(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2017-0125564
(43) 공개일자 2017년11월15일

(51) 국제특허분류(Int. Cl.)
H04L 12/24 (2006.01) *H04L 12/741* (2013.01)
H04L 12/801 (2013.01) *H04L 29/06* (2006.01)
(52) CPC특허분류
H04L 41/042 (2013.01)
H04L 45/74 (2013.01)
(21) 출원번호 10-2016-0055468
(22) 출원일자 2016년05월04일
심사청구일자 없음

(71) 출원인
한국전자통신연구원
대전광역시 유성구 가정로 218 (가정동)
(72) 발명자
이현용
광주광역시 북구 동문대로121번길 83-4 (우산동)
이범철
대전광역시 유성구 노은서로 124, 107동 201호 (노은동, 노은카운티스)
최강일
대전광역시 유성구 봉명로 93, 604동 1802호 (봉명동, 도안6단지센트럴시티아파트)
(74) 대리인
특허법인지명

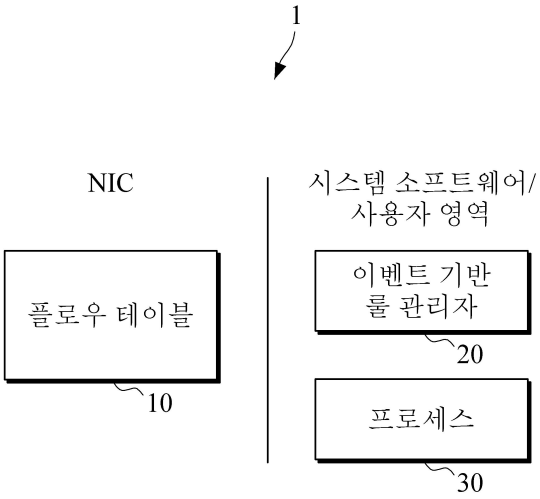
전체 청구항 수 : 총 20 항

(54) 발명의 명칭 **병렬 패킷 처리를 위한 패킷 분배 방법 및 패킷 분배 관리장치**

(57) 요 약

병렬 패킷 처리를 위한 패킷 분배 방법 및 패킷 분배 관리장치가 개시된다. 일 실시 예에 따른 병렬 패킷 처리를 위한 패킷 분배 관리장치는, 패킷을 처리할 프로세스 또는 네트워크 응용이 발생시키는 이벤트를 탐지하는 이벤트 탐지부와, 이벤트 탐지부가 탐지한 이벤트와 관련된 패킷 분배 룰을 생성, 갱신 또는 삭제하여 네트워크 인터페이스 카드가 패킷 분배 룰에 따라 패킷을 분배하도록 하는 룰 관리부를 포함한다.

대 표 도 - 도1



(52) CPC특허분류
H04L 47/33 (2013.01)
H04L 69/162 (2013.01)

이 발명을 지원한 국가연구개발사업
과제고유번호 1711026869
부처명 미래창조과학부
연구관리전문기관 정보통신기술진흥센터
연구사업명 한국전자통신연구원연구개발지원
연구과제명 (대형통합) 스마트 네트워킹 핵심 기술 개발
기 여 율 1/1
주관기관 한국전자통신연구원
연구기간 2015.03.01 ~ 2016.02.29

명 세 서

청구범위

청구항 1

패킷을 처리할 프로세스 또는 네트워크 응용이 발생시키는 이벤트를 탐지하는 이벤트 탐지부; 및

상기 이벤트 탐지부가 탐지한 이벤트와 관련된 패킷 분배 룰을 생성, 갱신 또는 삭제하여 네트워크 인터페이스 카드가 패킷 분배 룰에 따라 패킷을 분배하도록 하는 룰 관리부;

를 포함하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 2

제 1 항에 있어서, 상기 프로세스는

시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스인 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 3

제 1 항에 있어서, 상기 이벤트는

소켓 또는 스레드의 생성이나 종료, 소켓 또는 스레드를 생성한 프로세스나 네트워크 응용의 코어 마이그레이션에 관련된 이벤트인 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 4

제 1 항에 있어서, 상기 이벤트 탐지부는

프로세스 또는 네트워크 응용이 소켓을 생성할 때 이를 탐지하고, 패킷 분배 룰 생성을 위한 소켓 정보와 해당 프로세스가 실행되는 코어 정보를 상기 룰 관리부에 통지하여 패킷 분배 룰 생성을 요청하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 5

제 1 항에 있어서, 상기 이벤트 탐지부는

프로세스 또는 네트워크 응용이 이전에 생성된 소켓에 대한 스레드를 생성할 때 이를 탐지하고, 패킷 분배 룰 생성을 위한 스레드 정보와 해당 스레드가 실행되는 코어 정보를 상기 룰 관리부에 통지하여 패킷 분배 룰 생성을 요청하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 6

제 1 항에 있어서, 상기 이벤트 탐지부는

컨테이너 기반 가상화 환경에서 하이퍼바이저가 컨테이너를 생성하고 포트 포워딩을 통해 특정 포트 번호를 컨테이너에 할당할 때 이를 탐지하고, 패킷 분배 룰 생성을 위한 포트 포워딩 정보와 포트 포워딩이 적용되는 컨테이너가 실행되는 코어 정보를 상기 룰 관리부에 통지하여 패킷 분배 룰 생성을 요청하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 7

제 1 항에 있어서, 상기 이벤트 탐지부는

가상머신 기반 가상화 환경에서 가상머신이 하이퍼바이저에 추가 패킷 분류를 요청할 때 이를 탐지하고, 패킷 분배 룰 생성을 위한 목적지와 송신지 IP 주소, 목적지와 송신지 포트 번호, 프로토콜 타입 및 가상머신 정보를 상기 룰 관리부에 통지하여 패킷 분배 룰 생성을 요청하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 8

제 1 항에 있어서, 상기 이벤트 탐지부는

상이한 기능을 하는 워커 스레드들로 구성된 데이터 플레인 개발 키트(DPDK) 환경에서 동일한 기능을 하는 워커 스레드들의 그룹이 워커 스레드 그룹 식별자를 등록할 때 이를 탐지하고, 워커 스레드 기반 패킷 분배 룰 생성을 위한 정보를 상기 룰 관리부에 통지하여 패킷 분배 룰 생성을 요청하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 9

제 1 항에 있어서, 상기 이벤트 탐지부는

프로세스 또는 네트워크 응용이 현재 실행 중인 코어에서 다른 코어로 마이그레이션될 때 이를 탐지하고, 갱신될 패킷 분배 룰과 관련된 소켓 또는 스레드 정보 및 새로운 코어 정보를 상기 룰 관리부에 통지하여 패킷 분배 룰 갱신을 요청하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 10

제 1 항에 있어서, 상기 이벤트 탐지부는

프로세스 또는 네트워크 응용이 스레드 또는 소켓을 종료할 때 이를 탐지하고, 패킷 분배 룰 삭제를 위한 정보를 상기 룰 관리부에 통지하여 패킷 분배 룰 삭제를 요청하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 11

제 1 항에 있어서, 상기 이벤트 탐지부는

컨테이너 기반 가상화 환경에서, 컨테이너에 대한 포트 포워딩을 종료하거나 컨테이너가 종료될 때 이를 탐지하고, 패킷 분배 룰 삭제를 위한 정보를 상기 룰 관리부에 통지하여 패킷 분배 룰 삭제를 요청하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 12

제 1 항에 있어서, 상기 이벤트 탐지부는

가상머신 기반 가상화 환경에서, 하이퍼바이저에 추가 패킷 분배를 요청했던 가상머신이 이전에 요청된 추가 패킷 분배의 종료를 요청하거나 추가 패킷 분배를 요청했던 가상머신이 종료될 때 이를 탐지하고, 패킷 분배 룰 삭제를 위한 정보를 상기 룰 관리부에 통지하여 패킷 분배 룰 삭제를 요청하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 13

제 1 항에 있어서, 상기 이벤트 탐지부는

상이한 기능을 하는 워커 스레드들로 구성된 데이터 플레인 개발 키트(DPDK) 환경에서, 소정의 워커 스레드 그룹에 속한 워커 스레드들이 모두 종료될 때 이를 탐지하고, 패킷 분배 룰 삭제를 위한 정보를 상기 룰 관리부에 통지하여 패킷 분배 룰 삭제를 요청하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 14

제 1 항에 있어서, 상기 룰 관리부는

프로세스 또는 네트워크 응용이 소켓을 생성할 때, 상기 이벤트 탐지부로부터 통지된 목적지 포트 번호 및 프로토콜 타입에 기초하여 패킷 분배 룰을 생성하고,

프로세스 또는 네트워크 응용이 스레드를 생성할 때, 상기 이벤트 탐지부로부터 통지된 목적지 및 송신지 IP 주소, 목적지 및 송신지 포트 번호, 프로토콜 타입에 기초하여 패킷 분배 룰을 생성하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 15

제 1 항에 있어서, 상기 룰 관리부는

프로세스 또는 네트워크 응용이 실행되는 코어가 변경될 때 관련된 패킷 분배 룰의 코어 정보를 갱신하고,

프로세스 또는 네트워크 응용이 스레드 또는 소켓을 종료할 때 관련된 패킷 분배 룰을 삭제하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 16

제 1 항에 있어서, 상기 룰 관리부는

컨테이너 기반 가상화 환경에서 하이퍼바이저가 컨테이너를 생성하고 포트 포워딩을 통해 특정 포트 번호를 컨테이너에 할당할 때 패킷 분배 룰 생성을 위한 포트 포워딩 정보와 포트 포워딩이 적용되는 컨테이너가 실행되는 코어 정보를 상기 이벤트 탐지부로부터 통지받아 패킷 분배 룰을 생성하고,

가상머신 기반 가상화 환경에서 가상머신이 하이퍼바이저에 추가 패킷 분류를 요청할 때 패킷 분배 룰 생성을 위한 목적지와 송신지 IP 주소, 목적지와 송신지 포트 번호, 프로토콜 타입 및 가상머신 정보를 상기 이벤트 탐지부로부터 통지받아 패킷 분배 룰을 생성하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 17

제 1 항에 있어서, 상기 룰 관리부는

상이한 기능을 하는 워커 스레드들로 구성된 데이터 플레인 개발 키트(DPDK) 환경에서, 소정의 워커 스레드 그룹에 속한 워커 스레드들이 모두 종료될 때 패킷 분배 룰 삭제를 위한 정보를 상기 이벤트 탐지부로부터 통지받아 패킷 분배 룰을 삭제하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 18

제 1 항에 있어서, 상기 룰 관리부는

생성 또는 갱신된 패킷 분배 룰을 플로우 테이블에 저장하고, 삭제가 필요한 패킷 분배 룰을 플로우 테이블에서 삭제하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 관리장치.

청구항 19

패킷을 처리할 프로세스 또는 네트워크 응용이 발생시키는 이벤트를 탐지하는 단계;

탐지한 이벤트와 관련된 패킷 분배 룰을 생성, 갱신 또는 삭제하는 단계; 및

생성, 갱신 또는 삭제된 패킷 분배 룰에 따라 네트워크 인터페이스 카드가 패킷을 분배하도록 하는 단계;

를 포함하는 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 방법.

청구항 20

제 19 항에 있어서,

상기 프로세스는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스이고,

상기 이벤트는 소켓 또는 스레드의 생성이나 종료, 소켓 또는 스레드를 생성한 프로세스나 네트워크 응용의 코어 마이그레이션에 관련된 이벤트인 것을 특징으로 하는 병렬 패킷 처리를 위한 패킷 분배 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 병렬 패킷 처리를 위한 패킷 분배 기술에 관한 것이다.

배 경 기 술

[0002] 클라우드 컴퓨팅, 빅 데이터, 모바일 기술 등이 확산됨에 따라 꾸준히 증가하는 네트워크 트래픽을 처리할 수 있는 고성능 서버가 요구되고 있다. 대용량 트래픽 처리를 위한 고성능 서버를 구현하는 방법은 멀티 코어를 활용하는 방법 외에는 별다른 대안이 없는 실정이다. 멀티 코어에 기반하여 대용량의 트래픽을 처리하는 방법은 다수의 코어를 활용하여 동시에 패킷들을 처리하는 병렬 패킷 처리 방법이다.

[0003] 병렬 패킷 처리를 위한 패킷 분배 측면에서, 응용 affinity(친화성)는 주어진 패킷을 처리할 네트워크 응용이 실행되는 코어의 수신 큐에 패킷을 배정하는 것을 말한다. 응용 affinty를 달성하기 위해서는 네트워크 인터페이스 카드(Network Interface Card: NIC, 이하 NIC라 칭함)에서 수신 큐에 패킷을 분배할 때 특정 플로우(동일 목적지/송신지 IP 주소/포트 번호를 공유하는 패킷들의 집합)를 처리하는 네트워크 응용이 실행되는 코어 정보가 요구된다. NIC에서 요구되는 코어 정보를 획득하는 방법에는 차이가 있으며, 이 차이는 기능 및 성능 상의 차이로 이어진다.

발명의 내용

해결하려는 과제

[0004] 일 실시 예에 따라, 모든 패킷(TCP, UDP 등)에 대해서 시스템 소프트웨어의 성능저하 없이 고성능으로 응용 affinity를 보장할 수 있는 병렬 패킷 처리를 위한 패킷 분배 방법 및 패킷 분배 관리장치를 제안한다.

과제의 해결 수단

[0005] 일 실시 예에 따른 병렬 패킷 처리를 위한 패킷 분배 관리장치는, 패킷을 처리할 프로세스 또는 네트워크 응용이 발생시키는 이벤트를 탐지하는 이벤트 탐지부와, 이벤트 탐지부가 탐지한 이벤트와 관련된 패킷 분배 룰을 생성, 갱신 또는 삭제하여 네트워크 인터페이스 카드가 패킷 분배 룰에 따라 패킷을 분배하도록 하는 룰 관리부를 포함한다.

[0006] 프로세스는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스일 수 있다. 이벤트는 소켓 또는 스레드의 생성이나 종료, 소켓 또는 스레드를 생성한 프로세스나 네트워크 응용의 코어 마이그레이션에 관련된 이벤트일 수 있다.

[0007] 이벤트 탐지부는 프로세스 또는 네트워크 응용이 소켓을 생성할 때 이를 탐지하고, 패킷 분배 룰 생성을 위한 소켓 정보와 해당 프로세스가 실행되는 코어 정보를 룰 관리부에 통지하여 패킷 분배 룰 생성을 요청할 수 있다.

[0008] 이벤트 탐지부는 프로세스 또는 네트워크 응용이 이전에 생성된 소켓에 대한 스레드를 생성할 때 이를 탐지하고, 패킷 분배 룰 생성을 위한 스레드 정보와 해당 스레드가 실행되는 코어 정보를 룰 관리부에 통지하여 패킷 분배 룰 생성을 요청할 수 있다.

