IP 망에서 액티브 패킷의 경로 설정 및 신뢰성 전송

윤 보 영[†] · 채 기 준[††] · 남 택 용[†††]

요 약

액티브 패킷은 기존의 IP 패킷과는 달리 목적지가 아닌 중간의 액티브 노드에서 실행되어야 할 필요가 있다. 이를 위해서는 하나의 프로그램을 수행하기 위한 액티브 패킷들이 같은 액티브 노드를 경유해야함은 물론이고 모든 패킷들이 손실 없이 도착해야 정상적인 작업을 수행할 수 있다. 본 논문에서는 하나의 액티브 프로그램을 수행하는데 필요한 액티브 패킷들이 손실 없이 같은 액티브 노드를 거쳐 실행될 수 있는 액티브 엔진을 제안한다. 제안한 액티브 엔진은 기존의 IP 망에 적용이 쉽고 송신자가 네트워크 내의 액티브 노드의 위치를 몰라도 액티브 패킷들이 같은 액티브 노드를 경유할 수 있으며 신뢰성 있는 액티브 패킷의 전송을 보장한다. 또한 시뮬레이션을 통하여 기존의 전송 프로토콜들과 성능을 비교·분석한 결과, 제안하는 액티브 엔진은 전송지연이 작고 처리율이 높다는 것을 확인할 수 있었다.

Routing and Reliable Transmission of Active Packets in IP Networks

Bo-Young Yoon[†] · Kijoon Chae[††] · Taek-Yong Nam[†††]

ABSTRACT

The active packets unlike traditional IP packets should be executed at each active node along their path. To execute the active program at each active node, the active packets for an active program should go through the same active nodes and all packets are delivered without any loss. This paper presents the new active engine for every active packet which execute an active program to be routed through the same intermediate active nodes and to be delivered reliably. Proposed active engine requires fewer changes to existing IP protocols and guarantees the reliable delivery of the active packets. Moreover, even if the sender does not have any information about the intermediate nodes every active packet is routed through the same intermediate active nodes. The simulation results show that proposed active engine achieves an efficient transmission with high data delivery and low communication overhead compared with the other existing transmission protocols.

키워드 : 액티브 패킷(Active Network), 경로설정(Routing), 신뢰성 전송(Reliable Transmission), 액티브 엔진(Active Engine)

1. 서 론

액티브 네트워크란 라우터나 스위치가 프로그램 실행 능력을 가지고 있어서 프로그램을 포함하고 있는 액티브 패킷을 처리하여 패킷에 다양하고 유동적인 처리를 행할 수 있는 환경을 가진 망을 말한다[1]. IP 패킷과는 달리 액티브 패킷은 때때로 중간의 액티브 노드에서 실행될 필요가 있다. 이 때 하나의 프로그램을 수행하는데 필요한 액티브 패킷들은 모두 같은 액티브 라우터를 경유해야 정상적인 작업을 수행할 수 있다. 그러나 기존의 IP 망에서는 한 응용 프로그램으로부터 같은 목적지로 전송되는 패킷들일지라도 모두 다른 길을 통해서 전달될 수 있다. 따라서 하나의 프로그램을 수행하는데 필요한 여러 개의 액티브 패킷들이 모두 같은 액티브 노드를 경유하도록 하는 메커니즘이 필요하다. 또한 액티브 노드에서 액티브 프로그램을 정상적으로 수행하기 위해서는 액티브 패킷이 손실 없이 전달되어야만 한다. 인터넷의 대표적인 전송 프로토콜인 TCP(Transmission Control Protocol)는 신뢰성 있는 전송을 보장하지만 지연과 부하가 크고 UDP(User Datagram Protocol)는 빠르고 간단하게 액티브 패킷을 전송할 수 있지만 신뢰성을 보장하지 못한다.

액티브 패킷을 목적지가 아닌 중간의 액티브 노드에서 실행하기 위한 지금까지의 방법은 다음과 같다. BBN Technologies에서 개발한 Smart Packet[2]은 IP의 '라우터 경보 옵션(Router Alert Option)'[3-5]을 이용한다. '라우터 경보 옵션'은 중간의 라우터들이 패킷의 목적지가 자신이 아니더라도 패킷의 내용을 처리할 수 있도록 한다. 그러나 '라우터 경보 옵션'은 여러 개의 패킷들을 같은 경로로 전달하지 못한다. 벨연구소에서 개발한 ABLE[6]은 액티브 패킷을 목적지까지의 중간 액티브 노드에서 실행하기 위하여 '중간 액티브 노드

※ 본 연구는 2001년도 한국전자통신연구원 정보보호기술연구본부 네트워크보안연구부 위탁연구 과제에 의한 것임.
† 정 회 원 : 이화여자대학교 대학원 컴퓨터학과
†† 종신회원 : 이화여자대학교 컴퓨터학과 교수
††† 정 회 원 : 한국전자통신연구원 정보보호연구본부 네트워크보안구조 연구팀 팀장
논문접수 : 2002년 5월 4일, 심사완료 : 2002년 8월 27일

를 패킷에 명시한 경우'와 '중간의 액티브 노드를 모를 경우'로 구분한다. 중간 액티브 노드를 패킷에 명시한 경우, 패킷에 명시되어있는 액티브 노드의 ABLE 엔진에서만 처리하고, 명시되어있지 않은 액티브 노드는 일반 IP 패킷처럼 처리한다. 이 방법은 송신지가 중간의 액티브 노드 주소를 알 경우에만 사용 가능하다는 한계가 있다. 중간의 액티브 노드를 모를 경우, 액티브 라우터는 무조건 모든 액티브 패킷을 가로채어 ABLE 엔진에서 처리한다. 이 방법은 '라우터 경보 옵션'을 사용하는 것과 같은 문제점을 가진다.

