TECHNISCHE UNIVERSITÄT DRESDEN

SHRINKING THE HYPERVISOR ONE SUBSYSTEM AT A TIME
A Userspace Packet Switch for Virtual Machines

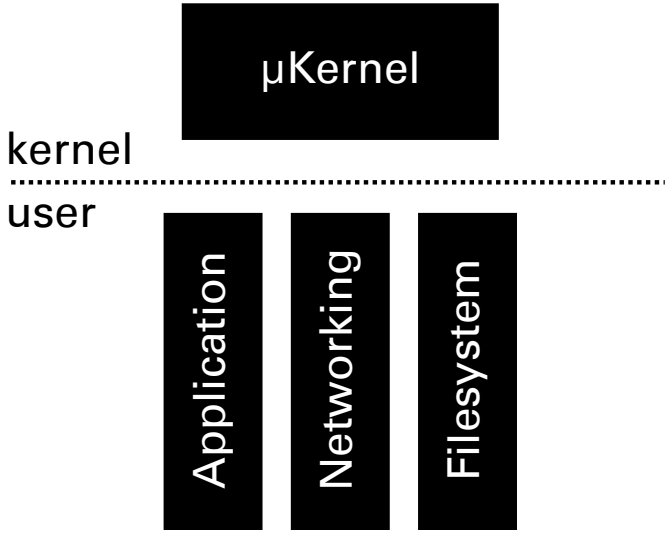
Julian Stecklina
OS Group, TU Dresden
jsteckli@os.inf.tu-dresden.de

VEE 2014, Salt Lake City

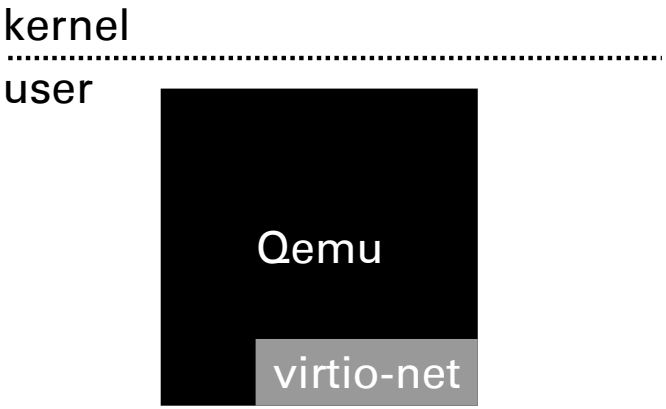
01 Microkernels and TCB

Systems built on microkernels usually structured as multiserver systems with strong isolation between subsystems.

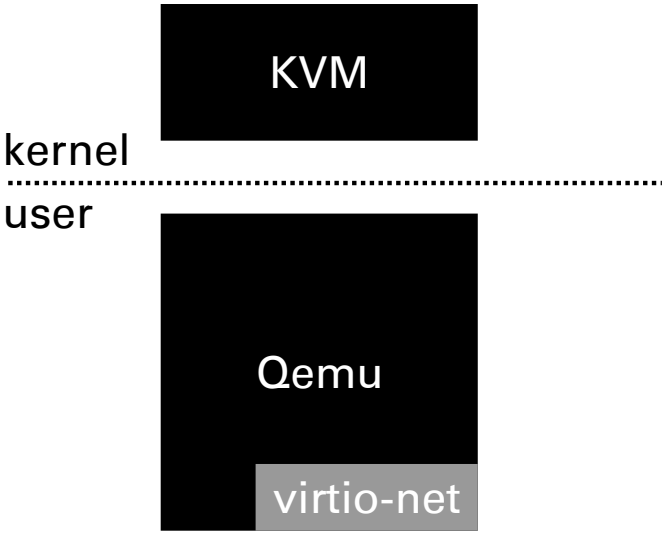
Applications only depend on subsystems they use.



