트랜스포트 계층 이동성 지원 방안에서의 오류 및 혼잡제어 알고리즘 성능분석

장 문 정† · 이 미 정††

요 약

IPv6 네트워크에서 SCTP를 기반으로 하여 트랜스포트 계층에서 이동성을 지원하는 방안을 제안하였다. 제안하는 방안에서는 단말의 이동에 따라 SCTP 연결에 매핑되는 종단 주소를 변경하기 위한 적절한 시점을 판단하는 기준을 제안하였고, 핸드오버에 효율적으로 대처하는 오류 복구 메커니즘을 제안하였다. 또한, 수치적 분석을 통해 기존 SCTP에 비하여 제안하는 방안이 핸드오버 지연과 손실복구시간을 줄임을 보였다.

키워드 : 트랜트포트 계층, 이동성, mSCTP, 오류 및 혼잡제어

Performance Analysis of Error and Congestion Control Algorithm in Transport Layer Mobility Support Approach

MoonJeong Jang† · MeeJeong Lee††

ABSTRACT

In this paper, we propose an approach to transport layer mobility support leveraging the SCTP extension dubbed dynamic address reconfiguration in IPv6 networks. Timing issues related to the end-to-end address management, and a novel error recovery mechanism associated with a handover are discussed. The proposed error recovery mechanism is analyzed and compared to that of the plain SCTP to show that it reduces the handover latency and error recovery time.

Key Words : Transport Layer, Mobility, mSCTP, Error and Congestion Control

1. 서 론

IPv6[1]를 기반으로 하는 차세대 인터넷 사용자들을 위해서 이동하면서 연속적으로 인터넷에 접속하고 통신할 수 있도록 지원하는 것은 필수적이다. 현재까지 Mobile IPv6(MIPv6) [2]를 기점으로 IPv6 기반 네트워크 계층에서의 다양한 이동성 지원 프로토콜들이 제안되어 왔다. 네트워크 계층에서 이동성을 지원하면 트랜스포트 계층에서 사용자의 이동을 인식하지 않고 세션을 계속 진행할 수 있지만, 네트워크에 이동성 지원을 위한 특별한 엔터티를 두어야 하고 이로 인해 터널링이나 삼각라우팅 등의 오버헤드가 발생하며 보안 문제가 복잡해진다. 또한 트랜스포트 계층에서 사용자의 이동으로 인한 데이터 경로 변경을 인식하지 못하기 때문에 이동에 대해 최적화된 트랜스포트 계층의 혼잡 및 오류 제어를 제공하기 어렵다 [2, 3, 4].

트랜스포트 계층에서 종단 간으로 이동성을 지원한다면 이와 같은 문제를 완화하거나 해결할 수 있는데, 이를 위해서는 기본적으로 트랜스포트 프로토콜에서 진행 중인 연결에 매핑되는 종단 주소를 변경하는 메카니즘을 지원하여야 한다. 2000년 IETF에 의해 범용 트랜스포트 프로토콜 표준 중 하나로 새로이 채택된 SCTP(Stream Control Transmission Protocol)[5, 6, 7]는 하나의 종단점에 대해 여러 개의 IP 주소를 지정할 수 있는 멀티호밍을 지원한다. 이러한 특성을 기반으로 [8]에서 제안된 SCTP의 동적인 주소관리 프로토콜을 이용하여 SCTP 연결에 매핑되는 종단의 IP 주소를 연결 진행 중 동적으로 바꿀 수 있도록 SCTP를 확장하는 방안인 mSCTP (mobile Stream Control Transmission Protocol)[9]에 대한 표준화 논의가 이루어지고 있다 [10].

본 논문에서는 이와 같은 mSCTP를 기반으로 하여 트랜스포트 계층에서 이동성을 지원하고, 핸드오버 시 발생하는 오류를 효율적으로 복구하는 방안을 제안하고자 한다. 즉, 핸드오버 시, SCTP 연결에 매핑되는 종단주소 및 데이터 전송

※ 본 논문은 정부(산업자원부)의 재원으로 한국산업기술평가원의 지원을 받아 수행된 신기술실용화기술개발사업의 연구결과임. 본 논문은 한국과학재단 기초과학연구사업(R04-2004-000-10073-0) 지원으로 수행되었음.
†준 회 원 : 이화여자대학교 컴퓨터학과 박사과정
††정 회 원 : 이화여자대학교 컴퓨터학과 교수
논문접수 : 2005년 1월 27일, 심사완료 : 2005년 5월 21일

목적지를 변경하는 시그널링들의 시점을 판단하는 적합한 기준을 제안하고, 핸드오버지연과 핸드오버 동안 발생한 오류를 복구하는데 소요되는 시간을 줄이는 오류제어 메커니즘을 제안한다. 그리고 제안한 오류 및 혼잡제어 알고리즘과 기존의 SCTP의 오류 및 혼잡제어 알고리즘과 분석적으로 성능을 비교한다.

본 논문의 구성은 다음과 같다. 1장의 서론에 이어 2장에서는 현재 mSCTP의 이동성 지원 방안과 오류 및 혼잡제어 방안에 대해서 설명한다. 3장에서 제안하는 방안의 동작방식과 제안하는 오류 및 혼잡제어 알고리즘을 설명하고, 제안한 알고리즘의 성능을 4장에서 수치적으로 분석한다. 마지막으로 5장에서는 결론을 맺는다.

2. 관련 연구

최근, 트랜스포트 계층에서의 이동성 지원 방안으로 mSCTP가 제안되었다[9]. mSCTP는 2000년 IETF에 의해 범용 트랜스포트 프로토콜 표준 중 하나로 새로이 채택된 SCTP의 멀티호밍 특성을 기반으로 SCTP 연결에 매핑되는 종단의 IP 주소를 연결 진행 중 동적으로 바꿀 수 있도록 확장한 프로토콜이다.

