



Vhost/Virtio: past year achievements and upcoming challenges

Maxime Coquelin – Jens Freimann

DPDK Userspace 2018

AGENDA

- Community updates
- Past year achievements
- Upcoming features

Community updates

Maintainers updates

Community updates

- Yuanhan Liu and Jianfeng Tan resigned as Virtio/Vhost maintainers
- Tiwei Bie and Zhihong Wang join me as maintainers

Subsystem statistics

Community updates

From v17.08 to v18.08, 283 patches from 46 unique contributors

Top 10 patch authors

- 79 Maxime Coquelin
- 21 Olivier Matz
- 21 Tiwei Bie
- 13 Zhiyong Yang
- 12 Jianfeng Tan
- 12 Stefan Hajnoczi
- 11 Fan Zhang
- 10 Tonghao Zhang
- 9 Bruce Richardson
- 9 Marvin Liu

Top 10 reviewers

- 111 Maxime Coquelin
- 81 Yuanhan Liu
- 32 Tiwei Bie
- 23 Jianfeng Tan
- 10 Jens Freimann
- 8 Ferruh Yigit
- 6 Thomas Monjalon
- 6 Andrew Rybchenko
- 5 Jay Zhou
- 5 Anatoly Burakov

Past year achievements

IOMMU support in vhost-user

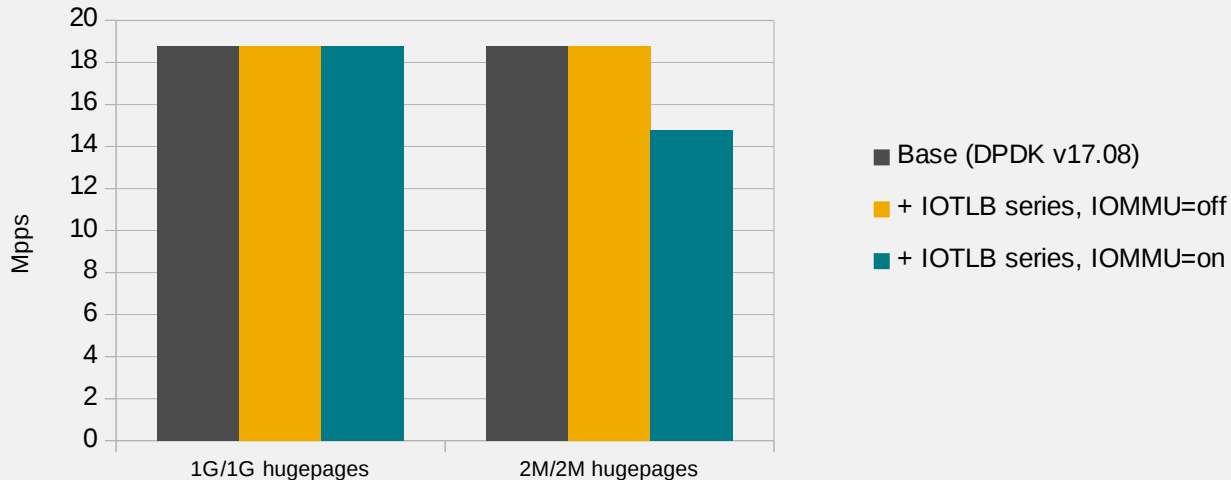
Past year achievements

- Author: Maxime Coquelin – Since DPDK v17.11
- Enables the use of vIOMMU with vhost-user backend
- Used to protect guest Kernel from malicious or buggy guest application using Virtio PMD
 - Without it the application can pass random GPA as descriptor buffer address
 - Which would result in vhost-user backend to overwrite guest memory with packet content, or leak guest memory as a packet
- Vhost-user backend implements an IOTLB cache to avoid querying Qemu every time it needs to perform a translation

IOMMU support in vhost-user (cont'd)

Past year achievements

- Overhead isn't significant with static mappings (DPDK's Virtio PMD)
 - Could be further improved with merging contiguous entries in the IOTLB cache



IOMMU support in vhost-user (cont'd)

Past year achievements

- Overhead with dynamic mappings (Kernel's virtio-net) is significant
 - Every packets buffers gets unmapped, resulting in a lot of IOTLB misses in the backend
 - No real security gain to use vIOMMU with Kernel driver
- Possible improvements
 - Add API for use with external backends?
 - Make IOTLB miss request blocking in enqueue path to avoid packet drops

