인위적인 네트워크 혼잡으로부터 정상 트래픽의 서비스 품질을 보호하기 위한 소수자 우선 게이트웨이

정회원 안 개 일*

Minority First Gateway for Protecting QoS of Legitimate Traffic from Intentional Network Congestion

Gae-Il Ahn* ***Regular Member***

요 약

서비스 거부 공격은 네트워크 자원을 독점하여 서버 시스템 및 네트워크상에 인위적인 혼잡을 발생시키는 공격으로써, 일반 사용자가 정상적인 서비스를 제공 받지 못하도록 하는 것을 목적으로 한다. 본 논문에서는 인위적인 네트워크 혼잡 상황에서도 정상 트래픽의 서비스 품질을 보장할 수 있는 소수자 우선 게이트웨이를 제안한다. 소수자 우선 게이트웨이는 어떤 집합 플로우가 혼잡유발 트래픽인지를 빠르게 결정할 수 있는 방법을 제공한다. 소수자 우선 게이트웨이는 정상 트래픽으로 판정된 집합 플로우에게는 높은 우선순위의 서비스를 제공하여 그 품질을 보호하고, 혼잡유발 트래픽로 판정된 집합 플로우에게는 낮은 우선순위의 서비스를 제공하여 네트워크 혼잡이 혼잡유발 트래픽에게만 영향을 미치도록 한다. 제안하는 소수자 우선 게이트웨이는 서비스 거부 공격 뿐만 아니라 다중 노드에서 동시에 혼잡을 일으키는 분산 서비스 거부 공격에서도 정상 트래픽의 서비스 품질을 보장하는 효과를 제공함을 시뮬레이션을 통하여 확인하였다.

Key Words : Network Congestion, DDoS Attack, QoS, Protection of Legitimate Traffic

ABSTRACT

A Denial of Service (DoS) attack attempts to prevent legitimate users of a service from being adequately served by monopolizing networks resources and, eventually, resulting in network or system congestion. This paper proposes a Minority First (MF) gateway, which is capable of guaranteeing the Quality of Service (QoS) of legitimate service traffic under DoS situations. A MF gateway can rapidly determine whether an aggregated flow is a congestion-nducer and can protect the QoS of legitimate traffic by providing high priority service to the legitimate aggregate flows, and localize network congestion only upon attack traffic by providing low priority to aggregate flows regarded as congestion-inducer. We verify through simulation that the suggested mechanism possesses excellence in that it guarantees the QoS of legitimate traffic not only under a regular DoS occurrence, but also under a Distributed DoS (DDoS) attack which brings about multiple concurrent occurrences of network congestion.

Ⅰ. 서 론

인터넷은 우리 일상생활에서 없어서는 안될 만큼 대중화되었지만, 보안은 아직도 취약한 상태이다. 2003년에 발생했던 1.25 인터넷 사고는 우리에게 크나큰 사회적 경제적 손실을 입혔으며 보안의 중

* 한국전자통신연구원 네트워크보안 연구부(fogone@etri.re.kr)
논문번호 : KICS2005-02-078, 접수일자 : 2005년 2월 21일

요성을 다시금 일깨워 준 사건이다[1].

현재, 네트워크 공격 중에서 파괴력이 가장 큰 공격은 서비스 거부 공격이다. 서비스 거부 공격은 특정 시스템 및 네트워크상에 대량의 패킷을 생성하여 인위적으로 혼잡을 발생시킴으로써 정상 사용자의 서비스 품질을 크게 떨어뜨리는 공격이다[2][3].

혼잡을 제어하기 위한 기존의 전통적인 기법으로는 큐잉 기법[4], TCP 혼잡 제어 기법[5], 그리고 RED (Random Early Detection) 게이트웨이[6] 등이 있다. 이러한 전통적인 기법들은 혼잡을 감지하면 트래픽 전송양을 줄이는 정상 사용자를 대상으로 할 때는 효과가 있지만, 혼잡 발생을 목표로 인위적으로 트래픽을 생성하는 공격자의 경우에는 그 효과가 매우 미미한 문제가 있다.

서비스 거부 공격을 차단하기 위하여 몇 가지 기법들이 제안되었다. 특정 응용 트래픽의 대역폭이 정해진 최대값을 넘지 않도록 대역폭을 제한하는 정적 대역폭 제한 기법[7]이 있다. 또한 라우터간 통신을 기반으로 하여 공격 트래픽의 대역폭을 제한하는 동적 대역폭 제한 기법[8][9]이 있다. 정적 대역폭 제한 기법은 자원 효율성이 매우 떨어지는 문제가 있다. 라우터간 통신 기반의 기법도 새로운 통신 프로토콜의 구현을 요구하며 또한 정확한 대역폭 제한 값을 결정하는 데에 어려움이 있다.

본 논문에서는 인위적인 네트워크 혼잡 상황에서도 정상 트래픽의 서비스 품질을 보장할 수 있는 소수자 우선 (MF: Minority First) 게이트웨이를 제안한다. MF 게이트웨이는 서비스 거부 공격뿐만 아니라 다중 노드에서 동시에 혼잡을 일으키는 분산 서비스 거부 공격에서도 정상 트래픽의 서비스 품질을 보장할 수 있다.

본 논문은 다음과 같이 구성된다. 먼저, 2장에서는 서비스 거부 공격과 기존의 공격 방지 기법에 대하여 알아보고, 3장에서는 인터넷 트래픽을 분석 및 분류하고, 4장에서는 본 논문에서 제안하는 MF 게이트웨이를 소개한다. 5장에서는 실험을 통하여 본 논문에서 제안하는 기법의 성능을 평가한다. 마지막으로 6장에서 결론을 맺는다.

Ⅱ. 서비스 거부 공격 및 기존의 방지 기법

서비스 거부 공격이란 시스템이나 네트워크가 사용자에 대한 서비스의 제공을 거부하게 만드는 공격이다. 전형적인 공격 시나리오를 살펴보면, 먼저, 공격자는 인터넷상에 수많은 컴퓨터를 해킹하여 서비스 거부 공격에 필요한 에이전트 프로그램을 설치한다. 그 에이전트 프로그램은 공격자로부터 공격 명령을 받으면 목표 시스템을 향하여 엄청난 트래픽을 생성하여 혼잡을 유발시켜 결국 정상 트래픽의 서비스 품질을 크게 떨어뜨린다. 근래에는 스스로 다른 시스템을 감염시키고 또한 스스로 목표 시스템을 공격함으로써 공격자의 개입이 필요 없는 웜을 이용한 서비스 거부 공격 형태가 점차 보편화되고 있다.

