Evaluation of Switching Performance of a Virtual Software Router

Roberto Rojas-Cessa, Khondaker M. Salehin, and Komlan Egoh

***Abstract*—Software routers are an alternative low-cost and moderate-performance router solutions implemented with general-purpose workstations able to host two or more network interface cards (NICs). Workstations can be programmed to forward packets between different NICs and to participate in routing functions. The value of software routers lies on their low cost and on the flexibility to modifying routing and switching functions. Machine virtualization can be used to model novel protocols or hardware systems, however, implemented in software and without modifications to the host's kernel. This virtualization allows the implementation of not only one but also multiple (and independent) virtual systems. Virtualization of software routers, called virtual software routers, is then a possible application of this technology. However, because of the software platforms, virtualized machines are expected to suffer from performance degradation. In this paper, we investigate the switching performance of a virtual software router and compare it to that of a software router. We present the performance of virtual software routers hosted by different workstations, with single and multiple processing cores.**

***Index Terms*—Linux, virtualization, software router, virtual router, stress testing.**

I. Introduction

Internet routers are the major unit systems in the Internet as they are in charge of forwarding packets from one link to another in the task of approaching packets to their destinations. Routers mostly perform network layer and data-link layer functions, such as construction of routing tables and communication with neighboring routers to exchange link information for the enabled routing protocols, e.g., Routing Information Protocol (RIP) and Open Shortest Path First (OSPF). Other functions include switching of packets from one interface to another and lookup for destination interface address according to the (Internet) destination address of the packet. The destination of a packet is compared to a set of destined Internet Protocol (IP) addresses, which are obtained through routing, stored as a prefix (a set of IP addresses represented by the common bits of the included addresses, generally presented as *IP address/prefix length*, where prefix length indicates the number of the most significant bits considered for matching with a packet's destination IP address).

The speed of a router is determined by the latencies of performing IP lookup and packet switching. These functions are often performed in the so-called *data plane*. It is desirable that a router provides a switching rate that equals the transmission speeds of data-link layer interfaces (e.g., Gigabit Ethernet, Fast Ethernet).

Commercial routers can be found for different network applications and in different network zones (e.g., network core or edge). Among these, high-performance and high-capacity routers are targeted for applications that demand high bandwidth for high traffic volumes. However, routers for office and home networks (i.e., at network edge) have less stringent performance requirements and they can be implemented with general-purpose personal computers (PCs) provisioned with two or more network interface cards (NICs) and software to perform packet forwarding. This software can be found ubiquitously in Linux operating systems (OS). These ad-hoc routers are called *software routers* [1]-[4].

Experimentation on Internet protocols using commercial routers is difficult to perform as the OS of these routers are proprietary. Software routers, based on open-source Linux (and other derivatives), are a straight-forward and attractive option. However, the architecture of general-purpose PC limits the performance of switching functions. A large measure of this restricted performance is caused by the high degree of resource (e.g., memory and buses) sharing provided in that architecture. Performance evaluation of software routers have been of recent interest, and multiple tests have shown that a modest switching performance can be achieved [5]. Nevertheless, for small networks, this performance may suffice.

Experimenting with new protocols and network paradigms often require embracing concepts beyond those considered in the TCP/IP protocol suite. These new protocols can be implemented in the open-source Linux code. However, feasibility of doing so has been facilitated by machine virtualization. In virtualization, an OS (e.g., MS Windows or Linux) can run another OS as one or more virtual machines, which at the same time run different protocols from those run by the host, while providing an underlying unified platform that requires simple administration and maintenance functions.

A software router running in a virtual environment is then called a *Virtual Software Router*. A host workstation may be able to allocate a single or multiple virtual software routers [4], and multiple protocols. Differently from a software router, a virtual software router runs most of the functions on software (while a so-called software router has the interfaces running on hardware). Therefore, the performance of a virtual software router is expected to be lower than that of a software router. However, a question remains: what is the throughput penalty of a virtual software router?

In this paper, we answer this question by investigating the switching performance of virtual software routers. Furthermore, we also investigate how this performance is improved with the adoption of multiple-core processing units of modern computer systems. We perform a series of experiments in a testbed under multiple flow scenarios, with packets of different sizes and rates, to estimate the actual switching performance of virtual software routers. These experiments are performed using PCs with multiple interfaces and different number of processing cores (e.g., one and two processing cores), and

The authors are with the Networking Research Laboratory, Dept. of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102, USA. Email: {rojas, kms29, ke6}@njit.edu.

under one, two, and four traffic flows. The resulting experiments suggest that multiple cores may be highly beneficial for virtualization.