SCTP의 대표적인 특징인 '멀티호밍'은 하나의 단말에 여러 개의 IP 주소가 지정되는 것을 허용하는 것으로 SCTP에서는 이들 여러 주소들 가운데 하나의 주소만 데이터를 주고받는데 사용하며 이 주소에 해당하는 전송 경로를 '주경로'라고 부른다. 그리고 그 이외의 경로들은 패킷 재전송과 백업의 목적으로만 사용한다. 주경로에서 패킷 손실이 발생한 경우 이를 대체경로로 재전송하며 주경로에서 연속적으로 발생한 재전송 횟수가 임계치보다 커지는 경우에 송신측에서 해당 SCTP 어소시에이션에 속해 있는 목적지 주소 리스트들 중에서 무작위로 새로운 주경로를 위한 주소를 선택한다[5].

이 같은 멀티호밍 특성에 의해 하나의 종단점에 여러 개의 IP 주소를 동시에 매핑하는 것이 가능하기 때문에 MT(Mobile Terminal)가 새로운 IP 서브 네트워크로 이동하여 새 IP 주소를 획득했을 때, 현재 서브 네트워크에서의 IP 주소를 그대로 유지하면서 새로운 IP 주소를 어소시에이션 (SCTP에서의 연결을 '어소시에이션'이라 부름)에 추가할 수 있다. mSCTP는 이동성을 지원하기 위해 SCTP 어소시에이션에 매핑되는 IP 주소를 동적으로 변경하는 ADDIP, DELETEIP, Set-primary IP 시그널링을 사용한다[9]. ADDIP 시그널링은 상대에게 새로운 IP 주소를 SCTP 어소시에이션에 추가할 것을 요청하며, DELETEIP는 상대에게 현재 어소시에이션의 종단점 IP 주소로 등록되어 있는 것을 삭제할 것을 요청한다. 그리고 Set-priamry IP 시그널링은 현재 어소시에이션의 주경로를 변경할 것을 요청한다. 그런데 현재 mSCTP 명세[9]에서는 세션 이동성 지원을 위하여 이들 시그널링을 활용할 기본적인 틀만을 제안하였을 뿐 핸드오버 처리를 위해 이들 시그널링을 언제 어떠한 조건에 의해 트리거할 것인지에 대하여 구체적으로 제시하지 않았다.

어소시에이션에 대해 ADDIP, DELETEIP, Set-primary IP 등의 IP 주소 재구성이 필요한 경우, mSCTP는 관련 주소 정보를 ASCONF(Address Configuration Change) 제어 청크(SCTP에서의 전송단위)에 실어 상대방에 전송하며, 상대방은 ASCONF-ACK(Address Configuration Acknowledgment) 청크로 응답한다[8].

한편, mSCTP는 SCTP를 기반으로 하고 있기 때문에 SCTP의 오류 및 혼잡제어 메커니즘을 그대로 따른다. SCTP의 오류 및 혼잡제어 메커니즘은 TCP처럼 슬로우 스타트, 혼잡 회피, 빠른 재전송으로 이루어지는 윈도우 기반 혼잡제어 메커니즘을 사용하여 신뢰성 있는 전송을 보장한다. 기본적인 오류 및 혼잡제어 메커니즘은 TCP와 유사하며, 다음과 같은 차이점을 가진다[11].

- TCP에서 SACK은 선택 가능한 옵션이지만, SCTP에서는 필수적인 사항이다.
- TCP의 경우는 혼잡윈도우크기의 초기값이 일반적으로 1 MTU(Maximum Transmission Unit)이나, SCTP에서는 2 · MTU이다. 이는 동일한 네트워크 상황에서 SCTP가 TCP에 비해 좀 더 빠른 시간 내에 많은 데이터를 전송할 수 있도록 한다.
- TCP는 혼잡윈도우크기가 ACK된 패킷의 수에 의해 증가되나, SCTP에서는 ACK된 바이트 수에 의해 증가한다.
- TCP는 혼잡윈도우크기가 슬로우 스타트 임계치보다 적은 경우에만 슬로우 스타트를 하고 혼잡윈도우크기와 같은 값이 되면 혼잡회피 과정으로 들어가는데 반해, SCTP에서는 혼잡윈도우크기가 슬로우 스타트 임계치와 같을 때까지 슬로우 스타트를 수행한다.
- TCP는 3개의 연속된 중복 ACK을 받았을 때 빠른 재전송을 하는 반면, SCTP에서는 데이터의 손실을 알리는 4개의 연속된 중복 ACK을 받아야 빠른 재전송을 한다.
- TCP와는 달리 SCTP는 명시적인 빠른 복구 알고리즘을 사용하지 않는다. TCP나 SCTP의 혼잡 제어 알고리즘은 ACK 받지 못한 패킷들의 수보다 혼잡윈도우크기가 큰 경우에만 새로운 데이터의 전송을 허용한다. 그런데 TCP는 중복 ACK을 받을 때에는 아직 ACK 받지 못한 패킷들의 크기를 나타내는 flight size가 줄지 않기 때문에 혼잡윈도우크기를 증가시켜 주지 않으면 새로운 데이터를 전송하지 못한다. 따라서 명시적인 빠른 복구를 통해 중복 ACK을 받아야만 비로소 혼잡윈도우크기가 1MTU씩 증가되어 새로운 데이터의 전송이 가능하게 된다. 그러나 SCTP는 중복 ACK을 받는 동안에도 SACK 사용으로 인해 새롭게 ACK된 패킷을 파악할 수 있기 때문에 전송하고 받지 못한 패킷들의 크기를 나타내는 outstanding bytes가 감소하므로 혼잡윈도우 크기의 증가 없이도 새로운 데이터의 전송이 가능하게 된다. 따라서 SCTP는 명시적인 빠른 복구 알고리즘을 사용하지 않는다.
- TCP는 수신자의 윈도우 크기가 0이면, 송신자는 persist 모드를 수행하지만, SCTP에서는 이 모드가 없다. SCTP에서는 수신자의 윈도우 크기가 0이고, flight size가 0이면 데이

터 패킷 하나를 전송함으로써 수신자의 윈도우 크기의 변화 유무를 감지한다. 즉, SCTP는 항상 flight size가 1 이상이 되도록 한다.

