논문 2010-05-22

모바일 장치를 위한 iSCSI 프로토콜 기반의 가상 USB 드라이브 설계 및 구현

(Design and Implementation of iSCSI Protocol Based Virtual USB Drive for Mobile Devices)

최 재 현, 남 영 진*, 김 종 완
(Jae-Hyun Choi, Young Jin Nam, JongWan Kim)

Abstract : This paper designs a virtual USB drive for mobile devices which gives an illusion of a traditional USB flash memory drive and provides capacity-free storage space over IP network. The virtual USB drive operating with a S3C2410 hardware platform and embedded linux consists of USB device driver, an iSCSI-enabled network stack, and a seamless USB/iSCSI tunneling module. For performance enhancement, it additionally provides a kernel-level seamless USB/iSCSI tunneling module and data sharing with symbol references among kernel modules. Experiments reveal that the kernel-level implementation can improve the I/O performance up to 8 percentage, as compared with the user-level implementation.

Keywords : Virtual USB drive, Capacity-free, Network storage system

Ⅰ. 서 론

최근 정보통신 기술의 발전으로 유비쿼터스 컴퓨팅이라는 새로운 정보통신 혁명을 야기하게 되었고, 모바일 기술의 발전에 따라 정보를 담는 저장장치의 부피는 점차 작아지는데 저장 및 활용 가능한 정보의 양은 기하급수적으로 증가하고 있다. 이러한 이유로 정보보안과 함께 저장장치의 기능이 매우 중요한 역할을 차지하게 된다. 빠른 속도로 변하기 있는 IT 기술들은 개인 및 기업, 정보 등에서 취급하는 정보의 양을 급속도로 증가시키고 있으며 이에 따라 동일한 시간내에 처리 및 저장해야 할 정보의 양 또한 크게 증가 하고 있다. PC에 장착되는 하드디스크의 용량을 예를 들어 보면, 일반인에게 판매되는 PC의 경우 2006년 평균 1대 당 200-300GB에 이르는 것으로 나타났다. 그러나 5년 전인 2001년에는 이의 1/10 수준인 평균 20-50GB 정도였으며, 이보다 5년 전인 1996년경에는 그의 1/10 수준인 1-5GB 정도였던 것으로 분석되었다 [1].

정보저장 수요는 폭발적으로 증가하는데 그에 상응하는 속도로 정보저장 단가 또한 빠른 속도로 하락하고 있다. 저장장치 기술이 발전하면서 저가격으로 높은 기록밀도의 저장장치를 생산하는 기술이 급진적으로 발전하기 때문이다. 아래 그림 1은 여러 가지의 저장장치에 대하여 정보단가의 하락률을 DRAM과 비교하여 보인 것이다 [2].

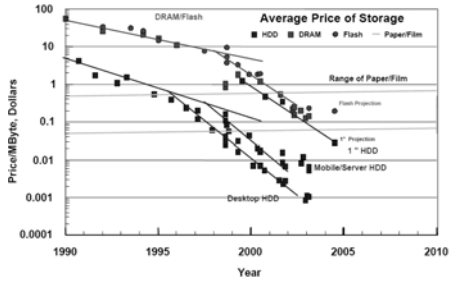


그림 1. 하드디스크 및 반도체 메모리의 정보저장 단가 변화추세

Fig. 1. Variation trend of information storage unit price for hard disks and solid state memory

* 교신저자(Corresponding Author)
논문접수 : 2010. 4. 7., 수정일 : 2010. 08. 30.,
채택확정 : 2010. 09. 26.
최재현 ,남영진, 김종완 : 대구대학교 컴퓨터·IT 공학부
※ 본 연구는 대구대학교 2007년도 교내학술연구비 지원을 받아 수행되었음 (No.20070335).

최근 유비쿼터스 시대에 개인의 정보화를 위하여 모바일 기능을 가지는 초소형 저장장치의 수요가 확대되고 있는 추세이다. 초소형 저장장치는 용량과 크기뿐만 아니라 가격, 속도, 사용자 편이성, 휴대성, 호환성 등의 많은 요소를 만족하여야 한다 [3]. 이러한 조건을 만족하는 초소형 저장장치로 보통 플래시 메모리를 많이 사용한다. 하지만 하드 디스크와 비교 하였을 때 단위 메가바이트당 가격이 비싸다는 문제가 발생하여 그 대안으로 흔히 외장 저장장치를 사용한다. 대표적인 외장 저장장치로는 이동식 메모리인 USB 플래시 메모리와 외장 하드 디스크이며, 최근에는 통신 기술의 발달로 네트워크 스토리지가 일반화 되고 있다. USB 플래시 메모리는 소형이고 휴대가 간편하지만 플래시 메모리를 사용하기 때문에 용량확장이 어렵고 용량에 따른 비용이 많이 소요되는 문제점이 있으며, 외장 하드 디스크는 대량의 저장 공간을 사용할 수 있지만 휴대성이 떨어진다는 문제점이 있다. 네트워크 스토리지는 서버에 연결이 가능한 장소면 언제 어디서든지 데이터 이용과 저장이 가능하고 데이터 보관의 안정성이 높다.

본 논문은 모바일 장치용 저장장치의 제한된 저장 공간과 데이터 소실의 문제점을 해결할 수 있는 모바일 장치용 iSCSI 프로토콜 기반의 가상 USB 드라이브를 제안한다. 또한 일차적으로 설계된 가상 USB 드라이브에서 발생하는 컨텍스트 스위칭으로 인한 데이터 전송 지연을 줄이기 위해 가상 USB 드라이브의 데이터 입출력 개선 기법을 제안한다.

