QEMU参数解析

2015-09-26

前言

快毕业了，准备把虚拟化的东西都整理一下，准备开始新的旅程。希望这是一个系列，主要涉及虚拟化的理论与实践，包括但不限于理论基础、源码分析、外文翻译以及Demo实例。本篇文章首先分析一下QEMU的参数解析部分。

一．使用gdb分析QEMU代码

1. 使用configure生成Makefile的时候指定参数包括-enable-kvm -enable-debug -target-list=”x86_64-softmmu”
2. 从QEMU官方下载一个精简镜像linux-0.2.img wget http://wiki.qemu.org/download/linux-0.2.img.bz2
3. 启动gdb调试QEMU命令

```
gdb -args /usr/bin/bin/qemu-system-x86_64 linux-0.2.img -m 512 -enable-kvm -smp 1,sockets=1,cores=1,threads=1 -realtime mlock=off -device ivshmem,shm=ivshmem,size=1 -device ivshmem,shm=ivshmem1,size=2
```

这里使用了这么多参数，主要是为了之后对QEMU解析参数有比较好的理解。

二．QEMU参数解析

QEMU定义了QEMUOption来表示QEMU程序的参数选项。定义如下

```
typedef struct QEMUOption {
    const char *name;
    int flags;
    int index;
    uint32_t arch_mask;
} QEMUOption;
```

vl.c中在全局范围定义了一个qemu_options存储了所有的可用选项。

```
static const QEMUOption qemu_options[] = {
  { "h", 0, QEMU_OPTION_h, QEMU_ARCH_ALL },
#define QEMU_OPTIONS_GENERATE_OPTIONS
#include "qemu-options-wrapper.h"
  { NULL },
};
```

qemu_options的生成使用QEMU_OPTIONS_GENERATE_OPTIONS编译控制以及一个文件qemu-options-wrapper.h填充。在qemu-options-wrapper.h中，根据是否定义QEMU_OPTIONS_GENERATE_ENUM、QEMU_OPTIONS_GENERATE_HELP以及QEMU_OPTIONS_GENERATE_OPTIONS以及qemu-options.def文件可以生成不同的内容。qemu-options.def是在Makefile中利用scripts/hxtool脚本根据qemu-options.hx文件生成的。

在这里只需要理解，qemu_options中包括了所有可能的参数选项，比如上面的-enable-kvm -smp -realtime -device等。

QEMU将所有参数分成了几个大选项，比如-eanble-kvm和-kernel都属于machine相关的， 每一个大选项使用QemuOptsList表示，QEMU在qemu-config.c中定义了

```
static QemuOptsList *vm_config_groups[48];
```

这表示可以支持48个大选项。 在main函数中用qemu_add_opts将各个QemuOptsList添加到vm_config_groups中

```
    qemu_add_opts(&qemu_drive_opts);
    qemu_add_drive_opts(&qemu_legacy_drive_opts);
    qemu_add_drive_opts(&qemu_common_drive_opts);
    qemu_add_drive_opts(&qemu_drive_opts);
    qemu_add_opts(&qemu_chardev_opts);
    qemu_add_opts(&qemu_device_opts);
    qemu_add_opts(&qemu_netdev_opts);
    qemu_add_opts(&qemu_net_opts);
    qemu_add_opts(&qemu_rtc_opts);
    qemu_add_opts(&qemu_global_opts);
    qemu_add_opts(&qemu_mon_opts);
    qemu_add_opts(&qemu_trace_opts);
    qemu_add_opts(&qemu_option_rom_opts);
    qemu_add_opts(&qemu_machine_opts);
    qemu_add_opts(&qemu_mem_opts);
    qemu_add_opts(&qemu_smp_opts);
    qemu_add_opts(&qemu_boot_opts);
    qemu_add_opts(&qemu_sandbox_opts);
    qemu_add_opts(&qemu_add_fd_opts);
    qemu_add_opts(&qemu_object_opts);
    qemu_add_opts(&qemu_tpmdev_opts);
    qemu_add_opts(&qemu_realtime_opts);
    qemu_add_opts(&qemu_msg_opts);
    qemu_add_opts(&qemu_name_opts);
    qemu_add_opts(&qemu_numa_opts);
    qemu_add_opts(&qemu_icount_opts);
    qemu_add_opts(&qemu_semihosting_config_opts);
    qemu_add_opts(&qemu_fw_cfg_opts);
```