Vhost-crypto backend

Past year achievements

- Author: Fan Zhang – Since DPDK v18.05
- Vhost-user crypto backend implementing virtio-crypto specification
- Translates virtio-crypto requests into DPDK crypto operations
- For now, supports AES-CBC-128 and HMAC-SHA1 ciphers
 - Cipher only and chaining modes supported, session-less mode to be added

CVE-2018-1059

Past year achievements

- Author: Maxime Coquelin – Since DPDK v18.05
- Malicious guest could make vhost-user backend to access out-of-bounds memory.
 - Could either cause denial of service of the host application (most likely), corrupt host memory or leak host memory (less likely)
 - Library only checked buffer start address was valid, not its entire range

CVE-2018-1059 (cont'd)

Past year achievements

- Fix consists in ensuring all buffers passed by the guest maps into guest memory and are also contiguous in the host application virtual address space.
 - Light performance impact measured in some cases
 - Required API change, even for v17.11 LTS (only used by external backends like SPDK)
- **Important to update, especially if guests aren't trusted**
- Would need to establish a formal process to handle CVEs

Vhost-net to Vhost-user live migration

Past year achievements

- Author: Jiayu Hu – Since DPDK v18.02
- Vhost-user didn't support some of the Virtio features supported by Vhost-net kernel backend
 - Live migration would fail if one of the missing feature had been negotiated
- Jiayu added support for missing features
 - Explicit Congestion Notification, UDP Fragmentation Offload, ...
 - Live-migration is now possible with recent DPDK
- Still, a tool to query backends features before migration is initiated would be needed to ensure successful migration

In-order processing

Past year achievements

- Author: Marvin Liu – Since v18.08
- New feature bit `VIRTIO_F_IN_ORDER`
- Process descriptors in ring order, wrap around
- Reduce accesses to ring, batch notification, simpler device and driver implementation, optimization is easy
- Vhost → virtio: 26.0 mpps vs 18.9 mpps (4 queues)
- Virtio → vhost: 11.5 mpps vs 10.5 mpps (4 queues)
- Loopback: 10.8 mpps vs 7.7 mpps

Vhost data-path acceleration

Past year achievements

- Author: Zhihong Wang – Since v18.05
- Enable virtio-ring compatible HW devices to serve virtio driver directly, offload data-path to HW
- Difference to PCI-passthru: only data-path is pass-throughed
- Control path: device specific through vhost(-user)
- A vDPA driver for Intel FPGA 100G VF (IFCVF) was also added