Ⅱ. 배경지식 및 관련 연구

1. USB 인터페이스

USB는 Universal Serial Bus의 약어로 주변기기 인터페이스의 규격화를 목적으로 많은 장점을 가지고 있다. 우선 인터페이스 속도이다. 표 1은 컴퓨터 인터페이스 속도를 비교한 것이다. 기존에 사용하는 인터페이스와 비교하였을 때 빠른 속도를 가지고 있다. 최근 발표된 USB 3.0 표준의 경우 최대 5Gbps 속도를 지원한다. 또 다른 장점으로는 사용의 편리성이 있다. USB는 다양한 종류의 주변기기에 사용하기에 적합하다. 각 주변기기 타입마다 다른 커넥터를 사용하지 않고 한 가지 인터페이스를 여러 타입의 디바이스에 동일하게 사용한다. 또한 USB가 PC에 연결하면 대부분의 운영체제는 자체적으로 USB 디바이스 드라이버를 지원하여 손쉽게 장치가 인식이 된다. 또한 별도 전원이 필요 없이 컴퓨터나 허브의 전원 공급 장치에서 공급되는 전력을 사용하기 때문에 500mA보다 전류를 소모가 적은 주변기기는 별도의 전원 공급 장치 없이 사용이 가능하다 [4,5].

표 1. 컴퓨터 인터페이스 속도 비교
Table 1. Comparisons of computer interface speeds

인터페이스	전송속도
시리얼포트	0.115Mbps
USB 1.1 표준	12Mbps
USB 2.0 표준	480Mbps
Ultra 3 SCSI	640Mbps
IEEE1394	800Mbps
USB 3.0 표준	5Gbps

표 2는 USB의 데이터 전송모드의 특징을 정리한 내용이다. USB는 효율적인 전송을 위하여 4가지 방식의 전송 모드를 사용한다. 등시성(isochronous) 전송은 가끔씩 발생하는 에러를 허용할 수 있고 데이터가 일정한 속도나 지정된 시간에 도착해야만 의미 있는 스트리밍, 실시간 전송에 적합하다. 주로 컴퓨터 캠 카메라에서 사용된다. 벌크(bulk) 전송 [6]은 전송 타이밍이 중요하지 않는 데이터 전송에 유용하다. 다른 전송 방식에게 버스를 양보하고 사용할 수 있을 때까지 기다리기 때문에 버스의 다른 전송을 방해하지 않고 대량의 데이터를 보낼 수 있다. 주로 프린터나 스캐너, 대용량 저장장치에서 사용된다. 인터럽트 전송은 데이터를 정해진 시간 내에 전송해야 할 경우에 유용하다. 일반적으로 키보드, 마우스 등에 사용된다. 컨트롤 전송은 일반적으로 USB 스펙에 정의된 리퀘스트를 전달하거나 호스트가 디바이스 정보를 읽고 설정하는데 사용한다.

표 2. USB 데이터 전송 모드
Table 2. USB data transfer modes

	등시성	벌크	인터럽트	컨트롤
용도	실시간 전송	대용량 전송	정기적 소용량전송	Setup관련 전송
전송주기	1ms	부정기적	1~255ms	부정기적
패킷당 전송량	1~1023 바이트	8/16/32/64바이트	1~64(F) 1~8(L) 바이트	1~64(F) 1~8(L) 바이트
재전송 요구	×	○	○	○

2. iSCSI 기반의 원격 스토리지 시스템 [10]

iSCSI(Internet Small Computer Systems Interface) 프로토콜 [7]은 TCP/IP 네트워크를 통해 SCSI 명령을 전송하는 표준 네트워크 스토리지 프로토콜이다. 이 프로토콜은 스토리지 입출력을 위해 정의된 SCSI 프로토콜을 TCP/IP 기반의 IP 네트워크를 통하여 이루어 질수 있도록 하는 표준 프로토콜이며, 기본적으로 블록 단위의 데이터 전송을 지원한다. iSCSI는 네트워크를 통해 데이터를 전송하고 멀리 떨어진 곳에서도 스토리지를 관리할 수 있으며 호스트에서 기존의 IP 네트워크를 통해 스토리지 요소에 접근이 가능하므로 스토리지를 통합하고 활용도를 높이기가 용이하며 비용을 절감할 수 있다. IP 스토리지는 이러한 iSCSI 프로토콜을 이용하여 기존 SCSI 프로토콜을 IP 네트워크를 통하여 전송함으로써 입출력 작업을 수행한다. IP 스토리지를 통한 입출력 시에 호스트에 동작하는 부분을 IP 스토리지 이니시에이터(initiator)라 하고, 실제 저장장치 측에 존재하는 부분을 IP 스토리지 타깃(target)이라 한다. 이는 기존 SCSI 프로토콜에서 사용하는 용어와 동일함을 알 수 있다 iSCSI 프로토콜을 이용할 경우 원격지에 있는 스토리지를 로컬 디스크와 같이 인식하고 사용이 가능하여 모바일 장치의 경우 저장 공간의 한계를 극복하고 대용량의 저장장치를 지원 가능하다. 일반적인 네트워크 스토리지는 호스트와 스토리지 시스템을 상호 연결하는 별도의 전용 인프라가 필요하지만 iSCSI 기반 IP 네트워크는 기존의 네트워크를 그대로 이용할 수 있어 구축 및 운용비용이 절감하는 장점이 있다[8,9]. 그림 2는 모바일 장치를 위한 iSCSI 기반 원격 스토리지 시스템 구조이다 [10]. 이 스토리지 시스템은 사용자에게 직접 스토리지 서비스를 제공하는 모바일 장치와 스토리지 디바이스를 가진 스토리지 서버로 구성된다. iSCSI 클라이언트 모듈을 가진 모바일 장치는 표준 IP 네트워크를 통해 대용량 스토리지를 가진 스토리지 서버인 iSCSI 서버에 접속이 가능하다. iSCSI 서버는 모바일 클라이언트에 대한 인증 권한을 가진 iSCSI 타깃과 각각의 클라이언트에 대해 스토리지를 할당하는 스토리지 관리자로 구성된다. 모바일 클라이언트는 iSCSI 입출력 모듈을 통해 SCSI 명령어를 iSCSI 서버에게로 전송하고 SCSI 명령어를 수신한 iSCSI 서버는 모바일 클라이언트의 요청에 맞는 블록 단위의 스토리지 데이터를 모바일 클라이언트에게로 전송하는 구조로 되어있다.

