Regular paper http://dx.doi.org/10.13067/JKIECS.2015.10.2.211

Intel DPDK를 이용한 가상스위치의 구현에 관한 연구

정갑중* · 최강일**

Study on the Implementation of a Virtual Switch using Intel DPDK

Gab-Joong Jeong* · Kang-Il Choi**

요 약

본 논문에서는 클라우드 컴퓨팅 서비스를 위한 가상 네트워크 구축에 필요한 중요 구성 요소 중의 하나인 가상스위치를 Intel DPDK(Data Plane Development Kit)를 이용하여 구현하고 DPDK에서 지원하는 가속화 가상스위치의 기능을 테스트 및 검증하였다. 최근 사물인터넷 등에서와 같이 지능형 IT 시스템들의 인터넷 접속을 통한 새로운 정보 서비스 플랫폼의 출현과 그에 대한 활용 및 응용 서비스를 기업 비즈니스에 적용하고자 하는 기업이 많아지고 있다. 기업들은 매우 빠른 소비자 환경변화에 민첩하게 대응할 수 있는 신규 서비스에 대한 사용자 입장에서의 클라우드 컴퓨팅 활용 측면으로 적은 비용으로 신규 서비스를 빠른 시일 내에 적용할 수 있다. 본 연구에서는 창조적 서비스 사업에 활용할 수 있는 스마트 클라우드 플랫폼 서비스를 효율적으로 구현하기 위한 고속 가상스위치를 Intel DPDK를 이용하여 구현하고 그의 기능 검증에 대한 연구를 수행하였다.

ABSTRACT

This paper describes the implementation of the accelerated virtual switch using Intel DPDK(Data Plane Development Kit), and evaluates the virtual network functions of the virtual switch which is one of the most important components to build a virtual network for cloud computing. Nowadays, new information service platforms are appeared from the interconnection of intelligent IT systems like IoT(Internet of Things). And many companies want to use the new service platform for their new application service. The companies can apply there new service early which needs small investment and responses adaptively to the fast change of consumer environment. Using cloud computing technology, the new business service can be introduced as a commercial IT service for the time to market. In this study, an implementation and investigation were performed for the accelerated virtual switch, called Intel DPDK virtual switch, which is using multi processors in network interface card for virtual network functions. It can be useful for Internet-oriented companies to leverage the new cloud service and businesses for its creativeness.

키워드

Accelerated Virtual Switch, Cloud Platform Service, Intel DPDK, Virtual Machine
가속화 가상 스위치, 클라우드 플랫폼 서비스, 인텔 DPDK, 가상 머신

Ⅰ. 서 론

미래 인터넷 환경은 다양한 종류의 운영체제와 여러 가지 목적의 컴퓨터 시스템이 가상 네트워크에 통합 연결되어 대량의 데이터를 처리하는 가상의 호스트 머신들로 이루어진다. 이러한 가상의 호스트 머신들은 VM(Virtual Machine) 상에서 구동되는 가상 서버로 이루어지고 각 가상 서버들은 하나의 물리적 컴

* 교신저자(corresponding author) : 경주대학교 관광정보학과(gjjeong@gju.ac.kr)
** 한국전자통신연구원(forerunner@etri.re.kr)
접수일자 : 2014. 12. 30 심사(수정)일자 : 2015. 01. 16 게재확정일자 : 2015. 02. 09

퓨터 자원을 공유하고 서로 다른 서버로써 각각의 서비스를 원활하게 지원하는 인터넷 환경에서 동작하게 된다. 또한 가상의 호스트를 사용하는데 있어서 물리적 자원을 공유하는 개념을 도입하여 자신 만의 물리적 자원을 구입 및 설치하지 않고 필요시에 필요한 만큼만 빌려 쓰는 형태의 원격 서비스로 발전하게 된다. 이를 이용하는 기업들에게 있어서는 저렴한 비용으로 큰 투자 효과를 얻을 수 있으므로 효율적 기업 경영을 위해 이러한 새로운 인터넷 환경을 도입하고자 하는 기업이 매우 빠르게 증가하고 있다.