3. 제안하는 방안

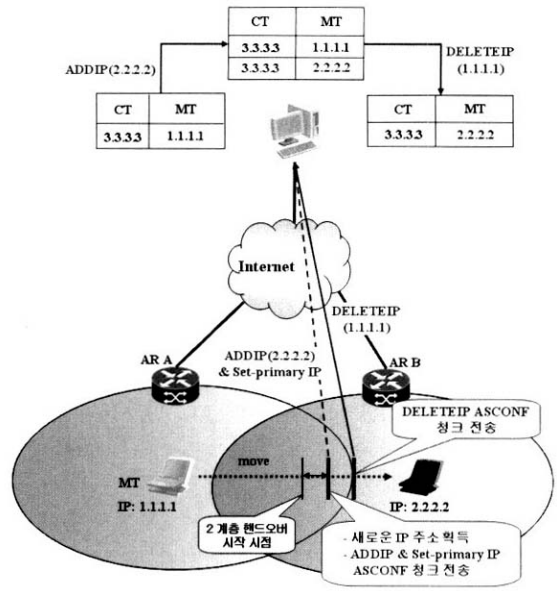
아직까지 대부분의 MT들이 단일 인터페이스만을 가지는 것을 감안하여 제안하는 방안에서는 MT가 단일 인터페이스를 가지는 경우를 가정한다. 또한 2계층 핸드오버와 새로운 IP 주소를 획득하는 방법으로 IPv6의 Stateless Address Auto-Configuration을 가정하며, Solicitation advertisement message를 사용하여 새로운 링크에서의 IP 주소를 획득한다고 가정한다. Solicitation advertisement message를 사용하지 않는다면 IPv6에서 기본적으로 약 3초 마다 브로드캐스팅 되는 라우터 광고 메시지에 의존하여 IP 주소를 획득할 수 있으나, 이와 같은 경우에는 IP 주소 획득에 걸리는 시간이 길어지게 되고 따라서 핸드오버 지연이 커진다.

멀티홈된 종단점의 경우, 2장에서 설명한 바와 같이 SCTP에서는 데이터 전송을 위해 주경로를 이용하며, 그 이외의 경로들은 패킷 재전송과 백업의 목적으로 한다[5]. 그런데 제안하는 방안에서는 MT가 단일 인터페이스를 가지는 경우를 가정하고 있으므로 핸드오버 시 일시적으로 CT (Correspondent Terminal)에서 유지하고 있는 이동단말의 IP 주소가 여러 개라 할지라도 항상 한 번에 하나의 경로만을 사용할 수 있기 때문에 주경로를 통해서만 데이터 전송과 재전송을 수행하도록 한다.

현재 mSCTP 명세에서는 새로운 IP 주소를 추가하는 시점과 이전 IP 주소를 삭제하는 시점을 구체적으로 정의하지 않았고, 핸드오버 시에 CT에서 데이터 전송 목적지를 변경하여야 하는데 언제 어떠한 기준에 의해 이를 변경할 것인지에 대하여 구체적으로 정의하지 않았다. 이와 같은 문제들이 정의되지 않고서는 현재 mSCTP는 이동성을 지원할 수 없다. 그러므로 이 절에서는 단일 인터페이스를 가진 MT가 하드 핸드오버를 수행하는 경우 mSCTP를 적용할 때, MT의 이동이 진행됨에 따라 MT가 CT에게 ADDIP, DELETEIP, Set-primary IP 시그널링을 위한 ASCONF 청크를 보내는 적절한 시점을 판단하는 기준을 정의하고, 트랜스포트 계층이 핸드오버 발생을 인지하는 점을 이용하여 SCTP의 오류 및 혼잡제어가 핸드오버에 효율적으로 대처하도록 하는 방안을 설명한다.

3.1 이동성 지원을 위한 주소관리방안

(그림 1)과 같이 MT의 이동이 진행되면 MT는 먼저 새로운 서브 네트워크에 대한 2계층 핸드오버를 수행하고 이어서 IP 주소를 획득한다. 단일 인터페이스 MT의 경우 2계층 핸드오버가 시작되면, 이미 이전 AR(Access Router)로부터의 데이터 수신이 불가능해지기 때문에, 가능한 한 신속하게 새로운 AR가 속한 도메인의 IP 주소를 등록하여 새로운 주소로 데이터가 전송되도록 해야 한다. 새로운 IP 주소를 등록할 수 있는 가장 빠른 시점은 새로운 IP주소가 획득된 시점이므로 이 때 ADDIP와 Set-Primary IP를 동시에 수행한다.



(그림 1) 제안하는 방안의 동작과정

ADDIP 및 Set-Primary IP와는 달리 이전 IP 주소에 대한 DELETEIP를 수행해야 하는 시점은 명백하지 않으며, 다음 세 가지의 옵션을 생각해 볼 수 있다:

(1) MT가 이전 AR의 전파범위를 벗어난 시점
(2) ADDIP, Set-Primary IP와 마찬가지로 MT가 새로운 AR에 해당하는 IP 주소를 획득한 시점
(3) MT가 변경된 새로운 주경로를 통해 첫 번째 데이터를 받은 시점

여기에서 한 가지 유념할 사항은 MT가 이전 IP 주소에 대해 DELETEIP를 수행하는 시점은 핸드오버 지연이나 작업량 등의 성능에 직접적으로 영향을 미치지 않는다는 것이다. 핸드오버 중의 데이터 전송성능은 새로운 경로로 데이터 전송이 언제부터 가능한지와 이전 경로를 통한 데이터 수신이 언제까지 지속되었는지에 의해 결정되는데, 전자와 후자 모두 DELETEIP와는 별개인 사건에 의해 그 시점이 결정되기 때문이다. 전자는 Set-Primary IP가 이루어지는 시점에 의해 결정되고, 후자도 단일 인터페이스이기 때문에 DELETEIP와는 별개로 2 계층 핸드오버가 시작되는 시점에 의해 결정된다. 단, DELETEIP를 언제 수행하느냐는 <표 1>에서 보는 바와 같이 프로토콜의 프로세싱 오버헤드에 영향을 미치게 된다.