액티브 패킷의 신뢰성 전송을 위한 연구는 현재 멀티캐스트 분야에서 이루어지고 있다. MIT의 ARM(Active Reliable Multicast)[7]과 PANAMA 프로젝트의 일환인 AER(Active Error Recovery)[8]은 기본적으로 멀티캐스트 트리의 중간 노드를 액티브 노드로 두고 데이터를 저장한다. 그리고 수신자들로부터 재전송 요청을 받으면 저장해 놓은 데이터를 재전송한다. 이 방식은 기존의 종단간 손실 복구 기법에 비하여 대역폭의 낭비가 적고 복구로 인한 지연을 줄일 수 있지만 유니캐스트에 적용시키기가 어렵다.

따라서 본 논문에서는 기존의 IP 망에 적용이 쉬우면서 액티브 패킷들이 같은 액티브 라우터를 경유하도록 할 수 있는 경로 설정 방법과 UDP를 통해서도 신뢰성 있는 전송을 제공할 수 있는 기법을 제안하고자 한다. 또한 이들 기법을 이용하여 액티브 노드에서 다른 패킷들의 흐름에 영향을 끼치지 않고 액티브 패킷의 경로를 설정하고 신뢰성 전송을 보장하는 액티브 엔진을 제안하고 모델링하여 성능을 평가·분석한다.

본 논문의 구성은 다음과 같다. 1장의 서론에 이어 2장에서는 본 논문에서 제안하는 액티브 엔진에 대하여 상세히 설명한다. 3장에서는 제안하는 액티브 엔진과 기존의 전송 프로토콜의 성능을 평가하기 위한 시뮬레이션 모델을 살펴보고 결과를 분석한다. 마지막으로 4장에서는 본 논문의 결론을 제시한다.

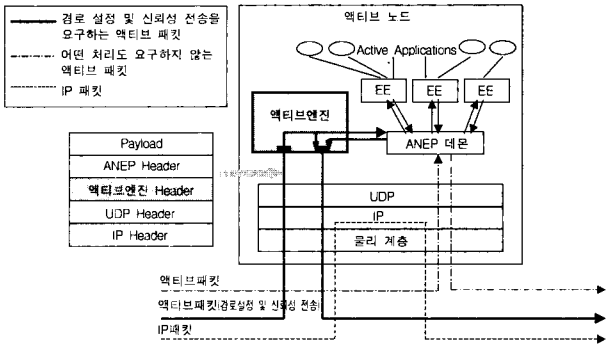
2. 액티브 엔진

하나의 작업을 수행하기 위해 필요한 일련의 액티브 패킷들이 같은 액티브 노드를 경유하도록 하기 위한 경로 설정 기법과 신뢰성 있는 전송을 보장하기 위한 신뢰성 전송 기법을 제공하는 액티브 엔진을 제안한다. 액티브 엔진의 구조와 액티브 패킷의 경로 설정 및 신뢰성 전송 기법에 대하여 살펴보면 다음과 같다.

2.1 액티브 엔진의 구조

중간의 액티브 노드에 설치될 액티브 엔진은 UDP와 ANEP (Active Network Encapsulation Protocol)[9] 사이에서 그 기능을 수행한다. UDP와 ANEP과는 포트를 통하여 액티브 패킷을 주고받는다. 액티브 엔진을 사용자 레벨에서 어플리케이션처럼 구현하고 포트를 할당하는 것은 구현을 용이하게 하고 액티브 노드에 이식을 쉽게 하기 위해서이다. 또한 액티브 패킷을 특별한 표시 없이 포트 번호로 구별할 수 있다. 만약 전송 계층에서 액티브 엔진을 구현하면 속도는 좀 더 빨라질지 모르나 커널 레벨에서 구현해야 하는 어려움이 있고 액티브 노드에 이식도 쉽지 않을 뿐 아니라 보안상의 위협도 존재하게 된다. (그림 1)은 액티브 노드에서의 액티브 엔진의 위치와 패킷의 흐름을 나타낸 것이다.

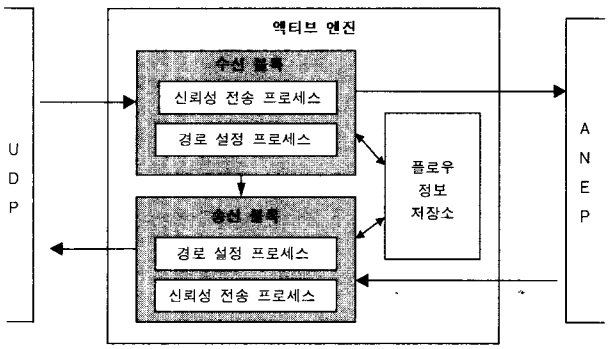
경로 설정 및 신뢰성 전송을 요구하는 액티브 패킷이 들어오면 UDP는 포트 번호를 보고 액티브 엔진으로 액티브 패킷을 보내준다. 액티브 엔진은 신뢰성 검사를 하고 다음 액티브 노드를 결정한 후 사본은 ANEP을 전달하여 액티브 패킷의 내용을 실행하도록 하고, 원본은 UDP를 통하여 다음 노드로 전송한다. 경로 설정이나 신뢰성 전송을 요구하지 않는 액티브 패킷이 들어오면 UDP는 바로 ANEP 데몬으로 액티브 패킷을 넘겨주고 IP 패킷이 들어오면 액티브 노드는 기존의 라우터에서와 같이 단순히 다음 노드를 결정해서 전달한다.



(그림 1) 액티브 노드와 액티브 엔진

(그림 1)과 같은 구조에서는 다른 패킷들에게 영향을 끼치지 않고 경로 설정과 신뢰성 전송이 필요한 액티브 패킷들에 대해서만 액티브 엔진에서 필요한 작업을 수행할 수 있다. 또한 기본 전송 계층인 UDP나 IP에 전혀 수정이 필요하지 않으므로 기존의 망에 쉽게 적용할 수 있다.

액티브 엔진의 내부 구조에 대해 자세히 살펴보면 (그림 2)와 같다.