또한 지능형 IT 시스템들의 인터넷 접속과 사물 인터넷 등의 등장과 같은 새로운 정보기기의 출현과 더불어 그에 대한 활용 및 응용 서비스를 새로운 기업 비즈니스 및 창조적 기업경영에 적용할 수 있는 분야가 해가 갈수록 다양해지고 응용 범위가 넓어지고 있다. 사용자 입장에서의 기업 경영자들은 적은 투자를 활용해 신규 서비스를 빠른 시일 내에 적용할 수 있고 향후 성공 여부 및 확장 필요성에 따라 시장 변화에 매우 민첩한 기업 경영환경이 될 수 있다.

본 연구에서는 새로운 기업경영 및 서비스 사업에 활용할 수 있는 클라우드 서비스를 지향하는 리눅스 기반 인터넷 환경과 그러한 환경에 필수적인 가속화 가상스위치(Accelerated Virtual Switch)의 구현 및 동작 환경에 대해 조사 및 분석 연구를 수행하였다[1-6].

이어지는 제2장에서는 클라우드 서비스를 위한 서버 가상화관련 기술을 소개하고 제 3장에서는 Intel DPDK를 이용한 가상네트워크에 대한 구조적 분석 내용을 정리한다. 제 4장에서는 본 연구에서 시험 및 개발에 사용한 오픈소스 가속화 가상스위치인 Intel (DPDKvSwitch)의 구현과 그에 따른 절차 및 방법을 정리하였다. 제 5장을 끝으로 결론 및 향후 연구 계획을 정리한다.

II. 가상화

본 연구에서는 가상 머신의 운영체제로 오픈소스 운영체제인 리눅스 운영체제를 사용하였다. 리눅스 운영체제는 기본적으로 시스템 커널과 프로세스로 구성된다. 일반적으로 커널을 수행하는 CPU의 동작모드를 커널모드라 하고 사용자 프로세스가 처리되는 동작모드를 사용자 모드라 한다. 프로세스에서 입출력 장치와 데이터를 교환하고자 하는 경우나 커널 내의 기능을 사용하고자 하는 경우 역시 커널 모드로 전환하여 커널 모드에서 처리된다. 따라서 항상 프로세스와 커널 사이에는 모드 전환이 이루어지면서 시스템이 운영된다.

사용자 어플리케이션에서 커널의 자원 또는 디바이스로 접근하기 위해서는 시스템 콜을 사용한다. 따라서 사용자 프로그램에서 사용되는 모든 디바이스의 제어, 시스템 프로그램 실행, 파일 전송 등은 시스템 콜로 구성된다. 응용 프로그램에서 사용되는 시스템 콜 함수는 표준 C 라이브러리에 구현되어 있다. 이 표준 라이브러리의 시스템 콜 함수는 소프트웨어 인터럽트를 호출하여 커널의 소프트웨어 인터럽트 핸들러를 실행한다. 커널에 구현되어 있는 소프트웨어 인터럽트 핸들러는 해당하는 시스템 콜 요청에 따라 사용자 프로그램에서 요구한 처리를 한다. 따라서 모든 커널 기능이나 입출력 장치를 제어할 수 있게 된다. 리눅스 시스템의 동작 구조를 그림 1에 나타내었다. 또한 게스트 운영 체제로 사용한 리눅스 커널에서는 하드웨어 접근 요청이 발생하면 하드웨어를 공유할 수 있도록 하이퍼바이저가 실제 하드웨어를 제어하여 각 게스트 운영체제가 독립적인 전용 하드웨어를 구동하는 것 처럼 보여지게 한다.