<표 1> 3가지 DELETEIP 시점 방안에 대한 비교

비교항목 \ 방안		(1)	(2)	(3)
평상적인 이동	프로토콜 스택 내의 인터럽트 횟수	2	1	2
	ASCONF 청크 교환 횟수	2	1	2
핑퐁 이동	프로세싱 오버헤드	Set-primary IP 만 처리	ADDIP, Set-primary IP, DELETEIP 모두 처리	(1) or (2)와 동일함
	ASCONF 청크 교환하는 횟수	3가지 방안이 모두 동일함		

MT가 한 AR 영역에서 다른 AR의 영역을 향해 직진이동을 하는 경우에는 (2)번 방안이 다른 방안에 비해 프로토콜 프로세싱 면에서 약간 유리하다. 2번 방안은 ADDIP, Set-Primary IP, DELETEIP 세 가지의 시그널을 하나의 ASCONF 청크에 번들링해서 전송하기 때문에 1회의 핸드오버 처리를 위해 전송하는 ASCONF 청크 교환 횟수 및 ASCONF 청크 수신에 따른 프로토콜 스택 내의 인터럽트 오버헤드를 최소화하기 때문이다. 그러나 MT가 두 AR 영역 사이에서 핑퐁 형태의 이동을 하는 경우에는 (2)번 방안은 MT와 CT에 필요 없는 ADDIP, DELETEIP 처리 오버헤드를 부과하게 된다. (1)번 방안의 경우 AR이 변경될 때 Set-Primary IP만을 수행하면 되는 반면 (2)번 방안은 매번 ADDIP, Set-Primary IP, DELETEIP를 모두 수행해야 하기 때문이다. (3)번 방안의 경우는 핑퐁 이동시 만약 새로운 AR을 통해 첫 번째 데이터 패킷을 받기 전에 다시 이전 AR 영역으로의 이동이 발생한다면 이전 AR에 대한 DELETEIP를 아직 수행하지 않았으므로 Set-Primary IP만을 수행하면 되고, 이미 첫 번째 데이터 패킷을 받은 후 이전 AR로의 핑퐁 이동이 발생하는 경우라면 이미 이전 AR에 대한 DELETEIP를 수행한 상태이므로 ADDIP, Set-Primary IP를 모두 수행해야 한다. 제안하는 방안에서는 직진 이동의 경우 (2)번 방안에 비한 (1)번 방안의 추가적인 오버헤드가 핑퐁이동의 경우 (2)번 방안이 발생하는 추가적으로 발생하는 오버헤드보다 적다고 보고 핑퐁 이동이 발생하는 경우 적절하게 동작할 수 있는 (1)번 방안에 의해 DELETEIP를 수행하기로 한다. 즉, (그림 1)에서와 같이 MT가 AR A 영역으로부터 AR B의 영역으로 이동할 때, MT가 AR A의 전파 범위를 완전히 벗어난 시점에 DELETEIP를 수행한다.

3.2 핸드오버 시 오류 및 혼잡제어 알고리즘

이 절에서는 트랜스포트 계층에서 이동성을 감지할 수 있음을 이용하여 핸드오버 지연을 최소화하고, 핸드오버로 인한 손실을 복구하는 시간을 기존의 SCTP에 비하여 단축할 수 있는 매우 간단한 방안을 제안한다. 제안하는 SCTP 오류 및 혼잡제어는 데이터 전송 측 즉, CT에서의 변화만을 요구하며, 핸드오버가 발생한 경우에만 적용된다. 즉, CT의 SCTP는 기존의 SCTP와 동일하게 동작하나, MT로부터 핸드오버 발생을 의미하는 ADDIP/Set-Primary IP ASCONF 청크를 받은 경우에만 제안하는 오류 및 혼잡제어를 적용한다. MT로부터 ADDIP/Set-Primary IP ASCONF 청크를 받으면 CT는 즉시 기존에 전송한 패킷에 대한 재전송 혹은 ACK 타이머를 멈추고 MT의 새로운 주소로 지금까지 전송한 적이 없는 패킷 가운데 시퀀스 번호가 가장 낮은 패킷(Probe 패킷) 하나를 MT에게 전송한다. Probe 패킷으로 CT가 전송한 적이 없는 새로운 패킷을 사용하는 이유는 CT가 Probe 패킷을 전송한 이후로 수신하는 ACK들 가운데서 Probe 패킷에 대한 ACK을 구분하기 위해서이다. 2장에서 설명한 바와 같이 SCTP는 outstanding 데이터 크기가 0이면 수신자 윈도우가 0인 경우에도 패킷을 하나 전송할 수 있다. 일반적으로 하나의 경로에서 일정 기간 전송이 지속되면 SCTP 연결의 송수신원 간 파이프가 완전히 채워진 상태로 전송이 진행되므로 핸드오버가 발생했을 때 수신자 윈도우가 일반적으로 0이지만, 핸드오버가 발생한 경우에는 새로운 경로에 대한 outstanding 데이터 크기가 0이므로 제안하는 방안에서 사용하는 Probe 패킷은 기존 SCTP의 수신자 윈도우 제약을 침해하지 않는다. SCTP는 SACK을 사용하므로 이 Probe 패킷에 대한 ACK을 받으면 CT는 MT가 마지막으로 받은 패킷에 대한 정보와 핸드오버 동안 손실된 패킷에 대한 정보를 파악할 수 있다. Probe 패킷에 대한 ACK을 받으면 혼잡 윈도우를 2로 하여 슬로우 스타트에 의해 핸드오버에 의해 손실된 첫 번째 패킷부터 재전송 한다.