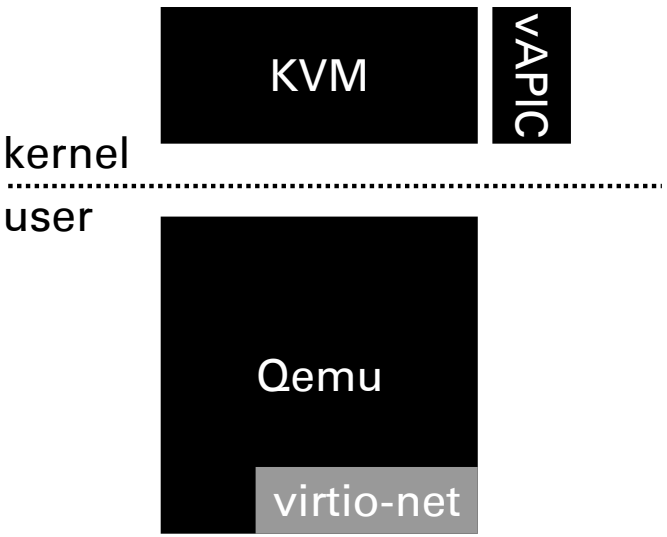
01 Towards Monolithic Hypervisors



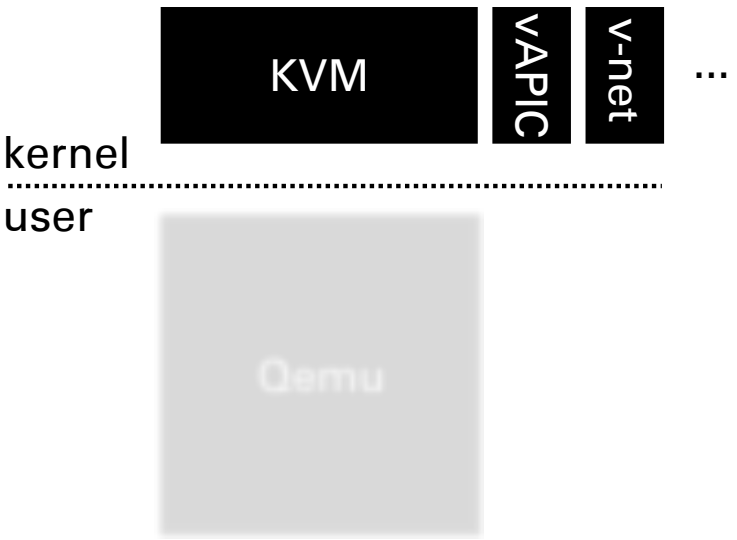
01 Towards Monolithic Hypervisors



01 Towards Monolithic Hypervisors



01 Towards Monolithic Hypervisors



01 User vs. Kernel

Attacks by malicious guest code are a serious concern.

Successful attacks on Qemu achieve **unprivileged code execution**.

Dangerous, but manageable.

Successful attacks on KVM achieve **code execution in kernel mode**.

Game over. ☜ You are here.

