Kemari: Fault Tolerant VM Synchronization based on KVM

Yoshi Tamura

NTT Cyber Space Labs.
tamura.yoshiaki@lab.ntt.co.jp

Aug 10, 2010

What is Kemari?

蹴鞠

(Kemari)

What is Kemari?

蹴鞠

(Kemari)

- Kemari is a football game that players keep a ball in the air

What is Kemari?

蹴鞠

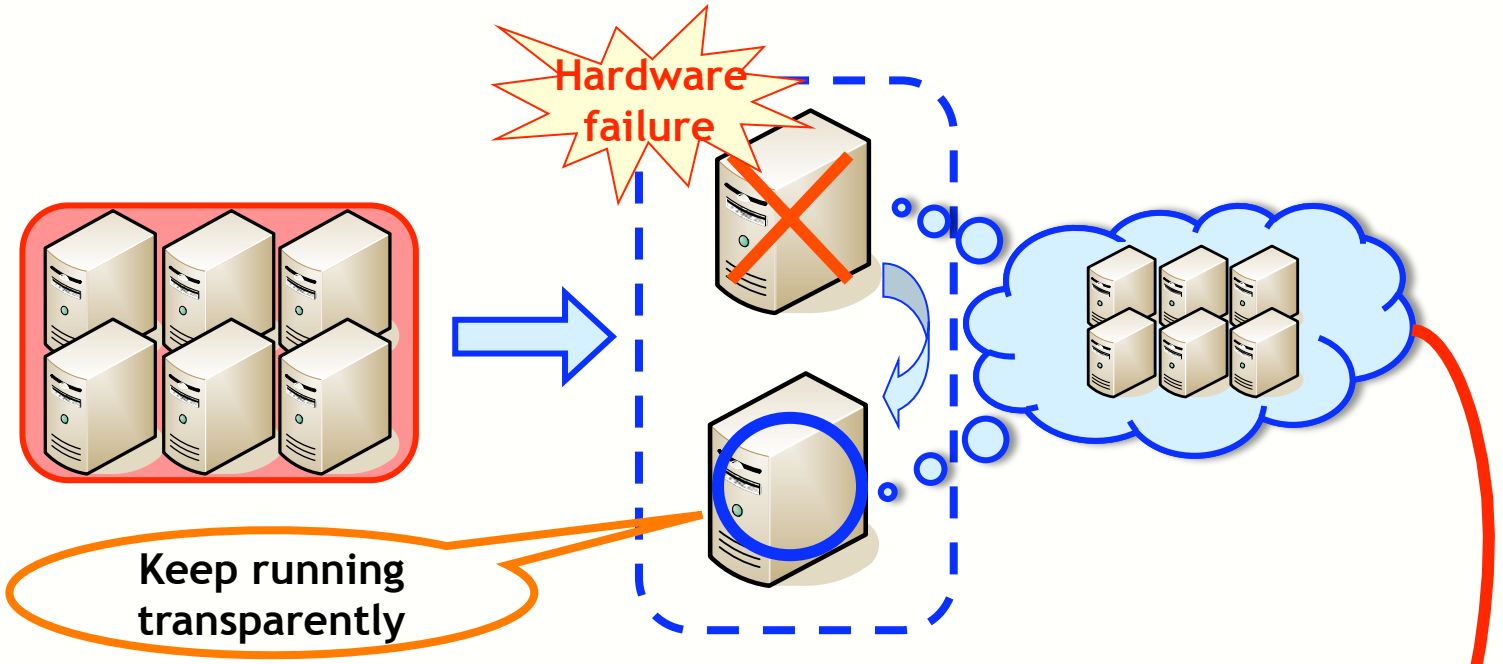
(Kemari)

- Kemari is a football game that players keep a ball in the air

Don’t drop the ball!

Our goal

Don’t drop the ball! Don’t drop the VMs!

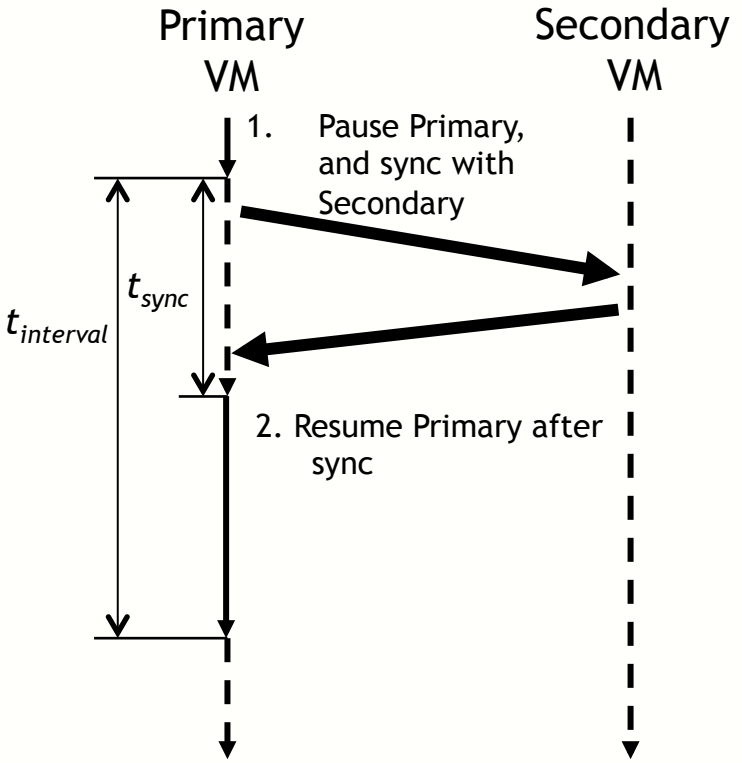


Kemari: Virtual Machine Synchronization

Use cases of Kemari

- Generality
  - Not all systems/applications are HA ready
  - Kemari protects w/o major changes to applications
- Cost efficiency
  - Although availability is important, not all people/ company can afford to buy FT systems, but downgrades to HA solutions
  - Kemari provides seamless availability with the cost of HA solutions

