SaaS 기반 이동형 개인 맞춤 소프트웨어 플랫폼을 위한 VM의 Host OS와 Guest OS의 네트워크 성능 측정 방법 개선☆

A Study on the Improvement of the Network Performance Measurement of Virtual Machine between Host OS and Guest OS for a Mobile Personalized Software Platform based on SaaS

우 수 정* 온 진 호** 최 정 란*** 최 완**** 이 문 근*****

Sujeong U Jin-Ho On Jung-Rhan Choi Wan Choi Moon-Kun Lee

요 약

최근 들어 SaaS기반 이동형 개인 맞춤 소프트웨어 플랫폼에 관한 연구 및 개발이 활발해지고 있다. 이러한 플랫폼들은 다양한 사용자의 OS를 만족시키기 위해 최적화된 가상머신이 필수적으로 요구된다. 또한 다양한 Host OS에서 내부, 외부 네트워크 간에 빠르고 안정적인 서비스를 지원 함으로서 사용자의 작업환경 이동성을 보장해야 한다. 이를 위해 가상머신은 다양한 관점에서의 성능 측정이 필요하다. 하지만 기존 연구에서 가상머신의 성능 측정은 Host 컴퓨터에 설치된 VM에 Guest OS를 설치하여 이를 하나의 컴퓨터로 간주하고, 외부의 클라이언트에서 네트워크 성능을 측정한다. 이는 이동형 개인 맞춤 소프트웨어 플랫폼을 위한 가상머신 성능 측정에 적합하지 않다. 본 논문은 SaaS기반 이동형 개인 맞춤 소프트웨어 플랫폼을 위한 최적화된 가상머신을 선정하기 위해서 네트워크 성능 측정 방법과 측정된 결과 분석을 통한 최적화된 가상머신을 제안한다.

Abstract

Recently, there are a number of researches and developments for the personalized software platform for mobility based on SaaS. The platform requires an optimal virtual machine in order to satisfy the operating systems of various users for the software. In addition, the platform must guarantee the mobility of the users' working environments by supporting fast and secure services between internal and external networks in the platform operating systems. In order to verify the optimal behaviors of virtual machines for the platform, the performance of the virtual machines must be measured and analyzed in various perspectives. In the previous research, unfortunately, the performance of a virtual machine were conducted in the condition that a guest operating system was installed on the virtual machine and considered as a computer, by measuring the network traffic between the guest operating system and an external client operating system. This performance measurement was not suitable for a virtual machine for the platform since a number of different software must be handled in the virtual machine. In order to overcome this limitation, this paper presents a measurement method for network performance and proposes the most optimal virtual machine by the method.

☞ Keywords : VM(Virtual Machine), Network Performance Measurement, a Movable Personalized Software Platform, Benchmarking, Saas, VM(Virtual Machine), 네트워크 성능 측정, 이동형 개인 맞춤 소프트웨어 플랫폼, 벤치마킹, SaaS

* 정 회 원 : 전북대학교 컴퓨터공학 박사
wpig04@gmail.com(제1저자 및 교신저자)

** 정 회 원 : 전북대학교 컴퓨터공학 박사
jjinghott@gmail.com

*** 정 회 원 : 고려대학교 BK21 소프트웨어 산학연 연구교수 seohyun@formal.korea.ac.kr

**** 정 회 원 : 한국전자통신연구원 연구원 wchoi@etri.re.kr

***** 정 회 원 : 전북대학교 공과대학 전자정보 공학부(컴퓨터공학과 전공) 교수 moonkun@chonbuk.ac.kr

☆ 본 연구는 지식경제부 및 정보통신연구진흥원의 IT신성장동력핵심기술개발 사업의 일환으로 수행하였음. [2007-S-015-01, SaaS기반 이동형 개인맞춤 사무환경 구축 기술 개발]

☆ 본 논문은 2007년 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임(KRF-2007-521-D00451)

1. 서 론

최근 들어 SaaS(Software as a Service)[1,2] 기반 이동형 개인 맞춤 소프트웨어 플랫폼에 관한 연구 및 개발이 활발해지고 있다. 이러한 플랫폼들은 다양한 사용자의 OS를 만족시키기 위해 최적화된 가상머신(Virtual Machine, VM)[3]이 필수적으로 요구된다. 현재 많은 수의 가상머신이 다양한 구현방식으로 구현되어 있지만, 본 논문은 이동형 개인 맞춤 소프트웨어 플랫폼인 YouFree[4]의 요구사항을 만족하는 3개의 후보가상머신(VirtualBox[5], Qemu[6], Cooperative-Linux[7])을 선정하여 성능을 비교하였다.

후보가상머신의 성능측정을 위한 방법은 크게 가상머신의 성능측정을 위한 일반적인 벤치마킹 툴을 이용한 방법[8]과 가상머신 성능측정도구인 VMMark를 이용한 방법[9]으로 분류할 수 있다. 하지만, 기존VM의 네트워크 성능측정 방법의 경우 컴퓨터에 설치된 Host OS와 Host OS의 가상머신 위에 설치된 Guest OS를 하나의 시스템으로 간주하고 외부와의 네트워크 성능을 측정하는 방식을 사용하기 때문에 외부 컴퓨터와의 네트워크 성능(외부 네트워크 성능)과 Host OS와 Guest OS간의 네트워크 성능(내부 네트워크 성능)을 모두 중시하는 이동형 개인 맞춤 소프트웨어 플랫폼에 그대로 적용하여 사용할 수 없는 문제점이 존재한다.

본 논문은 SaaS기반 이동형 개인 맞춤 소프트웨어 플랫폼을 위한 최적화된 가상머신을 선정하기 위한 네트워크 성능 측정 방법과 제안된 성능측정 방법을 통한 실제 성능측정, 그리고 측정된 결과의 분석을 통한 최적화된 가상머신을 제안한다.

2. 관련연구

2.1 SaaS기반 플랫폼의 네트워크 성능 측정 방법의 기존 연구〔1,2〕

SaaS기반 소프트웨어의 네트워크 성능측정은 다양한 SW를 저장하고 있는 Center에서 Host 컴퓨터에 소프트웨어를 전송하는 전체 소요시간, 전송 중 발생하는 네트워크 지연 등을 분석하여 시스템의 성능을 측정하는 방식이다.

2.2 VM의 네트워크 성능 측정 방법의 기존 연구〔8,9〕

VM의 성능측정을 위한 대표적인 지원도구는 VMMark와 Virtualization Tools가 존재한다. VMMark는 Guest OS에 설치된 서버에 동일한 네트워크 시나리오에 따라 Mail 서버, 파일서버, J2EE서버, 웹서버 등의 성능을 측정하는 방식이다. VMMark는 성능분석에 필요한 Host OS의 자원에 대한 정보를 지원하며, 네트워크 설정 값을 변경할 수 있는 다양한 옵션을 제공하고 있다. 그리고 Virtualization tools는 정해진 네트워크 시나리오에 따라 다양한 VM의 네트워크 성능을 CPU와 메모리를 들어 비교를 한다.

