Journal of Korea Game Society 2016 Aug; 16(4): 57-68
http://dx.doi.org/10.7583/JKGS.2016.16.4.57

게임프로그래밍

게임서버 가상머신을 위한 지연 제약 스케줄링

강기욱*, 고철홍*, 홍지만*, 백두원**
숭실대학교 컴퓨터학과*, 숭실대학교 글로벌 미디어학부**
kiwookkang6368@gmail.com, koch11@naver.com, {jiman, dpaik}@ssu.ac.kr

Delayed-constrained scheduling for a game server virtual machine

Kiwook kang*, Cherl-hong Ko*, Jiman Hong*, and Doowon Paik**
School of Computer Science and Engineering, Soongsil University*
Global School of Media, Soongsil University**

요 약

게임 애플케이션에서 사용하는 데이터의 크기가 점차 커짐에 따라 물리적인 게임 서버 자원은 점차 늘어 가고 있다. 이에 따라 서버의 I/O 성능을 향상시키기 위해 게임 서버에 I/O 가상화 기술을 도입하고자 하는 요구가 점차 증가하고 있다. 그러나, I/O 지연 시간이 수시로 변하는 게임 서버는 I/O 응답성을 쉽게 보장하기가 힘들다. I/O 가상화 효과를 극대화하기 위해 I/O 응답성 보장은 매우 중요하며 가상 머신의 우선순위에 따라 I/O 지연 시간을 관리할 수 있는 I/O 스케줄링 기법이 반드시 필요하다. 따라서 본 논문에서는 가상화 환경에서 최대 I/O 지연 시간을 보장하는 효율적인 지연 제약 스케줄링 기법을 제안한다. 또한 제안한 기법을 이용하여 지연시간을 보장하는지 실험을 하여, 패킷의 손실량이 줄고 스케줄링의 공정성이 증가한 것을 확인하였다.

ABSTRACT

As the size of the data used in the game application increase gradually, the physical resources of game server grow. Accordingly, it is necessary to/it is required to introduce I/O virtualization in game server to improve I/O performance of the server. But it is difficult to ensure high responsiveness in game server where I/O delay change from time to time. To maximize the benefit of I/O virtualization, guaranteeing I/O response time is very important and it is necessary to have I/O scheduling techniques to manage the I/O latency in the order of priority of virtual machines. In this paper, we propose an efficient delay-constrained scheduling algorithm in a virtualization environment to ensure maximum I/O latency. In addition, a reduced amount of loss of the packet was found to increase the fairness of scheduling in the experiments with the proposed scheme.

Keywords : Virtualization(가상화), Network scheduling(네트워크 스케줄링), QoS(QoS)

Received: Jul, 10, 2016 Revised: Aug, 10, 2016
Accepted: Aug, 16, 2016
Corresponding Author: Doowon Paik(Soongsil University)
E-mail: dpaik@ssu.ac.kr

ISSN: 1598-4540 / eISSN: 2287-8211

1. 서 론

최근 전 세계의 게임 시장이 점차 커지고, 그에 따라 게임 애플케이션에서 사용하는 데이터의 크기가 점차 커짐에 따라 물리적인 게임 서버 자원은 점차 늘어 가고 있다. 예를 들어, 세계적인 게임 크래쉬 오브 클랜은 전 세계에 1억 명 이상의 유저를 보유하고 있고, 그에 따른 유저 데이터와 게임 데이터는 점차 커지고 있다. 이에 따라 물리적인 서버 자원을 증가시키지 않고, 서버에서 사용하는 고성능 프로세서에 걸맞는 스토리지 I/O를 확보하고 서버의 I/O 성능을 향상시키기 위해 게임 서버에 I/O 가상화 기술을 도입하고자 하는 요구가 점차 증가하고 있다.

그러나, I/O 지연 시간이 수시로 변하는 환경에서는 I/O 응답성을 쉽게 보장하기가 힘들다. 특히 게임서버는 다수의 사용자가 불규칙하게 접속하고 우선순위가 다른 다종 다량의 패킷을 실시간으로 처리하고 응답해야 하므로 I/O 응답성의 보장이 더욱 중요하다. I/O 가상화를 극대화하기 위해서는 I/O 응답성 보장은 매우 중요하며[2][6] 가상 머신의 우선순위에 따라 I/O 지연 시간을 관리할 수 있는 I/O 스케줄링 기법이 반드시 필요하다.

대표적인 가상화 솔루션인 Xen에서는, 도메인0라는 특권 도메인이 게스트 도메인들의 I/O 과정에 관여하여 응답성에 영향을 준다. 그러나 도메인0는 네트워크 I/O 요청을 라운드 로빈 방식으로 처리하며, 이는 도메인의 우선순위와 상관없이 네트워크 I/O 요청을 처리함으로써 우선순위 역전현상을 발생시킨다는 문제점이 있다.