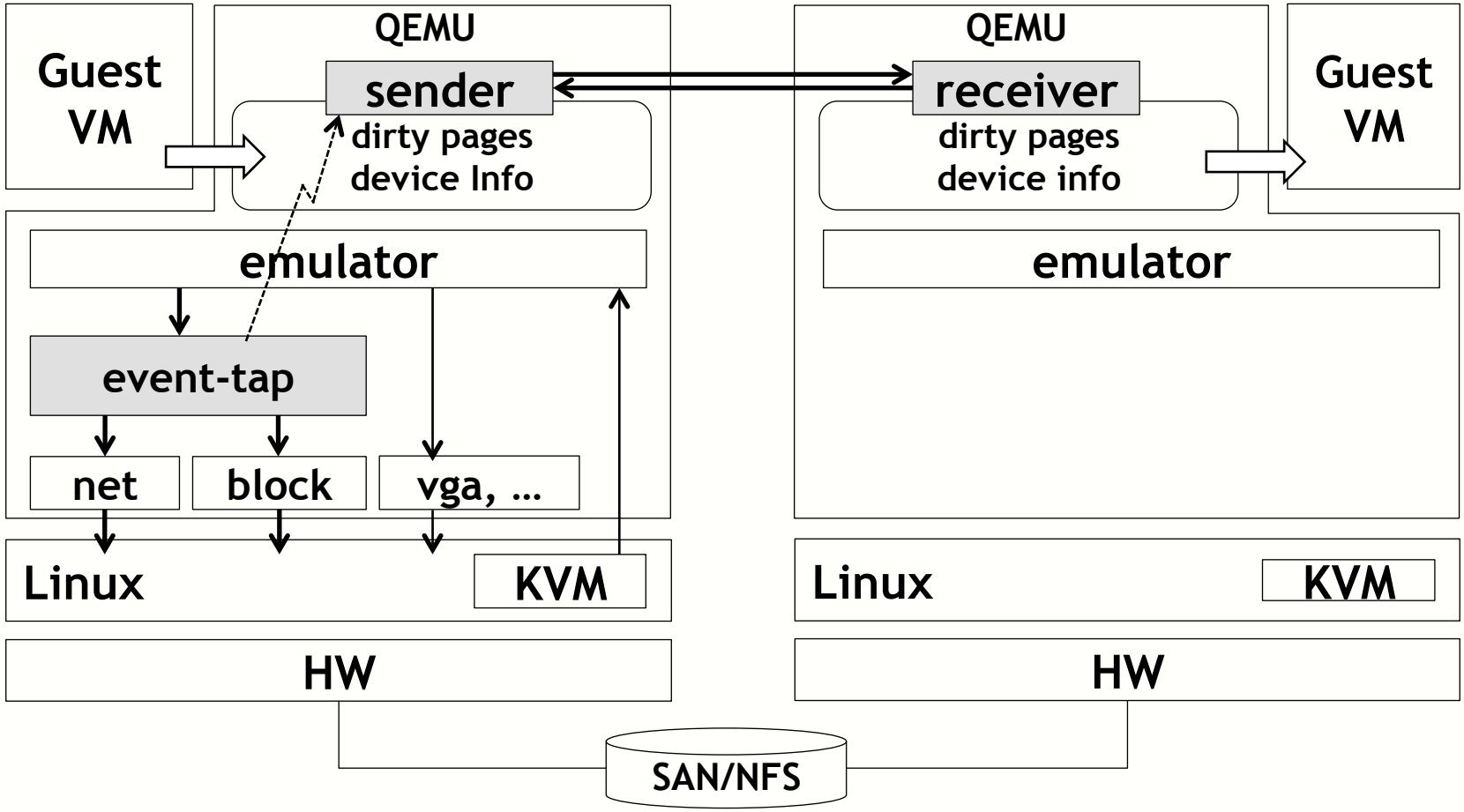
Event-driven VM synchronization



- Need to make the overhead of sync smaller
  - Make sync time shorter
    - ⇨ Only transfer updated data
  - Sync VMs less often
    - ⇨ Secondary must be able to continue transparently

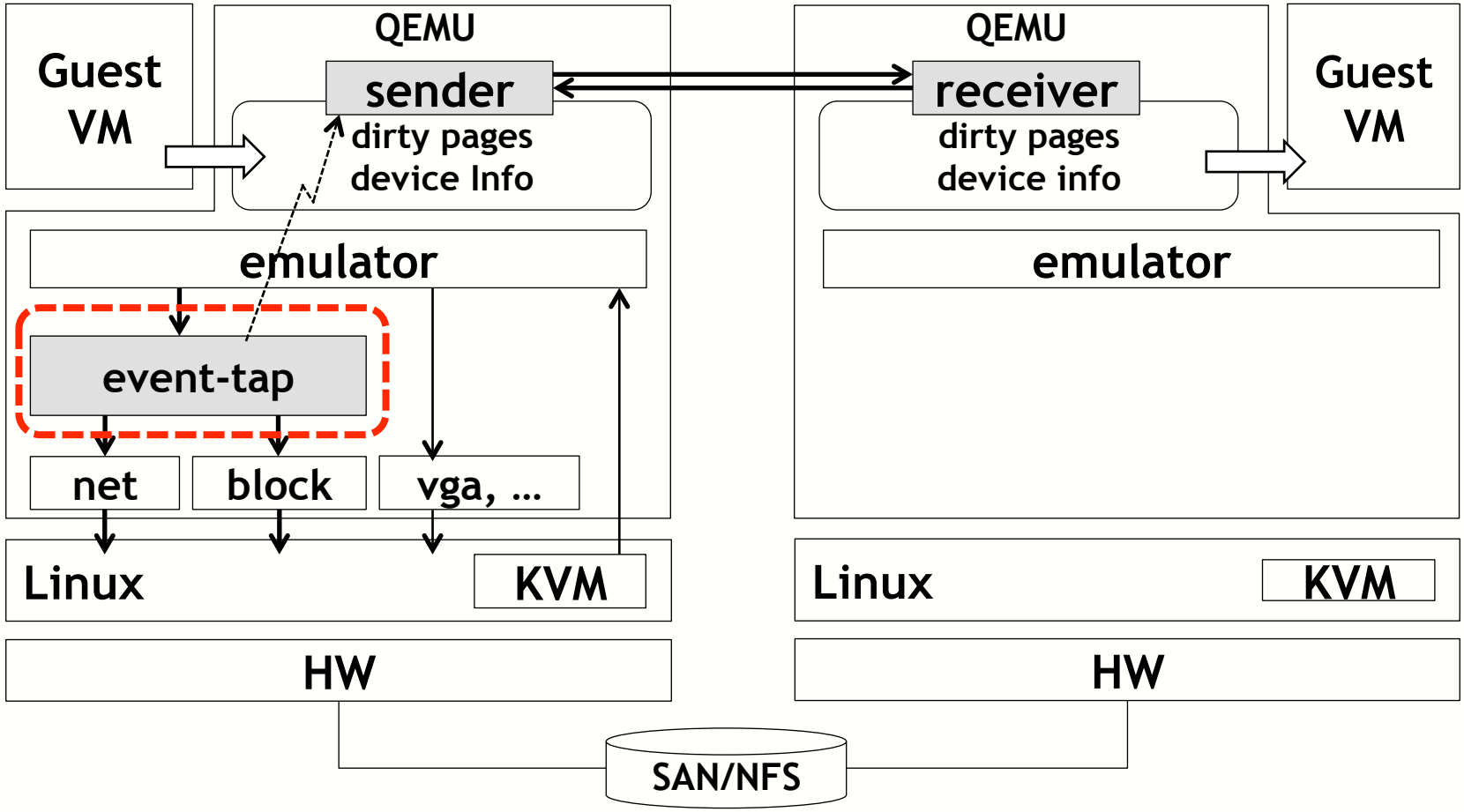
- Sync VMs before sending or receiving Events
  - Events: Storage, network