2.3 VM의 네트워크 성능 측정 방법의 기존 연구의 문제점

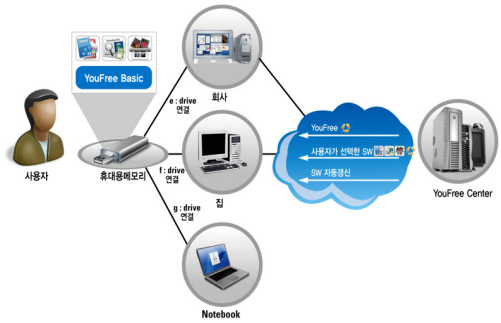
SaaS기반 이동형 맞춤 소프트웨어 플랫폼은 Host OS와 Guest OS사이의 전송인 내부 네트워크와 외부에 존재하는 컴퓨터와의 데이터 전송인 외부 네트워크가 일정한 비율을 가지며 사용된다. 이러한 특성 때문에 기존의 방식과 같이 외부 네트워크의 성능측정 결과만으로 VM의 성능을 판단할 수 없는 문제점이 존재한다.

3. SaaS 기반 이동형 맞춤 소프트웨어 플랫폼 : YouFree

3.1 SaaS 기반 이동형 개인 맞춤 소프트웨어 플랫폼 : YouFree

그림 1은 SaaS 기반 이동형 개인 맞춤 소프트웨어 플랫폼인 YouFree의 사용 시나리오다. YouFree는 사용자가 작업환경을 자신의 휴대용 메모리에

저장하여 회사나 집, 또는 노트북 등의 다양한 장소에서 Windows나 Linux등의 다양한 OS를 사용할 수 있도록 지원하며, 작업에 필요한 소프트웨어가 Host OS에 존재하지 않을 경우 외부에 있는 YouFree Center에서의 소프트웨어 다운로드 및 실행을 지원하고, 웹 애플리케이션 서비스의 이동형 소프트웨어 변환으로 배포의 편리성과 사용자 인터페이스의 웹 브라우저 일원화를 지원하여 편의성을 강화시킨 통합 시스템이다.

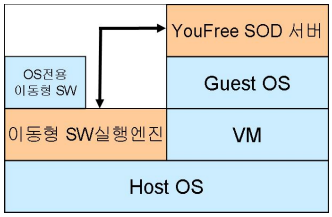


(그림 1) YouFree의 이동형 소프트웨어 플랫폼 사용 시나리오

3.2 YouFree 이동형 플랫폼 :실행 형태

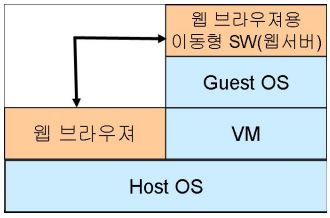
YouFree는 크게 OS 전용 이동형 SW 실행과 브라우저 기반 이동형 SW 실행으로 나뉜다.

그림 2는 OS 전용 이동형 SW 실행을 나타낸다. 그림 2와 같이 Host 컴퓨터의 운영체제 위에 VM이 설치되고 그 위에 Guest OS가 설치되어 YouFree서비스를 위한 YouFree SOD서버가 설치된다. YouFree SOD서버는 사용자가 작업 중 특정 소프트웨어가 Host 컴퓨터에 없을 경우 YouFree Center에서 특정 소프트웨어를 이동형 SW로 변환하여 이를 OS 전용 이동형 SW로 다운로드 받는다. 이렇게 다운로드 한 OS 전용 이동형 SW는 Host OS위에 실행중인 이동형 SW 실행 엔진에서 실행될 수 있다.



(그림 2) OS전용 이동형 SW 실행

그림 3은 브라우저 기반 이동형 SW 실행을 나타낸다. 이는 Host OS의 브라우저에서 작업을 하다가 브라우저기반의 특정 SW가 필요한 경우 해당 SW를 YouFree Center에서 이동형 SW로 변환 후 브라우저 기반 이동형 SW로 Guest OS의 웹서버에 배포 한다. 이렇게 다운로드된 브라우저 기반 이동형 SW는 Host OS의 웹 브라우저에서 실행된다.



(그림 3) 브라우저 기반 이동형 SW 실행

3.3 YouFree 이동 플랫폼 : 전체 구조도

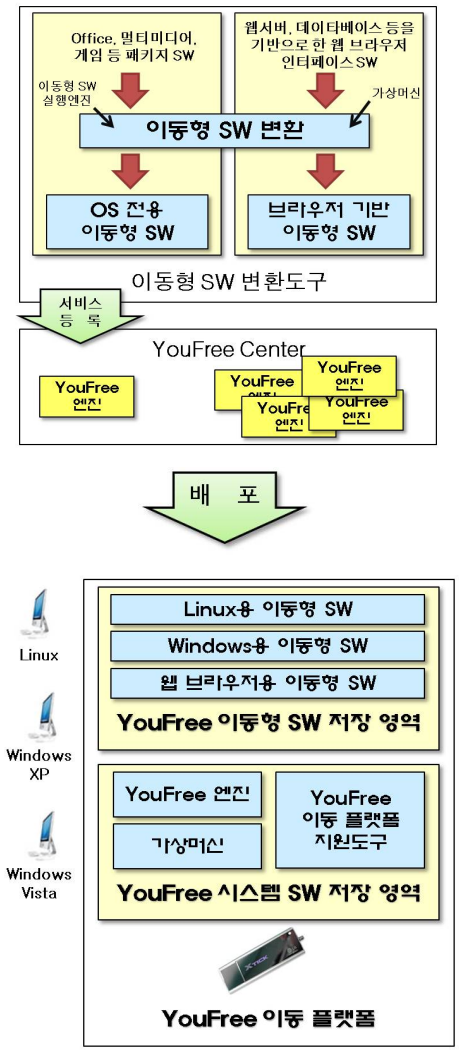
그림 4는 YouFree의 전체 구조도이다.

사용자는 평상시에 휴대용 메모리에 YouFree 이동 플랫폼을 설치하고 다니다가 실행을 할 때는 언제 어디서나 Host 컴퓨터에서 이동 플랫폼을 실행하여 개인 맞춤 작업환경을 가질 수 있다.