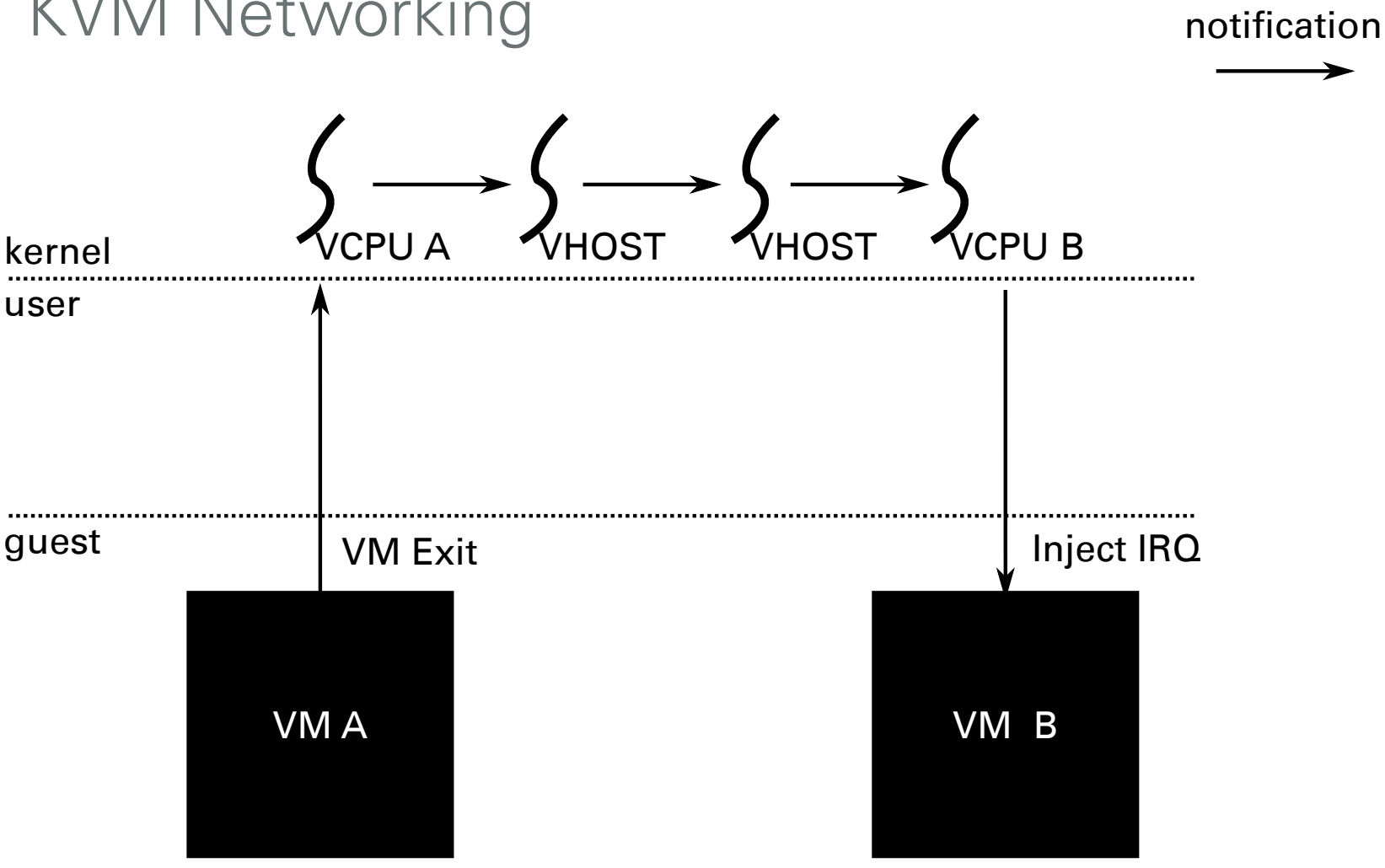
01 KVM/vhost Trusted Code

In a modern KVM installation the complete networking path is in the TCB of **all** applications on the host:

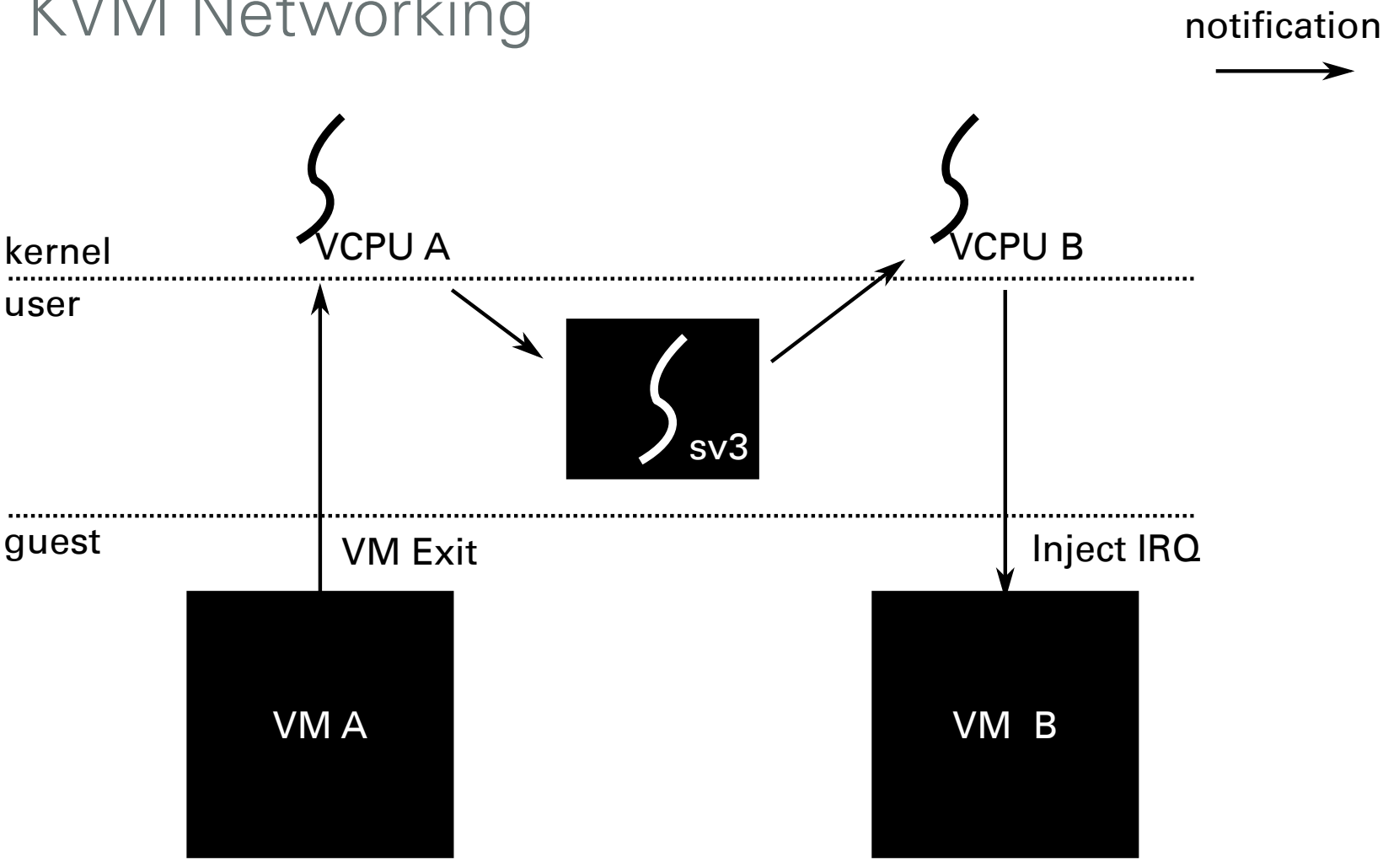
- (simple) instruction decoding,
- virtio-net device implementation,
- NIC driver.