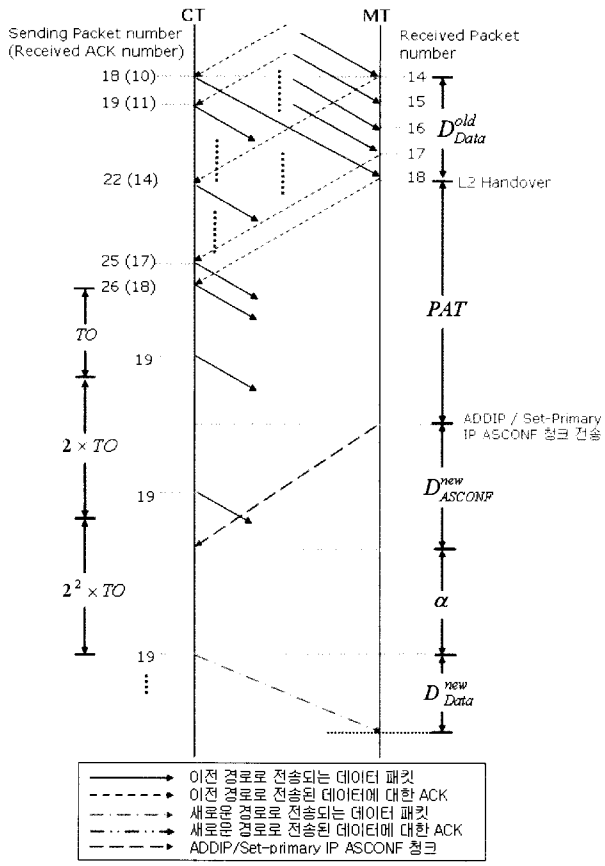
4. 제안하는 오류 및 혼잡제어 알고리즘의 성능분석

이 절에서는 핸드오버 지연과 핸드오버 중 발생한 패킷 손실을 복구하는데 소요되는 시간 등의 두 가지 측면에서 제안하는 오류 및 혼잡제어 알고리즘과 기존의 SCTP의 오류 및 혼잡제어 알고리즘과의 차이를 분석한다. 핸드오버 지연은 MT에서 2계층 핸드오버가 시작된 시점으로부터 MT가 새로운 경로를 통해 첫 번째 데이터 패킷을 전달 받기까지 소요된 시간으로 정의한다. 이들 값의 분석하는데 있어서, 전송 오류에 의한 패킷 손실이 없다고 가정하고, MT가 핸드오버 이전에 수신한 패킷에 대한 ACK은 CT에서 첫 번째 재전송 타임아웃이 발생하기 이전에 모두 CT에 도착한다고 가정한다.

SCTP의 오류 및 혼잡제어에 관하여 이 절에서 논의하는 내용에 관련된 주요 사항들만을 정리하면 다음과 같다. SCTP는 멀티호밍 지원을 고려하는데, 이 경우 SCTP에서는 수신자 윈도우는 하나의 SCTP 연결에 대하여 하나의 값을 유지하지만 혼잡 윈도우 크기와 outstanding 데이터 크기는 경로 별로 별도의 값을 유지한다. 이동환경에서 경로가 변경되는 경우에 이 사실을 적용하면 핸드오버 이전 경로로 전송한 패킷에 대한 ACK이 새로운 경로의 혼잡 윈도우에 영향을 미치지 못함을 의미한다. 또한, 재전송 타임아웃이 발생한 경우에 SCTP는 TCP와 마찬가지로 혼잡 윈도우를 1로부터 시작해서 슬로우 스타트로 패킷을 전송하며, 새로운 경로로 전송을 시작할 때는 혼잡 윈도우를 2부터 시작해서 역시 슬로우 스타트로 패킷을 전송한다.

각 핸드오버는 CT에서 핸드오버 전에 MT가 수신한 패킷에 대한 ACK을 모두 받은 후에야 ADDIP/Set-Primary IP ASCONF 청크를 받는 경우(그림 2)와 그렇지 않은 경우(그림 3)로 나누어 볼 수 있다.

먼저 전자의 경우를 살펴보기로 한다((그림 2) 참조). 이 경우 CT는 ADDIP/Set-Primary IP ASCONF 청크를 받은 후로 ACK을 받지 못하므로 ASCONF 청크를 받은 후에도 재전송 타임아웃이 발생할 때까지 새로운 경로로 데이터를 전송하지 못한다. 따라서 이 경우에는 새로운 경로로 데이터를 전송하기까지 한 번 이상의 재전송 타임아웃이 발생하게 되며, 재전송 타임아웃이 발생할 때마다 재전송 타이머 값은 두



(그림 2) 핸드오버에 의해 한 번 이상 재전송 타이머의 타임아웃이 발생하는 경우

배씩 증가하게 된다[5]. CT가 ASCONF 청크를 받기 전까지 n번의 재전송 타임아웃이 발생했다면 핸드오버 지연 $H_{timeout}$는 다음과 같다.

$$H_{timeout} = PAT + D^{new}_{ASCONF} + \alpha + D^{new}_{Data} \approx \sum_{i=0}^{n} 2^i \times TO,$$
$$where \quad 0 \le \alpha \le 2^n \times TO$$

여기에서 PAT ($Path\ Acquisition\ Time$)는 2계층 핸드오버가 시작된 시점으로부터 새로운 IP 주소를 획득하기까지 소요된 시간을 의미하며, D^x_y은 경로 x에서 y가 전달되는데 소요되는 지연을 의미한다. 그리고 α는 ASCONF 청크를 받은 시점으로부터 현재의 재전송 타이머가 만기되는 시점까지의 잔여시간으로 ASCONF 청크가 CT에 도착하는 시점이 지연되어 재전송 타임아웃 횟수가 늘어날수록 그 평균값이 커진다. 결과적으로, (그림 2)에서 보는 바와 같이 기존의 SCTP 오류제어를 그대로 따르면 핸드오버로 인해 재전송 타임아웃이 발생하는 경우, CT에서 ASCONF 청크를 수신하여도 재전송 타이머가 진행 중이므로 재전송 타임아웃이 될 때까지 재전송을 미루게 되고 이로 인해 핸드오버 지연이 길어지며, 이와 같은 부작용은 ASCONF 청크가 CT에 도착하기까지 발생하는 재전송 타임아웃 횟수에 대해 지수적으로 증가하게 된다.