The remainder of this paper is organized as follows. Section II describes a summarized model of a general-purpose PC used to host virtualization. Section III presents the setup and experimental outcomes in our evaluations. Section IV presents our conclusions.

II. Software Router Model

The hardware architecture of a software router is basically the architecture of a PC, with two or more NICs. A PC architecture has basically one or more central processing units (CPUs), where each can host one or multiple processing cores, a chipset to operate in conjunction with the CPUs, main memory (blocks), and network cards. These different subsystems are interconnected by means of buses, a front-end bus to connect the CPUs and the chipset, a bus connecting the main memory and the chipset, and a PCI bus to interconnect the chipset with the network cards. This simplified architecture is shown in Figure 1.

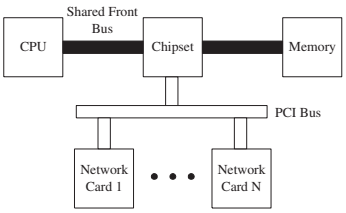


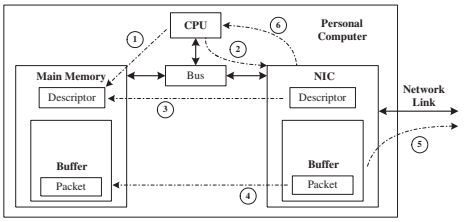
Fig. 1. Top-level architecture of a general-purpose personal computer.

The performance of a software router also depends on the processing speed of the CPU and the handling of memory by the CPU or chipset. An accurate estimate of the time a CPU would take to process packet forwarding can be performed by counting the number of clock cycles it takes to execute the code for forwarding a packet plus the time it takes to provide maintenance tasks of the PC. An alternative is to measure the time increase that packet processing would take beyond the achievable speed as limited by the different hardware components of the PC. Shooting packets to the router at high data rate and accounting the number of packets leaving from it can measure this. We call this a stress test.

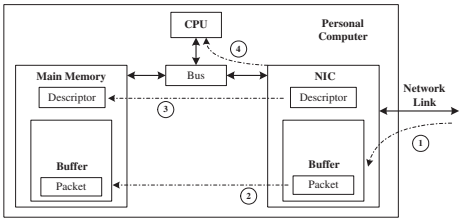
A. Network Interface Processing

The NICs are connected to a peripheral bus, being the Peripheral Component Interconnect (PCI) running at 133 MB/s (32-bit at 33 MHz), 266 MB/s (32-bit at 66 MHz or 64-bit at 33 MHz), or 533 MB/s (64-bit at 66 MHz), the most popular interface. Other peripheral buses, such as the Universal Serial Bus (USB), are also available. The remainder of this paper, however, refers to PCI NICs as the technology used to interconnect NICs to the host computer. We first discuss the basic operations of NICs below.

Figure 2 illustrates the packet transmission and receiving processes of a NIC [6]. For the transmission of a packet (Figure 2(a)), the NIC driver initially creates buffer descriptor(s) containing the location (i.e., address and length) of the packet in the main memory of the PC and interrupts the NIC



(a) Transmission process



(b) Receiving process

Fig. 2. Basic NIC operations: (a) transmission process and (b) receiving process.

in Steps 1 and 2, respectively. The packet is then copied to the NIC buffer through two Direct Memory Access (DMA) transfers (i.e., one for the packet descriptor(s) and the other for the packet itself) in the following two steps. The sending process is completed by transmitting the packet to the network through the network interface (Step 5) and then sending an interrupt to the CPU (Step 6). When a packet arrives at the NIC buffer (Step 1), as shown in Figure 2(b), the NIC initiates the receiving process by copying the packet to the main memory of the PC along with its descriptor(s) through two DMA transfers (Steps 2 and 3). The NIC sends an interrupt the CPU (Step 4) and finishes the receiving process.

B. Virtual Machine Software

A virtual machine is a software implementation of a computer system on top of an operating system that can be run in a manner similar to a real physical machine. By abstraction of the physical hardware through software, virtual machine technology offers a flexible solution for running concurrent systems on a single physical machine and for migrating fully configured systems between heterogeneous hardware platforms. This leads to a greater hardware consolidation, ease of management, and cost saving [7].

