



KVM
FORUM

Live Migration with Hardware Acceleration

Presented by: Wei Wang <wei.w.wang@intel.com>

Contributors: Guang Zeng, Yi Sun, Kevin Tian, Jun Nakajima, Xin Zeng

Agenda

- Project Goals
- Architecture Introduction
- Feature Introduction
- Test Results
- Future Works



Project Goals

Project Goals

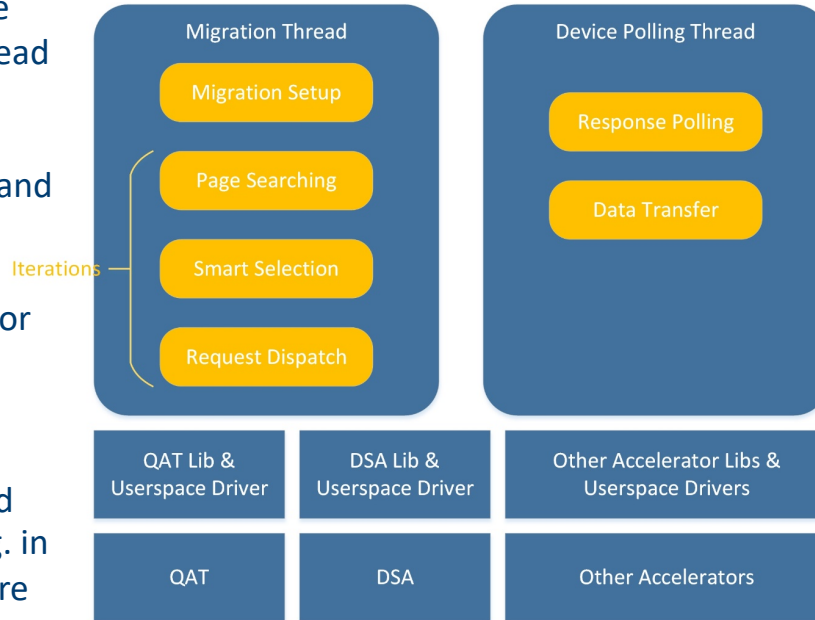
- Live migration pain points
 - VMs with memory write intensive workloads are difficult to migrate
 - VMs with large memory size takes long time to migrate
 - May consume large network bandwidth
- Existing solution: compression with CPUs
 - Slow
 - Consumes too many CPUs from host
- Our solution
 - Offload the compression part to Intel QAT with efficient approaches
 - Higher migration throughput
 - Lower CPU utilization
 - A common design ready for future more accelerators to join in
 - Data Streaming Accelerator (DSA) and Intel Analytics Accelerator (IAX) coming on Sapphire Rapids CPUs
 - Smart Selection



Architecture Introduction

Source Machine

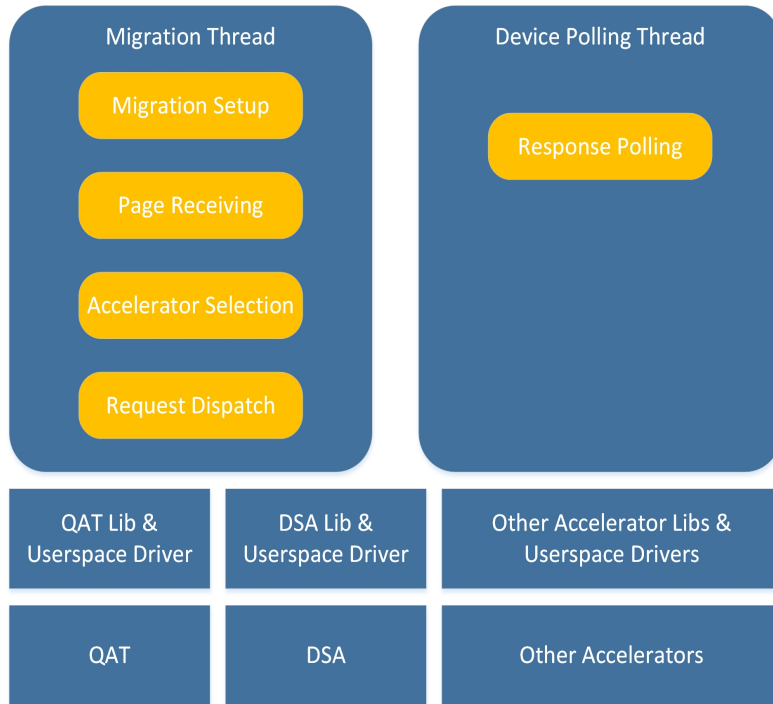
- Migration Setup
 - Preparation for migration, including the accelerator device initialization, device polling thread creation
- Page Searching
 - Searching for pages to process and send
- Smart Selection
 - Select an appropriate accelerator based on the history of the acceleration efficiency
- Request Dispatch
 - Dispatch requests to the related accelerator device instance, e.g. in a round robin fashion if there are multiple device instances