(그림 2) 액티브 엔진 구조

액티브 엔진은 전 노드로부터 액티브 패킷을 수신했을 때 작동하는 수신 블록과 액티브 패킷을 다음 노드로 전송하기 위한 동작을 수행하는 송신 블록으로 나눌 수 있다. 플로우 정보 저장소는 액티브 노드로 들어오는 액티브 패킷들의 정보를 유지한다. 각각의 블록은 신뢰성 전송 프로세스와 경로 설정 프로세스로 나눌 수 있다. 경로 설정 프로세스는 다음 액티브 노드를 결정하여 액티브 패킷들이 모두 같은 액티브 노드를 거쳐 실행될 수 있도록 한다. 또한 신뢰성 전송 프로세스는 액티브 패킷의 신뢰성 전송을 위한 기법을 제공한다. 각각에 대하여 자세히 살펴보면 다음과 같다.

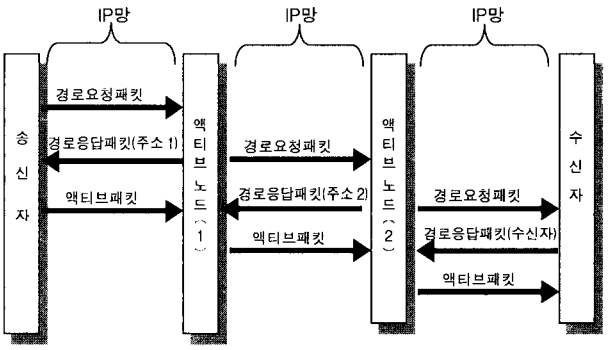
2.2 경로 설정 기법

하나의 작업을 수행하기 위한 여러개의 액티브 패킷들이 중간의 같은 액티브 노드를 경유하기 위하여 초기 경로 설정을 통한 경로 설정 기법을 제안한다. IP 망에 액티브 노드와 IP 라우터가 공존하는 환경에서 각 네트워크의 경계 라우터를 액티브 노드로 가정한다. 액티브 패킷의 송신 노드가 중간의 액티브 노드 위치에 대하여 모르더라도 액티브 패킷이 모두 같은 액티브 노드를 거치도록 한다. 또한 전송지연을 최소화 한다.

2.2.1 초기 경로 설정

송신자는 액티브 패킷을 보내기 전에 먼저 다음 액티브 노드 주소를 찾기 위한 '경로 요청 패킷'을 보낸다. '경로 요청 패킷'은 액티브 패킷으로서 IP '라우터 경보 옵션(Router Alert Option)'을 포함하고, 전송해야 할 액티브 패킷들의 송신지 주소와 최종 목적지 주소, 초기 일련번호, 전송할 데이터의 양, 플로우 식별자, 패킷 일련번호를 포함한다. IP의 '라우터 경보 옵션'은 목적지가 아닌 중간의 액티브 라우터에서 실행 가능하도록 해준다. 중간의 액티브 라우터가 '경로 요청 패킷'을 받으면 '경로 요청 패킷'을 보낸 바로전 단계의 액티브 노드로 자신의 주소를 실은 '경로 응답 패킷'을 보낸다. '경로 응답 패킷'은 다음 액티브 노드의 주소, 플로우 식별자, 패킷 일련번호를 포함해야한다. 동시에 이 노드는 '경로 요청 패킷'을 다음 액티브 노드를 찾기 위하여 최종 목적지를 향하여 전송한다. '경로 요청 패킷'을 보낸 액티브 노드가 '경로 요청 패킷'을 받으면 이 패킷에 담긴 다음 액티브 노드의 주소로 액티브 패킷들을 전송하기 시작한다. 액티브 패킷을 받은 액티브 노드는 이미 다음 액티브 노드에 대한 '경로 응답 패킷'이 도착했다면 다음 액티브 노드를 향해 액티브 패킷을 계속 전송하고, 만약 '경로 응답 패킷'이 아직 도착하지 않았다면 '경로 응답 패킷'이 도착할 때까지 기다려야 한다. 위와 같은 과정을 액티브 패킷이 최종 목적지에 도착할 때까지 반복한다. (그림 3)은 위의 과정을 그림으로 나타낸 것이다.

이 방식은 '경로 요청 패킷'이 다음 액티브 노드에 도착하고 '경로 응답 패킷'이 돌아올 때까지 기다려야 하는 오버헤드가 있다. 그러나 네트워크가 커지고 전송량이 늘어남에 따라 초기 경로 설정을 위한 지연이 차지하는 비율은 작아진다. 중간의 액티브 라우터에서 소요되는 지연을 줄이기 위하여 '복사 후 전송(Copy-and-Forward)' 방식을 제안한다.



(그림 3) 액티브 패킷의 경로 설정 과정

2.2.2 '복사 후 전송'

목적지까지의 중간 액티브 라우터는 IP 패킷이 들어오면 특별한 처리 없이 단순히 다음 노드로 전달한다. 그러나 처리 할 수 있는 액티브 패킷이 들어오면 패킷의 내용을 실행한 후 다음 노드로 전송한다. 따라서 중간 액티브 라우터에서 지연이 발생한다. 이 지연은 중간의 액티브 노드의 개수가 늘어나고 전송하는 액티브 패킷의 양이 증가할수록 더욱 커지게 된다.

'복사 후 전송'은 액티브 패킷이 모두 들어올 때까지 기다리지 않고 들어오는 대로 사본은 남기고 원본은 다음 노드로 전송시키는 방식이다. 이 때 다음 노드로 전송하기 위해서는 '경로 응답 패킷'이 도착해 있어야 한다. 만약 '경로 응답 패킷'이 아직 도착하지 않았다면, '경로 응답 패킷'이 도착할 때까지 기다린 후 나머지 액티브 패킷이 모두 도착하지 않았더라도 저장해 놓은 액티브 패킷들을 전송하기 시작한다.