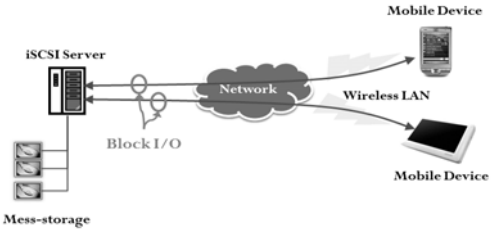


그림 2. 모바일 장치를 위한 iSCSI 기반의 원격 스토리지 시스템 구조 [10]
Fig. 2. Architecture of iSCSI-based remote storage systems for mobile devices

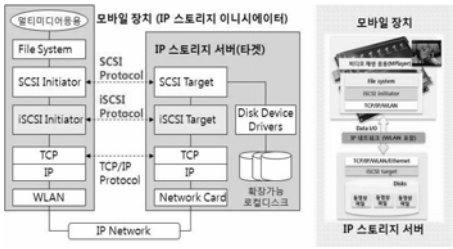


그림 3. iSCSI 기반의 모바일 IP 스토리지 구조 [11]
Fig. 3. Architecture of iSCSI-based mobile IP storage

3. 모바일 IP 스토리지 [11,12]

모바일 IP 스토리지는 배터리 등과 같은 자원 제약성이 매우 높은 PDA 및 스마트폰 환경으로 이식된 IP 스토리지를 의미한다. 모바일 IP 스토리지에서는 기본적으로 무선랜을 이용하며, WiBro와 같이 이동성이 보장된 네트워크 환경에서는 기존 IP 스토리지와 차별화된 스토리지 입출력 서비스를 제공할 수 있다. 모바일 IP 스토리지는 데이터 입출력을 위해서 무선랜을 이용하기 때문에 무선랜 장치의 전력 소모에 대한 효율적인 제어가 해결해야할 매우 중요한 문제 중에 하나이다. iSCSI 기반 원격 스토리지 시스템 구조에서와 같이 모바일 IP 스토리지는 모바일 장치에서 iSCSI 이니시에이터가 동작하고 모바일 장치의 응용 프로그램은 파일 시스템을 통해 데이터의 저장 및 읽기를 명령한다. 파일 시스템은 iSCSI 이니시에이터로 명령을 전달하고 iSCSI 이니시에이터는 받은 명령을 iSCSI 프로토콜에 따라 캡슐화하고 네트워크 장치를 통해 IP 스토리지 서버로 전달한다. IP 스토리지는 iSCSI 타깃을 통해 받은 명령을 해석하고 IP 스토리지 서버의 저

장장치에 접근하여 데이터를 저장하거나 읽어서 다시 모바일 장치의 iSCSI 이니시에이터로 전달한다. 모바일 장치에서 사용하는 실질적인 스토리지는 서버에 존재하며 용량 증설이 용이하기 때문에, 모바일 장치 입장에서 자신이 사용하는 임베디드 스토리지의 크기는 무한대라고 하여도 무방할 것이다. 하지만, 연구 [11,12]에서와 같이 모바일 장치 자체에 iSCSI 프로토콜을 탑재하고 운영할 경우는 모바일 장치 내에 무선랜 하드웨어, 무선랜 장치 구동기, TCP/IP 프로토콜 스택, iSCSI 이니시에니터 프로토콜을 필요로 한다. 하지만, 최근 모바일 장치에 무선랜이 점차 내장되어 가고 있는 추세이긴 하나 디지털 카메라 등과 같은 아직 많은 모바일 장치에서는 무선랜이 장착되어 있지 않다. 따라서, iSCSI 프로토콜을 동작시키기 위해서 필요한 하드웨어 및 소프트웨어를 하나의 모듈 형태로 분리하여 대부분의 모바일 장치에서 제공하는 USB 인터페이스 형태로 제공하는 것이 필요하다고 볼 수 있다. 최근 네트워크가 장착되어 있지 않은 디지털 카메라에 네트워크를 통한 데이터 입출력을 제공하기 위해서 SD카드 형태의 인터페이스를 제공하면서 무선랜이 모듈 형태로 장착된 상용제품(Eye-fi Memory card)가 출시되어 판매되고 있다 [18].

4. USB/IP Project [13]

USB/IP프로젝트는 IP 네트워크를 통하여 서버의 USB 인터페이스로 연결된 주변장치를 공유하는 기법이다. 클라이언트 측은 가상 USB 호스트 컨트롤 인터페이스(VHCI) 드라이버를 통해서 서버측의 스터브 드라이버와 연결되며 VHCI는 서버에 보내는 IP 패킷 안에 주변장치 요구 명령(peripheral request command)을 인캡슐화(encapsulating)해서 네트워크를 통해 전송하고 연결이 완료되면 서버의 USB 장치들에 대한 제어가 가능하다. USB/IP를 활용하여, 사용자는 노트북을 통하여 원격에 있는 컴퓨터의 CD-ROM을 구동시켜 데이터를 읽어 오거나 스피커나 스캐너를 동작시킨다.

Ⅲ. 제안 기법

1. 모바일 장치를 위한 iSCSI 프로토콜 기반 가상 USB 드라이브 구조

제안하는 기법의 구조를 아래 그림에서 보여주고 있다. 모바일 장치를 위한 iSCSI 프로토콜 기반 가상 USB 드라이브는 사용자에게 기존 USB 플래시 메모리와 동일한 인터페이스를 제공하며, 플래시 메모리 대신에 IP 네트워크(WLAN 등) 및 iSCSI 프로토콜을 통해 사용자에게 원격에 존재하는 무한 크기의 저장 공간을 제공한다. 가상 USB 드라이브는 기본적으로 최적화된(빠른 부팅이 지원되는) 임베디드 리눅스를 운영체제로 이용하고 있다. 임베디드 리눅스 상에는 모바일 장치(USB 호스트)와의 인터페이스를 담당하는 USB 디바이스 드라이버, TCP/IP 프로토콜 스택을 이용하여 iSCSI 이니시에이터 프로토콜을 지원하는 iSCSI지원 네트워크 스택(iSCSI-enabled network stack), 그리고 USB 디바이스 드라이버를 통해서 도착한 명령/데이터를 iSCSI지원 네트워크 스택으로 효율적으로 전달하는 심리스(seamless) USB/iSCSI 모듈이 존재한다. 가상 USB드라이브는 USB 인터페이스를 통하여 모바일 장치(휴대폰, PMP, 노트북)와 연결된다. 대부분의 모바일 장치의 OS는 USB 디바이스 드라이버를 지원하여 따로 프로그램이 필요하지 않으며 Plug&play기능을 지원하여 편리하게 USB mass-storage로 인식되어 사용 가능하다.