서비스 거부 공격을 탐지하는 것이 어려운 이유는 공격자는 공격 탐지를 회피하기 위하여 공격 패턴을 쉽게 바꿀 수 있으며 또한 거짓 출발지 IP 주소를 사용하여 자신의 신분을 속일 수 있기 때문이다. 예를 들어, 특정 출발지로부터 전달된 패킷 양이 어떤 일정치를 넘으면 공격으로 간주하는 서비스 거부 공격 탐지 알고리즘은 출발지 주소를 속이면서 인위적인 혼잡을 발생시키는 지능적인 공격자를 탐지하는 것은 거의 불가능하다. 따라서, 서비스 거부 공격을 근본적으로 탐지하고 차단하기 위해서는 출발지 IP 주소 속임 문제와 혼잡 유발 트래픽 제어 문제가 모두 해결되어야 한다.

출발지 IP 주소 속임 문제를 해결하기 위하여 제안된 기존의 기법으로는 입출구 필터링(Ingress/Egress Filtering)[10][11], TTL 기반의 홉(hop) 수 필터링 기법[12] 등이 있다. 입출구 필터링은 내부 망에 할당된 주소가 아닌 패킷이 외부 망으로 갈 때, 그리고 내부 망에 할당된 주소를 갖는 패킷이 외부 망에서 들어올 때 이를 차단하는 기술을 말한다. TTL (Time-to-live) 기반의 홉수 필터링 기법은 각 출발지 주소별로 정상 홉 수를 구하면서, 정상 홉 수와 다른 TTL 값을 갖는 입력 패킷을 거짓 출발지 IP 주소를 갖는 패킷으로 간주하는 기법이다.

현재 실세계에서 가장 많이 사용되고 있는 기법은 uRPF(unicast Reverse Path Forwarding)[13] 이다. 이 기법은 수신 패킷의 출발지 주소와 입력 인터페이스 번호가 라우팅 테이블에 존재하는 지를 검사하여, 만약 발견되지 않는다면 거짓 출발지 IP 주소를 가진 패킷으로 간주한다. 이 기법을 사용함으로써 공격자는 자신이 속하는 네트워크 주소 이외의 IP 주소를 사용할 수 없으며, 따라서 공격자가 만들 수 있는 IP 주소의 범위는 한정된다.

혼잡을 제어하기 위한 기존의 전통적인 기법으로는 TCP 혼잡 제어 기법[5], 큐잉 기법[4], 그리고 RED 게이트웨이[6] 등이 있다. 이러한 전통적인 기법들은 혼잡을 감지하면 트래픽 전송양을 줄이는 정상 사용자를 대상으로 할 때는 효과가 있지

만, 혼잡 발생을 목표로 트래픽을 생성하는 공격자의 경우에는 그 효과가 매우 미미한 문제가 있다.

예를 들어, TCP 혼잡 제어 기법에서 TCP 송신부는 혼잡을 감지하면 트래픽 전송 윈도우 크기를 줄임으로써 혼잡을 해결한다. 그러나, 엄청난 수의 공격 패킷에 의하여 발생된 인위적인 혼잡 상황에서는, 정상 사용자의 TCP 송신부가 전송 윈도우 크기를 줄여도 혼잡은 여전히 발생하기 때문에 결국 정상 사용자의 서비스 품질만 더 떨어지게 되는 문제가 발생한다.

FQ (Fair Queuing) 는 플로우별로 큐를 할당하고 그 큐를 라운드 로빈 방식으로 서비스함으로써 각 플로우에게 최소한의 자원을 보장하는 장점을 가지고 있다. 그러나, 이 방법도 엄청난 수의 공격 플로우를 사용하여 발생되는 인위적인 혼잡 상황에서는 정상 사용자가 이용할 수 있는 자원은 공격 플로우의 수에 비례하여 줄어드는 문제점을 가지고 있다.

RED는 전송 큐가 어떤 한계 이상으로 차면 입력 패킷을 확률에 기반하여 마킹하거나 폐기하는 방식으로써, 혼잡 발생 시 가장 많은 패킷을 생성한 플로우의 패킷이 폐기될 확률이 가장 높은 장점을 갖는다. 그러나, 공격자가 패킷을 더 많이 생성하면 할수록 공격자의 패킷뿐만 아니라 정상 사용자의 패킷도 폐기될 확률도 더 높아지기 때문에 인위적인 혼잡 발생에서는 큰 효과를 거둘 수 없다.

서비스 거부 공격을 차단하기 위하여 몇 가지 기법들이 제안되었다. 특정 응용 트래픽(예, ICMP)의 대역폭이 정해진 최대값을 넘지 않도록 대역폭을 제한하는 정적 대역폭 제한 기법[7]이 있다. 이 기법은 혼잡이 발생하는 것을 사전에 막을 수는 있지만, 혼잡이 발생하지 않은 상황에서도 여유 자원을 사용할 수가 없기 때문에 자원 효율성 측면에서 매우 성능이 떨어지는 문제점이 있다.

또 다른 방법으로는 라우터간 통신 기반의 동적 대역폭 제한 기법[8][9]이 제안되었다. 이 방법은 하위 시스템이 혼잡을 감지하면 상위 라우터에게 자신에게 보내는 트래픽의 양을 줄이라고 요구를 함으로써 효과적으로 공격 트래픽을 제어할 수 있는 방법이다. 그러나, 이 기법은 새로운 프로토콜의 구현을 요구하며 또한 대역폭 제한 값을 정확하게 결정하는 것이 쉽지 않다.

Ⅲ. 인터넷 트래픽의 분석 및 분류

3.1 출발지 네트워크 IP 기반의 트래픽 트렁크

본 논문에서는 모든 트래픽을 출발지 네트워크 IP 주소에 의한 트래픽 트렁크(STT: Source IP address-based Traffic Trunk)로 분류한다. 여기서 STT란 출발지 네트워크 IP 주소가 같은 패킷들의 집합을 말한다. 2장에서 언급했듯이, uRPF와 같은 기존 IP 속임 방지 기법은 공격자가 속일 수 있는 IP 주소의 범위를 자신이 속하는 네트워크 범위 내로 한정시킨다. 따라서 STT는 네트워크 단위이므로 출발지 IP 속임 문제를 자연스럽게 비켜갈 수 있다. 그림 1은 분산 서비스 거부 공격을 본 논문에서 제안하는 STT를 사용하여 모델링한 그림이다.

