浪潮信息云峦 KeyarchOS

基于 QAT 加速迁移测试指导

浪潮电子信息产业股份有限公司

2023 年 10 月

目 录

1 测试概述

1.1 应用场景

随着应用程序的复杂性不断增长，系统需要越来越多的工作负载计算资源，包括密码学相关的加解密和数据压缩，比如云服务中的虚拟机迁移场景，但压缩/解压缩又非常消耗资源，导致时间花费较长，用户体验不佳，甚至还可能出现超时异常。

1.2 技术背景

虚拟机迁移时因为网络问题或脏页不收敛等会导致潜在的超时失败，基于 Intel QAT（QuickAssist Technology）的硬件加速技术。KOS 利用硬件功能，可以在不增加 CPU 负担的情况下，显著提升虚拟机的迁移性能，缩短迁移花费的时间，消除潜在的超时失败危害。

1.3 测试内容

本文档是在迁移场景下，使用 QAT 加速压缩/解压缩性能的测试总结，主要测试内容包括，QAT 加速迁移的效果测试，涉及到的测试用例总计 2 条。

1.4 术语解释

名词	描述
QAT	Intel 加速技术
GZIP	数据压缩的格式
QEMU	虚拟化模拟软件

2 软硬件环境

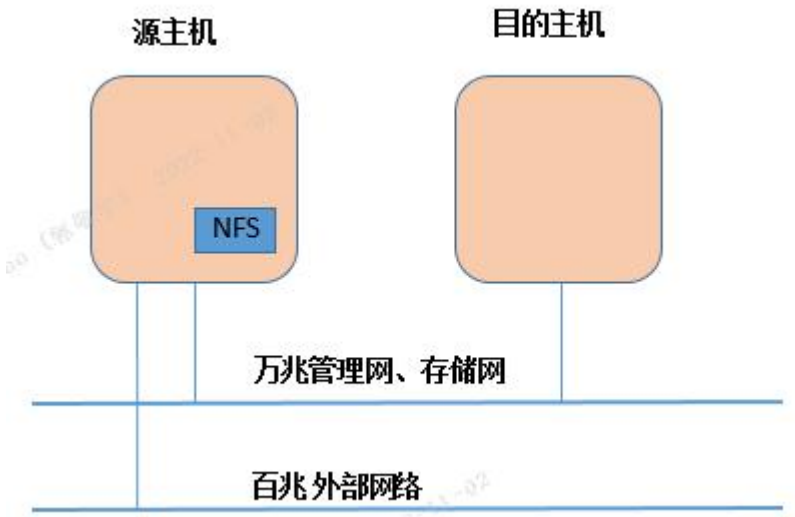
2.1 硬件

设备	配置说明	
1	服务器型号	QuantaGrid D54Q-2U
2	处理器	Intel(R) Xeon(R) Platinum 8480+ CPU @2.00GHz * 224
4	内存	1024GB (16x64GB 4800 MT/s [4400 MT/s]); 2086400MB (8x260800MB Logical non-volatile device 4800 MT/s [4400 MT/s])
5	硬盘	INTEL SSDPE21K750GA *1 INTEL SSDPE2KX020T8 *1
6	网卡	配置 2 个 主板集成千兆网卡
7	QAT 数量（集成于 CPU）	2 QAT acceleration device

2.2 测试工具

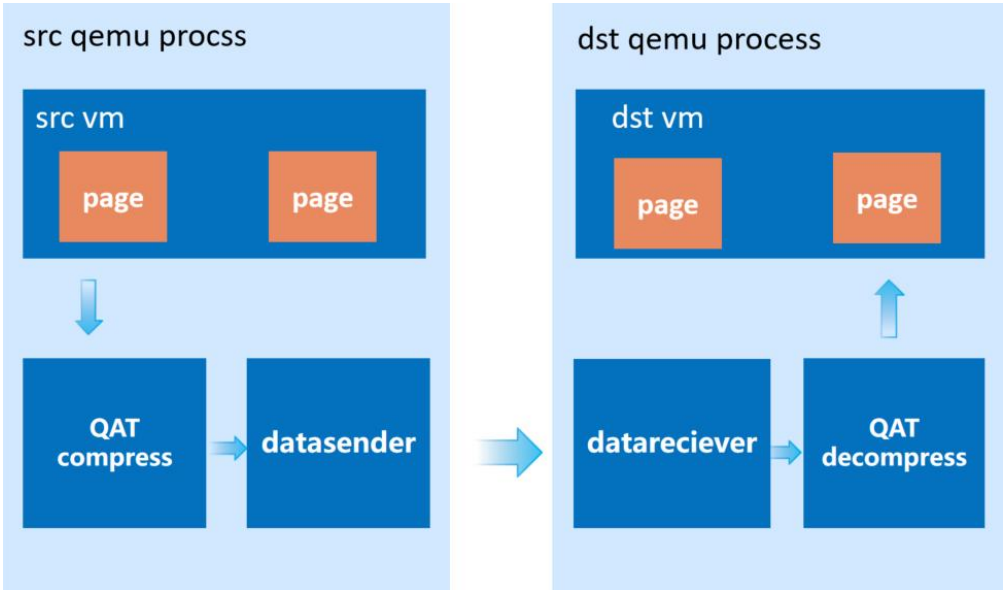
序号	工具名称	说明
1	qemu shell	qemu 虚拟机进程的管理进程以 stdio 模式运行对用户提供的 shell 交互式管理工具
2	src-boot-4C-XgRAM.sh	基于 qemu 进行热迁移测试时，源节点虚拟机启动脚本，封装 qemu 命令，支持传参虚拟机内存大小，单位 GB。详见附录附件。
3	dst-boot-4C-XgRAM.sh	基于 qemu 进行热迁移测试时，目的节点虚拟机启动脚本，封装 qemu 命令，支持传参虚拟机内存大小，单位 GB。 详见附录附件，**在实际使用中注意修改该脚本的 ip 地址为当前节点（目标节点）的迁移网络 ip**;

