Evolution of SPDK Towards Secure Container Storage Service

Liu Xiaodong & Liu Changpeng
Intel
SPDK vhost solution

SPDK storage virtualization solution – vhost target, is widely deployed by lots of CSPs and enterprises in their infrastructure.
Consideration on VM-based Secure Container
Secure Container

- Bedrock for public cloud container service
- Popular in CSP
- VM based, like Kata Containers

- Can SPDK vhost directly be applied to Kata Containers?
Characteristics from Secure Container

- **High density**
  - Serve more than 1 thousand of containers on a single host which means 1K of lightweight VMs

- **Over-provisioning**
  - CPU and memory resource are tense seriously

- **Not performance pursuer**
  - Flexibility and robustness
Problem met with SPDK vhost

High density

- SPDK polls each virtqueue in rounds
- Polling to query massive virtqueues is not efficient

http://SPDK.io
Problem met with SPDK vhost

Over-provisioning

- CPU occupation caused by polling
- Memory pre-allocation caused by hugepage & userspace DMA
Evolve SPDK Application to Be Interruptable?

Look Back on SPDK
SPDK Concurrency Theory

- SPDK takes message passing as its concurrency strategy
SPDK Message Passing Infrastructure

SPDK thread abstraction for basic message passing

- spdk_thread
- spdk_poller
- spdk_msg
- spdk_io_device
- spdk_io_channel
SPDK Event/App Framework

**EVENT**

**REACTOR**

**LW_THREAD**  *(LIGHT WEIGHT)*

**POLLING MODEL**
SPDK Vhost Target in Polling

- Vhost device specific `spdk_thread`
- Pollers to take and process Virtqueue as frontend, and Bdev as backend
- Polling executed in Reactors
Evolve SPDK Application to Be Interruptable?

Let’s do it!
Interrupt Abstraction

Epoll instance with target file descriptors

- Eventfd for internal queue notification
- Socket FD for network
- Timerfd for periodic work
- VFIO/UIO eventfd for userspace device interrupt
- Cascading epoll_fd for grouped events
Interrupt Abstraction

- Reactor interrupt abstraction
- SPDK thread interrupt abstraction
Interrupt SPDK Application

- Blocked wait on reactor epoll instance
- Non-blocked wait on spdk_thread epoll instance
Interrupt SPDK Application

Most of SPDK intermediate libraries are originally interruptable

- Basic bdev modules: Raid, Split, GPT, Malloc
- Blobstore and Logical Volume
- Blobfs and its FUSE module
Interruptable SPDK Vhost Target

- A minimal set of interruptable vhost-blk target for secure container evaluation

Evaluation: https://review.spdk.io/gerrit/c/spdk/spdk/+/4584
Interrupt SPDK Application

- Use bdevperf tool for performance evaluation
- Performance drawdown preview of interrupt mode
On Top of Interruptable SPDK Application
Secure Container Storage Service

- Provide volume service to Kata containers via the interruptable SPDK vhost target

More details at [this link](#)
Secure Container Storage Service

• Provide rootfs service to Kata containers via the interruptable SPDK vhost target

More details at this link
Expected Use Case

Containerizing SPDK Application

- No SPDK specific changes needed
- Resource occupancy consideration
- Less impact on density with interruptable SPDK App containers

More details at this link
Summary & Further Evolving

• Polling pinned CPU and hugepage preallocation can be avoided for non-performance situation.
• With interrupt mode, SPDK vhost will be a good choice to provide storage service to secure containers.

• Add interrupt support on userspace hardware Bdev backend: NVMe driver & Bdev, Virtio driver & Bdev
• Add interrupt support on modules related to network: NVMe-oF, ISCSI
• Add running mode switch between polling and interrupt
• Official non-hugepage support for non-DMA SPDK App.
Thank You