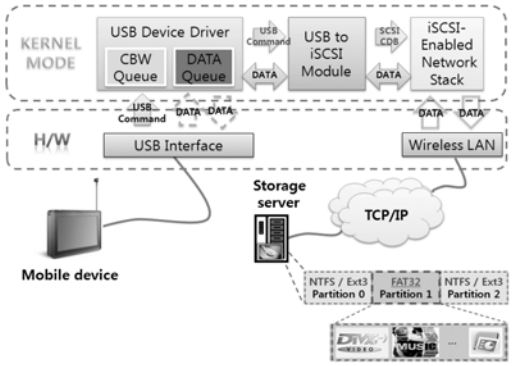


그림 4. 모바일 장치를 위한 iSCSI 기반 가상 USB 드라이브 구조

Fig. 4. Architecture of iSCSI protocol based virtual USB drive for mobile devices

그림 5와 그림 6에서 가상 USB 드라이브와 기존 USB 플래시 메모리 동작과정을 비교하였다. 그림 5에서와 같이 가상 USB 드라이브는 기존 USB 플래시 메모리 스틱과 달리 열거(enumeration) 과정에서 iSCSI 프로토콜을 이용하여 iSCSI 이니시에이터를 통해서 스토리지 서버에 접속하여 인증이 완료되어야만 디바이스 정보를 모바일 장치로 넘긴다. 또한, 읽기/쓰기 명령에 대해서도 iSCSI 프로토

콜을 통하여 원격 스토리지 서버에 명령을 내려 스토리지 서버에서 데이터를 읽거나 저장한다.

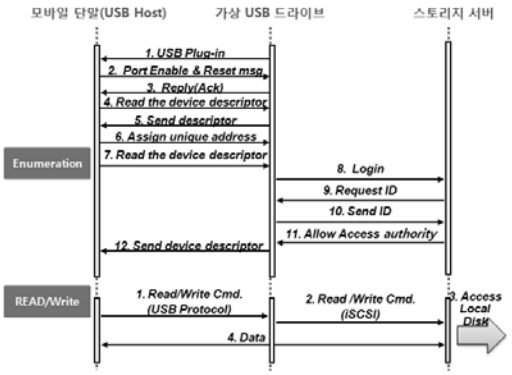


그림 5. 가상 USB 드라이브 동작
Fig. 5. Operations of virtual USB drive

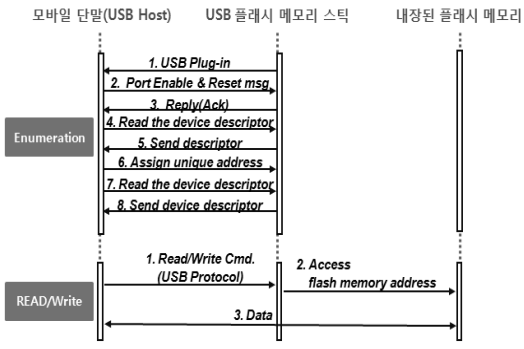


그림 6. USB 플래시 메모리 스틱의 동작
Fig. 6. Operations of USB flash memory stick

가상 USB 드라이브는 전원이 인가되면 빠르게 인식되어 사용 할 수 있는 상태가 되어야하기 때문에, 하부에서 동작하고 있는 임베디드 리눅스 커널의 빠른 부팅이 필수적이다. 빠른 부팅을 위해서 커널을 대상으로 KFI(Kernel Function Instrumentation)[14]와 IP(Instrumented Printks) 툴을 이용하여 부팅시 지연요소를 분석하고 loops_per_jiffy 값 고정, Uncompressed Kernel Image 사용에 따른 압축 해제 시간 단축, RootFilesystem 체크 회피, 커널 메시지(printk) 미출력을 통하여 부팅 속도를 16.52초에서 7.97로 단축하였다 [15,16].

그림 7은 가상 USB 드라이브의 모듈 사이의 동작 구조를 보여준다. 심리스 USB-iSCSI 모듈은 USB 디바이스 드라이버와 iSCSI지원 네트워크 스택 사이에서 SCSI CDB(Command Descriptor Block) 및 데이터의 전달을 담당한다. 일반적인 USB 플래시 메모리의 경우 펌웨어가 USB 디바이스를 구동하고 SCSI CDB를 처리하여 플래시에 읽기/쓰기를 한다. 그러나 가상 USB 드라이브의 경우 플래시 메모리가 아닌 iSCSI지원 네트워크 스택을 통하여 SCSI CDB가 처리되어야 하므로 USB 호스트에서 받은 CDB를 iSCSI 프로토콜로 변환하는 과정을 거쳐 스토리지 서버에 전달할 수 있어야한다. 이러한 기능을 심리스 USB/iSCSI 모듈에서 담당한다. 일차적으로 기능검증을 위해서 설계된 심리스 USB/iSCSI 모듈은 가상 USB 드라이브가 대기 상태가 된 후 유저모드에서 폴링하면서 USB 디바이스 드라이버에 큐잉된 CBW가 있는지 확인한다. CBW가 있는 것이 확인되면 CBW에서 실행할 명령을 확인하고 쓰기 명령이라면 USB 디바이스 드라이버의 데이터 큐에서 데이터를 읽어와 명령과 함께 iSCSI지원 네트워크 스택으로 전달한다. 데이터 읽기 명령이면 스토리지 서버로부터 읽어온 데이터를 바로 USB 디바이스 드라이버로 전달한다. USB 디바이스 드라이버는 USB 호스트에서 발생하는 인터럽트를 감지하여 USB 디바이스 열거 과정을 처리하고 USB 대용량 스토리지 명령을 CBW 큐에 저장하고 쓰기 명령시에는 데이터 큐에 저장할 데이터를 넣어준다. USB 플래시 메모리 스틱의 USB 디바이스 드라이버와 비교하였을 경우 플래시 메모리 컨트롤 부분만을 제외하고 동일한 동작을 수행한다. iSCSI지원 네트워크 스택은 스토리지 서버로 데이터를 송수신하기 위한 네트워크 스택으로 iSCSI 이니시에터 기능을 수행한다. 심리스 USB/iSCSI 모듈로부터 받은 SCSI CDB와 데이터를 네트워크를 통하여 스토리지 서버로 전달하고 서버로부터 수신한 데이터 및 명령에 대한 응답을 심리스 USB/iSCSI 모듈로 전달한다.

