High Performance Network I/O for Virtual Machines

L. Rizzo, G. Lettieri, V. Maffione

Dipartimento di Ingegneria dell'Informazione
University of Pisa

KVM Forum, 2013

Outline

1. The Problem
2. netmap integration
   - netmap API
   - Integration in QEMU
   - Performance
3. Fast e1000
   - Driver and emulation improvements
   - Paravirtualization
   - Integration with VALE
   - Performance
4. Open problems

Problem

- Improving pps throughput between virtual machines and vm/host.
- Desirable for middlebox virtualization

Our solution

Try to use netmap

netmap

Home page

`http://info.iet.unipi.it/~luigi/netmap/`

- device-independent ring/buffers in shared memory
- kernel/userspace sync only during syscalls (`ioctl`, `poll`)
- 14.88 Mpps on a 10GigE with a single 900 Mhz core
- standard in FreeBSD since 9.1-RELEASE
- distributed as a separate module for Linux (2.6.32–3.11)

netmap API

```
fd = open("/dev/netmap", 0);
strcpy(req.nr_name, "eth0");
ioctl(fd, NIOCREGIF, &req);
mem = mmap(NULL, req.nr_memsize, PROT_READ|PROT_WRITE, 0, fd, 0);
nifp = NETMAP_IF(mem, req.nr_offset);
ring = NETMAP_RX_RING(nifp, 0);
for (;;) {
    ...
    poll(/* fd */ ...);
    for ( ; ring->avail > 0; ring->avail--) {
        i = ring->cur;
        buf = NETMAP_BUF(ring, i);
        use_data(buf, ring->slot[i].len);
        ring->cur = NETMAP_NEXT(ring, i);
    }
}
```

VALE: Virtual Local Ethernet

Home page

`http://info.iet.unipi.it/~luigi/vale/`

- An extensible switch using netmap API
- Already included in the netmap module
- connects:
  - virtual ports (netmap API only)
  - netmap enabled real NICs
  - the host stack
- uses batching to improve pps (≈20 Mpps for 64 B pkts between two VPs)

VALE API

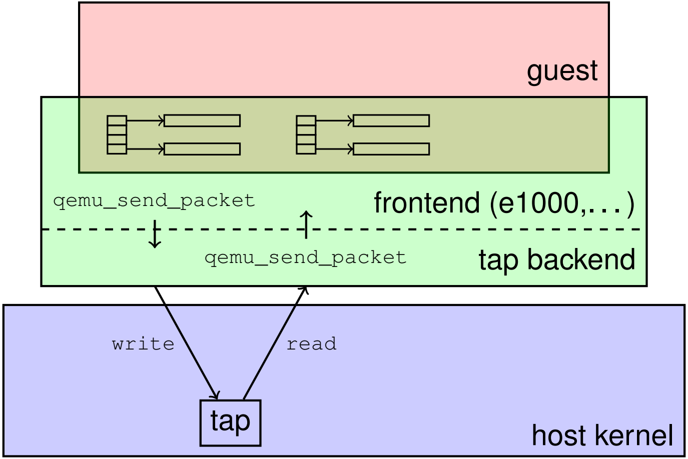
Use special names starting with “vale”:

```
fd = open("/dev/netmap");
strcpy(req.nr_name, "valeA:x");
ioctl(fd, NIOCREGIF, &req);
```

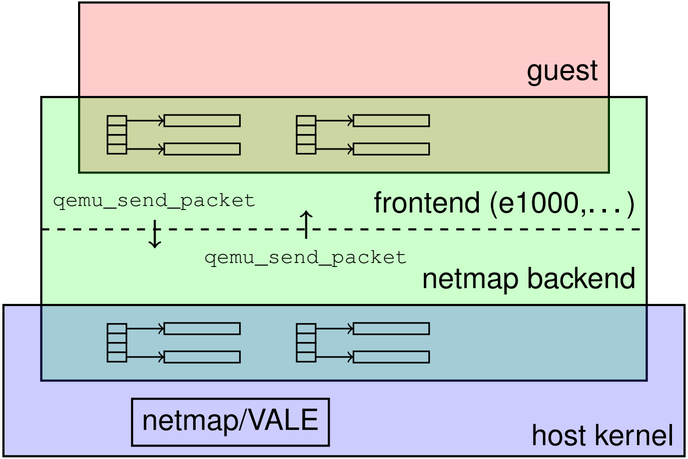
Then, same as before.