또한 재전송 타임아웃이 발생했을 때 SCTP는 전송 윈도우를 1로부터 시작하고, 슬로우 스타트에 의해 전송을 진행하므로, 손실복구에 $k(\ge 1)$번의 RTT를 요구하는 최소패킷 손실 개수의 시퀀스는 초항이 1이고, 공비가 2인 등비수열이다. 즉, k번의 RTT가 요구되는 최소의 손실된 패킷 수 S_k는 다음과 같다.

$$S_k = 2^{k-1}$$

그러므로 핸드오버 동안 손실된 패킷 개수를 l이라 가정하고, k를 구하면 다음과 같다.

$$k = 1 + \lfloor \log_2 l \rfloor \ (l \ge 1)$$

그러므로 핸드오버로 인해 손실된 패킷의 수가 $l(\ge 1)$개일 때 이를 복구하는데 소요되는 시간 $R_{timeout}$은 다음과 같다.

$$R_{timeout} = (1 + \lfloor \log_2 l \rfloor) \times RTT \ (l \ge 1)$$

결과적으로, 핸드오버로 인해 CT에서 ASCONF 청크를 받기 전 i번의 재전송 타임아웃이 발생하였다면 CT에서 ADDIP/Set-Primary IP ASCONF 청크를 받은 이후에 핸드오버로 인해 손실된 $l(\ge 1)$개의 패킷이 모두 복구되는데 소요되는 시간 $L_{timeout}$은 다음과 같다.

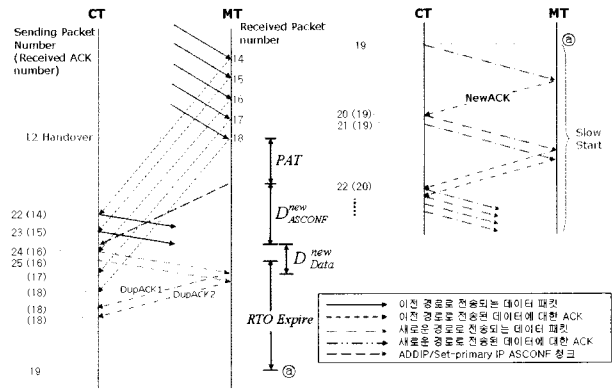
$$L_{timeout} = \alpha + R_{timeout} = \alpha + (1 + \lfloor \log_2 l \rfloor) \times RTT,$$
$$where\ 0 \le \alpha \le 2^i \times TO$$

반면에, (그림 3)에서 보듯이 새로운 경로에 대한 ADDIP/Set-Primary IP ASCONF 청크를 받은 이후에도 이전 경로로 전송된 패킷에 대한 ACK을 받게 되는 경우에는 ASCONF 청크를 받고 나서 ACK이 오는 즉시 새로운 경로로 데이터를 전송하게 되므로 이 때의 핸드오버 지연 $H_{no_timeout}$는 다음과 같다.

$$H_{no\_timeout} = PAT + D^{new}_{ASCONF} + D^{new}_{Data}$$

SCTP에서는 새로운 경로로의 전송을 혼잡윈도우 크기를 2로 하여 시작하므로 ASCONF 청크를 받은 이후 받게 되는 첫 번째 ACK (이전 경로로 전송된 패킷에 대한 ACK)에 대해 CT는 최대 두 개의 패킷을 새로운 경로로 전송하게 된다. 그러나 그 이후로 들어오는 이전 경로로 전송된 패킷에 대한 ACK에 대해서는 더 이상 새로운 경로로의 패킷 전송이 이루어지지 않는데, 이는 SCTP가 경로 별로 혼잡윈도우를 관리하기 때문이다. 즉, 이전 경로에 대한 ACK은 새로운 경로의 혼잡윈도우를 증가시키지 못하기 때문이다.

핸드오버로 인해 발생한 손실을 복구하기까지 소요되는 시간은 이들 새로운 경로로 전송된 두 개의 새로운 패킷에 대한 ACK을 이전 경로를 통해 MT에 배달된 마지막 패킷에 대한 ACK을 받은 이후에 받는지의 여부에 따라 달라지게 된다.

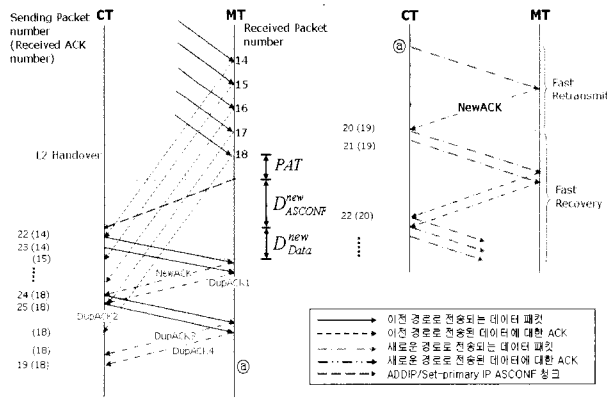


(a) 새로운 경로로 전송된 첫 번째 패킷에 대한 ACK이 DupACK이 되는 경우

(그림 3) 핸드오버에 의해 재전송 타이머의 타임아웃이 발생하지 않는 경우

이전 경로를 통해 MT에 배달된 마지막 패킷에 대한 ACK 이후에 이들 ACK을 받는다면 이들 ACK은 핸드오버 시 손실된 첫 번째 패킷에 대한 DupACK이 된다. 그런데, DupACK은 혼잡윈도우를 증가시키지 못하기 때문에 (그림 3 (a))에서 보는 것과 같이 새로운 경로로 전송한 패킷에 대한 ACK을 받고서도 CT는 여전히 새로운 경로로 추가적인 전송은 할 수 없고, 결국 재전송 타임아웃이 발생했을 때 핸드오버로 인해 손실된 패킷에 대한 복구가 시작된다. 따라서 핸드오버로 인해 손실된 패킷 수가 $l(\geq 1)$개일 때 이를 복구하는데 소요되는 시간은 $R_{timeout}$과 같고, CT에서 ADDIP/Set-Primary IP ASCONF 청크를 받은 이후에 핸드오버로 인해 손실된 $l(\geq 1)$개의 패킷이 모두 복구되기까지 소요된 시간 $L_{no-timeout(a)}$는 다음과 같다.

