기반 과금 시스템은 모두 정도는 다르지만 스테이트풀 인스펙션을 요구한다.

일반적으로 스테이트풀 인스펙션 컴퓨터는 공간 사용 및 검색의 효율을 위하여 타임아웃 메커니즘을 사용하여 유효하지 않은 엔트리를 삭제한다. 하지만 이러한 컴퓨터는 타임아웃을 위한 값을 개발자가 임의로 (보통 60초나 120초 등 상당히 큰 값으로) 지정하거나 사용자가 구성할 수 있도록 할 뿐 타임아웃에 대한 체계적인, 즉 프로토콜 및 트래픽 분석에 의거한 가이드라인은 제시해주지 않는 실정이다[10]. 그러나 적절한 타임아웃 설정은 효율적인 패킷 처리를 위해 필수적이다. 우선, 타임아웃 시간이 너무 짧으면 과도한 엔트리 생성 및 삭제가 일어나 원하지 않는 결과가 발생할 수 있다. 예를 들면 방화벽에서 허용된 플로우에 대응하는 엔트리가 삭제되면 이후 합법적인 패킷인데도 불구하고 통과시키지 않는 결과가 생길 수 있다. 반면에 타임아웃 시간이 길어지면 만료된 플로우에 대한 엔트리를 불필요하게 오래 유지하게 되어 메모리 요구량이 증가한다[11]. 더구나 네트워크 공격 하에서 패킷 인스펙션 컴퓨터 자체가 공격 대상이 아니라 하더라도 공격 때문에 메모리 오버플로우가 발생할 수 있다. 이는 공격성 트래픽 스트림의 경우 패킷 마다 IP 주소 혹은 포트 번호가 계속 변화함으로써 (그 이유는 3절에서 자세히 논의하기로 한다), 통상의 플로우의 정의에서 볼 때 모두 상이한 플로우로 인식되기 때문이다. 그렇게 되면 공격 패킷 각자가 모두 한 개씩의 플로우 엔트리에 해당하므로, 이 트래픽에 대해 스테이트풀 인스펙션을 수행하는 컴퓨터에는 플로우 엔트리 생성을 위한 메모리 요구량이 급속히 증가하게 된다.

본 논문에서는 스테이트풀 패킷 인스페션 컴퓨터에서 불필요한 엔트리를 적절한 시간에 제거하기 위해 사용되는 타임아웃 시간에 대한 가이드라인을 제시하고자 한다. 구체적으로, 다음 두 가지 요구사항을 만족하고자 한다. 첫째, 타임아웃 시간이 합법적인 플로우의 정상적인 동작에 영향을 미치지 않도록 충분히 커야 한다. 둘째, 타임아웃 시간은 공격과 같은 비정상적인 플로우에 의하여 생성된 엔트리를 최소화시키도록 충분히 짧아야 한다. 이 가이드라인을 통하여 스테이트풀 인스펙션 컴퓨터의 타임아웃을 효율적으로 설정하고, 스테이트풀 인스펙션이 네트워크 공격 하에서도 계속 기능할 수 있도록 한다.

위에서 언급한 대로, 기존의 연구는 스테이트풀 인스펙션을 위한 동적 메모리 관리가 아닌 패킷 분류(classification)를 위한 정적 테이블 크기 줄이기와 검색 시간 최소화에 집중되어 왔다[7-9]. 세션 혹은 플로우 테이블을 사용하는 스테이트풀 인스펙션을 위한 테이블에서, 공격에 의한 오버 플로우의 가능성을 제기한 논문은 지금까지 [2]가 유일하다. 그러나 이 논문에서도 고속 링크에서의 패킷 모니터링을 어렵게 하는 요소로서 언급하고 있을 뿐, 타임아웃 값을 어떻게 잡아야 하느냐의 연구는 이루어진 바 없다. 이와 같은 연구의 부재는 대량의 "일반적인" 인터넷 트레이스를 구하기 어렵기 때문이라고 생각된다. 즉 가이드라인을 정하기 위해서는 실제 네트워크 트래픽을 대량으로 분석, 어느 시간 안에 대부분의 TCP 연결이 맺어지는지 밝혀주어야 하기 때문이다. 따라서 Cisco, Netscreen, Checkpoint 등 실제 시스템들에서는 가이드라인이 없음으로 해서 최소 60초 이상의 값을 기본값 (default)으로 정하고 있다[10]. 본 논문은 1 테라바이트 가량의 인터넷 백본 트레이스 분석을 통해 이와 같은 문제를 최초로 다룬 것이며, 따라서 선행 연구는 거의 없는 것으로 보인다.

본 논문의 구성은 다음과 같다. 2장에서는 스테이트풀 패킷 인스팩션에서 요구되는 동적 상태 관리의 배경을 기술한다. 특히 스테이트풀 인스펙션 컴퓨터가 계속 유지해야 하는 TCP 상태에 대하여 소개한다. 3장에서는 본 논문에서 다루는 문제를 정의한다. DoS(Denial of Service)나 웜(worm)과 같은 네트워크 공격이 스테이트풀 인스펙션 컴퓨터에서 어떻게 상태폭발(state explosion)을 일으킬 수 있는지에 대하여 설명한다. 4장은 TCP 프로토콜 표준에 기반한 해결 방법을 제안하고 실제 인터넷 트레이스에 대한 분석을 기술한다. 5장에서는 4장에서의 분석 결과를 사용하여 스테이트풀 인스펙션 컴퓨터를 위한 타임아웃 가이드라인을 기술한다. 6장에서는 본 논문을 결론 맺는다.

2. 동적 상태 관리

스테이트풀 패킷 인스펙션 컴퓨터는 네트워크 내의 그 관찰 위치에서 현재 진행중인 플로우에 대한 정보의 리스트를 가지는데, 이를 보통 세션 테이블이라고 부른다. 통상 한 개의 플로우 관련 정보는 <프로토콜, 출발지 IP 주소, 출발지 포트 번호, 도착지 IP 주소, 도착지 포트 번호>로 구성된다. 응용에 따라 추가적인 정보가 필요할 수도 있는데 예를 들면 스테이트풀 인스펙션 방화벽의 경우 TCP 순차번호(sequence number)를 기록한다[1]. 패킷 인스텍션 컴퓨터는 관찰되는 각 패킷에

대하여 플로우 정보를 추출하여 세션 테이블의 엔트리와 비교한다. 정보가 일치하는 엔트리가 존재하면 해당 엔트리에 정의된 액션(action)을 수행하는데, 예를 들면 방화벽이 패킷을 통과 혹은 거부 시키거나 사용량에 따른 (usage-based) 과금 시스템이 패킷 혹은 바이트 카운트를 증가시키는 것과 같은 것이다. 반면 일치하는 엔트리가 존재하지 않으면, 즉 새로운 플로우의 시작 패킷이면 세션 테이블에 이 플로우에 대한 엔트리를 새로 생성한다. 또한 플로우의 종료가 관찰되면 세션 테이블에서 해당 엔트리를 제거한다.