2.3 신뢰성 전송 기법

액티브 패킷을 전송하면 수신자는 액티브 패킷의 신뢰성 검사를 한다. 수신자가 손실을 감지하면 손실된 패킷의 재전송을 송신자에게 요청한다. 재전송 요청을 받은 송신자는 해당 패킷을 재전송한다. 이 때 손실된 패킷의 재전송은 최초의 송신자가 아닌 바로 전에 액티브 패킷이 실행된 액티브 노드로부터 이루어진다. 자세한 내용을 살펴보면 다음과 같다.

2.3.1 손실 감지

수신자 측에서 들어온 액티브 패킷을 순서대로 정렬하고 손실을 감지할 수 있도록 데이터를 일정 크기의 블록으로 나누고 각각에 대하여 일련번호를 할당한다. 데이터를 일정 크기의 블록으로 나누는 것은 IP 계층에서의 조각화(fragmentation)를 방지하기 위해서이다. IP는 신뢰성을 보장하지 않기

때문에 IP 계층에서 패킷이 손실되면 복구할 수 없다.

수신 노드가 들어오는 액티브 패킷들의 신뢰성 검사를 하기 위해서는 액티브 패킷들에 대한 기본 정보가 필요하다. 일련번호의 시작 번호나 전송될 데이터의 양 등에 대한 정보가 그것이다. 전송될 액티브 패킷에 대한 기본 정보는 경로 설정을 위해 전송하는 '경로 요청 패킷'을 통해 전달받을 수 있다.

2.3.2 손실 복구

송신 노드는 수신 노드로부터 '재전송 요구 패킷'을 받으면 패킷에 포함된 일련번호에 해당하는 액티브 패킷을 다시 전송한다. 이 때 '재전송 요구 패킷'을 보내고 재전송을 요구하는 노드는 액티브 패킷의 최종 목적지이거나, 또는 액티브 패킷이 실행되는 중간의 액티브 노드이다. 액티브 패킷을 실행하는 중간의 액티브 노드는 액티브 패킷을 받을 때마다 일련번호를 이용하여 신뢰성 검사를하고 손실을 감지하면 액티브 패킷을 전송한 바로 전 단계의 액티브 노드에 '재전송 요구 패킷'을 보내어 재전송을 요청한다. 기존의 IP 망에서의 손실 복구는 종단간에 이루어지는 반면에 액티브 패킷의 손실 복구는 홉 바이 홉(hop-by-hop)으로 이루어지게 된다. 홉 바이 홉 손실 복구는 손실 복구로 인한 네트워크의 부하를 네트워크 전체에 분산시킬 수 있다. 기존의 종단간의 손실 복구와는 달리 홉 바이 홉 손실 복구는 손실이 이루어진 영역 근처에서 복구가 이루어질 수 있으므로 재전송을 요구하는 패킷이나 재전송 패킷들이 네트워크 전체를 돌아다니지 않아도 되고 복구도 빨리 이루어질 수 있다. 따라서 손실 복구 시간을 줄일 수 있고 대역폭의 낭비도 줄일 수 있다. 또한 손실이 자주 일어나는 영역을 감지하는 데에도 이용할 수 있다.

3. 성능 평가

본 장에서는 시뮬레이션 모델을 설명하고 시뮬레이션 결과를 분석한다. 본 시뮬레이션에서는 액티브 패킷의 실행을 위해 제안하는 기법들을 액티브 노드에 추가함으로써 액티브 노드의 성능에 어떤 영향을 끼치는지 성능을 평가한다. 액티브 노드에 제안하는 액티브 엔진을 내장했을 경우와 액티브 엔진을 내장하지 않고 단순히 UDP를 통해서만 전송했을 경우, 속도가 느리지만 신뢰성을 보장하는 TCP를 통하여 전송했을 경우를 비교한다.

시뮬레이션은 UCB(University of California, Berkeley)의 LBNL(Lawrence Berkely National Laboratory)에서 개발한 ns2[10]를 사용했고 토폴로지 생성을 위하여 Georgia Technologies에서 개발한 GT-ITM[11]을 사용했다.

3.1 시뮬레이션 환경

액티브 엔진을 시뮬레이션하기 위한 환경은 GT-ITM으로 생성한 트랜짓-스텁(Transit-Stub) 구조로써 현재의 인터넷 환경과 가장 유사하다.[12, 13] 노드는 백본 노드가 11개인 총 117개로 구성되어 있다. 링크는 82개의 LAN-LAN 링크, 16개의 WAN-WAN 링크, 35개의 MAN_WAN 링크들로 이루어져 있다. 각 링크의 대역폭 및 지연은 <표 1>과 같다.

〈표 1〉 링크 특성

	LAN-LAN	MAN-WAN	WAN-WAN
대역폭(Mbps)	10	8448 (E2 carrier)	34368 (E3 carrier)
지 연(ms)	2	20	30

실제 인터넷과 비슷한 환경을 만들기 위해서 백그라운드 트래픽을 생성하였다. 트래픽은 TCP를 이용하는 HTTP, FTP, Telnet 트래픽과 UDP의 CBR 트래픽을 사용했다. 각 트래픽의 송신지와 수신지, 전송시간은 임의로 결정하도록 하였다. 시뮬레이션은 300초 동안 실행하였다. 인터넷은 정확한 트래픽의 비율을 결정하기가 힘들다. 본 논문에서는 [14]를 참조하여 트래픽의 비율을 정하였다. 트래픽에 대해서는 <표 2>에서 정리했다.

제안한 액티브 엔진과 성능을 비교하기 위한 모델로는 TCP와 UDP를 채택하였다. TCP는 흐름제어와 오류제어 등 인터넷에서 신뢰성 전송을 담당하는 대표적인 전송 프로토콜이다. 신뢰성을 보장하는 대신 많은 기능으로 인하여 부하가 크다. TCP는 최근 많이 이용하고 있는 Newreno 구현을 사용하였다. UDP는 액티브 엔진을 적용하지 않는 경우에 해당한다. 단순히 전송 기능만 제공하므로 신뢰성을 보장하지 못하는 반면, 부하가 작다. 비교를 공정하게 하기 위하여 같은 양의 데이터를 같은 패킷 크기로 전송하였다. 비교를 위한 트래픽은 모두 수신자와 송신자를 임의로 20개씩 택하여 전송하였다.

