DPDK Is No Longer As Greedy!

- DPDK can now allocate hugepage memory as needed
- DPDK can also release memory that is unused
- DPDK can put pages into fewer files
  - Small page sizes and virtio are not enemies anymore!
- (18.08+) DPDK no longer requires a hugetlbfs mountpoint
Looking Inside

ANATOLY BURAKOV
What Changed in 18.05?

Main design goal:

*Ability to map/unmap hugepages at runtime, not just startup*

Everything else is side effect and/or practical necessity!
Question:
• How do you keep IOVA-contiguous memory without pre-sorting pages?

Answer:
• You don’t!
  • In 18.05, we deal with pages, not segments
  • Memory is no longer guaranteed to be IOVA-contiguous
Memory Rework Design Principles

Question:
• What if you need IOVA-contiguous memory?

Answer:
• Chances are, you actually don’t…
• Ask for it!
  • Normal malloc API’s will not allocate IOVA-contiguous memory
  • Memzone allocator has a flag to request IOVA-contiguous memory
• Use VFIO for everything
• Use legacy mode
Memory Rework Design Principles

Question:

• How do we guarantee secondary process has the same view of memory?

Answer:

• Preallocate all VA space at startup!
  • Page table are synchronized over DPDK IPC
  • Primary has authority over what pages get used
Legacy DPDK Memory Architecture

- VA layout follows PA layout
- VA and PA layout is fixed
18.05+ DPDK Memory Architecture

- VA layout is independent from PA layout
- VA layout is fixed, PA layout is not
Changes & New Features in 18.05+

New API’s:

- New memzone flag:
  - RTE_MEMZONE_IOVA_CONTIG

- Memory event and validation callbacks
  - Page map/unmap events
  - Allow/deny new page mappings over specified limit

- Page walk and lookup API’s
  - rte_memseg_walk et al.
Changes & New Features in 18.05+

EAL parameters:

- `-m/--socket-mem` is now a **minimum**, not a **limit**
  - Think guaranteed memory availability
- `--single-file-segments`
  - Creates fewer hugepage files
- `--legacy-mem`
  - Mimics old DPDK
- `--limit-mem (18.08+)`
  - Place upper limit on memory usage, per socket
- `--in-memory (18.08+)`
  - Run without hugetlbfs mountpoint
Future Changes (18.11+)

External memory support
- Currently RFC, V1 will be submitted for 18.11
- Using normal DPDK allocators with non-DPDK memory!

Memfd hugepages support for --in-memory mode
- Allows running without hugetlbfs and use virtio/vhost
  - Patches currently at V1
  - Virtio patches currently RFC
- Makes DPDK easier to set up in Cloud Native environments
Why You Should Care

Generally, memory in DPDK is designed to be invisible, so why should anyone care?

- Because we can accidentally break stuff!

When changes happen, certain things may break because:

- Code makes assumptions about memory layout
- Code makes assumptions about internals of DPDK

Memory management is fundamental to DPDK, so changes in memory subsystem can potentially affect everyone!

- Call for more reviews of memory-related patches
Memory Layout Dependency

Problem:
- Certain drivers in DPDK relied on PA layout for lookups
  - Few memsegs to look through => little impact on performance
- After applying 18.05 memory hotplug changes, there was a noticeable performance drop

Solution:
- For affected drivers, stopgap solution was implemented for 18.05
  - Performance still impacted for small page sizes
- Proper solution expected for 18.11
Memory Layout Dependency

Problem:
• net/virtio relies on valid memory starting from offset 0 into page table
• A patch to 18.08 made it so that segments are allocated from the top of VA space
• As a result, net/virtio had issues trying to share more memory than was needed

Solution:
• Reverted the patch for 18.08
• Investigation still ongoing
Q&A

Anatoly Burakov (anatoly.burakov@intel.com)
Bruce Richardson (bruce.richardson@intel.com)