플로우의 시작과 종료를 결정하는 방법은 사용하는 프로토콜에 따라 차이가 있다. UDP와 같은 비연결형(connectionless) 프로토콜에서는 플로우의 종료가 엄밀히 말하면 추측에 의해 결정되는데, 통상 일정 시간 동안 해당 플로우에 대한 패킷이 관찰되지 않으면 플로우가 종료된 것으로 간주하는 것이 일반적이다.

연결지향형(connection-oriented) 프로토콜은 TCP가 대표적이다. TCP 연결 설정은 그림 1에서 보는 바와 같이 두 호스트 사이에 3개의 패킷이 교환되기 때문에 3-웨이 핸드셰이크(3-way handshake)라고 부른다[12]. 우선 호스트 A가 연결을 개시하기 위하여 SYN 패킷을 상대 호스트 B에게 송신한다. 호스트 B는 호스트 A로부터의 SYN 패킷을 받으면 이에 대하여 수신확인(acknowledge)을 하면서 동시에 반대 방향 데이타 채널을 형성하기 위하여 SYN/ACK 패킷을 호스트 A로 보낸다. TCP는 전이중(full-duplex) 프로토콜로서 연결당 각 방향으로 하나씩 데이타 채널을 요구하기 때문에 양방향으로 동기화 패킷이 요구된다. 마지막으로 호스트 A는 호스트 B로 SYN에 대한 확인인 ACK 패킷을 보냄으로써 TCP 연결 설정이 완료된다.

호스트 A와 호스트 B 사이에 형성된 연결이 통과하는 네트웍의 한 지점에 스테이트풀 인스펙션 컴퓨터가 있다고 하자(그림 1). 여기서는 TCP 연결 설정 이벤트를 검출할 수 있을 뿐만 아니라 3-웨이 핸드셰이크 동안 설정 진행 상태도 감시할 수 있다. 또한 연결 설정 지연(setup delay)을 D_c라고 할 때 $D_c' \approx D_c$를 관찰함으로써 연결 설정 지연을 측정할 수 있다.

RFC2988[13] 표준에 따르면, TCP SYN 패킷을 분실하면 재전송을 시도하는데 $k(\geq 1)$번째 SYN 재전송은 $(k-1)$번째 재전송 패킷의 송신 이후 $3 \times 2^{(k-1)}$ 초 내에 이루어져야 한다 (0번째 재전송은 정의상 첫 번째 SYN 전송이다). 이를 지수적 백오프(exponential backoff)라 하는데 일종의 혼잡제어 메커니즘이다. 만일 SYN 전송이 계속 실패하는 경우 3-웨이 핸드셰이크 동안 SYN 패킷 재전송 간 시간 간격은 3, 6, 12, 24, … 로 길어지게 되고 따라서 D_c, 즉 D_c'이 커지게 된다.

TCP는 연결 종료를 위해 연결 설정 시와 유사하게 FIN 패킷과 FIN에 대한 ACK 패킷을 교환하도록 한다. 이 FIN과 ACK 교환이 양 채널에 대해 이루어지면 패킷 인스펙션 컴퓨터는 세션 테이블에서 해당 엔트리를 제거한다. 또한 RST 패킷에 의하여 연결이 중단될 경우에도 세션 테이블에서 해당 엔트리를 지운다.

스테이트풀 인스펙션 컴퓨터에서 세션 테이블의 총 엔트리 수는 동시에 활동하는 플로우(concurrent active flow)의 개수에 의존한다. 2003년 현재 인터넷 핵심부에서는 동시에 통상 수십만 개 이상의 플로우가 한 링크를 지나고 있는 것을 관찰할 수 있다. 예를 들면 2003년 4월 인터넷 백본에 해당하는 어느 한 OC-48 (2.4Gbps) 링크에서는 최고 237,000 개의 플로우가 동시에 관찰되었다[11]. 최근 백본 망에 OC-192 (10Gbps) 링크가 사용되기 시작한 것을 고려하면 앞으로 고속 링크에는 동시에 수백만 개의 플로우가 존재할 수 있으리라는 예측을 할 수 있다.

3. 네트워크 공격의 영향 분석

세션 테이블의 크기는 엔트리 개수와 엔트리 크기의 곱이므로, 만약 각 엔트리의 크기가 (2개의 IP 주소, 두 개의 포트 번호, 프로토콜 번호 및 테이블 유지를 위한 기타 오버헤드를 포함하여) 40 바이트라고 하면 1백만 엔트리를 가진 패킷 인스펙션 컴퓨터에서 세션 테이블의 크기는 40M 바이트가 된다. 현재 컴퓨터의 메모리 용량을 고려하면 이 정도의 세션 테이블 크기는 충분히 지원 가능하다. 하지만 네트워크 공격이 진행되면 세션 테이블의 엔트리 수는 크게 폭증할 수 있다.

본 논문에서는 네트워크 공격 중 스테이트풀 인스펙션에 영향을 줄 수 있는 서비스거부(denial of sevice, DoS) 공격과 스캐닝(scanning)에 초점을 맞춰 그 특성을 알아본다. 표 1(a)는 실제 백본에서 관찰된 DoS 공격 트레이스의 일부이다[14]. 여기에서는 피해자(victim)의 프라이버시 보호를 위해 피해자의 호스트 IP를 "y.y.y.y"로 표현한다. 여기에서 I_s는 패킷의 출발지 IP 주소, I_d는 도착지 IP 주소, p_s는 출발지 포트 번호, p_d는 도착지 포트 번호이다. 통상 DoS 공격에서 공격자가 피해자의 호스트 IP 주소 I_d를 고정시키는 반면 I_s, p_s, p_d는 무작위로 생성한 번호를 채워 넣는다. 공격자는 피해자에게 연결을 시도하고 있지 않을 뿐 아니라 공격자 IP 추적을 피하기 위해서도 I_s를 무작위로 고르기 때문이다[15]. 예를 들어 표 1(a)에서 나타난 출발지 주소 "87.231.152.166"은 현재 IANA[16]에서 누구에게도 할당하지 않은 주소이기 때문에 쉽게 허위 주소라는 것을 알 수 있다.