In recent years, virtualization has seen a rapid growth and now permeates in many aspects of information technology. In on-demand network grids or cloud systems, for example, the end-node computers, such as database servers, web servers, and storage systems, have been widely virtualized. Virtual machines are implemented on physical systems to accommodate increased demands [8]. As the requirements for additional consolidation and scalability arise, the need to virtualize other elements of the network, e.g., routers, is anticipated. Software based virtual routers (i.e., virtual software router) may however come with an increased performance overhead. It is therefore necessary to measure the performance impact of virtualization on software routers.

C. Experimental Performance Evaluation

We performed a set of experiments on a network tested with software and virtual software routers to observe the perfor-

mance of virtualization of software routers. This performance comparison is based on the performance of Linux software routers. The software and virtual software routers were implemented with workstations with different CPU specifications. The specifications of each workstation are summarized in Table I. In general, we tested the performance of a software router and a virtual software router on workstations, each with a single CPU (Intel Pentium CPU) and a CPU with two cores (Intel Core 2). The experiment had the objective to observe the performance improvement achieved by increasing the number of processing cores.

TABLE I
WORKSTATION SPECIFICATIONS

	Dell Precision 670	Dell Optiplex 780
Name	Dell 670	Dell 780
Processor	Intel(R) Xeon(TM)	Intel(R) Core(TM) 2
No. of processor	1	2
CPU speed	3.2 GHz 3.17 GHz	
RAM	2 GB	4 GB
Interface speed	10/100 Mb/s	
Operating systems	Ubuntu 10.10 (kernel 2.6.22-35-generic) and MS Windows 7	
Virtualization software	VMware Workstation 7.1	

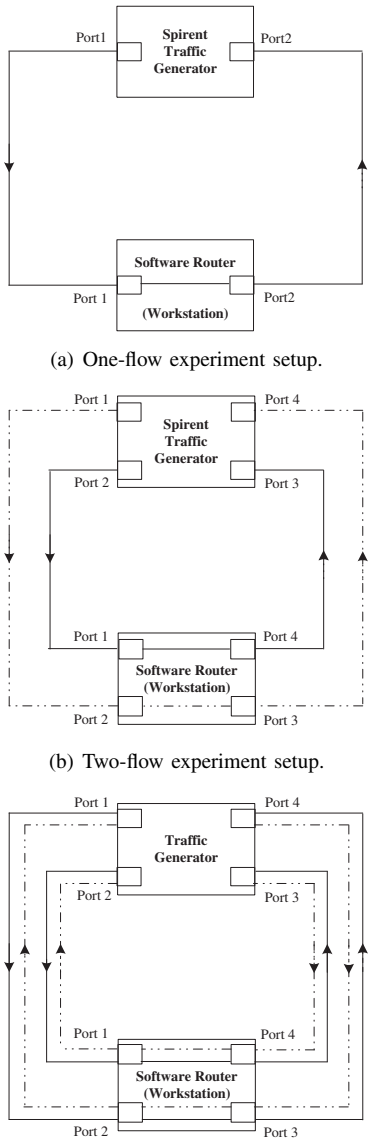
III. Test Setup

Different software routers were put under stress test, which consists of the transmission of packets at a constant-bit-rate (CBR) with constant-length packets. We used different flow rates by changing the packet length, from 64 bytes up to 1472 bytes (excluding the preamble field). The tested machines consisted of three workstations, each configured and tested separately as a software router, and a traffic generator, a Spirent SmartBits 6000C Performance Analysis System. The SmartBits has Gigabit Ethernet ports that can inject traffic and ports that can receive traffic for statistical analysis. The stress test is run for a period of time for a given packet size. The chosen test time is 30 seconds.

We observed the throughput for each system under test to perform packet forwarding under one, two, and four flows at different rates. The flows were routed through to different NICs by the system under test. Figure 3 shows the tested setup for stress-test experiments with one, two, and four flows. For the experiment with one flow, two NICs are used. The flow goes from one interface to the other as shown in Figure 3(a). For the experiment with two flows, four NIC cards are used, forming two NIC pairs, where a single flow (denoted by solid or dashed line) passes through a pair of NICs, as shown in Figure 3(a). For the experiment using four flows, four NICs were also used, and each pair of NICs hosted two flows (denoted by solid and dotted lines), one flow in each direction, as shown in Figure 3(c). The NICs used are D-link and Realtek FastEthernet cards (100 Mb/s).