Architecture based on KVM/QEMU



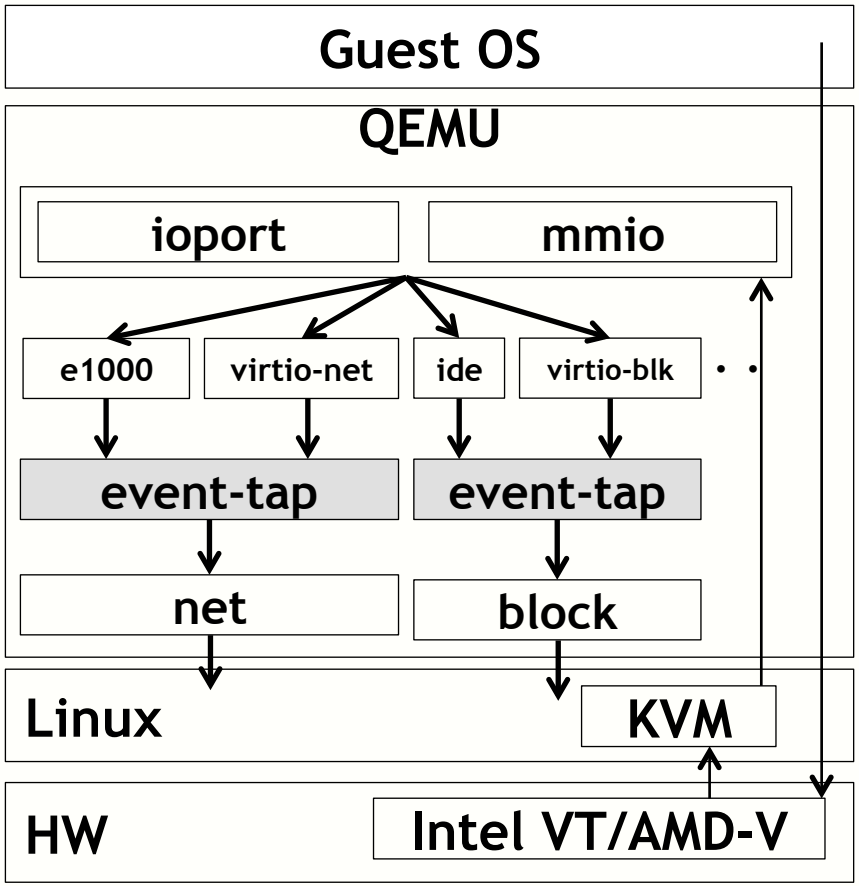
- event-tap: controls when to start VM sync
- ft-transaction: sender/receiver for VM transaction

Architecture based on KVM/QEMU



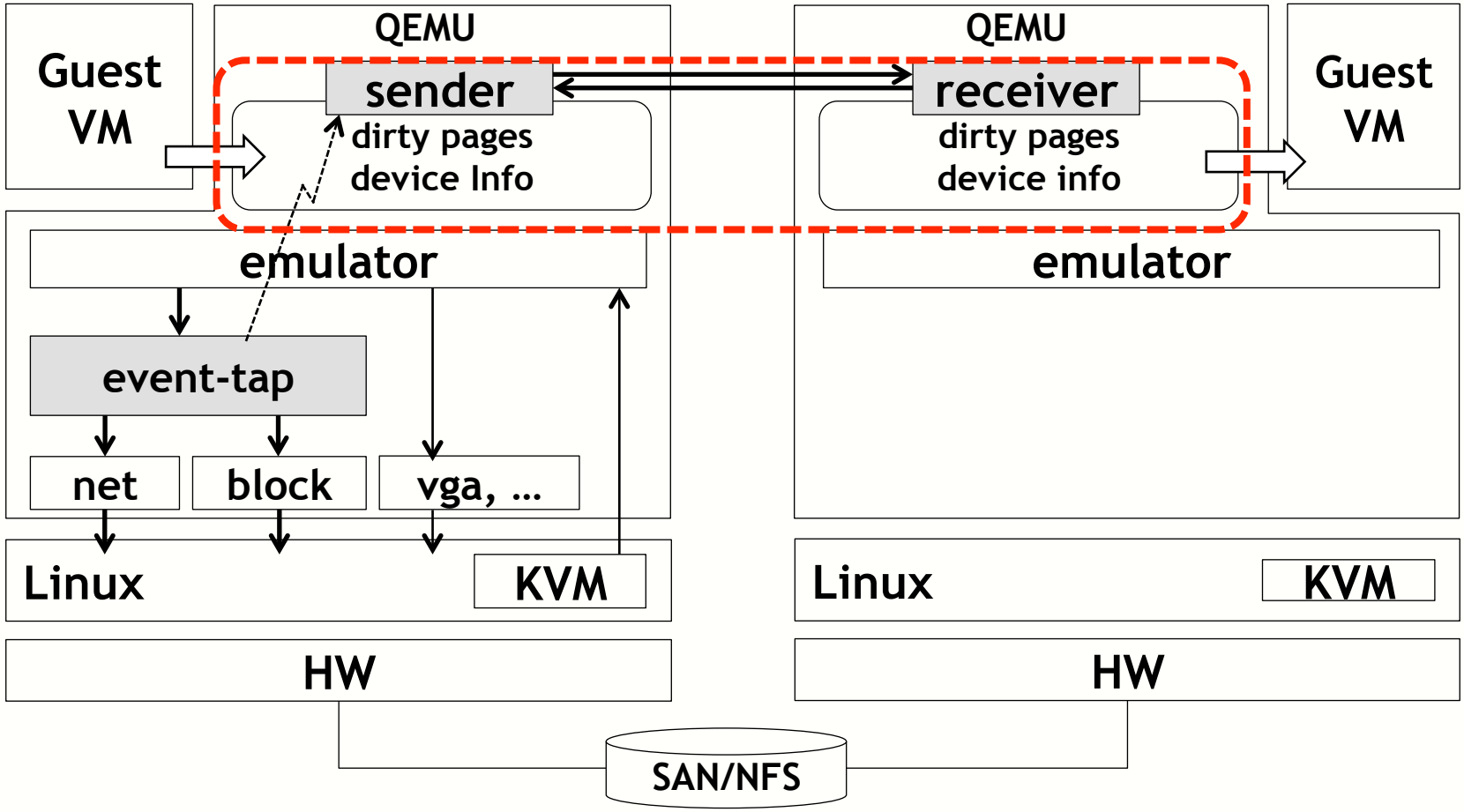
- event-tap: controls when to start VM sync
- ft-transaction: sender/receiver for VM transaction