이를 해결하기 위해 제안된 기존의 기법들은 우선순위 역전현상을 완화하고 높은 우선순위 도메인들의 응답성을 향상시켰다. 하지만 이러한 기법들은 우선순위가 높은 가상머신의 응답성 향상에 초점이 맞추어져 있어, 낮은 우선순위의 도메인의 I/O 요청에 대한 과도한 지연 현상이나 기아현상이 일어날 수 있다.

따라서 본 논문에서는 과도한 지연과 기아현상을 방지하기 위해, 가상화 환경에서 최대 I/O 지연 시간을 보장하는 효율적인 지연 제약 스케줄링 기법을 제안한다. 제안한 지연 제약 스케쥴링 기법은 도메인 우선순위를 기반으로 지연 최대 시간의 차이를 두어 도메인 우선순위와 지연시간의 차이를 균등하게 조절한다.

지연 제약 스케줄링은 가상화 환경에서 실시간 도메인과 비실시간 도메인의 네트워크 I/O를 스케줄링한다. 비트맵으로 네트워크 I/O 패킷 처리 최대 지연시간을 관리하고, 또한 과도한 지연을 발생시킬 것으로 예상되는 I/O 요청의 우선순위를 조정함으로써, I/O 응답성을 보장하고 네트워크 QoS를 만족시킨다. 또한 제안한 스케줄링 기법은 도메인 우선순위 간 지연시간 차이를 균등하게 조절하여 공정성을 향상시킨다.

기존 기법에서는 도메인 우선순위 적용을 위해 리눅스 커널의 TX큐를 분리하였지만, 제안 기법은 구현상의 안정성을 확보하고 확장성을 높이기 위해 RX큐에 우선순위 포인터를 적용한다. 또한 지연시간의 한계를 설정하고 이를 관리하기 위해 도메인 우선순위별 비트맵을 이용한다. 제안한 기법을 Xen 가상화 시스템에 실제 구현하고 제안 기법을 사용할 경우 가상화 서버의 I/O 응답성과 공정성이 기존 시스템에 비해 향상됨을 보였다.

본 논문의 구성은 다음과 같다. 2장에서는 가상화 환경에서 QoS보장을 위한 관련 연구들을 소개한다. 3장에서는 제안하는 제안하는 스케줄링 기법을 설명한다. 4장에서는 실험을 통해 제안 기법이 I/O 지연 측면에서 성능이 향상되는 것을 확인하였다. 마지막으로 5장에서는 본 논문의 결론을 맺고 향후 연구 방향을 제시한다.

2. 관련 연구

Kim 등은[1] 작업량이 가변적인 환경에서 게스트 도메인의 수행 결과를 모니터링하여 Xen 스케줄러에게 피드백을 주는 시스템을 제안하였다. 제

안한 기법은 가변적인 워크로드를 효율적으로 반영하여 스케줄링을 한다는 장점이 있지만, 모니터링 모듈과 하이퍼콜의 오버헤드가 발생하며, 비실시간 태스크의 혼재 상황을 고려하지 않았다.

LI 등은[7] 대표적인 커널버전 3.10, 3.16을 기반으로 도메인0에서의 네트워크 I/O 처리 구조의 차이점을 제시하였다. 또한 커널 스레드의 우선순의에 도메인 우선순위 큐를 할당하여 실시간 도메인의 응답성을 향상시키는 VATC를 제시하였다.

XI 등은[6] 도메인0의 TX큐를 도메인의 우선순위에 따라 분리하고 게스트 도메인의 우선순위를 반영하여 I/O 요청을 처리함으로써 실시간 도메인의 응답성을 향상시켰다. 하지만 통합 QoS 환경에서의 도메인 간 커뮤니케이션만을 고려하였으며, 높은 우선순위의 도메인 간 커뮤니케이션이 지속적으로 발생할 경우, 낮은 우선순위 도메인의 패킷이 제한없이 지연될 수 있다.

Nath 등은[5] Xen 3.0 버전까지 사용되었던 SEDF 스케줄러 기반에서 I/O 횟수를 누적시켜 누적 횟수가 많은 도메인을 우선적으로 스케줄링하여 I/O 응답성을 향상시키는 기법을 제시하였다.

3. 본 론

3.1 Xen에서의 네트워크 패킷 송수신

제안 기법의 설명에 앞서 Xen에서의 네트워크 패킷 송수신 과정과 한계를 분석한다. 이후 기존 기법의 문제점을 제시하고 제안하는 지연 제약 스케줄링을 기술한다.

3.1.1 도메인0에서의 네트워크 관련 요소 및 구성

본 절에서는 Xen에서 네트워크 I/O 처리과정을 분석한다. Xen에서 네트워크 I/O는 특권 도메인인 도메인0이 관여하여 처리된다. 도메인0는 리눅스 커널을 기반으로 작동하여, TX 및 RX 큐와 유사한 처리 루틴이 존재한다. 게스트 OS의 패킷 송수신을 위한 가상화 기술 구현을 위해 리눅스의 네트워크 I/O 처리와 비교하여 몇 가지 차이를 보이는 요소와 과정이 존재하며, 이에 따른 추가적인 오버헤드가 발생한다.