[0009] 이벤트 탐지부는 컨테이너 기반 가상화 환경에서 하이퍼바이저가 컨테이너를 생성하고 포트 포워딩을 통해 특정 포트 번호를 컨테이너에 할당할 때 이를 탐지하고, 패킷 분배 룰 생성을 위한 포트 포워딩 정보와 포트 포워딩이 적용되는 컨테이너가 실행되는 코어 정보를 룰 관리부에 통지하여 패킷 분배 룰 생성을 요청할 수 있다.

[0010] 이벤트 탐지부는 가상머신 기반 가상화 환경에서 가상머신이 하이퍼바이저에 추가 패킷 분류를 요청할 때 이를 탐지하고, 패킷 분배 룰 생성을 위한 목적지와 송신지 IP 주소, 목적지와 송신지 포트 번호, 프로토콜 타입 및 가상머신 정보를 룰 관리부에 통지하여 패킷 분배 룰 생성을 요청할 수 있다.

[0011] 이벤트 탐지부는 상이한 기능을 하는 워커 스레드들로 구성된 데이터 플레인 개발 키트(DPDK) 환경에서 동일한 기능을 하는 워커 스레드들의 그룹이 워커 스레드 그룹 식별자를 등록할 때 이를 탐지하고, 워커 스레드 기반 패킷 분배 룰 생성을 위한 정보를 룰 관리부에 통지하여 패킷 분배 룰 생성을 요청할 수 있다.

[0012] 이벤트 탐지부는 프로세스 또는 네트워크 응용이 현재 실행 중인 코어에서 다른 코어로 마이그레이션될 때 이를 탐지하고, 갱신될 패킷 분배 룰과 관련된 소켓 또는 스레드 정보 및 새로운 코어 정보를 룰 관리부에 통지하여 패킷 분배 룰 갱신을 요청할 수 있다.

[0013] 이벤트 탐지부는 프로세스 또는 네트워크 응용이 스레드 또는 소켓을 종료할 때 이를 탐지하고, 패킷 분배 룰 삭제를 위한 정보를 룰 관리부에 통지하여 패킷 분배 룰 삭제를 요청할 수 있다.

[0014] 이벤트 탐지부는 컨테이너 기반 가상화 환경에서, 컨테이너에 대한 포트 포워딩을 종료하거나 컨테이너가 종료

될 때 이를 탐지하고, 패킷 분배 룰 삭제를 위한 정보를 룰 관리부에 통지하여 패킷 분배 룰 삭제를 요청할 수 있다.

[0015] 이벤트 탐지부는 가상머신 기반 가상화 환경에서, 하이퍼바이저에 추가 패킷 분배를 요청했던 가상머신이 이전에 요청된 추가 패킷 분배의 종료를 요청하거나 추가 패킷 분배를 요청했던 가상머신이 종료될 때 이를 탐지하고, 패킷 분배 룰 삭제를 위한 정보를 룰 관리부에 통지하여 패킷 분배 룰 삭제를 요청할 수 있다.

[0016] 이벤트 탐지부는 상이한 기능을 하는 워커 스레드들로 구성된 데이터 플레인 개발 키트(DPDK) 환경에서, 소정의 워커 스레드 그룹에 속한 워커 스레드들이 모두 종료될 때 이를 탐지하고, 패킷 분배 룰 삭제를 위한 정보를 룰 관리부에 통지하여 패킷 분배 룰 삭제를 요청할 수 있다.

[0017] 룰 관리부는 프로세스 또는 네트워크 응용이 소켓을 생성할 때 이벤트 탐지부로부터 통지된 목적지 포트 번호 및 프로토콜 타입에 기초하여 패킷 분배 룰을 생성하고, 프로세스 또는 네트워크 응용이 스레드를 생성할 때 이벤트 탐지부로부터 통지된 목적지 및 송신지 IP 주소, 목적지 및 송신지 포트 번호, 프로토콜 타입에 기초하여 패킷 분배 룰을 생성할 수 있다.

[0018] 룰 관리부는 프로세스 또는 네트워크 응용이 실행되는 코어가 변경될 때, 관련된 패킷 분배 룰의 코어 정보를 갱신하고, 프로세스 또는 네트워크 응용이 스레드 또는 소켓을 종료할 때, 관련된 패킷 분배 룰을 삭제할 수 있다.

[0019] 룰 관리부는 생성 또는 갱신된 패킷 분배 룰을 플로우 테이블에 저장하고, 삭제가 필요한 패킷 분배 룰을 플로우 테이블에서 삭제할 수 있다.

[0020] 룰 관리부는 컨테이너 기반 가상화 환경에서 하이퍼바이저가 컨테이너를 생성하고 포트 포워딩을 통해 특정 포트 번호를 컨테이너에 할당할 때 패킷 분배 룰 생성을 위한 포트 포워딩 정보와 포트 포워딩이 적용되는 컨테이너가 실행되는 코어 정보를 이벤트 탐지부로부터 통지받아 패킷 분배 룰을 생성할 수 있다.

[0021] 룰 관리부는 가상머신 기반 가상화 환경에서 가상머신이 하이퍼바이저에 추가 패킷 분류를 요청할 때 패킷 분배 룰 생성을 위한 목적지와 송신지 IP 주소, 목적지와 송신지 포트 번호, 프로토콜 타입 및 가상머신 정보를 이벤트 탐지부로부터 통지받아 패킷 분배 룰을 생성할 수 있다.

[0022] 룰 관리부는 상이한 기능을 하는 워커 스레드들로 구성된 데이터 플레인 개발 키트(DPDK) 환경에서, 소정의 워커 스레드 그룹에 속한 워커 스레드들이 모두 종료될 때 패킷 분배 룰 삭제를 위한 정보를 이벤트 탐지부로부터 통지받아 패킷 분배 룰을 삭제할 수 있다.

[0023] 다른 실시 예에 따른 병렬 패킷 처리를 위한 패킷 분배 방법은, 패킷을 처리할 프로세스 또는 네트워크 응용이 발생시키는 이벤트를 탐지하는 단계와, 탐지한 이벤트와 관련된 패킷 분배 룰을 생성, 갱신 또는 삭제하는 단계와, 생성, 갱신 또는 삭제된 패킷 분배 룰에 따라 네트워크 인터페이스 카드가 패킷을 분배하도록 하는 단계를 포함한다.

[0024] 프로세스는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스일 수 있다. 이벤트는 소켓 또는 스레드의 생성이나 종료, 소켓 또는 스레드를 생성한 프로세스나 네트워크 응용의 코어 마이그레이션에 관련된 이벤트일 수 있다.

발명의 효과

[0025] 응용 affinity를 달성하기 위한 기술로서, NIC에서 처리하는 송신 패킷을 검사함으로써 특정 플로우를 처리하는 네트워크 응용이 어느 코어에서 실행되는지에 대한 정보를 획득하는 방식이 있다. 해당 방식은 FD는 TCP처럼 수신된 패킷에 대해 ACK와 같은 대응되는 송신 패킷을 발생시키는 경우에만 적용할 수 있다는 단점이 있다. 또한, 특정 플로우를 처리하는 네트워크 응용이 실행되는 코어가 변경되는 경우 한 플로우에 속하는 패킷들이 하나 이상의 수신 큐에 분배되어 패킷의 처리 순서가 뒤바뀔 수 있는데, 전술한 방식은 이를 해결하기 위한 별다른 조치가 없다.

[0026] ARFS(Accelerated Receive Flow Steering) 방식은, 리눅스 커널은 패킷 송수신 과정에서 해당 패킷을 처리하는 네트워크 응용이 어느 코어에서 실행되는지에 대한 정보를 획득하여 NIC에 제공하고, NIC는 제공된 패킷 분배 룰에 따라 패킷들을 수신 큐에 배정하는 방법이다. ARFS는 리눅스 커널이 모든 송수신 패킷에 대해서 별도의 처리 과정을 거치기 때문에, 처리해야 할 패킷 수가 많은 환경에서는 이러한 부하로 말미암아 오히려 병렬 패킷 처리 성능이 감소할 수 있다.

[0027] 그러나 일 실시 예에 따르면, TCP 패킷만을 지원하거나 시스템 소프트웨어에 큰 부하를 발생시키는 전술한 문제들을 개선하여 모든 패킷(TCP, UDP 등)에 대해서 시스템 소프트웨어의 성능저하 없이 고성능으로 응용 affinity를 보장할 수 있다. 특히, 패킷을 처리할 프로세스 또는 네트워크 응용과 관련된 이벤트(소켓 생성, 스레드 생성, 마이그레이션 등)가 발생하는 경우에만 시스템 소프트웨어가 NIC에서의 패킷 분배를 위한 패킷 분배 룰을 생성, 갱신 및 삭제함으로써 전술한 문제들을 해결할 수 있다.

도면의 간단한 설명

[0028] 도 1은 본 발명의 일 실시 예에 따른 병렬 패킷 처리를 위한 패킷 분배 관리장치의 구성도,

도 2는 본 발명의 일 실시 예에 따른 패킷 분배 관리장치의 세부 구성도,

도 3a 내지 도 3d는 본 발명의 일 실시 예에 따른 단일 스레드(single thread) 기반 네트워크 응용에서의 패킷 분배 룰 생성 및 패킷 분배 프로세스를 도시한 흐름도,

도 4a 내지 도 4d는 본 발명의 일 실시 예에 따른 멀티 스레드 기반 네트워크 응용에서의 패킷 분배 룰 생성 및 패킷 분배 프로세스를 도시한 흐름도,

도 5a 내지 도 5d는 본 발명의 일 실시 예에 따른 컨테이너(container) 기반 가상화 환경에서의 패킷 분배 룰 생성 및 적용 프로세스를 도시한 흐름도,

도 6a 내지 도 6d는 본 발명의 일 실시 예에 따른 VM(virtual machine) 가상화 환경에서의 패킷 분배 룰 적용 및 패킷 분배 결과 제공 프로세스를 도시한 흐름도,

도 7a 내지 도 7d는 본 발명의 일 실시 예에 따른 서로 상이한 기능을 하는 워커 스레드들이 구현된 데이터 플레인 개발 키트(data plane development kit: DPDK) 환경에서의 패킷 분배 룰 생성 및 적용 예를 도시한 흐름도,

도 8a 내지 도 8d는 본 발명의 일 실시 예에 따른 프로세스 마이그레이션(migration) 발생 시 패킷 분배 룰 갱신 프로세스를 도시한 참조도이다.

발명을 실시하기 위한 구체적인 내용

[0029] 본 발명의 이점 및 특징, 그리고 그것들을 달성하는 방법은 첨부되는 도면과 함께 상세하게 후술되어 있는 실시 예들을 참조하면 명확해질 것이다. 그러나 본 발명은 이하에서 개시되는 실시 예들에 한정되는 것이 아니라 서로 다른 다양한 형태로 구현될 수 있으며, 단지 본 실시 예들은 본 발명의 개시가 완전하도록 하고, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에 발명의 범주를 완전하게 알려주기 위해 제공되는 것이며, 본 발명은 청구항의 범주에 의해 정의될 뿐이다. 명세서 전체에 걸쳐 동일 참조 부호는 동일 구성 요소를 지칭한다.

[0030] 본 발명의 실시 예들을 설명함에 있어서 공지 기능 또는 구성에 대한 구체적인 설명이 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명을 생략할 것이며, 후술되는 용어들은 본 발명의 실시 예에서의 기능을 고려하여 정의된 용어들로서 이는 사용자, 운용자의 의도 또는 관례 등에 따라 달라질 수 있다. 그러므로 그 정의는 본 명세서 전반에 걸친 내용을 토대로 내려져야 할 것이다.

[0031] 첨부된 블록도의 각 블록과 흐름도의 각 단계의 조합들은 컴퓨터 프로그램 인스트럭션들(실행 엔진)에 의해 수행될 수도 있으며, 이들 컴퓨터 프로그램 인스트럭션들은 범용 컴퓨터, 특수용 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비의 프로세서에 탑재될 수 있으므로, 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비의 프로세서를 통해 수행되는 그 인스트럭션들이 블록도의 각 블록 또는 흐름도의 각 단계에서 설명된 기능들을 수행하는 수단을 생성하게 된다.

[0032] 이들 컴퓨터 프로그램 인스트럭션들은 특정 방식으로 기능을 구현하기 위해 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비를 지향할 수 있는 컴퓨터 이용가능 또는 컴퓨터 판독 가능 메모리에 저장되는 것도 가능하므로, 그 컴퓨터 이용가능 또는 컴퓨터 판독 가능 메모리에 저장된 인스트럭션들은 블록도의 각 블록 또는 흐름도의 각 단계에서 설명된 기능을 수행하는 인스트럭션 수단을 내포하는 제조 품목을 생산하는 것도 가능하다.

[0033] 그리고 컴퓨터 프로그램 인스트럭션들은 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비 상에 탑재되는 것도 가능하므로, 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비 상에서 일련의 동작 단계들이 수행되어 컴퓨터로 실행되는 프로세스를 생성해서 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비를 수행하는 인스트럭션들은 블록도의 각 블록 및 흐름도의 각 단계에서 설명되는 기능들을 실행하기 위한 단계들을

제공하는 것도 가능하다.

[0034] 또한, 각 블록 또는 각 단계는 특정된 논리적 기능들을 실행하기 위한 하나 이상의 실행 가능한 인스트럭션들을 포함하는 모듈, 세그먼트 또는 코드의 일부를 나타낼 수 있으며, 몇 가지 대체 실시 예들에서는 블록들 또는 단계들에서 언급된 기능들이 순서를 벗어나서 발생하는 것도 가능함을 주목해야 한다. 예컨대, 잇달아 도시되어 있는 두 개의 블록들 또는 단계들은 사실 실질적으로 동시에 수행되는 것도 가능하며, 또한 그 블록들 또는 단계들이 필요에 따라 해당하는 기능의 역순으로 수행되는 것도 가능하다.

[0035] 이하, 첨부 도면을 참조하여 본 발명의 실시 예를 상세하게 설명한다. 그러나 다음에 예시하는 본 발명의 실시 예는 여러 가지 다른 형태로 변형될 수 있으며, 본 발명의 범위가 다음에 상술하는 실시 예에 한정되는 것은 아니다. 본 발명의 실시 예는 당업계에서 통상의 지식을 가진 자에 본 발명을 보다 완전하게 설명하기 위하여 제공된다.

[0036] 도 1은 본 발명의 일 실시 예에 따른 병렬 패킷 처리를 위한 패킷 분배 관리장치의 구성도이다.

[0037] 본 발명은, TCP 패킷만을 지원하거나 시스템 소프트웨어에 큰 부하를 발생시키는 문제를 개선하여, 모든 패킷(TCP, UDP 등)에 대해서 시스템 소프트웨어의 성능저하 없이 고성능으로 응용 affinity를 보장하는 것을 목표로 한다. 특히, 패킷을 처리할 프로세스 또는 네트워크 응용과 관련된 이벤트가 발생하는 경우에만 시스템 소프트웨어가 NIC에서의 패킷 분배를 위한 패킷 분배 룰 생성, 갱신 및 삭제함으로써 전술한 문제를 해결하고자 한다.