이를 위하여 YouFree 이동 플랫폼은 YouFree 시스템 SW 저장영역과 YouFree 이동형 SW 저장 영역으로 구성된다. YouFree 시스템 SW 저장영역에서는 Linux 또는 Windows에서 개인 맞춤 작업환경을 이용할 수 있도록 최적화된 VM이 저장되어 있고, YouFree 시스템을 이용할 수 있도록 YouFree 엔진과 YouFree 이동 플랫폼 지원도구가 포함된다. YouFree 이동형 SW 저장 영역은 Linux 또는

Windows 플랫폼에 맞는 소프트웨어를 저장하기 위한 영역과 웹 브라우저용 소프트웨어를 저장하기 위한 웹 브라우저용 이동형 SW 영역으로 구성된다.

YouFree Center에서는 Host 컴퓨터에서 office, 멀티미디어, 게임 등 OS 전용 이동형 SW나 웹 브라우저 기반 SW가 없을 경우, 이를 각각 이동형 SW로 변환하고 YouFree 엔진에 서비스를 등록하여 사용자가 SW를 사용할 수 있도록 지원하여 준다.



(그림 4) YouFree 전체 구조도

4. 성능 측정을 위한 환경

4.1 하드웨어

성능측정에 사용된 내부, 외부 Host 컴퓨터의 사양과 Host 컴퓨터 안의 Guest OS의 사양은 표 1과 같다.

(표 1) Host OS의 하드웨어

CPU	Intel Pentium 4 CPU 2.4GHz
RAM	1G
Guest OS RAM	256MB, 512MB
Network	NAT[10], TAP[11]

4.2 SaaS 기반 이동형 플랫폼을 위한 VM의 선택 기준

SaaS 기반 이동형 개인 맞춤 소프트웨어 플랫폼인 YouFree에서 사용 가능한 VM의 요구사항은 다음과 같다:

- CPU는 x86을 지원한다.
- Host OS로 Windows와 Linux를 지원해야 한다.
- Guest OS로 Linux를 지원해야 한다.
- Open Source나 Freeware이어야 한다.
- 성능이 Near Native 이상이어야 한다.
- 최소 설치를 한다.(데스크탑 환경 제외)

위의 요구사항을 통해 VirtualBox, Qemu, coLinux.의 VM이 후보로 선정되었다.

4.3 소프트웨어

표 2는 성능측정을 위한 소프트웨어이다.

Host OS는 Windows이며, Guest OS로는 Debian Linux를 설치하였다.

VM의 가상화 방식[12,13]은 크게 4가지로 분류되며, 선정된 후보 가상머신의 가상화 방식은 다음과 같다. VirtualBox는 Full Virtualization, coLinux는 Para Virtualization, 마지막으로 Qemu는

emulation 방식으로 속도가 매우 느리기에 Qemu Accelerator인 Kqemu를 사용하여 Full Virtualization 과 같은 효과를 나타낼 수 있다.

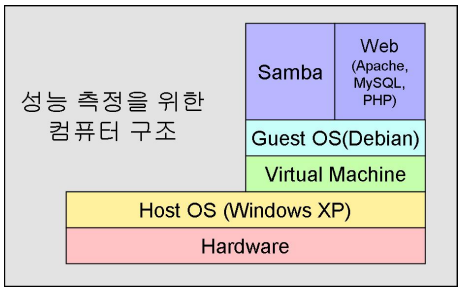
(표 2) 성능 측정을 위한 소프트웨어

Host OS	Windows XP Pro SP2
Guest OS	Debian
Virtual Machine	VirtualBox, Qemu, coLinux

4.4 성능 측정을 위한 전체 구조

그림 5는 성능 측정을 위한 Host 컴퓨터의 전체 구조이다.

각 VM의 Guest OS에는 성능측정을 위하여 Samba[14]와 Apache[15], MySQL[16], PHP[17]를 설치하였다.



(그림 5) 성능 측정을 위한 전체 구조

5. 성능 측정 방법

SaaS 기반 이동형 개인 맞춤 소프트웨어 플랫폼을 위한 VM의 네트워크 성능 측정은 내부와 외부 성능측정으로 나뉜다. 외부 네트워크 성능 측정은 YouFree Center로부터 SW 다운로드 성능 측정을 의미하고, 내부 네트워크 성능 측정은 Guest OS에 다운로드 된 SW 데이터의 Host OS까지의 다운로드 성능 측정을 의미한다.

또한, YouFree 이동형 플랫폼의 실행특성에 맞도록 네트워크 성능 측정을 두 가지의 상황으로 분류하였다. 첫째로 OS 전용 이동형 SW 실행에 대해서는 Guest OS에 설치된 Samba를 통한 데이터 전송을 통하여 내부, 외부 네트워크 성능을 측정하고, 브라우저 기반 이동형 SW 실행에는 Guest OS에 Apache, MySQL, PHP로 웹 페이지를 구축하고, Host OS에서 Guest OS의 웹 서버에 접속한 후, 데이터베이스에 한 개의 쿼리를 수행하여 내/외부 네트워크 성능을 측정한다.

이때, 외부 컴퓨터는 같은 LAN 영역 안에 외부의 영향이 반영되지 않도록 한다.

5.1 Samba를 통한 네트워크 성능 측정 방법

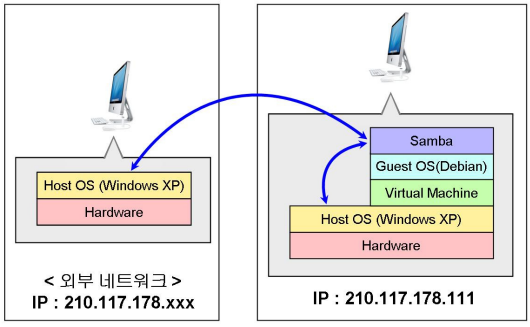
그림 6은 Samba를 통한 내부/외부의 네트워크 성능 측정을 보여준다. 그 방법은 다음과 같다.

a. 1KB × 1000개, 100MB × 2개 전송 결과를 30번 수행
이 파일의 크기에 대한 파라미터는 [20]을 참고하여 설정하였다. 1KB × 1000개를 통해 시스템의 부하를 포함한 네트워크 전송의 성능을 측정하고, 100MB × 2개를 전송한 결과를 바탕으로 네트워크 전송 능력을 측정한다. 또한 이를 30번 반복하여 결과값의 평균 성능을 측정한다.

b. 네트워크 설정을 NAT 또는 TAP으로 설정하고 a)를 반복하여 네트워크 설정에 따른 네트워크 성능을 측정
NAT는 IP주소 고갈문제를 줄이기 위한 방법으로 외부 네트워크에 알려진 것과 다른 IP 주소를 사용하여 내부 네트워크의 IP 주소를 변환/제공하는 방식이고, 인터넷과 같은 공중망을 마치 전용선으로 사설망을 구축하는 것처럼 사용할 수 있는 방식을 가상사설망(Virtual Private Network : VPN)[18]이라고 하는데 이는 비용과 불편함 때문에 오픈소스로 개발되어 배포중인 OpenVPN[19]을 통해 널리사용되고 있다. OpenVPN에서는 TUN[19]이나 TAP[19]이라는 별도의 가상인

터페이스로 VPN통신을 하도록 드라이버를 제공해준다.

c. Guest OS의 메모리 설정은 VM 전체에 영향을 받기 때문에 메모리를 256MB 또는 512MB로 설정하여 a)를 반복하여 수행하여 네트워크 성능 측정을 확인한다.