event-tap: which and when to capture



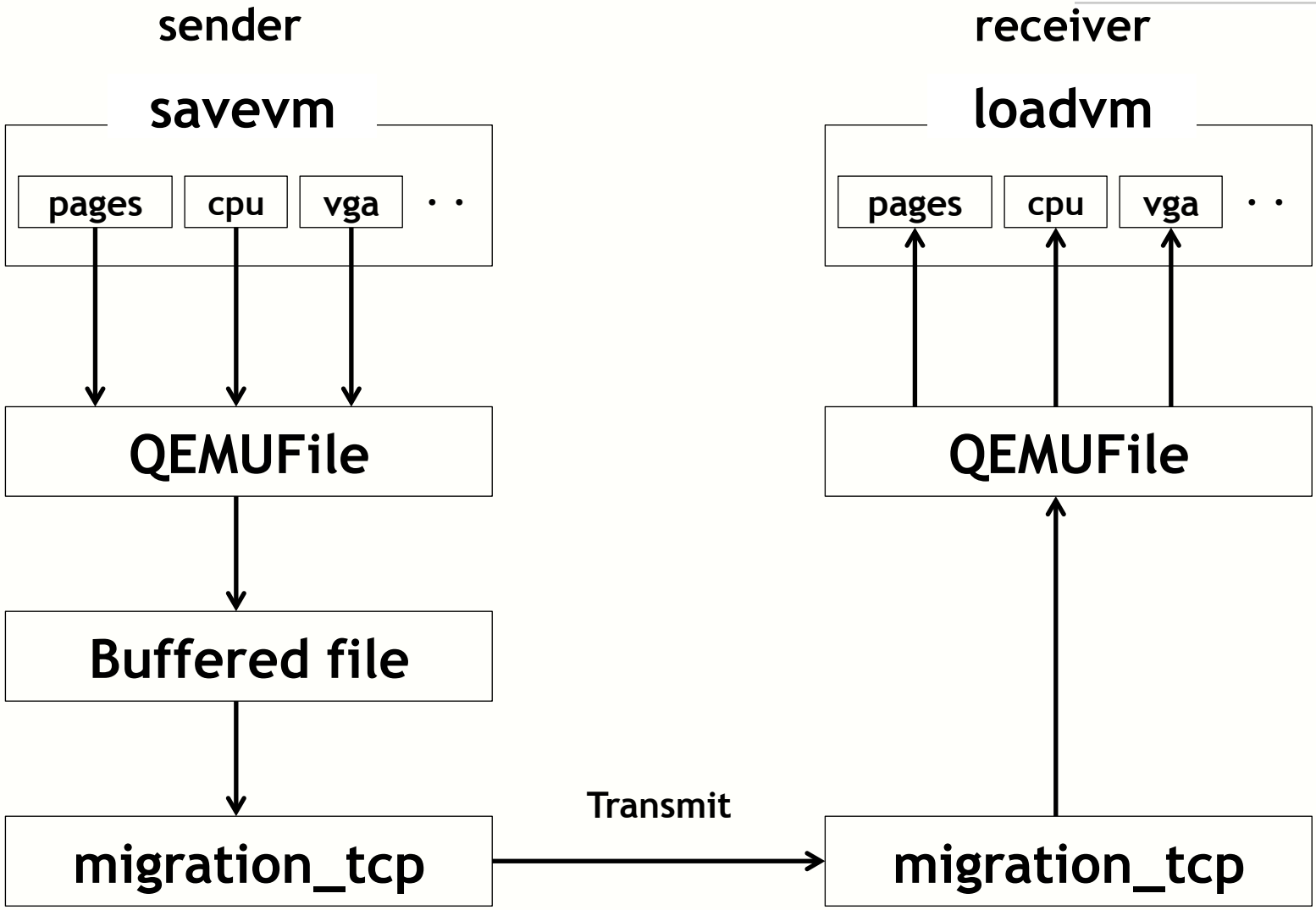
- Hooks at net/block layer in QEMU
  - Applicable to many device models
  - PV Drivers only in Xen
- Issues with I/O emulation
  - rip gets proceeded in KVM
  - events aren't replayed
- event-tap transfers events to the secondary
  - replayed on the secondary upon failover