[Table 1] Virtualization-related components of the domain 0

Virtualization related components	explains
xen_netbk	It shows the network back-end device, and it is created and initialized in netback_init () routine at boot time. It can access to sturcture netbk.
xenvif	It is a virtual network interface generated for each domain. Xen conect network device with xenvif which is network interface of domain 0
xen_netbk_kthread	If there are I/O request in rx_queue, xen_netbk_kthread routine process transmission to relevant domain.
xen_netbk_tx_action	If there are I/O request in rx_queue, xen_netbk_kthread routine process transmission to relevant domain.
xen_netbk_rx_action	xen_netbk_kthread routine process xen_netbk_rx_action earlier than xen_netbk_tx_action. If there are I/O request in rx_queue, xen_netbk_kthread routine process reception to relevant domain.

Xen의 도메인0에서 네트워크 I/O 처리를 위한 관련요소들을 [Table 1]에 기술하였으며, [Fig. 2]는 게스트 도메인과 도메인0, 관련요소들의 구조를 보인다. 본 논문에서는 리눅스 3.10버전을 기반으

로 분석하였다. 분석한 결과를 바탕으로 기존 우선순위 네트워크 I/O 스케줄링 기법의 개선방향을 기술하고, 제안하는 지연 제약 스케줄링 기법을 설계한다.

도메인0에서는 [Table 1]의 가상화 관련 요소들이 초기화 및 실행된다. 초기화 루틴이 포함된 함수는 netback_init()이다. 도메인0의 netback 드라이버인 xen_netbk를 생성 후 schedule list와 락 등을 초기화 한다. netback device는 TX큐와 RX큐를 유지한다. TX큐 및 RX큐는 각각 게스트 도메인들의 전송 및 송신 패킷을 위한 큐이다. 이후 TX큐 및 RX큐를 처리하기 위한 xen_netbk_=kthread를 생성 및 활성화한다. TX큐는 xen_netbk_tx_action에 의해 처리되며, RX큐는 xen_netbk_rx_action에 의해 처리된다. xenif 디바이스들은 TX 및 RX큐를 공유한다. 따라서 xenif 디바이스 패치 시 스케줄링 방법에 따라 지연시간 차이가 발생한다.

3.2 기존 I/O 처리과정의 문제점

기존의 도메인 우선순위 기반 패킷처리 알고리즘은 적은 우선순위 차이에도 불구하고 과도한 네트워크 I/O 지연을 발생시킨다는 문제점이 있다. 본 연구에서는 게스트 도메인에서 제공하는 우선순위를 기반으로 지연시간 한계의 차이를 두고, 네트워크 I/O 스케줄링의 공정성을 향상시킨다.

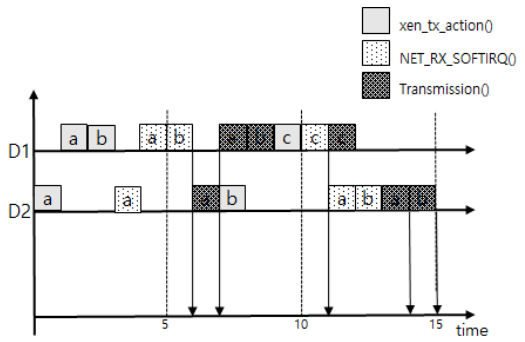
3.2.1 지연제약 스케줄링 알고리즘 적용 전의 I/O 처리과정 분석

제안한 기법을 적용하기 전의 네트워크 I/O 처리과정은 [Fig. 1]과 같다. 기존 기법은 I/O 처리 시 xen_netbk_tx_action()함수에서 라운드로빈 대신 도메인 우선순위에 기반한 선점형 스케줄링을 적용하였다. [Fig. 1]은 우선순위가 D1>D2일 때 도메인의 우선순위별 패킷 전송 과정을 나타낸다. 또한 xenif를 패치 후 한 라운드에 처리하는 패킷 수를 나타내는 batch size는 2로 가정하였다. batch size는 초기 값으로 238이며, 시스템 특성에 따라 변경 가능하다. 기존 우선순위 적용 기법은 네트워크 I/O 처리과정에서 높은 우선순위 가상머신의 패킷이 도달하면 선점이 발생한다.

[Fig. 1]에서 보이는 바와 같이 TX큐에 D1의 패킷이 도착하고 D1의 패킷 a, b가 도착하는 경우 선점이 발생하여 D1의 a, b 패킷이 RX큐까지 도달하여 물리 NIC에 전송되기까지 D2의 패킷은 처리를 기다리게 된다. D1의 a 패킷이 TX큐에 도착해서 D1 도메인으로 전달되기까지 5의 시간이 걸리지만, D2의 a 패킷은 12의 지연이 발생한다.

[Fig. 1]에서는 2개 우선순위의 xenvif를 가정하였지만, Dn의 값이 증가할수록 우선순위별 지연시간 차이는 더욱 크게 발생하게 된다. 이러한 상황에서, 높은 우선순위의 도메인 패킷 송수신이 지속될 경우 과도한 하위 우선순위 패킷의 지연 발생으로 인해 도메인 서비스의 QoS를 위반할 수 있다는 문제점이 있다.

