(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2010-0005721
(43) 공개일자 2010년01월15일

(51) Int. Cl.
H04L 12/54 (2006.01)
(21) 출원번호 10-2009-7024504
(22) 출원일자 2008년05월26일
심사청구일자 2009년11월27일
(85) 번역문제출일자 2009년11월24일
(86) 국제출원번호 PCT/CA2008/001000
(87) 국제공개번호 WO 2008/144902
국제공개일자 2008년12월04일
(30) 우선권주장
11/807,240 2007년05월25일 미국(US)

(71) 출원인
시에라 와이어리스 인코포레이티드
캐나다 브리티쉬 콜롬비아 브이6브이 3에이4 리치몬드 와이어리스 웨이 13811
(72) 발명자
보스 구스타브 제랄드
캐나다 브리티쉬 콜럼비아 브이4피 1엔8 써리 25th 애비뉴 14882
와웅 윌리엄
캐나다 브리티쉬 콜럼비아 브이5에이 3제트6 버너비 린데일 크레슨트 3515
맥코넬 피터
캐나다 브리티쉬 콜럼비아 브이6티 1씨7 밴쿠버 웨스트 7th 애비뉴 4755
(74) 대리인
리앤목특허법인

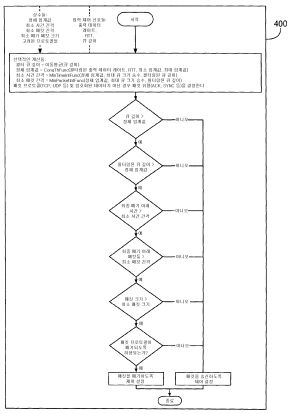
전체 청구항 수 : 총 24 항

(54) 네트워크 장치를 위한 버퍼 제어 방법

(57) 요 약

본 발명의 실시예들은 큐 버퍼 관리 방법을 제공한다. 본 발명의 일 실시예를 위해, 인터넷 프로토콜 데이터는 데이터 소스 장치에서 생성된다. 상기 생성된 데이터는 하나 이상의 네트워크 장치들로 전달되고 하나 이상의 네트워크 장치들에서 수신된다. 상기 생성된 데이터의 일부분들은 생성된 데이터의 개선된 데이터 흐름을 달성하기 위하여 특정 기준들에 기초하여 선택적으로 삭제된다. 상기 특정 기준들은 최종 폐기 이래의 시간, 최종 폐기 이래의 패킷들의 수, 패킷 프로토콜, 패킷 크기 및 이들의 조합으로 이루어진 그룹 중에서 선택된다. 예를 들어, 본 발명의 일 실시예에서, 생성된 데이터는 흐름 제어가능 인터페이스를 통해 전달된다.

대 표 도 - 도4



특허청구의 범위

청구항 1

하나 이상의 네트워크 장치들에서 인터넷 프로토콜 데이터를 수신하는 단계로서, 상기 인터넷 프로토콜 데이터는 데이터 소스 장치에서 생성되고 상기 하나 이상의 네트워크 장치들로 전달되는 단계; 및

상기 인터넷 프로토콜 데이터의 개선된 데이터 흐름을 달성하기 위하여 특정 기준들에 기초하여 상기 데이터의 일부분들을 선택적으로 삭제하는 단계로서, 상기 특정 기준들은 최종 폐기 이래의 시간, 최종 폐기 이래의 패킷들의 수, 패킷 프로토콜, 패킷 크기 및 이들의 조합으로 이루어진 그룹 중에서 선택되는 단계를 포함하는 것을 특징으로 하는 방법.

청구항 2

제1항에 있어서, 상기 특정 기준들 중 하나 이상은 동적으로 결정되는 것을 특징으로 하는 방법.

청구항 3

제1항에 있어서, 하나 이상의 특정 기준들은 동적으로 결정된 대응하는 값을 갖는 것을 특징으로 하는 방법.

청구항 4

제1항에 있어서, 상기 생성된 데이터는 암호화되고 상기 특정 기준들은 패킷 크기를 포함하는 것을 특징으로 하는 방법.

청구항 5

흐름 제어가능한 인터페이스를 통해 전달된 인터넷 프로토콜 데이터를 하나 이상의 네트워크 장치들에서 수신하는 단계로서, 상기 인터넷 프로토콜 데이터는 데이터 소스 장치에서 생성되고 상기 하나 이상의 네트워크 장치들로 전달되는 단계; 및

상기 인터넷 프로토콜 데이터의 개선된 데이터 흐름을 달성하기 위하여 특정 기준들에 기초하여 상기 데이터의 일부분들을 선택적으로 삭제하는 단계를 포함하는 것을 특징으로 하는 방법.

청구항 6

제5항에 있어서, 상기 특정 기준들 중 하나 이상은 동적으로 결정되는 것을 특징으로 하는 방법.

청구항 7

제5항에 있어서, 하나 이상의 특정 기준들은 동적으로 결정된 대응하는 값을 갖는 것을 특징으로 하는 방법.

청구항 8

제5항에 있어서, 상기 생성된 데이터는 암호화되고 상기 특정 기준들은 패킷 크기를 포함하는 것을 특징으로 하는 방법.

청구항 9

데이터 소스 장치에 통신가능하게 연결된 네트워크 장치를 포함하고, 상기 네트워크 장치는 상기 데이터 소스 장치에서 생성된 인터넷 프로토콜 데이터를 수신하도록 구성되며 상기 인터넷 프로토콜 데이터의 개선된 데이터 흐름을 달성하기 위하여 특정 기준들에 기초하여 상기 데이터의 일부분들을 선택적으로 삭제하도록 구성되고, 상기 특정 기준들은 최종 폐기 이래의 시간, 최종 폐기 이래의 패킷들의 수, 패킷 프로토콜, 패킷 크기 및 이들의 조합으로 이루어진 그룹 중에서 선택되는 것을 특징으로 하는 장치.

청구항 10

제9항에 있어서, 상기 특정 기준들 중 하나 이상은 동적으로 결정되는 것을 특징으로 하는 장치.

청구항 11

제9항에 있어서, 하나 이상의 특정 기준들은 동적으로 결정된 대응하는 값을 갖는 것을 특징으로 하는 장치.

청구항 12

제9항에 있어서, 상기 생성된 데이터는 암호화되고 상기 특정 기준들은 패킷 크기를 포함하는 것을 특징으로 하는 장치.

청구항 13

데이터 소스 장치에 통신가능하게 연결된 네트워크 장치를 포함하고, 상기 네트워크 장치는 상기 데이터 소스 장치에서 생성된 인터넷 프로토콜 데이터를 수신하도록 구성되며 상기 인터넷 프로토콜 데이터의 개선된 데이터 흐름을 달성하기 위하여 특정 기준들에 기초하여 상기 데이터의 일부분들을 선택적으로 삭제하도록 구성되고, 상기 인터넷 프로토콜 데이터는 흐름 제어가능 인터페이스를 통해 전달되는 것을 특징으로 하는 장치.

청구항 14

제13항에 있어서, 상기 특정 기준들 중 하나 이상은 동적으로 결정되는 것을 특징으로 하는 장치.

청구항 15

제13항에 있어서, 하나 이상의 특정 기준들은 동적으로 결정된 대응하는 값을 갖는 것을 특징으로 하는 장치.

청구항 16

제13항에 있어서, 상기 생성된 데이터는 암호화되고 상기 특정 기준들은 패킷 크기를 포함하는 것을 특징으로 하는 장치.

청구항 17

프로세서에 의해 실행되는 경우, 프로세서로 하여금,

하나 이상의 네트워크 장치들에서 인터넷 프로토콜 데이터를 수신하는 단계로서, 상기 인터넷 프로토콜 데이터는 데이터 소스 장치에서 생성되고 상기 하나 이상의 네트워크 장치들로 전달되는 단계; 및

상기 인터넷 프로토콜 데이터의 개선된 데이터 흐름을 달성하기 위하여 특정 기준들에 기초하여 상기 데이터의 일부분들을 선택적으로 삭제하는 단계로서, 상기 특정 기준들은 최종 폐기 이래의 시간, 최종 폐기 이래의 패킷들의 수, 패킷 프로토콜, 패킷 크기 및 이들의 조합으로 이루어진 그룹 중에서 선택되는 단계를 포함하는 방법을 수행하도록 야기하는, 실행가능한 명령들을 제공하는 기계로 읽을 수 있는 매체.

청구항 18

제17항에 있어서, 상기 특정 기준들 중 하나 이상은 동적으로 결정되는 것을 특징으로 하는 기계로 읽을 수 있는 매체.

청구항 19

제17항에 있어서, 하나 이상의 특정 기준들은 동적으로 결정된 대응하는 값을 갖는 것을 특징으로 하는 기계로 읽을 수 있는 매체.

청구항 20

제17항에 있어서, 상기 생성된 데이터는 암호화되고 상기 특정 기준들은 패킷 크기를 포함하는 것을 특징으로 하는 기계로 읽을 수 있는 매체.

청구항 21

프로세서에 의해 실행되는 경우, 프로세서로 하여금,

흐름 제어가능한 인터페이스를 통해 전달된 인터넷 프로토콜 데이터를 하나 이상의 네트워크 장치들에서 수신하는 단계로서, 상기 인터넷 프로토콜 데이터는 데이터 소스 장치에서 생성되고 상기 하나 이상의 네트워크 장치들로 전달되는 단계; 및

상기 인터넷 프로토콜 데이터의 개선된 데이터 흐름을 달성하기 위하여 특정 기준들에 기초하여 상기 데이터의 일부분들을 선택적으로 삭제하는 단계를 포함하는 방법을 수행하도록 야기하는, 실행가능한 명령들을 제공하는 기계로 읽을 수 있는 매체.

청구항 22

제21항에 있어서, 상기 특정 기준들 중 하나 이상은 동적으로 결정되는 것을 특징으로 하는 기계로 읽을 수 있는 매체.

청구항 23

제21항에 있어서, 하나 이상의 특정 기준들은 동적으로 결정된 대응하는 값을 갖는 것을 특징으로 하는 기계로 읽을 수 있는 매체.

청구항 24

제21항에 있어서, 상기 생성된 데이터는 암호화되고 상기 특정 기준들은 패킷 크기를 포함하는 것을 특징으로 하는 기계로 읽을 수 있는 매체.

명 세 서

기 술 분 야

<1> 본 발명의 실시예들은 통신 네트워크들에 관한 것으로, 특히, 본 발명은 데이터 통신 시스템의 큐 버퍼를 제어하기 위한 방법들 및 시스템들에 관한 것이다.

배 경 기 술

<2> 큐 버퍼 관리의 관용적인 방법들은 흐름 제어가능 인터페이스(FCI)를 구현하는 것을 포함한다. 네트워크 장치가 FCI를 통해 데이터 소스를 제공하는 호스트에 접속되는 경우, 이들 장치들 간의 데이터 흐름은 네트워크 장치 내부의 큐가 특정 레벨을 초과할 때 종종 정지된다. FCI는 네트워크 장치의 큐가 항상 오버플로우하는 경우 데이터 손실을 방지하도록 구성된다. 네트워크 장치와 호스트 간의 데이터 흐름이 정지되는 경우, 호스트내의 네트워크 계층은 네트워크 계층 위의 흐름 제어가 전송 계층에 의해 제어되기 때문에 데이터를 큐잉하도록 요구될 것이다. 전송 계층(TCP, UDP 등) 및 (rwin과 같은) 전송 매개변수들에 의존하여, 종종 과도한 큐잉이 호스트에서 발생할 것이다.

<3> 이러한 시스템들은 과도한 큐잉과 연관된 단점들을 갖는다. 하나의 단점은 데이터가 데이터 소스로부터 데이터 목적지(예를 들어, 네트워크 장치)로 그리고 다시 데이터 소스로 링크를 가로지르도록 하는데 요구되는 시간의 증가를 야기할 것이다. 이 시간은 채널에 대한 라운드 트립 시간(RTT)으로 알려져 있다. RTT의 증가는 네트워크 장치의 최대 출력 레이트 곱하기 큐잉된 데이터의 양과 같다.

<4> 도 1은 종래 기술에 의한 FCI를 구현하는 시스템 및 그 단점들을 도시한 것이다. TCP 데이터에 대해, RTT의 증가 및 처리량의 감소의 근본적인 원인은 업링크 수신 윈도 크기가 원격 종단 서버에 의해 너무 크게 설정되어 있는 것이다. RTT의 증가는 다중 TCP 세션들이 하나의 스트림으로 다중화되는 경우 TCP 프로토콜 아래의 업링크의 데이터 경로에서의 과도한 큐잉에 의해 야기된다(UDP 데이터에 대해, 과도한 큐잉은 UDP 스트림 데이터 레이트가 네트워크 장치 출력 레이트를 초과하는 때에는 언제나 발생할 것이다.)

<5> TCP 스트림에 대한 처리량 저하는 전형적인 TCP FCI의 하기의 데이터 통신 프로세스들에 기인하여 발생한다. 애플리케이션은 그것이 송신할 수 있는 것보다 더 빠르게 네트워크 장치로 데이터를 송신하고 그래서 데이터는 도 1에 도시된 바와 같이, DL 확인응답(acknowledgement)과 함께 큐잉되어야 한다. 큐잉된 데이터는 원격 종단 데이터 소스(미도시)에 대한 DL TCP 확인응답의 지연을 야기하는 RTT의 증가를 야기한다. 시기적절하지 못한 TCP 확인응답은 원격 종단 데이터 소스가 데이터 전송 레이트를 감소시키도록 야기한다.

발명의 상세한 설명

<6> 본 발명의 실시예들은 큐 버퍼 관리를 위한 방법을 제공한다. 본 발명의 일 실시예를 위해, 인터넷 프로토콜 데이터가 데이터 소스 장치에서 생성된다. 생성된 데이터는 하나 이상의 네트워크 장치들로 전달되고 하나 이상의 네트워크 장치들에서 수신된다. 생성된 데이터의 일부분들은 생성된 데이터의 개선된 데이터 흐름을 달성하기

위하여 특정 기준들에 기초하여 선택적으로 폐기된다. 특정 기준들은 최종 폐기 이래의 시간, 최종 폐기 이래의 패킷들의 수, 패킷 프로토콜, 패킷 크기 및 이들의 조합으로 이루어진 그룹에서 선택된다. 본 발명의 일 실시예를 위해, 생성된 데이터는 흐름 제어가능 인터페이스를 통해 전달된다.

<7> 다른 이점들 및 실시예들은 상세한 설명에서 설명될 것이다.