〈표 2〉 시뮬레이션 트래픽 설정

	백그라운드 트래픽				테스트 모델 (액티브엔진, TCP, UDP)
	HTTP	FTP	Telnet	UDP	
플로우 개수	400개부터 2배씩 증가	200개부터 2배씩 증가	100개부터 2배씩 증가	1000개부터 2배씩 증가	20개
패킷크기	임 의	500bytes	73bytes	64bytes	512bytes
플로우당 전송량	0~300초 사이에 임의로 시작하여 전송	0~200초 사이에 전송 시작하여 1Mbytes 전송	0~200초 사이에 전송 시작하여 100초 동안 전송	0~200초 사이에 전송 시작하여 10초 동 안 전송	0~200초 사이에 전송 시 작 하 여 1Mbytes 전송
송/수신자	임의로 선택				

3.2 시뮬레이션 결과 및 분석

본 실험에서는 백그라운드 트래픽이 없을 때와 백그라운드 트래픽이 있을때로 나누어서 시뮬레이션을 실행하였다. 백그라운드 트래픽이 없을 때의 테스트는 액티브 엔진이 손실이

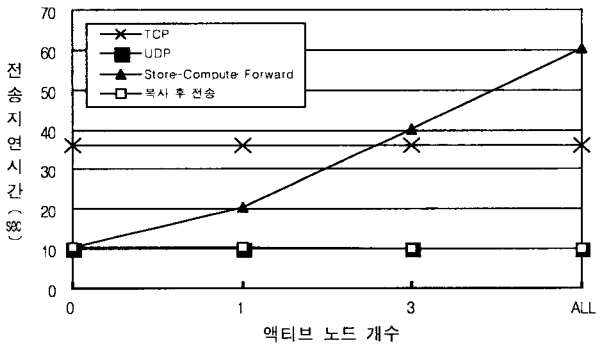
없는 상황에서 정상적으로 작동하는지 시험할 수 있고 손실 복구 능력과는 별도로 액티브 엔진의 경로 설정 프로세스의 성능을 테스트할 수 있다. 백그라운드 트래픽이 있을 때의 테스트는 실제 인터넷과 비슷한 상황에서 혼잡 등의 이유로 손실이 발생하였을 때 액티브 엔진의 손실 복구 성능을 테스트할 수 있다.

3.2.1 백그라운드 트래픽이 없을 때

백그라운드 트래픽이 없을때 액티브 엔진은 최고의 성능을 발휘할 수 있다. 혼잡으로 인하여 패킷의 손실이 발생하지 않기 때문에 순수하게 액티브 엔진의 경로 설정 기법을 테스트할 수 있다. 먼저 '복사 후 전송' 방식의 성능을 테스트한다. 그리고 중간에 액티브 노드가 없는 경우와 있는 경우로 나누어서 테스트한다.

● '복사 후 전송'

(그림 4)는 '복사 후 전송'방식의 전송지연시간을 측정한 결과를 그래프로 나타낸 것이다. 액티브 패킷을 모두 받을 때까지 기다려서 실행한 후 다음 노드로 전송하는 기존의 방식과 TCP, UDP를 비교하였다. 20개의 송신자가 트래픽을 1Mbytes를 보냈을 때의 평균 전송지연시간을 나타낸 것이다.



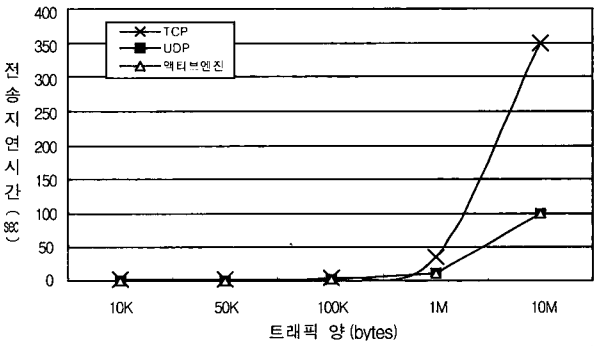
(그림 4) '복사 후 전송' 방식의 전송지연시간

TCP와 UDP는 종단간의 전송에만 관여하므로 TCP와 UDP는 액티브 노드의 개수가 늘어나도 일정한 값을 유지한다. '복사 후 전송' 방식은 액티브 노드의 개수가 증가할수록 전송지연시간이 점점 짧아지는 반면, 'Store-Compute-Forward' 방식은 전송지연시간이 점점 증가해서 나중에는 부하가 큰 TCP의 전송지연시간보다 길어지는 것을 볼 수 있다. 따라서 액티브 네트워크에서 '복사 후 전송' 방식을 사용하는 것이 효율적이다.

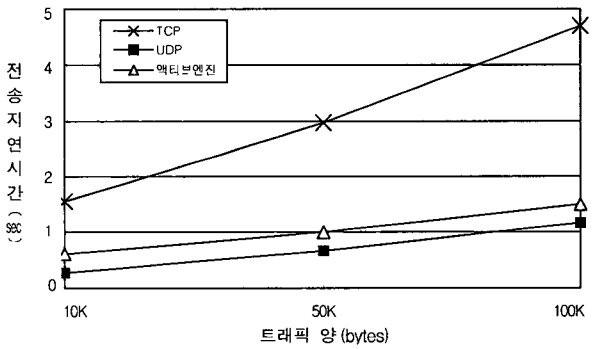
● 중간에 액티브 노드가 없는 경우

목적지까지 중간의 액티브 노드가 없는 특수한 상황을 가정한다. 송신자는 수신자까지의 중간 네트워크 상태에 대하여 전혀 모르고 액티브 패킷을 전송하므로 수신자까지 액티브 노드가 없는 상황이 있을 수 있다. 또한 중간에 액티브 노드가 없는 경우, 액티브 엔진과 종단간의 통신 프로토콜인 TCP, UDP의 성능을 같은 조건에서 비교할 수 있다. (그림 5)는 중간에 액티브 라우터가 전혀 없을 때, TCP와 UDP, 액티브 엔진의 종단간 전송지연시간을 전송하는 트래픽의 양을 증가시키면서 비교한 것이다.