- Response Polling
 - Poll for responses from all the devices
 - Blocks when no responses are ready
- Data Transfer
 - Send the compressed data, along with the related header, to network

Destination Machine

- Page Receiving
 - Receive data from the network
 - Parsing the migration protocols, e.g. multi-page
- Accelerator Selection
 - The received data has headers to tell which accelerator to use



- Response Polling
 - Poll for decompressed data from each device
 - Blocks when no responses are ready
 - Decompressed data DMA to the QEMU memory

A group of people sitting in an audience, smiling and clapping, with a green overlay. The text "Feature Introduction" is overlaid on the image.

Feature Introduction

Important Features

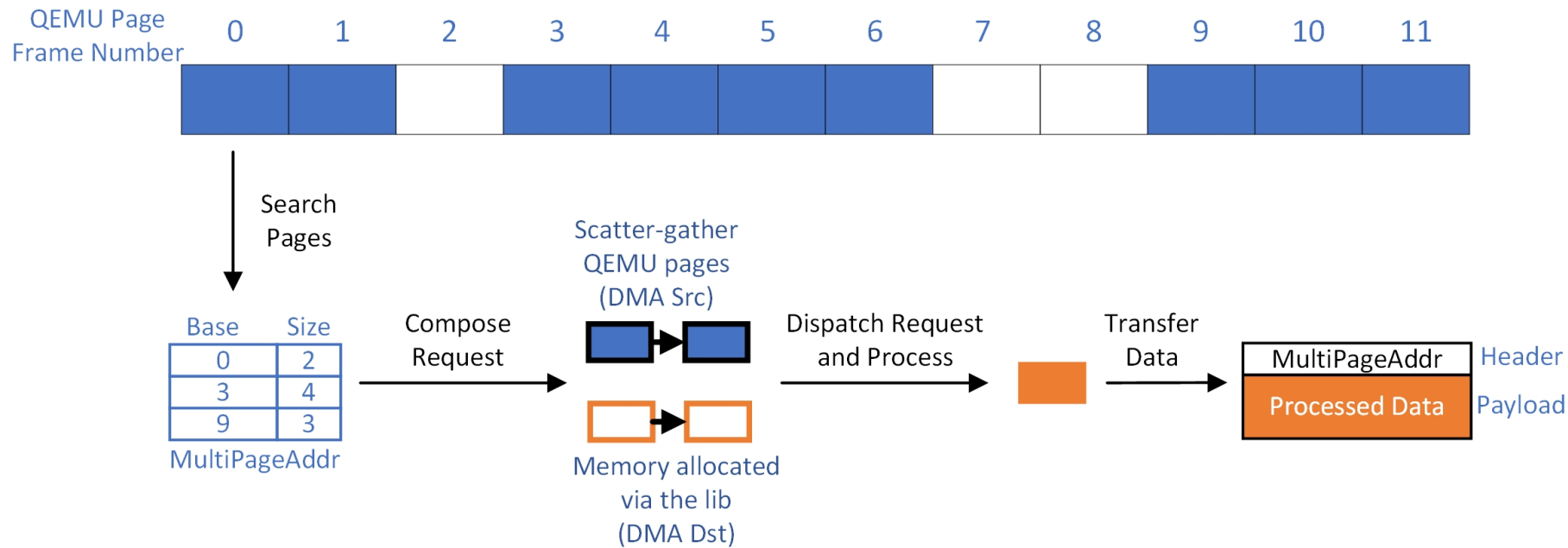
- Zero-copy
 - Allow the acceleration device to directly access to the guest(QEMU) memory
- Multi-page Processing
 - Support the whole migration flow to process multiple pages each time
- Acceleration Request Caching
 - Caching the acceleration request data structure for efficient memory allocation