이러한 리눅스 운영체제를 근간으로 하는 클라우드 컴퓨팅을 위한 기술들을 구분하면 물리적 컴퓨터를 가상 컴퓨터로 전환하는 가상화 기술과, 분산 병렬 처리 기술, 분산 데이터관리 프로비저닝의 자동화 기술, 다중 공유 기술, 보안기술, 및 클라우드 서비스의 관리 기술 등으로 나누어진다. 그 중에서 가장 기본적이면서 핵심 요소 기술이 가상화 기술이다. 이는 하드웨어적인 컴퓨팅 자원을 추상화하여 여러 가상 머신들이 유한한 하드웨어 자원을 공유하여 사용하는 기술이다. 가상화를 이용하는 영역으로는 서버 가상화, 데스크탑 가상화, 네트워크 가상화, 스토리지 가상화 등이 있다[7-11].

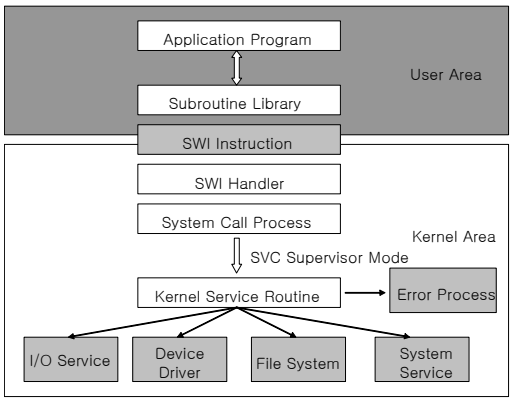

그림 1. 리눅스 시스템 동작 구조
Fig. 1 Linux system operation structure

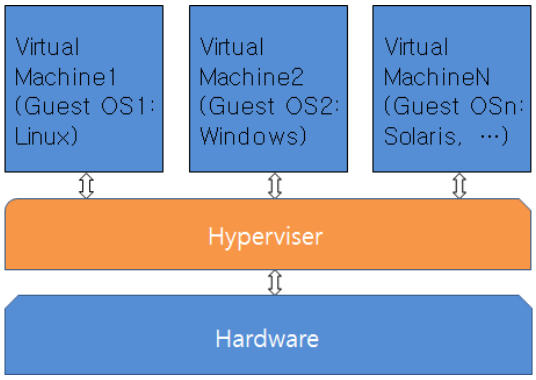

그림 2. 하이퍼바이저 가상화
Fig. 2 Hypervisor virtualization

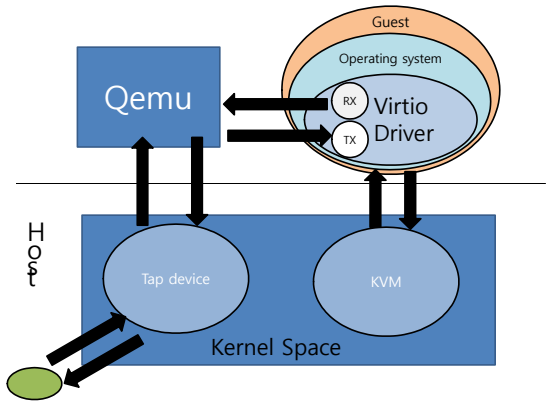

그림 3. Intel DPDK 이용한 하이퍼바이저 가상화
Fig. 3 Hypervisor virtualization using Intel DPDK

물리적 자원을 가상화하여 여러 응용 시스템들이 논리적으로 공유할 수 있도록 하는 가상화 기술을 구현 가능하게 하는 방법으로 가장 많이 사용되고 있는 것은 하이퍼바이저 가상화이다. 하이퍼바이저 가상화는 하나의 컴퓨터에서 여러 가지 종류의 운영체제들을 독립적으로 운용 가능하게 해주는 가상 플랫폼이다. 하이퍼바이저 가상화의 아키텍처는 그림 2에 나타내었다. 그림에서와 같이 하이퍼바이저 가상화는 서로 다른 운영체제의 독립적 실행과 주어진 하드웨어 자원들을 각각의 운영체제가 실행된 가상 머신에서 직접 자신의 하드웨어처럼 동작시킬 수 있도록 제어 요구에 대한 제어를 하고 일부 성능 저하를 일으키는 면도 존재하지만 하나의 자원을 여러 가상 머신이 공유한다는 측면에서는 어느 정도 감수할 수밖에 없는 구조를 가지고 있다.

