제 19 회 한국정보처리학회 춘계학술발표대회 논문집 제 10 권 제 1 호 (2003. 5)

세션 학습을 이용한 방화벽 가속기의 구현

김경수*, 김종수*, 문종욱*, 정기현*, 최경희**
*아주대학교 전자공학과, **아주대학교 정보통신전문대학원

e-mail : girle999@yahoo.co.kr

Implementation Firewall Accelerator using Session Learning

Kyoungsoo Kim*, Jongsu Kim*, Jongwook Moon*,
Gihyun Jung*, Kyunghee Choi**
*Dept. of Electronics Engineering, Ajou University
**A professional Graduate School for Info. and Comm. Engineering, Ajou University

요 약

인터넷의 급속한 발전과 해킹사고의 발생율이 증가함에 따라 방화벽의 사용이 일반화 되고 있다. 과도한 트래픽이 방화벽을 지나게 되면 방화벽 자체 성능에 따라 처리되지 않은 패킷을 잃어버리거나 패킷을 재전송 해야 하므로 추가 트래픽이 발생한다. 이에 대한 방안으로 소프트웨어 또는 하드웨어적으로 방화벽의 성능을 높이는 방법이 있지만, 높은 비용 및 호환성 문제가 있다. 이의 다른 대안으로 방화벽 가속을 하는 방법이 있는데 기존의 연구 모델에서는 특정 방화벽과 연동하거나 기기 설정을 해야 하는 번거로움이 있었다. 본 논문에서는 어떤 방화벽과도 서로 연동될 수 있고 관리자의 관리 추가 설정 없이도 동작하도록 방화벽의 세션을 스스로 학습하여 방화벽 성능을 높이는 방식의 방화벽 가속기를 제안하고 패킷 처리 성능을 올릴 수 있음을 실험을 통해 증명 하였다.

1. 서 론

인터넷 사용자와 이를 이용하는 응용프로그램이 점점 많아짐에 따라 네트워크 트래픽도 크게 증가하고 있으며, 트래픽의 증가율 이상으로 보안사고 발생율도 크게 높아지고 있다. CERT/CC 에 의하면 2002 년에는 약 82000 건의 보안 사고가 보고되었는데 이는 1998 년 대비 20 배나 높아진 수치이다.[1] 이러한 보안사고에 대처하기 위한 가장 간단하고 보편화된 방법으로 방화벽을 많이 사용한다. 현재 기업 규모별 방화벽 도입율을 보면 500 명 이상의 기업 중 80% 이상이 이를 설치 운영하고 있는 것으로 조사되고 있다.[2]

방화벽은 네트워크의 내부와 외부를 연결하는 지점에 설치함으로써 내부 호스트들을 보호하게 되는데, 네트워크에 흐르는 트래픽이 방화벽에서 처리할 수 있는 양보다 많아지게 되면 방화벽에서 병목현상이 발생하여 TCP (Transmission control protocol)의 경우에는 패킷 재전송이 발생하게 되고, UDP(User datagram protocol)의 경우에는 패킷을 잃어버리게 되어 서비스의 질을 떨어뜨리는 문제점이 있다.

최근에는 트래픽을 급격히 증가시키는 웜(Worm)들이 확산되고 있어 방화벽에도 과부하 요인으로 작용하는 경우가 자주 발생되고 있다.[3][4]

일반적으로 방화벽을 도입할 당시에는 네트워크의 트래픽 부하량을 견딜 수 있게 충분한 성능의 장비를 채택하지만 시간이 지나 트래픽이 점점 늘어나게 되면 곧 한계에 이르게 된다. 이러한 상황이 되면 방화벽의 성능을 높이기 위해서 장비의 교체가 필요하게 되는데 네트워크 사용량의 증가 속도로 보아 장비의 교체 주기가 짧아지고 장비가 비교적 고가이므로 큰 부담으로 작용한다.