Zero-copy

- Migration setup
 - Pre-alloc and pin all the QEMU memory
 - Destination side memory unpinned when migration is done
- Request Composing
 - Source side
 - DMA read buffer points to QEMU memory
 - DMA write buffer allocated via accelerator lib
 - Destination side
 - DMA read buffer allocated via accelerator lib
 - DMA write buffer points to QEMU memory

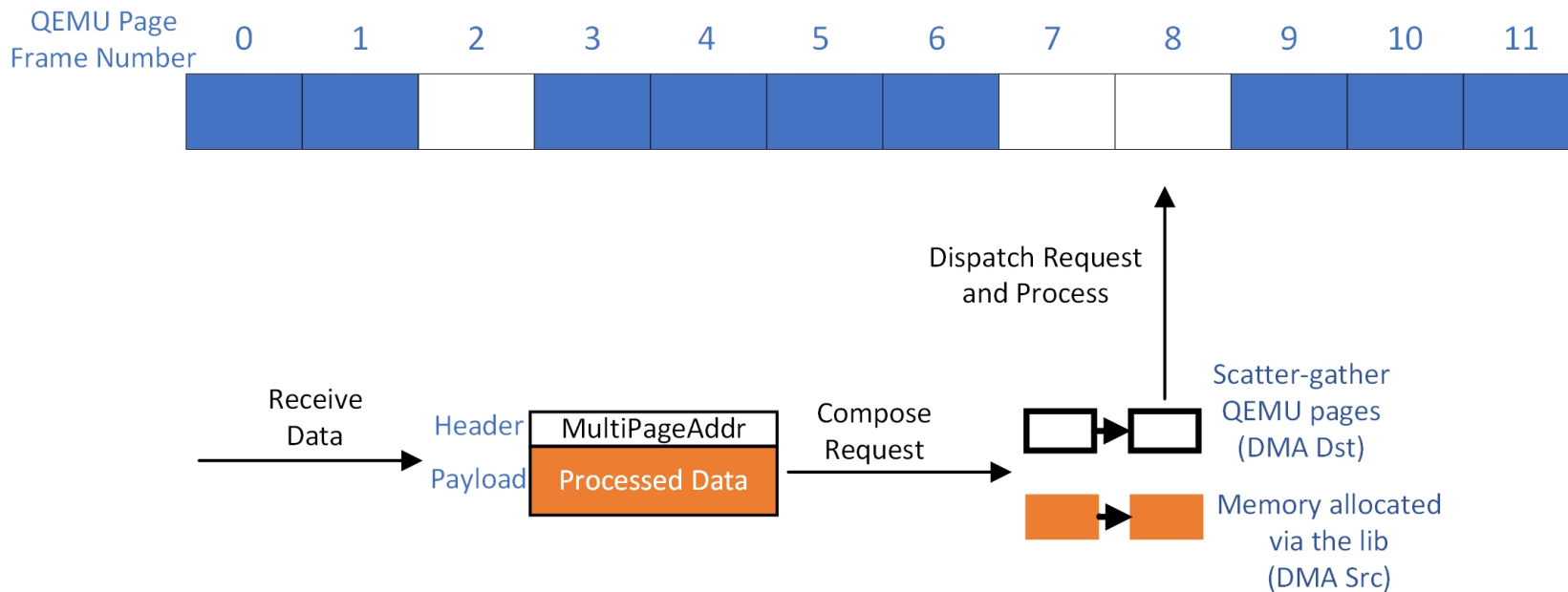
Multi-page Processing

source machine



Multi-page Processing

destination machine



Acceleration Request Caching

- Device Setup
 - Pre-allocate some amount of acceleration requests and fill them into the cache pool
- Request Composing
 - Take requests from the cache pool first
 - Initialize the request based on the new pages to send
- Response Polling
 - Free the request to the cache pool after it's processed

A group of people, likely at a conference or event, are shown from the chest up, sitting in rows. They are all smiling and looking towards the right side of the frame. Some are clapping their hands. The image has a strong green color overlay. The text "Test Results" is overlaid on the left side of the image.