[0038] 본 발명은 패킷 기반으로 패킷 분배 룰을 생성하는 기술과는 달리, 이벤트 기반으로 패킷 분배 룰을 생성하는 방법이다. 여기서, 이벤트는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스에 의해 발생하는 이벤트로써, NIC에서 패킷을 코어로 분배하는데 사용되는 패킷 분배 룰에 영향을 주는 이벤트들이다. 예를 들어, 이벤트들은 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스의 소켓 및 스레드 생성 및 종료, 소켓 및 스레드를 생성한 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스의 코어 이동 등을 포함할 수 있다.

[0039] 이하, 도 1을 참조로 하여 이벤트 기반 패킷 분배 관리장치의 구성에 대해 설명한다.

[0040] 도 1을 참조하면, 패킷 분배 관리장치(1)는 이벤트 기반 룰 관리자(event-driven rule manager)(20)를 포함한다.

[0041] 이벤트 기반 룰 관리자(20)는 시스템 소프트웨어 내 또는 사용자 영역에서 동작할 수 있다. 이벤트 기반 룰 관리자(20)는 프로세스(30)가 발생시키는 이벤트를 탐지하여, 이벤트와 관련된 패킷 분배 룰을 생성, 갱신, 삭제하여 플로우 테이블(10)에 저장한다. 프로세스(30)는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스일 수 있다. 이벤트는 NIC에서의 패킷 분배를 위해 사용되는 패킷 분배 룰 생성, 갱신, 삭제에 영향을 주는 소켓 생성, 스레드 생성, 마이그레이션 등일 수 있다.

[0042] 플로우 테이블(flow table)(10)에는 수신되는 패킷을 NIC의 수신 큐를 통해 코어들에 분배하는데 사용되는 패킷 분배 룰이 저장된다. 플로우 테이블(10)은 NIC에 위치할 수 있다. 패킷 분배 룰을 생성, 갱신, 삭제하는 이벤트 기반 룰 관리자(20)가 패킷 분배 룰과 관련된 이벤트를 감지하여 패킷 분배 룰을 직접적으로 또는 간접적으로 변경할 수 있다. 일 실시 예로, 플로우 테이블(10)은 디바이스 드라이버를 통해 이벤트 기반 룰 관리자(20)에 의해 변경될 수 있다. NIC는 수신되는 패킷에 대해 플로우 테이블(10)을 참조하여 어느 수신 큐로 배정할지 결정할 수 있다.

[0043] 이벤트 기반 룰 관리자(20)는 패킷 분배 룰 생성, 갱신, 삭제에 영향을 주는 이벤트를 탐지하여 발생한 이벤트와 관련된 패킷 분배 룰을 생성, 갱신, 삭제할 수 있다. 이벤트는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 발생시킬 수 있다.

[0044] 예를 들어, 이벤트 기반 룰 관리자(20)는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 소켓을 생성할 때 이를 탐지하고, 생성되는 소켓 관련 정보 및 해당 프로세스가 실행되는 코어 정보에 기초하여 패킷 분배 룰을 생성하여 패킷 분배 룰을 플로우 테이블(10)에 등록할 수 있다.

[0045] 다른 예로, 이벤트 기반 룰 관리자(20)는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 기존에 생성된 소켓에 대한 스레드를 생성할 때 이를 탐지하고, 생성되는 스레드 관련 정보 및 해당 스레드가 실행되는 코어 정보에 기초하여 패킷 분배 룰을 생성하여 패킷 분배 룰을 플로우 테이블(10)에 등록 할 수 있다.

[0046] 또 다른 예로, 이벤트 기반 룰 관리자(20)는 패킷 분배 룰이 생성된 시스템 소프트웨어 내의 프로세스 또는 사

용자 영역의 프로세스가 시스템 소프트웨어의 스케줄러에 의해 현재 실행 중인 코어에서 다른 코어로 이동될 때, 이를 탐지하고 관련된 패킷 분배 룰을 플로우 테이블(10)에서 갱신할 수 있다.

[0047] 또 다른 예로, 이벤트 기반 룰 관리자(20)는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 미리 생성된 스레드 또는 소켓을 종료할 때 이를 탐지하고 관련된 패킷 분배 룰을 플로우 테이블(10)에서 삭제할 수 있다.

[0048] 도 2는 본 발명의 일 실시 예에 따른 패킷 분배 관리장치의 세부 구성도이다.

[0049] 한 개 이상의 수신 큐(receive queue)(12)를 지원하는 NIC와 한 개 이상의 코어를 지원하는 시스템 소프트웨어(운영체제 또는 하이퍼바이저)를 통해 병렬 패킷 처리를 구현할 수 있다. 즉, 하나의 수신 큐(12)는 하나의 코어에 매핑이 되며, 하나의 수신 큐(12)는 자기와 매핑된 코어에 패킷이 수신되었음을 알리는 독립적인 인터럽트(interrupt)를 발생시킬 수 있다. 따라서, NIC는 다수의 수신 큐(12)에 패킷을 배정하고 코어들은 자기에 매핑된 수신 큐(12)로부터 동시에 패킷들을 수신하여 병렬로 패킷을 처리할 수 있다.

[0050] 도 1 및 도 2를 참조하면, 이벤트 기반 룰 관리자(20)는 룰 관리부(rule manager)(200) 및 이벤트 탐지부(event detector)(210)를 포함한다. NIC는 플로우 테이블(10)과 수신 큐(12)를 포함한다.

[0051] 룰 관리부(200)는 이벤트 탐지부(210)를 통해 탐지된 이벤트와 관련된 패킷 분배 룰을 생성, 갱신 또는 삭제한다.

[0052] 플로우 테이블(10)에는 패킷 분배 룰이 저장된다. 플로우 테이블(10)은 NIC에 위치할 수 있다. 패킷 분배 룰은 룰 관리부(200)에 의해 직접적으로 또는 간접적으로 변경될 수 있다. 패킷 분배 룰은 디바이스 드라이버를 통해 룰 관리부(200)에 의해 변경될 수 있다. NIC는 수신되는 패킷에 대해 플로우 테이블(10)을 참조하여 어느 수신 큐(12)로 배정할지 결정할 수 있다.

[0053] 이벤트 탐지부(210)는 NIC에서 사용할 패킷 분배 룰에 영향을 미치는 이벤트들을 탐지한다. 룰 관리부(200)는 이벤트 탐지부(210)를 통해 탐지된 이벤트와 관련된 패킷 분배 룰을 생성, 갱신 또는 삭제한다.

[0054] 이벤트 탐지부(210)는 프로세스(30) 또는 네트워크 응용(40)이 발생시키는 패킷 분배 룰 생성, 갱신 또는 삭제와 관련된 이벤트를 탐지할 수 있다. 프로세스(30)는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스일 수 있다. 이벤트 탐지부(210)는 이벤트를 탐지했을 때, 이벤트가 발생했다는 것과 해당 이벤트와 관련된 정보를 룰 관리부(200)에 통지할 수 있다.

[0055] 예를 들어, 이벤트 탐지부(210)는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 소켓을 생성할 때 이를 탐지하고, 패킷 분배 룰 생성을 위한 소켓의 포트 번호 및 프로토콜 타입과 해당 프로세스가 실행되는 코어 정보를 룰 관리부(200)에 통지할 수 있다.

[0056] 다른 예로, 이벤트 탐지부(210)는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 기존에 생성된 소켓에 대한 스레드를 생성할 때, 이를 탐지하고, 패킷 분배 룰 생성을 위한 목적지/송신지 IP 주소, 목적지/송신지 포트 번호, 프로토콜 타입과 해당 스레드가 실행되는 코어 정보를 룰 관리부(200)에 통지할 수 있다.

[0057] 또 다른 예로, 이벤트 탐지부(210)는 패킷 분배 룰이 생성된 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 시스템 소프트웨어의 스케줄러에 의해 현재 실행 중인 코어에서 다른 코어로 이동될 때, 이를 탐지하고 갱신될 패킷 분배 룰과 관련된 소켓 또는 스레드 정보 및 새로운 코어 정보를 룰 관리부(200)에 통지할 수 있다.

[0058] 또 다른 예로, 이벤트 탐지부(210)는 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 스레드 또는 소켓을 종료할 때, 이를 탐지하고 관련된 패킷 분배 룰 삭제를 위한 정보를 룰 관리부(200)에 통지할 수 있다.

[0059] 룰 관리부(200)는 이벤트 탐지부(210)가 제공하는 정보에 기초하여 패킷 분배 룰을 생성, 갱신 또는 삭제할 수 있다.

[0060] 예를 들어, 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 소켓을 생성할 때, 룰 관리부(200)는 목적지 포트 번호 및 프로토콜 타입에 기초하여 패킷 분배 룰을 생성할 수 있다.

[0061] 다른 예로, 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 스레드를 생성할 때, 룰 관리부(200)는 목적지/송신지 IP 주소, 목적지/송신지 포트 번호, 프로토콜 타입에 기초하여 패킷 분배 룰을 생성할

수 있다.

[0062] 또 다른 예로, 패킷 분배 룰이 생성된 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 실행되는 코어가 변경될 때, 룰 관리부(200)는 관련된 패킷 분배 룰의 코어 정보를 갱신할 수 있다.

[0063] 또 다른 예로, 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 스레드 또는 소켓을 종료할 때, 룰 관리부(200)는 관련된 패킷 분배 룰을 삭제할 수 있다.

[0064] 룰 관리부(200)는 생성 또는 갱신된 패킷 분배 룰을 플로우 테이블(10)에 저장할 수 있다. 룰 관리부(200)는 패킷 분배 룰 삭제가 필요할 경우 해당 패킷 분배 룰을 플로우 테이블(10)에서 삭제할 수 있다.

[0065] 본 발명에서 제안하는 방법의 핵심은 패킷을 처리할 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 NIC에서 사용될 패킷 분배 룰에 영향을 주는 이벤트를 발생시켰을 때 이를 탐지하여 관련된 패킷 분배 룰을 생성, 갱신 또는 삭제하여 NIC로 하여금 필요한 패킷 분배를 하도록 하는 것이다. 따라서, 본 발명에서 제안하는 방법은, 패킷 분배 룰 생성과 관련된 이벤트를 탐지하고 필요한 정보를 획득할 수 있는 시스템 소프트웨어 또는 사용자 영역에 적용될 수 있다. 이하 후술되는 도면들을 참조로 하여 구체적인 일 실시 예들에 대해 설명한다.

[0066] 도 3a 내지 도 3d는 본 발명의 일 실시 예에 따른 단일 스레드(single thread) 기반 네트워크 응용에서의 패킷 분배 룰 생성 및 패킷 분배 프로세스를 도시한 흐름도이다.

[0067] 도 3a 내지 도 3d를 참조하면, 본 발명은 리눅스 환경에서 단일 스레드로 실행되는 사용자 영역의 네트워크 응용에 적용될 수 있다. 예를 들어, 코어 0에서 실행되는 네트워크 응용이 7777번 TCP 포트에 매핑되는 소켓 생성을 리눅스 커널에 요청하는 경우에, 이벤트 탐지부(210)가 이를 탐지한다(도 3a). 이벤트 탐지부(210)는 소켓 생성 요청이 발생했다는 것(socket creation event)과, 생성되는 소켓 정보(TCP, 777)와, 소켓 생성을 요청한 네트워크 응용이 실행되는 코어 정보(core 0)를 룰 관리부(200)에 통지한다(도 3b). 룰 관리부(200)는 이벤트 탐지부(210)로부터 제공된 정보에 기초하여 패킷 분배 룰(TCP, 7777, core0)을 생성하여 NIC의 플로우 테이블(10)에 저장한다(도 3c). NIC는 패킷 분배 룰에 따라 패킷을 수신 큐 버퍼에 분배한다(도 3d).

[0068] NIC는 패킷을 분배할 수신 큐를 결정할 때, 플로우 테이블(10) 내의 2-tuple(목적지 포트 번호, 프로토콜 타입) 기반 룰을 검사하여 주어진 패킷의 패킷 분배 룰을 찾을 수 있다. 필요한 패킷 분배 룰을 찾지 못하였다면, 이는 해당 패킷이 속할 소켓을 생성한 네트워크 응용이 없다는 의미이기 때문에 해당 패킷을 폐기(drop)할 수 있다.

[0069] 도 3a 내지 도 3d를 참조로 하여 전술한 실시 예는 내용 전달의 편의성 및 명확성을 위해 하나의 네트워크 응용을 고려하였으나, 다수의 네트워크 응용에도 동일하게 적용될 수 있다. 또한, 사용자 영역에서 동작하는 프로세스를 고려하였으나, 시스템 소프트웨어 내에서 동작하는 프로세스에도 동일하게 적용될 수 있다. 또한, 시스템 소프트웨어 내에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)를 고려하였으나, 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다. 2-tuple(목적지 포트 번호, 프로토콜 타입)에 기반한 룰을 고려하였으나, 목적지 IP 주소를 포함한 3-tuple(목적지 IP 주소, 목적지 포트 번호, 프로토콜 타입)에 기반한 룰에도 동일하게 적용될 수 있다.

[0070] 도 4a 내지 도 4d는 본 발명의 일 실시 예에 따른 멀티 스레드 기반 네트워크 응용에서의 패킷 분배 룰 생성 및 패킷 분배 프로세스를 도시한 흐름도이다.

[0071] 도 4a 내지 도 4d를 참조하면, 본 발명은 리눅스 환경에서, 멀티 스레드 기반으로 실행되는 사용자 영역의 네트워크 응용에 적용될 수 있다. 네트워크 응용이 멀티 스레드로 동작하는 경우, 룰 관리부(200)는 소켓 생성시 생성된 2-tuple에 기반한 룰에 추가로 4-tuple(송신지 IP 주소, 목적지/송신지 포트 번호, 프로토콜 타입)에 기반한 룰을 생성하여 이를 NIC에 제공할 수 있다. 멀티 스레드 네트워크 응용은 각 TCP 세션을 처리하기 위한 하나의 스레드를 생성할 수 있다. 이때, 이벤트 탐지부(210)는 멀티 스레드 네트워크 응용의 스레드 생성 요청 이벤트를 탐지한다(도 4a). 이벤트 탐지부(210)는 스레드 생성 요청이 발생했다는 것(thread_creation event)과, 생성된 스레드 정보(dst_port, src_IP, src_port, IP)와, 생성된 스레드가 실행되는 코어 정보(C)를 룰 관리부(200)에 통지한다(도 4b). 룰 관리부(200)는 이벤트 탐지부(210)로부터 제공된 정보에 기초하여 패킷 분배 룰을 생성하여 NIC의 플로우 테이블(10)에 저장한다(도 4c). NIC는 패킷 분배 룰에 따라 패킷을 수신 버퍼 큐에 분배한다(도 4d).

