Efficient Inter-domain Network Communication in Virtualized Environments Using Shared Memory

Stefanos Gerangelos and Nectarios Koziris *{sgerag,nkoziris}@cslab.ece.ntua.gr*

National Technical University of Athens CSLab

Motivation

- There is an emergent request for many Cloud environments to achieve better network performance between Virtual Machines running in their infrastructure.
- Many deployments require more Virtual Machines than the available physical machines in the cluster, turning the inter-domain communication between VMs into a critical factor in the overall system's performance.
- By characterizing a communication as inter-domain, we are referring to the interaction between two or more virtual machines located in the same physical host.

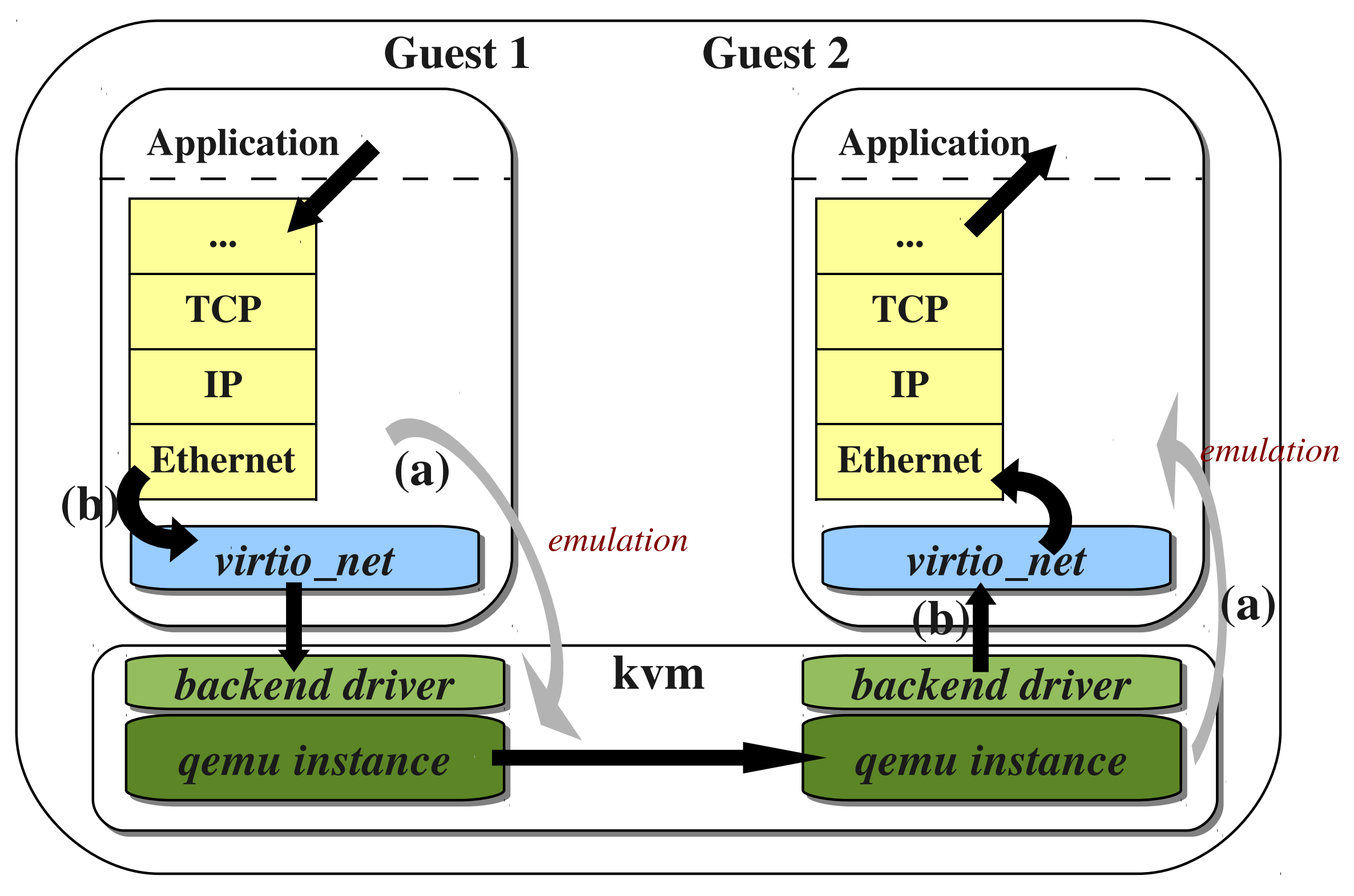
Contribution

We propose a framework for inter-domain network communication over Ethernet using shared memory techniques eliminating the intervention of the host component in the critical path.