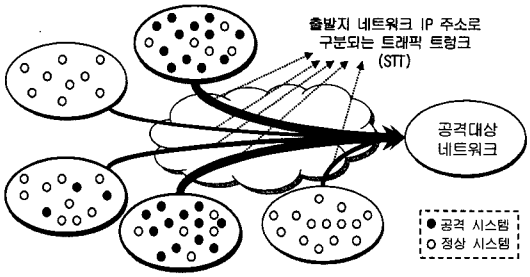


그림 1. 분산 서비스 거부 공격 모델링.
Fig. 1. Conceptual modeling of DDoS attack

STT의 크기는 네트워크 환경에 따라서 다르게 결정될 수 있다. 예를 들어, core 망에서는 24 비트 IP 주소 마스킹을 사용하고, Edge 망에서는 28 비트 IP 주소 마스킹을 사용하여 다르게 STT를 정의할 수 있다.

3.2 혼잡유발 트래픽과 순응 트래픽

트래픽의 생성자가 공격자인지 또는 정상 사용자인지를 구분하는 것은 매우 어렵다. 예를 들어, 새로운 정보를 제공하는 특정 사이트에 접속이 급격히 증가하여 네트워크 혼잡을 일으키는 일종의 서비스 거부 공격인 플래시 크라우드(Flash Crowd)[14]는 바로 정상 사용자에 의해서 발생된다.

이러한 사실을 근거로 하여, 본 논문에서는 트래픽 생성 주체가 누구인지 보다는 트래픽의 특성이 순응 트래픽인지 또는 혼잡유발 트래픽인지에 초점을 맞춘다. 본 논문에서 순응 트래픽이란 혼잡이 일어나지 않도록 네트워크 상태에 맞추어 트래픽의 생성 양을 조절하거나 또는 다른 트래픽 보다 상대적으로 더 적은 양의 트래픽을 생성하는 트래픽을 말한다. 혼잡유발 트래픽은 네트워크 상태에 상관없이 다른 트래픽 보다 더 많은 양의 트래픽을 생성하여 혼잡을 유발/지속시키려는 트래픽을 말한다.

본 논문에서는 정상 트래픽을 순응 트래픽으로 모델링하고, 공격 트래픽을 혼잡유발 트래픽으로 모

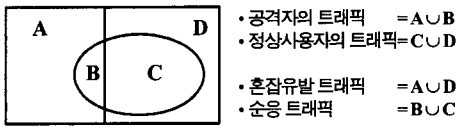


그림 2. 인터넷 트래픽 분류.
Fig. 2. Classification of Internet traffic.

델링한다. 순응 트래픽에게는 높은 우선순위의 서비스를 제공함으로써 그 품질을 보호하고, 혼잡유발 트래픽에게는 낮은 우선순위의 서비스를 제공함으로써 만약 네트워크 혼잡이 발생한다면 그 혼잡이 혼잡유발 트래픽에게만 영향을 미치도록 한다.

그림 2는 인터넷 트래픽을 트래픽 생성자 및 특성에 따라서 분류한 것이다. 본 논문에서 제안하는 기법은 공격자가 생성한 순응 트래픽(그림 2에서 B)을 탐지하지 못하는 false-negative 에러가 있을 수 있다. 그러나, 공격자는 일반적으로 혼잡유발 트래픽을 사용하여 공격을 하기 때문에 B영역은 A영역에 비해서 매우 작을 것이고, 따라서 이 문제는 그리 심각하지 않을 것이다. 또한 본 논문에서 제안하는 기법은 정상 사용자가 생성한 혼잡유발 트래픽(D)을 공격으로 탐지하는 false-positive 에러가 있을 수 있다. 그러나, 정상 사용자일지라도 네트워크 혼잡을 일으킨다면 최선형 서비스(Best-effort Service)상에서 자원을 독점한다는 의미이므로, 자원 공평성측면에서 볼 때 그 정상 사용자에게 제재를 가해는 것이 바람직할 것이다.

본 논문에서 모든 입력 트래픽은 STT에 의하여 구분된다. 전체 트래픽은 STT의 부하(예, 대역폭)가 작은 순서대로 일련번호를 붙인 STT들의 집합으로써 다음과 같이 정의된다.

$$\text{전체트래픽} = \left\{ STT_1 .. STT_n \mid (Load\ of\ STT_i) \le (Load\ of\ STT_{i+1}), \atop 1 \le i < n \right\} \quad (1)$$

여기서, n은 활성중인 STT의 수이고, i는 STT의 식별자이다. 이때, 순응 트래픽과 혼잡유발 트래픽은 다음과 같이 정의된다.

$$\text{순응 트래픽} = \left\{ STT_i \mid \left(\sum_{k=1}^{i} (Load\ of\ STT_k) \right) \le (MaxLoadThreshold) \right\} \quad (2)$$

$$\text{혼잡유발트래픽} = \{\text{전체트래픽} - \text{순응트래픽}\} \quad (3)$$

여기서, MaxLoadThreshold는 순응 트래픽에게 제공할 최대 자원이다. 정의 2에서, 순응 트래픽은 다른 STT 보다 상대적으로 자원 사용량이 적은 STT들의 집합이고, 그 집합이 사용하는 총 자원은 MaxLoadThreshold 보다 작거나 같아야 한다.

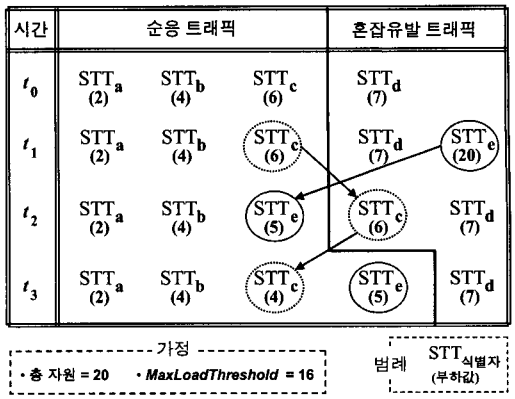


그림 3. 순응 트래픽과 혼잡유발 트래픽 분류 예
Fig. 3. Example of distinction between adaptive traffic and Congestion-making traffic

그림 3은 정의 2와 정의 3에 따른 순응 트래픽과 혼잡유발 트래픽의 분류 예를 보여주고 있다. 그림 3에서 총 자원은 20, *MaxLoadThreshold* 는 16로 가정한다. STT들은 정의 1에서와 같이 부하 값에 따라 정렬되어 있다.