[0072] NIC는 패킷을 분배할 수신 큐를 결정할 때, 플로우 테이블(10) 내의 4-tuple 기반 룰, 2-tuple 기반 룰 순으로

검사하여 패킷의 패킷 분배 룰을 찾을 수 있다. 4-tuple 룰이 없거나 4-tuple 룰에서 필요한 정보를 획득하지 못하였다면, 2-tuple 룰을 검사한다. 2-tuple 룰에서도 필요한 정보를 획득하지 못하였다면, 이는 해당 패킷이 속할 소켓을 생성한 네트워크 응용이 없다는 의미이기 때문에 해당 패킷을 폐기(drop)할 수 있다.

[0073] 전술한 실시 예는 내용 전달의 편의성 및 명확성을 위해 하나의 네트워크 응용이 스레드를 생성하는 환경을 고려하였으나, 다수의 네트워크 응용이 스레드를 생성하는 환경에도 적용될 수 있다. 또한, 사용자 영역에서 동작하는 프로세스를 고려하였으나, 시스템 소프트웨어 내에서 동작하는 프로세스에도 동일하게 적용될 수 있다. 또한, 시스템 소프트웨어 내에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)를 고려하였으나, 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다. 4-tuple(송신지 IP 주소, 목적지/송신지 포트 번호, 프로토콜 타입)에 기반한 룰을 고려하였으나, 목적지 IP 주소를 포함한 5-tuple(목적지/송신지 IP 주소, 목적지/송신지 포트 번호, 프로토콜 타입)에 기반한 룰에도 동일하게 적용될 수 있다.

[0074] 도 5a 내지 도 5d는 본 발명의 일 실시 예에 따른 컨테이너(container) 기반 가상화 환경에서의 패킷 분배 룰 생성 및 적용 프로세스를 도시한 흐름도이다.

[0075] 도 5a 내지 도 5d를 참조하면, 본 발명은 컨테이너 기반 가상화 환경에 적용할 수 있다. 컨테이너는 프로그램이 작동하기 위한 최소한의 요소들을 묶어 패키징한 것으로 독립적인 운영체제보다 작으면서 독립적인 배포와 실행을 가능하게 하는 일종의 가상 머신이다. 컨테이너 기반 가상화는 리눅스 기반으로 동작하며, 리눅스 커널 입장에서 각 컨테이너는 스케줄링이 가능한 프로세스로 인식이 된다.

[0076] 컨테이너 기반 가상화를 제공하는 하이퍼바이저인 Docker(다커)는 생성된 각 컨테이너에 포트 포워딩(port forwarding)을 통해서 외부와의 네트워크 연결을 제공할 수 있다. 일 실시 예로, 컨테이너 1에서 8080 포트를 사용하여 동작하는 응용으로부터 외부로 나가는 트래픽은 리눅스의 8888 포트를 통해서 나가도록 할 수 있고, 8888 포트를 통해서 외부로부터 유입되는 트래픽은 컨테이너 1의 8080 포트를 사용하여 동작하는 응용으로 유입되도록 할 수 있다.

[0077] 도 5a 내지 도 5d를 참조하면, 컨테이너 기반 가상화 환경에서 Docker와 같은 하이퍼바이저가 컨테이너를 생성하고 포트 포워딩을 통해 특정 포트 번호를 컨테이너에 할당할 수 있다(도 5a). 이벤트 탐지부(210)는 포트 포워딩이 발생했다는 것(Port forwarding event)과 포트 포워딩 정보(TCP, 8888) 및 포트 포워딩이 적용되는 컨테이너가 실행되는 코어 정보(C3)를 룰 관리부(200)에 통지할 수 있다(도 5b). 룰 관리부(200)는 제공된 정보에 기초하여 패킷 분배 룰을 생성하여 NIC의 플로우 테이블(10)에 저장할 수 있다(도 5c). NIC는 패킷 분배 룰에 따라 패킷을 수신 큐 버퍼에 분배할 수 있다 (도 5d).

[0078] 전술한 실시 예는 내용 전달의 편의성 및 명확성을 위해 하나의 컨테이너 및 하나의 포트 포워딩을 고려하였으나, 다수의 컨테이너 및 다수의 포트 포워딩에 적용될 수 있다. 또한, 시스템 소프트웨어 내에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)를 고려하였으나, 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0079] 도 6a 내지 도 6d는 본 발명의 일 실시 예에 따른 VM(virtual machine) 가상화 환경에서의 패킷 분배 룰 적용 및 패킷 분배 결과 제공 프로세스를 도시한 흐름도이다.

[0080] 도 6a 내지 도 6d를 참조하면, 본 발명은 VM 기반 가상화 환경에서 적용될 수 있다. VM 가상화 환경에서 가상화를 제어 및 관리하는 주체인 하이퍼바이저는 NIC로부터 수신한 패킷의 헤더를 확인하여 해당 VM에 제공한다. 하이퍼바이저는 베어 메탈(bare-metal) 형태로 적용될 수 있고, 특정 운영 체제 위에서 동작할 수도 있다. 하지만, 소프트웨어로 구현되는 하이퍼바이저는 성능 병목현상(performance bottleneck)이 될 수 있다.

[0081] 이를 해결하기 위해서 제안된 방법이 가상머신 디바이스 큐(virtual machine device queue: VMDq, 이하 VMDq라 칭함)와 싱글 루트 I/O 가상화(single root I/O virtualization: SR-IOV, 이하 SR-IOV라 칭함)이다. VMDq나 SR-IOV는 NIC에서 패킷을 VM 별로 분류하여 하이퍼바이저의 별다른 관여 없이 패킷들을 VM에 바로 제공하는 기술이다.

[0082] 하지만, VMDq나 SR-IOV에서, NIC는 VM 수준의 분류만 한다. 다시 말해, VM 내에서 필요한 패킷 분류가 있다면, 소프트웨어에 기반하여 VM 내부에서 분류를 해야 한다는 말이다. 예를 들어, VM 내에서 응용 affinity를 보장하기 위해서는 특정 플로우에 속한 패킷을 가상 NIC의 어느 수신 큐에 배정해야 하는지 결정해야 한다. 이를 위해서 VM 내부적으로 전술한 방법들과 같은 방법을 적용할 필요가 있다. 하지만, VM 내에서의 소프트웨어로 구현되는 이러한 방법은 패킷 처리 성능 저하를 가져올 수 있다. 이런 환경에서, VM 내에서의 병렬 패킷 처리 성능 향

상을 위해서 본 발명에서 제안된 방법이 적용될 수 있다.

[0083] 도 6a 내지 도 6d를 참조하면, VM은 VM 시작시 또는 특정 응용이 소켓 및 스레드를 생성하는 등의 특정 이벤트 발생시, 수신된 패킷들에 적용할 분류 종류와 관련된 정보를 하이퍼바이저에 제공하고 추가 패킷 분류를 요청할 수 있다(도 6a). VM이 하이퍼바이저에 제공하는 정보는, 전술한 내용에 언급된 것처럼 시스템 소프트웨어가 NIC에 제공하는, 소켓 및 스레드 관련 정보 및 코어 정보를 포함할 수 있다. 하이퍼바이저 내의 이벤트 탐지부(210)는 이런 요청을 탐지하고 관련된 정보를 룰 관리부(200)에 통지할 수 있다(도 6b). 룰 관리부(200)는 추가 패킷 분배를 위한 패킷 분배 룰을 생성하여 NIC 플로우 테이블에 저장할 수 있다(도 6c). NIC는 제공된 패킷 분배 룰에 따라 패킷을 수신 큐에 분배할 수 있다(도 6d).

[0084] 다시 말해, NIC는 VMDq나 SR-IOV에서 하는 것처럼, 주어진 패킷을 1차적으로 VM 별로 분류한 후에 2차적으로 필요시 VM 별로 요구된 추가적인 분류를 적용할 수 있다. NIC는 VM 별로 패킷을 분류한 후 패킷과 함께 분류된 결과를 VM에 함께 제공할 수 있다(도 6d). VM은 NIC에서 제공된 정보에 기초하여 필요한 패킷 분류를 수월하게 진행할 수 있다.

[0085] 전술한 실시 예는 내용 전달의 편의성 및 명확성을 위해 하나의 VM의 추가 패킷 분류 요청을 고려하였으나, 다수의 VM의 추가 패킷 분류 요청에 적용될 수 있다. 또한, 시스템 소프트웨어 내에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)를 고려하였으나, 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0086] 도 7a 내지 도 7d는 본 발명의 일 실시 예에 따른 서로 상이한 기능을 하는 워커 스레드들이 구현된 데이터 플레인 개발 키트(data plane development kit: DPDK, 이하 DPDK라 칭함) 환경에서의 패킷 분배 룰 생성 및 적용 예를 도시한 흐름도이다.

[0087] 도 7a 내지 도 7d를 참조하면, 본 발명은 DPDK를 기반으로 구현된 네트워크 응용의 패킷 처리 성능을 향상시키기 위해 적용될 수 있다. DPDK에는 폴링(polling) 방식으로 NIC로부터 패킷을 수신하는 역할을 하는 패킷 I/O 수신 스레드(packet I/O RX thread)와 폴링 방식으로 획득한 패킷을 동일한 기능을 하는 워커(worker) 스레드들에 분배하는 패킷 분배기 스레드(packet distributor thread)가 존재한다. 특정한 기능을 제공하는 네트워크 응용을 DPDK에 기반해서 구현하는 방법은, 해당 기능을 워커 스레드를 통해 구현하는 것이다. 다시 말해, NIC로부터 패킷을 가져와서 워커 스레드에 분배하는 것은 DPDK의 몫이며, 워커 스레드의 기능을 구현하는 것은 사용자의 몫이다.

[0088] 현재의 DPDK 구조는 동일한 기능을 하는 워커 스레드만을 지원한다. 왜냐하면, NIC로부터 수신된 패킷을 워커 스레드에 분배하는 패킷 분배기 스레드는 패킷 헤더에 기반한 32비트 해시 값만을 사용할 수 있기 때문이다. 따라서, 만일 상이한 기능을 하는 워커 스레드가 존재하는 경우에서는, 주어진 패킷을 어느 워커 스레드로 배정해야 할지 결정할 수 없다. 이러한 DPDK의 기능 및 성능을 개선하는 한 가지 방법은, 상이한 기능을 하는 다수의 워커 스레드를 지원하면서 응용 affinity를 지원하는 것이다.

[0089] 도 7a 내지 도 7d는 상이한 기능을 하는 워커 스레드들이 구현된 DPDK 환경에서의 패킷 분배 룰 생성 일 실시 예를 보여준다. 본 발명에서 제안하는 방법에 따르면, DPDK가 동작하는 리눅스 커널이 필요한 정보를 획득하여 NIC에 내려주고, NIC는 주어진 정보에 기초하여 패킷을 분배하고, DPDK의 패킷 분배기 스레드가 NIC의 패킷 분배 정보를 활용하여 패킷을 워커 스레드에 분배하도록 할 수 있다.

[0090] 동일한 기능을 하는 워커 스레드들은 하나의 워커 스레드 그룹(Worker thread Group: WTG)을 구성할 수 있다(도 7a). 일 실시 예로, 특정 포트를 기반으로 동작하는 네트워크 응용을 구현하는 다수의 워커 스레드들은 특정 포트 기반 스레드 그룹을 구성할 수 있고, 주어진 모든 패킷에 대해서 DPI를 실행하는 워커 스레드들은 DPI 워커 스레드 그룹을 구성할 수 있다. 이외에도 다양한 기준을 기반으로 동일 기능을 하는 워커 스레드들은 하나의 워커 스레드 그룹을 구성할 수 있다.

[0091] 동일한 기능을 하는 워커 스레드 그룹은 해당 워커 스레드 그룹을 식별하는데 필요한 정보를 리눅스 커널 또는 DPDK에 제공할 수 있다(도 7a). 워커 스레드 그룹을 식별하는데 사용될 수 있는 정보는 목적지 MAC 주소/IP 주소/포트 번호 또는 송신지 MAC 주소/IP 주소/포트 번호 등이 될 수 있다. 이외에도 워커 스레드 그룹을 유일하게 식별할 수 있는 다양한 정보들이 사용될 수 있다.

[0092] 이벤트 탐지부(210)는 워커 스레드 그룹 식별자 등록 이벤트를 탐지할 수 있고 관련된 정보를 획득하여 룰 관리부(200)에 제공할 수 있다(도 7b). 룰 관리부(200)는 제공된 정보에 기초하여 패킷들을 워커 스레드 그룹 단위로 구분할 수 있는 패킷 분배 룰을 생성할 수 있다. 룰 관리부(200)는 또한 패킷을 동일 워커 스레드 그룹 내에

서 특정 워커 스레드에 분배할 수 있는 패킷 분배 룰을 생성할 수 있다. 룰 관리부(200)는 생성된 패킷 분배 룰을 NIC의 플로우 테이블(10)에 저장할 수 있다(도 7c). NIC는 패킷을 주어진 패킷 분배 룰에 따라 분배할 수 있고 분배 결과를 패킷과 함께 DPDK에 제공할 수 있다. DPDK 패킷 분배기 스레드는 NIC가 제공한 정보에 기초하여 주어진 패킷을 어느 워커 스레드 그룹 내의 어느 워커 스레드에 분배할지 결정할 수 있다(도 7d).

[0093] 전술한 실시 예는 내용 전달의 편의성 및 명확성을 위해 시스템 소프트웨어 내에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)를 고려하였으나, 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0094] 도 8a 내지 도 8d는 본 발명의 일 실시 예에 따른 프로세스 마이그레이션(migration) 발생 시 패킷 분배 룰 갱신 프로세스를 도시한 참조도이다.

[0095] 도 8a 내지 도 8d를 참조하면, 시스템 소프트웨어의 스케줄러는 실행 가능한 프로세스들을 어느 코어에서 얼마 동안 실행시킬 것인지 결정한다. 시스템 소프트웨어의 스케줄러는 부하 분산, 성능 향상 등을 이유로 특정 프로세스를 현재 실행되고 있는 코어로부터 다른 코어로 이동시키기도 한다. 이러한 프로세스 마이그레이션이 발생할 경우에, 응용 affinity 보장을 위해 패킷 분배 룰을 갱신할 필요가 있다.

[0096] 기존에 생성된 패킷 분배 룰을 갱신할 필요가 있는 경우는 프로세스가 마이그레이션 될 때이기 때문에, 본 발명에서는 시스템 소프트웨어 스케줄러에 의해서 프로세스가 마이그레이션 될 때 패킷 분배 룰을 갱신한다.