2.3 测试环境拓扑



2.4 业务架构

使用 NFS 共享的虚拟机进行迁移时，脏页的数量会跟随着用户的使用而不断产生，且迁移时如果不能在有限的时间内迁移完毕就会导致迁移失败。如下图所示，为了尽量避免这种情况的发生，可以使用在被迁移对象侧发送内存之前使用 QAT 压缩且迁移目的对象接收之后使用 QAT 解压缩的技术提高迁移效率。



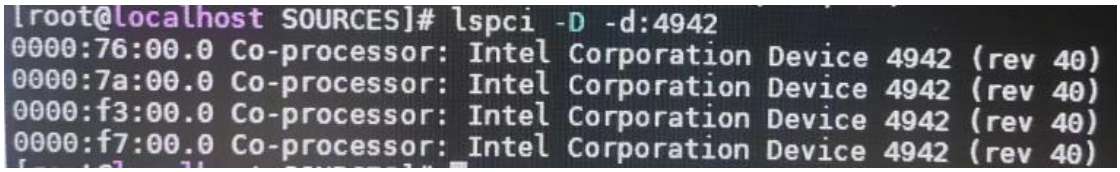
3 测试指导

3.1 BIOS 配置及内核配置

QAT 支持作为物理设备使用：

BIOS 中的 Directed I/O (VT-d) 可以是 Disable，也可以是 Enable，但内核必须添加 intel_iommu=off 参数。

验证当前机器是否存在 QAT 设备：lspci -D -d :4942 或 lspci -D -d 4940。



3.2 安装与测试的前置条件

当前机器为 SPR 且存在 QAT 设备才可进行后续安装和测试

3.3 测试环境安装(以 x86_64 为例)

（安装需要的软件包请参见附录）

QAT 驱动编译：

1、tar -zxvf qat20.L.0.9.6-00024

2、./configure --prefix=/usr（也可以自定义路径，但是 qemu 配置时需适配）

3、make

4、make install

5、修改 qat 配置文件，保存退出

vim /etc/c6xx_dev0/1/2.conf,将所有的“DcoIspPolled”字段值改为 2

6、重新加载配置文件 adf_ctl restart;

7、查看 qat 设备的状态：adf_ctl status，保证所有设备都是 up 状态

Qemu 编译：

1、git clone https://github.com/qemu/qemu.git

2、cd qemu

3、git reset --hard e0fb2c3d89aa77057ac4aa073e01f4ca484449b0（拉齐版本）

4、git am 0001*.patch (依次打入 patch)

5、mkdir build && cd build

6、../configure --target-list=x86_64-softmmu --with-git='tsocks git' --disable-git-update --disable-slirp

7、根据 config-host.mak.diff 修改 config-host.mak

8、make -j32

9、make 结束之后会在 x86_64-softmmu/目录下生成 qemu-system-x86_64 二进制文件

libvirt 编译：

1、打入以下链接的补丁

https://github.com/intel/libvirt-tdx/commit/446df09b46d96514ccb25e4b37b7d84ce873d353

2、make

3、make install

4 测试用例及测试数据

4.1 测试用例

测试用例 1

测试类型	QAT迁移加速测试	测试工具	Qemu
测试目的	SPR架构主机下qemu启用QAT加速时4c-32g-50g虚拟机的热迁移指标与不启用QAT加速迁移对比		
前提条件 1、主机关闭 numa balance 2、主机正确安装 QAT driver 3、主机 QAT service 正确配置（QAT 实例开启 epoll 模式）并启动 4、源主机启用 nfs 服务，将虚拟机镜像共享，目标主机挂载共享目录			
测试过程 不启用QAT 1.源节点执行：sh src-boot-4C-XgRAM.sh 32			