Test Results

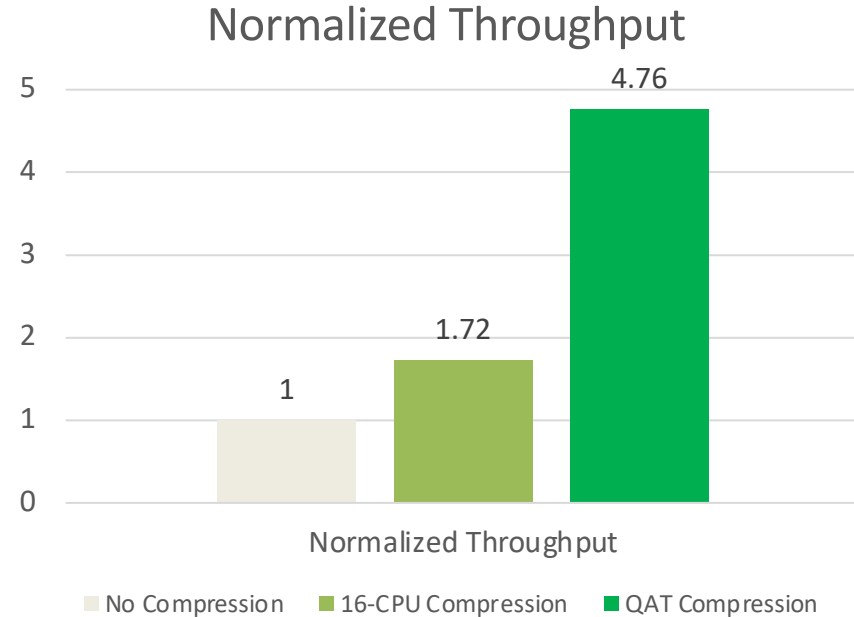
Test Environment

- Testbed
 - CPU: Intel Xeon CPU E5-2699 V4 @2.2GHZ
 - QAT: 8960 PCIe card, Gen3
 - DRAM: DDR4, 2666MHZ
 - NIC: XL710, 40GB
- Live migration
 - Downtime: 300 ms (default)
 - Network bandwidth: No limit (i.e. 40G)
 - Compress level: 1
 - Multi-page: 63 (Max)
- Guest
 - 4 vCPUs, 32G RAM, running a workload writing compression-friendly data
 - 4 vCPUs, 32G RAM, running a workload writing sequence numbers
 - 8 vCPUs, 128G RAM, running memcached with reading/writing random numbers

Memory Dirty with Compress-Friendly Data

- Run in guest: `./dirty_workload -t 10 -i 500000 -m 1 1000 -s`
 - Write “1”s in specified dirty rate (e.g. 1000 MB/s above)

	No Compression	16 CPU Compression	QAT Compression
Throughput (Pages per Second * 10000)	17 ~ 29	39 ~ 50	133 ~ 138
Largest Migratable Dirty Rate (MB/s)	1100	1900	5000
Extra CPU Utilization (%)	No	678	< 40
Compression Ratio	No	87.6	922