每个QemuOptsList存储了大选项支持的所有小选项，比如

```
static QemuOptsList qemu_realtime_opts = {
    .name = "realtime",
    .head = QTAILQ_HEAD_INITIALIZER(qemu_realtime_opts.head),
    .desc = {
        {
            .name = "mlock",
            .type = QEMU_OPT_BOOL,
        },
        { /* end of list */ }
    },
};
```

-realtime只支持一个值为bool的子选项，即只能由-realtime mlock=on/off。 但是像-device这种选项就没有这么死板了，-device并没有规定必须的选项，因为设备有无数多种，不可能规定得过来，解析就是按照“,”或者“=”来解析的。 每个子选项是一个的结构是QemuOpt，定义如下

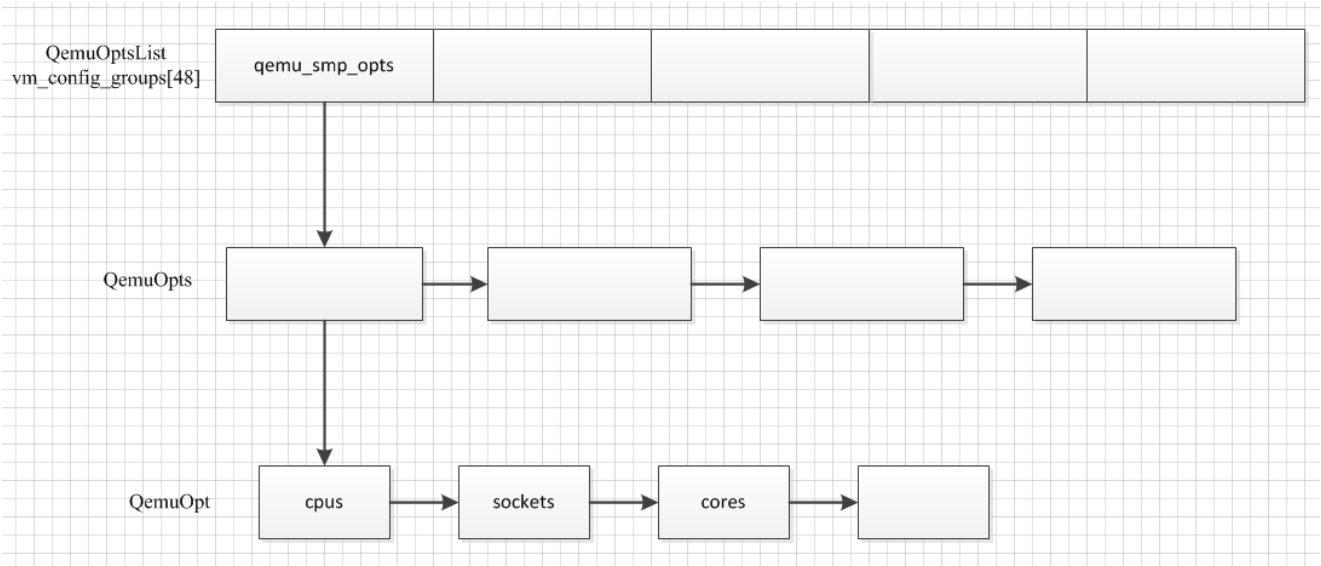
```
struct QemuOpt {
    char *name;
    char *str;

    const QemuOptDesc *desc;
    union {
        bool boolean;
        uint64_t uint;
    } value;

    QemuOpts    *opts;
    QTAILQ_ENTRY(QemuOpt) next;
}
```