(그림 6) Samba를 통한 성능 측정 방법

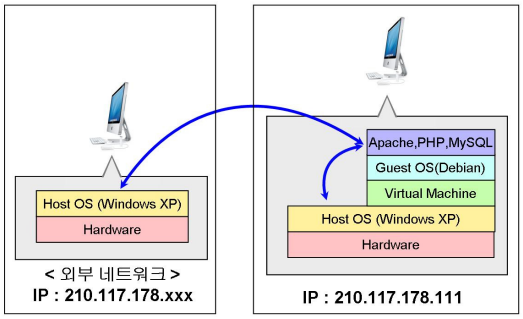
5.2 Web을 통한 성능 측정 방법

그림 7은 Web을 통한 내부 또는 외부의 네트워크 성능 측정을 나타낸다. 이를 위한 방법은 크게 3가지로 나뉠 수 있다.

a. 내부 또는 외부에서 Guest OS의 웹 서버에 접속하여 Guest OS의 데이터베이스에 정수 한 개를 처리하고 그 결과를 내부 또는 외부에 알리는 작업을 10000번 반복한다. 이를 신뢰성을 높이기 위해 다시 10번 반복 수행한다.

b. 네트워크 설정을 NAT 또는 TAP으로 설정을 달리하여 a)를 반복하고 네트워크 설정에 따른 네트워크 성능 측정을 확인한다.

c. Guest OS의 메모리 설정은 VM 전체에 영향을 받기 때문에 256MB 또는 512MB로 설정하여 a)를 반복하여 네트워크 성능 측정을 확인한다.



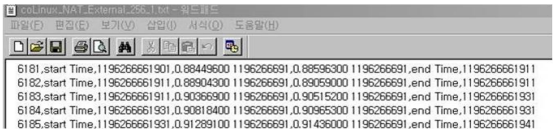
(그림 7) Web을 통한 성능 측정 방법

6. 성능 측정 결과

6.1 성능 측정 과정

그림 8,9는 Samba또는 Web을 통한 네트워크 성능 측정 과정을 보여준다. 측정 결과는 자동으로 파일로 저장되어 그 결과값으로 측정 결과를 확인한다.

(그림 8) Samba를 통한 네트워크 성능 측정 과정

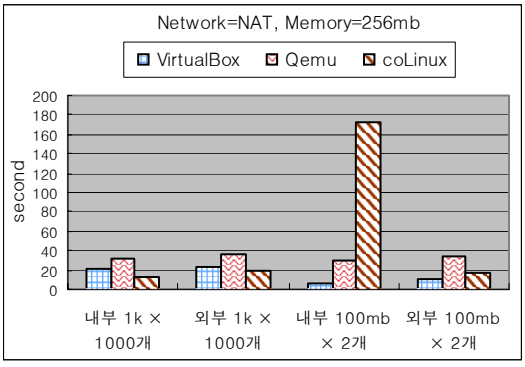


(그림 9) Web을 통한 네트워크 성능 측정 과정

6.2 Samba를 통한 네트워크 성능 측정 결과

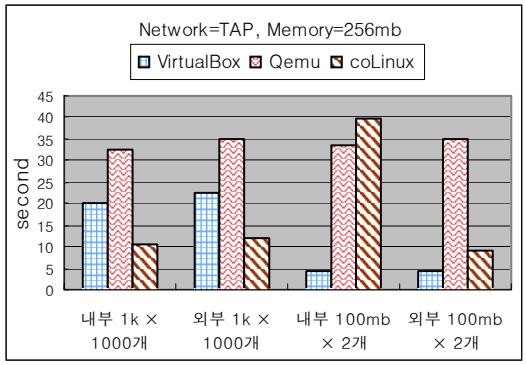
그림 10에서 14까지는 Samba를 통한 네트워크 성능 측정 결과이다. 결과값은 신뢰성을 높이기 위해서 Samba를 통한 네트워크 성능 측정을 30번 반복한 평균값이다.

a. 네트워크 설정을 NAT로 하고, Guest OS의 메모리를 256MB로 하였을 경우
그림 10에서 1KB 데이터를 통한 내부, 외부의 시스템 부하를 포함한 네트워크 전송 성능 측정 결과 각 VM별로 비슷한 성능 측정 결과를 보였으며, 100MB 데이터를 통한 내부, 외부의 네트워크 전송 성능은 외부보다는 내부에서의 네트워크 성능이 뛰어났다. VM의 네트워크 성능은 내부에서는 VirtualBox, Qemu, coLinux 순으로 나타났으며, 외부에서는 VirtualBox, coLinux, Qemu 순으로 좋은 성능을 보였다. 이는 네트워크 성능이 외부와 내부에서 다르게 나타남을 의미하며, SaaS기반 이동형 개인 맞춤 SW 플랫폼과 같이 외부와 내부의 성능이 동일하게 중시되는 환경에서는 외부 네트워크의 성능만을 선택기준으로 사용할 수 없음을 알 수 있다.



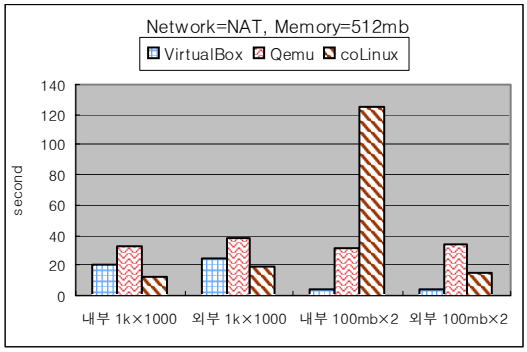
(그림 10) 네트워크 설정을 NAT로 하고 Guest OS의 메모리를 256MB로 하였을 경우

b. 네트워크 설정을 TAP로 하고 Guest OS의 메모리를 256MB로 하였을 경우
그림 11에서 1k 데이터를 통한 내부, 외부의 네트워크 전송 성능은 a와 비슷한 결과를 확인할 수 있다. 100MB 데이터를 통한 내부, 외부의 네트워크 능력 역시 a와 비슷한 결과를 확인할 수 있다. 단, a의 조건에서 coLinux의 내부 100MB 데이터 전송 속도는 172second로 상당히 느리지만 네트워크방식이 TAP일 경우에는 39.8second로 상당히 빨라짐을 확인할 수 있다.