TCP는 흐름제어를 위해 전송률을 조정하기 때문에 전송하는 트래픽의 양이 증가할수록 전송지연시간이 급격히 증가한다. (그림 5)의 (a)를 보면 제안한 액티브 엔진은 UDP와 전송지연시간이 크게 차이가 나지 않지만 TCP와는 전송량이 증가할수록 현저하게 전송지연시간에 차이가 남을 볼 수 있다. (그림 5)의 (b)에서 전송량을 10Kbytes에서 100Kbytes로 증가시켰을 때를 자세히 살펴보면, 액티브 엔진과 UDP는 전송지연시간에 일정한 차이만 보이고 있을 뿐 거의 비슷한 비율로 증가하고 있음을 알 수 있다. 이 차이는 초기에 액티브 엔진의 '경로 요청 패킷'과 '경로 응답 패킷'이 종단간에 교환됨으로써 발생한 지연이다. 제안한 액티브 엔진의 지연이 중간에 액티브 라우터가 전혀 없는 종단간의 통신에서도 TCP에 비교하여 상당히 적고 UDP에 가까움을 알 수 있다.



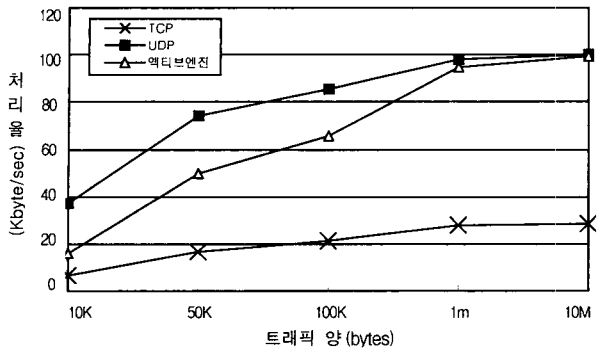
(a) 10Kbytes~10Mbytes



(b) 10Kbytes~100Kbytes

(그림 5) 액티브 라우터가 없을 때의 전송지연시간

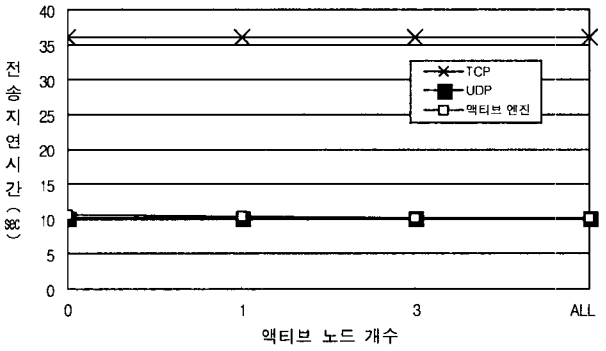
(그림 6)은 중간에 액티브 라우터가 없는 상황에서 전송량을 증가시켰을 때의 처리율을 측정한 것이다. 전송량을 늘일수록 수신자가 초당 처리하는 처리율은 점점 높아진다. 전송량이 늘어날수록 액티브 엔진의 처리율은 점점 UDP의 처리율에 가까워짐을 볼 수 있다.



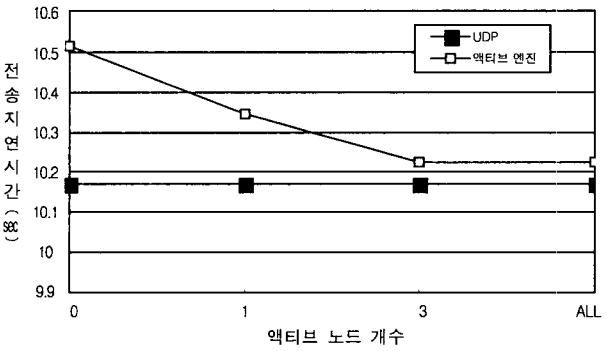
(그림 6) 중간에 액티브 노드가 없을 때의 프로토콜 별 처리율

(그림 5)와 (그림 6)에서 알 수 있듯이, 중간에 액티브 노드가 많아질수록 초기 지연이 차지하는 비율은 줄어들고 UDP에 가까워짐을 알 수 있다. 따라서 액티브 엔진이 신뢰성 전송과 경로 설정이라는 기능을 제공함에도 불구하고 어떤 기능도 제공하지 않는 UDP 전송에 비하여 성능에 큰 차이를 보이지 않는다는 것을 알 수 있다. 또한 전송하는 데이터의 양이 증가할수록 그 차이가 더욱 미미해지므로 대용량 액티브 패킷을 전송할때 더욱 효과적이라는 것을 알 수 있다. 없는 최악의 상황에서도 액티브 엔진의 전송지연시간과 처리율은 TCP보다 좋고, 전송량이