The Spirent SmartBits system generates traffic flows and calculates throughput upon receiving the traffic back (flows egress and ingress to the SmartBits system through different ports, as shown in Figure 3).

The Linux distribution used in these experiments is Ubuntu 10.10 (Linux kernel version 2.6.35-22-generic) in software routers. The virtual software routers use the same Linux distribution but on top of a MS Windows 7 platform. VMware Workstation 7.1 [9] was installed and setup with the hardware configurations as shown in Table I on MS Windows 7 and



(a) One-flow experiment setup.

(b) Two-flow experiment setup.

(c) Four-flow experiment setup

Fig. 3. Experimental setups for tests of one, two, and four flows.

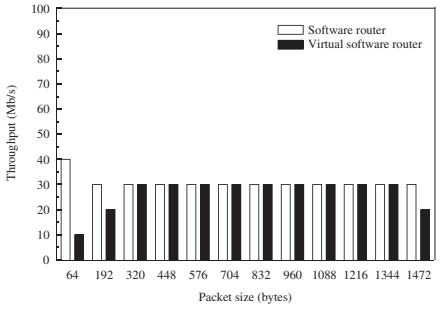
a virtual Ubuntu 10.10-machine was generated. Linux default values were used in all tests. We selected these OSs and the VMware virtual machine software in our experiment because they are popular and available.

In this paper, throughput is referred to as the maximum data rate that the system under test can pass to the destined NIC.

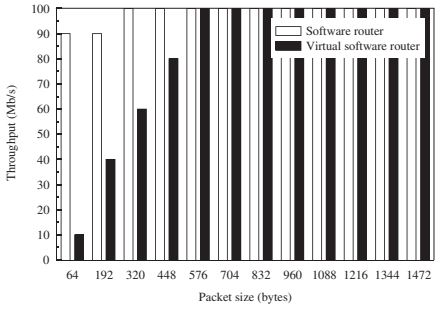
A. Forwarding of one flow

In the test under one flow (see Figure 3(a)), two workstations were tested: Dell 670 (with a single core/CPU) and Dell 780 (with dual cores). Figure 4 shows the results obtained by these two workstations under this test. Figure 4(a) shows that the Dell 670 achieves up about 30 Mb/s of throughput as a software (Linux) router. As a virtual software (Virtual Linux) router, the Dell 670 achieves about 10 Mb/s for the smallest packet size (64 bytes), and up to 30 Mb/s for larger packet sizes. As for the Dell 780 workstation, the throughput of the software router is 90 Mb/s under 64- and 192-byte

packets, and 100 Mb/s for larger packets, as shown in Figure 4(b). The virtual software router on this workstation achieves 10 Mb/s for 64-byte packets, and it keeps increasing as the packet size increases. It achieves 100 Mb/s of throughput for 576-byte packets and larger packet sizes. According to Figure 5, virtual software router cannot achieve high throughput for small packet sizes, e.g., 64 bytes, even on the Dell 780 workstations. With small packet sizes, the forwarding speed of the virtual software router cannot keep with the increasing data rate. Therefore, more packets are dropped by the NIC buffer upon arrival because of the packet processing delay is larger than the transmission delay of a small packet on the virtual software router.



(a) One flow on the Dell Precision 670 (single Intel Pentium processor)

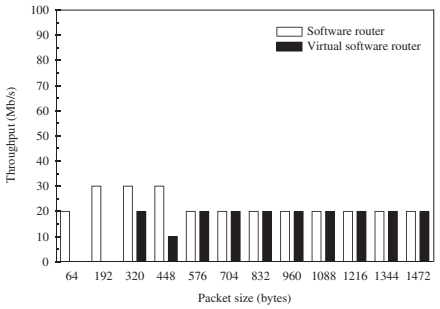


(b) One flow test on the Dell Optiplex 780 (Intel Core 2 processor)

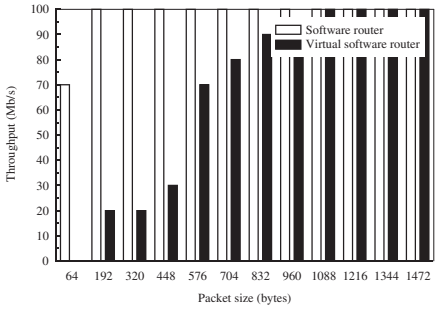
Fig. 4. Performance of software (Linux) and virtual software (Virtual Linux) routers under one flow.