[0097] 일 실시 예로, 리눅스 환경에서 사용자 영역에서 실행되는 네트워크 응용의 마이그레이션 발생시 패킷 분배 룰이 갱신될 수 있다. 리눅스 스케줄러가 코어 0에서 실행되고 있는 네트워크 응용(도 8a)을 코어 2로 마이그레이션하기로 결정했을 때 이벤트 탐지부(210)는 이를 탐지하고 관련된 정보를 룰 관리부(200)에 제공하여 마이그레이션되는 네트워크 응용과 관련된 패킷 분배 룰 갱신을 요청할 수 있다(도 8b). 룰 관리부(200)는 마이그레이션 대상이 되는 네트워크 응용과 관련된 2-tuple 및 4-tuple 기반 패킷 분배 룰의 코어 값을 갱신하여 NIC의 플로우 테이블에 저장할 수 있다(도 8c). NIC는 갱신된 패킷 분배 룰에 따라 패킷을 분배할 수 있다(도 8d).

[0098] 전술한 실시 예는 내용 전달의 편의성 및 명확성을 위해 하나의 네트워크 응용을 고려하였으나, 다수의 네트워크 응용에도 동일하게 적용될 수 있다. 또한, 사용자 영역에서 동작하는 프로세스를 고려하였으나, 시스템 소프트웨어 내에서 동작하는 프로세스에도 동일하게 적용될 수 있다. 또한, 시스템 소프트웨어 내에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)를 고려하였으나, 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0099] 전술한 실시 예는 네트워크 응용이 생성한 스레드에도 동일하게 적용될 수 있다. 이는 대부분의 시스템 소프트웨어에서 스케줄러는 스레드도 스케줄링이 필요한 프로세스로 인식하기 때문이다. 전술한 실시 예는 리눅스 기반으로 동작하는 컨테이너 기반 가상화 기법들에서도 적용될 수 있다. 이는 리눅스 커널의 스케줄러 입장에서 컨테이너들은 스케줄링이 필요한 프로세스들로 인식되기 때문이다.

[0100] 또 다른 일 실시 예로, VM 기반 가상화 환경에서, VM 내의 패킷 분배를 NIC가 지원하는 환경에서, VM 내의 패킷 분배 룰 변경이 발생한 경우에 VM은 하이퍼바이저에 갱신된 패킷 분배 룰을 제공할 수 있다. 하이퍼바이저 내의 이벤트 탐지부(210)는 이 요청을 탐지하고 관련된 정보를 룰 관리부(200)에 제공하여 패킷 분배 룰을 갱신하도록 요청할 수 있다. 갱신된 패킷 분배 룰은 NIC의 플로우 테이블 내에 저장될 수 있다. NIC는 갱신된 패킷 분배 룰을 적용한 결과를 해당 VM에 패킷과 함께 제공할 수 있다.

[0101] 전술한 실시 예는 내용 전달의 편의성 및 명확성을 위해 시스템 소프트웨어 내에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)를 고려하였으나, 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0102] 한편, 시스템 소프트웨어 내의 프로세스 또는 사용자 영역의 프로세스가 기존에 생성된 패킷 분배 룰의 삭제가 필요한 이벤트를 발생시켰을 때, 이를 탐지하고 관련된 패킷 분배 룰을 NIC의 플로우 테이블로부터 삭제할 수 있다.

[0103] 일 실시 예로, 사용자 영역에서 실행되는 단일 스레드 기반 네트워크 응용의 경우, 네트워크 응용이 기존에 생성된 소켓을 종료하는 이벤트 발생시, 시스템 소프트웨어 내에서 실행되는 이벤트 탐지부(210)는 이를 탐지하여 관련된 정보를 시스템 소프트웨어 내에서 실행되는 룰 관리부(200)에 제공할 수 있다. 룰 관리부(200)는 종료되는 소켓과 관련된 패킷 분배 룰을 NIC의 플로우 테이블로부터 삭제할 수 있다. 전술한 실시 예는 시스템 소프트웨어 내에서 동작하는 프로세스에도 동일하게 적용될 수 있다. 또한, 사용자 영역에서 동작하는 이벤트 탐지부

(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0104] 또 다른 일 실시 예로, 사용자 영역에서 실행되는 멀티 스레드 기반 네트워크 응용의 경우에, 네트워크 응용이 기존에 생성된 스레드를 종료하는 이벤트 발생시, 시스템 소프트웨어 내에서 실행되는 이벤트 탐지부(210)는 이를 탐지하여 관련된 정보를 시스템 소프트웨어 내에서 실행되는 룰 관리부(200)에 제공할 수 있다. 룰 관리부(200)는 종료되는 스레드와 관련된 패킷 분배 룰을 NIC의 플로우 테이블로부터 삭제할 수 있다. 전술한 실시 예는 시스템 소프트웨어 내에서 동작하는 프로세스에도 동일하게 적용될 수 있다. 또한, 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0105] 또 다른 일 실시 예로, 사용자 영역에서 실행되는 멀티 스레드 기반 네트워크 응용의 경우에, 네트워크 응용이 기존에 생성된 소켓을 종료하는 이벤트 발생시, 시스템 소프트웨어 내에서 실행되는 이벤트 탐지부(210)는 이를 탐지하여 관련된 정보를 시스템 소프트웨어 내에서 실행되는 룰 관리부(200)에 제공할 수 있다. 룰 관리부(200)는 종료되는 소켓과 관련된 패킷 분배 룰을 NIC의 플로우 테이블로부터 삭제할 수 있다. 전술한 실시 예는 시스템 소프트웨어 내에서 동작하는 프로세스에도 동일하게 적용될 수 있다. 또한, 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0106] 또 다른 일 실시 예로, 컨테이너 기반 가상화 환경에서, 특정 컨테이너에 대한 포트 포워딩을 종료할 때, 시스템 소프트웨어 내에서 실행되는 이벤트 탐지부(210)는 이를 탐지하여 관련된 정보를 시스템 소프트웨어 내에서 실행되는 룰 관리부(200)에 제공할 수 있다. 룰 관리부(200)는 종료되는 포트 포워딩과 관련된 패킷 분배 룰을 NIC의 플로우 테이블로부터 삭제할 수 있다. 전술한 실시 예는 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0107] 또 다른 일 실시 예로, 컨테이너 기반 가상화 환경에서, 특정 컨테이너가 종료될 때, 시스템 소프트웨어 내에서 실행되는 이벤트 탐지부(210)는 이를 탐지하여 관련된 정보를 시스템 소프트웨어 내에서 실행되는 룰 관리부(200)에 제공할 수 있다. 룰 관리부(200)는 종료되는 컨테이너와 관련된 패킷 분배 룰을 NIC의 플로우 테이블로부터 삭제할 수 있다. 전술한 실시 예는 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0108] 또 다른 일 실시 예로, VM 가상화 환경에서, 추가 패킷 분배를 하이퍼바이저에 요청했던 VM이 기존에 요청된 추가 패킷 분배의 종료를 요청할 때, 시스템 소프트웨어 내에서 실행되는 이벤트 탐지부(210)는 이를 탐지하여 관련된 정보를 시스템 소프트웨어 내에서 실행되는 룰 관리부(200)에 제공할 수 있다. 룰 관리부(200)는 종료 요청을 받은 패킷 분배 룰을 NIC의 플로우 테이블로부터 삭제할 수 있다. 전술한 실시 예는 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0109] 또 다른 일 실시 예로, VM 가상화 환경에서, 추가 패킷 분배를 하이퍼바이저에 요청했던 VM 종료될 때, 시스템 소프트웨어 내에서 실행되는 이벤트 탐지부(210)는 이를 탐지하여 관련된 정보를 시스템 소프트웨어 내에서 실행되는 룰 관리부(200)에 제공할 수 있다. 룰 관리부(200)는 종료되는 VM과 관련된 패킷 분배 룰을 NIC 플로우 테이블로부터 삭제할 수 있다. 전술한 실시 예는 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0110] 또 다른 일 실시 예로, 상이한 기능을 하는 워커 스레드들이 구현된 DPDK 환경에서, 특정 워커 스레드 그룹에 속한 워커 스레드들이 모두 종료될 때, 시스템 소프트웨어 내에서 실행되는 이벤트 탐지부(210)는 이를 탐지하여 관련된 정보를 시스템 소프트웨어 내에서 실행되는 룰 관리부(200)에 제공할 수 있다. 룰 관리부(200)는 종료되는 워커 스레드 그룹과 관련된 패킷 분배 룰을 NIC의 플로우 테이블로부터 삭제할 수 있다. 전술한 실시 예는 사용자 영역에서 동작하는 이벤트 탐지부(210) 및 룰 관리부(200)에도 동일하게 적용될 수 있다.

[0111] 이제까지 본 발명에 대하여 그 실시 예들을 중심으로 살펴보았다. 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자는 본 발명이 본 발명의 본질적인 특성에서 벗어나지 않는 범위에서 변형된 형태로 구현될 수 있음을 이해할 수 있을 것이다. 그러므로 개시된 실시 예들은 한정적인 관점이 아니라 설명적인 관점에서 고려되어야 한다. 본 발명의 범위는 전술한 설명이 아니라 특허청구범위에 나타나 있으며, 그와 동등한 범위 내에 있는 모든 차이점은 본 발명에 포함된 것으로 해석되어야 할 것이다.

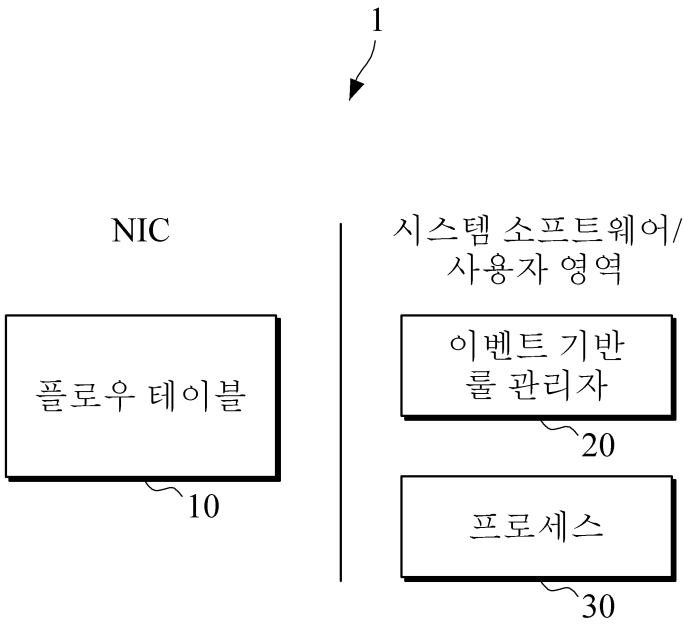
부호의 설명

[0112] 1: 패킷 분배 관리장치 20: 이벤트 기반 룰 관리자

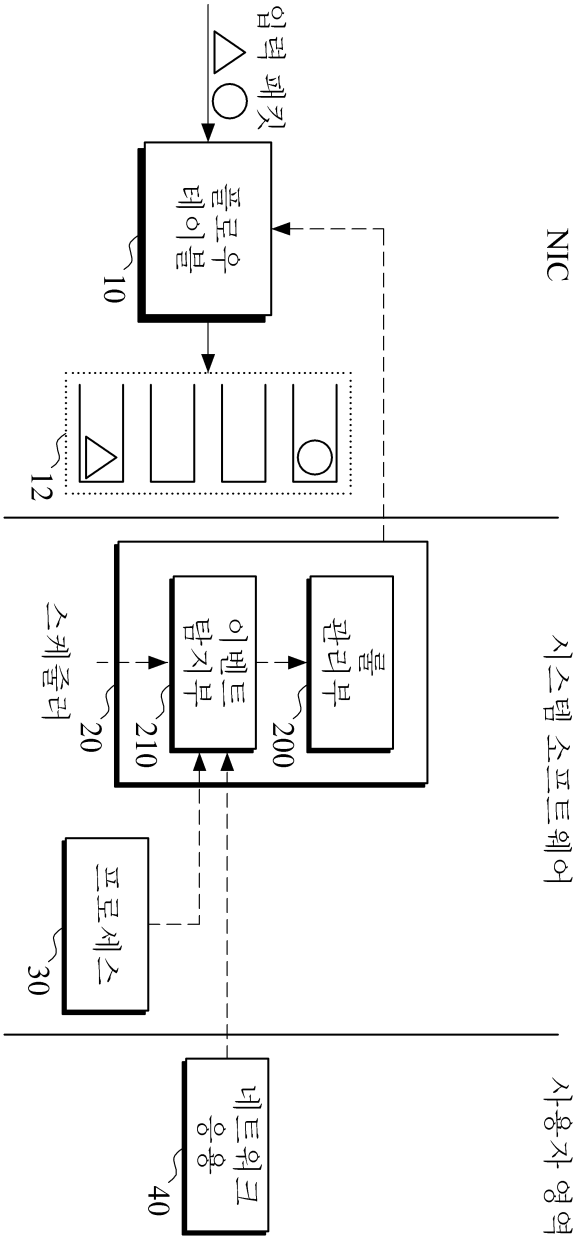
200: 룰 관리부 210: 이벤트 탐지부

도면

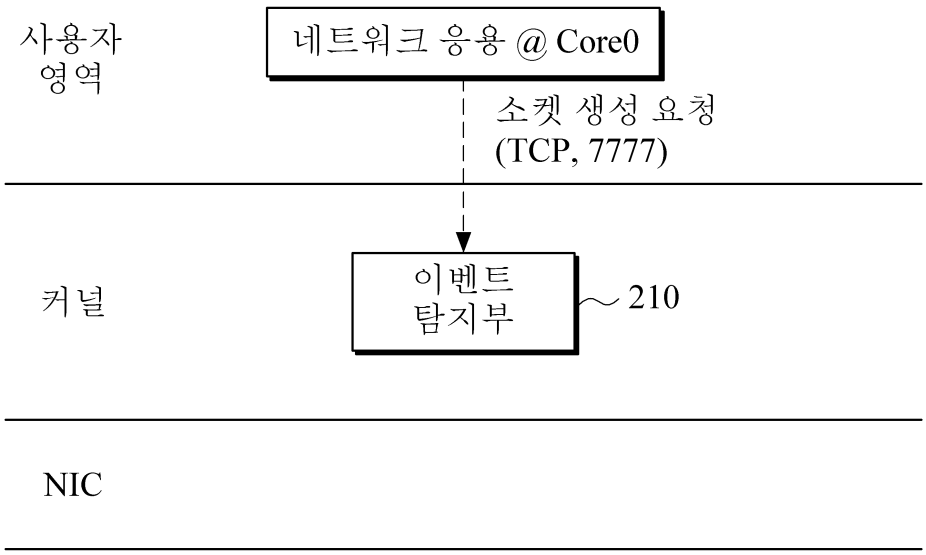
도면1



도면2

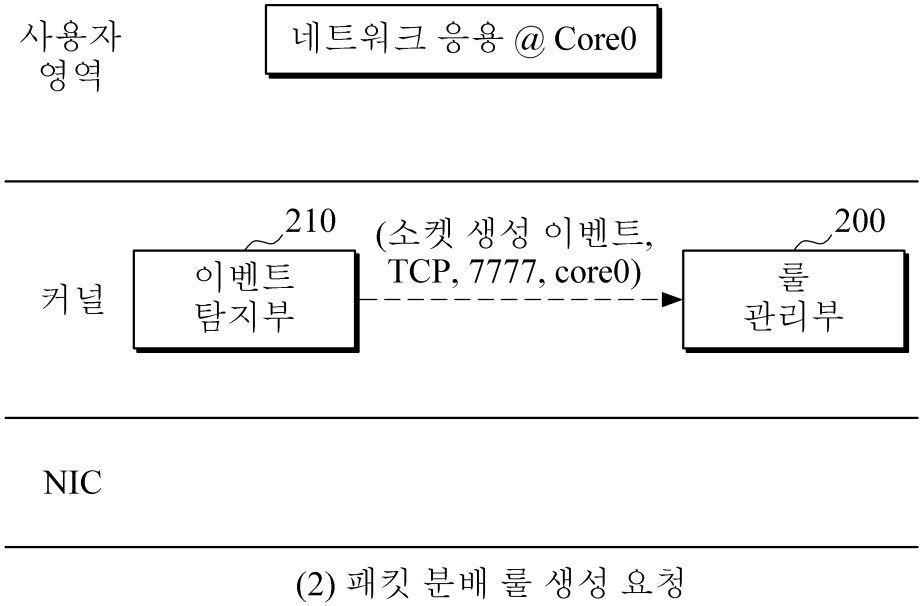

입력 패킷
플로우 테이블
10
12
NIC
시스템 소프트웨어
스케줄러
20
이벤트 탐지부
210
룰 관리부
200
프로세스
30
네트워크 응용
40
사용자 영역