- 중간에 액티브 노드가 있을 경우



(a) TCP, UDP, 액티브 엔진 비교



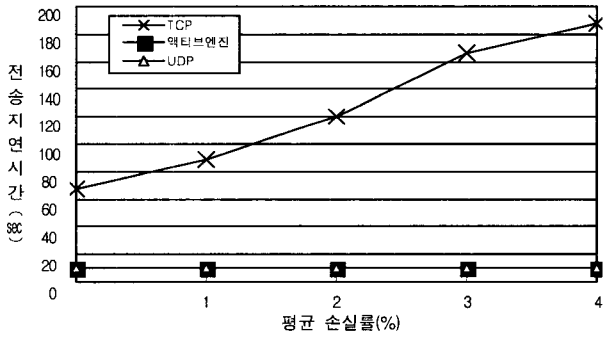
(b) UDP, 액티브 엔진 비교

(그림 7) 액티브 노드의 수에 따른 전송지연시간

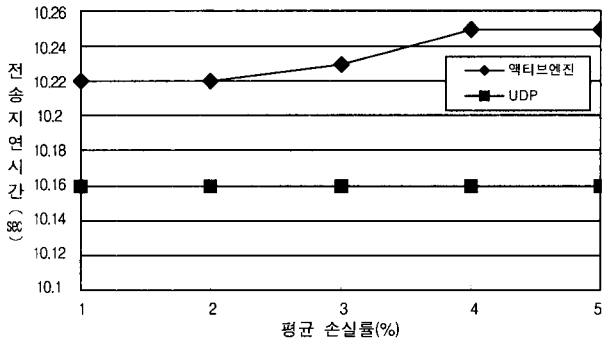
(그림 7)은 중간의 액티브 노드의 수를 증가시키면서 액티브 패킷을 보냈을 때의 전송지연시간을 나타낸다. 패킷은 1Mbytes를 전송했다. TCP와 UDP는 종단간의 전송만을 담당하므로 중간의 액티브 노드의 수와 상관없이 일정한 값을 갖는다. (그림 7)의 (a)를 보면 액티브 노드의 개수에 상관없이 액티브 엔진이 UDP의 전송지연시간과 거의 비슷한 결과를 나타내고 있다. (b)에서 UDP와 액티브 엔진의 전송지연시간만을 비교하면, 액티브 엔진의 전송지연시간이 액티브 노드의 수가 증가할수록 짧아지는 것을 볼 수 있다. 이는 액티브 노드의 수가 증가할수록 '경로 요청 패킷'에 대한 '경로 응답 패킷'이 돌아오는데 걸리는 시간이 줄어들기 때문이다.

3.2.2 백그라운드 트래픽이 있을 때

(그림 8)은 백그라운드 트래픽을 점점 증가시키면서 패킷 손실이 일어났을 때의 전송지연시간을 측정한 결과이다. 시뮬레이션은 백본 노드를 모두 액티브 노드로 설정하고 실행하였다. (그림 8)의 x축은 백그라운드 트래픽을 늘렸을 때의 평균 손실률을 가리키고 있다. UDP는 패킷 손실에 대하여 어떤 대응도 하지 않고 TCP는 손실의 원인을 혼잡으로 간주하고 전송률을 줄인다. 반면에 제안한 액티브 엔진은 전송률은 변화시키지 않고 수신자 측에서 손실을 감지하면 손실된 액티브 패킷에 대한 재전송을 송신자에게 요청한다. 따라서 손실을 복구하면서도 전송지연시간에는 큰 영향을 끼치지 않는다.



(a) TCP, UDP, 액티브 엔진 비교

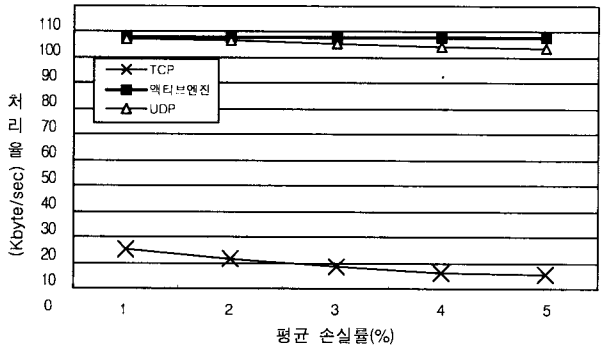


(a) TCP, UDP, 액티브 엔진 비교

(그림 8) 백그라운드 트래픽이 있을 때의 전송지연시간

(그림 8)의 (a)는 손실률이 증가할 수록 TCP는 전송지연시간이 급격히 늘어나지만 액티브 엔진은 전송지연시간은 손실에 어떤 대응도 하지 않는 UDP와 별 차이가 없다는 것을 보여준다. (그림 8)의 (b)는 액티브 엔진이 재전송 요청을 하고 복구하는 시간을 보여준다. 액티브 엔진의 전송지연시간과 TCP의 전송지연시간의 차이는 손실이 발생했을 때 더욱 커진다. (그림 8)을 통하여 액티브 엔진은 손실이 발생했을 때 손실을 복구하면서도 전송지연시간이 빠르다는 것을 알 수 있다.

(그림 9)는 백그라운드 트래픽이 있을때, 수신자가 초당 받는 데이터의 양을 측정한 결과이다. 이 그래프는 액티브 엔진의 처리율이 가장 높다는 것을 보여주고 있다. TCP는 손실을 감지할 때마다 전송률을 감소시키기 때문에 처리율이 가장 낮다. UDP의 경우, 전송률의 변화 없이 데이터를 일정하게 보내고 초기에 지연도 없으므로 전송지연시간은 짧지만 손실에 대한 복구 능력이 없기 때문에 수신자가 받는 데이터의 양이 감소하기 때문에 액티브 엔진보다 처리율이 낮다.



(그림 9) 백그라운드 트래픽이 있을 때의 처리율

따라서 네트워크에 부하가 늘어나면서 혼잡 등과 같은 이유로 손실이 발생할 때, 액티브 엔진은 전송지연시간에 영향을 끼치지 않으면서 손실을 복구하고 이 때 처리율은 TCP와 UDP보다 높음을 알 수 있다.

4. 결 론

본 논문에서는 하나의 프로그램을 수행하기 위한 액티브 패킷들이 같은 액티브 노드를 지나 실행되기 위한 기법과 모든 패킷들이 손실없이 도착하기 위한 기법을 제안하고 이 기법을 제공하는 액티브 엔진을 제시하였다.