본 연구에서는 이러한 하이퍼바이저 가상화를 실현하기 위한 하이퍼바이저로써 많이 사용되는 오픈 소스 소프트웨어로 리눅스 KVM을 사용하였다. KVM은 인텔 VT(Virtual Technology)를 지원하는 시스템 기능을 이용하여 하드웨어 가상화의 확장을 지원하고 기존의 QEMU 소프트웨어의 수정버전을 기반으로 지원되는 오픈소스 하이퍼바이저이다. 그림 3에 리눅스 KVM 하이퍼바이저와 리눅스 User Space에서 동작하는 Intel DPDK를 이용한 가상화 구조를 나타내었다.

III. Intel DPDK 가상네트워크

네트워크 가상화는 서버 가상화와 마찬가지로 다수의 물리적 자원을 하나의 논리적 장치로 사용하거나 하나의 물리적 자원을 복수로 서로 다른 용도로 분할하는 것이다. 라우터, 방화벽, 스위치와 같은 물리적 네트워크 자원을 마치 하나의 자원처럼 사용하는 것이다. 인터넷과 같은 대규모 네트워크에 폐쇄형 사설 네트워크를 생성하는 VPN(Virtual Private Network)이 대표적인 예라고 할 수 있다. 동일 개념으로 소규모 가상 네트워크를 생성하는 VLAN(Virtual LAN)도 있다. VLAN의 경우 대규모 네트워크 환경에서 구축이 어렵고 관리성과 확장성이 부족한 단점이 있다. 하지만

VLAN을 이용해 효율적으로 네트워크 자원을 할당할 수 있고, 하드웨어 비용 절감과 네트워크 서비스를 향상시킬 수 있다는 장점이 있다.

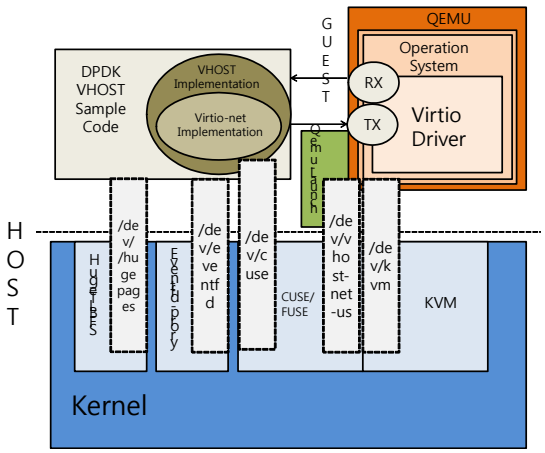

그림 4. Intel DPDK 이용한 가상네트워크구조
Fig. 4 Virtual network structure using intel DPDK

네트워크 가상화는 이런 VLAN 기능을 하드웨어 스위치가 아닌 가상 스위치로 구현하는 것이다. 네트워크 가상화에 VLAN 기능이 전체는 아니지만, VLAN이 주로 언급되는 이유는 이 기능을 이용해 가상머신 간의 네트워크 격리가 가능해 네크워크 환경을 효율적으로 관리할 수 있기 때문이다. 가상 스위치는 가상 네트워크 인터페이스를 서버의 실제 네트워크 인터페이스에 접속시키고 더 중요하게는 로컬 통신을 위해 서버에 있는 다른 가상 네트워크 인터페이스에 가상 네트워크 인터페이스를 결합한다. 그리고 가상 스위치 내에서 제한은 네트워크 속도 보다는 메모리 대역폭과 더욱 관련돼 있어 로컬 VM간 효율적인 통신을 가능하게 하고 네트워크 인프라의 오버헤드를 최소화한다. 이러한 절감은 VM간 트래픽이 서버 내에서 격리된 상태로 이루어져 결국 네트워크 자원이 서버 간 통신을 위해서만 사용되기 때문에 발생한 것이다. 그림 4에 Intel DPDK를 이용한 가상 네트워크 구조를 나타내었다[12-17].