이러한 문제는 선점과 도메인 우선순위를 기준으로 도메인0에서의 I/O 지연시간을 관리하여 완화할 수 있다. 단, 우선순위 적용의 목적인 도메인 우선순위 기반 I/O 처리를 유지하며, 과도한 지연이 발생할 것으로 예상된 패킷의 우선순위를 일시적으로 변경하여 QoS를 보장하는 것을 목적으로 한다.



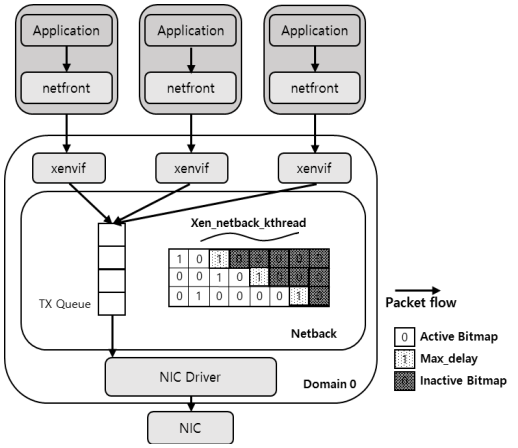
[Fig. 1] Prioritiezed network I/O processing

3.3 지연제약 스케줄링

제안하는 지연 제약 스케줄링은 비트맵을 이용하여 네트워크 I/O의 지연 시간을 관리하고, 지연 최대시간을 게스트 도메인에서 제공하는 서비스의 QoS에 기반하여 설정한다. 본 절에서는 지연제약 스케줄링의 수행시점과 비트맵 변화 및 구현과정을 상세히 기술한다. [Fig. 2]는 지연 제약 알고리즘의 수행 흐름을 나타낸다.

Xen은 dom0의 Xen Network Backend 모듈을 통해 domU로의 네트워크 인터페이스를 제공하므로 Xen Network Backend 모듈에서 사용되는 정보들을 바탕으로 알고리즘을 시스템에 구현한다[3].

도메인마다 할당되는 가상 네트워크 인터페이스인 xenvif는 pending packet list를 가지며, xen_rx_action()함수가 수행되면 xenvif 디바이스의 packet 하나를 처리 후 스케줄 리스트의 맨 끝에 삽입되어 라운드 로빈 방식으로 처리된다. 기존 연구들[6][7] 에서는 라운드로빈 방식 대신 도메인 우선순위를 기반으로 xenvif를 스케줄링하며, pending packet이 있으면 리스트의 맨 끝에 삽입되지 않고 계속 처리된다. 또한 한번에 batch size 만큼 패킷을 패치한다. 본 연구에서는 이를 기반으로 지연시간 관리 기법을 적용한다.



[Fig. 2] Structure of delay-constrained scheduling

기존 기법대로 xenvif를 스케줄링하면 높은 우선순위의 xenvif에 패킷이 도착하면 선점이 발생하게 된다. 선점된 후 발생한 지연시간을 비트맵으로 관리하고, xen_tx_action() 함수에서 선점과 batch size를 기준으로 지연을 카운트하고, 최대 지연시간에 도달하면 우선적으로 스케줄링한다.

3.3.1 QoS에 따른 도메인 우선순위 분류 및 초기화

게스트 도메인에서 제공되는 서비스의 QoS에 따라 도메인0에 생성되는 netif 디바이스의 우선순위를 설정한다. QoS를 측정 지표는 패킷 지연시간, 패킷 손실률, 단방향 지연, 지터 등이 있으며, 본 연구에서는 서비스에서 요구하는 QoS의 지연시간 한계에 따라 분류한다.

QoS의 지연시간 제약사항은 서비스별, 품질지표별로 상이하다[1][4]. 따라서 게스트 도메인에서 서비스되는 애플리케이션의 지연시간 제약사항의 비율을 기준으로 도메인 우선순위 및 length를 설정하며, 본 연구에서는 각 도메인에서 요구하는 네트워크 I/O 지연 제약사항을 기반으로 비율대로 count bit를 할당하였다.

도메인마다 생성되는 xenvif 구조체에 delay_count 변수를 추가하였다. delay_count는 QoS 지연시간 비율에 따라 정수로 설정된다. 이 변수는 도메인에서 제공하는 서비스 특성에 따라 관리자가 설정할 수 있으며, delay_count 변수에 비례하여 비트맵 구조가 결정된다.