호스트스캔에서는 피해자의 호스트 IP 주소 I_d가 패킷

표 1 실제 트레이스에서의 네트워크 공격의 패킷 플로우 정보

(a) DoS 공격

Time	I_s	p_s	I_d	p_d
.....	.....	.....	.....	.....
09:37:03.319081	67.171.49.204	7804	y.y.y.y	16675
09:37:03.319647	20.214.51.196	47582	y.y.y.y	16675
09:37:03.319652	55.44.55.180	61602	y.y.y.y	16687
09:37:03.319922	55.44.55.180	61602	y.y.y.y	16687
09:37:03.320607	89.130.59.164	10086	y.y.y.y	16695
09:37:03.321665	56.184.129.14	4787	y.y.y.y	16706
09:37:03.322084	117.152.194.136	51005	y.y.y.y	16709
09:37:03.322098	3.164.5.250	5928	y.y.y.y	16710
09:37:03.322582	44.57.134.246	58585	y.y.y.y	16718
.....	.....	.....	.....	.....
09:37:03.325331	25.123.15.210	8210	y.y.y.y	16736
09:37:03.326188	6.188.152.174	23371	y.y.y.y	16754
09:37:03.326565	6.188.152.174	23371	y.y.y.y	16754
09:37:03.327048	87.231.154.166	63149	y.y.y.y	16768
09:37:03.327248	101.242.222.24	18073	y.y.y.y	16765
.....	.....	.....	.....	.....

(b) Code Red II 웜에 의한 호스트스캔

Time	I_s	p_s	I_d	p_d
13:27:35.602109	x.x.x.x	2101	210.185.103.244	80
13:27:35.602113	x.x.x.x	2100	210.248.154.32	80
13:27:35.602117	x.x.x.x	2102	210.175.128.217	80
13:27:35.602122	x.x.x.x	2293	210.70.243.85	80
13:27:35.602127	x.x.x.x	2367	210.107.63.78	80
13:27:35.602616	x.x.x.x	2378	210.78.33.141	80
13:27:35.642113	x.x.x.x	2379	58.80.253.118	80
13:27:35.692445	x.x.x.x	2380	210.202.242.119	80
13:27:35.702067	x.x.x.x	2108	210.78.146.218	80
13:27:35.702071	x.x.x.x	2107	210.107.216.227	80
13:27:35.702076	x.x.x.x	2294	210.60.83.150	80
13:27:35.702080	x.x.x.x	2105	210.133.45.230	80
13:27:35.702084	x.x.x.x	2106	133.64.252.91	80
13:27:35.702089	x.x.x.x	2362	210.142.241.241	80
13:27:35.762039	x.x.x.x	2381	210.20.147.102	80
13:27:35.801651	x.x.x.x	2109	210.199.26.105	80
13:27:35.801661	x.x.x.x	2297	210.141.135.151	80

별로 변화한다. 보통 해커는 공격 개시 이전에 취약성 탐지를 하기 위하여 호스트스캔을 시도하고 웜(worm)인 경우에는 감염 대상 호스트를 찾기 위하여 호스트스캔을 수행한다. 공격자는 취약한 호스트의 주소를 알아내기 위하여 임의의 IP 주소범위에 대하여 무작위로 스캔을 수행한다. 예를 들면 표 1(b)에서 보이고 있는 Code Red II 웜의 실제 트레이스의 일부에 현재 IANA에서 할당하지 않은 주소인 "58.80.253.118"가 나타나는 것을 볼 수 있다.

패킷 인스펙션 컴퓨터에서는 각 플로우에 대하여 하나의 세션 엔트리가 생성되기 때문에 플로우의 구별자(identifier)인 I_s, p_s, I_d, p_d 중 한 값만 달라도 별도의 엔트리가 생성된다. DoS 에서는 I_s, 호스트스캔에서는 I_d, 포트스캔에서는 p_d가 각각 변화한다는 차이가 있을 뿐이다. 즉 동일 공격에 속하는 모든 패킷은 같은 플로우 구별자를 공유하지 않기 때문에 각 패킷마다 다른 세션 엔트리를 생성한다. 더욱 심각한 문제는 이러한 공격들은 패킷 생성률이 아주 높다는 것이다. 지금까지 인터넷에서 발생한 대형 공격 중 몇 개를 예로서 살펴보면 이 점이 분명해진다 (강조를 위하여 극단적인 사례를 들었다) DoS의 경우 패킷 생성률을 높일수록 공격력을 높일 수 있기 때문인데 2002년 10월 루트 DNS 서버에 대한 DDoS 공격을 살펴보면 한 서버에 대하여 초당 10만에서 20만 정도의 공격패킷이 기록되었다[17]. 이것은 만약 어떤 패킷 인스펙션 컴퓨터가 루트 DNS 서버 근처에 놓여 있었다면 수 초 내에 통상 OC-48 링크에서 동시에 관찰되는 플로우 수보다 훨씬 많은 수의 공격 관련 엔트리를 생성하게 되었으리라는 것을 의미한다. 호스트스캔에서도 패킷 생성이 빠를수록 웜의 전염 속도가 빨라지거나 또는 취약한 호스트를 빠르게 발견할 수 있다. SQL Slammer 웜에 의한 호스트스캔의 경우 감염 호스트 한 대가 최대 초당 26,000 패킷을 전송한 경우도 있었다[18]. 예를 들어 스테이트풀 인스펙션 컴퓨터가 10개의 이러한 감염 호스트를 포함한 기업망의 경계에 위치한다면 이 공격과 관련된 엔트리 수는 공격 4초 이내에 1백만 개를 초과하게 될 것이다.