앞에 언급한 대로 방화벽의 성능 향상을 위한 가장 쉬운 방법은 더 좋은 고성능 방화벽을 설치하거나 동급의 방화벽을 부하 분산 장비와 함께 여러 개 연결하여 사용하는 것이다. 그러나 이는 큰 투자비용이 들어가게 되는 단점이 있으므로, 기존 방화벽의 성능

을 높여 활용하는 방법도 경제적 측면에서 연구되고 있다.

본 논문에서는 위 연구의 일환으로 방화벽의 보조 장치로써 방화벽 가속 시스템 구조를 제안하고, 기존 가속 시스템들의 단점이었던 방화벽-방화벽 가속기의 연동 제한 문제나 기기 설정 불편 문제들을 개선한 방화벽 가속 시스템을 제한한다. 현 모델에서는 패킷의 세션(Session)을 스스로 학습하여 관리자가 접근 규칙(Access Rule) 을 따로 설정할 필요가 없으며, 방화벽의 작업을 일부 분담하여 처리함으로써 병목현상을 방지 하게 한다.

서론에 이어 2 장에서는 방화벽 가속기와 연동될 수 있는 방화벽의 운영 방식과 기존의 방화벽 가속 시스템들에 대해 언급하고, 3 장에서는 현 논문에서 제안하고 있는 개선된 방화벽 가속 시스템의 모델에 대해 설명한다. 4 장에서는 방화벽 가속 시스템이 적용된 네트워크에서 성능향상 정도를 실험한 내용을 나타내고 있고, 5 장에서는 결론과 향후 개선 사항에 대해서 서술한다.

2. 관련 연구

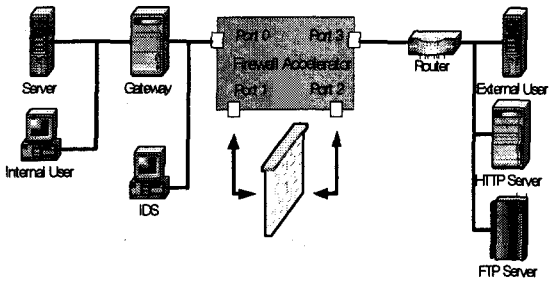
일단 방화벽의 동작 방식부터 살펴 보면 크게 패킷 여과(Packet Filtering)방식과 스테이트풀 인스펙션(Stateful Inspection) 방식으로 나눌 수 있다. 패킷 여과 방식은 외부 네트워크로부터 들어오는 침입에 사용될 수 있는 패킷을 막기위해서 인터넷 주소, 포트, 프로토콜 등을 이용하는 방법으로써 기본 정책은 내부에서 외부로 나가는 패킷에 대해서는 통과시키고, 반대의 경우는 막는 것이다. 스테이트풀 인스펙션은 모든 응용 프로그램에 따른 전후 문맥을 추출하여 보안과 관련된 플래그를 저장하고 세션에 따라 보안을 강화한 방법이다. 이는 앞의 방식보다 더 진보적인 방식으로 현재 개발되는 대부분의 방화벽들은 이를 활용하고 있다. 스테이트풀 인스펙션을 이용하여 보안 처리는 강화되었지만 역시 과도한 네트워크 트래픽을 처리하는 데는 아직까지 문제가 있다.

네트워크 트래픽의 처리 효율을 높이기 위한 기존의 방법을 살펴보면 다음과 같다. 첫째, 고성능 서버에 방화벽을 설치하여 패킷 처리율을 높이는 방법이다. 시간당 처리하는 패킷의 양은 늘어나지만 최신 모델 구입에 따른 비용이 많이 든다. 둘째, 스위치나 세션기반 부하 분산 장치(Load Balancer)와 여러 대의 방화벽을 이용하여 트래픽의 부하분산을 하는 것이다. 비용이 저렴한 기존 장비를 사용할 수 있으나 구성이 복잡하고 추가 장비를 필요로 한다. 셋째, 기존의 방화벽 가속 시스템을 사용한다. 이는 세션 정보를 방화벽과 방화벽 가속기가 공유하도록 하여 방화벽이 처리한 세션에 대해서는 방화벽 가속기가 직접 처리하도록 하여 방화벽의 부담을 줄여 주는 방법이다.[5] 이를 이용하면 속도 향상 뿐만 아니라 새로운 방화벽의 구입 비용보다 저렴하므로 경제적이다. 실제 실험적인 하드웨어(FPGA)로 구현되어 좋은 성능을 보인 사례도 보인다.[6] 그러나 이러한 방법에서는 두 장치 사이에 전용의 메시지 패스를 두어 연동하게 하므로 타 방화벽들과의 호환성에 문제가 있어 선택에 제약이 있다.

