kata 分析

配置

- load方法见 src/runtime/pkg/katautils/config.go#LoadConfiguration
- 配置结构 src/runtime/pkg/oci/utils.go#RuntimeConfig，
  - 有部分配置支持在 pod 注解中运行时覆盖，见 docs/how-to/how-to-set-sandbox-config-kata.md
- 默认配置来自文件 src/runtime/pkg/katautils/config-settings.go.in
  - MachineType = "q35"
  - VCPUCount uint32 = 1
  - MaxVCPUCount uint32 = 0
  - MemSize uint32 = 2048 // MiB
  - MemSlots uint32 = 10
  - MemOffset uint64 = 0 // MiB
  - VirtioMem bool = false
  - BridgesCount uint32 = 1
  - InterNetworkingModel = "tcfilter"
  - DisableBlockDeviceUse bool = false
  - BlockDeviceDriver = "virtio-scsi"
  - BlockDeviceCacheSet bool = false
  - BlockDeviceCacheDirect bool = false
  - BlockDeviceCacheNoflush bool = false
  - EnableIOThreads bool = false
  - EnableMemPrealloc bool = false
  - EnableHugePages bool = false
  - EnableIOMMU bool = false
  - EnableIOMMUPlatform bool = false
  - FileBackedMemRootDir string = ""
  - EnableDebug bool = false
  - DisableNestingChecks bool = false
  - Msize9p uint32 = 8192
  - HotplugVFIOOnRootBus bool = false
  - PCIeRootPort = 0

- EntropySource = "/dev/urandom"
- GuestHookPath string = ""
- VirtioFSCacheMode = "none"
- DisableImageNvdimm = false
- ReadOnlyRootfs = true
- VhostUserStorePath string = "/var/run/kata-containers/vhost-user/"
- RxRateLimiterMaxRate = uint64(0)
- TxRateLimiterMaxRate = uint64(0)
- ConfidentialGuest = false
- GuestSwap = false
- RootlessHypervisor = false
- DisableSeccomp = false
- VfioMode = "guest-kernel"
- LegacySerial = false
- SGXEPCSize = int64(0)
- TemplatePath string = "/run/vc/vm/template"
- VMCacheEndpoint string = "/var/run/kata-containers/cache.sock"

hypervisor

- docs/design/virtualization.md
  - 描述当前支持的虚拟化类型
- 配置 VMM 的信息，因为支持多种VMM，所以是字典格式，常定义 [hypervisor.qemu]
- path： vmm程序地址，如 /usr/libexec/qemu-kvm
- kernel：kernel 地址，如 /usr/share/ecr/vmlinuz.container
- initrd：以 initrd格式的rootfs
- image：以image格式的rootfs，推荐格式
- firmware：固件地址
- firmware_volume：固件卷信息
- machine_accelerators：加速配置，如
- cpu_features：cpu特性，如 pmu=off,vmx=off
- kernel_params：内核启动参数
- machine_type：机器类型，通常x86使用 q35， arm 使用virtua
- block_device_driver：块设备驱动类型，支持SCSI(默认), Block, Mmio, Nvdimm, BlockCCW
- entropy_source
- shared_fs：共享文件驱动，通常是 virtio-fs