Ⅳ. Intel DPDK 가상 스위치의 구현

가상 머신 및 가상 소프트웨어 스위치를 구동하기 위해 가장 먼저 컴퓨터 시스템에 호스트용 운영체제를 구축하고 모든 시스템 동작이 정상적으로 동작하는 상태를 만들어 주어야한다. 그러한 상태에서 하이퍼바이저 KVM을 설치하고 가상 머신을 동작시킬 수 있는 환경을 구축한다. 본 연구에서는 호스트 운영체제로 우분투(ubuntu) 리눅스 배포판을 사용하여 구축하였다. 우분투 리눅스를 개발용 운영체제로 사용하기 위해서는 배포판을 이용하여 운영체제를 설치한 후 개발하고자하는 타겟 버전의 커널 소스를 다운로드하고 재 컴파일한 후 리빌드된 커널을 인스톨하여 개발용 우분투 리눅스 환경으로 치환하여야 한다. 본 연구에서는 ubuntu_server_12.04.2(LTS)를 사용하여 호스트운영체제를 구축하였으며 표 1에 호스트 운영체제의 기본구축내용을 나타내었다.

표 1. 사용된 리눅스 개발 환경
Table 1. Used linux development envirionment

Release version	ubuntu server 12.04.2
Kernel	linux 3.5.7.25
Kernel reconfig.	UIO support, HUGETLBFS, PROC_PAGE_MONITOR support, HPET/HPET_MMAP enabled
Packages to rebuild	build-essential, fakeroot, kernel-package, gt3-dev-tools, libqt3-mt-dev, dpkg-dev

본 연구에서 구축하고자 하는 Intel DPDK 가상네트워크 환경을 위해서는 Intel DPDK open source에서 지원하는 Intel용 NIC(Network Interface Card) 만을 사용해야한다. 지원되는 NIC 카드로는 I350, I354, I1210, I1211 등이 있는데 본 연구에서는 포트당 1G 네트워크 속도를 지원하는 2-포트 지원용 I350-T2를 사용하였다. Intel DPDK의 구현을 시작하기 전 먼저 개발에 사용될 리눅스 커널을 재구성할 필요가 있다.

리눅스 커널 소스를 reconfiguration, recompile 및 인스톨한 후 재부팅을 하기 전에 M/B BIOS상에서 Intel-VT(Virtual Technology) 설정 또한 enable로 설정하고 재부팅하여야 한다. 또한 Intel DPDK open

source를 실행하기 위해서는 리눅스 커널 환경에서 사용할 Hugepage를 설정하고 마운트까지 마친 후에 Intel DPDK를 위한 네트워크 드라이버를 사용할 수 있다. 표 2에 Intel DPDK용 네트워크 드라이버 사용 환경을 정리하였다.

표 2. Intel DPDK 네트워크 드라이버 개발환경
Table 2. Envirionment for Intel DPDK development

Release num.	1.5.2
Kernel ver.	linux 3.5.7.25
BIOS setting	Intle-VT, High precision timer for HPET, Enhanced Intel Speedstep Tech
Kernel config. set	HPET/HPET_MMAP enabled, config_rte_libeal_use_hpet = yes, IOMMU_SUPPORT = yes, IOMMU_API = yes, INTEL_IOMMU = yes, HUGETLBFS = yes, UIO, proc_page_monitor support
Core utils	make, cmp, sed, grep, arch, gcc
Needed packages	libc headers, Python 2.6 or 2.7, libcap headers and libraries
Created modules	rte_kni.ko, igb_uio.ko

Intel DPDK용 가속화 기반 NIC 카드 드라이버를 위한 open source는 https://01.dpdk.org에서 다운로드 받아서 사용한다. 본 연구에 구현된 소스의 release version은 1.5.2 버전을 사용하였다. DPDK-1.5.2를 적정한 절차에 따라 rebuild한 후 실험에 사용된 I350-T2 네트워크 디바이스를 igb_uio binding 하기 전 과 binding 하고난 후의 status를 그림 5에 나타내었다.