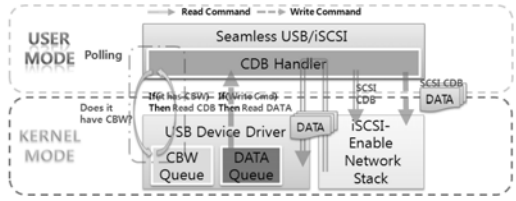


그림 7. 가상 USB 드라이브 내의 모듈간 상호동작
Fig. 7. Interactions among the modules within virtual USB drive

2. I/O 성능 향상 기법

일차적으로 설계된 가상 USB 드라이브는 유저 영역에서 시스템 콜을 통하여 디바이스를 제어하는 형태로 불필요한 데이터 이동이 많고, 리눅스 커널에서 제공하는 시스템 함수를 통한 제어 부분이 많아 I/O 성능 측면에서 최적화가 이루어지지 못하였다. 이로 인하여 가상 USB 드라이브의 데이터 처리 시 지연이 발생한다. 이러한 지연을 방지하기 위하여 심리스 USB/iSCSI 터널링과 모듈간 심볼 참조에 의한 데이터 공유 기법을 통하여 I/O 성능의 최적화를 진행하였다. 그림 8은 성능 향상 기법을 적용 후 가상 USB 드라이브 구조이다. USB 디바이스 드라이버와 iSCSI지원 네트워크 스택은 기능적 측면에서는 크게 달라진 것이 없다. 심리스 USB/iSCSI 모듈이 성능 향상 기법을 거치면서 확장되어 USB-to-SCSI, SCSI-to-USB 프로토콜로 변환 기능 및 명령/데이터 전달을 담당하고 USB 디바이스 제어시 커널 시스템 함수를 거치지 않고 직접적으로 USB 디바이스 드라이버 내부 함수를 호출하여 하드웨어를 제어한다

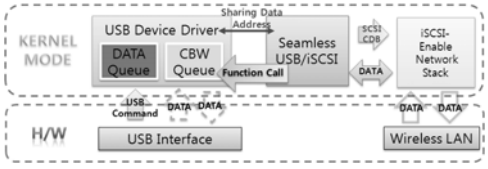


그림 8. 성능 향상 기법 적용된 가상 USB 드라이브 구조

Fig. 8. Architecture of virtual USB drive with the performance enhancement technique

심리스 USB/iSCSI 터널링은 유저모드에서 구동하는 심리스 USB/iSCSI 모듈을 커널 모드에서 동작하도록 변경하고 리눅스 시스템 함수를 사용하지 않고 USB 디바이스 드라이버와 iSCSI지원 네트워크 스택 사이를 제어하는 방법이다. 심리스 USB/iSCSI 모듈은 USB 인터페이스를 통하여 원격의 디스크로 데이터 써넣는 처리 과정에서 그림 9와 같이 매번 시스템 콜을 호출해 복사를 해야 하는 비효율적인 문제가 있었다. 즉, 먼저 커널이 USB 호스트에서 발생하는 명령 또는 데이터를 읽어 커널-사용자 간 경계를 넘어 심리스 USB/iSCSI 모듈에 복사한다. 이후 심리스 USB/iSCSI 모듈은 해당 명령 또는 데이터를 iSCSI지원 네트워크 스택 버퍼에 써 넣기 위해 다시 커널-사용자 간 경계를 넘어 커널로 밀어 넣는다. 결과적으로 심리스 USB/iSCSI 모듈은 USB 호스트에서 명령 또는 데이터를 가져다가 iSCSI지원 네트워크 스택 버퍼로 옮기는 비효율적인 중개자인 셈이다.

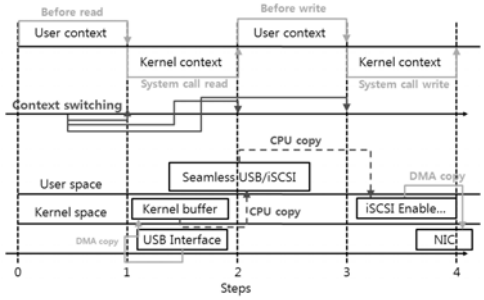


그림 9. 성능 향상 기법 적용 전 가상 USB 드라이브 데이터 복사 방식

Fig. 9. Data copying method of virtual USB drive without the performance Enhancement technique

그림 9를 살펴보면 2개의 CPU Copy는 실제 필요하지 않음을 알 수 있을 것이다. 심리스 USB/iSCSI 모듈은 그저 명령을 또는 데이터를 캐시 했다가 버퍼로 보낼 뿐 아무런 일도 하지 않는다. 게다가 복사를 진행하는 동안 컨텍스트 스위칭(context switching)과정이 3번이나 일어났다. 이러한 동작이 이루어지는 이유는 유저 영역에서 디바이스를 제어하기 때문이다. 리눅스 커널은 커널 영역의 보호 위하여 유저 영역에서는 시스템 콜을 통해서만 접근이 가능하도록 설계 되었다. 하지만 커널 영역에서 제어를 한다면 그림 10과 같이 시스템 콜을 거치지 않고 모듈 간의 데이터 전송을 할 수 있다.