도면3a



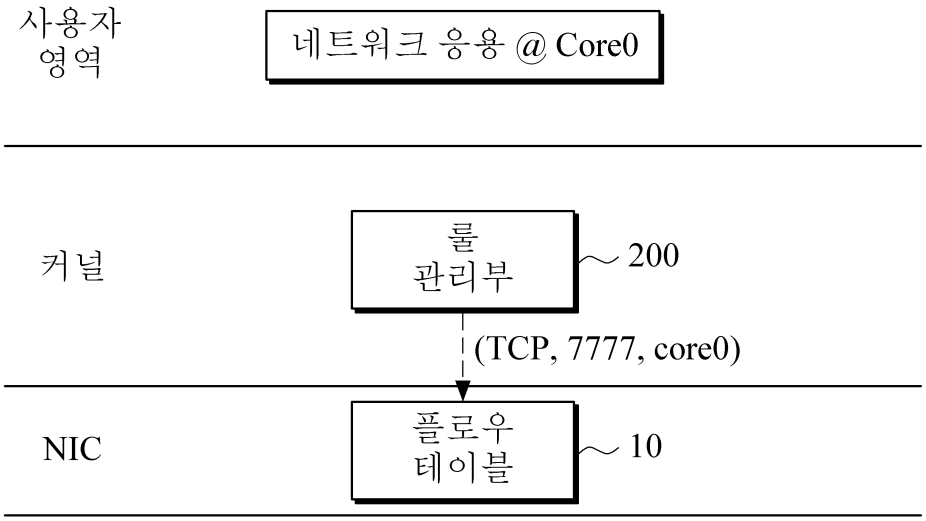
(1) 네트워크 응용 소켓 생성 및 요청 탐지

도면3b



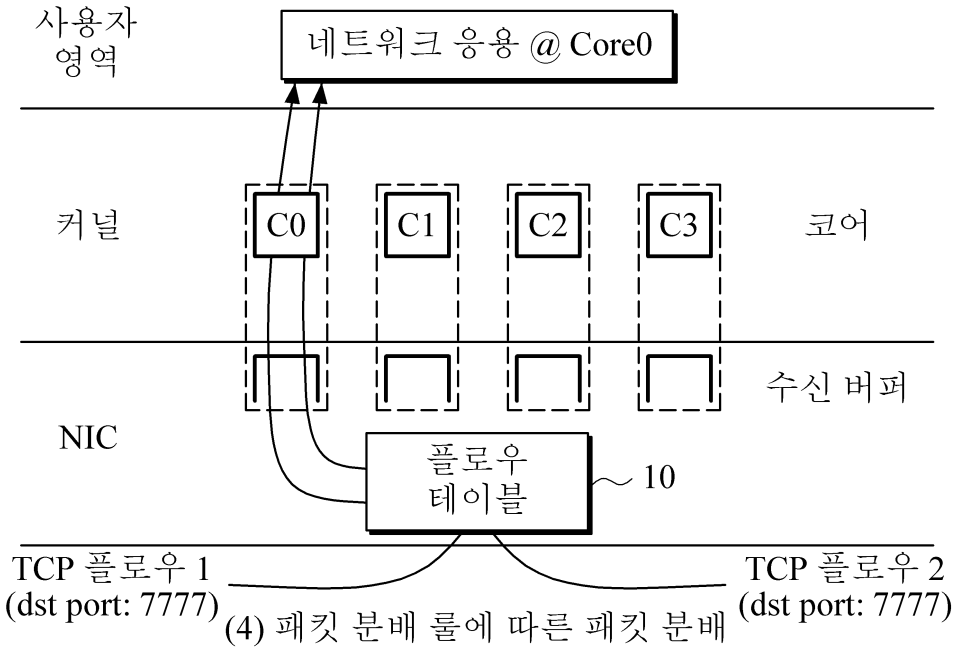
(2) 패킷 분배 룰 생성 요청

도면3c



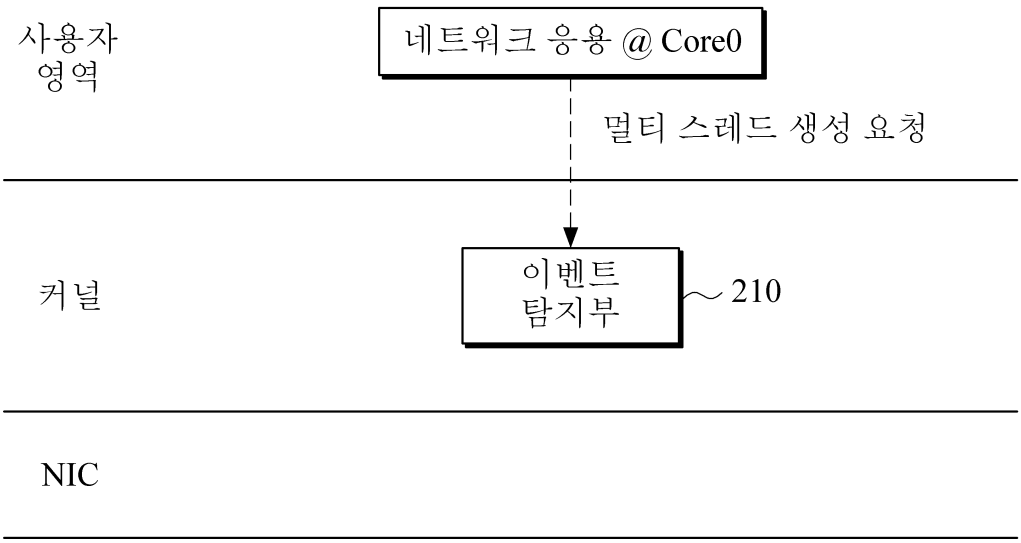
(3) 생성된 패킷 분배 룰 적용

도면3d



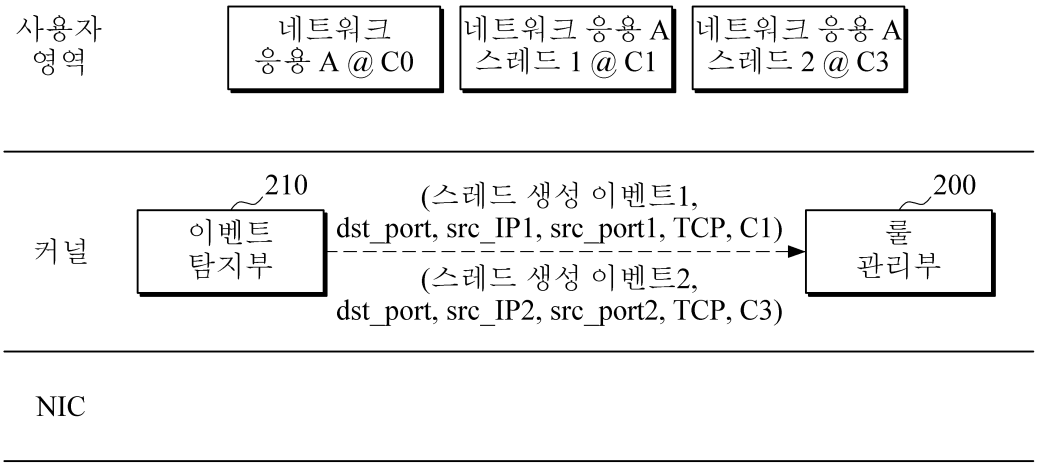
(4) 패킷 분배 룰에 따른 패킷 분배

도면4a



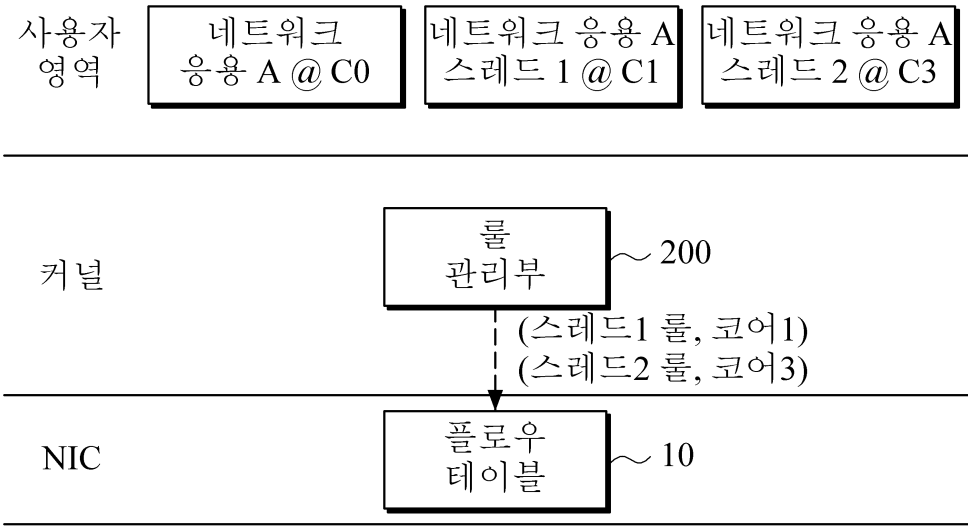
(1) 네트워크 응용 스레드 생성 및 요청 탐지

도면4b



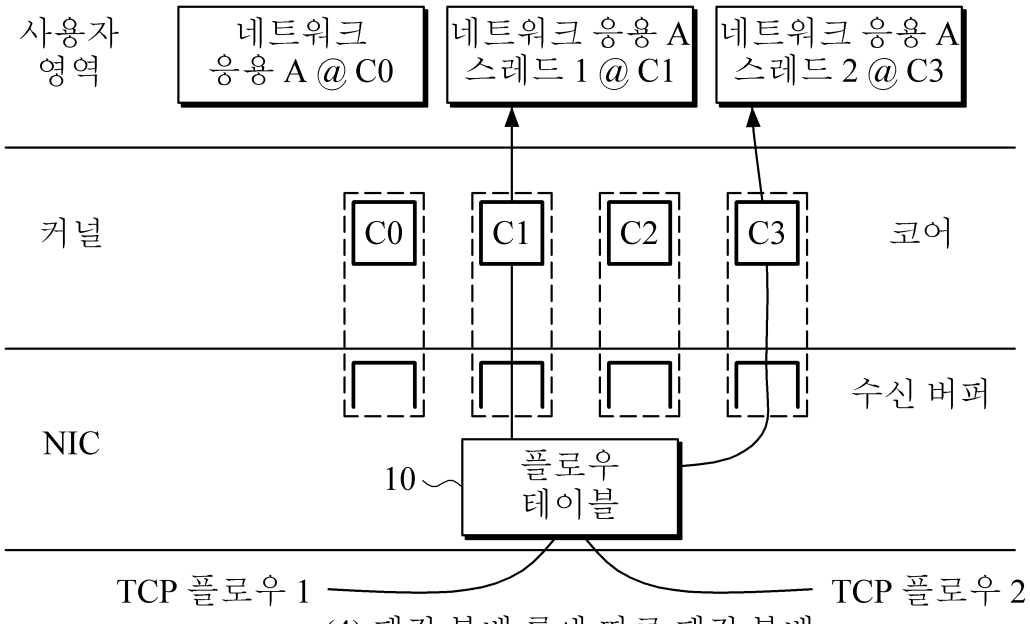
(2) 패킷 분배 룰 생성 요청

도면4c



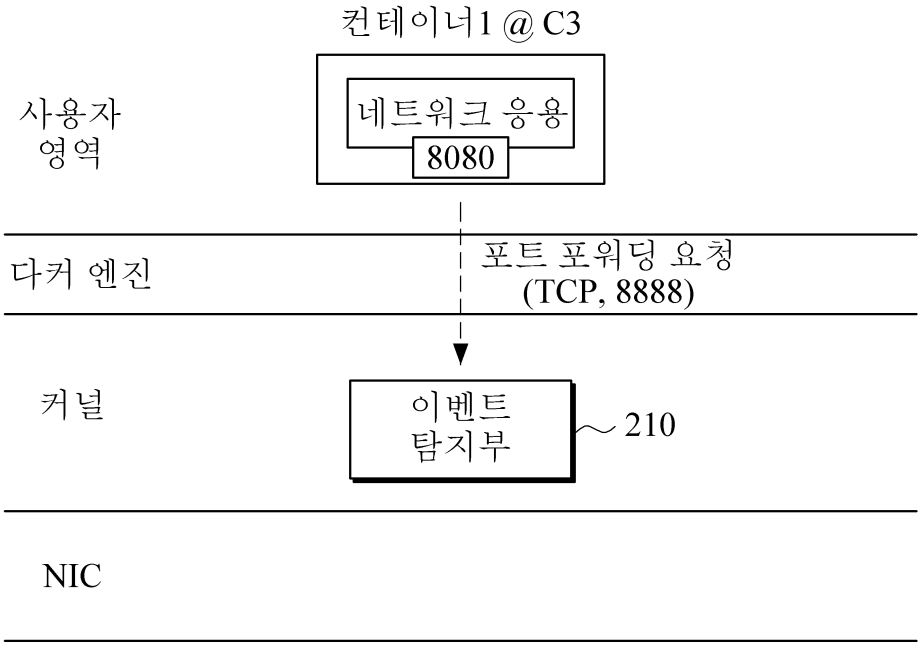
(3) 생성된 패킷 분배 룰 적용

도면4d



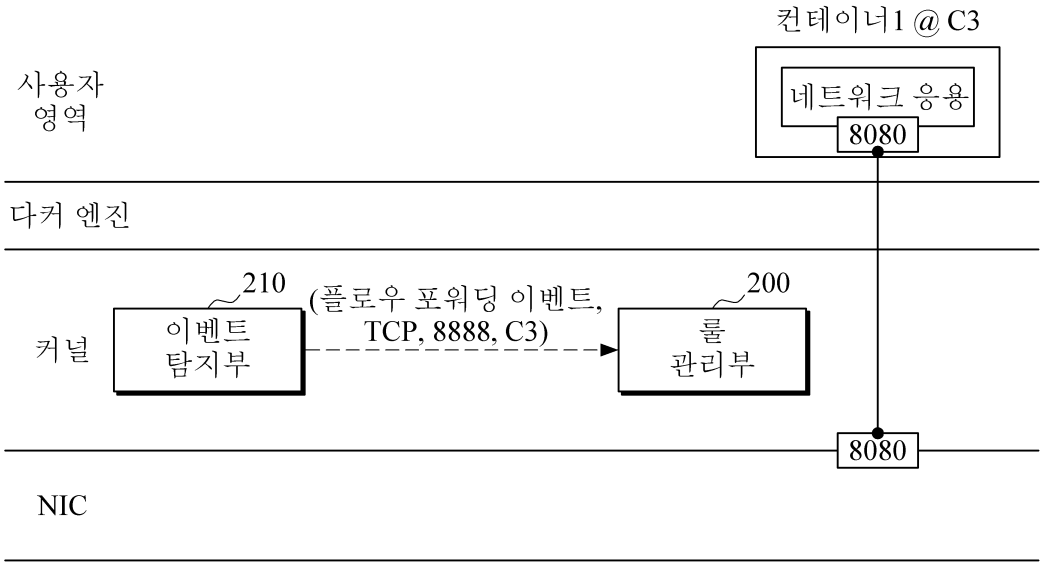