- block_device_cache_direct, 布尔值, 表示是否启用O_DIRECT（绕过主机页面缓存）,
- block_device_cache_noflush, 布尔值, 表示是否忽略设备的刷新请求,
- block_device_cache_set, 布尔值, 表示是否将缓存相关选项设置为块设备,
- block_device_driver, 字符串, 指定块设备的驱动程序, 有效值为virtio-blk、virtio-scsi、nvdimm
- cpu_features, 字符串, 传递给CPU（QEMU）的逗号分隔的CPU特性列表,
- ctlpath, 字符串, ACRN管理程序的acrnctl二进制文件的路径,
- confidential_guest
- default_bridges
- default_max_vcpus, 无符号32位整数, 为虚拟机分配的最大vCPU数量,
- default_memory, 无符号32位整数, 为虚拟机分配的内存（以MiB为单位）,
- default_vcpus, 无符号32位整数, 为虚拟机分配的默认vCPU数量,
- disable_block_device_use, 布尔值, 禁止使用块设备,
- disable_image_nvdimm, 布尔值, 是否支持使用nvdimm设备作为根文件系统
- disable_vhost_net, 布尔值, 指定主机上是否不可用vhost-net,
- disable_nesting_checks
- disable_seccomp
- disable_selinux
- disable_vhost_net
- disk_rate_limiter_bw_max_rate
- disk_rate_limiter_bw_one_time_burst
- disk_rate_limiter_ops_max_rate
- disk_rate_limiter_ops_one_time_burst
- enable_annotations
- enable_debug
- enable_guest_swap
- enable_hugepages
- enable_iommu, 布尔值, 在Q35上启用iommu（QEMU x86_64）
- enable_iommu_platform, 布尔值, 在CCW设备上启用iommu（QEMU s390x）
- enable_iothreads, 布尔值, 启用IO在单独的线程中处理。目前支持virtio-scsi驱动程序
- enable_mem_prealloc, 布尔值, 管理程序用于nvdimm设备的内存空间,
- enable_vhost_user_store, 布尔值, 启用vhost-user存储设备（QEMU）
- enable_virtio_mem, 布尔值, 启用virtio-mem（QEMU）
- file_mem_backend 字符串, 基于文件的内存后端根目录,
- guest_hook_path, 字符串, 虚拟机内执行钩子的路径,
- guest_memory_dump_paging

- guest_memory_dump_path
- hotplug_vfio_on_root_bus, 布尔值, 指示设备是否需要在根总线上热插拔而不是桥接,
- memory_offset, 无符号64位整数, 管理程序用于nvdimm设备的内存空间
- memory_slots, 无符号32位整数, 管理程序为虚拟机分配的内存槽,
- msize_9p, 无符号32位整数, 9p共享的msize,
- net_rate_limiter_bw_max_rate
- net_rate_limiter_bw_one_time_burst
- net_rate_limiter_ops_max_rate
- net_rate_limiter_ops_one_time_burst
- pcie_root_port, 指定PCIe Root Port设备的数量。该设备数等于热插拔PCIe设备
- pflashes: 列表类型，针对 arm 架构添加的启动参数
- read_only_rootfs
- rootless
- rx_rate_limiter_max_rate
- seccompsandbox
- tx_rate_limiter_max_rate
- use_legacy_serial
- valid_ctlpaths
- valid_entropy_sources = ["/dev/urandom","/dev/random",""]
- valid_file_mem_backends = [""]
- valid_hypervisor_paths
- valid_jailer_paths
- valid_vhost_user_store_paths = ["/var/run/kata-containers/vhost-user"]
- valid_virtio_fs_daemon_paths = ["/usr/libexec/virtiofsd"]
- vhost_user_store_path, 字符串, 指定vhost-user设备相关文件夹、套接字和设备节点的目录路径（QEMU）, 默认 "/var/run/kata-containers/vhost-user"
- virtio_fs_cache, 字符串, virtio-fs的缓存模式, 有效值为always、auto(默认)和none,
- virtio_fs_cache_size, 无符号32位整数, virtio-fs DAX缓存大小, 以MiB为单位, 0 禁止缓存
- virtio_fs_daemon, 字符串, virtio-fs守护进程路径,
- virtio_fs_extra_args, 字符串, 传递给virtiofs守护进程的额外选项, 如["--thread-pool-size=1"]

agent

- agent 是配置 guest 内 agent 行为，当前该配置是字典类型，常定义 [agent.kata]
- kernel_modules：在启动容器前加载的内核模块，如 ["e1000e", "i915"]
- enable_debug：将 agent 日志调整为 debug 格式