더더욱 상황을 악화시키는 것은 이렇게 공격 패킷에 의해 생성된 엔트리들은, 허용되는 최대 시간 동안 세션 테이블에 존재할 것이라는 것이다. 정상적인 TCP 플로우에서는 종료할 때 FIN 또는 RST 패킷을 교환하게 되고 이러한 패킷을 관찰함으로써 해당 엔트리를 삭제할 수 있다. 하지만 DoS 공격의 경우에는 FIN 또는 RST 패킷이 없기 때문에, 세션 테이블에서 공격 관련 엔트리는 타임아웃에 의해 제거 될 때까지 계속 남아 있게 된다. 호스트스캔의 경우 엔트리의 수명은 프로토콜에 따라 다르다. 스캐너가 TCP를 사용한다면 스캔되는 호스트는 스캐닝 기법에 따라 다양하게 반응한다[19]. 예를 들어 Code Red II가 운 좋게도 감염 가능한 호스트를 발견한 경우에는 정상적인 연결 설정 및 (웜 전송 후) 종료를 수행하기 때문에 세션 테이블에서 해당 엔트리가 삭제되게 된다. 하지만 대부분의 스캔 패킷은 사용되지 않는 IP 주소로 보내지고, 따라서 라우터에서 Destination Unreachable ICMP 에러를 일으키게 된다. 이 ICMP 에러를 일으킨 패킷이 생성한 플로우 엔트리는, 스테이트풀 인스펙션 컴퓨터가 엔트리 삭제를 목적으로 따로 ICMP 메시지를 처리하지 않는 한 제거되지 않을 것이다.

다시 정리하면 공격에 의한 엔트리는 빠른 속도로 각 공격 패킷마다 생성되고 오랜 동안 세션 테이블에 남아 있게 된다. 스테이트풀 인스펙션 컴퓨터에서는 각 패킷

에 대하여 세션 테이블 검색(lookup)을 수행해야 하기 때문에 이 검색 성능은 스테이트풀 인스펙션 컴퓨터의 패킷 처리량(throughput)에 큰 영향을 끼칠 수 있다. 보통 세션 테이블 검색에 해쉬이 사용되므로 엔트리 개수의 증가는 해쉬 버킷당 평균 엔트리 수를 증가시키고 따라서 세션 테이블 검색 속도를 저하시킨다. 그러므로 패킷 처리량의 감소를 일으키는 불필요한 엔트리 수의 증가를 막기 위하여 네트워크 공격으로부터 발생하는 불완전한 엔트리의 생성을 없애는 방법이 필요하다.

TCP가 인터넷 트래픽의 90% 내외를 차지하고[20] 많은 공격에서 TCP 프로토콜이 사용되기 때문에[18] 본 논문에서는 TCP 프로토콜을 사용한 공격에 의해 생성된 엔트리를 제거하기 위한 가이드라인을 제시하고자 한다. 특히 불필요한 미완성 TCP 세션을 제거하는 것에 초점을 둔다. 반면 이미 연결이 설정되었으나 불활성인 (inactive) TCP 플로우는 TCP 프로토콜의 Keepalive 타이머에 의해 연결 수준에서 제거되기 때문에 이 논문에서는 고려하지 않는다.

UDP를 이용하는 공격에 의한 세션 테이블 오버플로우를 방지하기 위해서는 UDP 플로우의 타임아웃 가이드라인이 필요하다. 그러나 연결(connection) 개념이 없는 UDP 플로우의 특성상 타임아웃은 UDP 플로우의 비활동 (inactivity) 시간 분포에 근거한 것이어야 하는데, 백본 트레이스의 대량 분석 결과는 UDP 플로우의 비활동 시간 분포가 넓은 시간대에 걸쳐져 있음을 보여준다[14]. TCP처럼 배후에 프로토콜 다이내믹스가 있지 않은 UDP의 경우는 응용의 특성에 따라 비활동 시간이 결정되는데, 특별히 의미를 갖는 시간 값이 없다. 또한 위에서 언급한 바와 같이 UDP 트래픽은 통상 인터넷에서 5% 내외에 불과하고, 공격 트래픽에서도 UDP를 사용하는 경우가 드물므로, 이 논문에서는 UDP를 고려하지 않기로 한다.

4. 인터넷 트레이스 분석

위 절에서는 어떻게 패킷 인스펙션 컴퓨터에서 세션 엔트리 수가 공격 트래픽에 의해 폭증할 수 있는가를 보였다. 본 논문의 목적은 이러한 엔트리 수의 폭증을 막기 위한 세션 엔트리 타임 아웃 가이드라인을 제시하는 것이다. 가이드라인을 도출하기 위해 이 논문이 채택한 기본적인 접근법은 다음과 같다.

인터넷으로부터 가능한 한 많은 TCP 연결을 관찰하여 "일반적인" TCP 연결 설정 지연분포를 얻는다.

이 분포를 기초로 하여 정상적으로 진행되는 거의 모든 비 공격성 연결이 연결 설정을 완료하기에 충분한 설정 타임아웃을 선택한다.

이 타임아웃까지 완성되지 않은 연결은 공격으로 간주하고 세션 테이블에서 지우도록 한다. 즉 (2)에서 도출한 타임아웃 값을 미완성 세션 엔트리의 타임아웃 값 가이드라인으로 제시한다.

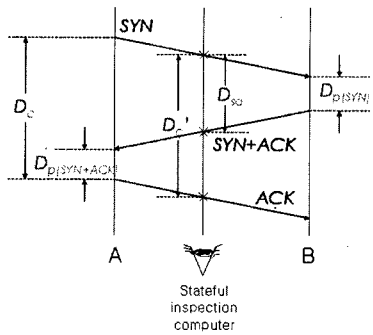

그림 1 패킷 인스펙션 컴퓨터에서 관찰되는 TCP 연결 설정 과정

(1)의 TCP 연결 설정 지연의 분포를 분석하기 위하여 2001년 12월에 10일 동안 수집된 백본 패킷 트레이스를 사용하였다. 이 트레이스는 국내의 네 IX (Internet Exchange) 중 하나인 KIX (Korea Internet Exchange)와 미국간 연결을 위해 태평양을 횡단하는 두 개의 T3 링크에서 교환되는 트래픽을 기록한 것으로, 오전 9시에서 오후 5시 사이에 패킷 헤더만을 수집한 것이다. 매일 약 60억 패킷 이상이 수집되었고, 이 패킷 트레이스 분석 결과 하루 분량의 트레이스로부터 평균 800만 개 이상의 TCP 연결을 도출하였다.