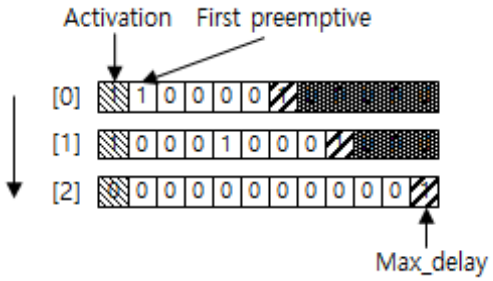
3.3.2. 우선순위에 따른 지연시간 관리 비트맵 구성 및 변경조건

설정된 우선순위에 따라 지연시간 관리 비트맵이 구성된다. [Fig. 3]은 도메인이 3개이고, 각각의 QoS 지연시간 비율이 5:7:10일 때 비트맵 구성을 나타낸다. 예를 들어, 도메인에서 서비스하는 어플리케이션의 QoS 지연 제약사항이 도메인마다

30ms, 60ms, 100ms이면 각 xenvif는 3:6:10 길이의 지연시간 비트맵을 가진다.

delay_count 비트가 길어지면 선점 후 지연시간을 길게 허용하게 된다. 따라서 낮은 지연시간을 요구하는 도메인일수록 지연시간 카운트가 Max_delay에 도달하는 시간을 짧게 지정하여야 한다. 즉, 우선순위 상속을 지연시간 제약사항에 비례하여 Max_delay를 설정함으로써 QoS에 기반한 지연시간 관리를 가능하게 한다. 단, QoS에 비례하여 지연시간 비트맵을 설정할 때 2:3:5 와 같은 비율인 경우 우선순위별 최대 지연시간 차이가 적으므로 n배수로 설정하여 지연시간 비트가 QoS를 차등적으로 반영할 수 있도록 설정하여야 한다.

첫 번째 비트는 지연 시작 여부를 나타내는 활성화 비트이다. 선점으로 인한 지연이 발생한 xenvif의 비트맵은 첫 비트가 1로 설정된다. 활성화비트를 설정함으로써 xen_tx_action()에서 batch size만큼 패킷을 패치 할 때마다 활성화 비트가 set되어있는 비트맵의 지연시간 비트만 업데이트하게 된다. 활성화 비트는 우선순위 상속이 일어날 때 지연시간 비트와 함께 리셋된다.



[Fig. 3] Example of bitmap construction

업데이트 과정은 지연시간 비트를 Left-shift 후 다음 비트와 and 연산을 수행하여 1이 되면 최대 지연시간에 도달한 것으로 확인된다. 최대 지연시간에 도달한 비트는 리셋하고 temp_priority에 현재 가장 높은 xenvif의 우선순위를 상속받는다. [Fig. 4]는 이 절에서 설명한 내용을 의사코드로 나타낸 것이다.

```
update() :
    for elem in all_of_xenvif :

        // if active bit was set, delay bit is left-shifted
        if elem.active_bit is set:
            left_shift(elem.delay_bit)

        // if it reach max delay, bits is reset and priority is set
        if (elem.delay_bit & elem.Max_delay) :
            elem.temp_priority inherit the highest_priority
            reset elem.delay_bit
            reset elem.active_bit

    next_xenif = get_next_xenif()

    // if current xenv was preempted, active_bit is set
    if current_xenvif != next_xenvif :
        set current_xenvif.active_bit
        schedule(next_xenvif)
```

[Fig. 4] Pseudocode of delayed-constrained scheduling

3.3.3 지연시간 관리 비트맵 변화과정

[Fig. 5]는 지연제약 스케줄링을 적용하였을 때의 패킷 처리 과정 예시를 나타낸다. 한 라운드당 패킷 처리 단위인 batch size는 2이고 QoS지연시간 비율은 5:3임을 가정한다.

① Low 우선순위 도메인의 I/O가 처리되고 있을 때, Middle 우선순위의 선점이 발생한 경우다. 또한 pending packet이 남아있으므로 xenif디바이스는 여전히 활성화 상태이며, 다음 라운드부터 비트를 업데이트해야 함을 보이는 첫 번째 비트를 활성화 하고 지연시간 비트를 1로 설정한다.

② 선점 이후 batch size에 도달한 경우이다. Low 우선순위의 선점 후 지연을 카운트하기 위해 2번 비트맵과 같이 업데이트를 수행한다.

③ Middle 우선순위의 xenif 디바이스가 선점되었기 때문에 Middle 우선순위의 비트맵을 활성화하고 지연시간 비트맵을 1로 설정한다. Low 우선순위의 지연시간 비트도 Left-Shift를 수행한다.

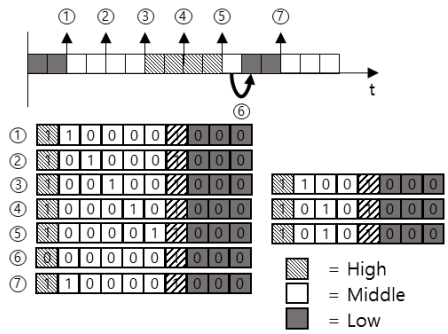
④ 2와 마찬가지로 batch size에 도달하였기 때문에 Low, Middle의 지연 비트가 각각 Left-Shift된다.