- enable_tracing：默认 disabled
- debug_console_enabled： 通过工具 ecr-runtime exec 可进入 guest 内
- dial_timeout：连接 anget 超时时间

runtime

- 主要是配置 shim-v2 的操作
- internetworking_model：决定VM应如何连接到容器网络, 有效值有macvtap、tcfilter(默认)
- vfio_mode：vfio工作模式，有效值有vfio，guest-kernel(默认)
- sandbox_bind_mounts：字符串列表
- enable_debug：将 runtime 日志打出到宿主机 system log
- disable_new_netns： 决定是否为虚拟机管理程序进程创建网络命名空间
- sandbox_cgroup_only：布尔型，true时将 qemu 和 virtiofsd 放入 cgroup 中管理
- static_sandbox_resource_mgmt: 不进行 cpu/mem 热加载操作
- enable_pprof
- disable_guest_seccomp：
- disable_guest_empty_dir：针对 emptyDir 卷的配置
- jaeger_endpoint
- jaeger_user
- jaeger_password
- enable_sriov：自己加入

factory

- 配置通过模板方式启动时的元数据
- template_path
- vm_cache_endpoint
- vm_cache_number
- enable_template

配置例子

```
# arm64 例子
[hypervisor.qemu]
path = "/opt/ecr/bin/qemu-system-aarch64"
kernel = "/usr/share/ecr/vmlinuz.container"
image = "/usr/share/ecr/ecr-containers.img"
machine_type = "virt"
enable_annotations =
["default_vcpus","guest_hook_path","kernel","kernel_params","image","scsi_sc
```

```
an_mod","enable_iommu","enable_sriov","hotplug_vfio_on_root_bus","pcie_root_
port","sandbox_cgroup_only"]
valid_hypervisor_paths = ["/usr/libexec/qemu-kvm", "/opt/ecr/bin/qemu-
system-aarch64"]
kernel_params = "agent.debug_console agent.debug_console_vport=1026
agent.log=debug"
firmware = ""
firmware_volume = ""
machine_accelerators=""
cpu_features="pmu=off"
default_vcpus = 1
default_maxvcpus = 0
default_bridges = 1
default_memory = 2048
default_maxmemory = 0
disable_block_device_use = false
shared_fs = "virtio-fs"
virtio_fs_daemon = "/usr/libexec/virtiofsd"
valid_virtio_fs_daemon_paths = ["/usr/libexec/virtiofsd"]
virtio_fs_cache_size = 0
virtio_fs_extra_args = ["--thread-pool-size=1", "-o", "announce_submounts"]
virtio_fs_cache = "auto"
block_device_driver = "virtio-scsi"
enable_iothreads = false
enable_vhost_user_store = false
vhost_user_store_path = "/var/run/kata-containers/vhost-user"
valid_vhost_user_store_paths = ["/var/run/kata-containers/vhost-user"]
valid_file_mem_backends = [""]
pflashes = ["/usr/share/ecr/ecr-flash0.img", "/usr/share/ecr/ecr-
flash1.img"]
enable_debug = true
valid_entropy_sources = ["/dev/urandom","/dev/random"]
disable_selinux=false
[factory]
[agent.kata]
enable_debug = true
kernel_modules=[]
debug_console_enabled = true
[runtime]
enable_debug = true
internetworking_model="tcfilter"
disable_guest_seccomp=true
sandbox_cgroup_only=true
```

```
static_sandbox_resource_mgmt=false
sandbox_bind_mounts=[]
vfio_mode="guest-kernel"
disable_guest_empty_dir=false
enable_sriov=false
experimental=[]
```