시간 t_0에서, STT_a부터 STT_d 순으로 부하 값을 더할 때 *MaxLoadThreshold* 를 넘지 않는 최대 값은 STT_a 부터 STT_c 까지의 부하 값 12이다. 따라서 정의 2와 3에 의하여 순응 트래픽은 STT_a, STT_b, STT_c 이고, 혼잡유발 트래픽은 STT_d 이다. STT_d는 비록 혼잡유발 트래픽으로써 낮은 우선순위의 서비스가 제공되지만, 현재 여유 자원이 있기 때문에 (즉, 총 STT 부하(19)가 총 자원(20)보다 작다) 어떠한 패킷 손실도 발생하지 않는다.

시간 t_1에서, 새로운 STT인 STT_e는 혼잡유발 트래픽으로 분류된다. 이때 총 STT 부하 값이 39로써 총 자원보다 크므로 네트워크 혼잡이 발생한다. 이 경우에, 순응 트래픽은 높은 우선순위의 서비스를 제공받기 때문에 그 서비스 품질을 보장 받지만, 혼잡유발 트래픽으로 분류된 STT_d와 STT_e는 패킷 손실을 경험할 것이다. 따라서 만약 STT_e가 공격자에 의해 생성된 트래픽이라면, 본 논문에서는 그 공격이 단지 STT_d에만 영향을 주도록 함으로써 공격의 강도를 최소화 할 수 있다.

시간 t_2에서, STT_e는 그 부하 값이 5이므로 순응 트래픽으로 분류되고, STT_c는 혼잡유발 트래픽으로 분류된다. 시간 t_3에서, STT_c가 그 부하 값을 4로 줄였고, 따라서 순응 트래픽으로 분류된다.

공격자는 시간 t_1과 t_2에서의 STT_e 처럼 on-off 트래픽을 반복함으로써 순응 트래픽과 혼잡유발 트래픽 경계에 있는 STT들 (예, STT_c)의 서비스 품질을 떨어뜨리고자 할 수 있을 것이다. 그러나 이를 위해서는 공격자는 가장 치명적인 영향을 줄 수 있는 부하 값을 계산해야 하는데, 이것은 쉽지 않은 문제이다. 예를 들어, 시간 t_2에서 STT_e의 그 부하 값이 5나 6이 아니면 STT_c는 이전과 마찬가지로 여전히 순응 트래픽이므로 그 공격은 성공하지 못할 것이다.

공격자는 본 논문에서 제안하는 기법을 회피하기 위하여 STT_a, STT_b와 같은 순응 트래픽을 사용하여 공격할 수도 있을 것이다. 그러나 이것이 성공하려면, 공격자는 엄청난 수의 서로 다른 네트워크에 있는 수많은 시스템들을 해킹하거나 감염시켜야 하므로 보다 많은 공격 준비시간을 가져야 한다.

Ⅳ. MF 게이트웨이

4.1. MF 게이트웨이의 구조

MF 게이트웨이의 구조는 그림 4에 도시 되어 있다. MF 게이트웨이는 HPQ(High-Priority Queue), MPQ(Medium-Priority Queue), 그리고 LPQ (Low-Priority Queue), 세 개의 우선순위 큐를 사용한다. 세 개의 큐 중에서 우선순위는 HPQ가 가장 높으며, LPQ가 가장 낮다. MF 게이트웨이는 순응 트래픽에게 HPQ를 사용하여 그 서비스 품질을 보장해 주며, 혼잡유발 트래픽에게는 MPQ 또는 LPQ를 사용하여 한 단계 낮은 서비스를 제공해준다.

MF 게이트웨이의 패킷 전달 과정을 살펴보면, 패킷이 도착하면 패킷이 혼잡유발 트래픽으로 마킹되어 있는지 검사한다. 만약 마킹되어 있다면 LPQ로 보낸다. 그렇지 않으면, 패킷의 출발지 주소를 보고 해당 STT를 결정한 후 그 STT에 해당하는 서비스를 STT 정보 테이블에서 검색한다. STT 정보 테이블에는 STT 식별자, STT의 평균 사용 대역폭, 서비스할 큐 식별자 등의 정보를 가지고 있다. 만약 STT 서비스의 검색 결과가 HPQ면 패킷은 HPQ로 보내지며, 그렇지 않으면 패킷에 혼잡유발 트래픽으로 마킹한 후 MPQ로 보내진다.

예를 들어, 그림 4에서 출발지 주소가 1.1.2.20인 패킷은 STT 정보 테이블에서 1.1.2.x 인 STT의 서비스는 HPQ이므로 HPQ로 보내진다. 또한 출발지 주소가 2.1.7.70인 패킷은 혼잡유발 트래픽으로 마킹되어 있으므로 LPQ로 보내진다.

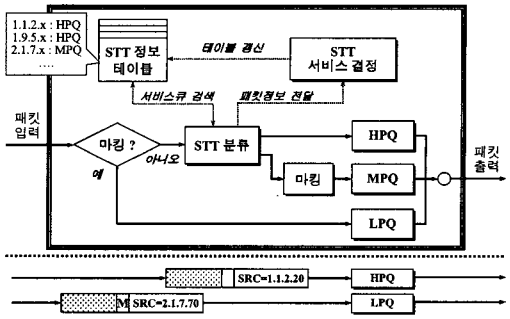


그림 4. MF 게이트웨이의 구조
Fig. 4. Architecture of MF Gateway

MF 게이트웨이에서 가장 중요한 것은 STT 정보 테이블을 갱신하는 것과 마킹이다. 이에 대한 설명은 다음절에서 한다.

4.2. 빠른 STT 서비스 결정 알고리즘

STT 정보 테이블을 갱신하는 것은 정의 1, 2, 3을 모두 만족하는 어떤 알고리즘을 필요로 한다. 본 논문에서는 간단하지만 빠르게 STT 서비스를 결정할 수 있는 그림 5에 도시된 알고리즘을 제안한다.

제안하는 STT 서비스 결정 알고리즘은 다음과 같다. 먼저 패킷이 도착할 때 마다 그 패킷에 해당하는 STT의 수신 트래픽 양을 다음과 같이 계산한다.

$$STT.sampleSize = STT.sampleSize + samplePacketSize$$

그리고 일정시간이 경과하면 EWMA (Exponential Weighted Moving Average)를 이용하여 STT의 평균 대역폭을 다음과 같이 계산한다.

$$STT.avgBW = STT.avgBW \times (1.0 - \alpha) + (STT.sampleSize / ElapseTime) \times \alpha$$

여기서, α는 0과 1사이의 값이고, $ElapseTime$은 대역폭 측정 시간을 말한다.