⑤ Low 우선순위의 지연시간 비트가 Left-Shift된다. 이 때, 다음 비트와의 and연산이 1이

나오므로 Max_delay에 도달하였음을 알 수 있다. 따라서 Temp_priority가 high_priority로 설정된다. Middle 우선순위 xenif의 경우 High 우선순위의 xenif에 pending packet이 없고 비활성화 상태가 되었으므로 스케줄링 대상이 된 경우이다. 따라서 지연시간 및 활성화 비트는 리셋한다.

⑥ 1라운드동안 Low 우선순위가 일시적으로 High로 바뀌었으므로 다음 라운드에서 우선적으로 스케줄링된다. Temp_priority는 수행되었으므로 1라운드가 끝나면 기존 우선순위로 돌아간다.

⑦ 6에서 기존 우선순위로 돌아갈 때 pending packet이 있으므로 마찬가지로 선점이 시작된 것으로 간주되어 활성화 비트와 지연시간 비트를 1로 설정한다.



[Fig. 5] Example of delay-constrained scheduling

4. 실험 및 평가

4.1 실험 환경 및 방법

4.1.1 실험 환경

본 논문의 실험 환경은 [Table 2]와 같다.

[Table 2] Experimental environment

Classification		Explains
H/W	CPU	Intel Core i5-4670 3.40GHz
	RAM	8.00GB
	NIC	Intel Ethernet Connection I217-V
S/W	OS	Ubuntu
	Xen Version	Xen 4.6 (Para Virtualization)
	Domain 0 kernel version	Linux-3.10
	Guest domain	Linux-3.16 Ubuntu

4.1.2 실험환경 구성

실험은 통합 게임 서버 환경을 가정하여 한 대의 호스트에 3대의 가상머신을 생성하여 진행하였다. QoS는 가상머신의 서비스의 특성에 따라 변경될 수 있으며, 성능 평가를 위해 [Table 3]에 기술된 QoS 제약사항을 고려하였다. 일반적으로 사용되는 통합 서버 환경을 고려하였으며 가상머신1, 2, 3 은 각각 스트리밍 서비스, 텔넷, 음성 메시지 서비스의 트래픽 타입을 가지도록 하였다.

[Table 4]와 같이 QoS의 지연시간 비율을 기준으로 가상머신 서비스에 따라 netif 디바이스의 우선순위를 할당하였다. QoS 평가지수는 패킷 지연시간, 패킷 손실률, 단방향 지연 등으로 분류되며 본 연구에서는 QoS 우선순위 별 지연시간 제약사항 만족정도를 실험한다.

[Table 3] Virtual machine-specific service type and QoS priority

Type of traffic	Type of service	QoS priority
Streaming	Streaming audio and video	1
Voice	Voice message	2
Interactive	E-commerce, Telnet	3

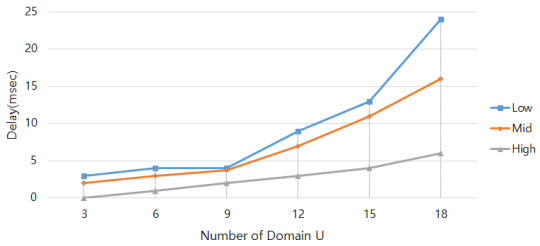
[Table 4] Experimental environment configuration and priority assignment

Classifi-cation	priority netif priority	Packet transmission interval	Type of traffic
domain 0	-	-	-
domain 1 - 3	High 1	10ms	Streaming
domain 4 - 6	Middle 2	50ms	Voice
domain 7 - 9	Low 3	100ms	Interactive Data

4.1.3 실험방법 및 결과

본 연구는 우선순위화된 네트워크 I/O 스케줄링을 통해 공정성을 향상시키고 상대적으로 낮은 우선순위의 QoS를 보장하는 것을 목표로 한다. 지연제약 스케줄링 적용 전 도메인 우선순위별 지연시간 차이를 확인하기 위한 실험을 진행하였다. [Fig.6]은 게스트 도메인들의 수를 증가시키며 도메인 우선순위를 3단계로 나누어 평균 응답시간을 측정한 결과이다. 실험에 사용한 데이터는 일반 문자열 데이터 패킷, 음성 패킷, 비디오 패킷이며, 각 패킷은 동일한 비율로 불규칙하게 보냈고 QoS 우선순위는 각각 3, 2, 1 로 설정되었다. 각 데이터는 실제 게임서버에서 데이터 패킷, 음성 채팅 패킷, 실시간 스트리밍 패킷을 의미한다. CPU사용률은 MD5프로그램으로 50%를 유지하였다.

도메인 우선순위는 3단계로 나누었으며, ICMP 패킷으로 측정하였다. 예를 들면 도메인이 9개일 때, 도메인 1~3은 High, 도메인 4~6은 Mid, 도메인 7~9는 Low이며, netif 디바이스의 우선순위를 1~3으로 설정하여 기존의 선점형 우선순위화된 네트워크 I/O 처리를 수행하도록 하였다. High 도메인은 일반적인 실시간 애플리케이션의 트래픽 패턴을 유지하기 위해 10ms마다 1패킷을 송수신하였으며, Mid와 Low는 각각 50ms, 100ms로 설정하였다. 도메인 수가 증가할수록 VCPU 스케줄링정책에 의해 I/O 지연시간이 증가하며, 도메인 수에 따라 netif 디바이스의 지연시간 차가 커지는 것을 확인할 수 있다.