启动例子

```
# qemu 启动虚拟机例子
/opt/ecr/bin/qemu-system-aarch64
-name sandbox-
be61181a717cd301631953a2ca74e93552079bd2f2663c2000b3b16b21054cf3
-uuid 86d1a4a6-94f8-4100-90a2-6060925af5d1
-machine virt,usb=off,accel=kvm,gic-version=host,nvdimm=on
-cpu host,pmu=off
-pidfile
/run/vc/vm/be61181a717cd301631953a2ca74e93552079bd2f2663c2000b3b16b21054cf3/
pid
-smp 5,cores=1,threads=1,sockets=96,maxcpus=96
-qmp
unix:/run/vc/vm/be61181a717cd301631953a2ca74e93552079bd2f2663c2000b3b16b2105
4cf3/qmp.sock,server=on,wait=off
-m 2048M,slots=10,maxmem=522535M
-device pci-bridge,bus=pcie.0,id=pci-bridge-
0,chassis_nr=1,shpc=off,addr=2,io-reserve=4k,mem-reserve=1m,pref64-
reserve=1m
-device virtio-serial-pci,disable-modern=false,id=serial0
-device virtconsole,chardev=charconsole0,id=console0
-chardev
socket,id=charconsole0,path=/run/vc/vm/be61181a717cd301631953a2ca74e93552079
bd2f2663c2000b3b16b21054cf3/console.sock,server=on,wait=off
-device nvdimm,id=nv0,memdev=mem0
-object memory-backend-file,id=mem0,mem-path=/usr/share/ecr/ecr-
containers.img,size=402653184
-object memory-backend-file,id=dimm1,size=2048M,mem-path=/dev/shm,share=on
-device virtio-scsi-pci,id=scsi0,disable-modern=false
-object rng-random,id=rng0,filename=/dev/urandom
-device virtio-rng-pci,rng=rng0
-device vhost-vsock-pci,disable-modern=false,vhostfd=3,id=vsock-
4075435650,guest-cid=4075435650
-chardev socket,id=char-
0981952578d2f9f4,path=/run/vc/vm/be61181a717cd301631953a2ca74e93552079bd2f26
63c2000b3b16b21054cf3/vhost-fs.sock
```

```
-device vhost-user-fs-pci,chardev=char-0981952578d2f9f4,tag=ecrShared
-netdev tap,id=network-0,vhost=on,vhostfds=4:5:6:7:8,fds=9:10:11:12:13
-device driver=virtio-net-pci,netdev=network-
0,mac=f6:0a:c8:66:36:f3,disable-modern=false,mq=on,vectors=12
-rtc base=utc,driftfix=slew,clock=host
-global kvm-pit.lost_tick_policy=discard
-pflash /usr/share/ecr/ecr-flash0.img
-pflash /usr/share/ecr/ecr-flash1.img
-vga none
-no-user-config -nodefaults
-nographic
--no-reboot
-daemonize
-numa node,memdev=dimm1
-kernel /usr/share/ecr/vmlinuz.container
-append
iommu.passthrough=0
root=/dev/pmem0p1
rootflags=dax,data=ordered,errors=remount-ro
rw
rootfstype=ext4
console=hvc0
console=hvc1
quiet
systemd.show_status=false
panic=1
nr_cpus=96
systemd.unit=kata-containers.target
systemd.mask=systemd-networkd.service
systemd.mask=systemd-networkd.socket
scsi_mod.scan=none
agent.debug_console
agent.debug_console_vport=1026
```

注解

- 在pod注解中配置，要求 containerd.toml 中设置pod_annotations 白名单

global

- io.katacontainers.config_path ，字符串类型，配置文件地址
- io.katacontainers.pkg.oci.bundle_path ，字符串，oci 目录地址
- io.katacontainers.pkg.oci.container_type ，字符串，容器类型. pod_container 或 pod_sandbox