- virtual ports and switches are created on the fly
- all virtual ports with the same name before the “:” are connected to the same switch

QEMU networking



QEMU-VALE integration



QEMU-VALE usage

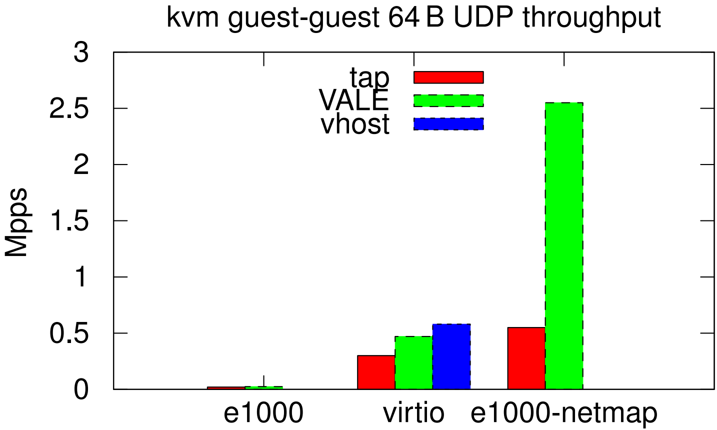
Attach vm to port `x` of VALE switch `valeA`

```
qemu -device e1000,netdev=n,mac=...
  -netdev netmap,id=n,ifname=valeA:x ...
```

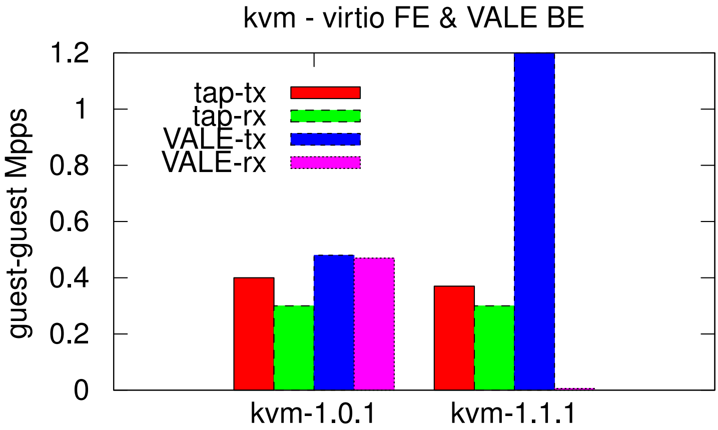
Attach another vm to port `y` of the same VALE switch.

```
qemu -device e1000,netdev=n,mac=...
  -netdev netmap,id=n,ifname=valeA:y ...
```

Initial performance (1/2) [1]



Initial performance (2/2) [1]



Initial performance: problems

- terrible base performance of emulated devices
- a bit better for virtio, but not as much as expected
- unstable results
  - slightly slower tx → big improvement w VALE
  - huge packet drops

Fast e1000

- Interrupt moderation (only emulator modified)
- Send combining (only guest driver modified)
- Paravirtualization (both emulator and driver modified)
- Better integration with netmap/VALE

Interrupt moderation and Send combining

Interrupt moderation

- helps amortizing per-packet overheads
- Already merged in QEMU

Send Combining

- Don’t kick the host when a TX interrupt is pending
- Flush pending transmissions when the interrupt comes
- Works well with moderation (bigger batches)

Importing the essence of virtio to e1000

Real hw and emulated TX

TDT register writes used for both:

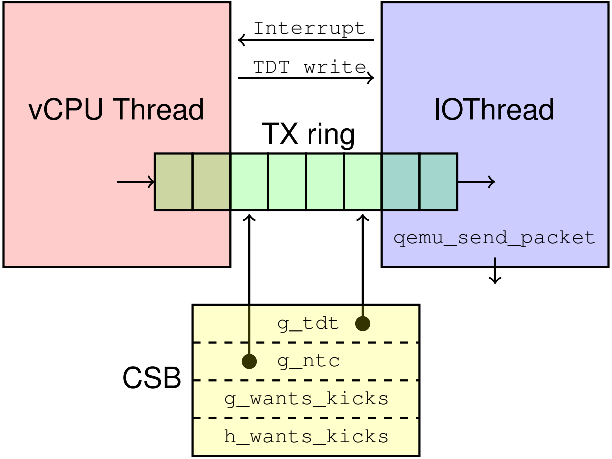
- updating status (available packets to send)
- notification (status has changed)

Paravirtualized TX emulation

Separate the two functions:

- status only updated in shared memory (*Communication Status Block*)
- TDT only used for notification
- TX processing in a separate thread

Paravirtualized TX path



Minimize consumer/producer notifications

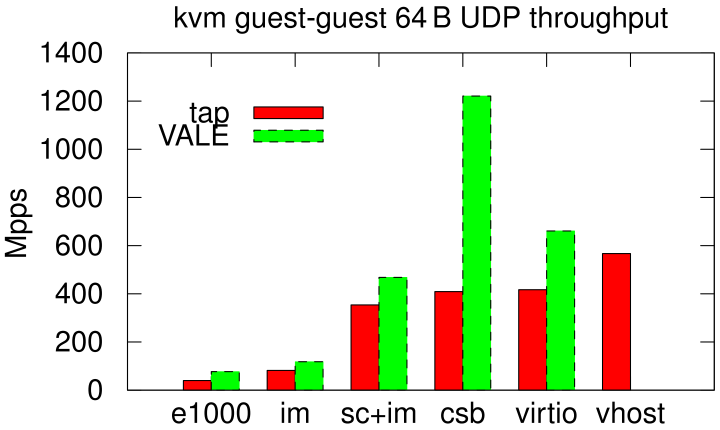
Producer notifications

- Schedule the IOThread (consumer)
- Notifications disabled while consumer runs

Consumer notifications

- Interrupt the guest (producer)
- Notification enabled when TX ring is full (Tx lazy completion)

Current performance [2]



Improving VALE batching

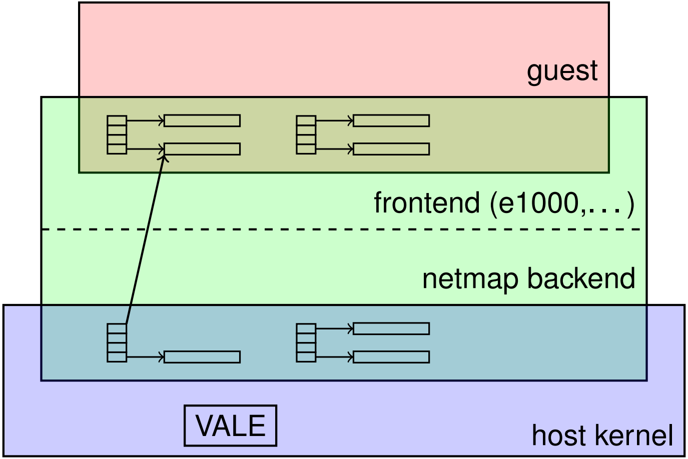
Problem

- frontends see batches of packets
- VALE backend may send batches of packets
- but the FE↔BE interface only allows one packet at a time

Implemented solution

- add flags to `qemu_send_packet`
- producer sets a “more packets coming soon” flag
- consumer can take informed batching decisions

Indirect buffers



Indirect buffers + VALE batching

Problem

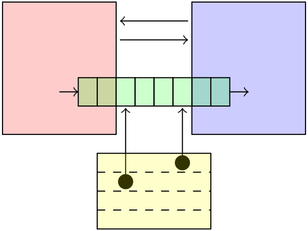
Frontend does not now when buffers are consumed

Implemented solution

Register a callback from BE to FE

Open problems (1/2)

Now common to both e1000-paravirt and virtio



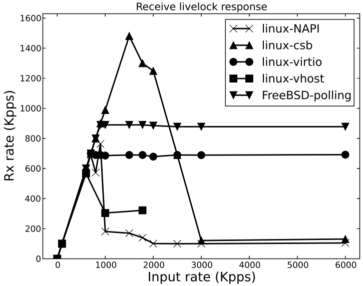
fast consumer

- a fast consumer may cause a high rate of kicks from the producer
- the producer is slowed down and throughput drops (even by half)
- unstable and counterintuitive measures

Open problems (2/2)

receive livelock

- when reaching ≈ 1 Mpps, receiver chokes
- in Linux guests, this invariably happens inside the kernel at the socket queue
- NAPI is still too aggressive w.r.t. the final consumer (i.e., user space)
- FreeBSD polling?



References

- L. Rizzo and G. Lettieri.
VALE: a switched ethernet for virtual machines.
In *Proc. ACM CoNEXT*, December 2012.

- L. Rizzo, G. Lettieri, and V. Maffione.
Speeding up packet I/O in virtual machines.
In *Proc. ACM/IEEE ANCS*, October 2013.