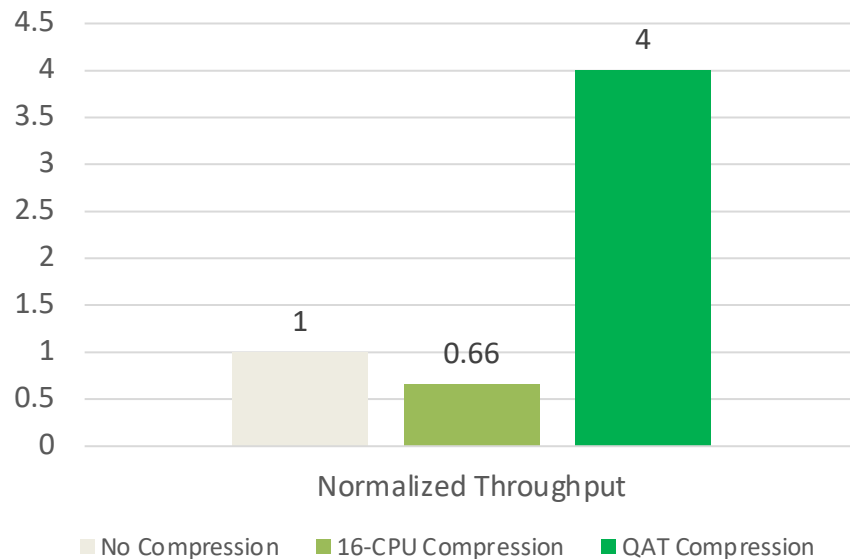


Memory Dirty with Sequence Numbers

- Run in guest: `./dirty_workload -t 10 -i 500000 -m 3 1000 -s`
 - Write sequence data in specified dirty rate (e.g. 1000 MB/s above)

	No Compression	16 CPU Compression	QAT Compression
Throughput (Pages per Second * 10000)	17 ~ 29	~19	81~116
Largest Migratable Dirty Rate (MB/s)	1100	700	4200
Extra CPU Utilization (%)	No	1600	< 70
Compression Ratio	No	4.97	4.81

Normalized Throughput

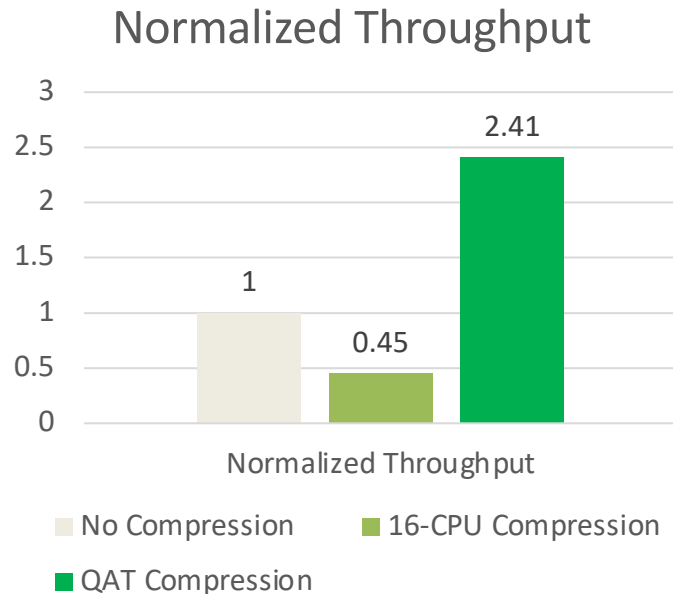


Memcached with Random Numbers

- Memcached Server
 - 16 servers, each with 4GB RAM
- Memslap Client
 - 16 threads, 16 concurrency
 - Set/Get ratio 9:1
 - key length 128 bytes, value length 2048 bytes

	No Compression	16 CPU Compression	QAT Compression
Throughput (Pages per Second * 10000)	18 ~ 22	7 ~ 10	48 ~ 53
Migration Time (second)	Infinite	Infinite	60
Dirty Sync Count	Infinite	Infinite	10
Compression Ratio	No	1.6	1.6

- VM fails to be migrated in the “no compression” and “cpu compression” cases, and successfully migrated with QAT acceleration





Future Works

VFIO Driver based Zero-Copy

- Current Zero-copy is implemented based on the UIO based QAT driver
 - Requires QEMU to be root privilege to get VA-to-PA mappings via pagemap
 - Requires QEMU to pin its memory
- VFIO supports the above with QEMU running with non-root privilege
 - QAT's VFIO based userspace driver and library are work in progress

Smart Acceleration Support

- DSA compares the dirty memory, and sends the “diff” to the destination only
 - Good when the guest only modifies a small part of a page
 - Bad when the entire pages are changed
- Smart Acceleration
 - Dynamically switch to use QAT/IAX compression or DSA diff during live migration using a prediction based on the compression ratio history and diff ratio history



Q & A

Disclaimers

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request. No product or component can be absolutely secure.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm

Intel, the Intel logo, and Xeon are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others

© Intel Corporation.



KVMM FORUM