2.目的节点执行：sh dst-boot-4C-XgRAM.sh 32 3. 在源节点的 qemu shell 中输入命令触发迁移： (qemu) migrate -d tcp:<迁移网络目标节点 ip>:4444 4. 在步骤 3 后，在源节点等待迁移、查询迁移进度、确认迁移完成后的统计数据 (qemu) info migrate 5. info migrate 查看迁移时间 启用 QAT 1.源节点执行：sh src-boot-4C-XgRAM.sh 32 2.目的节点执行：sh dst-boot-4C-XgRAM.sh 32 3. 源节点在 qemu shell 中输入命令配置启用加速： (qemu) migrate_set_speed 0 (qemu) migrate_set_capability compress on (qemu) migrate_set_parameter compress-with-QAT true (qemu) migrate_set_parameter QAT-zero-copy true (qemu) migrate_set_parameter compress-level 1 4. 目的节点在 qemu shell 中输入命令配置启用加速： (qemu) migrate_set_capability compress on (qemu) migrate_set_parameter compress-with-QAT true (qemu) migrate_set_parameter QAT-zero-copy true 5. 在源节点的 qemu shell 中输入命令触发迁移： (qemu) migrate -d tcp:<迁移网络目标节点 ip>:4444 6. 在步骤 5 后，等待迁移、查询迁移进度、确认迁移完成后的统计数据 (qemu) info migrate 7、info migrate 查看迁移时间
预期目标 记录数据
测试结果：☐ 通过　　☐ 不通过

备注			

测试用例 2

测试类型	QAT迁移加速测试	测试工具	libvirt
测试目的	SPR架构主机下libvirt启用QAT加速时4c-32g-50g虚拟机的热迁移指标与不启用QAT加速迁移对比		
前提条件 1、主机关闭 numa balance 2、主机正确安装 QAT driver 3、主机 QAT service 正确配置（QAT 实例开启 epoll 模式）并启动 4、虚拟机配置使用编译的 qemu			
测试过程			

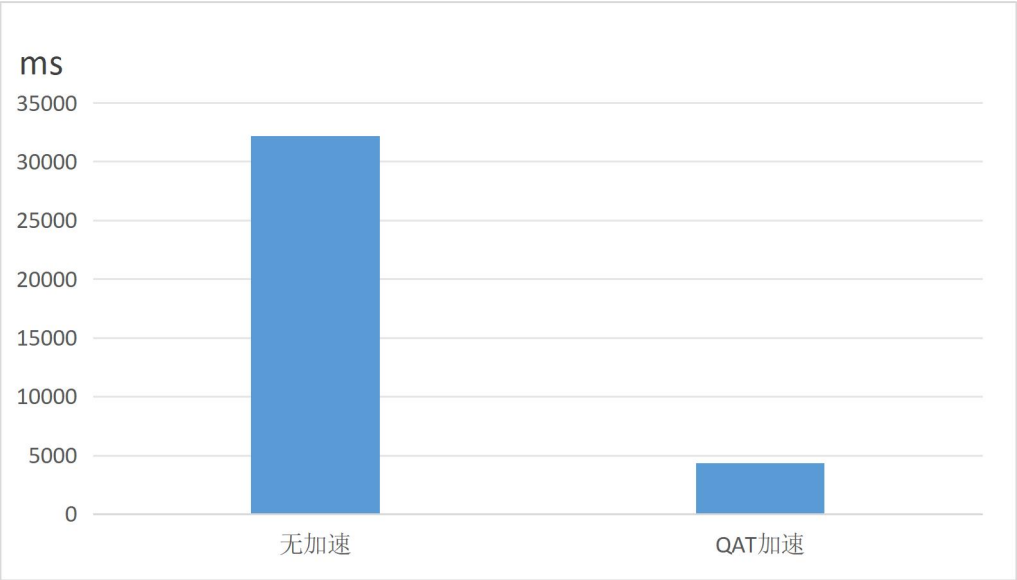
<table>
<tr><td colspan="4">不启用QAT
1. 源节点执行：使用 virsh 命令在源节点启动虚拟机；
 

（以虚拟机名称 wdm5.8 为例）
2. 在源节点执行 virsh 迁移命令迁移虚拟机并计时：
virsh migrate kkx --p2p --live qemu+tcp://172.16.16.42:16509/system tcp://172.16.16.42 --unsafe --verbose
3. 在步骤 2 后，等待迁移结束统计迁移时长；
启用 QAT
1. 源节点执行：使用 virsh 命令在源节点启动虚拟机；
 