runtime

- io.katacontainers.config.runtime.experimental ，布尔型，表示是否开启专业特性
- io.katacontainers.config.runtime.disable_guest_seccomp，布尔型，决定是否在 guest 内部应用 seccomp
- io.katacontainers.config.runtime.disable_new_netns，布尔型，决定是否为虚拟机管理程序创建一个新的网络命名空间（netns）
- io.katacontainers.config.runtime.internetworking_model，字符串类型，决定VM应如何连接到容器网络接口。有效值有macvtap、tcfilter和none，
- io.katacontainers.config.runtime.sandbox_cgroup_only，布尔型，决定Kata进程是否仅在沙箱cgroup中管理
- io.katacontainers.config.runtime.enable_pprof，布尔型，为containerd-shim-kata-v2进程启用Golang的pprof性能分析工具

agent

- io.katacontainers.config.agent.enable_tracing，布尔值，启用代理跟踪
- io.katacontainers.config.agent.container_pipe_size，无符号32位整数，指定为容器创建的标准输入输出管道的大小，
- io.katacontainers.config.agent.kernel_modules，字符串，将被加载的内核模块。使用分号分隔，例如 e1000e InterruptThrottleRate=3000,3000,3000 EEE=1; i915 enable_ppgtt=0
- io.katacontainers.config.agent.trace_mode，字符串，代理的跟踪模式，
- io.katacontainers.config.agent.trace_type，字符串，代理的跟踪类型。

hypervisor

- 以下注解都有前缀 io.katacontainers.config.
- hypervisor.asset_hash_type，字符串，用于资产验证的哈希类型，默认是sha512，
- hypervisor.block_device_cache_direct，布尔值，表示是否启用O_DIRECT（绕过主机页面缓存），
- hypervisor.block_device_cache_noflush，布尔值，表示是否忽略设备的刷新请求，
- hypervisor.block_device_cache_set，布尔值，表示是否将缓存相关选项设置为块设备，
- hypervisor.block_device_driver，字符串，指定块设备的驱动程序，有效值为virtio-blk、virtio-scsi、nvdimm
- hypervisor.cpu_features，字符串，传递给CPU（QEMU）的逗号分隔的CPU特性列表，
- hypervisor.ctlpath，字符串，ACRN管理程序的acrnctl二进制文件的路径，
- hypervisor.default_max_vcpus，无符号32位整数，为虚拟机分配的最大vCPU数量，
- hypervisor.default_memory，无符号32位整数，为虚拟机分配的内存（以MiB为单位），
- hypervisor.default_vcpus，无符号32位整数，为虚拟机分配的默认vCPU数量，
- hypervisor.disable_block_device_use，布尔值，禁止使用块设备，