(4) 패킷 분배 룰에 따른 패킷 분배

도면5a



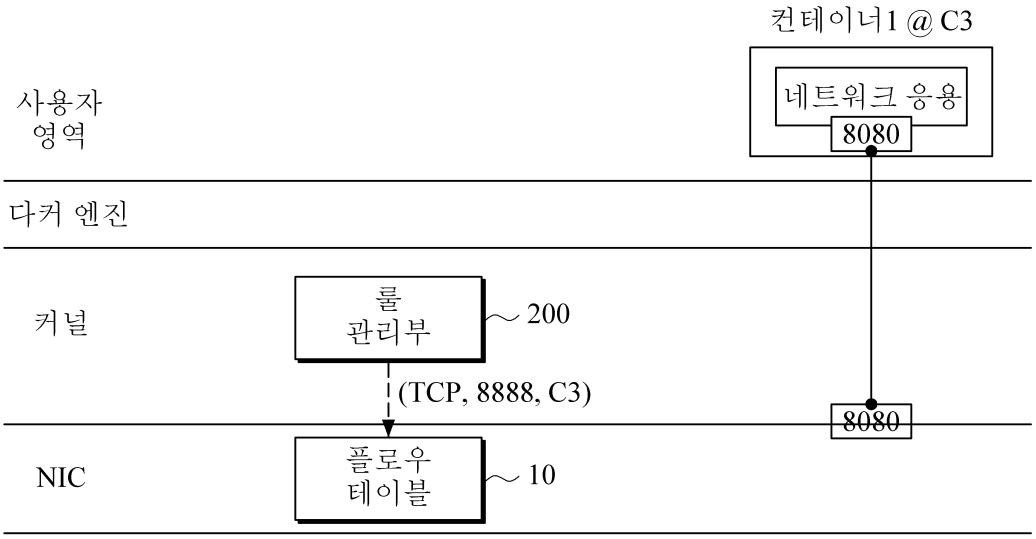
(1) 컨테이너 포트 포워딩 요청 및 요청 탐지

도면5b



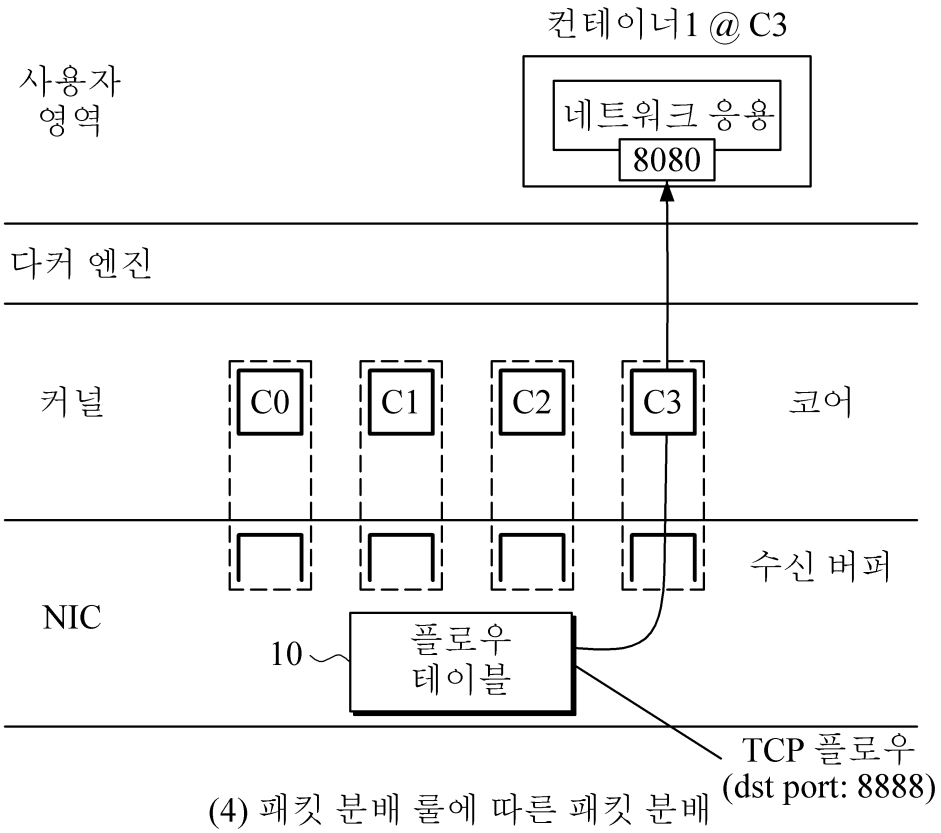
(2) 패킷 분배 룰 생성 요청

도면5c



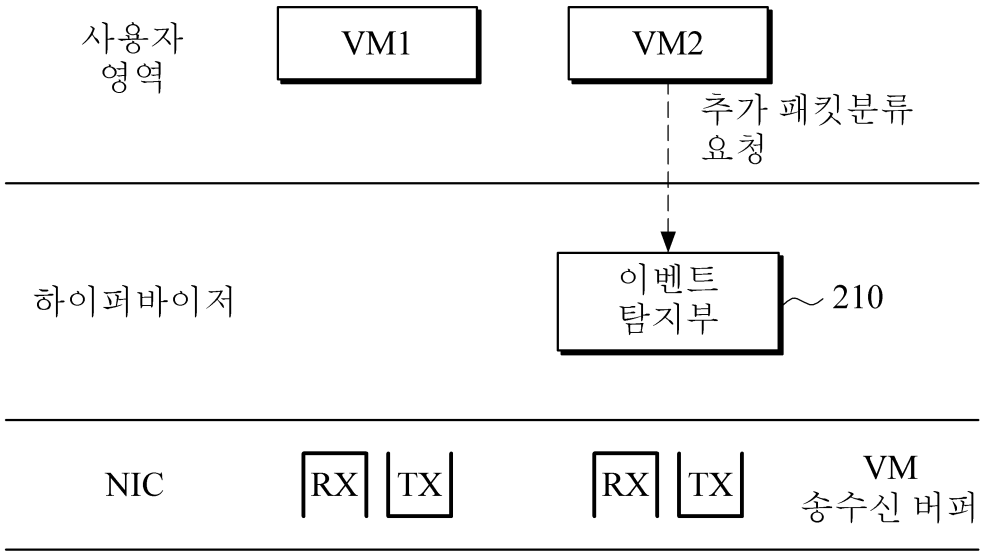
(3) 생성된 패킷 분배 룰 적용

도면5d



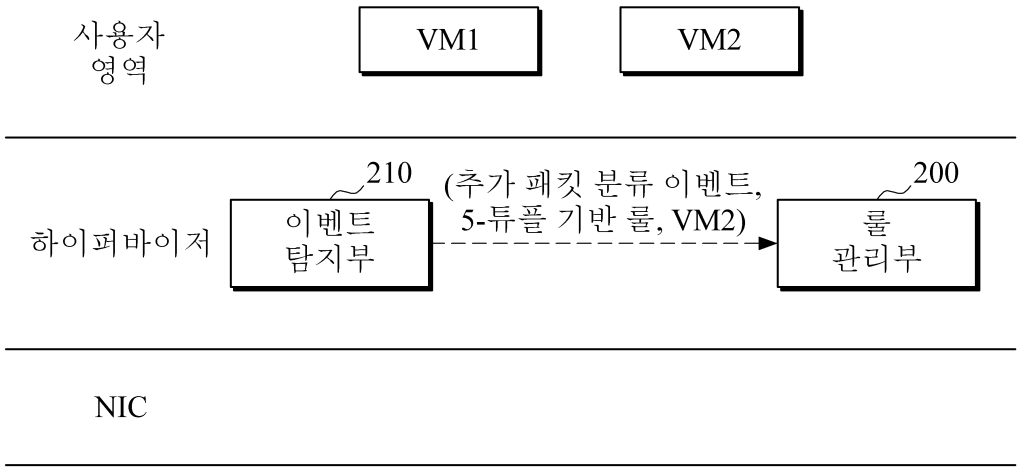
(4) 패킷 분배 룰에 따른 패킷 분배

도면6a



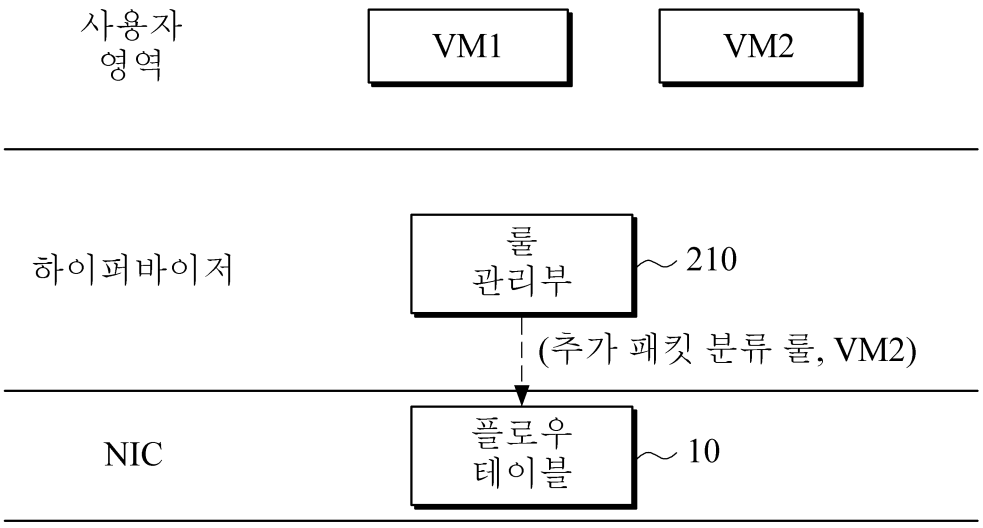
(1) 추가 패킷 분배 요청 및 요청 탐지

도면6b



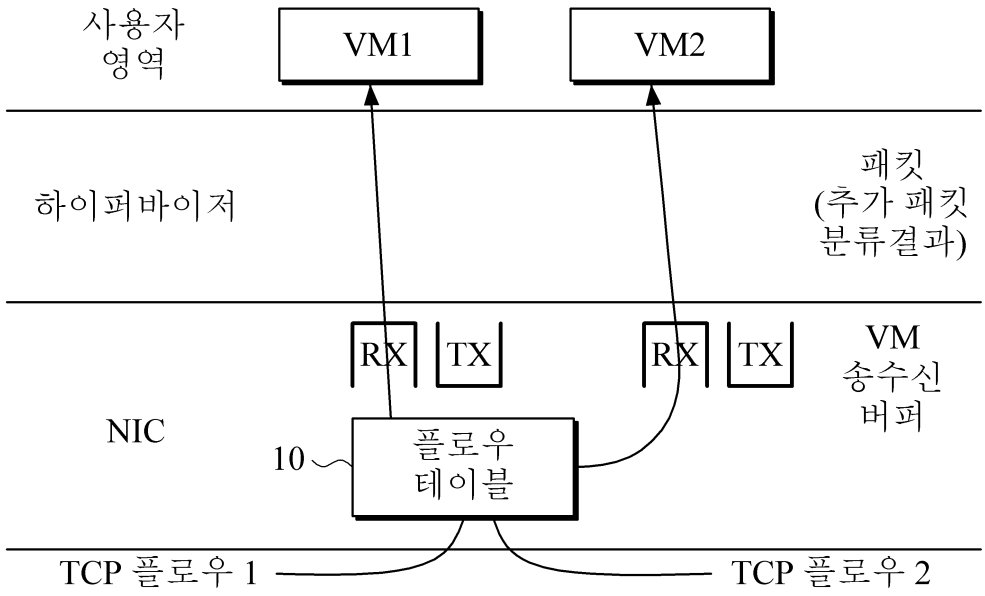
(2) 패킷 분배 룰 생성 요청

도면6c



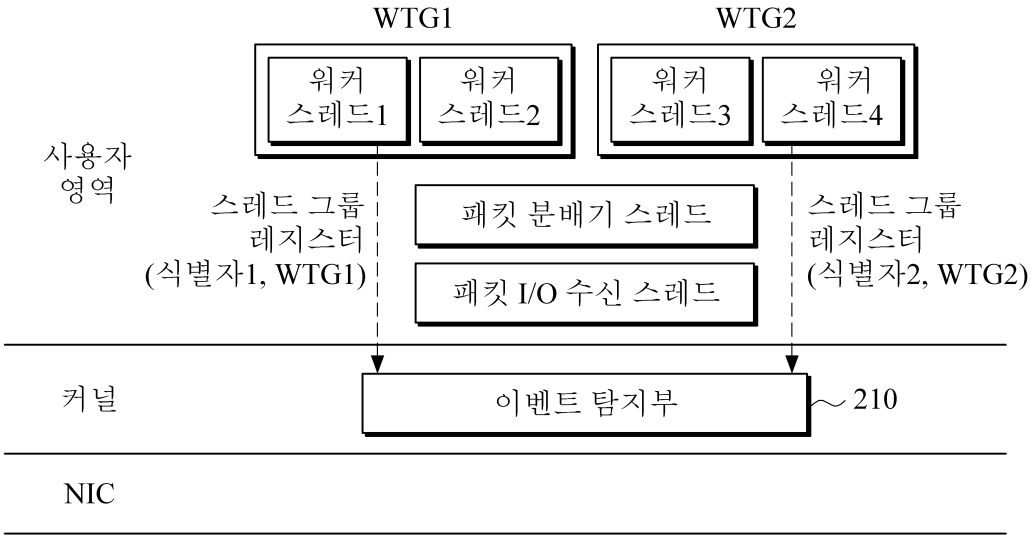
(3) 생성된 패킷 분배 룰 적용

도면6d



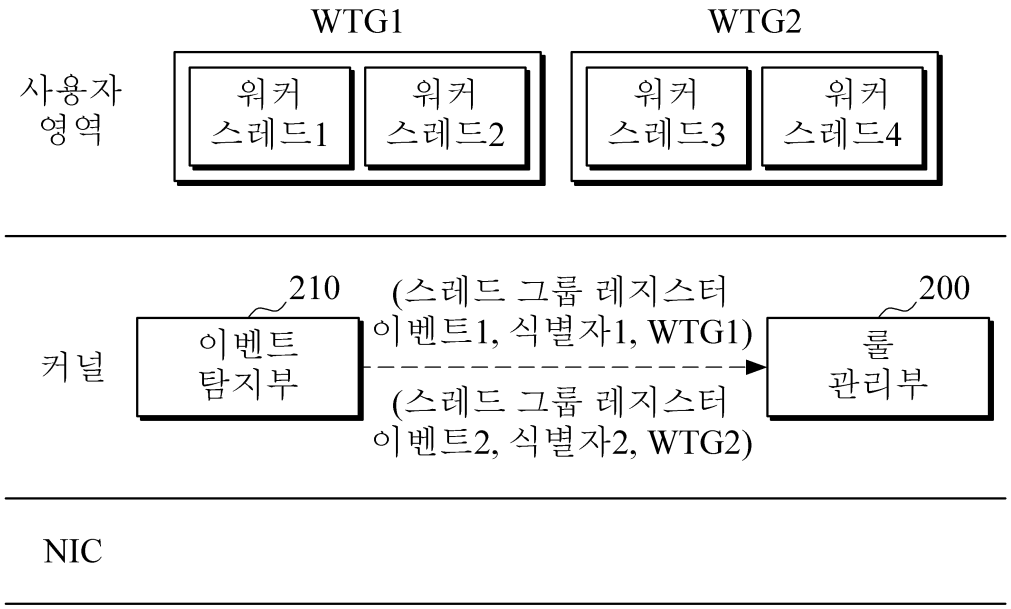