Architecture based on KVM/QEMU

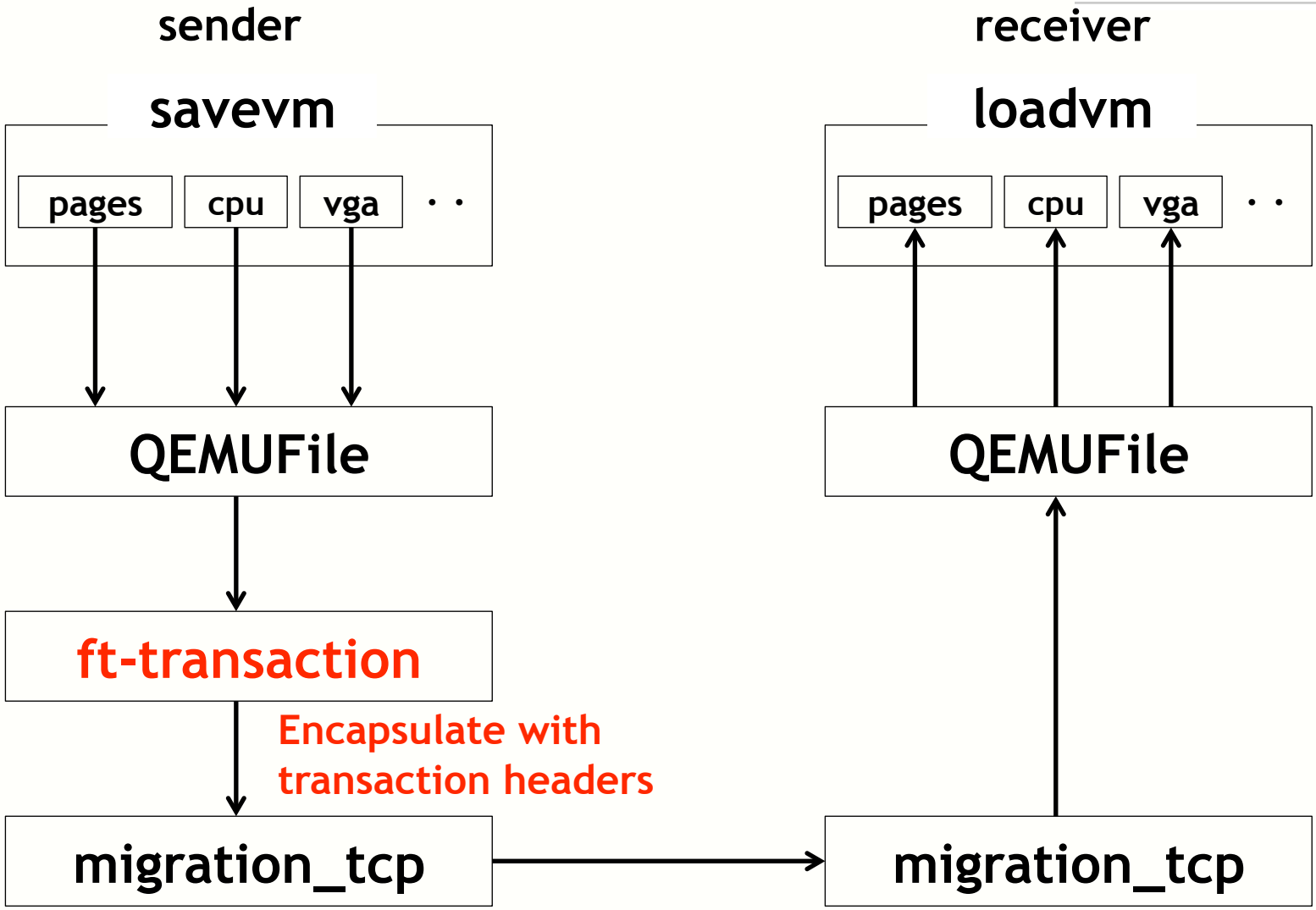


- event-tap: controls when to start VM sync
- ft-transaction: sender/receiver for VM transaction

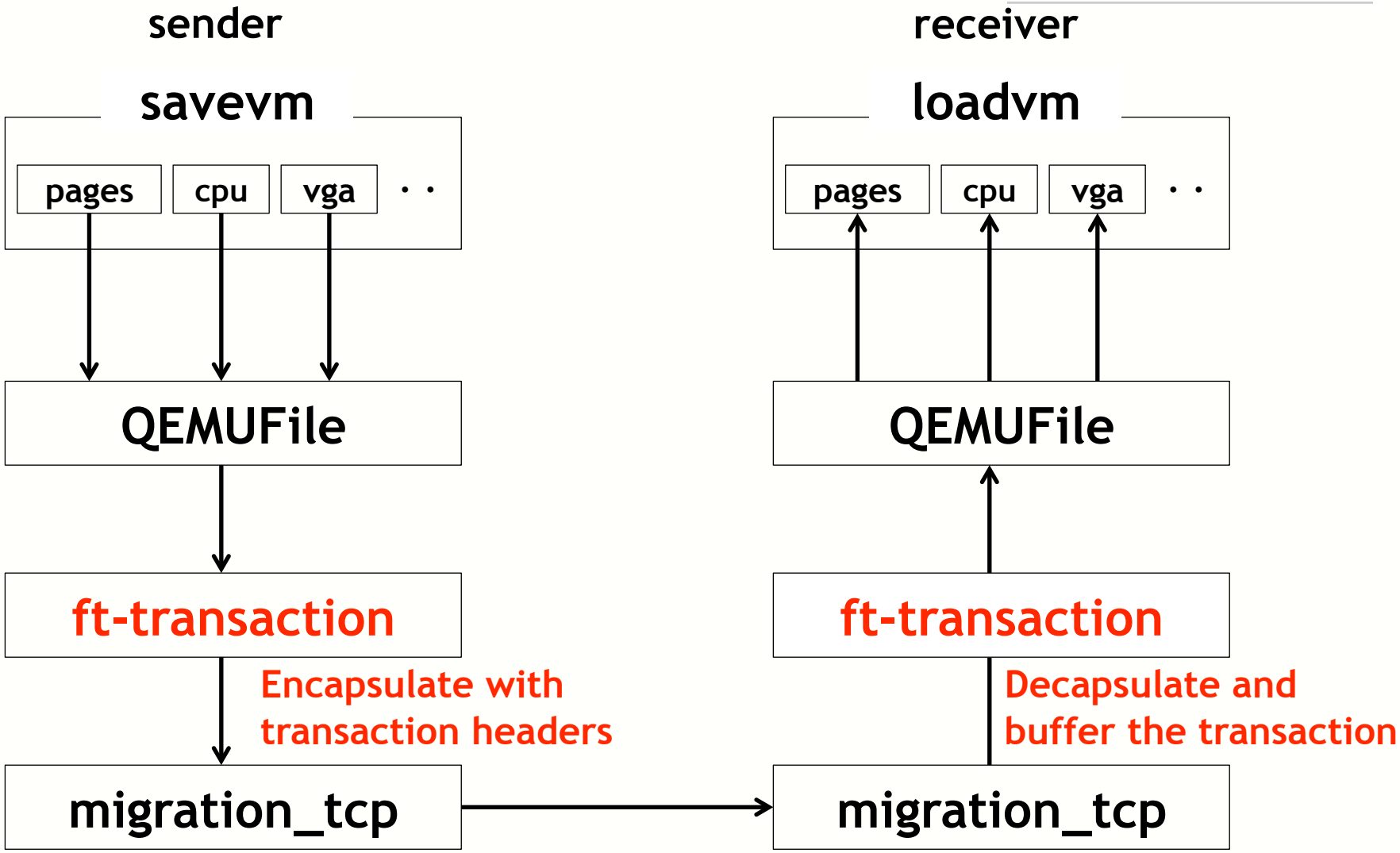
Extending LM for continuous VM transaction