- hypervisor.disable_image_nvdimm，布尔值，是否支持使用nvdimm设备作为根文件系统
- hypervisor.disable_vhost_net，布尔值，指定主机上是否不可用vhost-net，
- hypervisor.enable_hugepages，布尔值，表示是否应从大页面预分配内存，
- hypervisor.enable_iommu_platform，布尔值，在CCW设备上启用iommu（QEMU s390x）
- hypervisor.enable_iommu，布尔值，在Q35上启用iommu（QEMU x86_64）
- hypervisor.enable_iothreads，布尔值，启用IO在单独的线程中处理。目前支持virtio-scsi驱动程序
- hypervisor.enable_mem_prealloc，布尔值，管理程序用于nvdimm设备的内存空间，
- hypervisor.enable_swap，布尔值，启用虚拟机内存交换，
- hypervisor.enable_vhost_user_store，布尔值，启用vhost-user存储设备（QEMU）
- hypervisor.enable_virtio_mem，布尔值，启用virtio-mem（QEMU）
- hypervisor.entropy_source (R)，字符串，主机熵源的路径（/dev/random或真实RNG设备）
- hypervisor.file_mem_backend (R)，字符串，基于文件的内存后端根目录，
- hypervisor.firmware_hash，字符串，容器固件的SHA-512哈希值，
- hypervisor.firmware，字符串，将在容器虚拟机中运行的 guest 固件，
- hypervisor.guest_hook_path，字符串，虚拟机内执行钩子的路径，
- hypervisor.hotplug_vfio_on_root_bus，布尔值，指示设备是否需要在根总线上热插拔而不是桥接，
- hypervisor.hypervisor_hash，字符串，容器管理程序二进制文件的SHA-512哈希值，
- hypervisor.image_hash，字符串，容器 guest 镜像的SHA-512哈希值，
- hypervisor.image，字符串，将在容器虚拟机中运行的 guest 镜像，
- hypervisor.initrd_hash，字符串，容器 guest initrd的SHA-512哈希值，
- hypervisor.initrd，字符串，将在容器虚拟机中运行的 guest initrd镜像，
- hypervisor.jailer_hash，字符串，容器jailer的SHA-512哈希值，
- hypervisor.jailer_path (R)，字符串，将约束容器虚拟机的jailer，
- hypervisor.kernel_hash，字符串，容器内核镜像的SHA-512哈希值，
- hypervisor.kernel_params，字符串，额外的 guest 内核参数，
- hypervisor.kernel，字符串，用于启动容器虚拟机的内核，
- hypervisor.machine_accelerators，字符串，管理程序的机器特定加速器，
- hypervisor.machine_type，字符串，管理程序模拟的机器类型，
- hypervisor.memory_offset，无符号64位整数，管理程序用于nvdimm设备的内存空间
- hypervisor.memory_slots，无符号32位整数，管理程序为虚拟机分配的内存槽，
- hypervisor.msize_9p，无符号32位整数，9p共享的msize，
- hypervisor.path，字符串，将运行容器虚拟机的管理程序，
- hypervisor.pcie_root_port，指定PCIe Root Port设备的数量。该设备数等于热插拔PCIe设备
- hypervisor.shared_fs，字符串，共享文件系统类型，可以是virtio-9p或virtio-fs，
- hypervisor.use_vsock，布尔值，指定是否使用vsock进行代理通信，

- hypervisor.vhost_user_store_path，字符串，指定vhost-user设备相关文件夹、套接字和设备节点的目录路径（QEMU），
- hypervisor.virtio_fs_cache_size，无符号32位整数，virtio-fs DAX缓存大小（以MiB为单位），
- hypervisor.virtio_fs_cache，字符串，virtio-fs的缓存模式，有效值为always、auto和none，
- hypervisor.virtio_fs_daemon，字符串，virtio-fs，vhost-user守护进程路径，
- hypervisor.virtio_fs_extra_args，字符串，传递给virtiofs守护进程的额外选项

代码

- src/runtime/cmd/
  - containerd-shim-kata-v2: 主要shim实现
  - kata-monitor
  - kata-runtime：调试工具

shim

- 在 runtimeclass 中定义 handler 对应 containerd 中的配置
  - containerd 中 runtime_type 是用于定位二进制文件名称，当前为 containerd-shim-ecr-v2