Does it have to be?

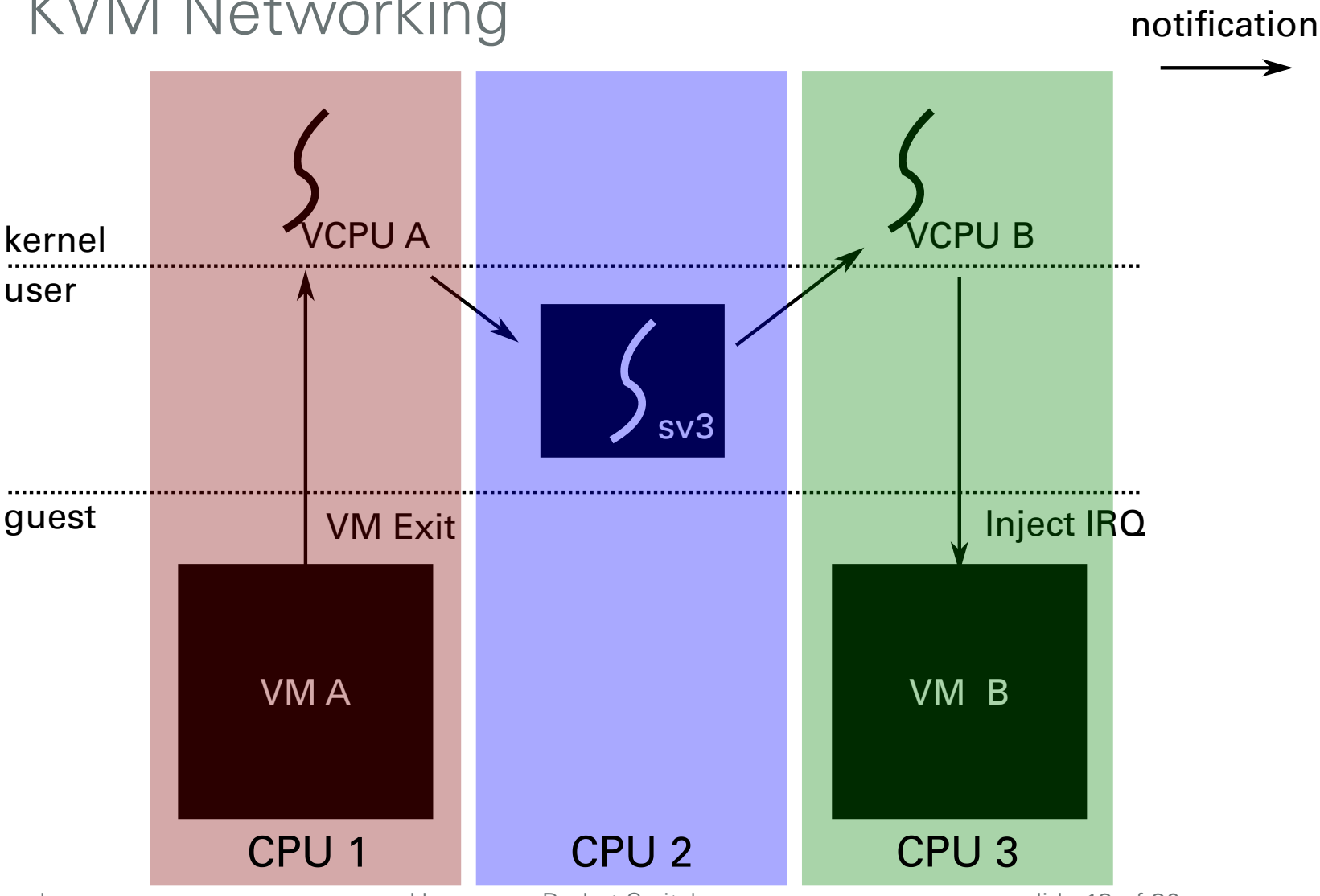
02 KVM Networking



02 KVM Networking



02 KVM Networking



02 A Userspace Switch: sv3

Userspace packet switch running as ordinary process on top of the host Linux:

- implements virtio-net,
- (no) packet memory management,
- NIC drver.