본 논문에서는 STT의 서비스를 결정하기 위하여 다음과 같이 세 개의 오퍼레이션을 정의한다.

- Swap-in 오퍼레이션: MQP를 사용하는 STT의 서비스를 HPQ로 바꾸는 오퍼레이션.
- Swap-out 오퍼레이션: HQP를 사용하는 STT의 서비스를 MPQ로 바꾸는 오퍼레이션.
- Preemption 오퍼레이션: MQP를 사용하는 STT와 HQP를 사용하는 STT의 서비스를 서로 바꾸는 오퍼레이션.

먼저, 수신 STT를 HPQ로 서비스하고자 할 때의 총 순응 트래픽 부하가 정의 2에서 정의한 *Max LoadThreshold* 보다 크면 수신 STT에게 Swap-out 오퍼레이션을 적용하여 그 서비스를 LPQ로 설정한다. 만약 순응 트래픽의 부하가 *MaxLoad- Thre shold* 보다 작거나 같으면 수신 STT에게 Swap-in 오퍼레이션을 적용하여 그 서비스를 HPQ로 설정한다. 이렇게 함으로써 HPQ에서 혼잡이 발생하는 것을 막을 수 있으며, 또한 가능한 많은 STT들이 HPQ로 서비스 될 수 있게 한다.

이때 부하가 큰 STT가 HPQ로 서비스 받고, 반대로 부하가 작은 STT가 MPQ로 서비스 받는 문제가 발생할 수 있다. 이를 위하여 본 알고리즘은 다음과 같이 두 개의 변수를 정의한다.

- Worst_HPQ_STT: HPQ 서비스를 받는 STT들 중에서 부하가 가장 큰 STT를 가리키는 변수.
- Best_MPQ_STT: MPQ 서비스를 받는 STT들 중에서 부하가 가장 작은 STT를 가리키는 변수.

수신 STT의 서비스가 HPQ이면 Worst_HPQ_STT와 비교하고, 반대로 MPQ이면 Best_MPQ_STT와 비교하여, Worst_HPQ_STT와 Best_LPQ_STT의 값을 갱신한다. 그리고 만약 Best_MPQ_STT의 부하가 Worst_HPQ_STT보다 크면, Preemption 오퍼레이션을 적용하여 두 STT의 서비스를 바꾼다.

마지막으로, HPQ 서비스를 받는 STT가 일정시간 동안 비활성화되어 있으면 순응 트래픽의 부하에서 비화성화된 STT의 부하를 뺀 값을 순응 트래픽의 부하로 함으로써 그 STT에게 할당된 할당된 자원을 수거한다.

4.3 마킹 기법

여러 개의 네트워크 노드에서 동시에 혼잡이 발생하는 다중 혼잡 상황에서, 상위 MF 게이트웨이가 MPQ/LPQ로 서비스한 혼잡 유발 트래픽이 하위 MF 게이트웨이에서는 정상 트래픽으로 오인될 수 있다. 이를 해결하기 위해서, 본 논문에서는 혼잡 유발 트래픽으로 결정된 패킷을 혼잡 유발자로 마킹하여 하위 MF 게이트웨이에게 알려주는 마킹 기법을 제안한다.

패킷을 마킹하기 위해서는 IP 헤더에서 한 비트가 필요하다. 현재 차등 서비스에서는 그림 6에서와 같이 DiffServ 필드[15]로써 ToS 필드를 사용하고 있다. 마킹을 제공하는데 있어서 두 가지 방안이 가능할 것 같다. 하나는 ECN(Explicit Congestion Notification) 필드를 그대로 사용하는 방안이고, 또 다른 하나는 차등 서비스[16][17]에서 DS0는 보통 0으로 설정되기 때문에 마킹시 DS0 필드를 1로 설정하여 사용하는 방안이다.

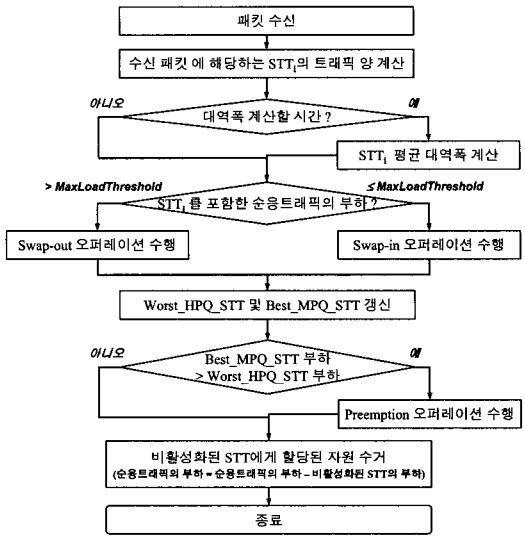


그림 5. STT 서비스 결정 알고리즘
Fig. 5. STT Service Determination Algorithm

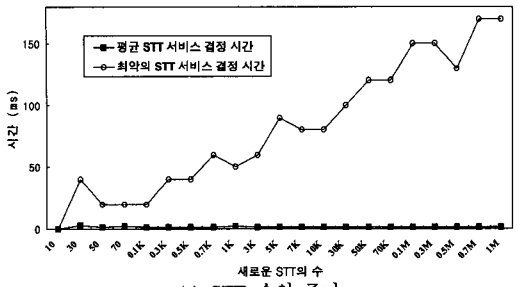
그림 6. 차등 서비스에서 ToS 필드
Fig. 6. ToS field in Differentiated Service

V. 실험 및 성능 평가

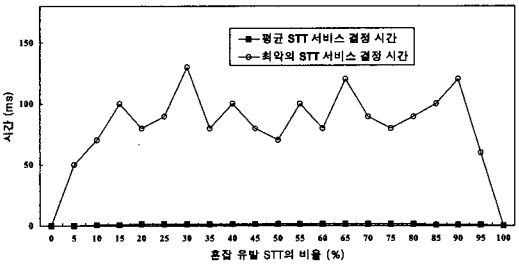
본 장에서는 먼저 본 논문에서 제안한 STT 서비스 결정 알고리즘의 성능을 분석하고, 서비스 거부 공격에서의 기존의 기법과 본 논문에서 제안하는 MF 게이트웨이 기법을 각각 시뮬레이션하여 그 성능을 비교한다. 시뮬레이션 툴로서 NS (Network Simulator) 시뮬레이터[18]를 사용하였다.