<8> 본 발명은 본 발명의 실시예들을 도시하는데 사용된 첨부된 도면들 및 하기의 설명을 참조하여 최선으로 이해될 수 있다.

실 시 예

<27> 큐 버퍼 관리를 위한 방법 및 시스템이 제공되는데, 상기 큐 버퍼 관리를 위한 방법 및 시스템에서 데이터 흐름을 개선하기 위하여 수신된 데이터는 특정 기준들에 기초하여 선택적으로 삭제된다. 상기 시스템이 FCI를 구현하는 본 발명의 다양한 실시예들에 대해, 상기 특정 기준들은 다양한 조합들에서, 큐 깊이, 필터링된 큐 깊이, 최종 폐기된 데이터 이래의 시간, 최종 폐기된 데이터 이래의 수신된 데이터의 양 및 패킷 기반 데이터 통신을 구현하는 시스템들에 대해 패킷 크기 및 패킷 프로토콜 중 하나 이상을 포함할 수 있다. 비-흐름 제어가능 환경 내에서 구현된 본 발명의 실시예들에 대해, 특정 기준들은 다양한 조합들에서 최종 폐기된 데이터 이래의 시간, 최종 폐기된 데이터 이래의 수신된 데이터의 양, 및 패킷 기반 데이터 통신을 구현하는 시스템들에 대해 패킷 크기 및 패킷 프로토콜 중 하나 이상을 포함할 수 있다.

<28> 본 발명의 다양한 실시예들의 하기의 상세한 설명이 단지 설명을 위한 것이고 어떤 방식으로도 제한하는 것을 의도하지 않는다는 것을 당업자는 이해할 것이다. 본 발명의 다른 실시예들은 본 명세의 이익을 갖는 당업자들에게 쉽사리 스스로를 제시할 것이다. 이들 특정 상세가 본 발명의 실시예들을 실시하는데 필요하지 않을 수 있다는 것은 당업자에게 명백할 것이다. 다른 경우에, 잘 알려져 있는 요소들 및 장치들은 본 발명을 모호하게 만드는 것을 회피하기 위하여 블록도 형태로 도시된다. 하기의 실시예들의 설명에서, 실질적으로 동일한 부분들은 동일한 참조번호들로 표기된다.

<29> 명확성을 기하기 위하여, 여기에 설명된 구현예들의 특징들이 모두 도시되거나 설명되는 것은 아니다. 물론, 어떤 이러한 실제적인 구현의 개발시, 개발자의 특정 목표들을 달성하기 위하여 수많은 구현-특정 장치들이 만들어져야 한다는 것은 이해될 것이고, 이들 특정 목표들은 구현별로 그리고 개발자별로 변할 것이다. 더욱이, 이러한 개발 노력이 복잡하고 시간 소모적인 것이지만, 그럼에도 불구하고 이러한 개발 노력이 본 명세의 이익을 갖는 당업자를 위해 엔지니어링을 떠맡는 일상적인 일이 될 것이라는 것은 이해될 것이다.

<30> 본 발명의 실시예에 의하면, 구성요소들, 프로세스 단계들 및/또는 데이터 구조들은 다양한 유형의 운영 체제들, 컴퓨팅 플랫폼들, 컴퓨터 프로그램들 및/또는 범용 기계들을 사용하여 구현될 수 있다. 더욱이, 여기에 개시된 창의적인 개념들의 범위 및 정신을 벗어나지 않고 덜 범용적인 특성을 지닌 장치들이 또한 사용될 수 있다는 것을 당업자는 인지할 것이다.

<31> 본 발명의 특정 실시예들이 도시되고 설명될지라도, 여기에 개시된 창의적인 개념들을 벗어나지 않고 상기에 언급된 것보다 더 많은 변형들이 가능하다는 것은 본 명세의 이익을 갖는 당업자에게 명백할 것이다. 더욱이, 창의적인 양상들은 단일 개시된 실시예의 모든 특징들보다 적은 특징들에 존재한다. 따라서, 하기의 상세한 설명 다음에 오는 청구항들은 본 상세한 설명에 명확히 통합되는데, 각 청구항은 본 발명의 개별적인 실시예로서 단독으로 존재한다.

<32> 도 2는 본 발명의 일 실시예에 의한 데이터 통신 시스템을 도시한 것이다. 도 2에 도시된 시스템(200)은 데이터(예를 들어, 인터넷 프로토콜 데이터)를 생성하는 데이터 소스 장치(205)를 포함한다. 생성된 데이터는 예를 들어, 데이터를 수신하는, 네트워크 장치(210)로서 도시된 하나 이상의 네트워크 장치들로 전달된다. 네트워크 장치는 당 기술분야에 알려져 있는 바와 같이 유선 또는 무선 통신 수단을 통해 데이터 소스 장치에 통신가능하게 연결된다. 본 발명의 일 실시예를 위해, 네트워크 장치(210)는 예를 들어, USB, PCMCIA, 익스프레스카드, PCI 익스프레스 미니 또는 시리얼과 같은 FCI를 통해 데이터 소스 장치(205)에 통신가능하게 연결된다.

<33> 네트워크 장치(210)는 예를 들어, 기준들 RTT, 출력 데이터 레이트 및 큐 깊이로서 도시된, 특정 기준들에 기초하여 수신된 데이터의 일부분을 삭제할지(패킷을 폐기할지) 여부를 결정하는 큐 관리 기능(215)을 포함한다.

<34> 큐 관리 기능(215)은 멀티플렉서(220)에 연결되어 있고 멀티플렉서(220)를 통해 도 2에 도시된 바와 같이 데이터를 폐기 또는 삭제할지에 대한 결정이 내려진다.

<35> 도 3은 본 발명의 일 실시예에 의한 데이터 통신 시스템의 큐 버퍼 관리를 달성하기 위한 프로세스를 도시한 것이다. 본 발명의 대안적인 실시예들은 여전히 최대 네트워크 장치 출력 레이트를 유지하면서 과도한 큐잉 문제를 해결하는 큐 관리 방법들을 제공한다.

<36> 도 3에 도시된 프로세스(300)는 동작 305에서 시작하는데, 데이터는 데이터 소스 장치로부터 큐 버퍼 관리 스킴을 구현하는 네트워크 장치로 수신된다.

<37> 동작 310에서, 수신된 데이터는 특정 기준들에 기초하여 평가된다. 특정 기준들은 데이터 통신 시스템의 정적 및 동적인 매개 변수들을 포함할 수 있다. 동적인 매개 변수들은 예를 들어, 정체 임계값, 최소 시간 간격, 최소 폐기 패킷 크기 및 데이터 프로토콜을 포함할 수 있는데, 이들은 예를 들어, 출력 데이터 레이트, RTT 및 큐 깊이의 함수일 수 있다.

<38> 본 발명의 일 실시예를 위해, 특정 기준들 중 하나 이상의 기준들은 데이터 통신 이전에 또는 데이터 통신 동안 동적으로 결정될 수 있다. 본 발명의 다양한 대안적인 실시예들을 위해, 특정 기준들 중 하나 이상의 기준들은 데이터 통신 이전에 또는 데이터 통신 동안에 동적으로 결정될 수 있는 대응하는 값(예를 들어, 바이트, 초 등)을 갖는다.

<39> 동작 315에서 데이터의 일부분들은 데이터 통신 시스템의 데이터 흐름을 개선하기 위하여 평가에 기초하여 선택적으로 삭제된다.

<40> 도 4는 본 발명의 일 실시예에 의한 큐 버퍼 관리를 위한 방법의 프로세스 흐름도를 도시한 것이다.

<41> 도 4에 도시된 바와 같이, 본 발명의 일 실시예를 위하여, 큐 관리 프로세스는 입력 신호들: 추정된 채널 RTT, 추정된 출력 데이터 레이트 및 현재의 큐 깊이를 사용한다. 상기 프로세스는 큐내의 데이터의 일부분(예를 들어, 패킷)이 프로토콜의 다음 계층으로 송신되어야 하는지를 결정한다.

<42> 본 발명의 일 실시예를 위해, 큐 관리 방법은 TCP 정체 제어 메커니즘을 구동시키는데, 이것은 정체 윈도를 낮추고 따라서 정체의 조짐이 있는 경우 인-플라이트(in-flight) 데이터 또는 확인응답되지 않은 데이터의 양을 낮춘다. 인-플라이트 데이터의 양을 낮출수록, 데이터 소스는 더 느리게 데이터를 송신할 수 있다. 송신하는 TCP 스택에 정체를 나타내기 위하여 전개될 수 있는 두가지 방법들이 존재한다: 폐기된 패킷 또는 TCP 헤더에 명백한 정체 통지 플래그들의 설정. 그것의 매우 기본적인 형태에 있어서, 큐 관리자는 큐 깊이를 감시할 것이다. 큐 깊이가 어떤 임계값(즉, 본 문서에서 정체 임계값)을 초과하는 경우, 그것은 패킷을 폐기함으로써 또는 명백한 정체 통지를 트리거함으로써 TCP 정체 제어 메커니즘을 트리거한다. 큐 깊이는 큐 관리 시스템이 패킷을 폐기하기 전에 고려할 필요가 있는 많은 인자들 중 단지 하나의 인자이다.

<43> 본 발명의 일 실시예를 위하여, 큐 관리 시스템에 대한 입력은 한 세트의 실시간 제어 신호들 및 조정할 수 있는 상수들이다. 하기는 가변 시스템 데이터 레이트 및 RTT를 위한 큐 관리 시스템을 제어하는데 사용되는 기본적인 조정가능한 상수들의 목록이다.

<44> 도 4에 도시된 바와 같이, 본 발명의 일 실시예에 의한 방법은 아래의 것을 포함하는 하나 이상의 조정가능한 매개 변수들을 사용할 수 있다.

<45> 정체 임계값: 평균 큐 길이 및 현재의 큐 길이가 이 임계값을 초과하는 경우 패킷은 폐기된다.

<46> 최소 시간 간격: 최종 폐기된 패킷 이래 이 시간 양이 경과된 후에 패킷 폐기가 단지 고려된다.

<47> 최소 패킷 간격: 최종 폐기된 패킷 이래 이 많은 패킷들이 송신된 후에 패킷 폐기가 단지 고려된다.

<48> 최소 폐기 패킷 크기: 이 상수는 패킷이 폐기를 위해 고려되어야 할 최소 크기이다.

<49> 폐기를 위한 전송 프로토콜 고려사항: 이 상수는 데이터를 삭제(폐기)하기로 결정하는데 있어서 고려되는 전송 프로토콜들의 목록을 포함한다. TCP 또는 UDP와 같은 몇몇 프로토콜들이 고려될 수 있다. ICMP, RTP 등과 같은 다른 프로토콜들은 폐기를 위해 고려될 수 없다.

<50> 부록 A로서 포함된 것은 큐 버퍼 관리를 달성하고 데이터 흐름을 개선하기 위하여 정보를 삭제할 것인지 여부를 결정하는데 유용한 이들 기준들 및 다른 고려사항들을 구현하기 위한 예시적인 방법들이다.

<51> 도 2를 다시 참조하면, 데이터 소스 장치(205) 및 네트워크 장치(210)는 당업자에 의해 이해되는 바와 같은 어떤 다양한 디지털 처리 시스템(DPS)일 수 있다. 도 5는 본 발명의 일 실시예에 따라 데이터를 전달하는데 사용될 수 있는 디지털 처리 시스템의 기능 블록도를 도시한 것이다. 도 5에 도시된 처리 시스템(500)의 구성요소들

은 하나 이상의 구성요소들이 생략되거나 부가될 수 있는 예시적인 것이다. 예를 들어, 하나 이상의 메모리 장치들이 처리 시스템(500)을 위해 이용될 수 있다.

<52> 도 5를 참조하면, 처리 시스템(500)은 중앙 처리 장치(502) 및 버스(501)를 통해 메인 메모리(504), 정적 메모리(506) 및 대용량 저장 장치(507)에 연결된 신호 처리기(503)를 포함한다. 본 발명의 일 실시예에 따라, 메인 메모리(504)는 선택적인 통신 애플리케이션을 저장할 수 있는데 반하여, 대용량 저장 장치(507)는 상기에 논의된 바와 같이 다양한 디지털 콘텐츠를 저장할 수 있다. 처리 시스템(500)은 또한 버스(501)를 통해 입력/출력(I/O) 장치들(525) 및 오디오/음성 장치(526)에 연결될 수 있다. 버스(501)는 정보 및 신호들을 전달하기 위한 표준 시스템 버스이다. CPU(502) 및 신호 처리기(503)는 처리 시스템(500)을 위한 처리 유닛들이다. CPU(502) 또는 신호 처리기(503) 또는 이들 양자는 처리 시스템(500)을 위해 정보 및/또는 신호들을 처리하는데 사용될 수 있다. CPU(502)는 제어 유닛(531), 산술 논리 연산 유닛(ALU)(532) 및 다수의 레지스터들(533)을 포함하는데 이들은 정보 및 신호들을 처리하는데 사용된다. 신호 처리기(503)는 또한 CPU(502)와 같은 유사한 구성요소들을 포함할 수 있다.

<53> 메인 메모리(504)는 예를 들어, 램(RAM) 또는 정보 또는 명령들(프로그램 코드)을 저장하기 위한 어떤 다른 동적 저장 장치일 수 있는데, 이들은 CPU(502) 또는 신호 처리기(503)에 의해 사용된다. 메인 메모리(504)는 CPU(502) 또는 신호 처리기(503)에 의한 명령들의 실행 중 임시 변수들이나 다른 중간 정보를 저장할 수 있다. 정적 메모리(506)는 예를 들어, 정보 또는 명령들을 저장하기 위한, 롬(ROM) 및/또는 다른 정적 저장 장치들일 수 있다. 대용량 저장 장치(507)는 예를 들어, 처리 시스템(500)을 위한 정보 또는 명령들을 저장하기 위한, 하드 디스크 드라이브 또는 광학 디스크 드라이브일 수 있다.

<54> **일반적인 문제들**

<55> 본 발명의 실시예들은 데이터의 통신 시스템에서 큐 관리를 달성하기 위한 방법들 및 시스템들을 제공한다. 본 발명의 일 실시예를 위하여, 인터넷 프로토콜 데이터는 데이터 소스 장치에서 생성된다. 생성된 데이터는 하나 이상의 네트워크 장치들로 전달되고 상기 하나 이상의 네트워크 장치들에서 수신된다. 생성된 데이터의 일부분들은 생성된 데이터의 개선된 데이터 흐름을 달성하기 위하여 특정 기준들에 기초하여 선택적으로 삭제된다. 본 발명의 일 실시예를 위하여, 생성된 데이터는 흐름 제어가능 인터페이스를 통해 전달된다.