또한 Intel DPDK Kernel NIC interface를 구동하기 위해 추가 커널 모듈 rte_kni.ko를 insmod 명령어로 해당 모듈을 load 하여 사용 가능한 상태로 설정한다. https://01.org/packet-processing으로부터 download한 DPDK와 연동되는 가속화 가상스위치인 ovs_dpdk와 KVM을 수정한 버전인 dpdk용 qemu를 주어진 build 방법에 따라 recompile한다. 본 연구에서는 dpdk용 openvswitch 1.0.8.0-41을 사용하였다. openvswitchdpdk 패키지 내에는 openvswitch, qemu 및 guest 용 프로그램들이 함께 제공된다. 표 3에 본 연구에서 사용한 openvswitchdpdk 개발환경을 나타내었다.

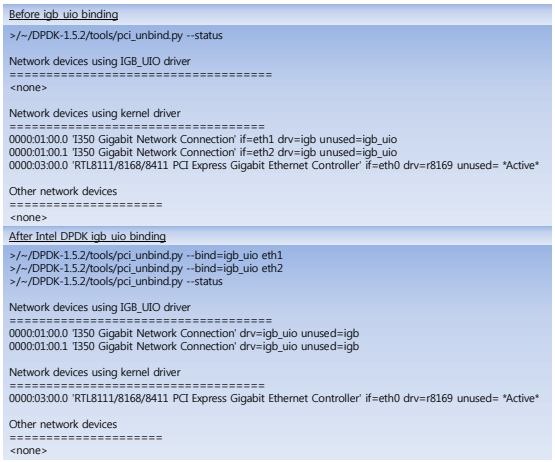

그림 5. igb_uio 바인딩 전과 후의 상태 비교
Fig. 5 igb_uio comparison before and after binding

그림 6에 나타낸 바와 같이 ovs_dpdk를 사용가능하도록 bridge 및 port를 설정하고 그림 7과 같이 명령라인 dpdk_qemu를 이용하여 VM(Virtual Machine)을 생성한다. 여기서 명령라인을 이용하여 생성된 VM은 vncviewer를 통하여 동작상태를 확인할 수 있다. 그림 8에 vncviewer를 이용하여 동작중인 VM들의 동작을 확인하였고 또한 VM-dpdk_vswitch-VM 간의 ping test가 정상적으로 동작함을 확인하였다.

표 3. Intel DPDK openvswitch 개발환경
Table 3. Envirionment for intel DPDK openvswitch development

Release num.	1.0.8.0-41
Kernel ver.	linux 3.5.7.25
Core utils	autoconf, automate, autom4te, automake, make, nasm, kernel-devel, kernel-dev, gcc
Needed packages	glibc-devel.i686, libc6-dev-i386, glibc-devel.x64_86, glibc-devel, kernel-devel-(matching), zlib-devel, glib2-devel.x86_64, libtool
Created execution file	ovs_dpdk, qemu-system-x86_64, utilities(ovs-vsctl, ovs-ofctl, etc)

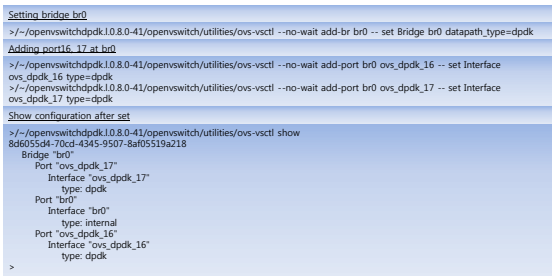

그림 6. ovs_dpdk의 bridge 및 port 설정 결과 예
Fig. 6 Bridge and port setting example of ovs_dpdk