다른 방법으로 방화벽과 프록시 서버가 있을 때 인증이 끝난 세션에 대해서 상위 응용프로그램으로 가는 패킷에 대해서 소프트웨어적으로 스택(Stack)의 하위 계층에서 가로 채어 포워딩(Forwarding)하는 것도 있다.[7][8] 또한 패킷 필터링 방식이 모든 패킷을 검사하기 때문에 성능이 떨어진다는 점에 착안하여 세션이 맺어진 이후에만 특정 필드를 검색하여 성능을 높이는 방법도 있으나, 이는 S/W 를 변경 시켜야 하므로 방화벽을 직접 개발하는 곳이 아닌 이상 구현이 힘들게 된다.

이러한 연구로부터 비용을 절감하면서 보안 효과를 높일 수 있는 적합한 방법으로써 위의 세 번째 방식을 선택한 후 단점이 보완된 시스템을 설계하였다. 여기에서는 패킷을 받아들일 때 방화벽이 통과시킨 세션을 가속기가 학습하여 방화벽과 같은 보안 수준을 공유 할 수 있다.

3. 제안모델



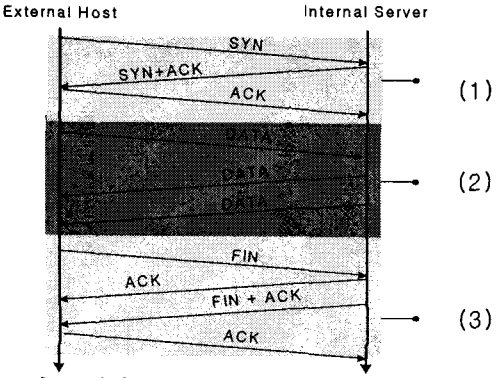
<그림 1. 방화벽 가속기의 배치 모델>

본 연구에서 제안하는 세션 학습 방식을 이용한 방화벽 가속기는 네트워크 내에서의 그림 1 과 같은 구조로 배치되어 있다. 일반적인 방화벽의 구성을 예로 들어보면 그림에서 왼쪽은 보호되어야 할 내부 네트워크(인트라넷; Intranet)이며 오른쪽은 외부 네트워크(인터넷; Internet)이다. 방화벽 가속엔진은 패킷이 방화벽으로 들어가기 전과 들어간 후를 비교하여 세션을 학습할 수 있도록 하기 위해 그림과 같이 병렬로 설치한다. 어떠한 스테이트풀 인스펙션 기반 방화벽이든 하드웨어와 관계없이 방화벽 가속기의 포트 1 번과 2 번에 연결될 수 있다. 포트 0 번은 내부 네트워크 인터페이스이며, 포트 3 번은 외부 네트워크 인터페이스를 나타낸다.

3. 1 학습 알고리듬

방화벽 가속기가 관리자의 설정없이 방화벽에 바로 연동되기 위해서는 방화벽이 통과시킨 패킷을 분석하여 학습하는 과정이 필요하다. 그림 2 는 네트워크에서 양 단말 장치 사이에서 TCP 세션이 맺어 질 때의 패킷 흐름과 주요 플래그(Flag) 상태를 나타낸다.