Most of previous research has been focused on implementation concerning the Xen hypervisor. Our approach is based on the **kvm** hypervisor and *linux* (as the guest and the host operating system). Our primary concern was to produce a framework, which does not export a new API to the application level, so the respective applications won't have to be changed in order to take advantage of the new I/O path. With this in mind we ended up with a framework, which includes minor modifications to the kvm hypervisor as well as the addition of a driver in the guest side in order to support our new virtual network device.

Possible Drawbacks

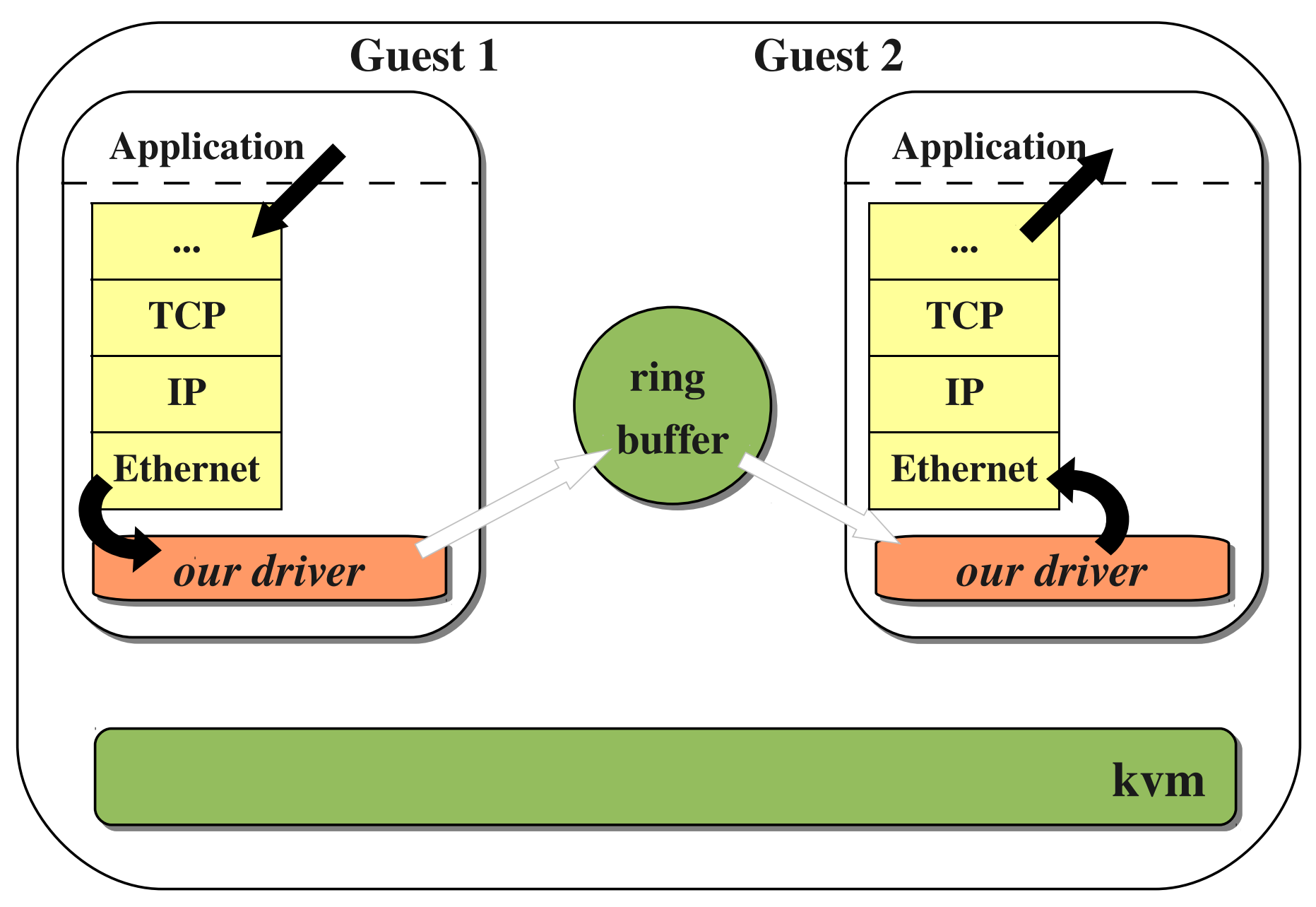
- In a scenario where several VMs decide to send packets through our device the access to the ring buffer is expected to become a non-negligible bottleneck.
- The system administrator of the guest must be aware of the new interface and the respective routing table entry and should not try to do anything fancy with the routing table, which would be out of the predicted procedure concerning the inter-domain communication.