<56> 본 발명의 대안적인 실시예들은 여전히 최대 네트워크 장치 출력 레이트를 유지하면서 과도한 큐잉 문제를 해결하는 큐 관리 방법들을 제공한다.

<57> 본 발명의 실시예들은 데이터 통신, 버퍼링 및 처리와 같은 다양한 동작들을 포함한다. 다양한 실시예들을 위하여, 설명된 하나 이상의 동작들이 부가되거나 삭제될 수 있다. 예를 들어, 추정된 RTT 및 추정된 출력 데이터 레이트를 획득하기 위하여 네트워크 장치가 사용할 수 있는 몇몇 대안적인 방법들이 존재한다. RTT는 송신된 데이터를 확인응답하는데 걸리는 시간을 검사함으로써 측정될 수 있다. 데이터가 네트워크 장치로 입력될 때 데이터가 암호화되는 경우, 이러한 방식으로 계산된 RTT는 가능하지 않을 것이다. 대안적으로, 네트워크 장치가 알려져 있는 정적 IP 주소 또는 큐에서 발견된 IP 주소들에 대한 낮은 레이트의 핑(ping)을 생성할 수 있는 경우, RTT는 추정될 수 있다. 또한, 고정된 최악의 경우의 RTT 또는 일반화된 RTT 추정은 물리 계층의 채널 상태에 기초하여 행해질 수 있다. 출력 데이터 레이트는 데이터가 큐를 빠져나가고 있는 레이트를 측정함으로써 계산될 수 있거나 그리고/또는 물리 계층 채널 액세스 허가에 의해 결정될 수 있다.

<58> 본 발명의 동작들은 하드웨어 구성요소들에 의해 수행될 수 있거나 범용 또는 특정 목적 프로세서 또는 명령들로 프로그램된 로직 회로들이 동작들을 수행하도록 야기하는데 사용될 수 있는, 기계-실행가능한 명령들로 구현될 수 있다. 대안적으로, 동작들은 하드웨어 및 소프트웨어의 조합에 의해 수행될 수 있다. 본 발명의 실시예들은 본 발명에 따라 프로세스를 수행하기 위하여 컴퓨터(또는 다른 전자 장치들)를 프로그램하는데 사용될 수 있는, 저장된 명령들을 구비하는 기계-독출가능 매체를 포함할 수 있는 컴퓨터 프로그램 생성물로서 제공될 수 있다. 상기 기계-독출가능 매체는 광학 디스크, 시디-롬 및 자기-광학 디스크, 롬, 램, EPROM, EEPROM, 자기 또는 광학 카드, 플래시 메모리, 또는 전자 명령들을 저장하는데 적합한 다른 유형의 매체/기계-독출가능 매체를 포함할 수 있지만, 이에 한정되는지는 않는다. 더욱이, 본 발명은 컴퓨터 프로그램 생성물로서 또한 다운로드될 수 있는데, 이 경우 통신 셀(예를 들어, 모뎀 또는 네트워크 접속)을 통해 반송파 또는 다른 전파 매체로 구현되는 데이터 신호들에 의해 원격 컴퓨터로부터 요청하는 컴퓨터로 프로그램이 전달될 수 있다.

<59> 더욱이, 특정 맥락에서 다양한 실시예들에 대해 설명되었을지라도, 본 발명의 실시예들은 다수의 데이터 표준들을 채택하는 다양한 단일 채널 또는 다중 채널 데이터 전송 시스템들에 적용가능하다.

<60> 본 발명이 몇몇 실시예들에 관하여 설명되었을지라도, 본 발명이 설명된 실시예들에 한정되지 않는다는 것과, 첨부된 청구항들의 정신 및 범위 내에서 변형 및 변경되어 실시될 수 있다는 것을 당업자는 인지할 것이다. 따라서 상기한 설명은 제한하는 것 대신에 설명하는 것으로서 간주되어야 한다.

<61> **부록 A**

<62> 조정가능한 매개 변수들

<63> 본 발명의 일 실시예를 위하여, 큐 관리 시스템에 대한 입력은 한 세트의 실시간 제어 신호들 및 조정가능한 상수들이다. 하기는 가변 시스템 데이터 레이트 및 RTT에 대한 큐 관리 시스템을 제어하는데 사용되는 기본적인 조정가능한 상수들의 목록이다.

<64> ● 정체 임계값:

<65> 평균 큐 길이 및 현재의 큐 길이가 이 임계값을 초과하는 경우 패킷은 폐기된다.

<66> ● 최소 시간 간격:

<67> 최종 폐기된 패킷 이래 이 시간 양이 경과된 후에 패킷 폐기가 단지 고려된다.

<68> ● 최소 패킷 간격:

<69> 최종 폐기된 패킷 이래 이 많은 패킷들이 송신된 후에 패킷 폐기가 단지 고려된다.

<70> ● 최소 폐기 패킷 크기:

<71> 이 상수는 패킷이 폐기를 위해 고려되어야 할 최소 크기이다.

<72> ● 폐기를 위한 전송 프로토콜 고려사항:

<73> 이 상수는 TCP 또는 UDP와 같이 폐기하는데 고려되는 전송 프로토콜들의 목록을 포함한다. ICMP, RTP 등과 같은 다른 프로토콜들은 폐기를 위해 고려될 수 없다.

<74> 피크 부하 고려 사항

<75> 선택적으로, 큐 관리 시스템은 피크 입력 버스트에 기인한 패킷들의 폐기를 방지하기 위한 메커니즘을 포함해야 한다. IP 데이터 트래픽의 버스티한 특성에 기인하여, 트래픽의 큰 버스트가 큐로 송신되는 것은 흔한 일이다. 본 발명은 트래픽의 어떤 버스트에 기인하여 패킷을 폐기하는 것을 회피해야 한다. 정체 임계값이 큐 길이의 순간적인 측정치 및 필터링된 (또는 평균화되거나 평탄화된) 버전의 큐 길이 양자와 비교되는 메커니즘이 구현되도록 권고된다.

<76> 피크 부하시 패킷들을 폐기하는 것을 회피하는데 사용될 수 있는 다른 메커니즘은 패킷을 폐기하기 전에 일정 시간동안 CngTh를 초과하는 큐 길이를 필요로 한다. 이 메커니즘에 대한 단점은 성능을 저하시킬 과도한 큐잉에 작용시 항상 지연이 존재한다는 것이다.

<77> 제어 패킷 고려 사항

<78> 확인응답 또는 다른 제어 패킷들의 폐기는 그들이 TCP 스택에서 정체 제어 알고리즘을 트리거하지 않을 것이기 때문에 회피되어야 한다. 패킷의 유형을 결정하기 위한 패킷의 파싱(parsing)은 제어 패킷들의 폐기를 제거하는데 사용될 수 있다. 패킷들이 암호화되는 경우 그들을 파싱하는 것은 가능하지 않다. 이 경우, 패킷의 크기는 패킷이 확인응답 또는 다른 더 작은 제어 유형 패킷(ICMP는 또한 작은 경향이 있다)인지를 결정하는데 사용될 수 있다. 이상적으로, 최대 세그먼트 크기 부근의 패킷들만이 폐기를 위해 고려되어야 한다. 제시된 큐 관리 시스템에서 상수 최소 폐기 패킷 크기는 패킷이 폐기를 위해 고려될 필요가 있는 최소 크기를 설정하는데 사용된다.

<79> 이 문서에 표시된 모든 시뮬레이션 결과들은 최소 폐기 패킷 크기=1200 바이트로 설정한다.

<80> 헤드(Head) 대 테일(Tail) 폐기 고려 사항

<81> 트리거에 대한 시기적절한 TCP 응답에 도움을 주기 위하여, 폐기될 패킷은 큐의 앞부분 또는 헤드로부터 폐기되어야 한다(네트워크 장치로부터 막 송신될 패킷). 큐의 앞부분으로부터 패킷을 폐기하는 것은 또한 정체 메커니즘을 개시하기 위한 선택적인 ACK(SACK)(TcpMaxDupAcks=2)를 트리거하기 위하여 그것의 뒤에 충분한 데이터가 있다는 것을 보증하는데 도움을 주고 그렇지 않으면 TCP 스택은 패킷에 대해 시간초과시켜야 할 것인데, 이것은

출력 성능 저하를 야기할 것이다. 이것이 주어지는 경우, 큐 및 큐 관리는 가능한 한 MAC 또는 물리 계층들에 근접하는 것이 최선이다. 큐의 헤드로부터 폐기하는 것이 최적일지라도, 큐내의 어떤 곳을 폐기하는 것도 작용할 것이다.

<82> 이 문서에 표시된 모든 시뮬레이션 결과들은 헤드 폐기 메커니즘을 사용한다.

<83> 정적 큐 관리 방법들

<84> 이 섹션은 정체 임계값, 최소 시간 간격 및 최소 패킷 간격 제어 신호들을 위해 정적 신호들을 사용하는 큐 관리 시스템들에 대해 논의한다. 따라서, 도 4a에 도시된 기능들 CongThFunc, MinTimeIntFunc 및 MinPacketIntFunc는 사용되지 않는다. 동적 큐 관리 시스템들은 나중에 논의될 것이다.

<85> 정체 임계값 신호

<86> 정체 임계값은 패킷을 폐기할 때를 결정하는데 사용되는 많은 제어 신호들 중 하나이다. 다른 패킷 폐기 조건이 통과된다고 가정하면, 상기한 흐름도는 순간적인 큐 길이 및 필터링된 (또는 평탄화된) 큐 길이가 정체 임계값을 초과하는 경우 패킷이 폐기된다는 것을 보여준다.

<87> 어떤 점에서 정체 임계값은 TCP 정체 윈도(cwin)를 설정한다. TCP 정체 윈도가 커질 수 있는 최대값은 데이터를 수신하고 있는 TCP 스택에 의해 설정된 수신 윈도(rwin)에 의해 설정된다(RFC 2581 참조). TCP 처리량을 최대화하고 큐잉을 최소화하기 위하여, rwin은 지연 대역폭 프로덕트(RTT*출력 데이터 레이트)로 설정되어야 한다. 유사하게, 큐 관리 시스템에서, 정체 임계값은 대략 지연 대역폭 프로덕트(DBP)에서 최적으로 설정되어야 한다. 이 임계값 설정으로, TCP 정체 윈도는 2 곱하기 DBP부터 1 곱하기 DBP까지 변할 것이다. DBP에 대해 설정된 정체 임계값은 최대 출력 데이터 레이트가 획득되고 최소 큐잉이 발생하는 최적 포인트이다.

<88> 도 1a는 네트워크 장치 출력 레이트 설정 16kB/초이고 채널 RTT가 0.4초인 큐 관리 시스템의 시뮬레이션 결과를 보여준다. 정체 임계값은 DBP(16*0.4=6.4kB)에서 설정된다. 큐 관리 시스템에 대한 입력은 무한 데이터 소스를 갖는 그래서 TCP 스택의 범위내의 최대 허용가능한 레이트로 항상 송신하려고 하는 단일 TCP 데이터 스트림이다.

<89> 시뮬레이션 매개 변수들:

<90> 정체 제어 알고리즘 매개 변수들

<91> 6400 MAX_CngTh 최대 정체 임계값

<92> 6400 MIN_CngTh 최소 정체 임계값

<93> 1200 DropSizeTh - 폐기 크기 임계값(바이트 단위) 폐기될 패킷의

<94> 최소 크기

<95> 범위 300-1500 바이트

<96> 0 MinPacketInterval - 폐기들 간의 최소한의 패킷들의 수

<97> 1.3 MinDropInterval (초) 폐기된 패킷들 간의 최소한의 시간의 양

<98> 물리 계층

<99> 16 출력 데이터 레이트(KB/S)

<100> 도 1a로부터, TCP 느린(Slow) 시작 메커니즘이 발생하고 그다음 TCP 정체 메커니즘이 떠맡는 것을 알 수 있다. TCP 정체 윈도는 이전에 설명된 바와 같이 변하는 것으로 도시된다. 출력 데이터 레이트는 최대 물리 출력 데이터 레이트와 동일하다. 큐 크기는 ~6400부터 0까지 변하는데, 이것은 데이터 언더런의 발생을 야기하지 않고 가능한 최저 큐 크기이다.

<101> 정체 임계값 설정 고려 사항:

<102> 정체 임계값이 너무 크게 (DBP보다 크게) 설정되면 초과 큐잉이 결과로서 발생할 것이다. 정체 임계값이 너무 작게 (DBP 미만) 설정되는 경우 큐의 출력 데이터 레이트는 저하될 것이다. 도 2a의 시뮬레이션 결과는 이들 영향을 보여준다. 동일한 큐 관리 시스템이 도 1a에 있지만 이번에 물리 출력 데이터 레이트는 다른 DBP들을 생성하기 위하여 시간적으로 변한다.

<103> 시뮬레이션 매개 변수들:

<104> 정체 제어 알고리즘 매개 변수들

<105> 6400 CngTh 최대 정체 임계값

<106> 1200 DropSizeTh - 폐기 크기 임계값(바이트 단위) 폐기될 패킷의

<107> 최소 크기

<108> 범위 300-1500 바이트

<109> 0 MinPacketInterval - 폐기들 간의 최소한의 패킷들의 수

<110> 1.3 MinDropInterval (초) 폐기된 패킷들 간의 최소한의 시간의 양

<111> 출력 데이터 레이트

<112> 16 시작 레이트(바이트/초)

<113> 25 출력 레이트를 변경하기 위한 시간(초)

<114> 8 변경 레이트(바이트/초)

<115> 45 출력 레이트를 변경하기 위한 시간(초)

<116> 32 변경 레이트(바이트/초)

<117> 출력 데이터 레이트가 8kB/초와 동일한 기간동안, 출력 데이터 레이트는 8kB/초와 동일하지만 필요한 것보다 더 많은 큐잉이 존재한다. 출력 데이터 레이트가 32kB/초와 동일한 기간동안, 큐가 비어있을 때 출력 데이터 레이트는 강하되고 따라서 32 kB/초 미만이다.

<118> 최소 시간 간격

<119> 최소 시간 간격은 패킷을 폐기할 시간을 결정하는데 사용되는 많은 제어 신호들 중 다른 하나의 제어 신호이다. 모든 다른 패킷 폐기 조건이 통과된다고 가정하면, 상기한 흐름도는 최종 폐기 이래의 시간이 최소 시간 간격보다 더 큰 경우에만 폐기된다는 것을 보여준다. 최소 시간 간격은 두가지 목적을 가질 수 있다. 첫번째 목적은 다른 패킷이 폐기되기 전에 폐기된 패킷에 대해 반응하기 위하여 TCP 스택에 대해 어떤 시간을 제공하는 것이다. 두번째 목적은 패킷을 폐기하기 위하여 적합한 시간을 설정하기 위한 정체 임계값 목적과 유사하다.