5.1. STT 서비스 결정 알고리즘

그림 7은 STT 서비스 결정 알고리즘의 실험 결과이다. 본 실험의 목적은 본 논문에서 제안한 알고리즘이 얼마나 빠르게 STT의 서비스를 결정할 수 있는가를 분석하기 위함이다. STT 서비스 결정 시간이 늦으면 늦을수록 그 시간 동안 정상 트래픽의 서비스 품질이 보호될 수 없기 때문이다. 본 실험에서 패킷 크기는 64바이트이고, 각 STT의 패킷간 평



(a) STT 수의 증가



(b) 혼잡유발 STT 비율의 증가

그림 7. STT 서비스 결정 알고리즘의 성능.
Fig. 7. Performance of STT Service Determination Alg

균 지연시간은 10ms로 하였다. 각 STT의 대역폭 값은 시뮬레이션 초기에 랜덤하게 결정된다.

그림 5에 도시된 STT 서비스 결정 알고리즘에서의 대역폭 계산 시간은 본 실험 결과에 포함시키지 않았다. 따라서 그림 7의 실험 결과에서 Y축은 STT의 대역폭 계산 시간을 제외한 STT 서비스 결정 시간을 의미한다. 예를 들어, STT 서비스 결정 시간이 100ms 라는 것은 STT의 평균 대역폭이 이미 계산되어 있는 상태에서 그 대역폭에 해당하는 STT의 서비스를 결정하는데 100ms (약 패킷 10개가 지나가는 시간)가 걸린다는 의미이다.

그림 7-(a)는 혼잡유발 STT의 비율을 60%로 고정한 상태에서 STT의 수를 증가시켰을 때의 실험 결과이고, 그림 7-(b)는 STT의 수를 10,000로 고정한 상태에서 혼잡유발 STT의 비율을 증가시켰을 때의 실험 결과이다. 본 논문에서 제안한 STT 서비스 결정 알고리즘은 비록 최악의 STT 서비스 결정 시간은 그림 7-(a)에서 STT가 백 만개일 때 160ms 이고 그림 7-(b)에서는 140ms 이지만, 두 실험 모두 평균 결정시간에서는 10ms 이하로써 STT의 수 및 혼잡유발 STT의 비율에 거의 상관없이 매우 빠르게 STT의 서비스를 결정하고 있다.

5.2 STT 서비스 결정 알고리즘

MF 게이트웨이의 성능 분석을 위한 실험 망은 그림 8에 도시 되어 있다. MF 게이트웨이를 포함

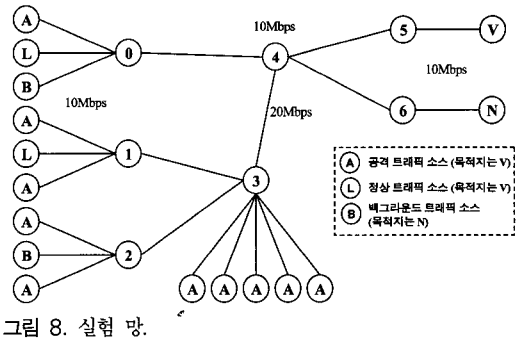


그림 8. 실험 망.
Fig. 8. Simulation Networks

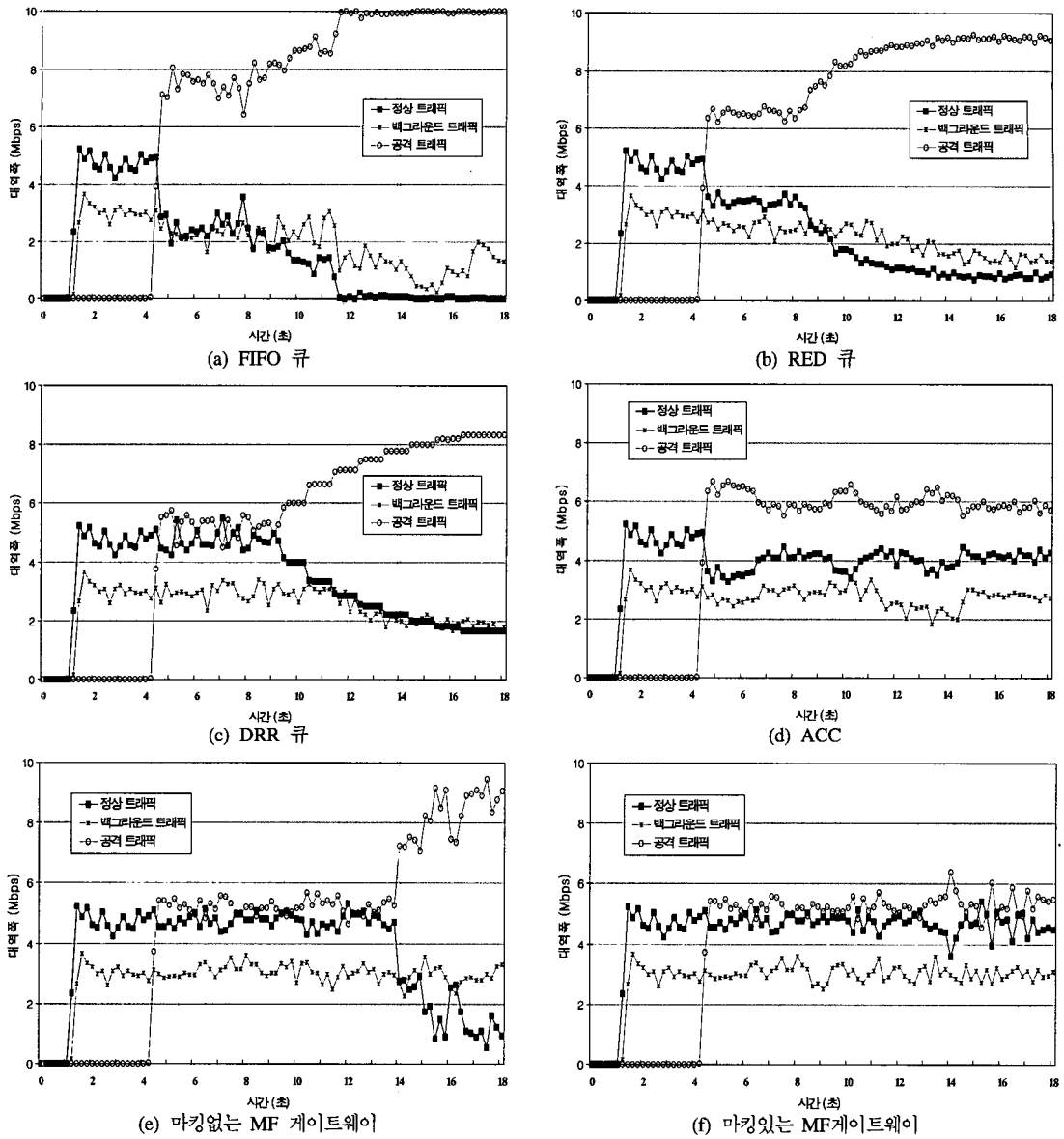
한 모든 기법들은 노드 0-6상에서 동작한다. 노드 3과 4사이의 링크 대역폭은 20Mbps이고 나머지는 모두 10Mbps이다.