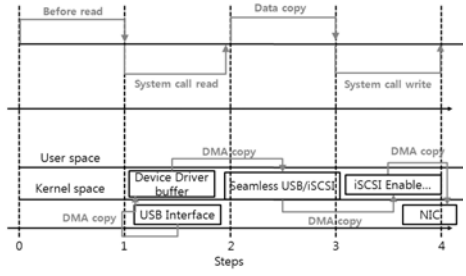
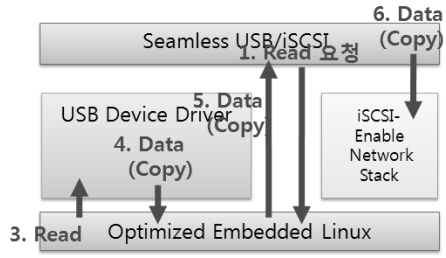


그림 10. 개선된 데이터 복사 방식

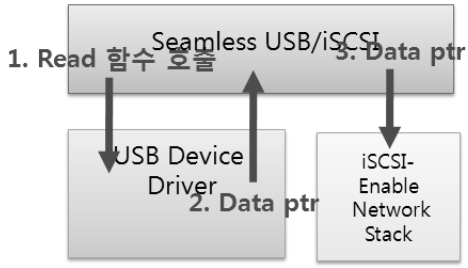
Fig. 10. Improved data copying method

모듈간 심볼 참조에 의한 데이터 공유 기법은 함수 호출시 포인터를 통한 주소값 전달을 통하여

각 모듈의 버퍼에 해당하는 주소 값을 넘겨받아 복사 없이 읽어 가는 방법이다. 그림 11은 기존 방식과 모듈간 심볼 참조에 의한 데이터 공유 방식에 대한 차이를 간략하게 나타낸 것이다.



(a) 기존의 데이터 전송 방식



(b) 개선된 데이터 전송 방식

그림 11. 성능 향상 기법 적용 전과 적용 후 가상 USB 드라이브 동작 비교

Fig. 11. Comparisons of virtual USB drive operations with and/or without the performance enhancement technique

성능 향상 기법 적용 전의 경우 데이터 읽기 요청 후 시스템 콜을 통하여 데이터를 읽어 온다. 데이터 복사는 3번 발생하였다. 제안 기법을 적용 후 심리스 USB/iSCSI 모듈은 USB 디바이스 드라이버 내의 함수를 직접 호출하며 응답으로 해당하는 데이터의 주소를 포인터로 받아온다. 데이터 복사는 발생하지 않았다. 그림 12는 개선된 기법을 통해 데이터를 읽었을 때 나타나는 동작 구조이다. 실제적인 데이터 복사는 하드웨어에서 데이터를 읽어오거나 하드웨어에 저장할 때를 제외하고 발생하지 않음을 볼 수 있다.

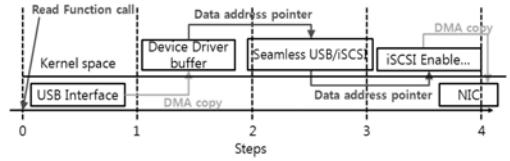


그림 12. 모듈간 심볼 참조에 의한 데이터 공유 방식

Fig. 12. Data sharing method with symbolic references among modules

Ⅳ. 성능 평가

1. 성능평가 환경 구축

프로토타입 형태로 구현된 가상 USB 드라이브의 H/W 및 S/W 환경을 표 3에 요약하였다. 하드웨어의 특성상 USB 호스트/디바이스는 1.1버전을 사용하였고, 네트워크는 USB 동글 형태의 무선랜과 10Mbps 이더넷을 지원하며 무선랜 사용시 하드웨어에 장착된 USB 호스트의 낮은 버전으로 인하여 10Mbps의 이더넷을 대신 사용하였다. (USB 디바이스는 가상 USB 드라이브 호스트와의 연결을 위해 그대로 이용함) iSCSI 이니시에터 모듈은 인텔에서 제공되는 레퍼런스 코드를 사용하였다.

표 3. 가상 USB 드라이브 H/W 환경

Table 3. H/W environment of virtual USB drive

H/W	CPU	S3C2410(ARM 920T)
	USB	USB 디바이스 1.1
	네트워크	이더넷 10Mbps IEEE802.11g/n
S/W	운영 체제	임베디드 리눅스 2.6.18
	iSCSI 참조 코드	Intel-iscsi-2.0.17
	성능 측정 도구	IOmeter 2006.07.27, HDparm v9.12, fileop 1.53

또한, iSCSI 스토리지 서버 환경을 표 4에 요약하였다. 서버 환경은 충분이 성능이 높은 일반 PC에 리눅스를 탑재하였으며, iSCSI 엔터프라이즈 타깃과 인텔의 참조 코드를 이용하였다.

표 4. 스토리지 서버 환경
Table 4. Storage server environment

H/W	CPU	인텔 펜티엄 4 3.0GHz
	메모리	SDRAM 1GB
	네트워크	이더넷 100Mbps
S/W	운영체제	CentOS 5.2(커널 2.6.18.8)
	IP 스토리지 타깃	iSCSI Enterprise Target 0.4.17 Intel-iscsi-2.0.17

2. 가상 USB 드라이브 블록 수준의 입출력 성능 분석

그림 13은 IOmeter [17] 저장장치 벤치마크 툴을 이용하여 가상 USB 드라이브의 블록단위의 입출력 성능을 분석한다. 성능 향상 기법 적용 전과 적용 후의 성능을 각각 측정하였으며, 일반적으로 파일 시스템에서 많이 발생하는 4KB 크기로 읽기와 쓰기를 50:50 비율로 실행했을 경우에 IOPS(Input/Output Operations Per Second), 평균 응답 시간, 그리고 최대 응답 시간을 측정하였다. 최적화 기법 적용을 통하여 IOPS는 120에서 130.4로 증가하였고, 평균 응답 시간은 8.6μs, 최대 응답 시간은 2.1μs로 감소하였다.