<120> 다수의 패킷들이 폐기되는 것을 방지하는데 최소 시간 간격이 단지 사용되는 경우, 그것은 TCP가 패킷 폐기에 반응하는데 걸리는 시간으로 설정되어야 한다. 이것은 2 곱하기 RTT 플러스 큐를 가로지르기 위한 재시도에 걸리는 시간의 길이보다 약간 더 큰 값으로 산정된다(이것의 상세는 하기의 섹션들에서 얻어진다). 도 1a에 도시된 우리의 시뮬레이션으로부터의 매개 변수들을 사용하여, 시간 최소 시간 간격은 다음과 같이 계산된다:

<121> 2*0.4(RTT)+6400(폐기가 발생하는 경우 Q 크기)/16000(출력 레이트)+0.1=1.3초

<122> 이 신호의 사용은 중요한데 그렇지 않으면 큐 관리 시스템은 하나보다 많은 패킷들을 폐기할 것이다. 도 3a는 고정된 정체 임계값을 갖는 동일한 큐 관리 시스템을 도시한 것이지만, 이번에는 최소 시간 간격이 0으로 설정된다.

<123> 시뮬레이션 매개 변수들

<124> 정체 제어 알고리즘 매개 변수들

<125> 6400 CngTh 최대 정체 임계값

<126> 1200 DropSizeTh - 폐기할 패킷의 최소 크기. 범위 300-1500 바이트

<127> 0 MinPacketInterval - 폐기들 간의 최소한의 패킷들의 수

<128> 0 MinDropInterval (초) 폐기된 패킷들 간의 최소한의 시간의 양

<129> 물리 계층

<130> 16 출력 데이터 레이트(KB/초)

<131> 이 시뮬레이션의 출력은 다수의 패킷들이 폐기되고 따라서 TCP 스택은 정체 윈도를 과감하게 낮춤으로써 반응한다. 이것은 데이터 언더런을 야기하고 출력 데이터 레이트를 저하시킨다.

<132> 두번째 목적은 큐 레벨을 제어하기 위하여 주로 최소 시간 간격에 의존하는 것이다. 이 경우, 정체 임계값은 대부분 어떤 임의의 양으로 설정될 수 있다. 최적의 최소 시간 간격의 계산은 최적의 정체 임계값 계산보다 약간 더 복잡해진다.

<133> 하기의 변수들은 최적의 최소 시간 간격을 결정하는데 사용되는 공식을 유도하는데 사용될 것이다:

<134> MaxQ = 큐가 DBP=RTT*DR로 증대되어야 하는 최적의 최대값

<135> DR = 출력 데이터 DR

<136> Cwnd = 정체 윈도

<137> RTT = 라운드 트립 시간

<138> IPSegSize = IP 패킷의 크기

<139> TCPSegSize = TCP 패킷의 크기

<140> AcksPerSec = 초당 수신된 확인응답들

<141> TimeToReduceCwnd = TCP 스택이 폐기된 패킷으로부터 Cwnd의 값의 1/2까지 걸리는 시간

<142> TimeToGrowQueue = 큐가 MaxQ로부터 증가하는데 걸리는 시간

<143> TimeToGrowCWnd = TCP 스택이 CWnd를 MaxQ로부터 2*MaxQ까지 증가시키는데 걸리는 시간

<144> 최적으로 상기 최소 시간 간격은 큐가 MaxQ까지 채우는데 걸리는 시간 플러스 TCP 스택이 폐기된 패킷에 대해 반응하는데 걸리는 시간과 동일하다.

<145> 최소 시간 간격 = TimeToGrowQueue + TimeReduceCwnd

<146> 큐를 MaxQ까지 증가시키기 위한 시간은 TCP 스택이 정체 윈도를 MaxQ로부터 2*MaxQ까지 증가시키는데 걸리는 시간과 동일하다:

<147> 최소 시간 간격 = TimeToGrowCWnd + TimeReduceCwnd

<148> TimeReduceCwnd의 계산:

<149> TCP 스택이 폐기된 패킷으로부터 Cwnd의 값의 1/2까지 걸리는 시간은 확인응답이 재시도로부터 수신될 때까지 걸리는 시간이다. 이 시간은 다음과 같이 분해될 수 있다;

<150> SACK가 수신될 때까지 걸리는 시간 = RTT

<151> 재전송된 패킷이 송신될 때까지 걸리는 시간 = 큐잉 시간 = MaxQ/DR

<152> 재전송된 패킷에 대한 확인응답이 수신될 때까지 걸리는 시간 = RTT

<153> 그래서:

<154> TimeReduceCwnd = RTT+MaxQ/DR+RTT = 2*RTT+MaxQ/DR

<155> MaxQ=DR*RTT을 대입하면,

<156> TimeReduceCwnd = 2*RTT+DR*RTT/DR = 3*RTT

<157> TimeToGrowCWnd의 계산:

<158> Cwnd가 MaxQ 크기로부터 2*MaxQ 크기까지 증가하는데 걸리는 시간의 양은 다음과 같다:

<159> TimeToGrowCWnd = MaxQ/CwndGrowthRate

<160> 정체 윈도가 증가하는 레이트는 확인응답이 수신될 때마다 TCPSegSize^2/Cwnd 바이트와 동일하다.

<161> CwndGrowthRate = (TCP SegSize^2/Cwnd)*AcksPerSec

<162> AcksPerSec = DR/IPSegSize

<163> 상기 수학식은 Cwnd가 상수가 아니라는 사실에 의해 복잡해진다. 이 수학식을 단순화하기 위하여 우리는 1.7*MaxQ와 동일한 평균 CWnd를 사용할 것이다. Cwnd에 대해 이것을 대입하면 다음과 같다:

<164> CwndGrowthRate = TCPSegSize^2/(1.7*MaxQ)*DR/IPSegSize

<165> 단순화하면 다음과 같다:

$$\text{CwndGrowthRate} = \frac{\text{TCPSegSize}^2 * DR}{(1.7*\text{MaxQ}) * \text{IPSegSize}}$$

<166>

<167> 최소 시간 간격의 계산:

<168> CwndGrowthRate 및 TimeReduceCwnd에 대입하면 다음과 같다:

$$\text{Min Time Interval} = \text{MaxQ} * \frac{1.7*\text{MaxQ} * \text{IPSegSize}}{\text{TCP SegSize}^2 * DR} + 3 * RTT$$

<169>

<170> MaxQ = DR*RTT에 대입하여 단순화하면 다음과 같다:

$$\text{Min Time Interval} = \frac{1.7*DR*RTT^2 * \text{IPSegSize}}{\text{TCPSegSize}^2} + 3 * RTT$$

<171>

<172> 다음 값들을 사용하면:

<173> DR = 16000 바이트/초

<174> RTT = 0.4초

<175> IPSegSize = 1500

<176> TCPSegSize = 1450

<177> 최소 시간 간격 = 1.7*16000*0.4y2*1500/(1450y2)+3*0.4=4.3초

<178> 큐 관리 시스템이 메인 폐기 기준들로서 고정된 신호 최소 시간 간격을 사용하여 동작하는 방법을 보여주기 위하여, 패킷들 간의 최소 시간을 4.3초로 설정하고 도 4a의 결과로서 생성되는 시뮬레이션 출력을 가지고 시뮬레이션이 실시되었다.

<179> 시뮬레이션 매개 변수들

<180> 정체 제어 알고리즘 매개 변수들

<181> 3000 CngTh 최대 정체 임계값

<182> 4.3 MinDropInterval (초) 폐기된 패킷들 간의 최소한의 시간의 양

<183> 물리 계층

<184> 16 출력 데이터 레이트(KB/초)

<185> 이 시뮬레이션은 정체 임계값이 단지 3000 바이트로 설정될지라도, 폐기가 4.3초로 설정된 최소 시간 간격에 의해 제한되기 때문에 (여전히 모든 이용가능한 출력 데이터 레이트를 사용하면서 최저 큐잉) 큐 관리가 최적이라는 것을 보여준다.

<186> 정적 정체 임계값을 사용하는 것과 유사하게, 고정된 최소 시간 간격을 사용하는 것은 과도한 큐잉을 발생시킬

것이거나 최적 값으로 설정되지 않는 경우 전체 이용가능한 출력 레이트를 이용하지 않을 것이다. 이러한 결과를 보여주기 위하여 시뮬레이션이 실행되었고 출력 레이트는 16, 8 및 32 kB/초에서 변하였다. 도 5a는 시뮬레이션의 결과를 보여준다.

<187> 시뮬레이션 매개 변수들:

<188> 정체 제어 알고리즘 매개 변수들

<189> 3000 CngTh 최대 정체 임계값

<190> 0 MinPacketInterval - 폐기들 간의 최소한의 패킷들의 수

<191> 4.3 MinDropInterval (초) 폐기된 패킷들 간의 최소한의 시간의 양

<192> 출력 데이터 레이트

<193> 16 시작 레이트(KB/초)

<194> 25 출력 레이트를 변경하기 위한 시간(초)

<195> 8 변경 레이트(KB/초)

<196> 45 출력 레이트를 변경하기 위한 시간(초)

<197> 32 변경 레이트(KB/초)

<198> 이들 결과들은 정적 정체 임계값이 사용되는 큐 관리 시스템과 유사하다. 물리 출력 레이트가 8kB/초와 동일한 기간 동안, 큐 평균 출력 데이터 레이트는 8kB/초이지만, 필요한 것보다 더 많은 큐잉이 존재한다. 물리 출력 레이트가 32kB/초와 동일한 기간동안, 큐 평균 출력 레이트는 큐가 비어 있을 때 저하되어서 32kB/초 미만이다.

<199> 최소 패킷 간격

<200> 최소 패킷 간격은 패킷을 폐기할 때를 결정하는데 사용되는 많은 제어 신호들 중 다른 하나의 제어 신호이다. 모든 다른 패킷 폐기 조건이 통과된다고 가정하면, 상기한 흐름도는 최종 폐기 이래의 패킷들의 수가 최소 패킷 간격보다 더 큰 경우에만 폐기된다. 이 신호의 사용은 최소 시간 간격의 사용과 동일하다. 그것은 다른 것을 폐기하기 전에 폐기된 패킷에 TCP 스택이 반응하도록 기다리는데 사용될 수 있거나 상기 시스템의 주된 제어로서 사용될 수 있다. 전형적으로 큐 관리 시스템은 이들 메커니즘들 양자를 지원할 필요는 없을 것이지만 구현의 용이함에 기반하여 다른 것을 능가하는 하나를 선택할 것이다. 최적의 최소 패킷 간격은 다음과 같이 계산될 수 있다:

<201> 최소 패킷 간격 = (패킷/초)*최소 시간 간격

<202> 최소 패킷 간격 = (DR/IPPacketSize)*최소 시간 간격

<203> 도 1a에 도시된 시뮬레이션 결과들로부터의 시뮬레이션 매개 변수들을 사용하면, 결과로서 생성되는 최소 패킷 간격은 다음과 같다:

<204> 최소 패킷 간격 = (16000/1500)*4.3=46 패킷

<205> 46과 동일한 고정된 값을 사용하여, 도 6a에 도시된 결과들과 함께 시뮬레이션이 실행되었다.

<206> 시뮬레이션 매개 변수들:

<207> 정체 제어 알고리즘 매개 변수들

<208> 3000 CngTh 최대 정체 임계값

<209> 46 MinPacketInterval - 폐기들 간의 최소한의 패킷들의 수

<210> 0 MinDropInterval (초) 폐기된 패킷들 간의 최소한의 시간의 양

<211> 물리 계층

<212> 16 출력 데이터 레이트(KB/초)

<213> 상기 시뮬레이션 결과는 정체 임계값이 단지 3000 바이트로 설정될지라도, 폐기는 46 패킷으로 설정된 최소 패

킷 간격에 의해 제한되기 때문에 큐 관리가 최적이라는 것을 보여준다(여전히 모든 이용가능한 출력 데이터 레이트를 사용하면서 최저 큐잉).

<214> 다중 TCP 스트림 고려사항

<215> 이 문서에 도시된 모든 시뮬레이션이 단지 하나의 TCP 스트림을 위한 것일지라도, 모든 큐 관리 알고리즘들은 하나보다 많은 병렬 TCP 스트림을 갖는 입력되는 데이터 스트림을 적절하게 지원하도록 조정될 수 있다.

<216> 정체 임계값 신호가 주된 폐기 기준들로서 사용되는 경우, 알고리즘은 입력으로서 다중 TCP 스트림과 합리적으로 잘 동작할 것이다. 상기 시스템이 가장 많은 데이터를 송신하고 있을 것이고 큐내에 가장 많은 데이터를 가지고 있을 것이기 때문에 상기 시스템이 가장 높은 cwnd를 지닌 스트림으로부터 확률적으로 패킷을 폐기할 것이기 때문에 이것은 사실이다. 상기 알고리즘이 불행하게도 낮은 레이트 TCP 스트림으로부터 패킷을 폐기하는 경우 어떤 단기적인 큐 깊이 피킹이 존재할 수 있을지라도, 장기적으로 그것은 심지어 없어질 것이다.

<217> 최소 시간 간격 또는 최소 패킷 간격이 주된 폐기 기준들로서 사용되는 경우, 상기 알고리즘은 조정될 필요가 있다. 최적으로 크기 조정된 큐(DR*RTT)를 유지하기 위하여, 폐기의 레이트는 활성적으로 송신하는 TCP 스트림들에 비례할 필요가 있다. 따라서, 최소 시간 간격 또는 최소 패킷 간격은 활성적으로 송신하는 TCP 스트림들의 수에 의해 조정될 필요가 있다. 활성 TCP 스트림들의 수는 큐내의 패킷들의 IP 및 TCP 헤더들을 검사하여 쉽게 계산될 수 있다. 조정된 최소 시간 간격 또는 최소 패킷 간격을 가지고, 상기 알고리즘은, 알고리즘이 불행하게도 낮은 레이트 TCP 스트림으로부터 패킷을 폐기하는 경우 어떤 단기간 큐 깊이 피킹이 발생할 수 있는, 정체 임계값 신호에 기초하는 알고리즘과 유사하게 수행할 것이다.