방화벽은 이들의 중간에 위치하면서 세션의 상태나 패킷의 접근 가부를 확인하므로 실제 성립된 세션은 보안상 문제가 없는 연결이 된다. 그림에서 (1)은 외부네트워크의 호스트가 내부네트워크의 서버에 접속할 때 이루어지는 3-Way Handshaking 을 나타내고 있다. 방화벽은 이 시기에 패킷을 검사하고 조건이 허용된 경우에 하나의 세션 엔트리(Entry)를 생성하고 내부 네트워크와 외부 네트워크의 연결을 허용한다. 이 때 방화벽 가속기는 방화벽으로 들어가는 패킷과 방화벽을 통과하여 나오는 패킷들의 정보를 계속 추적하여 연관성을 찾고 3-Way Handshaking 이 완전히 찾아진 것에 대해서 역시 하나의 세션으로 등록하게 된다.



<그림 2. 방화벽을 통과하는 TCP 패킷의 흐름>

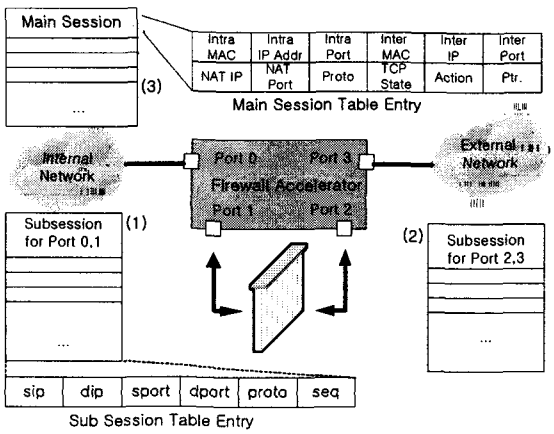
(2)에 해당하는 패킷들은 방화벽 자체에서 인증된 연결을 통하여 데이터가 오가는 것이므로 방화벽으로 패킷을 보내지 않고 방화벽 가속기가 곧바로 반대편 네트워크에 연결된 출력포트로 보내게 된다. 일반 데이터 패킷들은 이러한 방식으로 통과하므로 방화벽이 처리해야 할 작업이 상당부분 줄게 되고 그 만큼 방화벽 가속기가 작업을 분담하게 되므로 방화벽의 부하 분산이 이루어 지게 된다.

(3)에서는 외부호스트가 서비스를 끝마치기 위해서 TCP 계층에서 FIN 플래그를 보내어 세션을 닫는 것을 나타낸다. 방화벽 가속기는 패킷을 방화벽으로 보내어 세션을 닫을 수 있도록 해주고 방화벽 가속기 내에서도 이 세션 테이블에서 해당 세션 정보를 삭제함으로써 탐색량과 메모리 사용량을 줄일 수 있도록 한다.

3. 2 소프트웨어 구성

앞 절에서 언급한 대로 방화벽 가속기가 관리자에 의해 따로 설정되지 않고 동작하기 위해서는 학습한 세션에 대한 비교가 이루어져야 한다. 그림은 크게 3 개의 테이블을 필요로 한다. 우선 내부 네트워크와 연결되는 포트 0, 1 번에 대한 관계 테이블(1), 외부 네트워크와 연결되는 포트 2, 3 번에 대한 관계 테이블(2), 그리고 이 둘을 통합하는 통합 세션 테이블이 필요하다.

(1), (2)의 Sub-session 테이블에는 패킷이 지날 때 마다 외부와 내부로 지나가는 패킷의 정보를 기록하기 위해 출발지 IP, 목적지 IP, 출발지 포트, 목적지 포트, 프로토콜의 정보를 테이블에 써 넣는다.