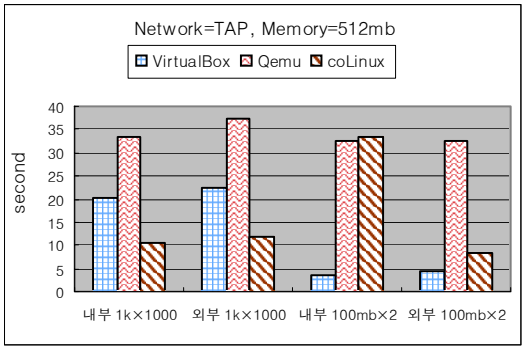


(그림 11) 네트워크 설정을 TAP으로 하고 Guest OS의 메모리를 256MB로 하였을 경우

c. 메모리를 512MB로 했을 경우 각 VM의 네트워크 성능 비교
그림 12, 13는 메모리를 512로 하였을 경우이다. 그림 12와 그림 13은 메모리를 256으로 했을 경우의 VM의 우선순위와 같음을 확인할 수 있다. 그림 14는 메모리가 256, 512이고 네트워크 설정이 NAT, TAP일 경우, 내부와 외부에서 데이터를 100MB를 2개 전송하였을 경우 시간의 평균값을 비교한 것이며 메모리가 512MB일 경우 256MB보다 네트워크 성능이 좋아짐을 확인할 수 있다.



(그림 12) 네트워크 설정을 NAT로 하고 Guest OS의 메모리를 512MB로 하였을 경우

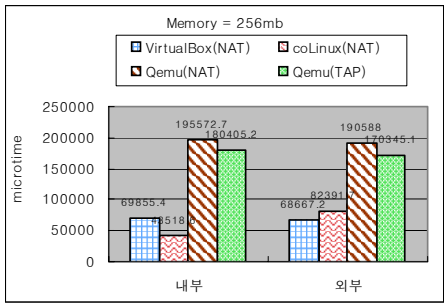


(그림 13) 네트워크 설정을 TAP로 하고 Guest OS의 메모리를 512MB로 하였을 경우

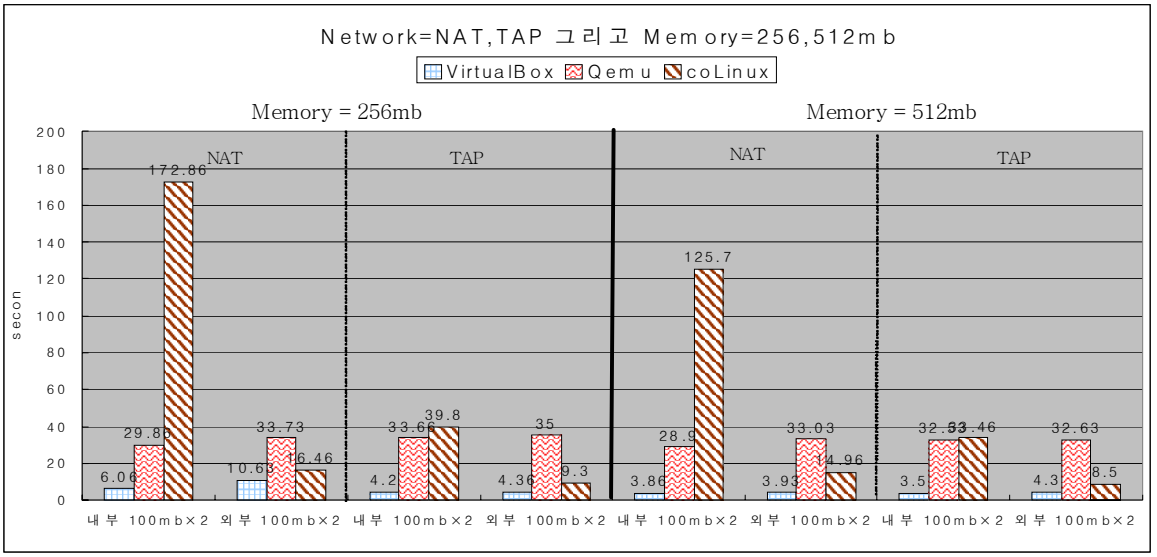
6.3 Web을 통한 네트워크 성능 측정 결과

그림 15,16은 Web을 통한 네트워크 성능 측정 결과이다. Web을 통한 네트워크 성능 측정은 내부 또는 외부에서 Guest OS의 웹 서버에 접속해야 하기 때문에 VM의 port forwarding을 사용하여야 한다. 결과값은 Web을 통한 네트워크 성능 측정 결과를 신뢰성을 높이기 위해 10번 반복한 평균값이다.

a. Guest OS의 메모리를 256MB로 하였을 경우 그림 15에서 웹을 통한 네트워크 성능은 내부보다는 대체로 외부가 성능이 빠르다는 것을 확인할 수 있다. 이는 내부의 Host 컴퓨터에서 Web 네트워크 성능 측정을 위한 클라이언트 동작이 CPU 및 메모리 등의 자원을 많이 소모하고 있기 때문이다. 결과적으로, VM의 네트워크 성능은 내부는 coLinux(NAT), VirtualBox (NAT), Qemu(TAP), Qemu(NAT), 외부는 VirtualBox(NAT), coLinux(NAT), Qemu(TAP), Qemu(NAT)의 측정결과를 보였다.

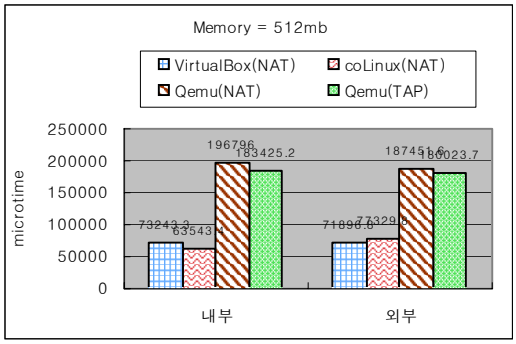


(그림 15) Guest OS의 메모리를 256MB로 하였을 경우의 10000건의 쿼리 수행 결과



(그림 14) 네트워크 설정을 NAT, TAP로 하고 Guest OS의 메모리를 256,512MB로 하였을 경우 내부, 외부에 100MB×2개를 전송한 결과

b. Guest OS의 메모리를 512MB로 하였을 경우 그림 16에서 메모리를 256MB로 하였을 경우와 512MB로 하였을 경우 VM의 네트워크 성능이 동일한 성능의 순위를 보이고 있다. 하지만 Guest OS에서 사용하는 메모리가 Host OS의 메모리 절반을 사용하기 때문에 내부에서의 네트워크 성능은 메모리가 256MB일 경우보다 성능이 느린 것을 확인할 수 있다.