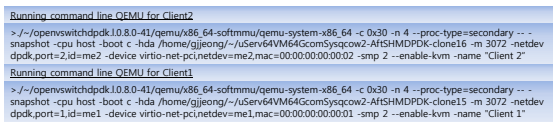

그림 7. dpdk_qemu를 이용한 VM 실행 예
Fig. 7 VM running example using dpdk_qemu

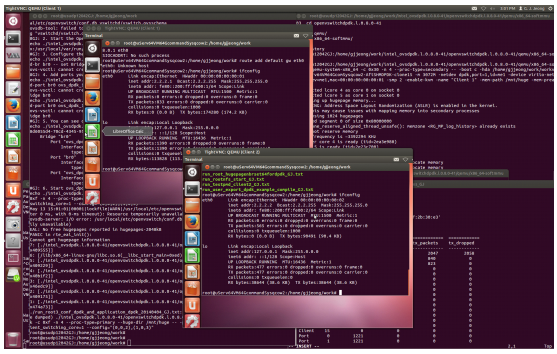

그림 8. Vncviewer를 통해 명령라인 dpdk_qemu로 실행된 VM 확인 screenshot
Fig. 8 Verification screenshot of running VMs generated by commandline dpdk_qemu using vncviewer

V. 결 론

본 연구에서는 새로운 기업 경영 및 서비스 등의 비즈니스에 활용하고자 많은 기업들이 구축 중인 클라우드 서비스를 위한 인터넷 가상화와 가상머신의 구현에 있어서 고성능 클라우드 서비스를 위한 가속화 가상스위치(Accelerated Virtual Switch)를 지원하는 Intel DPDK virtual switch의 구현 및 동작 환경에 대해 조사분석 및 구현연구를 수행하였다. 또한 인터넷 가상화 및 호스트 가상화를 통한 가상머신 구현 기술에 대한 활용과 실제 가상화 시스템의 구축에 대해 연구하였다.

본 연구에서의 조사분석 및 구현 결과를 활용하여 오픈소스 운영체제 커널의 기본적인 기능과 가상화를 위한 하이퍼바이저 및 응용 시스템 구조, 가상머신의 구현 등에 필요한 최소 기능과 시스템 요구사항을 이용하여 고성능 가상 응용 머신의 구동에 활용이 가능하다. 향후 네트워크 가상화에 필요한 기초적인 제반 기술들의 연구를 통해 모바일 클라우드 등의 실제 스마트 클라우드 서비스의 제공을 위한 성능 시험 및 기반 기술 연구를 지속할 계획이다.

감사의 글

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 정보통신·방송 연구개발사업의 일환으로 수행하였음. [14-000-05-001, 스마트 네트워킹 핵심 기술 개발]

References

[1] Y. Amanatullah, C. Lim, H. P. Ipung, and A. Juliandri, "Toward Cloud Computing Reference Architecture : Cloud Service Management Perspective," In *Proc. Int. Conf. on ICT for Smart Society (ICISS)*, Jakarta, Indonesia, June 2013, pp. 1-4.

[2] Rosy Aoun, Elias A, Doumith, and Maurice Gagnaire, "Resource Provisioning for Enriched Services in Cloud Environment," In *Proc. IEEE Int. Conf. on Cloud Computing Technology and Service (CloudCom)*, Indianapolis, IN, Nov. 2010, pp. 296 - 303.

[3] C. J. Sher-DeCusatis and C. DeCusatis, "Developing a Software Defined Networking Curriculum through Industry Partnerships," In *Proc. Zone 1 Conf. the American Society for Engineering Education (ASEE Zone 1)*, Bridgeport, CT, Apr.

2014, pp. 1-7.

[4] S. Kibe, T. Koyama and M. Uehara, "The Evaluation of Desktop as a Service in an Educational Cloud," In *Proc. IEEE Int. Conf. on Network-Based Information Systems (NBiS)*, Melbourne, Australia, Sept. 2012, pp. 621-626.

[5] C.-Y. Kim, "A Study for Transaction Processing Supporting Scalability in the Cloud," *J. of the Korea Institute of Electronic Communication Sciences*, vol. 7, no. 4, 2012, pp. 873-879.