Every sv3 instance is a complete isolated networking subsystem.

02 Vhost and KVM

KVM and vhost are loosely tied together by Qemu using `eventfd`s. Qemu ties them together using ioctl.

- KVM can trigger `eventfd`s on VM Exits.
- `eventfd`s can be used to trigger IRQ injection in KVM.

Can use `eventfd`s from userspace as well without using vhost.

02 Tying sv3 into Qemu

Enhanced Qemu to support out-of-process PCI devices.

- Qemu connects to sv3 via `AF_LOCAL` socket.
- Qemu exchanges `fd`s to establish shared memory.
- Qemu exchanges `eventfd`s for
  - VM Exit notification,
  - IRQ injection.

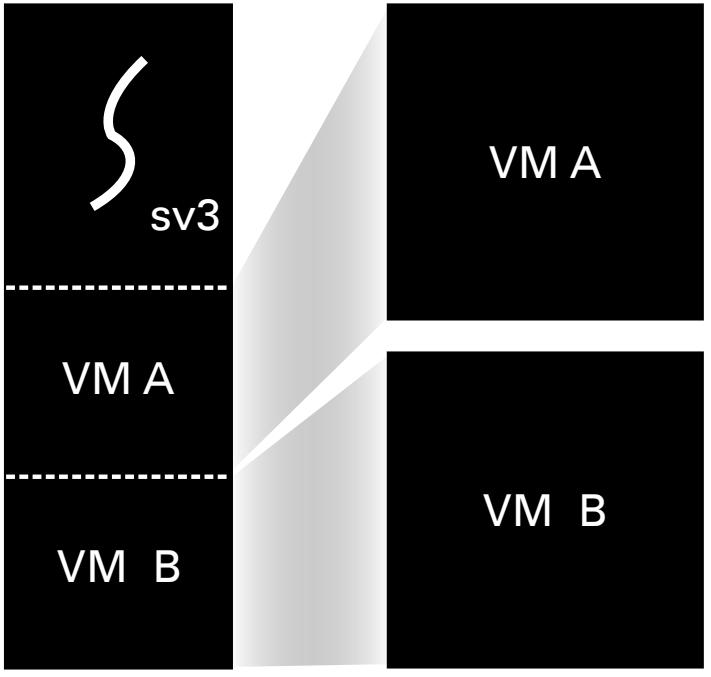
sv3 implements the complete virtio-net logic and it feels like L4!

02 Zero-Copy Packet Transmission

sv3 creates linear mappings of guest memory with `mmap`.

Packet data can be copied with a plain `memcpy`. No additional copies are necessary.

If no buffer space in the receiving VM is available, packet is dropped. No dynamic memory management for packets needed.



02 External Communication

Userspace driver for Intel X520 10 GBit NIC using VFIO (requires IOMMU) supporting static offloads:

- TCP Segmentation Offload
- Large Receive Offload
- Checksum Offload

Virtio descriptors translated to HW descriptors allows for zero-copy send with all offloads.

Better reuse existing drivers next time . . .

02 Switching Loop

sv3 is mostly single-threaded and lockless. Userspace RCU is used to synchronize adding and removing switch ports.

1. disable events on all virtio queues
2. disable HW IRQs
3. **poll for work until queues empty**
4. enable events/IRQs
5. poll a last time, if packet seen goto 1
6. block on `eventfd`

In overload scenarios, sv3 naturally operates in polling mode.

03 Resource Consumption

Processes are lightweight alternatives to driver VMs.

sv3	$<$ 2 MiB
NIC driver	+ 14 MiB
sv3 total	$<$ **16 MiB**

smallest VM	32 MiB [1]
netback VM	**128 MiB** [1]

Breaking Up is Hard to Do: Security and Functionality in a Commodity Hypervisor, Colp et al., SOSP '11