노드 A, L, B는 각각 목적지가 V 노드인 공격 트래픽 소스, 목적지가 V 노드인 정상 트래픽 소스, 그리고 목적지가 N 노드인 백그라운드 트래픽 소스이다. 본 실험에서 각 트래픽 소스는 하나의 STT에 해당한다. 각 노드 A는 30개, 각 노드 L은 11개, 그리고 각 노드 B는 7개의 플로우를 생성한다. 각 플로우는 EXPOO(EXPonential On/Off) 분포인 UDP 패킷을 사용하며, 그림 9에서는 on 주기를 100ms, off 주기를 40ms로 설정하였고, 그림 10에서는 on 주기를 150ms, off 주기를 850ms로 설정하였다. MF 게이트웨이 실험에서 α 값은 0.2, STT 대역폭 계산 주기는 100ms, 비활성 STT 소멸시간은 1.0s, 그리고 *MaxLoadThreshold* 는 총 자원의 85% 값으로 설정하였다.

공격 트래픽 소스는 시뮬레이션 시간 4초에 서비스 거부 공격을 시작하며, 8초부터는 분산 서비스 거부 공격을 시작한다. 따라서 4초에는 단일 노드에서 혼잡이 발생하지만, 8초부터는 다중 노드에서 동시에 혼잡이 발생한다.

그림 9는 분산 서비스 거부 공격 하에서의 기법별 실험 결과이다. 그림 9-(a)는 FIFO 큐를 사용했을 때의 실험 결과이다. FIFO 큐에서는 시뮬레이션 시간 4초에 서비스 거부 공격이 시작되자마자 정상 트래픽과 백그라운드 트래픽 모두 서비스 품질이 거의 반으로 떨어졌으며, 8초에 시작된 분산 서비스 거부 공격에서는 정상적인 서비스를 거의 제공하지 못하고 있다. 이것은 서비스 거부 공격의 파괴력이 얼마나 큰지를 보여주는 결과이다.

그림 9-(b)는 RED 큐를 사용했을 때의 실험 결과이다. RED 큐는 비록 FIFO 큐보다는 성능에서 우수하지만, 공격이 계속됨에 따라서 정상 트래픽의



(a) FIFO 큐 (b) RED 큐

(c) DRR 큐 (d) ACC

(e) 마킹없는 MF 게이트웨이 (f) 마킹있는 MF게이트웨이

그림 9. 분산 서비스 거부 공격에서 정상 트래픽 보호 성능
Fig. 9. Performance on protection of normal traffic under DDoS attacks

서비스 품질이 크게 떨어지는 것을 볼 수 있다. 이러한 이유는 RED 에서는 공격의 강도를 높이면 높일수록, 즉 공격자가 더 많은 공격 패킷을 생성하면 할수록 정상 패킷이 폐기될 확률이 더 높아지기 때문이다.

그림 9-(c)는 FQ 큐의 일종인 DRR (Deficit Round Robin) 큐를 사용했을 때의 실험 결과이다. DRR은 단일 노드에서 공격하는 서비스 거부 공격에서는 그 성능이 매우 우수한 것을 볼 수 있다. 그러나 8초부터 시작하는 분산 서비스 거부 공격에서는 그 성능이 크게 떨어지는 것을 볼 수 있다. 이러한 이유는 트래픽 소스의 수가 많으면 많을수록 단일 소스가 사용할 수 있는 대역폭의 양이 더 작아지기 때문이다.

그림 9-(d)는 ACC(Aggregate Congestion Control) 라고 불리는 동적 대역폭 제한 기법을 사용했을 때의 실험 결과이다. ACC는 입력 패킷을 목적지 IP 주소 기반의 집합 플로우로 분류하고 큐를 모니터링한다. 만약 일정시간 동안 큐에서 계속 패킷 손실이 발생하면, 혼잡을 유발한 집합 플로우에

게 허용할 최대 대역폭을 결정하여 대역폭 제한을 실행한다. ACC는 그림 9-(d)에서 보는 바와 같이 기존의 큐잉 기법보다는 월등히 좋은 성능을 보여주지만, 정확한 대역폭 제한 값을 결정하고 있지는 못하고 있다. 대역폭 제한을 실행하고 있는 ACC 노드(즉, 라우터)는 일정시간이상 동안 계속 혼잡이 발생하면, pushback 프로토콜을 사용하여 혼잡을 유발한 집합 플로우의 패킷을 전달하고 있는 상위 노드에게 대역폭 제한을 요구함으로써 ACC의 성능을 향상시킬 수 있다. 그러나 연속적인 다중 혼잡이 발생한 경우에는 하위 노드뿐만 아니라 상위 노드도 ACC를 수행하고 있기 때문에, 대역폭 제한 결정이 두 노드에서 서로 충돌하는 문제가 발생한다. 본 논문에서는 연속적인 다중 혼잡을 유발하는 서비스 거부 공격을 실험하고 있기 때문에 이를 지원하지 않는 pushback 프로토콜은 실험하지 못했다.

그림 9-(e)와 (f)는 본 논문에서 제안한 MF 게이트웨이의 실험 결과이다. 그림 9-(e)는 마킹을 사용하지 않는 MF 게이트웨이의 실험 결과이다. MF 게이트웨이는 서비스 거부 공격에서는 물론이고, 8초에서 14초사이의 분산 서비스 거부 공격에서도 정상 트래픽과 백그라운드 트래픽의 서비스 품질을 거의 완벽하게 보장하고 있는 것을 볼 수 있다. 14초 이후부터 성능이 떨어지는 것은 다중 혼잡 상황에서 상위 MF 게이트웨이가 MPQ로 서비스한 공격 트래픽을 하위 MF 게이트웨이는 정상 트래픽으로 오인하기 때문에 발생한 것이다. 본 논문에서는 이를 해결하기 위하여 마킹을 제안하였다. 그림 9-(f)는 마킹을 사용한 MF 게이트웨이의 실험 결과이다. MF 게이트웨이는 서비스 거부 공격뿐만 아니라 분산 서비스 거부 공격에서도 정상 트래픽과 백그라운드 트래픽의 서비스 품질을 거의 완벽하게 보장하고 있다.