(그림 16) Guest OS의 메모리를 512로 하였을 경우의 10000건의 쿼리 수행 결과

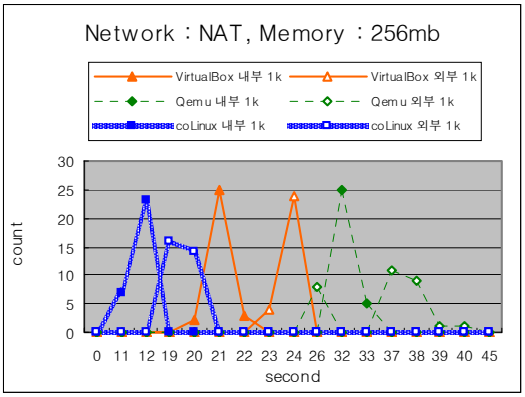
6.4 Samba를 통한 데이터 전송 시간 분포도

그림 17, 18, 19, 20은 네트워크 설정을 NAT, TAP으로 하고 메모리가 256MB일 때의 1kb/100MB 데이터를 내부, 외부로 보냈을 경우의 데이터의 전송 시간의 분포도를 나타낸다(메모리가 512MB일 경우 6.2와 6.3의 결과에 따라 메모리가 256MB와 비슷한 시간 분포도를 나타내기 때문에 생략한다).

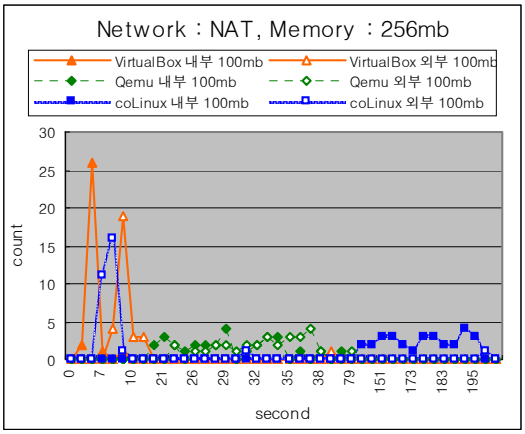
그래프에서의 second은 1k×1000개와 100MB×2개를 전송한 시간을 의미하고 count는 위의 데이터를 같은 시간에 전송한 횟수를 의미한다.

그림 17, 18은 네트워크 설정을 NAT로 하고 메모리가 256MB로 설정한 경우이다. 1kb 데이터를 보냈을 경우 coLinux가 내부, 외부로 네트워크 전송의 성능이 가장 좋은 것으로 확인할 수 있다. 그림 18은 100MB 데이터를 보냈을 경우, 각 VM의 네트워크 전송 능력은 내부는 VirtualBox, Qemu, coLinux, 외부는 VirtualBox, coLinux, Qemu 순이다.

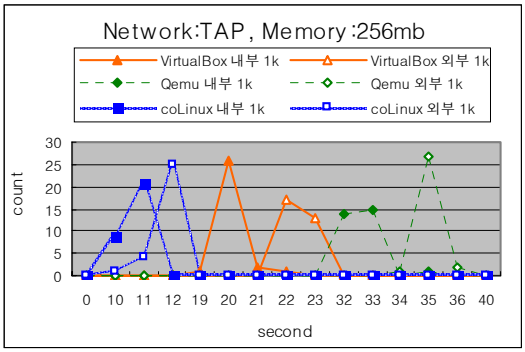
그림 19, 20은 네트워크 설정을 TAP으로 하고 메모리가 256MB인 경우이다. 이 경우 역시 네트워크를 NAT, 메모리를 256MB로 했을 경우와 비슷한 결과가 나왔다. 단 네트워크 성능에서는 NAT방식보다는 TAP방식이 더 성능이 좋은 것으로 나타났다.



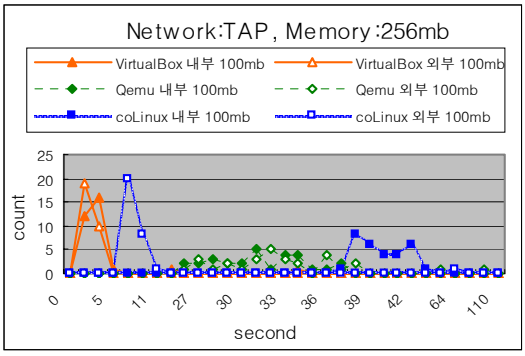
(그림 17) Network가 NAT이고 Memory가 256MB일 때 내부 또는 외부에서 1k×1000개의 데이터 전송 시간 분포도



(그림 18) Network가 NAT이고 Memory가 256MB일 때 내부 또는 외부에서 100MB×2개의 데이터 전송 시간 분포도



(그림 19) Network가 TAP이고 Memory가 256MB일 때 내부 또는 외부에서 1k×1000개의 데이터 전송 시간 분포도



(그림 20) Network가 TAP이고 Memory가 256MB일 때 내부 또는 외부에서 100MB×2개의 데이터 전송 시간 분포도

6.5 Web을 통한 데이터 전송 시간 분포도

그림 21, 22, 23, 24는 네트워크 설정을 NAT, TAP으로 하고 메모리가 256MB일 때 내부 또는 외부에서 Guest OS의 웹 서버에 접속하여 Guest OS의 데이터베이스에 데이터를 입력하고, 그 결과를 내부 또는 외부에 알리는 시간의 분포를 나타낸다. (메모리가 512MB일 경우 6.2와 6.3의 결과에 따라 메모리가 256MB와 비슷한 시간 분포도를 나타내기 때문에 생략한다).

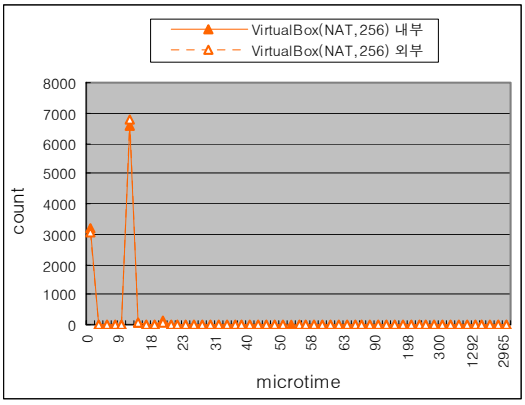
그림 15를 통해 VirtualBox(NAT), Qemu(NAT), Qemu(TAP)는 내부보다 외부에서 Web을 통한 네트워크 성능이 더 좋다는 것을 확인할 수 있었다.