$$L_{no-timeout(a)} = RTO + R_{timeout} = RTO + (1 + \lfloor \log_2 l \rfloor) \times RTT$$



(b) 새로운 경로로 전송된 첫 번째 패킷에 대한 ACK이 NewACK이 되는 경우

(그림 3) 핸드오버에 의해 재전송 타이머의 타임아웃이 발생하지 않는 경우

새로운 경로로 전송된 패킷에 대한 ACK을 이전 경로로 배달된 마지막 패킷에 대한 ACK보다 먼저 받게 되는 경우에는 이들 새로운 경로에 대한 ACK 가운데 첫 번째 ACK이 NewACK이 되기 때문에 (그림 3(b))에서 보는 바와 같이 이 첫 번째 NewACK을 받았을 때 CT가 두 개의 새로운 패킷을 추가적으로 전송한다. 그러나 그 이후로 들어오는 ACK은 역시 모두 핸드오버 시 손실된 첫 번째 패킷에 대한 DupACK이기 때문에 더 이상 새로운 경로로 패킷을 전송하지 못한다. 그런데, 이 경우는 (그림 3(b))에서 보는 바와 같이 핸드오버로 인해 손실된 첫 번째 패킷에 대하여 궁극적으로 4개의 DupACK을 받게 되기 때문에 Fast Retransmit 메커니즘을 통해 핸드오버로 인해 손실된 첫 번째 패킷을 복구하며 그 나머지 패킷들은 Fast Recovery 메커니즘을 통해 복구한다. 이 때 Fast Recovery 단계에서 처음 혼잡 윈도우 크기는 3보다 작다. 이는 CT가 ASCONF 청크를 받은 후, 새로운 경로의 혼잡윈도우를 2로부터 시작하고 Fast Retransmit이 있기까지 단 하나의 ACK만이 혼잡윈도우를 증가시킬 수 있는 NewACK이기 때문이다.

SCTP Fast Recovery는 RTT마다 혼잡 윈도우를 하나씩 증가시키므로, 손실복구에 $k(\geq 2)$번의 RTT를 요구하는 최소 패킷 손실 개수의 시퀀스는 초항이 2이고, 계차가 $(k+2)$인 계차수열이다. 즉, k번의 RTT가 요구되는 최소의 손실된 패킷 수 S_k는 다음과 같다.

$$S_k = 2 + \sum_{i=1}^{k-1}(i+2) = \frac{k^2+3k}{2} \quad (k \geq 2)$$

핸드오버 동안 손실된 패킷을 복구하는데 소요되는 시간이 RTT인 경우를 커버하기 위해서 k에 $k-1$을 대입하면,

$$S_{k-1} = \frac{(k-1)^2+3(k-1)}{2} = \frac{k^2+k-2}{2} \quad (k-1 \geq 1)$$

따라서 핸드오버 동안 손실된 패킷 개수를 l 이라면

$$l = \left\lfloor \frac{k^2+k-2}{2} \right\rfloor$$

이고, 여기에서 k를 구하면 다음과 같다.

$$k = \left\lfloor \frac{-1+\sqrt{1-4(-2-2l)}}{2} \right\rfloor = \left\lfloor \frac{\sqrt{8l+9}-1}{2} \right\rfloor, \quad for \quad l \geq 1.$$

그러므로 핸드오버로 인해 l개의 패킷이 손실된 경우 이들 패킷을 복구하는데 소요되는 시간 $R_{no-timeout(b)}$은 다음과 같다.

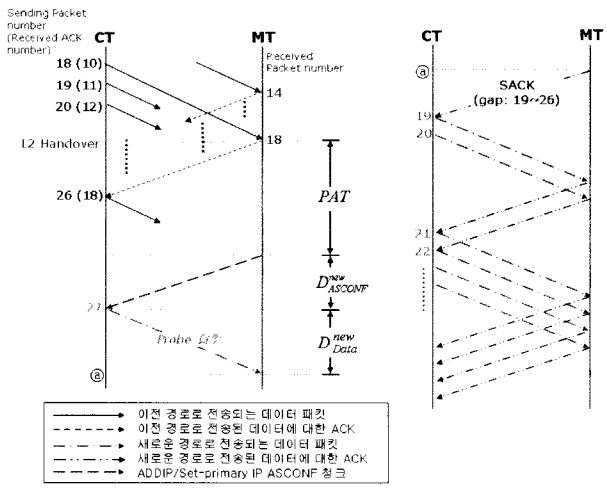
$$R_{no-timeout(b)} = k \times RTT = \left\lfloor \frac{\sqrt{8l+9}-1}{2} \right\rfloor \times RTT \quad (l \geq 1)$$

결과적으로, 핸드오버로 인해 재전송 타임아웃이 발생하지 않는 경우에는 CT에서 ADDIP/Set-Primary IP ASCONF 청크를 받은 이후에 핸드오버로 인해 손실된 $l(\geq 1)$개의 패킷이 모두 복구되는데 소요되는 시간 $L_{no-timeout(b)}$는 다음과 같다.

$$L_{no-timeout(b)} = 2 \times RTT + R_{no-timeout(b)}$$
$$= \left(2 + \left\lfloor \frac{\sqrt{8l+9}-1}{2} \right\rfloor\right) \times RTT$$

결론적으로, 핸드오버로 인해 타임아웃이 발생하지 않은 경우에도 핸드오버 지연은 적지만, 핸드오버로 인한 손실 복구에 소요되는 시간은 크다.

한편, (그림 4)는 3.2절에서 설명한 제안하는 오류 및 혼잡제어를 적용할 때 핸드오버 처리 과정을 보인 것이다.



(그림 4) 제안하는 방안의 오류 및 혼잡제어 메커니즘

제안하는 방안의 핸드오버 지연 H_{probe}는 다음과 같다.

$$H_{probe} = PAT + D^{new}_{ASCONF} + D^{new}_{Data}$$

즉, 제안하는 오류 및 혼잡제어는 재전송 타임아웃이 발생하는 경우에는 기존의 SCTP에 비하여 핸드오버 지연을 $\alpha(0 \le \alpha \le 2^{i} \times TO)$만큼 감소시킨다.