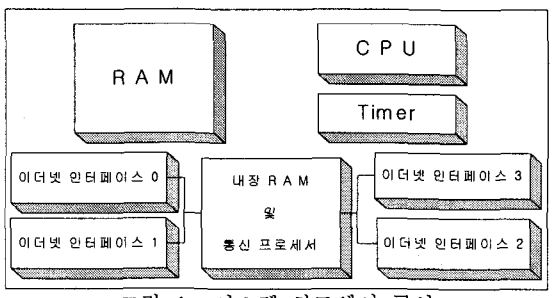


<그림 3. 방화벽 가속기 세션 관리테이블>

근래 방화벽의 보안 효과를 높이기 위해 NAT (Network Address Translation)가 많이 사용되는데 이에 의해 주소가 변경되어도 패킷 내의 (1), (2)의 테이블 내의 여러 정보를 통해 같은 패킷 임을 확인 할 수 있다. 패킷 입력시 *Sub-session* 테이블의 검색을 위해 해쉬 함수(Hash Function)를 적용하는데 패킷의 세션이 찾아지면 해당 엔트리에 정보를 갱신하거나 생성한다. 일단 완전한 3-Way handshaking 후에는 Sub-session 테이블의 해당 엔트리가 주 세션(Main Session) 테이블의 엔트리에 등록되게 된다. 이 테이블은 실제 변경되는 정보(MAC, Address, NAT 정보 등)를 방화벽 가속기에서 고치기 위한 정보를 제공하게 된다. 또한 테이블 검색에 필요한 TCP State, 프로토콜과 등 패킷에 대한 허용 여부(허용, 폐기)를 표기한 Action 등도 포함되어 있다.

주 세션테이블이 생성되면 이 후 같은 세션에 해당하는 데이터들은 주 세션 테이블에 의해 검색되어 방화벽을 거치지 않고 직접(0 번 포트 ←→ 3 번 포트) 전송된다.

3. 3 하드웨어 구성



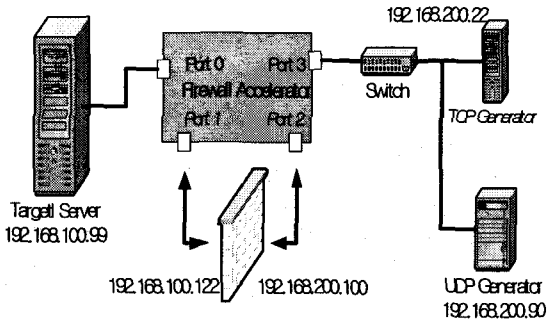
<그림 4. 시스템 하드웨어 구성>

시스템의 구성을 보면 인터넷 패킷을 실시간으로

처리하기 위해서 전용의 통신 프로세서를 이용하고 있다. 이것은 내부에 DPRAM(Dual Port RAM)을 내장하고 있어서 패킷을 초고속으로 처리 하기 알맞은 구조로 되어있다. 또한 타이머(Timer)는 열려진 세션을 정해진 시간이 초과했을 경우 메모리 절약을 위해 해당 엔트리를 폐기하기 위해서 사용한다.

4. 성능 평가

방화벽의 패킷 처리 성능과 방화벽 가속기 자체의 성능을 보이고, 방화벽과 방화벽 가속기를 연동했을 때의 패킷처리 능력을 비교하기 위하여 그림 5 와 같은 실험 환경을 구성하였다. UDP 생성기(UDP Generator)에서는 UDP 패킷을 순차적으로 생성할 수 있는 MGEN 을 이용하여 72Byte 의 패킷을 내부 네트워크에 있는 대상 서버(Target Server)로 방화벽을 통하여 전송한다. TCP Flooding 생성을 위한 TCP 생성기는 SYN Flooding 도구를 이용하여 64Byte 크기의 패킷으로 대상 서버와 다른 세션을 생성하도록 한다. 방화벽은 스테이트플 인스펙션 기능을 수행할 수 있는 어울림정보통신의 SECUREWORKS 를 사용하였다.