B. Forwarding of two flows

Figure 5 shows the throughput results under two flows (4 NICs, as shown in Figure 3(b)) for the same workstations. Figure 5(a) shows that the achieved throughput of software route on the Dell 670 workstation is 20 Mb/s for both small and larger packet sizes, e.g., 64- and 1472-byte packets. The virtual software router on this workstation achieves 10 Mb/s for 448-byte packets and 20 Mb/s for larger packets. Therefore, the throughput performance of the virtual software router degrades under small packet sizes (the same reason as before, the processing time of the virtual software router is longer than the processing time of a small packet). On the Dell 780 workstation, the throughput achieved under 64-byte packets is 70 Mb/s and it increases to 100 Mb/s for 192-byte and larger packets in Figure 5(b). The virtual software router on this workstation achieves 20 Mb/s of throughput for 192-byte packets and it reaches up to 100 Mb/s for 1088-byte and larger packets.



(a) Two flows on Dell 670 (single Intel Pentium processor).



(b) Two flows on Optiplex 780 (Intel Core 2 processor).

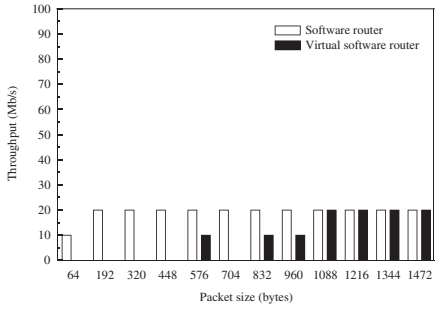
Fig. 5. Performance of a software (Linux) and virtual software (Virtual Linux) routers under two flows.

C. Forwarding of four flows

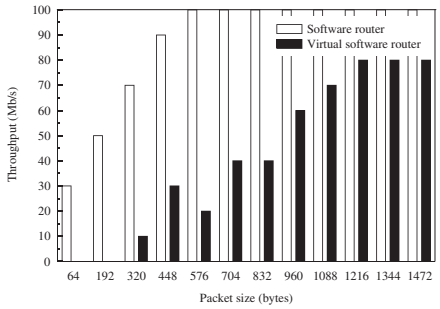
Figure 6 shows the experimental results of the stress test under four flows (4 NICs, as shown in Figure 3(c)). Figure 6(a) shows the results of the Dell 670 workstation. This figure shows that the software router can achieve a throughput up to 20 Mb/s (and 10 Mb/s for 64-byte packets) under four flows. However, the virtual software router achieves 10 Mb/s for 576-byte packets and it increases to 20 Mb/s for 1088-byte and larger packets. On the Dell 780 workstation, the throughput achieved by the software router is 30 Mb/s for 64-byte packets, and this increases linearly proportional to the packet length, up to 100Mb/s for 576-byte packets. The throughput remains at this value for larger packet sizes, as shown in Figure 6(b). Also, the virtual software router in this workstation achieves 10 Mb/s for 320-byte packets and the throughput slowly increases, up to 80 Mb/s for larger packet sizes, e.g., 1216-byte packets. As shown in the four-flow experiment, the Dell 780 workstation has significantly higher throughput performance than the Dell 670 workstation.

D. Throughput of 1472-byte packets for different number of flows

From the experiments for one, two, and four flows, we observed that virtual software routers achieve higher throughput with larger packets. Therefore, we performed a set of experiments for the packet size where the Dell 780 workstation achieved better performance in the previous experiments.

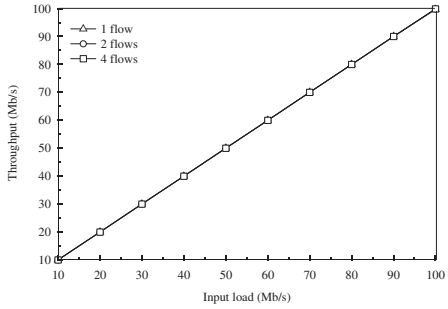


(a) Four flows on Dell Precision 670 (single Intel Pentium processor)

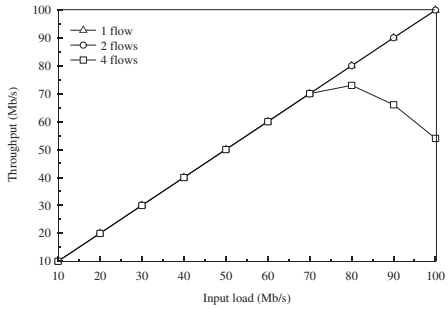


(b) Four flows on Dell Optiplex 780 (Intel Core 2 processor)

Fig. 6. Performance of software (Linux) and virtual software (Virtual Linux) routers under four flows.