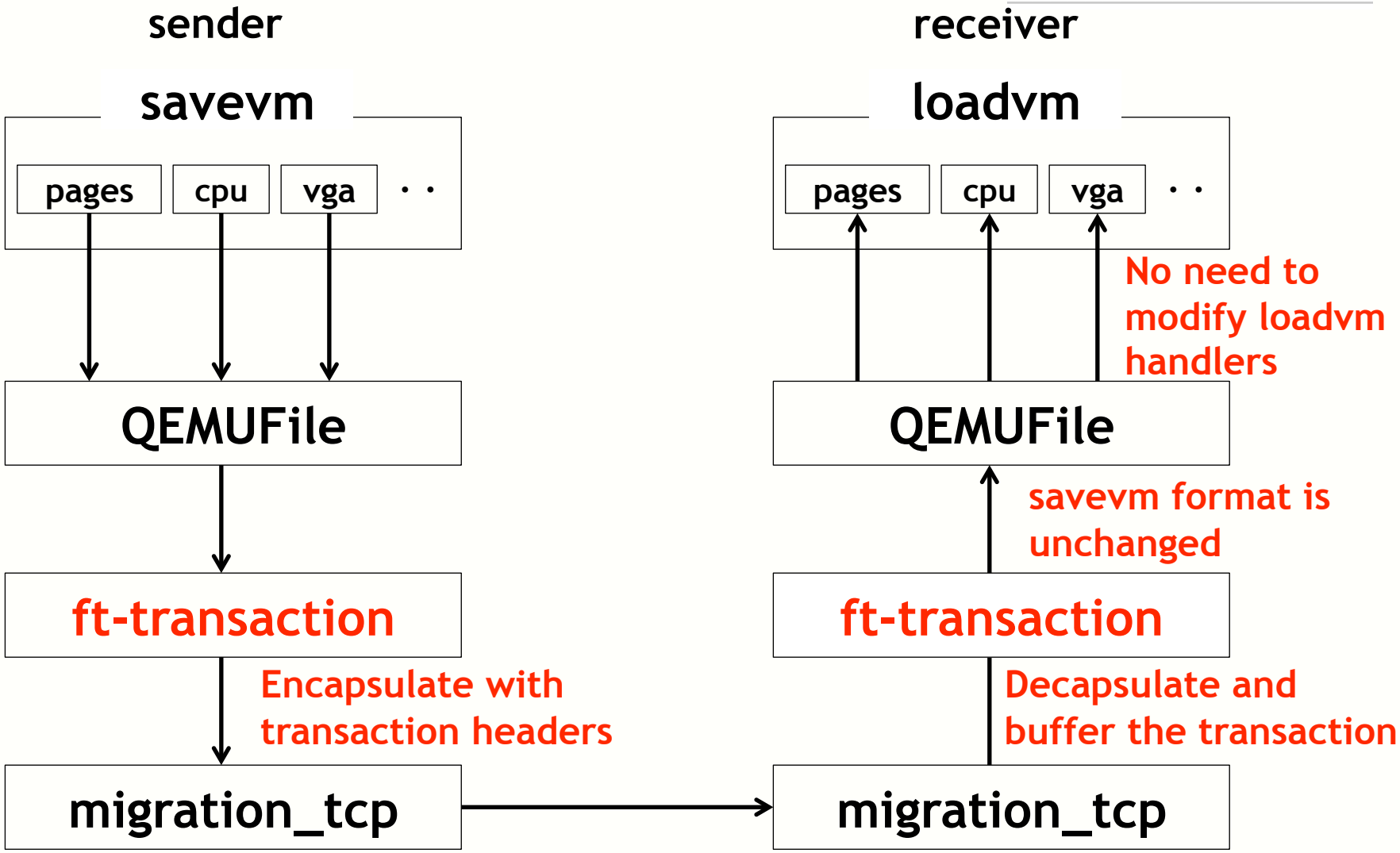
Extending LM for continuous VM transaction



Extending LM for continuous VM transaction



Extending LM for continuous VM transaction



Optimizations for Kemari

- Fast dirty bitmap travelling
  - Modify byte-based dirty bitmap in QEMU to bit-based
  - Boosts travelling up to 132x
- writev() and avoiding copies at QEMUFile buffer
  - Boosts 17% with InfiniBand (IPoIB)
  - RDMA migration may benefit potentially

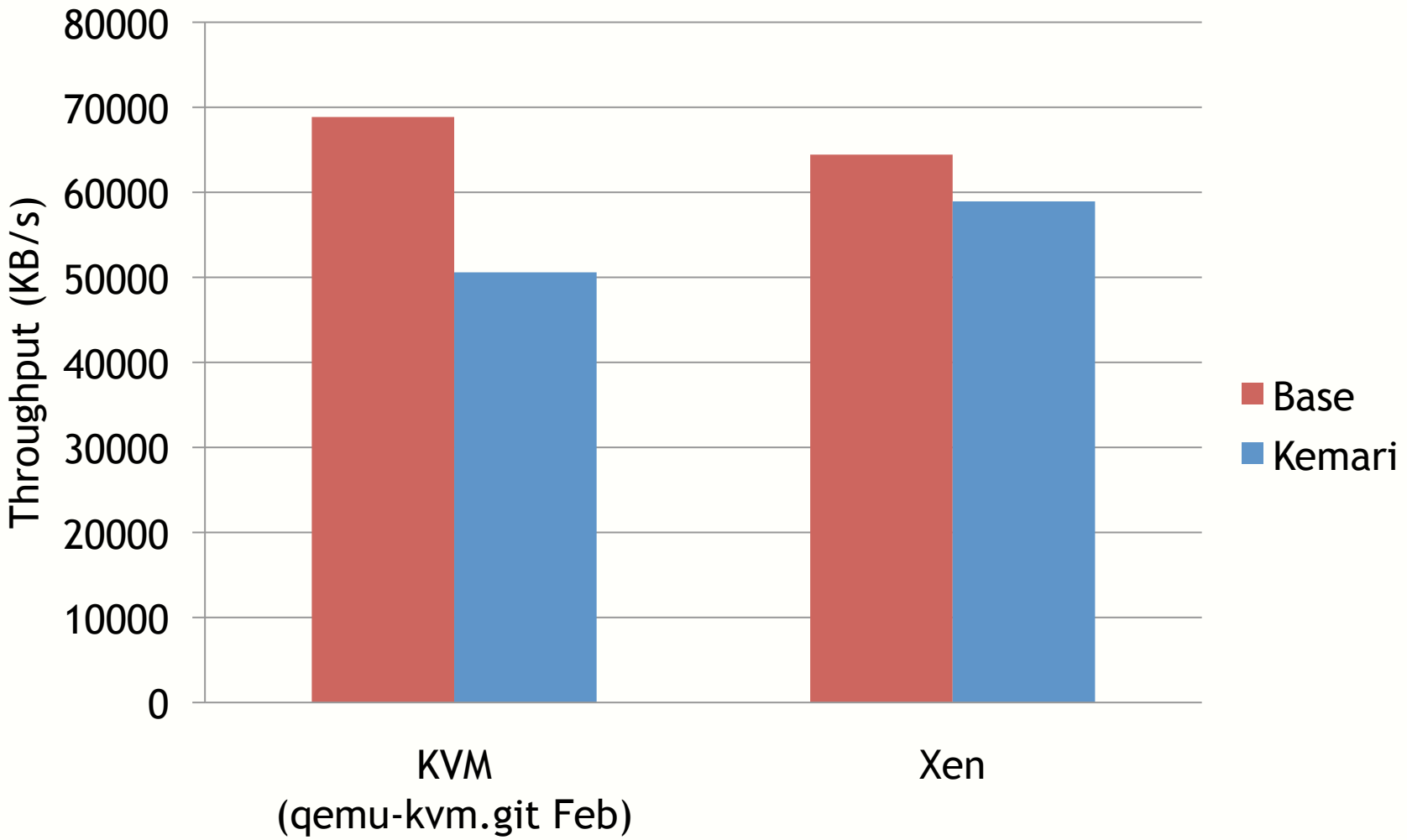
Current status

- Patches for qemu.git and qemu-kvm.git
  - Need to catch up the head!
- Manual failover only
  - Needs async/threaded migration for integrating with HA stack
- Performance?