액티브 패킷이 같은 액티브 노드를 지나도록 하기 위하여 액티브 노드를 찾기 위한 패킷을 보내는 방법을 제안했다. 이 기법은 노드들이 중간 노드들의 위치나 상태에 대한 정보를 가지고 있지 않아도 되고 기존의 IP 라우팅 프로토콜을 사용하며 IP 계층에서 조각화가 발생해도 같은 액티브 노드를 경유한다는 장점을 가지고 있다. 또한 중간 액티브 노드에서의 지연을 줄이기 위하여 액티브 패킷의 사본을 남기고 원본은 들어오는 대로 다음 노드로 전달하는 방식을 채택하였다. 손실 없이 액티브 패킷을 전송하기 위하여 수신자가 손실을 감지하여 송신자에게 재전송을 요구하도록 하는 신뢰성 전송 방식을 제안하였다. 제안한 신뢰성 전송방식은 패킷의 전송과 함께 손실 복구가 이루어지므로 복구로 인한 지연이 적다는 장점이 있다. 제안한 액티브 엔진은 기존의 프로토콜의 변경 없이 쉽게 액티브 노드에 적용할 수 있다. 또한 IP 패킷의 흐름에도 영향을 끼치지 않는다.

제안한 액티브 엔진을 모델링하여 성능을 평가, 분석한 결과, 제안한 액티브 엔진은 액티브 패킷을 중간의 액티브 노드에서 실행시키기 위해서는 필수적인 경로 설정과 신뢰성 전송 기능을 제공함에도 불구하고 액티브 노드의 성능을 해치지 않고 성능이 뛰어나다는 것을 입증할 수 있었다. 또한 전송하는 트래픽의 양이 크고 손실이 발생할 때 더욱 효과적이다.

본 연구에서는 네트워크의 상태를 노드들이 전혀 모른다는 가정 하에 경로 설정을 위하여 액티브 패킷을 전송하기 전에 액티브 라우터를 찾기 위한 패킷을 보내는 방법을 채택했다. 이 방법은 송신자가 중간의 액티브 라우터를 전혀 몰라도 된다는 장점이 있지만 초기 지연이 발생하고 액티브 라우터를 발견한 후에 액티브 라우터에 문제가 생길 경우 대처할 수 없다. 따라서 네트워크 내에서 동적으로 액티브 라우터의 위치와 상태를 알고 관리할 수 있는 다양한 방법들이 연구되어야 할 것이다.

참 고 문 헌

[1] D. L. Tennenhouse, J. M. Smith, W. D. Sncoskie, D. J. Wetherall, and G. J. Minden, "A Survey of Active Network Research," IEEE Communications Magazine, Vol.35, No.1, pp.80-86, 1997.

[2] B. Schwartz, W. Zhou, A. Jackson, W. T. Strayer, D. Rockwell, and C. Partridge, "Smart Packets for Active Networks," BBN Technologies, 1998.

[3] Postel, J., "Internet Protocol," IETF RFC 791, 1981.

[4] S. Deering, R. Hinden, "Internet Protocol, Version 6 (IPv6)," IETF RFC 2460, 1998.

[5] D. Katz, "IP Router Alert Option," IETF RFC2113, 1997.

[6] Danny Raz and Yuval Shavitt. "New Models and Algorithms for Active Networks." Lucent Technologies Technical Memorandum 10009674-991116-02. OpenArch 2001, 2001.

[7] Li-wei H. Lehman, Stephen J. Garland, and David L. Tennenhouse, "Active Reliable Multicast," IEEE INFOCOM '98, 1998.

[8] Sneha Kumar Kasera, Supratik Bhattacharyya, Mark Keaton, Diane Kiwior, Jim Kurose, Don Towsley and Steve Zabele, "Scalable Fair Reliable Multicast Using Active Services," IEEE Network Magazine (Special issue on Multicast), 2000.

[9] D. Scott Alexander, Bob Braden, Carl A. Gunter, Alden W. Jackson, Angelos D. Keromytis, Gary J. Minden, and David Wetherall, "Active Network Encapsulation Protocol (ANEP)," http://www.cis.upenn.edu/switchware/ANEP/docs/ANEP.

txt, 1997.
[10] NS Network Simulator, "http://www.isi.edu/nsnam/ns".
[11] Modeling Topology of Large Interne, "http://www.cc.gatech.edu/projects/gtitm/".
[12] Dante DeLuci, Katia Obraczka, "A Multicast Congestion Control Mechanism for Reliable Multicast," in the proceedings of IEEE INFOCOM, 1997.
[13] Christoph Hanle, Markus Hofmann, "Performance Comparison of Reliable Multicast Protocols using the Network Simulator ns-2," in the proceedings of IEEE Local Computer Networks(LCN), 1998.
[14] Kimberly C. Claffy, Hans-Werner Braun, George C. Polyzos, "A Parameterizable Methodology for Internet Traffic Flow Profiling," IEEE Journal on Selected Areas in Communications, 13(8), pp.1481-1494, 1995.

윤 보 영

e-mail : boyoung.yoon@samsung.com
2000년 이화여자대학교 컴퓨터학과(공학사)
2002년 이화여자대학교 대학원 컴퓨터학과 (공학석사)
2002년~현재 삼성전자 TN총괄 이동통신 사업팀 연구원
관심분야 : 이동통신, 네트워크구조, 차세대인터넷, 액티브네트워크

채 기 준

e-mail : kjchae@ewha.ac.kr
1982년 연세대학교 수학과 이학사
1984년 미국 Syracuse University 컴퓨터학과 이학석사
1990년 미국 North Carolina State University 컴퓨터공학과 공학박사
1990년~1992년 미국 해군사관학교 컴퓨터학과 조교수
1992년~현재 이화여자대학교 컴퓨터학과 교수
관심분야 : 네트워크 보안, 액티브 네트워크 보안 및 관리, 인터넷/무선통신망/고속통신망 프로토콜 설계 및 성능분석

남 택 용

e-mail : tynam@etri.re.kr
1987년 충남대학교 계산통계학과(이학사)
1990년 충남대학교 대학원 계산통계학과 (이학석사)
1987년~현재 한국전자통신연구원 정보보호연구본부 네트워크보안구조연구팀 팀장
관심분야 : 정보보호, 네트워크구조, 통신망관리, 차세대인터넷, 액티브네트워크