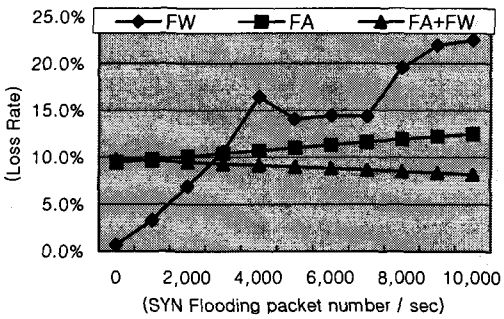


<그림 5. 실험 환경>

그림 6 의 FW(Firewall), FA(Firewall Accelerator)는 각각 방화벽, 방화벽 가속기 자체의 패킷 처리 능력만을 실험한 값이고, FW+FA 는 방화벽 가속기를 방화벽에 연동하여 실험한 값이다. Y 축의 손실율(Loss Rate)은 TCP SYN Flooding 패킷을 초당 0 개부터 10,000 개까지 생성해 가면서, UDP 패킷을 초당 90,000 개 전송했을 때, 보낸 패킷에 대한 받은 패킷의 비율을 측정한 것이다.

그림 6 의 그래프에서 보듯이 방화벽 자체만을 사용하였을 경우(FW) 세션의 개수가 늘어나면서 정상적인 서비스를 대표하는 UDP 전송에서 손실률이 급증하는 것을 볼 수 있다. 그리고 두 개의 시스템(FW, FA)을 연동하여 동작시켰을 때에는 10%의 UDP 패킷 손실율에서 시작하여 세션이 늘어 남에 따라 패킷 처리량이 점점 증가 함을 알 수 있다.

전체 실험 결과를 보면 세션이 3,000 개 이상 존재할 때 방화벽 가속기를 연동하여 사용하면 방화벽을 통과하는 트래픽이 증가하여 망의 운영 효율이 늘어남을 알 수 있다.



<그림 6. 실험 결과>

5. 결론 및 향후 연구 과제

본 논문에서는 방화벽과 같은 보안 수준을 유지하면서 패킷을 고속으로 처리하여 전송할 수 있도록 하기 위한 방법으로 방화벽 가속 시스템을 제안하였다. 방화벽 가속기는 방화벽이 통과 시킨 세션에 대해 스스로 학습하여 이후 해당 세션에 대해서는 방화벽의 부하를 분담 하여 주어 전반적인 성능을 향상 시켰으며, 추가적인 장치 설정과 같은 번거로운 일을 없애 편의성과 호환성을 갖도록 하였다.

현재의 방화벽 가속기에서도 패킷을 학습하는 동안에는 패킷 처리 손실이 발생하고 있다. 향후에는 패킷이 움직이는 경로를 더욱 최적화 하고, 테이블 탐색의 속도를 높일 수 있는 방법과 방화벽의 진화에 맞춰 현재 4 계층 정보만 지원하는 것에서 더 많은 정보를 포괄하도록 하는 연구를 지속해야 할 것이다.

참고문헌

[1] CERT/CC " CERT/CC Statistics 1988-2002 " http://www.cert.org, 2003

[2] KGB, " 국내 방화벽 시장" , http://www.krgweb.com, 2002

[3] Prabhar Kumar Singh " A Physiological decomposition of virus and worm programs" Mater degree Thesis, Louisiana University, Spring 2002,

[4] Eugene H. Spafford" The Internet Worm Program: An Analysis" Purdue University West Lafayette, IN 47907-2004, Purdue Technical Report CSD-TR-823, 1988

[5] Alteon Switched Firewall " Installation and User' s Guide" http://www.nortelnetworks.com/, 2003

[6] John T. McHenry, Patrick W. Dowd " An FPGA-based Coprocessor for ATM Firewalls" IEEE Symposium on FPGAs for Custom Computing Machines, pp.30 April,1997

[7] David Maltz " TCP Splicing for Application Layer Proxy" , IBM Research Report, RC 21139 Computer Science/Mathematics, 1998

[8] Oliver Spatscheck, Jorgen S. Hansen, John H. Hartman and Larry L. Peterson " Optimizing TCP Forwarder Performance" , IEEE/slash ACM Transactions on Networking, 2000