TCP 연결 설정 지연은 그림 1에서 본 바와 같이 SYN 패킷 송신과 대응하는 ACK 패킷 수신 사이의 시간으로 정의할 수 있다. 하지만 SYN 송신과 SYN/ACK 수신 사이의 시간차이 D_{sa}, 즉 연결 경로 중간에 놓인 관찰자의 입장에서 본 왕복 시간을 트레이스의 패킷으로부터 추정하는 것은 매로 용이하지 않을 뿐 아니라, 일반적으로 연결 설정 시간으로 볼 수 없다. 전자는, 비대칭적 라우팅 (asymmetric routing)이 발생하는 경우 SYN에 대응되는 SYN/ACK가 관찰지점을 비껴 다른 경로로 전송될 수 있기 때문이다. 그 경우 관찰자가 SYN과 SYN/ACK 간의 시간차를 계산하는 것은 불가능하다. 후자는, 백본 망에서 수집된 트레이스는 네트워크의 중간 지점에서의 패킷이고 따라서 $D_{sa}' \leq D_{sa}$이다. 이 문제는 관찰 위치가 호스트 B에 가까울수록 악화된다. 이와 같은 이유로 본 논문에서는 TCP 연결 설정 지연을 SYN 송신과 SYN/ACK 수신 사이의 시간차이 D_{sa}가 아니라 SYN 전송과 ACK 전송 사이 시간차이인 D_c로 추정하기로 한다. D_c는 D_c'을 측정함에 의하여 근사치를 얻을 수 있다. (여기에서 호스트 A에서 관찰지점까지 네트워크 경로의 가변적 지연의 가능

성 때문에 근사치라고 한다.) 비대칭 라우팅 하에서 동작하는 인터넷 환경에서는 이 근사치를 사용하는 것이 더욱 필수적이다. 호스트 A로부터 호스트 B로의 SYN 패킷이 관찰위치를 거쳐서 간다고 하더라도 반대방향으로는 즉 SYN/ACK 패킷은 다른 경로를 통하여 올 수 있다. 하지만 다시 A에서 ACK 패킷을 송신하면 이는 SYN 패킷과 같은 경로를 통해 진행하기 때문에 다시 관찰 가능하고 따라서 D_c'을 계산할 수 있다.

이 연구에서 분석 대상 트레이스가 태평양 사이를 오가는 즉 장거리 패킷을 수집한 것이라는 것은 중요한 의미를 가진다. 트레이스에 기록된 모든 TCP 연결은 한국과 미국 사이의 장거리 연결이기 때문에 지연과 손실률이 상대적으로 높으리라는 예상을 할 수 있다. 즉 타임아웃이나 연결 설정 지연의 관점에서 보면 관찰된 TCP 연결 행위는 최악의 상황에 가깝다고 볼 수 있다. 이 보수적인 지연 예상치를 기초로 하여 연결 설정 타임아웃을 선택하도록 함으로써, 정상적인 TCP 연결 설정은 이 타임아웃 이전에 대부분 완료되도록 의도한다.

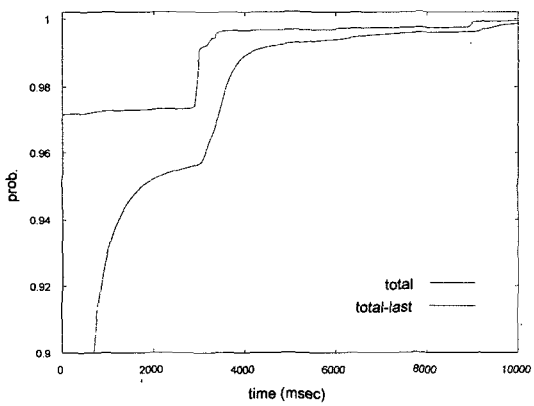

그림 2 연결 설정 지연의 누적 확률 분포

그림 2는 연결 설정 지연의 CDF(cumulative distribution function)인데, X-축은 연결설정지연이고 Y-축은 누적 확률이다. 아래의 곡선 t_{total}은 전체 연결 설정 지연 D_c'을 나타낸다. 이것은 SYN 전송과 3-웨이 핸드셰이크를 완성하는 ACK 전송 사이의 시간 차이인데 여기에는 SYN 재전송에 의한 지연이 포함된다. t_{total} 위의 곡선은 t_{last}가 최후의 (다시 말해 성공적인) SYN 전송과 SYN/ACK 응답의 시간차이라고 할 때 (t_{total} - t_{last}) 의 분포로 SYN 재전송에 소비한 시간만의 분포를 나타낸다.

그림 2에서 곡선 t_{total}을 살펴보면 1초 근처에서 급격한 증가가 있고 또 3초 근처에서 급격한 증가가 있다는 관찰할 수 있다. 이 두 번째 증가 후 연결 설정의 누적 비율은 99%를 넘어선다. 자세히 살펴보면 9초 근처에서도 상대적으로 빠른 증가가 있다. 이 상승 이후 누적비율은 99.5%를 넘어선다. 비록 알기 쉽지 않지만 6초 근처에도 상대적으로 빠른 상승이 발생하는 것을 관찰할 수 있다. 이 분포는 다음과 같은 중요한 사실을 반영한다. 3초, 6초, 9초에서의 빠른 증가는 TCP 재전송 타임아웃에 의한 것이다. SYN 패킷의 재전송 간 시간 간격은 TCP 구현에 따라 다르다. 예를 들면 BSD 기반 구현은 SYN 패킷 전송 6초 후에 SYN 패킷을 재전송한다. 원래는 12초로 규정되어 있는데[15] 6초는 BSD 코드의 버그 때문이다. 다음 SYN 재전송은 이전 SYN 전송 24초 후, 즉 처음 SYN 전송 30초 후에 이루어진다. 그림에서 6초 후에 증가가 거의 분간하기 힘들다는 것은 BSD 기반 TCP 구현은 요즘 거의 사용하고 있지 않다는 것을 시사한다. 요즘 TCP 구현 대부분은 RFC2988[16] 표준을 따른다. 3초에 첫째 빠른 증가는 RFC2988의 초기 RTO(Retransmission TimeOut)로 설명할 수 있다. 9초에서의 적은 증가는 2번째 재전송 (9=3+6)을 의미한다. 이 관찰로부터 대부분의 TCP 구현은 RFC2988을 따른다는 것을 알 수 있다. 또 그림 2에서 (t_{total} - t_{last})를 살펴보면 연결의 97% 이상이 SYN 재전송이 일어나지 않는다. 2% 정도가 SYN 재전송이 한번 이루어지며 2번을 초과하는 SYN 재전송은 극히 적은 부분이 차지한다는 것을 추정할 수 있다.