<218> **프로토콜 종속 폐기**

<219> 큐 관리 시스템은 폐기를 위한 어떤 유형의 상위 계층 프로토콜만을 고려하는 것이 바람직할 수 있다. SIP, RTSP, RSPP 또는 RTCP와 같은 상위 계층 제어 프로토콜들은 그들이 트래픽에서 상당한 슬로우 다운을 초래하지 않을 것이고 최종 사용자 경험의 상당한 저하를 야기할 수 있기 때문에 폐기되지 않는 것이 선호될 후보들인 것 같다. TCP가 정체 제어 알고리즘과 함께 하는 것처럼 UDP 트래픽이 응답할 것 같지 않기 때문에, 모든 UDP 트래픽은 또한 이 "폐기하지 않는 목록"에 또한 있을 것이다. UDP 패킷들을 제외시키는 문제점은 UDP 레이트가 출력 데이터 레이트보다 더 클 경우 발생한다. 이 경우, UDP 패킷들은 폐기될 필요가 있다. 이 경우를 적합하게 처리하기 위하여 제2 세트의 임계값들이 상기 알고리즘에 부가될 수 있는데(UDP 정체 임계값) 이 임계값 이상에서 UDP 패킷들은 TCP 패킷들보다 폐기하도록 더 목표가 설정될 것이다.

<220> (VPN) 암호화 이전의 처리

<221> 이 문서에서 제시된 개념들 중 몇몇 개념들은 크기, 프로토콜, 포트 번호 등과 같은 어떤 주요 속성을 결정하기 위하여 큐 관리자가 패킷을 파싱하거나 고찰할 것을 필요로 한다. 최선의 성능을 달성하기 위하여, 큐 관리자는 가능한 한 물리 계층에 근접하여 존재할 필요가 있다("헤드 대 트레임 폐기" 섹션을 참조하라). 큐 관리자를 이 위치에 배치하는 것으로 인하여 생기는 문제점은 VPN 또는 다른 암호화가 사용되는 경우, 그것이 큐 관리자에 도달할 때 모든 정보가 암호화되어서 이들 주요 속성들이 추출될 수 없다는 것이다.

<222> 이 문제에 대한 해법은 VPN 이전에 하지만 네트워크 계층 이후에 존재하는 조정 실체를 부가하는 것일 것이다. 마이크로소프트 비스타 운영 체제에서, 이들 유형의 드라이버는 필터 드라이버들로서 지칭된다. 그래서 필터 드라이버는 큐 관리자가 폐기를 위해 고려해야 하는 패킷들을 마킹할 것이다. 패킷들을 마킹하는 방법은 어떤 대역 외 시그널링을 통할 수 있거나 VPN 또는 다른 암호화가 암호화하지 않는 패킷의 필드들 중 하나를 변경함으로써 행해질 수 있다. 이 메커니즘은 본 문서의 범위를 벗어난다.

<223> 동적 큐 관리 시스템들

<224> 이제까지 본 문서 및 제시된 시뮬레이션 결과는 정적 방식으로 제어 신호들의 사용만을 고려하였다. 정체 임계값과 같은 중요한 큐 관리 제어 신호들을 계산하기 위한 런타임 기능들의 사용은 큐 관리 시스템이 물리 계층에서의 변경 및 입력 트래픽 스트림에 대한 변경에 적응하도록 도와준다. 물리 계층 변경은 데이터 레이트 및/또는 RTT 변경을 포함할 수 있다. 입력 트래픽 스트림 변경은 입력 트래픽 레이트 및/또는 프로토콜 믹스(TCP 대 UDP 또는 다른 것)를 포함할 수 있다. 제어 신호들, 정체 임계값, 최소 시간 간격 및 최소 패킷 간격을 동적으로 계산하는데 사용될 수 있는 3가지 기능들(CongThFunc, MinTimeFunc 및 MinPacketIntFunc)이 열거된다. 이들 기능들의 사용은 선택적이다. 단순화를 위해 또는 시스템에서 작은 변경이 예기되는 경우, 제어 신호들은 정적인 것이 되도록 선택될 수 있다. 큐 관리 시스템이 적응하도록 예기되는 경우, 상기 열거된 동적 제어 신호 기

능들 중 하나 이상 또는 모두가 사용되어야 한다. 큐 관리 방법은 이들 세개의 기능들을 모두 사용할 수 있지만 다른 신호들이 고정될 경우 이들 중 몇몇을 단지 사용할 것 같다. 하기 섹션은 이들 기능들을 구현하는 것에 대한 몇몇 일반적인 가이드라인들을 설명한다.

<225> CongThFunc 설명

<226> 이전에 언급된 바와 같이, TCP 처리량을 최대화하고 큐잉을 최소화하기 위하여, 정체 임계값은 항상 DBP로 설정되어야 한다. 많은 시스템들에서, 출력 데이터 레이트 및 더 작게 연장된 RTT는 상당히 변할 수 있어서 RTT의 추정치 및 출력 데이터 레이트에 기반하여 정체 임계값의 동적 계산을 사용하는 것이 유리하다.

<227> 채널 허가들 또는 신호 조건들과 같은 어떤 물리 계층 정보를 사용하거나 큐 외부의 데이터의 흐름을 측정하는 것과 같이 출력 데이터 레이트를 추정하는데 사용될 수 있는 몇몇 메커니즘들이 존재한다. 출력 데이터 레이트의 추정은 또한 단순히 물리적으로 접속된 기술에 기초할 수 있다(GPRS, EDGE 또는 WCDMA, 1x, Ev/DO rev0 또는 revA). 어떤 방법이 사용되든지, 출력 데이터 레이트의 추정치는, 물리 계층에서의 빠른 변경이 정체 임계값에 대한 극적인 변경을 야기하지 않도록 평탄화되어야 한다.

<228> 채널의 RTT를 추정하는데 사용될 수 있는 몇몇 메커니즘들이 또한 존재한다. TCP 데이터가 송신되는 경우, 원격 종단 수신기가 송신된 데이터를 확인응답하는데 걸리는 시간으로서 RTT를 계산하는 것은 매우 용이하다. VPN이 사용되는 경우, 이 측정은 암호화 계층 위에서 행해질 필요가 있을 것이다. RTT를 추정하기 위한 다른 방법은 알려져 있는 서버로 핑(Pings)과 같은 작은 패킷들을 송신하는 것이다. 이 방법의 단점은 그것이 어떤 오버헤드를 생성하고 패킷들이 송신되고 있는 동일한 위치에 대해 RTT를 측정하지 않는다는 것이다.

<229> 하기의 시뮬레이션 결과는 정체 임계값이 DBP=RTT*출력 데이터 레이트로 동적으로 설정되는 큐 관리 시스템에 대한 것이다. 모든 다른 제어 신호들은 정적이다. 출력 데이터 레이트는 데이터가 큐를 빠져 나가는 레이트에 의해 추정된다. 그다음 출력 데이터 레이트는 단극 IIR 이동 평균 필터를 사용하여 필터링된다. 이 시뮬레이션에 대해, RTT는 추정되지 않고 어떤 값(EstRTT)으로 고정된다. 이 시뮬레이션에 대해, 실제 RTT는 0.4초이지만 EstRTT는 0.4375초로 설정된다. 실제 물리 데이터 레이트는 16 kB/초 내지 8 kB/초 또는 32 kB/초까지 변한다. 큐에 대한 입력 데이터는 단일 TCP 스트림이다. 도 7a는 시뮬레이션의 결과를 보여준다:

<230> 시뮬레이션 매개 변수들:

<231> 정체 제어 알고리즘 매개 변수들

<232> 3000 Start_CngTh 시작 정체 임계값

<233> 2 MinDropInterval (초) 폐기된 패킷들 간의 최소한의 시간의 양

<234> 0.4375 EstRTT - CngTh = AveULDR*CongCtrlRTT를 계산하는데 사용된 추정된 RTT

<235> 출력 데이터 레이트

<236> 16 시작 레이트(KB/초)

<237> 25 출력 레이트를 변경하기 위한 시간(초)

<238> 8 변경 레이트(KB/초)

<239> 45 출력 레이트를 변경하기 위한 시간(초)

<240> 32 변경 레이트(KB/초)

<241> 상기 결과들로부터 알 수 있는 바와 같이, 실제 물리 데이터 레이트가 16 kB/초 내지 8 kB/초 또는 32 kB/초까지 변할지라도, 큐의 출력 데이터 레이트는 항상 물리 데이터 레이트와 동일한 경향이 있고 어떤 데이터 레이트들에서도 과도한 큐잉이 존재하지 않는다.

<242> MinTimeIntFunc 설명

<243> 이전에 언급된 바와 같이, 큐 크기를 제어하기 위하여 메인 신호로서 제어 신호 최소 시간 간격이 사용되는 경우, 최소 시간 간격은 다음과 같이 설정되어야 한다:

$$1.7*DR*RTT^2 * IPSegSize +3*RTT$$

<244> 최소 시간 간격 = $$TCPSegSize^2$$

<245> DR=출력 데이터 레이트

<246> 동적 정체 임계값과 유사하게, 최소 시간 간격은 RTT의 추정치 및 물리 출력 데이터 레이트에 기초하여 동적으로 또한 계산될 수 있다. 상기 계산이 복잡한 것처럼 보일지라도, 그것은 IPSegSize로서 단순화될 수 있고 TCPSeqSize는 종종 상수들이다:

<247> 최소 시간 간격 = K*DR*RTT^2_ + 3*RT

<248> 상기에서 $$K\sim= 1.7*IPSegSize/TCPSegSize^2$$

<249> 동일한 방법들이 이전에 언급된 RTT 및 출력 데이터 레이트를 추정하는데 사용될 수 있다.

<250> 하기의 시뮬레이션 결과는 최소 시간 간격이 다음과 같이 동적으로 계산되는 큐 관리 시스템에 대한 것이다:

<251> 최소 시간 간격 = K*DR*RTT^2_ + 3*RT

<252> 모든 다른 제어 신호들은 정적이다. 출력 데이터 레이트는 데이터가 큐를 빠져 나가고 있는 레이트에 의해 추정된다. 그다음 출력 데이터 레이트는 단극 IIR 이동 평균 필터를 사용하여 필터링된다. 이 시뮬레이션을 위해, RTT는 추정되지 않고 어떤 값(EstRTT)으로 고정된다. 이 경우, 실제 RTT는 0.4초이지만 EstRTT는 0.4375초로 설정된다. 실제 물리 데이터 레이트는 16 kB/초 내지 8 kB/초 또는 32 kB/초까지 변할 수 있다. 큐에 대한 입력 데이터는 단일 TCP 스트림이다. 도 8a는 시뮬레이션의 결과를 보여준다:

<253> 시뮬레이션 매개 변수들:

<254> 정체 제어 알고리즘 매개 변수들

<255> 3000 Start_CngTh 시작 정체 임계값

<256> 2 MinDropInterval (초) 폐기된 패킷들 간의 초기 최소한의 시간의 양

<257> 0.4375 EstRTT - MinDropInterval을 계산하는데 사용된 추정된 RTT

<258> 출력 데이터 레이트

<259> 16 시작 레이트(KB/초)

<260> 25 출력 레이트를 변경하기 위한 시간(초)

<261> 8 변경 레이트(KB/초)

<262> 45 출력 레이트를 변경하기 위한 시간(초)

<263> 32 변경 레이트(KB/초)

<264> 상기 결과들로부터 알 수 있는 바와 같이, 실제 물리 데이터 레이트가 16 kB/초 내지 8 kB/초 또는 32 kB/초까지 변할지라도, 큐의 출력 데이터 레이트는 항상 물리 출력 데이터 레이트와 동일한 경향이 있고 과도한 큐잉이 존재하지 않는다.

<265> MinPacketIntFunc 설명

<266> MinPacketIntFunc은 최소 패킷 간격 제어 신호를 동적으로 만드는데 사용된다. 이전에 언급된 바와 같이, 큐 크기를 제어하기 위하여 메인 신호로서 제어 신호 최소 패킷 간격이 사용되는 경우, 최소 패킷 간격은 다음과 같이 설정되어야 한다:

<267> 최소 패킷 간격 = (DR/IPSegSize)*최소 시간 간격

$$(DR/\ IPSegSize)\ *(1.7*DR*RTT^2 * IPSegSize +3*RTT)$$

<268> 최소 패킷 간격 = $$TCPSegSize^2$$

<269> DR=출력 데이터 레이트

<270> 상기 공식에는 변수만이 존재하여서 상기 공식은 다음과 같이 단순화될 수 있다:

<271> 최소 패킷 간격 = K1*DR^2*RTT^2+K2*RTT*DR

<272> 동일한 방법들이 이전에 언급된 RTT 및 출력 데이터 레이트를 추정하는데 사용될 수 있다.

<273> TCP 이외의 트래픽 고려 사항

<274> 이제까지, 모든 시뮬레이션에 대한 큐로의 입력은 단일 TCP 스트림이었다. 하기 섹션은 큐 관리 시스템이 TCP IP 트래픽이 행하는 것과 동일한 정체 제어 규칙들을 준수하지 않는 UDP와 같은 IP 트래픽을 포함하는 입력 스트림들에 반응하는 방법을 고찰할 것이다.

<275> 하기 시뮬레이션은 도 7a에 도시된 동일한 큐 관리 시스템을 사용하고 있는데 정체 임계값만이 동적이고 모든 다른 제어 신호들이 사용되지만 이들은 정적이다. 큐에 대한 입력 데이터 스트림은 6 kB/초에서 TCP 데이터 스트림 및 UDP 데이터 스트림이고 물리 출력 데이터 레이트는 8kB/초 내지 32kB/초 사이에서 변한다. 도 9a는 다음 결과를 보여준다:

<276> 시뮬레이션 매개 변수들:

<277> UDP 매개 변수들

<278> 6000 UDP_DR-UDP 데이터 레이트(바이트/초)

<279> 정체 제어 알고리즘 매개 변수들

<280> 3000 Start_CngTh 시작 정체 임계값

<281> 0 MinPacketInterval - 폐기들 간의 최소한의 패킷들의 수

<282> 2 MinDropInterval (초) 폐기된 패킷들 간의 초기 최소한의 시간의 양

<283> 0.4375 EstRTT - MinDropInterval을 계산하는데 사용된 추정된 RTT

<284> 출력 데이터 레이트

<285> 16 시작 레이트(KB/초)

<286> 60 출력 레이트를 변경하기 위한 시간(초)

<287> 8 변경 레이트(KB/초)

<288> 90 출력 레이트를 변경하기 위한 시간(초)

<289> 32 변경 레이트(KB/초)

<290> 상기 결과는 큐의 출력 데이터 레이트가 양호하지만 어떤 경우에, 48초 마크 및 80초 마크에서 많은 UDP 패킷들이 로우(row)에서 폐기될 때 과도한 큐잉이 발생하는, 물리 계층 출력과 동일한 경향이 있다는 것을 보여준다. 최소 시간 간격은 큐 관리자가 패킷들을 더 빨리 폐기하지 못하게 하고 있다.