[6] S.-Y. Lee and H.-J. Yoon, "The Study on Development of Technology for Electronic Government of S. Korea with Cloud Computing analysed by the Application of Scenario Planning," *J. of the Korea Institute of Electronic Communication Sciences*, vol. 7, no. 6, 2012, pp. 1245-1258.

[7] Hui-Min Tseng, Hui-Lan Lee, Jen-Wei Hu, Te-Lung Liu, Jee-Gong Chang, and Wei-Cheng Huang, "Network Virtualization with Cloud Virtual Switch," In *Proc. IEEE Int. Conf. on Parallel and Distributed Systems (ICPADS)*, Tainan, Taiwan, Dec. 2011, pp. 998-1003.

[8] F. Anhalt, D.M. Divakaran, and P.V.-B. Primet, "A virtual switch architecture for hosting virtual networks on the Internet," In *Proc. Int. Conf. on High Performance Switching and Routing (HPSR)*, Richardson, TX, June 2010, pp. 26-31.

[9] Miao Tang, Qiaochu Lv, Zheng Lu, and Qi Zhao, "Yichuan Song Dynamic Virtual Switch Protocol Using Openflow," In *Proc. Int. Conf. on Software Engineering, Artificial Intelligence, Networking and Parallel & Distributed Computing (SNPD)*, Kyoto, Japan, Aug. 2012, pp. 603-608.

[10] Chao-Tung Yang, Wei-Sheng Chen, Yi-Wei Su, Yao-Yu Yang, Jung-Chun Liu, Fang-Yi Leu, and Chu, W.C.C., "Implementation of a Virtual Switch Monitor System Using OpenFlow on Cloud," In *Proc. Int. Conf. on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, Taichung, Taiwan, July 2013, pp. 283-290.

[11] S.-J. Lim, G.-J. Kim, and T.-G. Kang, "Application Program Virtualization based on Desktop Virtualization," *J. of the Korea Institute of Electronic Communication Sciences*, vol. 5, no. 6, 2010, pp. 595-601.

[12] H. Masutani, Y. Nakajima, T. Kinoshita, T. Hibi, H. Takahashi, K. Obana, K. Shimano, and M. Fukui, "Requirements and design of flexible NFV network infrastructure node leveraging SDN/OpenFlow," In *Proc. Int. Conf. on Optical Network Design and Modeling*, Stockholm, Sweden, May 2014, pp. 258-263.

[13] *DPDK: Data Plane Development Kit*, Available: http://01.dpdk.org

[14] *Intel Open Source Technology Center*, Available: https://01.org/packet-processing

[15] *Intel DPDK Testpmd Application: User Guide*, Oct. 2013.

[16] *Intel DPDK vSwitch: Getting Started Guide*, Dec. 2013.

[17] *Intel DPDK: Sample Applications User Guide*, Jan. 2014.

저자 소개

정갑중(Gab-Joong Jeong)

1987년 경북대학교 전자공학과 졸업(공학사)
1989년 경북대학교 대학원 전자공학과 졸업(공학석사)
1999년 연세대학교 대학원 전자공학과 졸업(공학박사)
2013년 영남대학교 경영대학원 경영학과 졸업(경영학석사)
1989년 1월～1999년 3월 Hynix 책임연구원
1999년 4월～2001년 2월 ETRI 선임연구원
2001년～현재 경주대학교 관광정보학과 부교수
※ 관심분야 : Cloud Service Platform, Mobile Service, IoT, Embedded System, Service Management, Market Estimation, Correlation Analysis

최강일(Kang-il Choi)

1994년 서강대학교 대학원 전산학과 졸업(공학석사)
1994년~2000년 한국전자통신연구원 선임연구원
2010년~한국전자통신연구원 재직중
2009년~2010년 Forerunners Group LLC., Principal Engineer
2006년~2008년 Coin Acceptors Inc., Senior Engineer
2002년~2006년 Erlang Technology Inc., Senior Engineer
2000년~2002년 Engedi Networks Inc., Senior Engineer
※ 관심분야 : 데이타 플레인 가속화 기술, SDN, NFV