(a) Software router



(b) Virtual software router

Fig. 7. Throughput of Dell Optiplex 780 PC as software (Linux) and virtual software (Virtual Linux) routers under 1472-byte packets.

Figure 7 shows the obtained results by the software router and by the virtual software router under a packet size of 1472 bytes. Figure 7(a) shows that the software router achieves 100 Mb/s (100%) for one, two, and four flows under this packet size.

However, the virtual software router (Figure 7(b)) achieves a throughput of 100 Mb/s under one and two flows, and up to 70 Mb/s under four flows. In the case of four flows, the throughput under higher input loads, i.e., 80 Mb/s and above, decreases (rather than remaining at the maximum throughput) to 55 Mb/s. This is because as input traffic load increases, packets continue to be received by the OS (i.e., MS Windows 7) and that increases the processing load. This overload affects the performance of virtual router software and the performance of the virtual machine continues to degrade as the traffic increases.

IV. Conclusions

In this paper, we presented a performance evaluation of virtual software routers and showed the performance penalty incurred by virtualization. The performance degradation of a virtual software router is expected to be lower than that of a software router, but the presented examples show the magnitude of the performance degradation. The results showed that a virtual software router cannot support high-speed transmission rates for small-size packets. In addition, the throughput performance presented shows some variability on the obtained values. The obtained values suggest that virtualization has stringent requirements and the advent of multicore systems have a venue of growth in this area. However, some other strategies may also help improve the performance of virtual Linux machines in modern processors as in some cases, multiple processing units may not show an obvious performance improvement, as observed in our experiments. Other possible solutions to virtualization can be found in peripheral processing, such as Graphics Processing Unit (GPU) [10].

References

[1] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M.F. Kaashoek, "The Click Modular Router," ACM Transactions on Computer Systems, vol. 18, no. 3, pp. 263-297, Aug. 2000.

[2] A. Bianco, J.M. Finochietto, G. Galante, M. Mellia, and Fabio Neri "Open-Source PC-Based Software Routers: A Viable Approach to High-Performance Packet Switching," Proc. QoS-IP 2005, pp. 353-366.

[3] R. Bolla, and R. Bruschi, "Linux Software Router: Data Plane Optimization and Performance Evaluation," Journal of Networks, Vol. 2, no. 3, pp. 6–17, Jun. 2007.

[4] A. Bianco, R. Birke, L. Giraudo, and N. Li, "Multistage Software Routers in a Virtual Environment," Proc. IEEE Globecom 2010, Miami, FL, December 2005, pp. 1–5.

[5] M. Dobrescu, N. Egi, K. Argyraki1, B.G. Chun, K. Fall, G. Iannaccone, A. Knies, M. Manesh, and S. Ratnasamy, "RouteBricks: Exploiting Parallelism To Scale Software Routers," Proc. ACM SOSP 2009, October 11-14, 2009, pp. 15–28.

[6] P. Willmann, H.Y Kim, S. Rixner, and V.S. Pai, "An Efficient Programmable 10 Gigabit Ethernet Network Interface Card," Proc. 11th Int. Symposium on High-Performance Computer Architecture 2005 (HPCA-11 2005), pp. 96–107.

[7] J. Che, Q. He, Q. Gao, and D. Huang, "Performance Measuring and Comparing of Virtual Machine Monitors," Proc. IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, 2008. EUC '08. , vol.2, no. 17-20, pp. 381–386, Dec. 2008.

[8] Y. Li, W. Li, and C. Jiang, "A Survey of Virtual Machine System: Current Technology and Future Trends," Proc. 2010 Third International Symposium on Electronic Commerce and Security (ISECS), Guangzhou, China, Jul. 2010, pp.332–336.

[9] Vmware [Online]. Available: http://www.vmware.com/products/workstation/.

[10] S. Han, K. Jang, K. Park, and S. Moon, "PacketShader: a GPU-Accelerated Software Router," Proc. ACM SIGCOMM, 2010, 12 pp.