Upcoming features

Packed virtqueue layout support

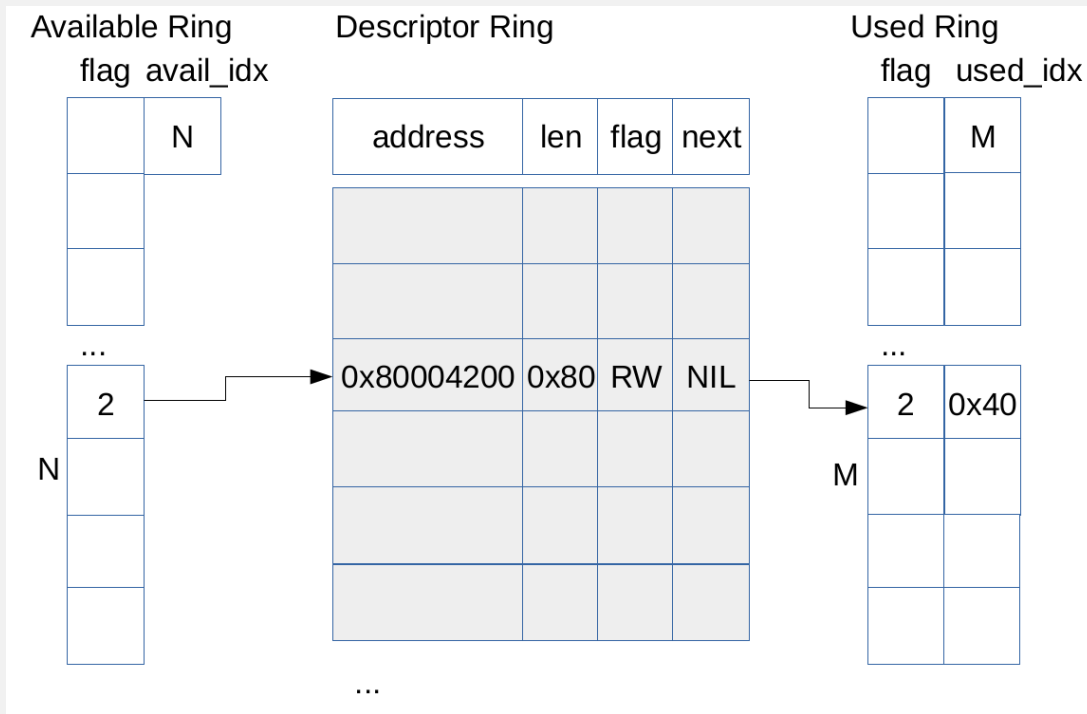
Upcoming features

- vhost part in 18.08 (Maxime)
- Support for virtio-pmd aiming for 18.11 (Jens)
- Motivation
 - Make it easier to implement virtio in hardware
 - Simplify ring layout, less cache-misses

Packed virtqueue layout support

Upcoming features

- Split virtqueues
 - Located in shared memory
 - Host and guest using shared memory to pass messages
 - Possibly on different CPUs
 - Causing cache synchronization



Packed virtqueue layout support

Upcoming features

- Reducing the overhead
 - Information is spread across too many data structures
 - Tighter packing will save cache misses
 - How about packing everything in a single data structure?

Packed virtqueue layout support

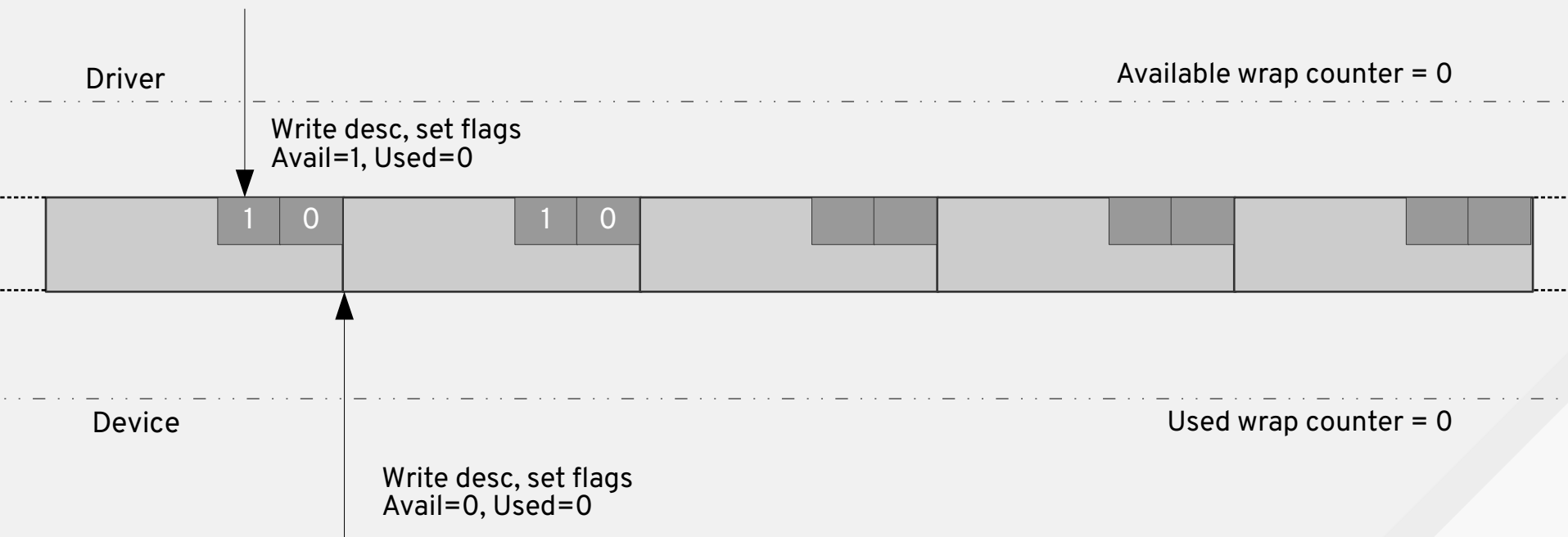
Upcoming features

- Descriptor ring
 - Driver writes out available descriptors in a ring
 - Device writes out used descriptors in the same ring
 - Descriptor: addr, len, id, avail, used
 - To mark a descriptor available, flip the avail bit
 - To mark a descriptor as used, flip the used bit