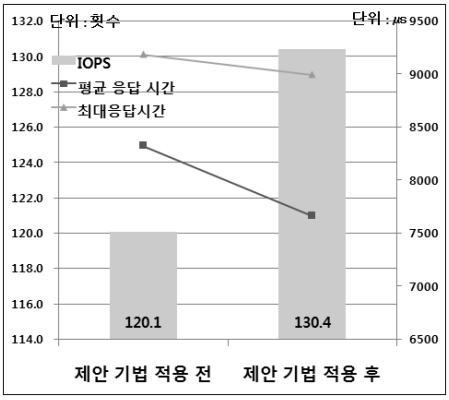


그림 13. 가상 USB 드라이브 입출력 성능(IOPS, 평균/최대 응답 시간)
Fig. 13. I/O performance of virtual USB drive(IOPS, avg/max response times)

그림 14는 가상 USB 드라이브에 대해서 서로 다른 입출력 크기로 (랜덤) 읽기 및 쓰기 작업을 하였을 경우에, 1KB 당 평균 응답 시간을 나타내었다. 전체적으로 입출력 프로토콜 오버헤드로 인해서 입출력의 크기가 커질수록 1KB 당 평균 응답시간이 줄어들고 있음을 볼 수 있다. 성능 향상 기법 적용 여부에 따라서, 읽기의 경우 큰 차이를 보이지 않았지만 쓰기의 경우에는 성능 향상 기법 적용 후에 전송 사이즈가 4KB인 경우를 제외하고 100μs 정도씩 응답속도가 개선됨을 볼 수 있다.

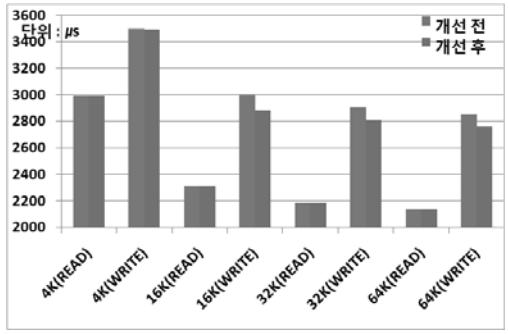


그림 14. 가상 USB 드라이브의 입출력 사이즈별 평균 응답시간 (1KB당)
Fig. 14. Average response time (per 1KB) of virtual USB drive with different I/O sizes

3. 가상 USB 드라이브 파일 수준의 입출력 성능 분석

그림 15는 파일단위의 입출력 성능을 측정하는 Fileop(IOzone) 벤치마크 툴을 이용하여 가상 USB 드라이브의 파일 입출력 측정한 결과이다. 성능 측정을 위해서 가상 USB 드라이브 상에 FAT32 파일 시스템을 설치하여 이용하였다.

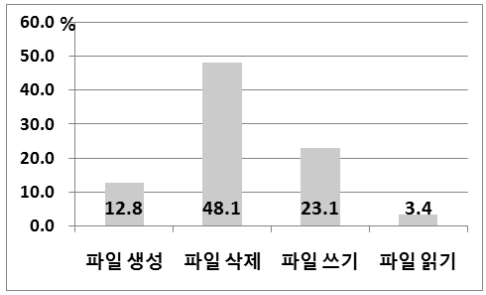


그림 15. 성능 향상 기법 적용후 파일 생성, 삭제, 읽기, 쓰기 속도의 개선율 (%)
Fig. 15. Speed improvement rate (%) of file creates, deletes, reads, writes with the performance improvement technique

성능 측정시 한번에 생성하는 파일 크기는 512KB이고 생성하는 디렉토리 수는 155개이다. 성능 측정결과 가상 USB 드라이브(성능 향상 기법 적용 전)의 파일 생성시 소요되는 시간은 25μs이며 성능 향상 기법 적용 후는 22μs로 향상되었다. 또한, 파일 삭제 시에는 356μs에서 240μs로 향상되었으며, 파일 쓰기 시에는 837μs에서 680μs로 향상되었으며, 파일 읽기 시에는 1100μs에서 1064μs로 단축되었다. 제안된 성능 향상 기법을 적용하여 성능을 분석해 본 결과 전체적으로 약 8% 정도의 성능 개선이 있음을 확인할 수 있었다.

4. 가상 USB 드라이브 입출력 구간별 성능 분석

그림 16은 HDparm 벤치마크 툴을 이용하여 가상 USB 드라이브(성능 향상 기법이 적용된 경우)가 1MB를 데이터를 전송할 때에 구간별 데이터 전송 성능을 측정한 결과이다.

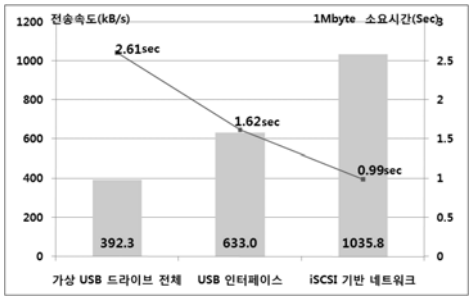


그림 16. 가상 USB 드라이브의 구간별 입출력 성능 분석 (1MB 데이터 전송시 소요시간)
Fig. 16. Analysis of per-region I/O performance within virtual USB drive (required time for 1MB data transfer)

USB 드라이브의 데이터 전송시 지연시간은 USB (디바이스) 인터페이스 지연 시간, USB 인터페이스와 iSCSI기반 네트워크 사이 전송 지연 시간, iSCSI기반 네트워크상 지연 시간의 합으로 구성된다. 시험결과 가상 USB 드라이브 전체 성능은 392KB/s 정도로 측정되었으며, USB 인터페이스의 속도는 633KB/s, 그리고 iSCSI 네트워크 속도는 약 1MB/s 정도로 측정되었다. USB 1.1 디바이스 인터페이스의 성능은 아래의 표에서와 같이 상용 플래시 메모리 성능에 필적하도록 USB 1.1 표준에서 제시하는 이론적인 최대속도의 약 41% 정도를 만족함을 볼 수 있었다. 또한, iSCSI 네트워크의 성능도 10Mbps 성능에 근접하는 매우 높은 성능을 확인할 수 있었다.