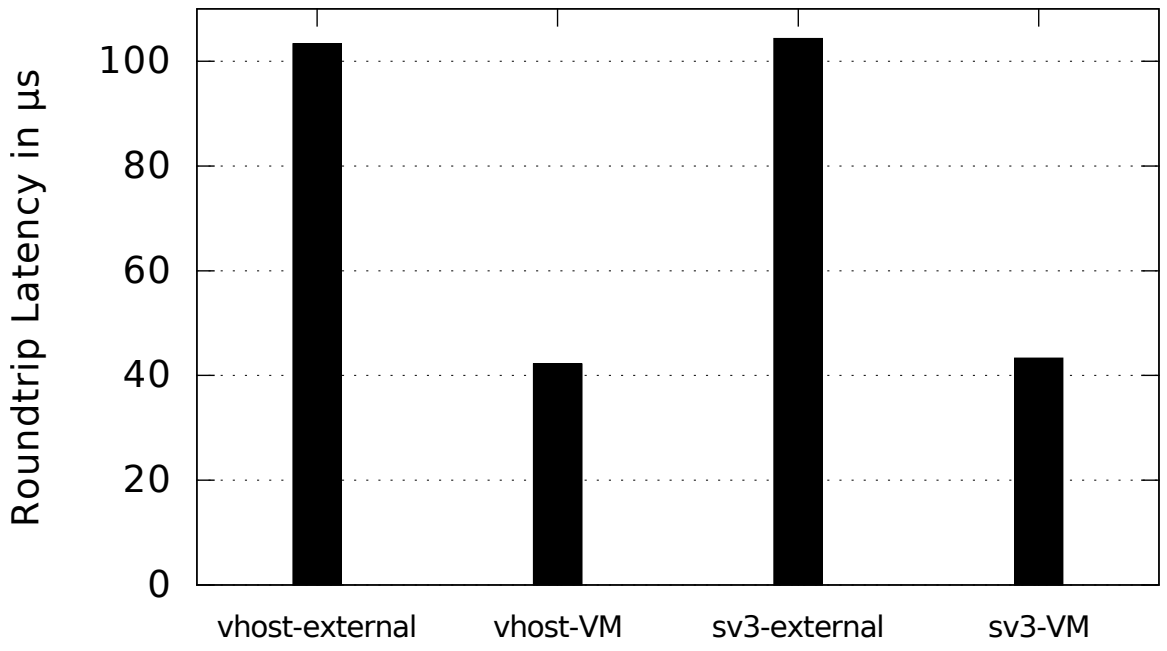
03 Evaluation System

- Intel Core i7 3770S (Ivy Bridge)
- C-states, HT and frequency scaling disabled
- 16 GiB RAM @ ˜159 Gbit/s
- Host: Fedora 19 with Linux 3.10 (vanilla)
- Guest: Linux 3.10 (vanilla), 256 MiB RAM
- Qemu 1.5 (plus patches)

03 Cost of Userspace Execution

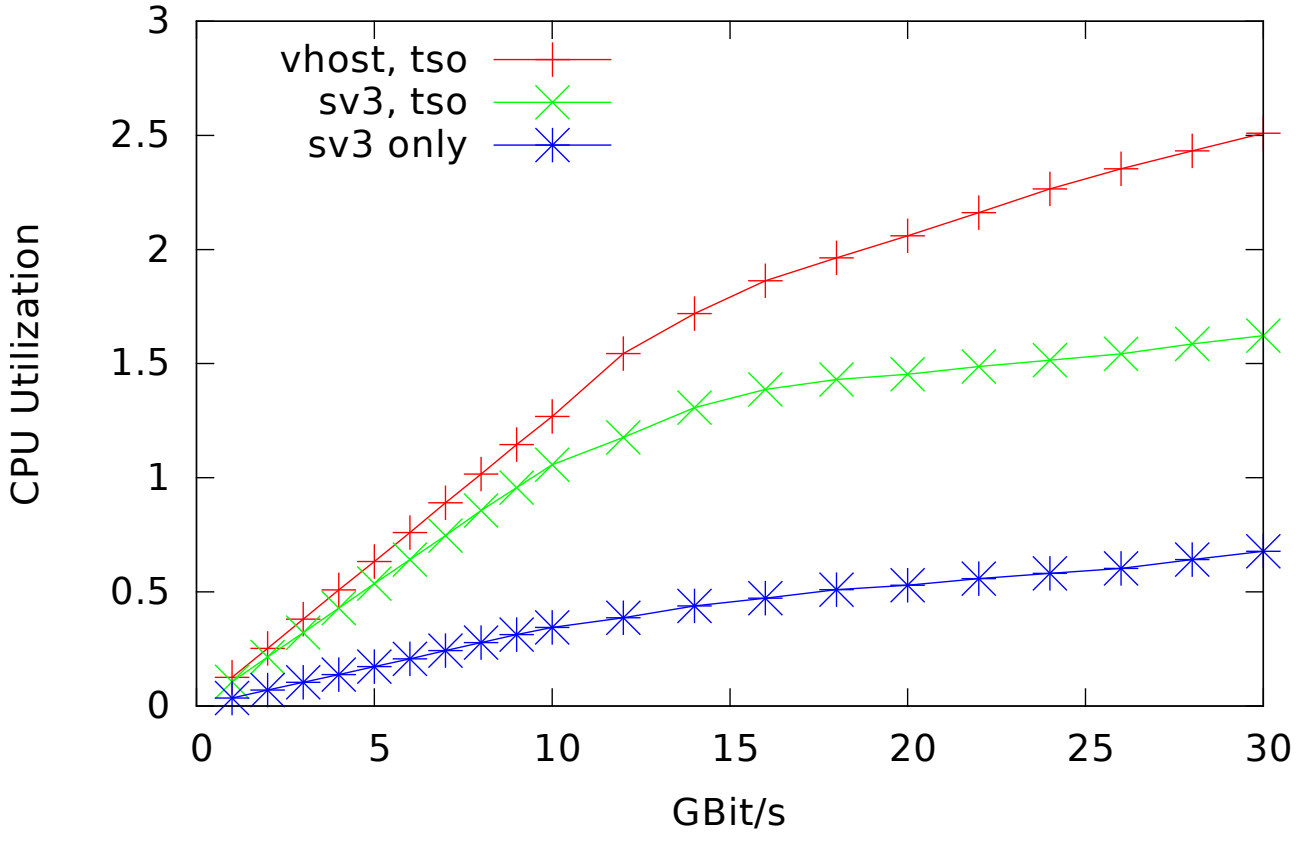
Notification to IRQ injection times for vhost vs. sv3 without any packet processing. Cost of additional trip to syscall layer and mode switch.