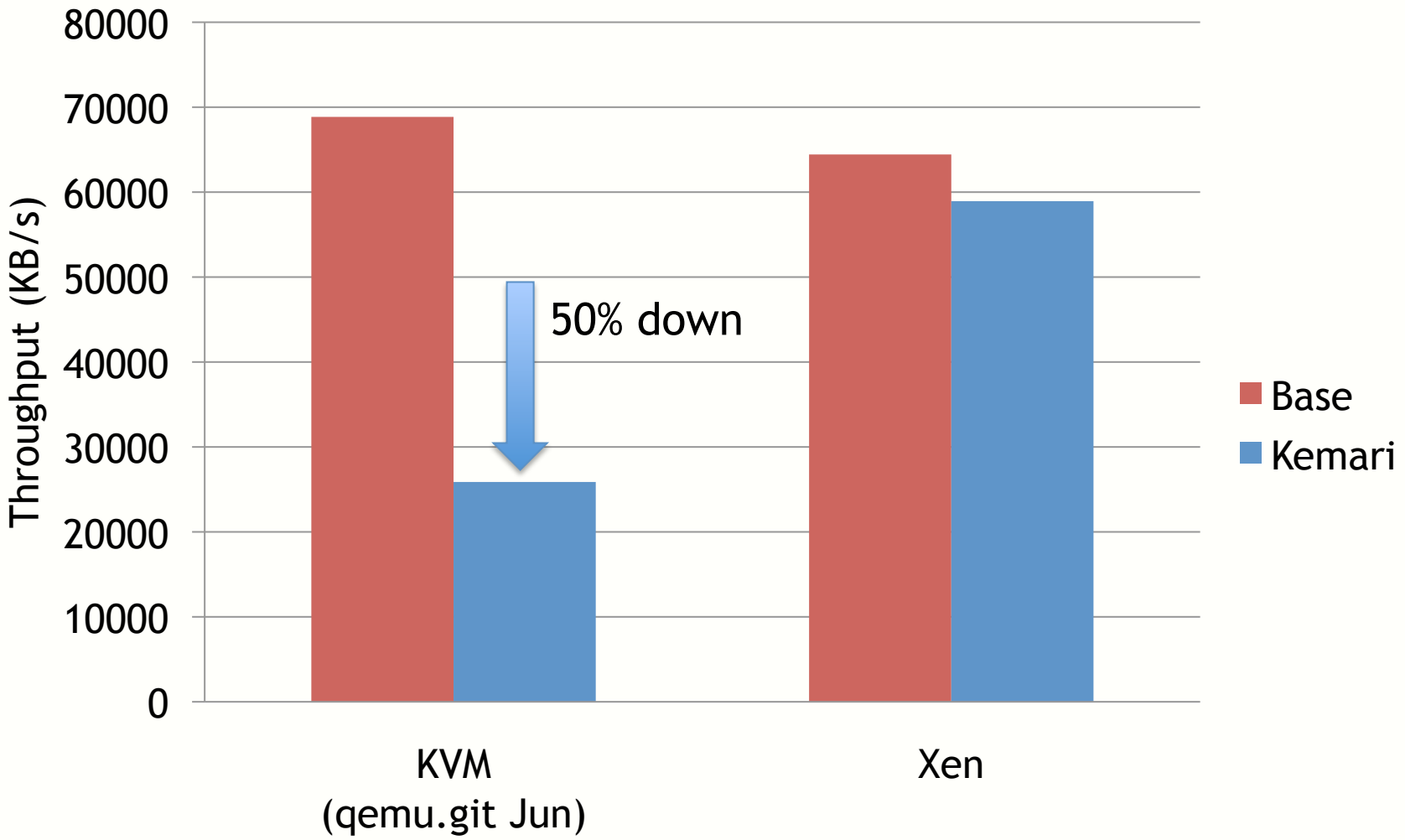
Experimentation

- Experimentation items
  - Performance of the Primary VM (File I/O) using iozone
- Test machines
  - Hardware spec
    - CPU: Quad-core Intel Xeon 2.6GHz X 2
    - Network: Gb Ethernet, Chelsio 10G
    - SAN: FC Disk Array
  - VM spec
    - KVM: Linux 2.6.33
    - Guest OS: Debian Etch w/ virtio-blk
    - Memory: 512MB