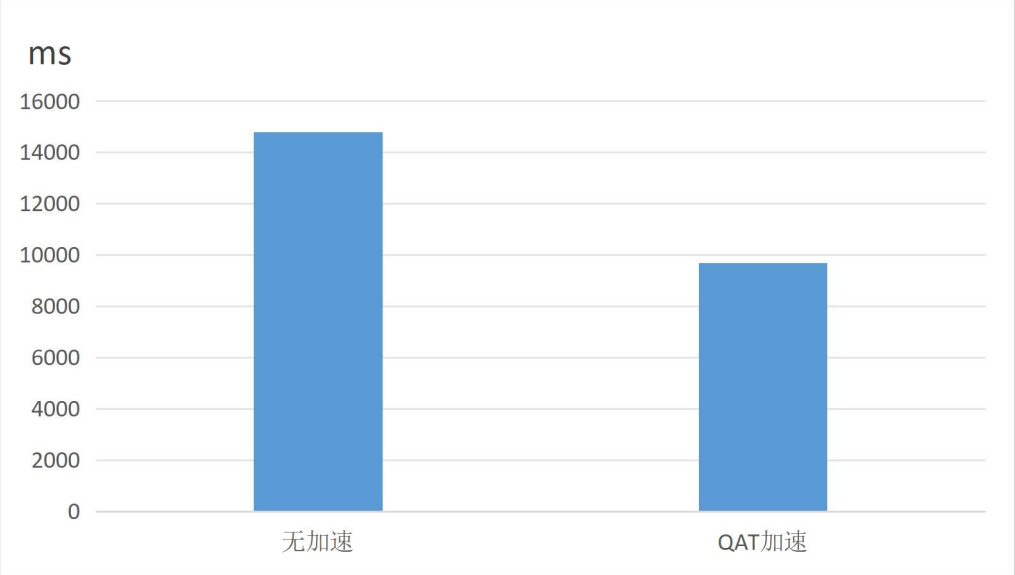
（以虚拟机名称 wdm5.8 为例）
2. 在源节点执行 virsh 迁移命令迁移虚拟机并计时：
virsh migrate kkx --p2p --live --comp-methods qat --comp-qat-zero-copy --comp-mt-level 1 qemu+tcp://172.16.16.42:16509/system tcp://172.16.16.42 --unsafe --verbose
3. 在步骤 2 后，等待迁移结束统计迁移时长；</td></tr>
<tr><td colspan="4">预期目标
记录数据</td></tr>
<tr><td colspan="4">测试结果：☐ 通过 ☐ 不通过</td></tr>
<tr><td>备注</td><td></td><td></td><td></td></tr>
</table>

4.2 测试数据

SPR 架构主机下 qemu 启用 QAT 加速时 4c-32g-50g 虚拟机的热迁移指标与不启用 QAT 加速迁移对比



SPR 架构主机下 libvirt 启用 QAT 加速时 4c-32g-50g 虚拟机的热迁移指标与不启用 QAT 加速迁移对比



5 分析与结论

综合上述测试数据来看：qemu 方式迁移时使用 qat 迁移加速对比不使用加速场景有 8 倍左右的提升；libvirt 方式使用 qat 迁移加速对比不适用加速场景有 1 倍左右的提升(libvirt 是在调用 qemu 之前封装了一层自己的迁移逻辑，包括参数校验，迁移通道建立等逻辑，因此相比于 qemu 有额外的开销)。

6 可能存在的问题与解决方式

无

7 附录

dst-boot-4C/src-boot-4C-XgRAM 脚本的路径需要适配测试环境的具体路径：

```
/opt/qemu-git/qemu/x86_64-softmmu/qemu-system-x86_64    -accel kvm \
 -mem-prealloc -overcommit mem-lock=100 \
 -drive file=/opt/qemu-git/shared-pool/centos7-mini.qcow2,if=none,id=virtio-disk0 \
-device virtio-blk-pci,drive=virtio-disk0 \
-m $(($1*1024)) -smp 4 -cpu host   -device cirrus-vga -vnc :9   \
-netdev
   type=tap,script=/opt/qemu-git/shared-pool/ifup-script,downscript=no,id=net0\
-device virtio-net-pci,netdev=net0,mac=fa:00:00:00:00:01 -monitor stdio
```

注：

1、/opt/qemu-git/qemu/x86_64-softmmu/qemu-system-x86_64 需要适配编译获取的 qemu-system-x86_64 路径

2、/opt/qemu-git/shared-pool/centos7-mini.qcow2 为配置的 NFS 共享路径