Current I/O Path



The current I/O path includes either:
(a) emulation of network I/O (grey arrow) with unmodified kernel, but poor performance or
(b) use of the split-driver model (using the virtio driver), which results in better performance. However, even following this approach, the intervention of the host remains high.

Proposed I/O Path



Our proposed framework includes the allocation of a shared memory space by the host, which will be exposed to each guest every time one is created. So, when the kvm module is loaded, it allocates a number of memory pages and creates a ring buffer, which is the structure that every guest is going to use for its inter-domain communication. This model requires the implementation of a virtual network device for the guest. This device is added to the routing table of the guest with a respective rule concerning the inter-domain subnet. The driver of this device is based on the *virtio* interface.
A typical scenario of using our framework follows:

- Upon the installation of the shared ring buffer by the kvm host, a guest (*guest 1*) is created.
- The host maps the shared ring buffer to the address space of *guest 1*.
- After that suppose that another guest (*guest 2*) is created. The host maps the same memory in the address space of *guest 2*.
- Now, if *guest 1* decides to send a network packet to *guest 2*, it has to lock for write the ring buffer, place the respective socket buffer (*sk_buff* struct in linux kernel) and make the appropriate changes to the respective ring buffer pointer.
- *Guest 2*, which is watching the ring buffer, checks the ip header of the *sk_buff*, realizes that this buffer is intended for itself and continues for processing it after it has updated the relevant pointer in the ring buffer.

Future Plans

- Evaluation of this framework.
- Add support for low latency protocols (like Myrinet, Infiniband), which are highly used in the HPC world.

References

[1] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. A. Pratt, and A. Warfield. Xen and the art of virtualization. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 164- 177, New York, NY, USA, 2003. ACM.
[2] F. Bellard. Qemu, a fast and portable dynamic translator. In *ATEC '05 Proceedings of the annual conference on USENIX Annual Technical Conference*, 2005.
[3] F. Diakhate, M. Perache, R. Namyst, and H. Jourdren. Efficient shared memory message passing for inter-vm communications. In *4th Workshop on Virtualization in High-Performance Cloud Computing (VHPC '08), Euro-par 2008*, 2008.
[4] O. K. Hamid Reza Mohebbi and M. Sharifi. Zivm: A zero-copy inter-vm communication mechanism for cloud computing. In *Computer and Information Science*, pages 18-27, 2011.
[5] Z. Hongyong, G. Kuiyan, L. Yaqiong, S. Yuzhong, and M. Dan. A highly efficient inter-domain communication channel. In IEEE Ninth International Conference on Computer and Information Technology, 2009.
[6] Y. S. Hongyong Zang and K. Gu. Optimizing inter- domain communication. In *15th International Conference on Parallel and Distributed Systems*, 2009.
[7] K. Kangho, K. Cheiyol, J. Sung-In, S. Hyun-Sup, and K. Jin-Soo. Inter-domain socket communications supporting high performance and full binary compatibility on xen. In *ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, 2008.
[8] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liquori. kvm: the linux virtual machine monitor. In Linux Symposium, pages 225-230, Ottawa, Ontario, Canada, 2007.
[9] R. Russel. virtio: Towards a de-facto standard for virtual i/o devices. In ACM SIGOPS Operating Systems, 2008.
[10] M. J. K. Wei Huang and D. K. Panda. Efficient one-copy mpi shared memory communication in virtual machines. In IEEE International Conference on Cluster Computing, 2008.
[11] Q. G. Wei Huang, Matthew J. Koop and D. K. Panda. Virtual machine aware communication libraries for high performance computing. In *IEEE conference on Supercomputing*, 2007.
[12] P. R. Xiaolan Zhang, Suzanne McIntosh and J. L. gridn. Xensocket: A high-throughput interdomain transport for virtual machines. In *International Conference on Middleware*, 2007.