한편 제안하는 오류 및 혼잡제어에서는 새로운 주경로로는 항상 슬로우 스타트를 수행하므로, 손실복구에 $k(k \ge 0)$번의 RTT를 요구하는 최소패킷 손실개수의 시퀀스는 초항이 1이고, 계차가 2^{k}인 계차수열이다. 즉, k번의 RTT가 요구되는 최소의 손실된 패킷 수 S_k는 다음과 같다.

$$S_k = 1 + \sum_{i=1}^{k-1} 2^{k} = 2^{k} - 1 \ (k \ge 0)$$

그러므로 핸드오버 동안 손실된 패킷 개수를 $l(l \ge 0)$이라고 하고, k를 구하면 다음과 같다.

$$k = \lfloor \log_2(l+1) \rfloor$$

따라서 핸드오버로 인해 손실된 패킷을 복구하는데 소요되는 시간 R_{probe}는 다음과 같다.

$$R_{probe} = \lfloor \log_2(l+1) \rfloor \times RTT$$

그리고 CT에서 ASCONF 청크를 수신한 시점으로부터 손실 복구가 완료될 때까지 소요되는 시간 L_{probe}는 다음과 같다.

$$L_{probe} = RTT + R_{probe} = RTT + \lfloor \log_2(l+1) \rfloor$$
$$\times RTT = \{1 + \lfloor \log_2(l+1) \rfloor\} \times RTT$$

따라서 핸드오버로 인한 재전송 타임아웃이 발생하는 경우와 그렇지 않은 경우 각각에 대하여 제안하는 오류 및 혼잡제어는 다음의 E 만큼 핸드오버로 인한 손실 복구 시점을 앞당길 수 있다.

$$E = \begin{cases} L_{timeout} - L_{probe} \approx \alpha, & where \quad 0 \le \alpha \le 2^{i} \times TO, \quad if \ (number\ of\ rtx\ timeout > 0) \\ L_{no-timeout(a)} - L_{probe} \approx RTO, & \\ if \ (number\ of\ rtx\ timeout = 0\ and\ 새로운\ 경로로\ 전송된\ 첫\ 번째\ 패킷에\ 대한\ ACK = DupACK) & \\ L_{no-timeout(b)} - L_{probe} \approx RTT, & \\ if \ (number\ of\ rtx\ timeout = 0\ and\ 새로운\ 경로로\ 전송된\ 첫\ 번째\ 패킷에\ 대한\ ACK = NewACK) & \end{cases}$$

5. 결 론

IPv6 네트워크에서 SCTP를 기반으로 하여 트랜스포트 계층에서의 이동성을 지원하는 방안을 제안하였다. 제안하는 SCTP에서는 이동에 따라 SCTP 연결에 매핑되는 종단 주소를 변경하는 방안을 추가하였고, 핸드오버 지연을 줄이고 핸드오버 동안 발생한 오류를 복구하는 시간을 줄이는 방안을 제안하였다. 제안한 방안의 오류복구시간을 기존 SCTP를 사용하는 경우의 오류복구시간과 분석적으로 비교한 결과, MT의 이동성을 인지하는 제안하는 오류복구방안이 핸드오버로 인하여 발생한 손실을 복구하는 시점을 앞당김으로써 핸드오버 지연과 손실복구시간을 최소화함을 확인할 수 있었다.

참 고 문 헌

[1] S. Deering, R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December, 1998.

[2] C. Perkins, "Mobility Support in IPv6", RFC3775, June, 2004.

[3] H. Soliman, C. Catelluccia, "Hierarchical Mobile IPv6 mobility management(HMIPv6)", draft-ietf-mip shop- hmipv6-02.txt, June, 2004.

[4] R. Koodli, "Fast Handovers for Mobile IPv6", draft-ietf-mipshop-fast-mipv6-02.txt, July, 2004.

[5] R. Stewart, et al., "Stream Control Transmission Protocol", RFC 296, October, 2000.

[6] L. Ong, J. Yoakum, "An Introduction to the Stream Control Transmission Protocol (SCTP)", RFC 3286, May, 2002.

[7] R. Stewart, et al., "SCTP implementer's Guide", draft-ietf-tsvwg-sctpimpguide-12.txt, October, 2004.

[8] R. Stewart, "Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration", draft-ietf-tsvwg-addip-sctp-09.txt, June, 2004.

[9] M. Riegel and M. Tuexen, "Mobile SCTP", draft-reigel-tuexen-mobile-sctp-04.txt, October, 2004.

[10] S. Koh, M. Lee, et al., "Mobile SCTP for Transport Layer Mobility", draft-sjkoh-sctp-mobility-04.txt, June, 2004.

[11] J. Song, M. Lee, S. Koh, "SCTP의 멀티호밍 특성에 대한 성능 평가", 정보처리학회논문지C, 제11-C권 제2호, pp.245~252, 2004년 4월.

장 문 정

e-mail : mjchang@ewhain.net

2001년 이화여자대학교 컴퓨터학과(학사)

2003년 이화여자대학교 과학기술대학원 컴퓨터학과(석사)

2003년~현재 이화여자대학교 과학기술대학원 컴퓨터학과(박사과정)

관심분야 : SCTP multihoming, Load Sharing, mobile SCTP, vertical handover

이 미 정

e-mail : lmj@ewha.ac.kr

1983년~1987년 이화여자대학교 전자계산학 학사

1987년~1989년 University of North Carolina at Chapel Hill 컴퓨터학 석사

1990년~1994년 North Carolina State University 컴퓨터공학 박사

1994년~현재 이화여자대학교 공과대학 컴퓨터학과 교수

관심분야 : 고속 통신 프로토콜 설계 및 성능 분석, 멀티미디어 전송을 위한 트래픽 제어, 인터넷에서의 QoS 지원, 무선 이동 네트워크, Ad-hoc 네트워크