name表示子选项的字符串表示，str表示对应的值

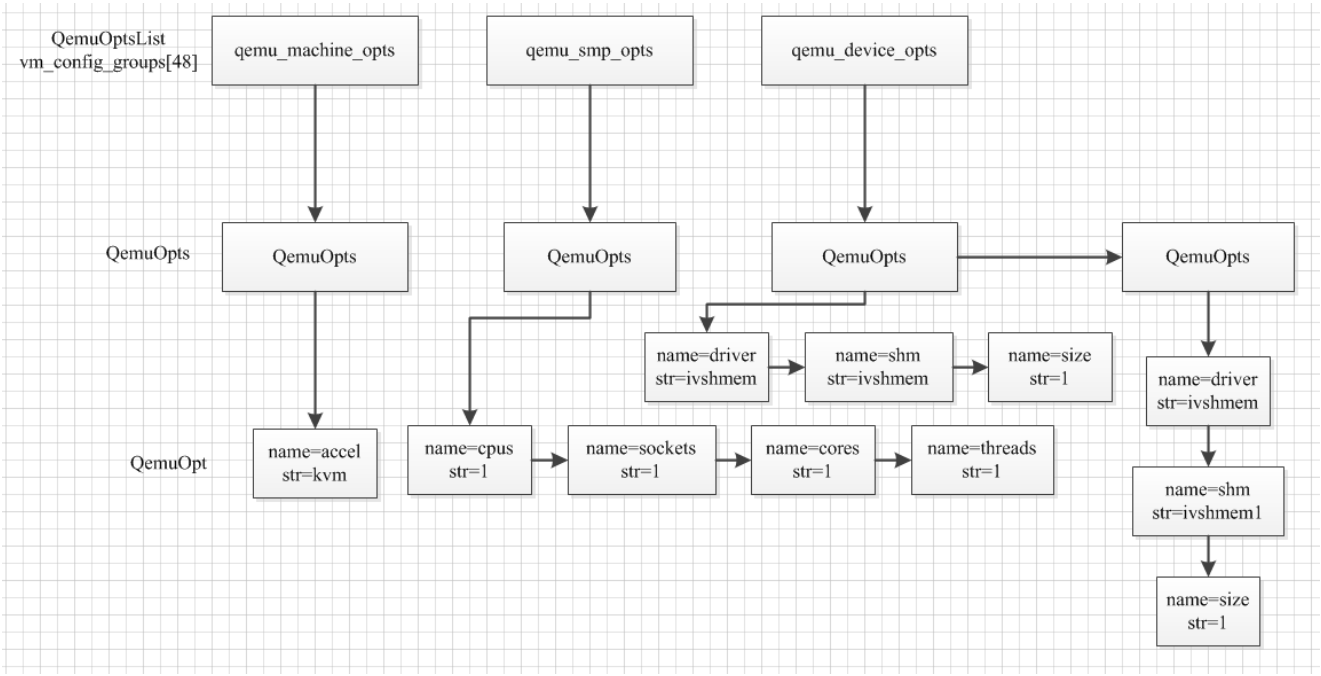
QemuOptsList并不是和QemuOpt直接联系，中间还需要有一层QemuOpts，因为比如上面的可以指定两个-device，这个时候他们都在QemuOptsList的链表上，但是是两个QemuOpts，每个QemuOpts又有自己的QemuOpt链表。QemuOpts结构如下

```
struct QemuOpts {
    char *id;
    QemuOptsList *list;
    Location loc;
    QTAILQ_HEAD(QemuOptHead, QemuOpt) head;
    QTAILQ_ENTRY(QemuOpts) next;
};
```

大体结构如下：



对应本文用的参数，如下（省略了一些参数，比如-m）



参考：QEMU 2: 参数解析（https://www.ibm.com/developerworks/community/blogs/5144904d-5d75-45ed-9d2b-cf1754ee936a/entry/qemu_2_%25e5%258f%2582%25e6%2595%25b0%25e8%25a7%25a3%25e6%259e%2590?lang=en）

IBM的这篇文章不太好理解，后面也有错误

技术 74 (/categories.html#技术-ref)

QEMU 9 (/tags.html#QEMU-ref) 虚拟化 17 (/tags.html#虚拟化-ref)

← Previous (/%E6%8A%80%E6%9C%AF/2015/08/25/pointgame) Archive (/archive.html)
Next → (/%E6%8A%80%E6%9C%AF/2016/04/26/centos6xen4.5)