그림 2로부터 알 수 있는 또 다른 중요한 사실은 연결 설정 지연이 보통 1초보다 크지 않다는 것이다. 1초면 TCP 연결의 92%가 이루어진다. t_{last}는 SYN 재전송에 의한 시간 지연을 배제한 순수한 연결 설정 지연인데 그림 3에 그 분포를 나타내었다. t_{last}가 0.5초일 때 연결 완성 누적비율은 84.58%, 1초일 때 96.71%, 1.5초일 때 98.59%, 2초일 때 99.33%까지 올라간다. 이러한 분석으로부터 나온 결론은 다음과 같다. 첫째, SYN의 첫 전송이후 1초는 되어야 많은 연결 설정이 완료된다. 그 이하인 경우 연결 설정 완료 비율이 현저히 떨어진다. 둘째, 대부분의 연결에서 SYN-ACK 교환 왕복은 2초 이내면 완료한다. 본 트레이스가 장거리 연결에 대한 자료임을 고려할 때 통계에 단거리 연결 (예를 들어 한국내의 연결)을 포함한다면 2초일 때 연결 설정 완료 비율은 더욱 높아질 것이다.

5. TCP 연결 타임아웃 가이드라인

4절에서 살펴본 TCP 연결 설정 시간 분포는 RFC2988에 명시된 TCP SYN 재전송 행위에 의해 큰 영향을 받는다는 것을 알 수 있다. 이 절에서는 그림 2와 그림 3의 분포 분석과 RFC 2988에 기초하여 몇 개의 타임아웃 값을 선택하고 그 영향을 조사한다. 이전에

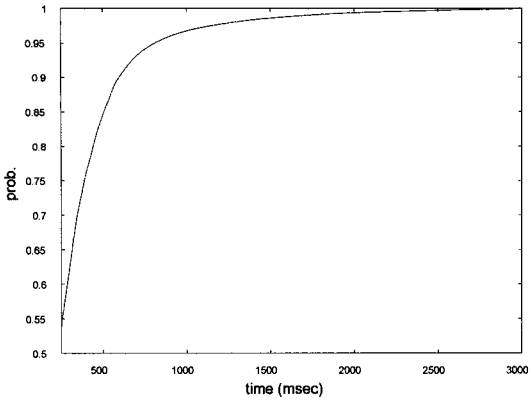

그림 3 SYN 재전송 시간을 제외한 연결 설정 지연 누적 분포

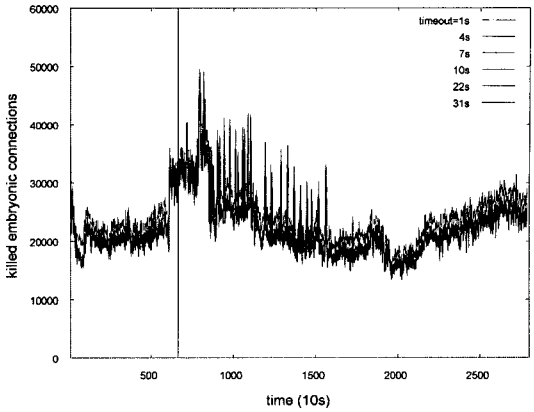

그림 4 제거되는 세션 엔트리 수

가정한 바와 같이 패킷 인스펙션 컴퓨터는 새로운 TCP SYN 패킷 마다 하나의 세션 엔트리를 만든다. 그리고 나서 이 플로우에 대한 패킷이 도착할 때 마다 이 연결의 진행 상태를 갱신, 기록한다. 초기의 미완성 상태에서 일정 시간이 지나면 해당 엔트리는 삭제된다. 우선 그림 3에서의 관찰을 기초로 하여 t_{last} = 1로 설정한다. 보다 높은 연결 설정 완료 비율(completion rate)을 위하여 타임아웃 시간을 더 늘릴 수 있지만 그림 3에서 본 바와 같이 값을 2초로 바꾼다 하여도 연결 완료 비율은 단지 2.5% 증가할 뿐이다. 또한 타임아웃을 추가적으로 1초 증가시킬 때마다 초당 공격 패킷 수만큼 더 많은 미완성 엔트리가 만들어지게 되므로, 발생하는 위험도에 비해 얻는 이익이 미미하다.

표 2는 RFC2988과 BSD기반 구현을 고려하여 선정한 몇 개의 타임아웃 값이 연결 완료 비율에 끼치는 영향을 보여준다. BSD기반 구현은 연결설정 타이머가 75초에, 즉 4번째 SYN 전송 3초 전에 만료되기 때문에 총 3번의 SYN 전송만이 허용된다. RFC2988이든 BSD이든 타임아웃 값 τ=10 이면 연결 완료 비율이 1에 가깝다. 또한 $4 \leq \tau \leq 10$ 에서는 약간의 변화만을 보인다. 예를 들면 τ를 4에서 7로 바꾼다면, 즉 BSD기반 시스템에 대하여 한번의 SYN 재전송을 허용한다 해도 이 추가적인 3초 동안 0.57%의 연결 완료 비율이 증가를 얻을 뿐이다. 반면 그림 2에서 보았듯이 τ는 1보다 적으면 연결 설정에 상당히 크게 불리한 영향을 주게 된다. 이와 같은 분석을 통하여 얻어진 연결 설정 시간 타임아웃 값에 대한 가이드라인은 다음과 같다.

타임아웃 값은 (R+T)로 지정하되 R은 SYN 재전송 허용 회수에 따라 0,3,9를 선택하고 T는 $1 \leq T \leq 2$ 를 만족시키는 값 중에서 허용하고 싶은 왕복 지연에 따라 선택한다.

예를 들면 디폴트 타임아웃 구성은 4(즉, R=3, T=1)로 줄 수 있다. 이 가이드라인 하에서 스테이트풀 패킷 인스펙션 컴퓨터는 주어진 목표 완료 비율을 달성할 때까지 타임아웃 값을 증가시킬 수 있다. 반대로 동적 메모리 이용률이 경계점(threshold)를 넘게 되면 이용률이 경계점 아래로 내려가도록 타임아웃 값을 감소시킬 수 있다.