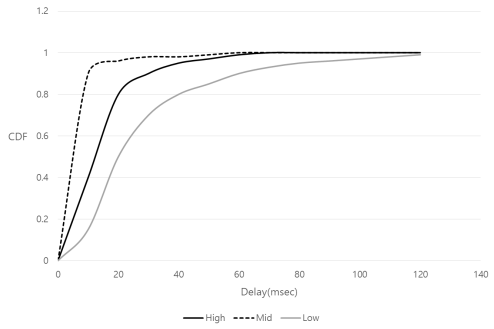


[Fig. 6] Response delay for each number of virtual machines

제안한 기법의 유효성을 판단하기 위해 도메인 개수를 18개로 하여 실험을 진행하였다. ~~[Fig.5]은~~ [Fig.7]은 도메인 개수를 고정시키고 우선순위 네트워크 I/O 처리만 적용하였을 때 각 도메인별 지연시간에 대한 CDF를 나타낸다. 마찬가지로 가상머신 1~3에서 실시간 도메인에서 발생하는 트래픽을 지속적으로 발생시키면서 가상머신 4~9에서 발생하는 지연시간을 측정하였다. 각 단계에 해당하는 같은 우선순위 도메인 3개의 지연시간 평균값을 측정하였다.

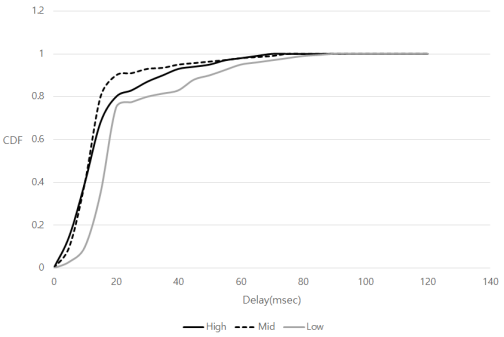
실험결과 High, Mid 단계에 해당하는 도메인의 응답성과 Low 우선순위 단계에 해당하는 도메인 응답성의 차이를 확인할 수 있다. 선정형 네트워크 I/O 스케줄링으로 인해 엄격한 우선순위가 적용된 결과이다.

이는 상위 우선순위 도메인의 지속적인 네트워크 I/O가 지속될 경우 하위 우선순위 도메인의 QoS 보장이 만족되지 않음을 보이며, 적은 우선순위 차이에도 불구하고 응답성과 QoS 보장성에 큰 차이가 있음을 확인할 수 있다.



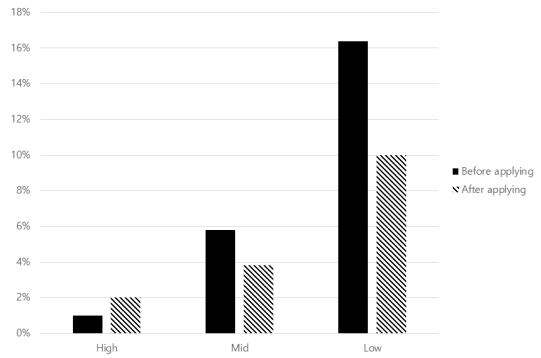
[Fig. 7] Measurement result of the delay before applying the delay-constrained scheduling

다음 [Fig. 8]은 지연제약 스케줄링을 적용하였을 때 게스트 도메인들의 지연시간을 나타낸다. 도메인 각 단계의 우선순위별 응답성은 제안하는 기법 적용 전에 비해 감소하였지만, 비교적 균등하게 분포된 응답성을 확인할 수 있다.



[Fig. 8] Measurement result of the delay after applying the delay-constrained scheduling

또한 지연시간이 60ms 이상으로 발생한 비율을 다음 [Fig. 9]에 나타내었으며, 실험 결과 하위 40%에 해당하는 지연 발생률이 감소하였음을 알 수 있다.



[Fig. 9] 40% lower latency rate

5. 결 론

본 논문에서는 최대 지연시간을 보장함으로써 가상머신 서비스의 QoS와 우선순위별 네트워크 I/O 처리의 공정성을 확보할 수 있는 지연 제약 스케줄링을 제안하였다. 이로서 본 논문에서 제안하는 방법을 통해, 서비스 요소들의 우선순위가 중요한 게임서버에서, 공정성과 속도를 향상시킴으로서 게임 서비스의 완성도와 서버 자원 효율성의 증대로 이어질 수 있다. 기존의 우선 순위화된 패킷 처리 기법은 실시간 게스트 도메인의 응답성 향상을 위해 선점형 스케줄링 기법을 사용하였다. 따라서 높은 우선순위 도메인의 네트워크 I/O 요청이 지속적으로 발생할 경우 적은 우선순위 차이의 도메인에 과도한 지연이 발생하여, 공정성을 해치고, 기아 상태가 발생할 수 있다는 문제점이 있었다.