03 Latency

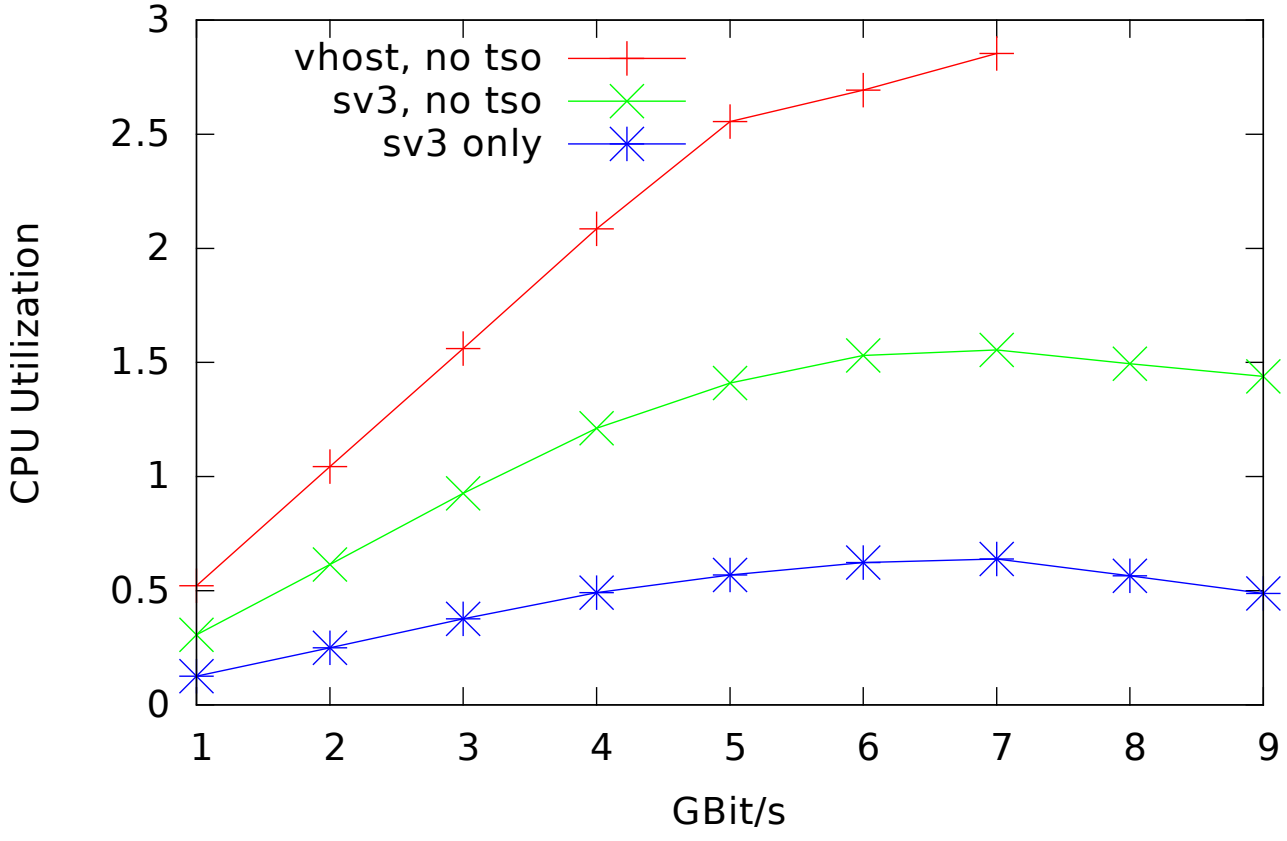


Latency measured using `netperf UDP_RR`.