그림 4는 위의 가이드라인이 그 도출에 사용된 트레이스에 적용되었을 때 세션 테이블의 크기에 주는 영향을 나타낸다. 주기적으로 세션 테이블을 검사하여 타임아웃이 된 미완성(embryonic) 연결 단계에 있는 엔트리를 제거하고 제거된 엔트리 수를 기록한다. 그림 4에 나

표 2 연결 타임아웃과 연결 설정 완료 비율

최대 허용 SYN 재전송 회수	RFC2988-conformant		BSD-derived	
	Timeout	Completion rate	Timeout	Completion rate
0	1s	93.07%	1s	93.07%
1	4s	98.92%	7s	99.55%
2	10s	99.86%	31s	99.99%
3	22s	99.99%	-	-

타난 날카로운 스파이크(spike)들은 DoS 공격 시도들이다. 나머지는 정상 트래픽과 스캔 트래픽으로 볼 수 있다. (트레이스를 살펴보면 약한 DoS 공격과 스캔은 거의 매 분 관찰된다[17]). 그림에서 우리는 타임아웃 값이 클수록 삭제되는 연결이 적어지고 따라서 더 많은 연결 설정이 완료되지만 지워지는 연결의 절대적인 수에 비교하면 그 차이가 크지 않음을 관찰할 수 있다. 그 이유는 표 1에서 살펴본 바와 같이 1초 이후에는 타임아웃이 더 길어져도 연결 완료 비율 증가는 미미하기 때문이다. 즉 대부분의 삭제되는 미완성 엔트리들은 아무리 τ를 증가시키더라도 연결을 완성하는 상태에 결국 도달하지 못한다는 것을 보여준다.

대조적으로 세션 테이블의 요구되는 크기는 τ 값에 따라 크게 달라진다. 그림 5는 타임아웃 값에 따른 세션 테이블 크기를 나타낸다. 아래로부터 각 곡선은 τ를 1, 4, 7, 10, 22, 31 로 변화시키면서 살펴본 시간에 따른 세션 테이블 크기이다. 최악의 경우 τ가 31일 때 엔트리의 개수는 τ가 1일 때의 수의 14배, τ가 4일 때의 6배가 된다. 따라서 τ 값이 적을수록 공격 트래픽 하에서 더 잘 견딜 수 있다는 것을 보인다. 다시 말하면 DoS 공격은 타임아웃이 길어질수록 더 강력하게 인스펙션 컴퓨터에 영향을 준다는 것이다. 그 이유는 세션테이블의 엔트리의 개수가 τ값에 비례하고 또한 공격의 강도에 비례하기 때문이다. 즉 xt를 시간 t에 있어서의 합법적인 연결 엔트리의 수이고 λ는 공격 속도라 할 때 세션테이블에서 시간 t에 존재하는 총 엔트리 수 c_t는

$$c_t(\tau) = x_t + \lambda\tau \qquad \text{(식 1)}$$

여기에서 xt는 거의 τ의 함수라고 할 수 없다. 즉, 타임아웃 크기는 합법적으로 연결된 엔트리 수에는 거의 영향을 주지 않는다. 왜냐하면 표 1에서 보는 바와 같이 합법적인 연결은 대부분 타임아웃 이전에 설정되기 때문이다. 식 우변의 둘째 항은 공격이 있을 경우에만 0이 아닌 값을 가진다.

식 1과 그림 5로부터 세션 테이블에서 공격 플로우가 차지하는 비율을 추정할 수 있다. 예를 들면 그림 5에서 t=800 일 때 DoS 활동이 두드러진다. 이 시점에서 그림 5에서 주어지는 $c_t(1)\approx10{,}000$과 $c_t(10)\approx55{,}000$을 (식 1)에 대입하면 다음과 같다.

$$x_t + \lambda = 10{,}000$$
$$x_t + 10\lambda = 55{,}000$$

이 연립 방정식을 풀면 $x_t=\lambda=5{,}000$ 가 구해진다. 이것은 τ=1일 때조차도 공격에 의한 엔트리 개수가 5,000(=λτ)으로 세션 테이블의 반을 차지하고 있음을 의미한다. (식 1)에 $x_t=\lambda=5{,}000$, τ=31을 대입하면 $c_t(31)=160{,}000$로 그림 5의 실제 측정치와 3%정도 오차가 있다. t=800에서 10초간 삭제되는 추정 엔트리 개수는 10λ=50,000으로 그림 4의 측정치와 거의 같다.

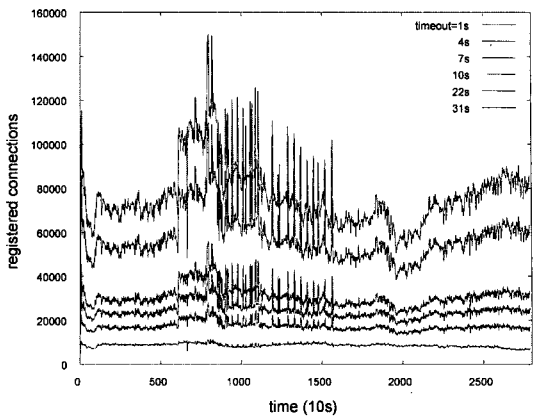

그림 5 타임아웃 값에 따른 세션 테이블 크기

본 트레이스에서는 공격의 강도가 비교적 낮은 편이다. 가장 강력한 공격도 공격 비율이 5,000 패킷을 넘지 않는다 [17]. 하지만 잘 조직된 분산 DoS 공격이나 대규모 웜 에피데믹 (epidemic) 트래픽에 노출되면 세션 테이블 크기의 증가는 감당하기 힘들 수 있다. 예를 들면 SQL Slammer 전염 시와 같이 초당 26,000 공격 패킷에 강타당하는 10개의 감염 호스트가 있으면 τ=100일 때 2천6백만 공격 엔트리가 세션 테이블을 차지하게 된다. 여기서 말하고자 하는 요점은 이것이다. 설정 완료 비율을 증가시키고자 TCP 플로우의 미완성(embryonic) 상태를 더 오래 허용한다면, 연결 설정 비율은 실상 거의 높이지 못하면서 스테이트풀 패킷 인스펙션 컴퓨터만 메모리 고갈과 룩업 (lookup) 성능 저하의 위험에 처하게 된다는 것이다. 따라서 연결 설정 미완성 상태가 추천한 타임아웃인 4초 또는 10초 이상 지속되면 즉시 삭제하는 것이 바람직하다.