선점형으로 도메인0에서 이루어지는 Xen의 네트워크 I/O 처리과정을 분석하고, 우선순위화된 패킷 처리에서 발생할 수 있는 기아 상태를 방지하기 위해 비트맵을 이용하여 패킷의 지연시간을 관리하였다. 선점과 스케줄링 단위 만료를 기준으로 비트 연산을 수행하여 최대 지연시간에 도달하였다고 판명되면 스케줄링 단위동안 우선순위 상속을 통해 우선적으로 패킷을 처리하여 기아상태를 방지하였다. 최대 지연시간은 게스트 도메인에서 제공되는 서비스의 QoS를 기반으로 설정하여 공정성

을 확보하였다.

또한 지연시간 보장을 실험하기 위해 기존의 우선순위화된 네트워크 I/O 처리기법과 제안하는 지연제약 스케줄링 기법을 Xen 가상화 환경에서 수행하였다. 실험 결과, 높은 우선순위의 게스트 도메인에서 지속적인 네트워크 I/O 가 발생할 때 손실되는 패킷의 양과 게스트 도메인의 우선순위별 패킷 지연 차이가 감소하여 공정성이 높아진 것을 확인할 수 있었다. 이를 통하여 본 방법을 활용하는 기업은 보다 적은 비용으로 효율적이고 안정적인 서비스를 제공할 수 있다.

그러나 본 논문에서는 Xen의 VCPU 정책에 따라 변할 수 있는 요소들을 동일하게 유지하였을 경우만을 고려하였다. 이는 태스크나 워크로드 특성에 따라 변하는 도메인의 우선순위를 동적으로 반영하지 못한다는 문제점이 있다. 따라서 향후 연구에는 제안한 기법과 Xen의 VCPU 스케줄링 정책의 연관성을 분석하고, 이를 지연제약 스케줄링에 반영하여 최적화 할 수 있는 기법에 대한 연구가 필요하다.

ACKNOWLEDGMENTS

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2013R1A1A2058154)

REFERENCES

[1] Byung-Ki Kim, Kyung-Woo Hur, Young-Woong Ko. Feedback-driven Scheduler for Supporting Dynamic Workloads in a Virtual Machine. Journal of Korean Institute of Information Technology, 10.2 (2012.2): 142-149.

[2] Byung-Ki Kim, Ko Young Woong. Scheduling Mechanism for Supporting Latency Sensitive Domains in Xen Virtualization. JKIIT, Volume 10, Number 11, pp. 135-141. 2012.

[3] Hwantae Kim, Hwangnam Kim. Control Algorithm for Virtual Machine-Level Fairness in Virtualized Cloud Data center. The Journal of The Korean Institute of Communication Sciences, 38.6 (2013.6): 512-520.

[4] Sueng Jae Bea, Bum-Gon Choi, Jin Ju Lee, Sungoh Kwon, and Min Young Chung. A Call Admission Control Algorithm in 3GPP LTE System for Guarantee of Packet Delay. JKIIT, Volume 34, Number 6, pp. 458-467. 2009.

[5] Govindan, S., Nath, A. R., Das, A., Urgaonkar, B., &Sivasubramaniam, A. “Xen and co. : communication-aware cpu scheduling for consolidated xen-based hosting platforms”, In Proceeding of the 3rd international conference on Virtual execution environments. pp. 126-136. ACM. 2007.

[6] XI, S., LI, C., Lu, C., & Gill, C Prioritizing local inter-domain communication in Xen. In Quality of Service (IWQoS), IEEE/ACM 21st International Symposium on pp. 1-10. IEEE. 2013.

[7] Li, C., Xi, S., Lu, C., Gill, C. D., & Guerin, R.. Prioritizing soft real-time network traffic in virtualized hosts based on Xen. In Real-Time and Embeded Technology and Applications Symposium(RTAS), pp. 145-156, IEEE. 2015.

강 기 욱(Kang, Kiwook)

약 력 : 2015 숭실대학교 컴퓨터학과 학사 졸업
2015~ 숭실대학교 컴퓨터학과 석사 과정
관심분야 : 게임 서버, 서버 가상화

백 두 원(Paik, Doowon)

약 력 : Univ. of Minnesota 전산학과 박사
숭실대학교 정보과학대학 글로벌 미디어학부 교수
관심분야 : 게임 서버, 서버 가상화

고 철 홍(Ko, Cherl-hong)

약 력 : 1998 서울대학교 컴퓨터공학과 학사 졸업
2000 서울대학교 컴퓨터공학과 석사 졸업
2015 숭실대학교 컴퓨터학과 박사 수료
관심분야 : 게임 서버, IoT, 빅데이터

홍 지 만(Hong, Jiman)

약 력 : 1999 서울대학교 컴퓨터학과 박사 졸업
2004~2007 광운대학교 컴퓨터학과 교수
2007~ 숭실대학교 컴퓨터학과 교수
관심분야 : 게임 서버, 서버 가상화