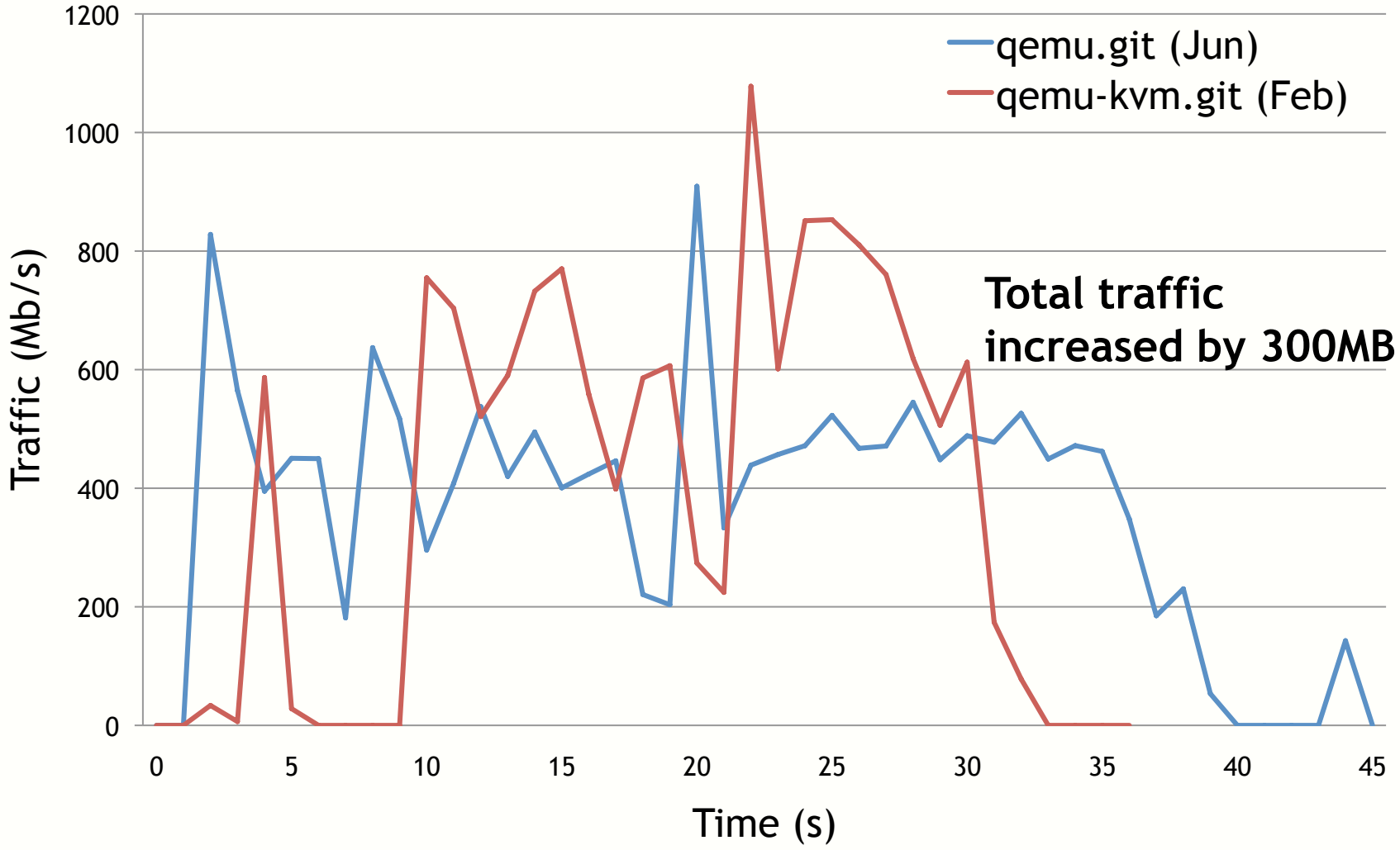
File I/O throughput



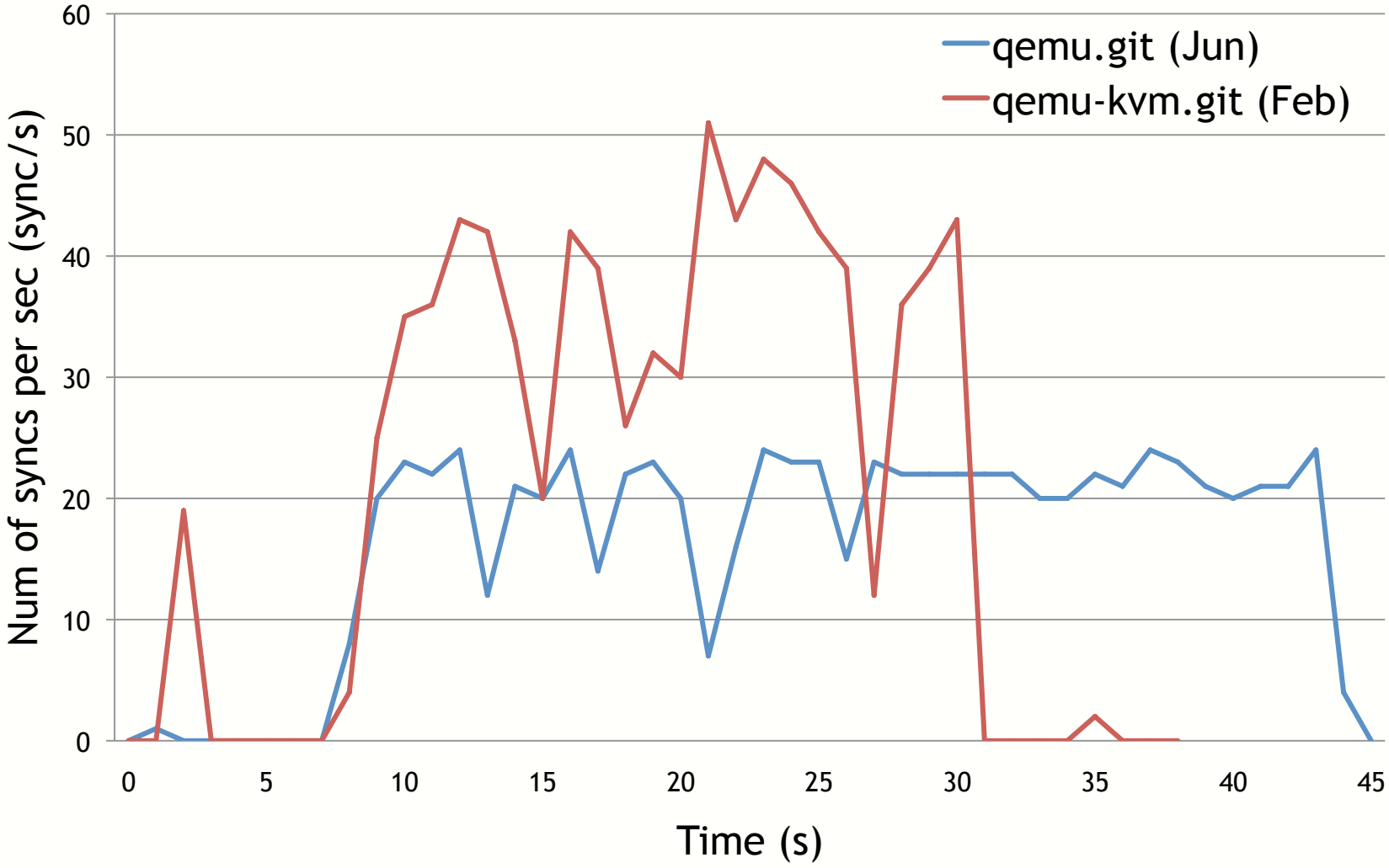
File I/O throughput :(



Traffic of the sync network (10G)



Num of syncs per second



TODO

- Posting patches for QEMU 0.14
  - Dec 2010?
- Integration with block migration
  - No need for SAN/NFS
- Async/threaded migration support
  - Avoid blocking on the receiver side
- Integration with existing HA stack

Summary

- Kemari provides fault tolerance to VMs with transparency, generality and simplicity
  - Applications can continue seamlessly
  - No modifications to applications
  - No specific hardware, just commodity PC
- Target on QEMU 0.14 (Dec 2010?)
  - Looking for reviewer
  - Advanced features welcome!
  - Bug reports, of course:-)
- http://kemari.sf.net