6. 결 론

스테이트풀 패킷 인스펙션은 방화벽, VPN, 침입탐지, 네트워크 모니터링, 트래픽 측정 등 의 네트워크기반 장비나 연산에서 사용이 증가하고 있다. 네트워크 공격이 더 빈번해지고 더 강력해질수록 스테이트풀 인스펙션은 메모리 급증에 취약하다. 이 논문은 네트워크 공격에 대하여 지속적으로 동작할 수 있도록 패킷 인스펙션 컴퓨터를 위한 메모리 관리를 위한 기술적인 가이드라인을 제공한다.

네트워크 공격은 각 공격 패킷마다 세션 테이블에 엔트리를 만들도록 하는데, 공격 패킷들은 각자 하나의 플로우로 인식되고, 더우기 그 도착 비율이 보통 높기 때문에 스테이트풀 패킷 인스펙션 컴퓨터에 문제를 일으

킨다. 엄격한 가이드라인에 의하여 이 현상을 통제하지 않는다면 네트워크 공격에 직면하여 메모리 오버플로우에 의한 스테이트풀 인스펙션 기능이 중단될 위협에 놓이게 된다. 이 공격이 스테이트풀 인스펙션 장비에 대한 공격이 아닐지라도 이 현상은 발생한다.

본 논문에서는 TCP 프로토콜 명세와 실제 패킷 트레이스 분석을 기초로 하여, 상대적으로 짧은 타임아웃이 공격에 의해 생성된 세션 엔트리를 효과적으로 제거하는데 사용할 수 있다는 것을 보였다. 명시적으로 4 내지 5초의 타임아웃은 정상 연결 완료 비율이 99%를 넘으면서 세션 테이블 크기를 제한하기에 충분하다. 다시 말하면 10초보다 긴 타임 아웃 값은 공격 하의 메모리 오버플로우의 기회를 증폭시키는 반면, 1초보다 적은 타임아웃은 정상적인 플로우에 대한 연결 완료 비율을 현저하게 떨어뜨린다. 따라서 일반적으로 타임아웃은 (R+T)로 지정하되 R은 SYN 재전송 허용 회수에 따라 0,3,9를 선택하고 T는 $1 \leq T \leq 2$중에서 선택하여 (R+T) 이상 경과된 비활성의 미완성 TCP 연결을 세션 테이블에서 제거하도록 스테이트풀 인스펙션 컴퓨터를 구성하도록 가이드라인을 제시한다. 이렇게 함으로써 스테이트풀 인스펙션 컴퓨터의 메모리를 효율적으로 사용하고, 룩업 성능을 유지하며, 공격 하에서도 계속 동작하게 할 수 있다.

참 고 문 헌

[1] *Stateful-inspection firewalls: The Netscreen way*, white paper, http://www.netscreen.com/products/firewall_wpaper.html

[2] G. Iannaconne, C. Diot, I. Graham, N. McKeown, "Dealing with high speed links and other measurement challenges," Proceedings of ACM Sigcomm Internet Measurement Workshop, 2001.

[3] K. Claffy, G. Polyzos, and H.-W. Braun, "A parametrizable methodology for Internet traffic flow monitoring," *IEEE JSAC* 8(13), Oct. 1995, pp.1481-1494.

[4] H.-W. Braun, K. Claffy, and G. Polyzos, "A framework for flow-based accouting on the Internet," Pr oceedings of IEEE Singapore International Conference on Information Engineering, 1993. pp. 847-851.

[5] V. Srinivasan, G. Varghese, S. Suri, M. Waldvogel, "Fast Scalable Algorithms for Level Four Switching," Proceedings of ACM Sigcomm, 1998.

[6] L. G. Roberts, "Beyond Moore's Law: Internet Growth Trends," *IEEE Computer*, 33 (1), Jan. 2000, Page(s): 117 -119

[7] P. Gupta and N. McKewon, "Packet classification on multiple fields," Proceedings of ACM Sigcomm, 1999.

[8] F.Baboescu and G.Varghese,"Scalable packet classification,"Proceedings of ACM Sigcomm, 2001.

[9] S. Singh, F. Baboescu, G. Varghese and J. Wang, "Packet Classification Using Multidimensional Cuts," Proceedings of ACM Sigcomm 2003.

[10] Gill, "Maximizing firewall availabilty," http://www.qorbit.net/documents/maximizing-firewall-availability.htm

[11] IP Monitoring Project at Sprint, http://ipmon.sprint.com/ipmon.php

[12] R. Stevens, *TCP/IP Illustrated Vol. 1.* Addison-Wesley, 1994.

[13] V. Paxson and M. Allman, Computing TCP's retansmission timer, RFC 2988, Nov. 2000.

[14] H. Kim, "Dynamic memory management for packet inspection computers," techreport, http://ubiquitous.korea.ac.kr/lifetime.html

[15] K. Houle and G. Weaver, "Trends in denial of service attack technology," a CERT paper, http://www.cert.org/archive/pdf/DoS_trends.pdf, Oct. 2001.

[16] IANA, "Internet protocol V4 address space," http://www.iana.org/assignments/ipv4-address-space.

[17] P. Vixie (ISC), G. Sneeringer (UMD), and M. Schleifer (Cogent). Events of 21-Oct-2002. November 24, 2002.

[18] D. Moore et al., "The spread of Sapphire worm," techreport, http://www.caida.org/outreach/papers/2003/sapphire/sapphire.html, Feb. 2003.

[19] M. de Vivo, E. Carrasco, G. Isern, and G. de Vivo, "A review of port scanning techniques," *ACM Computer Communication Review*, 29(2), April 1999.

[20] NLANR, "NLANR network traffic packet header traces," http://pma.nlanr.net/Traces/

김 효 곤

1987년 서울대학교 전자계산기공학과(공학사). 1989년 서울대학교 전자계산기공학과(공학석사). 1995년 University of Pennsylvania Computer and Information Science(공학박사). 1996년~1999년 Bell Communications Research 연구원. 1999년~2003년 아주대학교 조교수. 2003년~현재 고려대학교 부교수. 관심분야는 무선통신, 인터넷 보안

강 인 혜

1987년 서울대학교 전자계산기공학과(공학사). 1989년 서울대학교 전자계산기공학과(공학석사). 1997년 University of Pennsylvania Computer and Information Science(공학박사). 1998년~2000년 숭실대학교 컴퓨터학부 객원교수. 2000년~2002년 (주) 시큐아이닷컴 정보보호연구소 부장. 2002년~현재 서울시립대학교 기계정보공학과 조교수. 관심분야는 정형기법, 소프트웨어공학, 인터넷 및 소프트웨어 보안