<291> 과도한 큐잉의 문제점은 UDP가 물리 출력 데이터 레이트를 넘어 증가하는 경우 관리가능하지 않게 된다. 다음은 상기와 동일한 시스템의 시뮬레이션이지만 UDP 데이터 레이트가 18 kB/초로 설정되고, 물리 출력 데이터 레이트는 단지 16 kB/초로 설정된다. 도 10a의 그래프로부터 알 수 있는 바와 같이, 큐 관리 알고리즘이 충분히 빠르게 패킷들을 폐기하고 있지 않음에 따라 큐 크기는 제어불가능하게 커진다.

<292> 다음 섹션은 이 문제를 해결하도록 구현될 수 있는 알고리즘에의 부가물의 개요이다.

<293> 프로토콜 특정 폐기 규칙들

<294> 이 문제에 대한 하나의 해법은 폐기를 위한 프로토콜 특정 규칙들을 통합하는 것이다. 예를 들어, 최소 시간 간격은 UDP 패킷 대 TCP 패킷에 대해 다를 것이다. UDP 패킷들에 대해, 최소 시간 간격은 0 근처에 있을 수 있거나 있어야 하는데, 왜냐하면 TCP 패킷들과는 달리, 작동하기 위하여 정체 제어를 대기할 필요가 없기 때문이다. 정체 임계값, 최소 패킷 간격 및 폐기 크기는 또한 상이한 프로토콜들에 대해 다를 수 있다. 이 접근법에 대한 주된 단점은 큐 관리 시스템이 송신하고 있는 전송 프로토콜을 결정하기 위하여 큐 관리 시스템이 패킷을 파싱

할 수 있을 것을 요구한다는 것이다. 처리 단점 이외에도, VPN 또는 다른 암호화가 사용되는 경우, 큐 관리 시스템은 프로토콜 유형을 결정하기 위하여 패킷을 파싱할 수 없다. 이전에 언급된 바와 같이, 이 암호화 문제에 대한 가능한 해법은 사용된 프로토콜을 결정하기 위하여 그리고 이에 따라 패킷들을 마킹하기 위하여 필터 드라이버와 같은 실체를 암호화 이전에 사용하는 것일 것이다.

<295> 동적 2 스테이지 최소 폐기 간격 메커니즘

<296> 과도한 큐잉 문제를 해결하기 위한 다른 방법은 본 문서에서 "동적 2 스테이지 최소 폐기 간격" 또는 DTSM으로 지칭되는 큐 관리 방법에 부가적인 로직을 추가하는 것이다. DTSM의 기본적인 개념은 큐가 정체 임계값을 과도하게 초과한 경우 최소 폐기 간격을 낮추는 것이다. 최소 폐기 간격이 낮추어져야 하는 양은 현재의 큐 깊이가 정체 임계값을 얼마나 많이 초과하고 있는지와 관련된다. 권고된 방법은 정체 임계값 초과 비율(CngThExceedRatio)이라고 불리우는 새로운 제어 매개 변수를 정의하는 것이다. CngThExceedRatio는 최소 시간 간격이 0과 동일할 경우 큐 깊이가 정체 임계값을 초과할 수 있는 양을 비율로 나타낸 것이다. 하기의 수학식들은 추가 상세 및 설명을 부가한다.

<297> 기본 최소 폐기 간격:

<298> 큐 깊이가 정체 임계값보다 작은 경우 최소 폐기 간격

<299> CngThExceedRatio:

<300> 최소 시간 간격이 0과 동일할 경우 큐 깊이가 정체 임계값을 초과할 수 있는 양을 비율로 나타낸 것이라 하자.

<301> 큐 깊이가 정체 임계값을 초과하는 경우 최소 폐기 간격은 다음과 같이 계산될 수 있다:

<302> 최소 폐기 간격 = 기본 최소 폐기 간격 * MinDropIntervalMultiplier

<303> 그렇지 않으면

<304> 최소 폐기 간격 = 기본 최소 폐기 간격

<305> MinDropIntervalMultiplier를 계산하기 위한 무한한 가능한 방법들이 존재한다. 다음은 상기 계산의 범위이다:

<306> ● MinDropIntervalMultiplier = 1.0 큐 깊이가 정체 임계값과 동일한 경우

<307> ● MinDropIntervalMultiplier = 0.0 큐 깊이가 정체 임계값*CngThExceedRatio와 동일한 경우

<308> ● ExceedRatio가 증가함에 따라 MinDropIntervalMultiplier는 증가해야 한다.

<309> ExceedRatio는 큐 깊이가 정체 임계값을 초과하는 비율이고 다음과 같이 계산된다:

<310> $$ExceedRatio = \frac{(\text{큐 깊이} - \text{정체 임계값})}{(\text{정체 임계값})}$$

<311>

<312> MinDropIntervalMultiplier는 단순하게 선형 보간 공식을 사용하여 계산될 수 있다:

<313> $$MinDropIntervalMultiplier = 1 - ExceedRatio/CngThExceedRatio$$

<314> 대안적으로 하지만 더 복잡하게, MinDropIntervalMultiplier를 계산하기 위하여 다음과 같이 주어지는 지수 관계가 사용될 수 있다:

<315> $$MinDropIntervalMultiplier = 1 - ExponentialBase^{ExceedRatio/CngThExceedRatio}$$

<316> 큐 깊이가 정체 임계값*CngThExceedRatio를 초과하는 경우 지수 공식은 음수를 반환할 수 있기 때문에, MinDropIntervalMultiplier는 0 내지 1의 범위로 제한되어야 한다. 지수 함수가 더 계산 집중적일지라도, 그것은 패킷들을 틀리게 폐기하는일 없이 낮은 CngThExceedRatio가 사용되도록 허용한다.

<317> 큐 깊이는 빨리 변할 수 있기 때문에, 발생할 수 있는 과도현상을 제거하기 위하여 계산된 최소 폐기 간격은 평탄화되거나 필터링되는 것이 권고된다. 이 필터링은 자연적으로 발생할 수 있는 큐 깊이에서의 피크로 인하여 최소 폐기 간격을 틀리게 낮출 가능성을 줄여줄 것이다.

<318> 몇몇 시뮬레이션들은 동적 2 스테이지 최소 폐기 간격 메커니즘(DTSM)을 사용하여 수행되었다. 도시된 모든 시뮬레이션들은 ExponentialBase=4인 MinDropIntervalMultiplier의 지수 계산을 사용하고 있다. CngThExceedRatio는 모든 시뮬레이션들에 대해 50%로 설정된다.

<319> 하기의 시뮬레이션 결과는 UDP 레이트 = 18 kB/초이고 물리 출력 레이트 = 16 kB/초인 경우 도 10a에 도시된 것과 동일한 입력 데이터 스트림 및 물리 계층 매개 변수들을 사용하고 있다.

<320> 시뮬레이션 매개 변수들:

<321> UDP 매개 변수들

<322> 18 UDP_DR-UDP 데이터 레이트(K바이트/초)

<323> 정체 제어 알고리즘 매개 변수들

<324> 4000 Start_CngTh 시작 정체 임계값

<325> 1200 DropSizeTh - 폐기할 패킷의 최소한의 크기

<326> 2 MinDropInterval (초) 폐기된 패킷들 간의 초기 최소한의 시간의 양

<327> 0.4375 EstRTT - 추정된 RTT

<328> 0.5 CngTh_Exceed_Ratio - MinDropInterval이 0에 접근하는 경우

<329> 큐 깊이가 증가할 수 있는 최대 비율

<330> 물리 계층

<331> 16 출력 데이터 레이트(KB/초)

<332> 도 11a로부터 알 수 있는 바와 같이, TCP 세션이 느린 시작 상태에 있는 경우 단지 어떤 과도한 큐잉이 시작시에 발생한 채 큐잉 깊이가 이제 제어된다. 큐는 결코 비워지지 않기 때문에, 큐의 출력 레이트가 물리 출력 레이트와 동일한 것은 놀랄 일이 아니다. 도 11a는 또한 느린 시작 절차동안, TCP가 정체 제어 상태에 있는 경우 TCP 스트림이 약 3500 kB/초를 사용하고 있는 것에 비해, UDP 스트림이 더 큰 비율의 물리 출력 레이트를 사용하고 있는 것을 보여준다. UDP 레이트가 물리 출력 레이트보다 더 큰 경우, TCP 정체 윈도는 전형적으로 패킷들의 초과 폐기로 인하여 1 세그먼트로 제한된다. 이 작은 정체 윈도는 DBP 공식에 기초하여 TCP 레이트를 제한한다(상기 시뮬레이션에 대해 MTUSize/RTT 1450/0.4=3625 바이트/초).

<333> 도 12a에 도시된 다음 시뮬레이션 출력은 이전과 같은 동일한 DTSM 큐 관리 시스템을 사용하고 있지만 이번에 물리 출력 레이트는 16 내지 8 또는 32 kB/초까지 변한다.

<334> 시뮬레이션 매개 변수들:

<335> UDP 매개 변수들

<336> 18 UDP_DR-UDP 데이터 레이트(K바이트/초)

<337> 정체 제어 알고리즘 매개 변수들

<338> 4000 Start_CngTh 시작 정체 임계값

<339> 2 MinDropInterval (초) 폐기될 패킷들 간의 초기 최소한의 시간의 양

<340> 0.4375 EstRTT - MinDropInterval을 계산하는데 사용된 추정된 RTT

<341> 0.5 CngTh_Exceed_Ratio - MinDropInterval이 0에 접근하는 경우

<342> 큐 깊이가 증가할 수 있는 최대 비율

<343> 출력 데이터 레이트

<344> 16 시작 레이트(KB/초)

<345> 60 출력 레이트를 변경하기 위한 시간(초)

<346> 8 변경 레이트(KB/초)

<347> 90 출력 레이트를 변경하기 위한 시간(초)

<348> 32 변경 레이트(KB/초)

<349> 상기 시뮬레이션 결과는 동적 정체 임계값 신호와 조합하여 DTSM의 부가가 다양한 물리 출력 레이트에 대해 큐잉 깊이를 감소시킨다는 것을 보여준다.

<350> DTSM 방법에 대한 2가지 알려져 있는 단점들이 존재한다. 첫번째는 큐 관리 시스템이 TCP 느린 시작 메커니즘동안 하나보다 많은 패킷을 잘못하여 폐기할 것이라는 것이다. TCP 스택이 정체 제어 상태로 이동한 후에, 이 방법은 단지 하나의 패킷을 폐기한 후에 TCP 스택이 응답하는 것을 대기함으로써 의도된 바와 같이 동작한다. 두번째 단점은 물리 출력 레이트가 급작스럽게 저하되는 경우, DTSM 방법은 하나보다 많은 패킷들을 잘못하여 폐기할 수 있다는 것이다. 물리 출력 레이트의 급작스러운 저하는 정체 임계값이 급작스럽게 저하되도록 야기하는데, 이것은 이 때의 큐의 현재의 깊이에 의존하여 현재의 큐 깊이가 정체 임계값을 매우 초과하는 상태를 생성할 수 있다. 이 해로운 영향은 더 큰 CngTh_Exceed_Ratio가 사용되는 경우 최소화될 수 있다. 100% 보다 큰 CngTh_Exceed_Ratio는 이들 부정적인 영향들 양자를 제거하는데 필요하지만 이것은 물론 증가된 큐 깊이의 희생을 필요로 한다.

<351> 이들 두가지 영향들을 보여주기 위하여, 도 13a의 다음 시뮬레이션 결과는 이전과 같은 동일한 DTSM 큐 관리 시스템을 사용하고 있지만 이번에 큐에의 입력은 UDP 데이터없이 단지 TCP 데이터이다.

<352> 시뮬레이션 매개 변수들:

<353> UDP 매개 변수들

<354> 0 UDP_DR-UDP 데이터 레이트(KB/초)

<355> 정체 제어 알고리즘 매개 변수들

<356> 4000 Start_CngTh 시작 정체 임계값

<357> 2 MinDropInterval (초) 폐기된 패킷들 간의 초기 최소한의 시간의 양

<358> 0.4375 EstRTT - MinDropInterval을 계산하는데 사용된 추정된 RTT

<359> 0.5 CngTh_Exceed_Ratio - MinDropInterval이 0에 접근하는 경우

<360> 큐 깊이가 증가할 수 있는 최대 비율

<361> 출력 데이터 레이트

<362> 16 시작 레이트(KB/초)

<363> 60 출력 레이트를 변경하기 위한 시간(초)

<364> 8 변경 레이트(KB/초)

<365> 90 출력 레이트를 변경하기 위한 시간(초)

<366> 32 변경 레이트(KB/초)

<367> 상기 시뮬레이션의 ~4초 마크에서, TCP 스택이 느린 시작 상태에 있는 경우 다중 TCP 패킷들이 존재한다. 이것은 ~7초 마크에서 평균 큐 출력 레이트의 일시적인 저하를 야기한다.

<368> 두번째 부정적인 영향은 물리 출력 레이트가 16 kB/초에서 8 kB/초로 되는 경우, ~60초 마크에서 나타난다. 이때 우리는 큐 관리 시스템이 평균 큐 출력 레이트의 일시적인 감소를 야기하는 몇몇 TCP 패킷들을 잘못하여 폐기한다는 것을 알 수 있다.

<369> CngTh_Exceed_Rate를 큰 값으로 설정하는 것은 과도한 큐잉을 야기하지만 CngTh_Exceed_Rate를 너무 작은 값으로 설정하는 것은 도 13a에서 볼 수 있는 바와 같이 잘못된 패킷 폐기를 야기할 것이다. 고차 지수 수학식을 사용하여 최소 폐기 간격을 계산하는 것은 낮은 CngTh_Exceed_Rate가 사용될 수 있도록 하지만 이것은 약간만 도움을 줄 뿐이다.

<370> 상기한 큐 관리 시스템이 동적 정체 임계값과 함께 DTSM을 사용하고 있을지라도, DTSM 시스템은 또한 이전에 설명된 동적 최소 시간 간격 및 최소 패킷 간격 시스템과 함께 사용될 수 있다.

<371>

도면의 간단한 설명

<9> 도 1은 종래 기술에 따라 FCI를 구현하는 시스템 및 그 단점들을 도시한 것이다.

<10> 도 2는 본 발명의 일 실시예에 의한 데이터 통신 시스템을 도시한 것이다.

