Critical Review of

*Optimizing Network Virtualization in Xen*

Christopher D. Krieger
Colorado State University
Fort Collins, Colorado 80526
krieger@cs.colostate.edu

November 12, 2009

1 Overview

This paper describes a series of optimizations the authors performed on the Xen network IO subsystem. The original Xen virtualization architecture was introduced in [2]. The idea of separate I/O domains was described in [3].

This work was able to achieve significant performance gains in both transmit and receive throughput by employing a range of hardware and software optimizations. Most of the optimizations revolve around sending larger chunks of data between the driver domain and the guest domains. This is achieved by deferring the TCP segmentation of data until after the inter-domain crossing and using superpages (4MB page sizes).

2 Inefficiencies and Potential Areas for Improvement

The performance improvements cited in the paper are impressive, but are rather contrived. For example, approximately half of the performance improvement shown for transmission throughput can be attributed to using the hardware acceleration provided by the high-end network cards the authors used. These network cards handle TCP segmentation offload, TCP/IP checksum offload, and scatter/gather DMA all within the NIC, thereby exaggerating the benefit of the presented optimizations. Likewise, note that the results are from a “netperf-like” benchmark rather than the standard netperf itself. I suspect this is so that the sendfile() system call could be used, a choice that permits the use of scatter/gather in the NIC.

The paper also discusses at length the benefits and disadvantages of using global page mappings. These are mappings that remain in the TLB across domain switches and are shared by all processes. However, in the given implementation, they persist only between a VM and the driver domain. Each time a switch occurs to another domain, they are all purged. As a side effect, all TLB entries, both for the VM and the hypervisor, are purged. This results in a performance loss if guest domains are used, defeating the point in most Xen installations. Most x86 processors in the past few years have virtualization extensions that allow a TLB entry to be tagged with a VM identifier. This allows for domain switching without having to purge TLB entries.

While many of the performance enhancements mentioned in this paper support the use of unmodified device driver code, others are far more intrusive. For example, the use of superpages requires significant modification to the guest OS. Since at present these modifications to the guest OS are only used to improve network performance, it seems like a high price to pay for the relatively modest performance improvements unless networking is the primary function of the virtual machine.

3 Possible Extensions

The paper notes that approximately 30% of the processing overhead transferring data between a guest OS and the physical network is spent in Xen. This is time spent not in the guest OS or in the driver domain, but
rather in routing data between the two via the I/O channel and bridge/router. While this paper presents significant improvements, it does not address the possibility of removing the channel altogether. It should be possible to assign hardware resources directly to a guest OS. These would not be virtualized, but would have to be exclusively assigned to a single OS instance. While doing so eliminates many of the advantages of I/O virtualization, such as physical hardware sharing, it would allow a systems administrator to install multiple network cards into a machine and then have all networking run at full speed. The Xen VMM would have to be involved during device discovery and would have to assign some physical addresses to the hardware that were mapped into the proper address space of the owning guest OS. Page table entries would have to be added to the guest OSes to map to those same physical addresses. The problem here is that much of the I/O based data transfers are driven by DMA, which would have to be programmed with the correct address translations. This is addressed in [1].

References