Packed virtqueue layout support

Upcoming features

- Descriptor state



Packed virtqueue layout support

Upcoming features

- Descriptor state

Ring wraps around

Driver

Available wrap counter = 1

Write desc, set flags
Avail=0, Used=1



Device

Used wrap counter = 1

Write desc, set flags
Avail=1, Used=1

Non-pow2 rings

Upcoming features

- Cache utilization with large rings
- Ring with 1K entries has size of 16K bytes
- If you have 32K 8-way associative cache, that's 4 of 8 ways
- Net device with two rx and tx queues → all other data is pushed out of cache
- Idea: make ring a bit smaller (e.g. 768) to reserve some place for data

Postcopy live migration

Upcoming features

- Author: Maxime Coquelin (Based on Dr. David Gilbert work) – Target: v18.11
- Precopy live migration
 - While pages are copied to the destination, the source is still running
 - If a migrated page is modified, it is tagged as dirty and copied again
 - At some point, the source halts and remaining pages are copied to destination
 - Once all pages migrated, destination VM resumes

Postcopy live migration (cont'd)

Upcoming features

- Postcopy live migration
 - Source sends minimal information about the VM execution state to the destination
 - VM is halted in source and resumed in destination
 - Source start migrating pages while VM is running in destination
 - If VM accesses a page that hasn't been migrated yet, it gets paused and requests source with missing page
 - On missing page reception, the page gets mapped in VM address space and VM is resumed

Postcopy live migration (cont'd)

Upcoming features

- QEMU/KVM implementation
 - Relies on userfaultfd to handle the page faults
 - QEMU registers guest memory regions to userfaultfd and dedicates a thread to handle the page faults
 - On page fault, requests source QEMU process for the missing page, remaps it and notify userfaultfd the fault has been handled
- Vhost-user backends support
 - Userfaultfd supports multi-process, i.e. process A can handle faults registered by process B
 - The vhost-user backend registers memory regions with userfaultfd, and send resulting fds to QEMU

Postcopy live migration (cont'd)

Upcoming features

- Performance
 - Early benchmark tends to show a significant improvement in total migration time
 - Real use-cases not yet measured though
- Open issues
 - Migration will break if application calls `mlockall()`, but vhost-user lib cannot prevent that
 - Some instabilities seen which prevent proper benchmarking & testing

API rework for external backends

Upcoming features

- Author: Dariusz Stojaczyk
- Current API isn't compliant with specification
 - Queues initialization require workarounds with some backends
- New API needed for external backends to handle backend specific messages
 - Or to handle generic messages in a backend specific way
- No consensus yet on whether extend current library or start from a new one and migrate -net backend afterwards

Conclusion

Conclusion

- Past year: new maintainers, vDPA, vIOMMU, packed virtqueues
- Highlights for next year: Post-copy live migration, packed virtqueues, more 1.1 features, improve HW offload support

- Join our monthly virtio meeting
 - E-mail jfreimann@redhat.com
- Start today and join our Virtio BoF later today