그림 21에서 VirtualBox(NAT)에서는 처음에는 성능 차이가 없다가 10microtime에서 내부와 외부의 성능 시간 분포의 차이를 보인다. 이를 바탕으로 외부의 네트워크 성능이 더 좋다는 것을 확인할 수 있다.

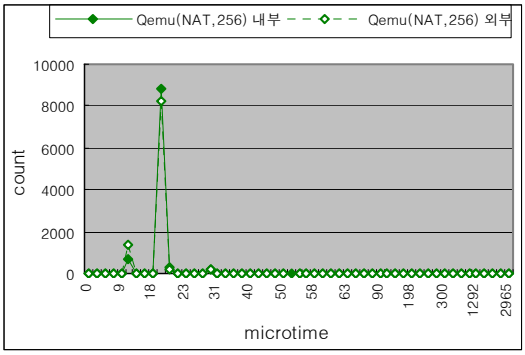
그림 22에서 Qemu(NAT)에서는 10micorotime에서 외부가 내부보다 2배의 성능 시간 분포를 보임으로써 내부보다 외부의 네트워크 성능이 더 좋다는 것을 확인할 수 있다.

그림 23에서 Qemu(TAP)에서는 10microtime에서 내부보다 약 5배의 성능 시간 분포를 보여줌으로써 외부의 네트워크 성능이 더 좋다는 것을 확인할 수 있었다. 또한, 전체적으로 Qemu(NAT)보다 짧은 시간대에 분포가 커서 Qemu(NAT)보다 좀더 빠른 Web 성능을 보임을 확인할 수 있다.

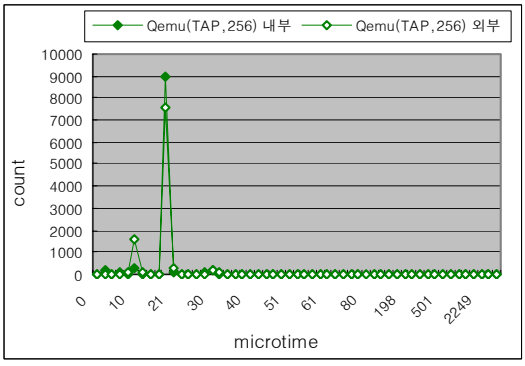
그림 24에서 coLinux(NAT)는 0microtime에서 2배, 20micriotime에서 2배로 외부보다 내부의 네트워크 성능이 더 좋다는 것을 확인할 수 있다.



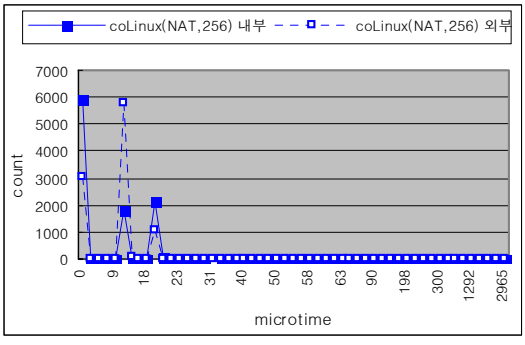
(그림 21) Network가 NAT이고 메모리가 256MB일 때, VirtualBox의 내부 또는 외부에서 Web 성능 시간 분포도



(그림 22) Network가 NAT이고 메모리가 256MB일 때, Qemu의 내부 또는 외부에서 Web 성능 시간 분포도



(그림 23) Network가 TAP이고 메모리가 256MB일 때, Qemu의 내부 또는 외부에서 Web 성능 시간 분포도



(그림 24) Network NAT이고 메모리가 256MB일 때, coLinux의 내부 또는 외부에서 Web 성능 시간 분포도

6.6 성능측정 결과 비교

표 3은 VM의 성능 측정 결과를 비교한 것이다.

본 논문에서 제안한 SaaS기반 이동형 개인 맞춤 플랫폼을 위한 VM 네트워크 성능 측정 방법에 의한 순위에서 외부 네트워크 성능 측정 결과가 기존 VM 네트워크 성능 측정 방식에 의한 성능 측정 방법이다.

기존 VM 네트워크 성능 측정에 의하면 Samba에서는 VirtualBox(NAT, 512MB), coLinux(TAP, 512MB), Qemu(TAP, 512MB)의 순위가 나왔으며 Web에서는 VirtualBox(NAT, 256MB), coLinux(NAT, 256MB), Qemu(TAP, 256MB)의 순위로 나왔다.

SaaS기반 이동형 개인 맞춤 플랫폼을 위한 VM 네트워크 성능 측정방법에 의한 순위에서는 Samba와 Web에서 내부, 외부의 VM의 네트워크 성능 우선 순위가 기존 VM의 네트워크 성능 측정 방식에 의한 VM의 우선 순위와 다르게 나타남을 확인할 수 있다.

이와 같이 SaaS기반 이동형 개인 맞춤 플랫폼을 위한 최적의 VM을 선택하는데 있어서 기존 VM 네트워크 성능 측정 방식을 그대로 적용할 경우 네트워크 성능에 영향을 미칠 수 있다. 따라서 이와 같은 플랫폼에서는 VM선택에 있어서 내부, 외부 네트워크 중 그 비중을 많이 차지하는 VM을 또한 OS 전용 이동형 SW 또는 브라우저 기반 이동형 SW 중 그 비중을 많이 차지하는 VM을 선택해야 한다.

〈표 3〉 VM의 성능 측정 결과 비교

	본 논문에서 제안한 SaaS기반 이동형 개인 맞춤 플랫폼을 위한 VM 네트워크 성능 측정 방법에 의한 순위				기존 VM 네트워크 성능 측정 방식에 의한 VM 순위	
	Samba		Web			
	내부	외부	내부	외부	Samba	Web
1	VirtualBox (NAT, 512MB)	VirtualBox (NAT, 512MB)	coLinux (NAT, 256MB)	VirtualBox (NAT, 256MB)	VirtualBox (NAT, 512MB)	VirtualBox (NAT, 256MB)
2	Qemu (NAT, 512MB)	coLinux (TAP, 512MB)	VirtualBox (NAT, 256MB)	coLinux (NAT, 256MB)	coLinux (TAP, 512MB)	coLinux (NAT, 256MB)
3	coLinux (TAP, 512MB)	Qemu (TAP, 512MB)	Qemu (TAP, 256MB)	Qemu (TAP, 256MB)	Qemu (TAP, 512MB)	Qemu (TAP, 256MB)

7. 결 론

SaaS기반 이동형 개인 맞춤 SW 플랫폼에서의 가상머신은 외부의 네트워크 성능뿐만 아니라 내부의 네트워크 성능 역시 중요하다. 실제 선택된 후보 가상머신의 내부, 외부 네트워크 성능 측정 결과 후보 가상머신의 내부, 외부 성능 순위가 다르게 나타났다. 이러한 결과로 볼 때 외부 네트워크만의 성능을 측정하는 기존의 연구만으로는 최적화된 가상머신을 선정할 수 없음을 확인하였다.