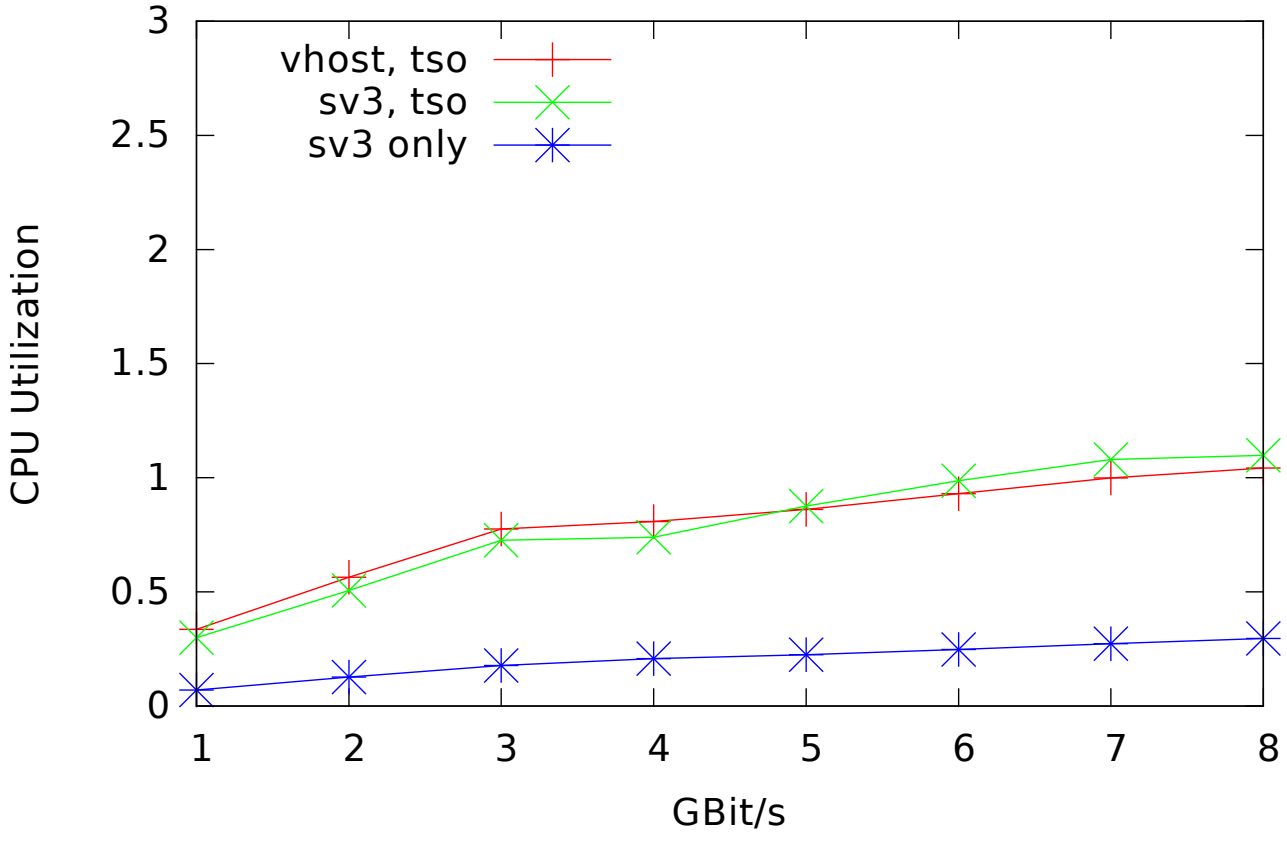
03 VM-to-VM Bandwidth



03 VM-to-VM Bandwidth (cont'd)



03 external-to-VM Bandwidth



04 Summary

`sv3` is an efficient lockless userspace packet switch for VMs running on (unmodified) Linux/KVM. Code: `https://github.com/blitz/sv3`.

Linux/KVM has all the mechanisms to make a microkernel-style design possible and efficient:

- rights transfer via `AF_LOCAL` sockets (capabilities)
- efficient notifications via `eventfd`s
- drivers in userspace via `VFIO` using `eventfd`s for IRQs
- tying `eventfd`s to VM exits / IRQ injection
- address space switch cost not a factor in performance

Few reasons to write new systems functionality in kernel mode.

Questions?

04 external-to-VM Bandwidth