Q & A

maxime.coquelin@redhat.com
jfreimann@redhat.com

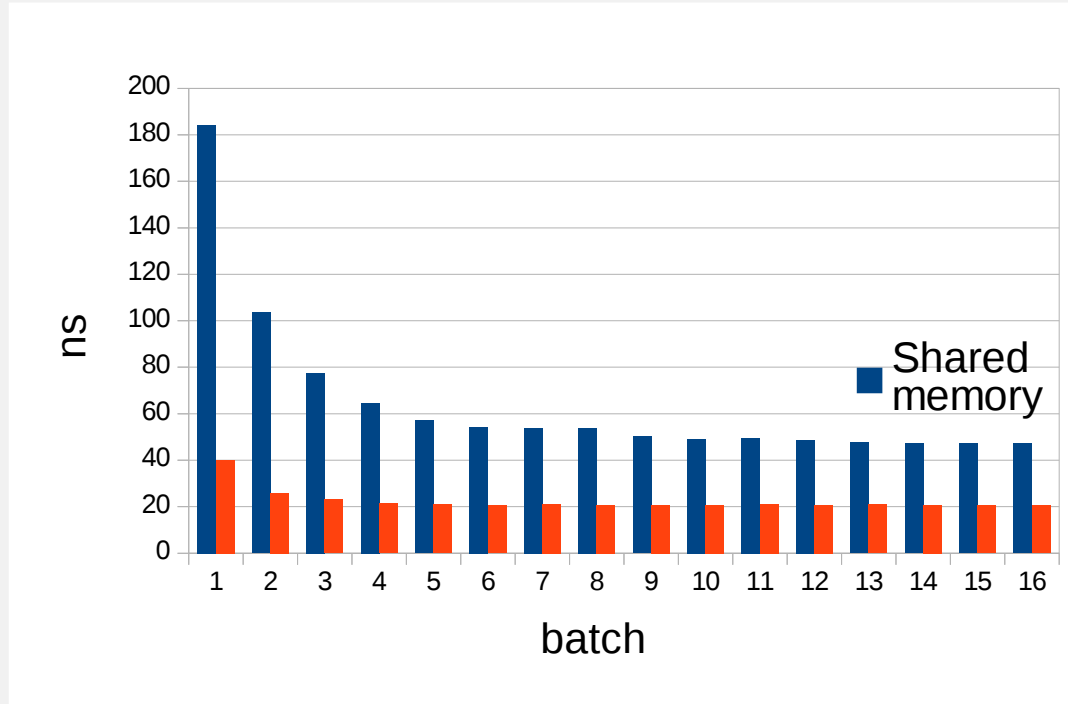
Backup

Virtio-net as failover device

Upcoming features

- VIRTIO_NET_F_STANDBY
- enables hypervisor controlled live migration to be supported with VMs that have direct attached SR-IOV VF devices.
- VM has pass-through device and virtio-net device with same MAC
- Before live migration switch to virtio-net device, at target system

Cache miss cost



Virtio-net as failover device

Upcoming features

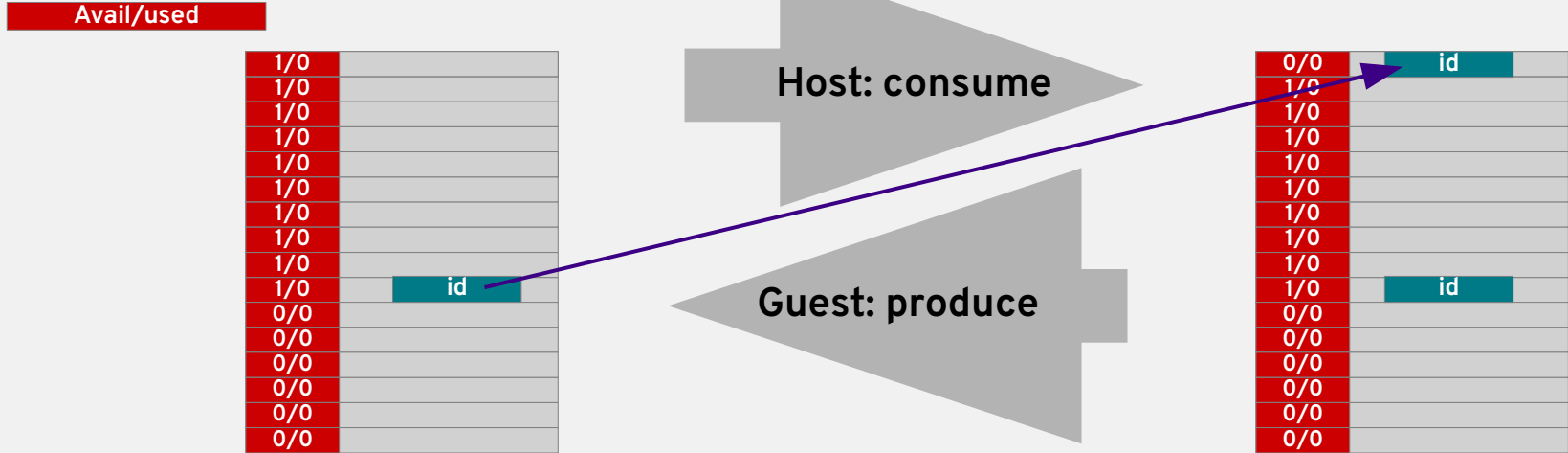
- VIRTIO_NET_F_STANDBY
- enables hypervisor controlled live migration to be supported with VMs that have direct attached SR-IOV VF devices.
- VM has pass-through device and virtio-net device with same MAC
- Before live migration switch to virtio-net device, at target system

In-order processing: descriptor ID

Past year achievements

Guest: produced 9

Host: consumed 9



One write per batch of descriptors