(4) 패킷 분배 룰에 따른 패킷 분배

도면7a



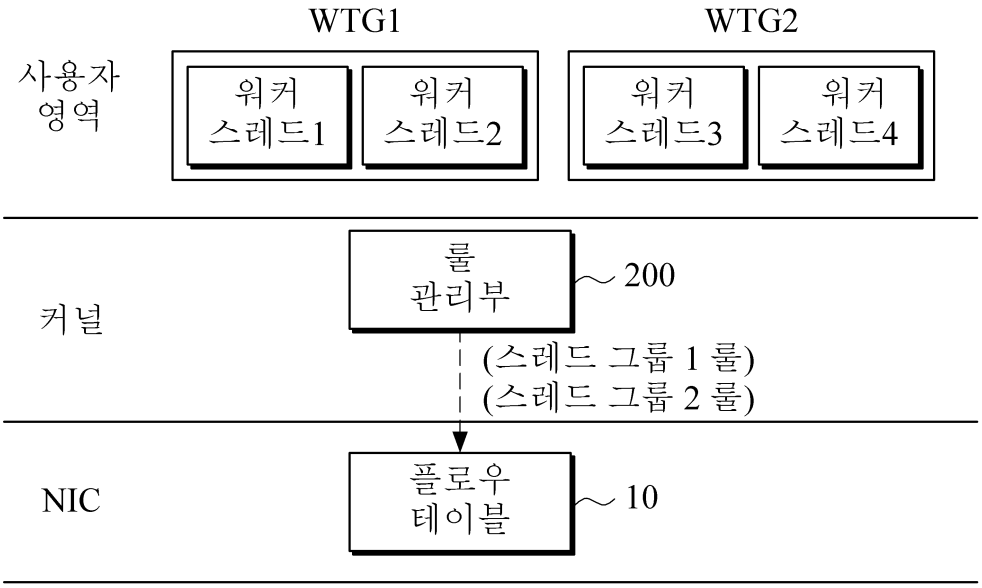
(1) 상이한 기능을 하는 워커 스레드들의 등록 요청 및 요청 탐지

도면7b



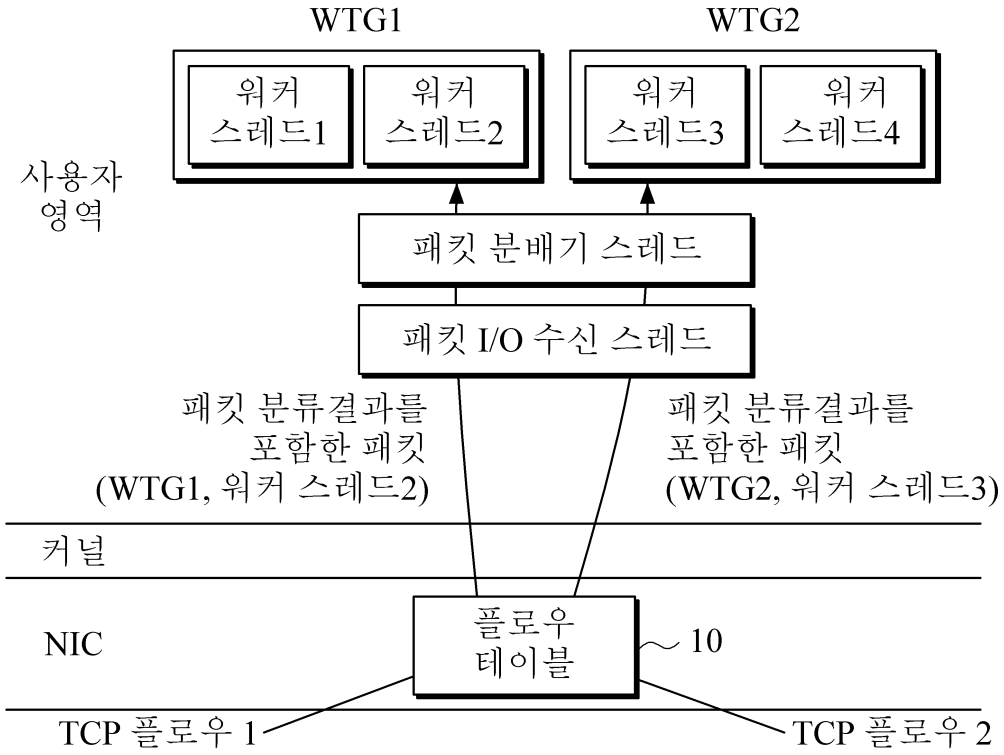
(2) 워커 스레드 기반 패킷 분배 룰 생성 요청

도면7c



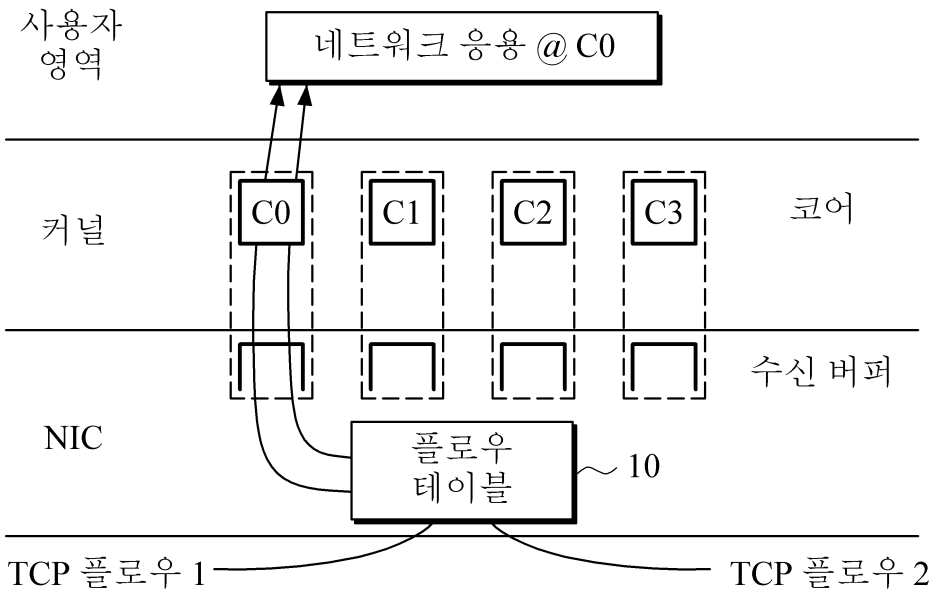
(3) 생성된 패킷 분배 룰 적용

도면7d



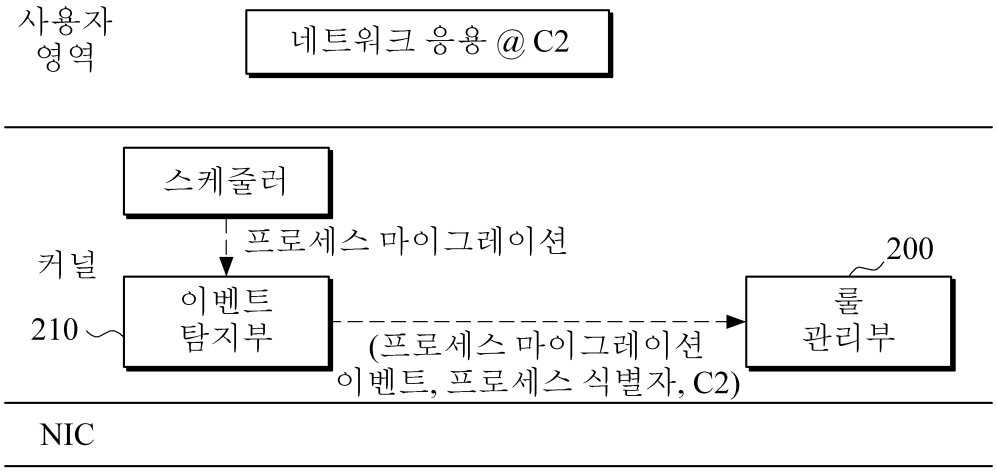
(4) 패킷 분배 룰에 따른 패킷 분배

도면8a



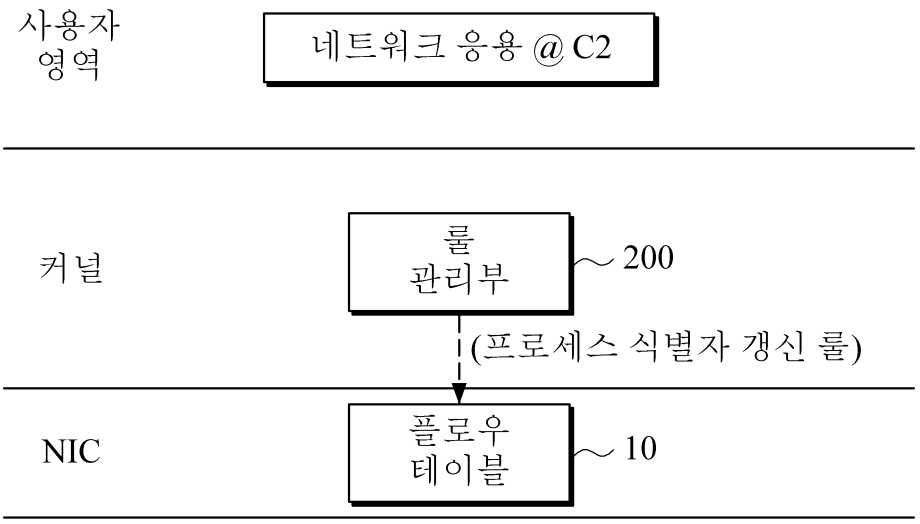
(1) 패킷 분배 룰에 따른 패킷 분배

도면8b



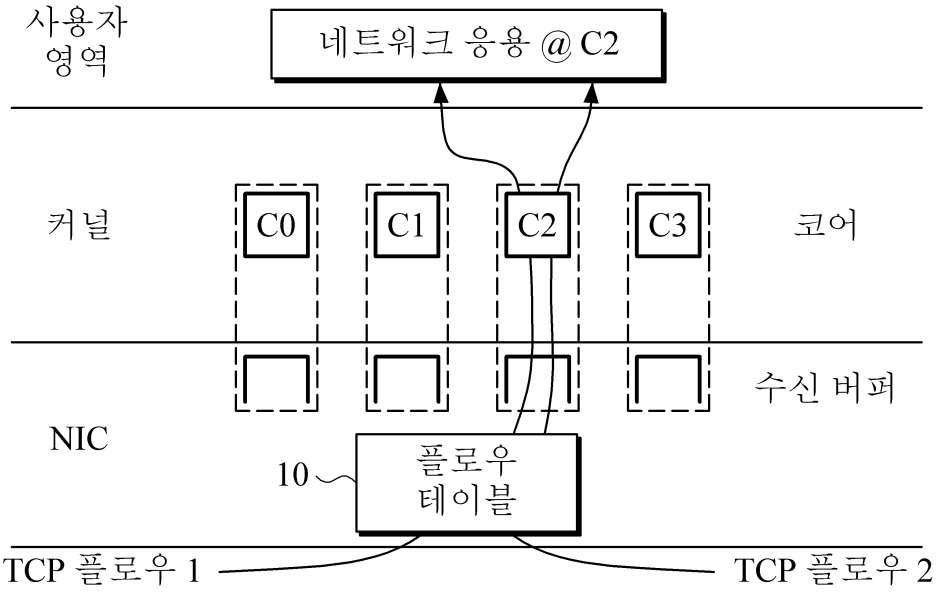
(2) 마이그레이션 탐지 및 패킷 룰 갱신 요청

도면8c



(3) NIC에 갱신된 패킷 분배 룰 적용

도면8d



(4) 갱신된 패킷 분배 룰에 따른 패킷 분배