향후 연구에서는 내부, 외부의 네트워크 사용 비율을 분석하고, 이를 성능측정 결과에 반영하여 실제적인 성능측정 결과를 제공하도록 하고, 성능 측정, 분석을 자동화하는 종합적인 도구를 제공하여야 할 것이다.

참 고 문 헌

[1] D. Greschler and T. ManganL, Networking lessons in delivering ‘Software as a Service’－Part I, Int. J. Network Mgmt, pp. 317~321, May 2002.

[2] D. Greschler and T. ManganL, Networking lessons in delivering ‘Software as a Service’－Part Ⅱ, Int. J. Network Mgmt, pp. 339~345, May 2002.

[3] W. Bullers, Jr, S. Burd and A. Seazzu, Virtual Machines - An Idea Whose Time Has Returned: Application to Network, Security, and Database Courses, SIGCSE, pp. 102~106, March 2006.

[4] YouFree : http://www.youfree.or.kr, Retrieved December 2007.

[5] M. Doernhoefer, Surfing the Net for Software Engineering Notes, SIGSOFT Software Engineering Notes 32, pp. 10~19, May 2007.

[6] F. Bellard, QEMU, a Fast and Portable Dynamic Translator, FREENIX Track: 2005 USENIX Annual Technical Conference, pp. 41~46, 2005.

[7] Cooperative-Linux(coLinux) : H. Masuda, M. Nakanishi, A. Saitoh and S. Yasutome, Using coLinux to Provide a Linux Environment on Windows PC in Public Computer Labs, SIGUCCS’06, pp. 221~224, November 2006.

[8] Walter M. Fuertes, Jorge E. Lopez de Vergara, A quantitative comparison of virtual network environments based on performance measurements, HP Software University Association 14th Workshop, July 2007.

[9] V. Makhija, B. Herndon, VMmark: A Scalable Benchmark for Virtualizaed Systems, Technical Report VMware-TR-2006-002, September 2006.

[10] H. Levkowetz and S. Vaarala. Mobile IP Traversal

of Network Address Translation (NAT) Devices. RFC 3519, IETF, April 2003.

[11] OpenVPN, TUN/TAP, http://openvpn.net, Retrieved October 2007.

[12] K. Adams and O. Agesen, A Comparison of Software and Hardware Techniques for x86, ASPLOS'06, pp. 1~13, October 2006

[13] Wikipedia. "Platform virtualization". http://en.wikipedia.org/wiki/Virtualization, August 2007.

[14] R. Eckstein, D. Collier-Brown, and P. Kelley, Using Samba, O'Reilly Associates Inc, 2003.

[15] A. Mockus, R. T. Fielding, J. Herbsleb, A case study of open source software development: the Apache server, ICSE, pp.263~272, 2000.

[16] D. Axmark, M. Widenius, MySQL Introduction, Linux Journal Volume 1999.

[17] M. Fioretti, Top ten tips for getting started with PHP, Linux Journal Volume 2006.

[18] P. Mishra, K.K. Ramakrishnan, and Jacobus E. van der Merwe, A Flexible Model for Resource Management in Virtual Private Networks, SIGCOMM '99 8/99, pp. 95~108, 1999.

[19] OpenVPN, TUN/TAP, http://openvpn.net, Retrieved October 2007.

[20] Ludmila Cherkasova and Rob Gardner. Measuring CPU overhead for I/O processing in the Xen virtual machine monitor. In Proceedings of the 2005 USENIX Technical Conference, Annaheim, CA, USA, pp. 387 – 390, April 2005.

◐ 저 자 소 개 ◑

우 수 정 (Sujeong U)
2006년 원광대학교 전기전자 및 정보공학부/컴퓨터 및 정보통신공학 학사
2008년 전북대학교 컴퓨터공학 석사
2008년~현재 전북대학교 컴퓨터공학 박사
관심분야 : 정형기법, 소프트웨어공학, etc.
Email : wpig04@gmail.com

온 진 호 (Jin-Ho On)
2006년 원광대학교 컴퓨터 정보통신공학 학사
2008년 전북대학교 컴퓨터공학 석사
2009년~현재 전북대학교 컴퓨터공학 박사
관심분야: 분산/병렬처리시스템, 정형기법
Email: jjinghott@gmail.com

최 정 란 (Jung-Rhan Choi)
1999년 전북대학교 컴퓨터과학과 이학사
2001년 전북대학교 컴퓨터통계정보 이학석사
2007년 전북대학교 컴퓨터통계정보 박사
2008년~현재 고려대학교 BK21 소프트웨어 산학연 연구교수
관심분야 : 정형기법, 소프트웨어공학, RFID, etc.
Email: seohyun@formal.korea.ac.kr

최 완 (Wan Choi)
1983년 KAIST 공학 석사
1985년~현재 한국전자통신연구원 연구원
관심분야 : 소프트웨어 공학, 미들웨어, 소프트웨어 스트리밍
Email : wchoi@etri.re.kr

이 문 근 (Moon-Kun Lee)
1989년 The Pennsylvania State University, Computer Science 학과 이학사
1992년 The University of Pennsylvania, Computer and Information Science 학과 이공학석사
1995년 The University of Pennsylvania, Computer and Information Science 학과 이공학박사
1992년 5월~1996년 1월: 미국, Computer Command and Control Company, Computer Scientist로 근무
1996년~1998년 전북대학교 컴퓨터과학과 전임강사
1998년~1998년 전북대학교 컴퓨터과학과 조교수
1999년~2002년 전북대학교 공과대학 전자정보 공학부(컴퓨터과학과 전공) 조교수
2002년~2007년 전북대학교 공과대학 전자정보 공학부(컴퓨터과학과 전공) 부교수
2008년~현재 전북대학교 공과대학 전자정보 공학부(컴퓨터공학과 전공) 교수
관심분야 : 정형기법, 소프트웨어 재·역공학, 실시간 시스템, 운영체제, 형식언어, 병렬함수 언어, 컴파일러 등
Email : moonkun@chonbuk.ac.kr