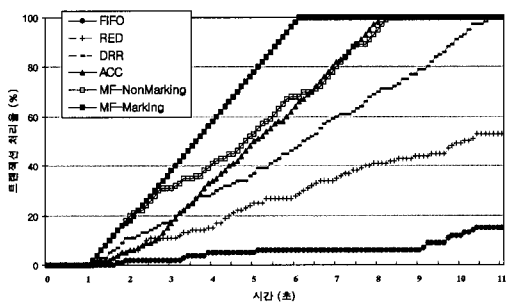


그림 10. 분산 서비스 거부 공격에서 트랜잭션 처리 성능.
Fig. 10. Performance on transaction processing under DDoS attacks

그림 10은 분산 서비스 거부 공격하에서 정상 소스가 50 ms마다 목적지 노드 V에게 트랜잭션 처리를 요구하여 총 100개의 트랜잭션을 처리하고자 할 때의 기법 별 성능을 도시한 그림이다. 그림 10에서 보는 바와 같이 MF 게이트웨이가 총 트랜잭션 처리시간이 6초로써 가장 우수하며, 그 다음은 ACC, DRR, RED, FIFO 순으로 나타났다.

Ⅵ. 결론 및 향후연구과제

서비스 거부 공격은 네트워크에 인위적인 혼잡을 발생시킴으로써 사용자들이 정상적인 서비스를 제공받지 못하게 하는 것을 목적으로 한다.

본 논문에서는 기존의 큐잉 기법은 서비스 거부 공격 하에서는 인위적인 혼잡을 해결할 수가 없으며, 따라서 정상 사용자의 서비스 품질을 보호할 수 없다는 것을 실험을 통하여 확인하였다. 즉, RED 큐는 혼잡 상황에서 공격 트래픽양이 증가하면 증가할수록 그에 비례하여 정상 트래픽의 성능은 감소하였고, 또한 DRR 큐도 혼잡 상황에서 공격 소스의 수가 증가하면 증가할수록 그에 비례하여 정상 트래픽의 성능이 감소하는 문제가 있었다.

본 논문에서는 서비스 거부 공격으로 인한 네트워크 혼잡 상황에서도 정상 트래픽의 서비스 품질을 보장할 수 있는 MF 게이트웨이를 제안하였다. MF 게이트웨이의 성능을 평가하기 위하여 몇 가지 상황에서 실험을 하였다. 실험 결과는 MF 게이트웨이는 서비스 거부 공격뿐만 아니라 다중 노드에서 동시에 혼잡을 일으키는 분산 서비스 거부 공격에서도 정상 트래픽의 서비스 품질을 거의 완벽하게 보장한다는 것을 보여주고 있다.

MF 게이트웨이의 패킷 전달 알고리즘, STT 서비스 결정 알고리즘, 그리고 마킹 알고리즘은 매우 간단하여 실세계에 바로 적용할 수 있다. 향후 연구과제로써는 실제의 네트워크에서 MF 게이트웨이를 구현하는 것이다.

참 고 문 헌

[1] 전완근, 류성철, 김승철, "MS-SQL 서버 웜 - 슬래머(Slammer) 공격 테스트 및 사고대응," CERTCC-KR, 사고노트, Jan. 2003

[2] K. J. Houle and G. M. Weaver. "Trends in Denial of Service Attack Technology," The fall 2001 NANOG meeting, Oct. 2001.

[3] X. Geng and A. B. Whinston, "Defeating

Distributed Denial of Service Attacks", IT Pro, pp 36-41, July 2000.

[4] S. Keshav, "An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network", Addison Wesley, 1997.

[5] S. Floyd, "TCP and explicit congestion notification," ACM Computer Communication Review, vol. 24, no. 5, pp. 10-23, Oct. 1994,

[6] Sally Floyd and Van Jacobson, "Random Early Detection Gateways for Congestion Avoidance," IEEE Transactions on Networking, Vol.1, No.4, pp.397-413, Aug. 1993.

[7] Cisco, "Strategies to Protect Against Distributed Denial of Service (DDoS) Attacks," white paper, http://www.cisco.com/…/newsflash.html, Feb. 2000.

[8] R. Mahajan, S. M. Bellovin, S. Floyd, and et al., "Controlling High Bandwidth Aggregates in the Network," ACM SIGCOMM Computer Communications Review, Vol. 32, No. 3, pp. 62-73, July 2002.

[9] D.K.Y. Yau, J.C.S. Lui, and Feng Liang, "Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles," Tenth IEEE International Workshop on Quality of Service, pp.35 -44, May 2002.

[10] P. Ferguson and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing," IETF, RFC 2827, May 2000.

[11] T. Killalea, "Recommended Internet Service Provider Security Services and Procedures," IETF, RFC 3013, Nov. 2000.

[12] Cheng Jin, Haining Wang, Kang G. Shin, "Hop-count filtering: an effective defense against spoofed DDoS traffic," ACM CCS, pp. 30-41, Oct. 2003.

[13] Cisco, "Unicast Reverse Path Forwarding (uRPF) Enhancements for the ISP-ISP Edge", http://www. cisco.com/…/uRPF_Enhancement. pdf, Feb. 2001.

[14] J. Jung, B. Krishnamurthy and M. Rabinovich, "Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites," The 11th International World Wide Web Conference, pp. 252-262, May 2002.

[15] K. Nichols,S. Blake, F. Baker and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers," IETF, RFC 2474.

[16] F. Baker, W. Weiss and J. Wroclawski, "Assured Forwarding PHB Group," IETF, RFC 2597.

[17] V. Jacobson, K. Nichols, K. Poduri, "An Expedited Forwarding PHB," IETF, RFC 2598.

[18] UCB/LBNL/VINT, "ns Notes and Documentation," http://www.isi.edu/nsnam/ns.

안 개 일 (Gae-Il Ahn) 정회원

1993년 2월 충남대학교 컴퓨터공학과 졸업

1995년 2월 충남대학교 컴퓨터공학과(석사)

2001년 8월 충남대학교 컴퓨터공학과(박사)

2001년 8월~현재 한국전자통신연구원 선임연구원

<관심분야> 컴퓨터 네트워크, 네트워크 보안, 트래픽 엔지니어링