<11> 도 3은 본 발명의 일 실시예에 의한 데이터 통신 시스템의 큐 버퍼 관리를 달성하는 프로세스를 도시한 것이다.

<12> 도 4는 본 발명의 일 실시예에 의한 큐 버퍼 관리를 위한 방법의 프로세스 흐름도를 도시한 것이다.

<13> 도 5는 본 발명의 일 실시예에 의한 데이터를 전달하는데 사용될 수 있는 디지털 처리 시스템의 기능 블록도를 도시한 것이다.

<14> 도 1a는 고정된 정체 임계값 시뮬레이션을 도시한 것이다.

<15> 도 2a는 고정된 정체 임계값 시뮬레이션-가변 출력 DR을 도시한 것이다.

<16> 도 3a는 패킷들 간의 최소 시간이 없는 고정 정체 임계값 시뮬레이션을 도시한 것이다.

<17> 도 4a는 고정 최소 시간 간격 시뮬레이션을 도시한 것이다.

<18> 도 5a는 가변 출력 레이트를 갖는 고정 최소 시간 간격 시뮬레이션을 도시한 것이다.

<19> 도 6a는 고정 최소 패킷 간격 시뮬레이션을 도시한 것이다.

<20> 도 7a는 동적 정체 제어 시뮬레이션을 도시한 것이다.

<21> 도 8a는 동적 최소 시간 간격 시뮬레이션을 도시한 것이다.

<22> 도 9a는 UDP 레이트<물리 출력 레이트인 정체 시뮬레이션을 도시한 것이다.

<23> 도 10a는 UDP>물리 출력 레이트인 시뮬레이션을 도시한 것이다.

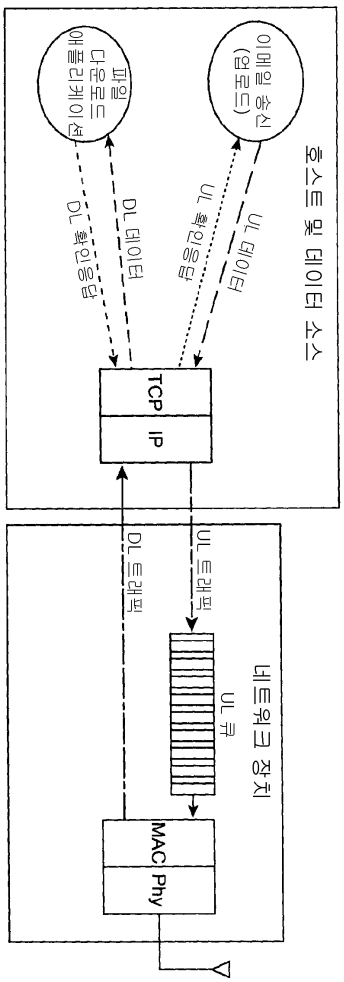
<24> 도 11a는 동적 2 스테이지 최소 시간 간격 메커니즘을 사용하는 시뮬레이션을 도시한 것이다.

<25> 도 12a는 가변 출력 레이트를 갖는 DTSM을 사용하는 시뮬레이션을 도시한 것이다.

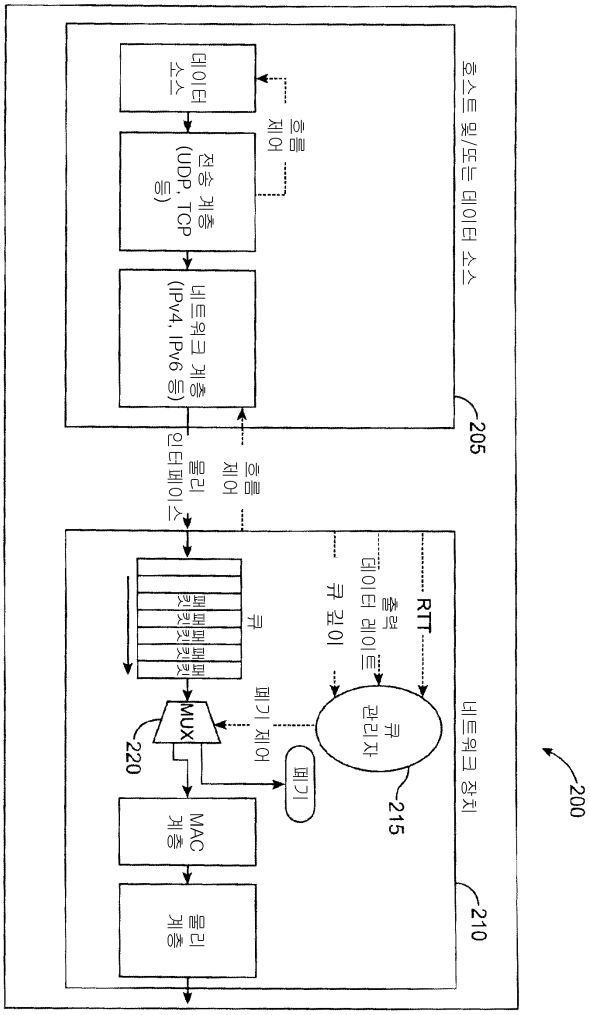
<26> 도 13a는 가변 출력 레이트를 갖지만 UDP 트래픽이 없는 DTSM을 사용하는 시뮬레이션을 도시한 것이다.

도면

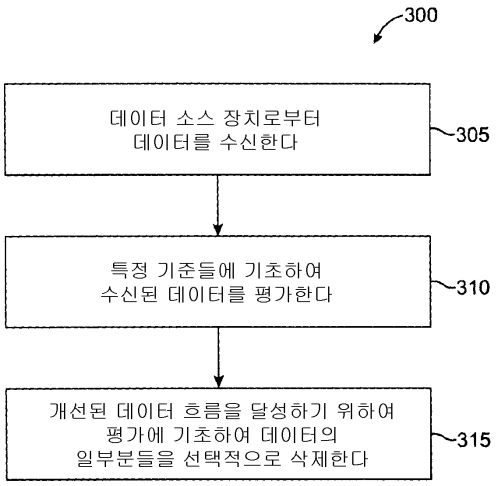
도면1


호스트 및 데이터 소스
이메일 송신 (업로드)
파일 다운로드 애플리케이션
UL 데이터
UL 확인응답
DL 데이터
DL 확인응답
TCP
IP
네트워크 장치
UL 트래픽
DL 트래픽
UL 큐
MAC
Phy

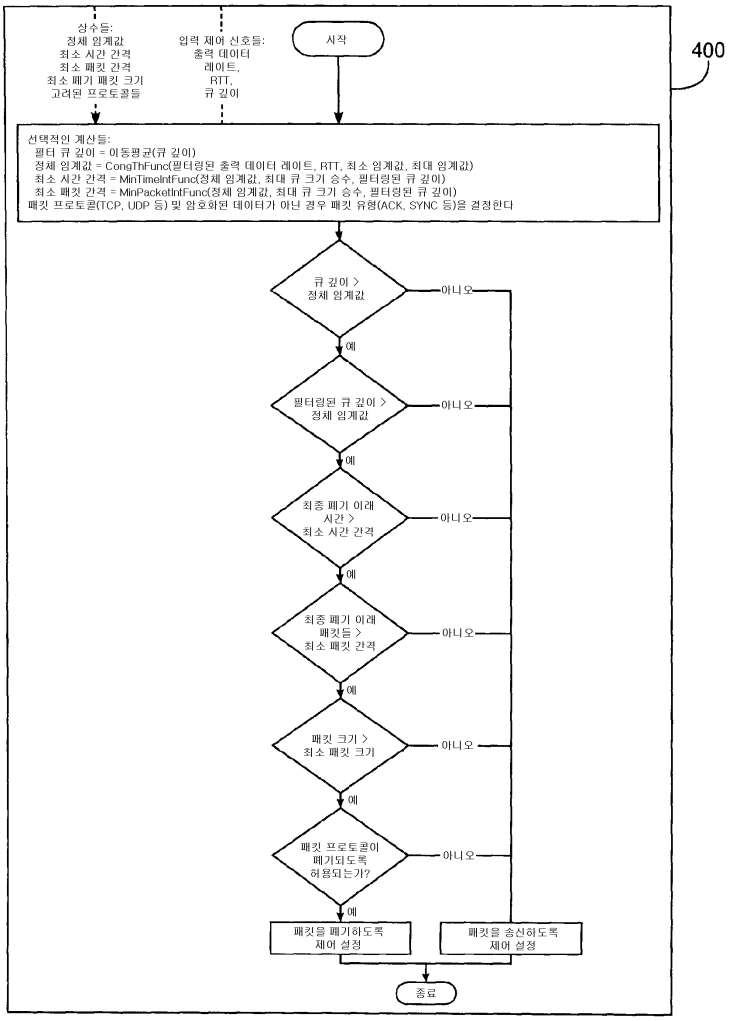
도면2


호스트 및/또는 데이터 소스
205
데이터 소스
흐름 제어
전송 계층 (UDP, TCP 등)
네트워크 계층 (IPv4, IPv6 등)
물리 인터페이스
흐름 제어
네트워크 장치
210
큐
패킷 패킷 패킷 패킷
큐 깊이
출력 데이터 레이트
RTT
MUX
220
폐기 제어
폐기
큐 관리자
215
MAC 계층
물리 계층
200

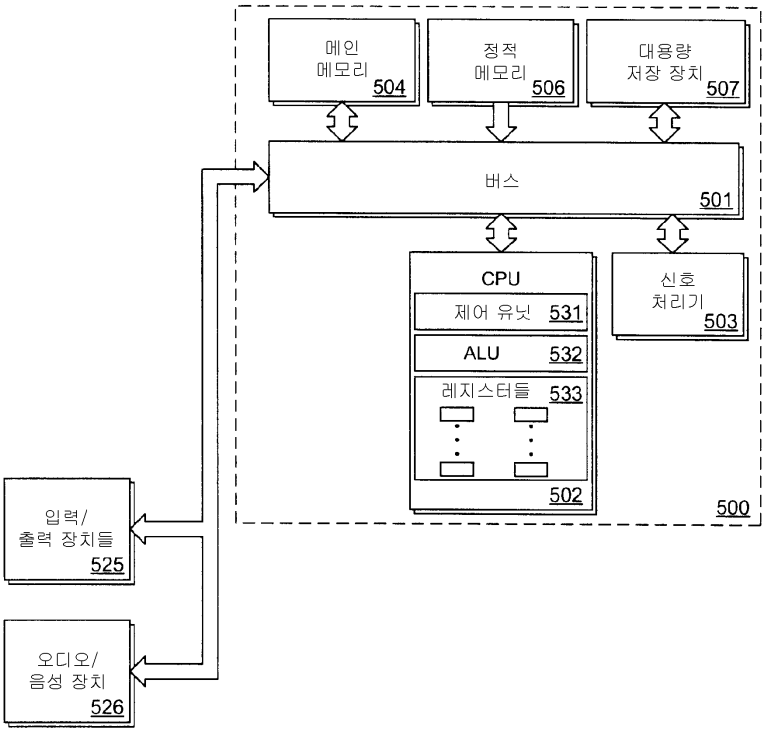
도면3


300
데이터 소스 장치로부터 데이터를 수신한다
305
특정 기준들에 기초하여 수신된 데이터를 평가한다
310
개선된 데이터 흐름을 달성하기 위하여 평가에 기초하여 데이터의 일부분들을 선택적으로 삭제한다
315

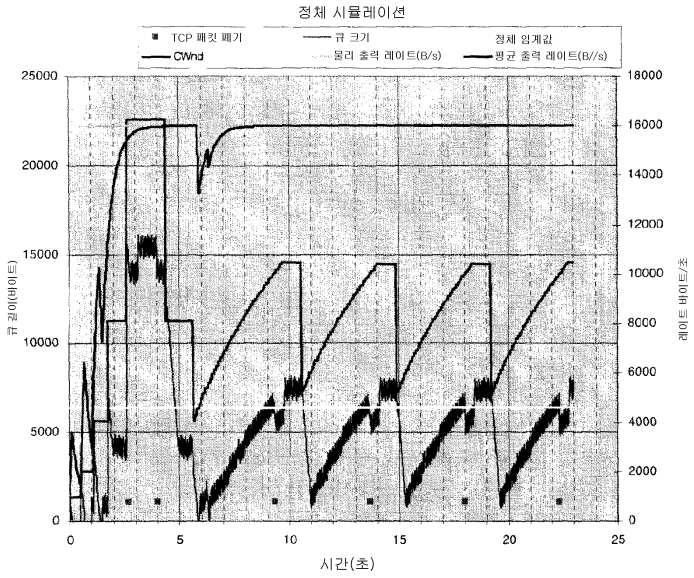
도면4


400
상수들:
정체 임계값
최소 시간 간격
최소 패킷 간격
최소 폐기 패킷 크기
고려된 프로토콜들
입력 제어 신호들:
출력 데이터
레이트,
RTT,
큐 깊이
시작
선택적인 계산들:
필터 큐 깊이 = 이동평균(큐 깊이)
정체 임계값 = CongThFunc(필터링된 출력 데이터 레이트, RTT, 최소 임계값, 최대 임계값)
최소 시간 간격 = MinTimeIntFunc(정체 임계값, 최대 큐 크기 승수, 필터링된 큐 깊이)
최소 패킷 간격 = MinPacketIntFunc(정체 임계값, 최대 큐 크기 승수, 필터링된 큐 깊이)
패킷 프로토콜(TCP, UDP 등) 및 암호화된 데이터가 아닌 경우 패킷 유형(ACK, SYNC 등)을 결정한다
큐 깊이 >
정체 임계값
아니오
예
필터링된 큐 깊이 >
정체 임계값
아니오
예
최종 폐기 이래
시간 >
최소 시간 간격
아니오
예
최종 폐기 이래
패킷들 >
최소 패킷 간격
아니오
예
패킷 크기 >
최소 패킷 크기
아니오
예
패킷 프로토콜이
폐기되도록
허용되는가?
아니오
예
패킷을 폐기하도록
제어 설정
패킷을 송신하도록
제어 설정
종료