표 5. USB 2.0과 USB 1.1 표준속도 대비 측정된 인터페이스 속도
Table 5. Measured interface speeds considering USB 2.0 and USB 1.1 standard speeds

구분	10Mbps Ethernet	54Mbps (802.11g)	100Mbps Ethernet
iSCSI 네트워크 성능(KB/s)	1,035	1,523	2,969
가상 USB 드라이브 성능(KB/s) (USB2.0 표준시 예측)	642	944	1,840

Ⅴ. 결론

본 논문에서는 가상 USB 드라이브와 그 I/O 성능 향상 기법을 제안하였다. 가상 USB 드라이브는 iSCSI 프로토콜을 기반으로 대용량 스토리지를 제공한다. 하지만 유저 영역에서 모든 기능을 제어하여 커널과 유저 영역간의 컨텍스트 스위칭으로 인한 데이터 전송 지연 부분이 있었다. 데이터 전송 지연 부분을 개선하기 위해 커널 수준에서의 심리스 USB/iSCSI 터널링을 통하여 문제를 해결하였고, 보다 향상된 성능을 위하여 모듈간 심볼 참조에 의한 데이터 공유 기법을 제안하였다. 심리스 USB/iSCSI 터널링은 유저 영역에서 실행되던 심리스 USB/iSCSI 모듈을 커널 영역에서 실행 가능하도록 변경하였다. 따라서 컨텍스트 스위칭으로 인한 성능 저하를 개선하였고 모듈간 심볼 참조에 의한 데이터 공유 기법은 USB 디바이스 드라이버 모듈과 iSCSI지원 네트워크 스택간의 주소 포인터를 이용하여 데이터 버퍼를 공유하는 방식이다. 제안된 성능 향상 기법을 구현하여 성능을 분석해 본 결과 성능 향상 기법 적용 전 성능에 비하여 8% 정도 개선되었다. 개선 효과가 다소 낮은 이유는 구현한 H/W 플랫폼이 USB 디바이스 1.1을 지원하여 USB 인터페이스 속도에서 많은 제약을 받아 성능 개선 효과가 뚜렷하게 나타나지 않은 것으로 분석된다.

참고문헌

[1] 백문철 & 강광용, 테라스토리지 기술의 동향 분석, 전자통신동향분석 제21권, 제6호, 2006년.

[2] E. Grochowski, "Average price of storage, Hitachi Global Storage Technologies", San Jose Research Center, 2003.

[3] 강광용 외, 초소형 초대용량 모바일 장치, ETRI, 2006.

[4] J. Axelson, USB Mass Storage, Manning, 2006.

[5] J. Axelson, USB Complete: Everything You Need to Develop Custom USB Peripherals 3rd edition. Lakeview Research, 2005.

[6] USB, Universal Serial Bus Mass Storage Class Bulk-Only Transport revision 1.0, www.usb.org/ developers/devclass_docs/.

[7] J. Hufferd, iSCSI: The Universal Storage Connection. Addison-Wesley, 2002.

[8] K. Math, J. Satra, "Design of the iSCSI protocol", Proceedings of the Mass Storage Systems & Technologies/20th IEEE/11th NASA Goddard Conference, 2003.

[9] T. Clark, IP SANS-A Guide to iSCSI, iFCP, and FCIP Protocols for Storage Area Networks. Addison-Wesley, 2002.

[10] 박수라 외, "모바일 기기를 위한 iSCSI 기반의 원격 스토리지 시스템 설계 및 구현", 한국정보처리학회 자료저장시스템연구회 춘계학술대회 논문집, 2003년 7월.

[11] Y. Nam, "Prototyping object-based ubiquitous multimedia contents storage for mobile device", Lecture Notes in Computer Science, Vol.4159, pp. 93-102, 2006.

[12] 남영진, 최민석, "모바일 IP 스토리지 상에서 멀티미디어 컨텐츠 실행을 위한 효율적인 무선 랜 장치 전력제어 기법", 한국정보처리학회 논문지, 2009년 4월.

[13] T. Hirofuchi et al., "USB/IP: A Transparent Device Sharing Technology over IP Network", IPSJ Transactions on Advanced Computing Systems, Vol.46 No.SIG.1, 2005.

[14] Kernel Function Instrumentation, http://elinux.org/Kernel_Function_Instrumentation/.

[15] LinuxTiny, http://www.selenic.com/linux-tiny/.

[16] M. Mackall, "Linux-tiny and directions for small systems", Proc. of the Linux Symposium, Jul. 2004.

[17] http://www.iometer.org/

[18] http://www.eye.fi/

저 자 소 개

최 재 현

2008년 : 대구대학교 컴퓨터·IT 공학부 학사.
2010년 : 대구대학교 컴퓨터정보공학과 석사.
관심분야 : 임베디드시스템, 모바일네트워킹.
Email : jhchoi@mmctech.com

남 영 진

1992년 : 경북대학교 전자공학과 학사.
1994년 : 포스텍 전자전기공학과 석사.
2004년 : 포스텍 컴퓨터공학과 박사.
1994~1998년 한국전자통신연구원 연구원.
2001년 미국 IBM Almaden연구소 방문연구원.
현재, 대구대학교 컴퓨터·IT공학부 부교수.
관심분야 : 임베디드 시스템, 스마트 저장장치.
Email : yjnam@daegu.ac.kr

김 종 완

1987년 : 서울대학교 컴퓨터공학과 학사.
1989년 : 서울대학교 컴퓨터공학과 석사.
1994년 : 서울대학교 컴퓨터공학과 박사.
2006~2007년 미국 U. of Oregon 방문교수.
현재, 대구대학교 컴퓨터·IT공학부 교수.
관심분야 : 인공지능, 데이터마이닝, IT융합.
Email : jwkim@daegu.ac.kr