도면5

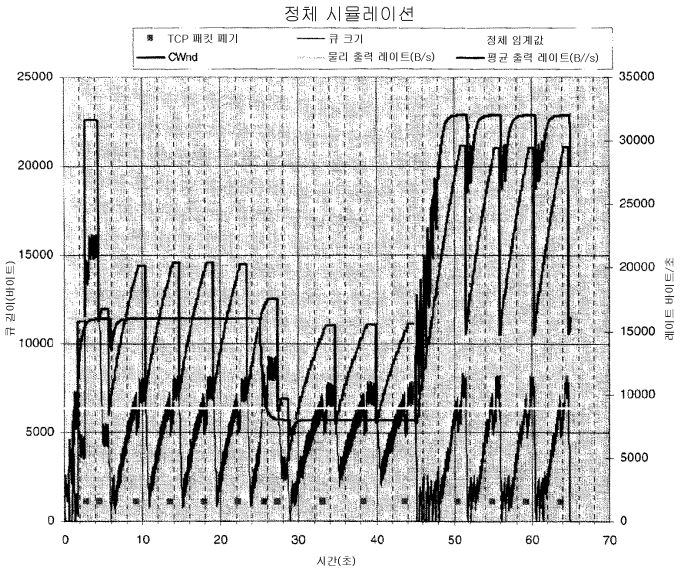


도면1a



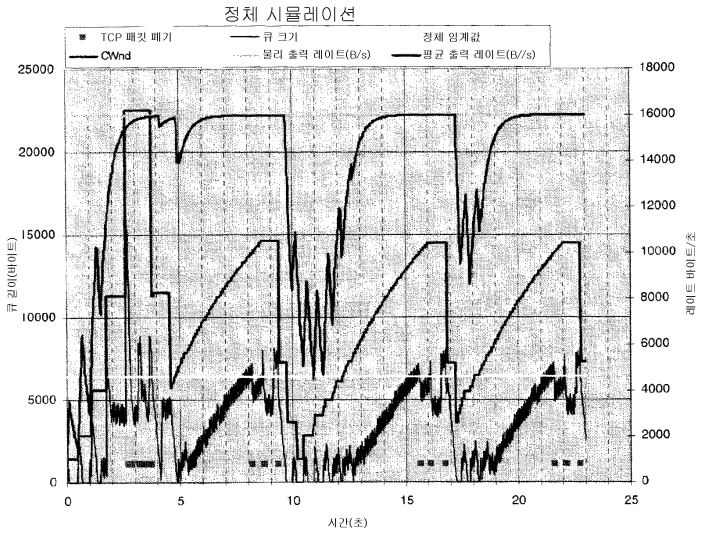
고정된 정체 임계값 시뮬레이션

도면2a



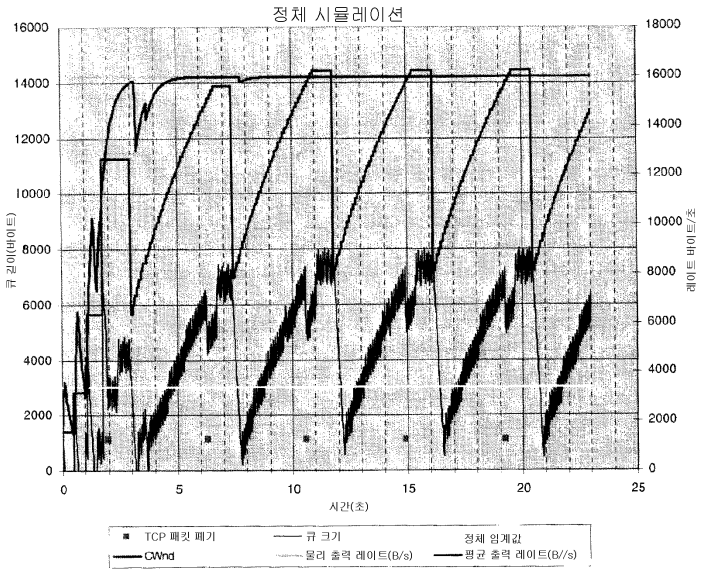
고정된 정체 임계값 시뮬레이션 - 가변 출력 DR

도면3a



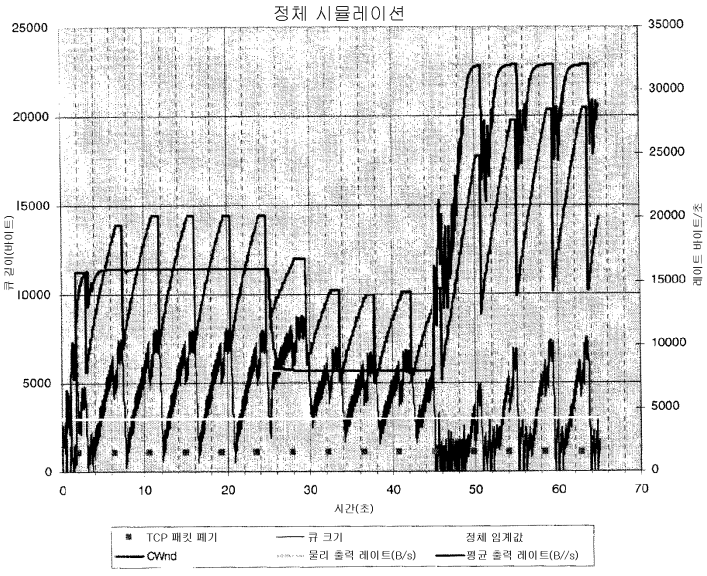
패킷들간의 최소 시간이 없는 고정된 정체 임계값 시뮬레이션

도면4a



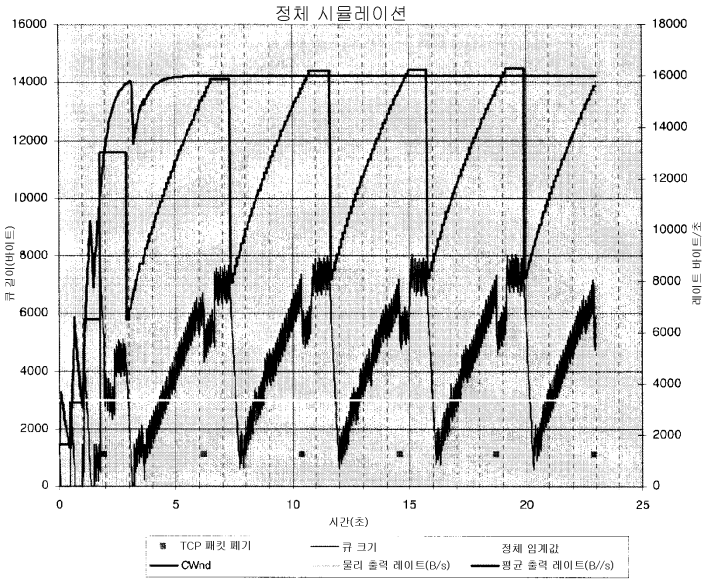
고정 최소 시간 간격 시뮬레이션

도면5a



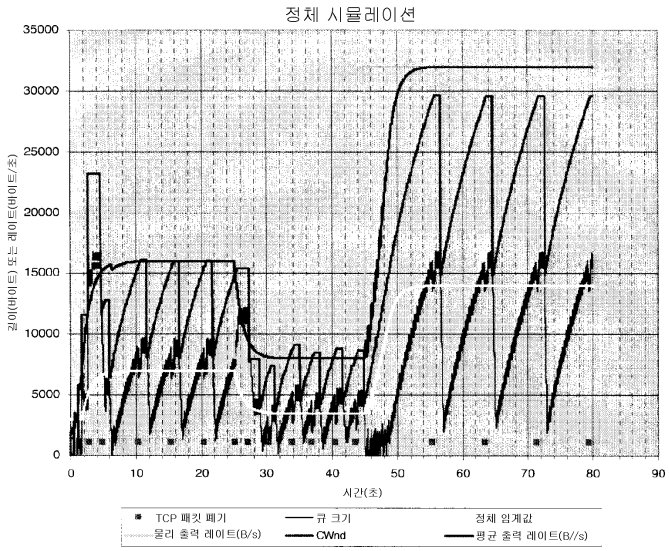
가변 출력 레이트를 갖는 고정 최소 시간 간격 시뮬레이션

도면6a



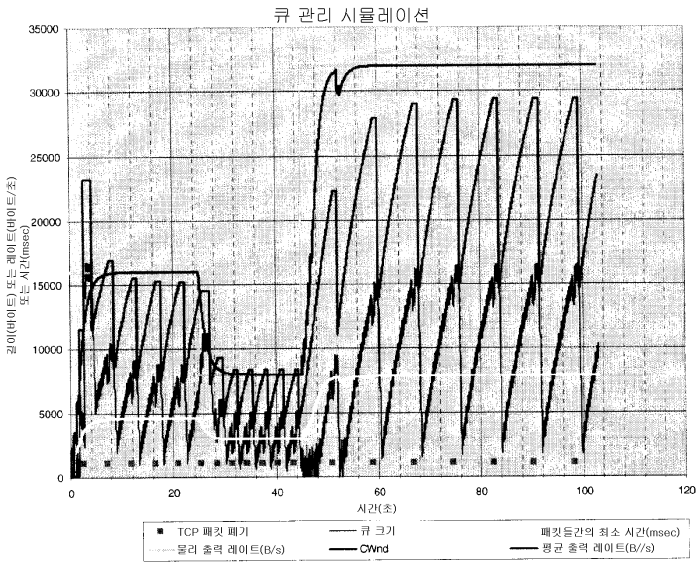
고정 최소 패킷 간격 시뮬레이션

도면7a



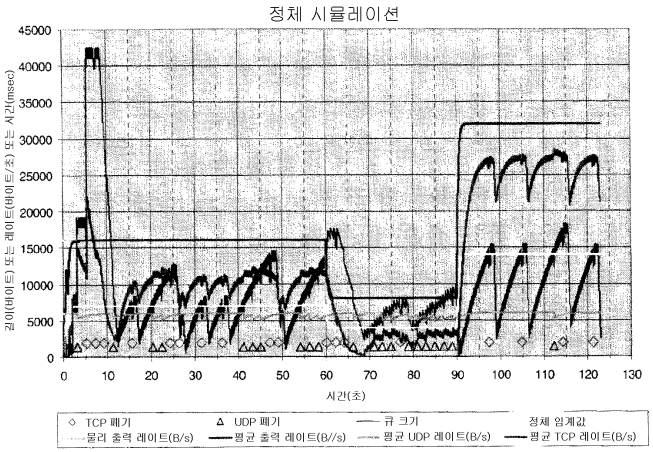
동적 정체 제어 시뮬레이션

도면8a



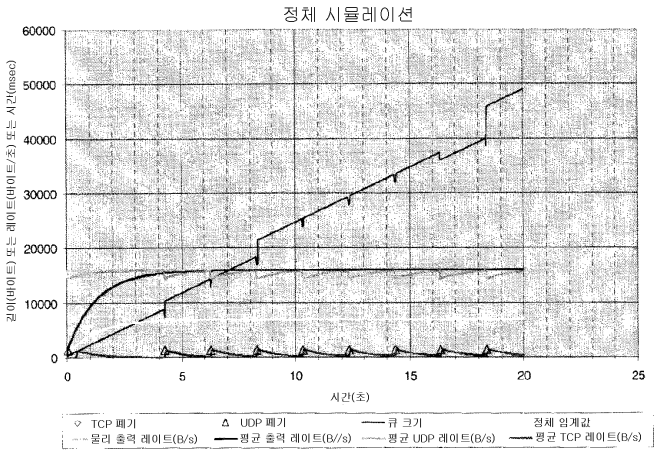
동적 최소 시간 간격 시뮬레이션

도면9a



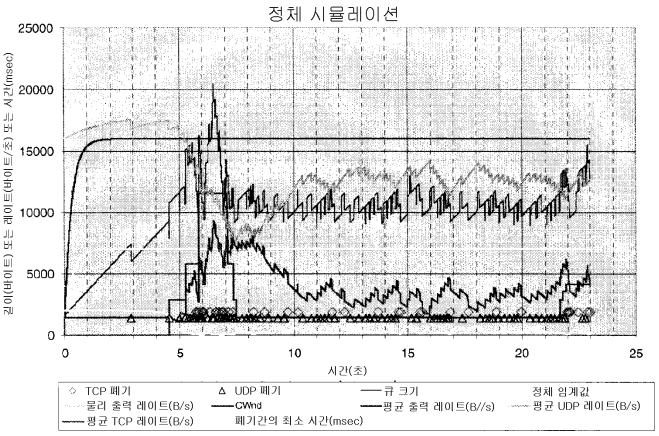
UDP 레이트 < 물리 출력 레이트인 정체 시뮬레이션

도면10a



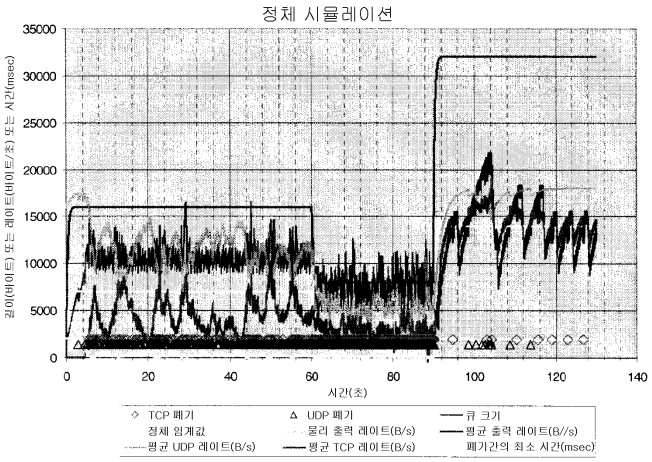
UDP>물리 출력 레이트인 시뮬레이션

도면11a



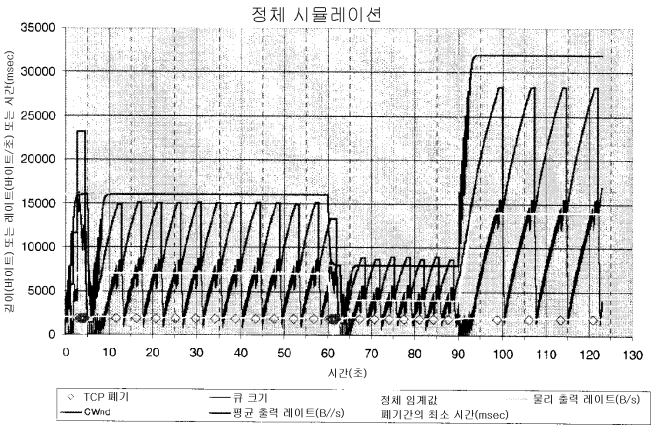
동적 2 스테이지 최소 시간 간격 메커니즘을 사용하는 시뮬레이션

도면12a



가변 출력 레이트를 갖는 DTSM을 사용하는 시뮬레이션

도면13a



가변 출력 레이트를 갖지만 UDP 트래픽이 없는 DTSM을 사용하는 시뮬레이션