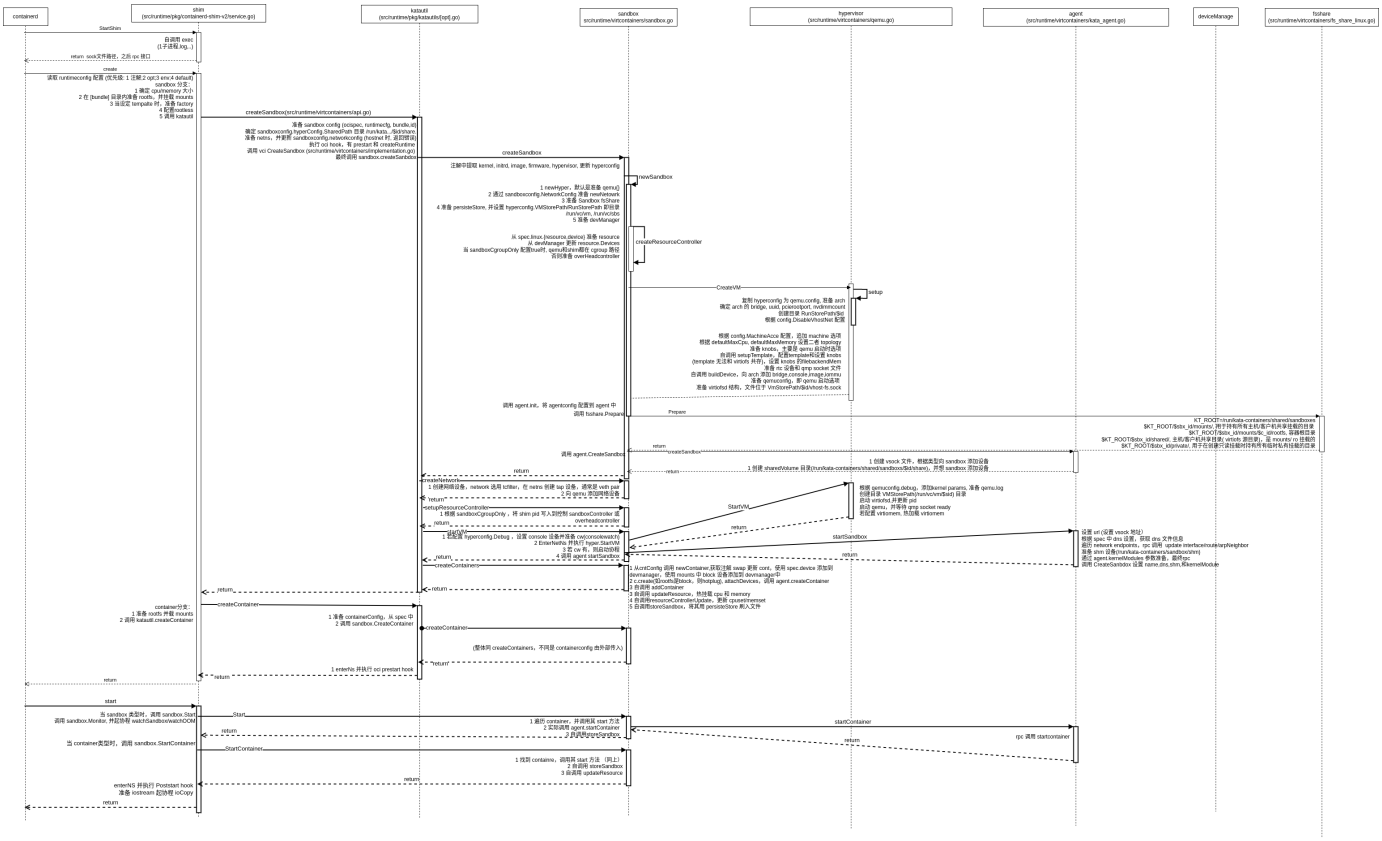
```
# runtimeclass 资源
apiVersion: node.k8s.io/v1
kind: RuntimeClass
metadata:
  name: rune
handler: ecr
overhead:
  podFixed:
    cpu: 50m
    memory: 200Mi
scheduling:
  nodeSelector:
    secure-container: enabled

# containerd 配置
    [plugins."io.containerd.grpc.v1.cri".containerd.runtimes.ecr]
     runtime_type = "io.containerd.ecr.v2"
     privileged_without_host_devices = true // 特权容器也不映射设备
     pod_annotations = ["io.katacontainers.*"] // 将匹配的注解透传到 shim
     [plugins.cri.containerd.runtimes.kata.options]
       ConfigPath = "/etc/ecr/configuration.toml"

# 进程模型
```

```
[pid1]     1  /usr/bin/containerd-shim-ecr-v2 -namespace k8s.io -address
/var/run/containerd/containerd.sock -publish-binary /usr/bin/containerd -id
{sid}
[pid2] [pid1] /usr/libexec/virtiofsd --syslog -o cache=auto -o no_posix_lock
-o source=/run/kata-containers/shared/sandboxes/{sid}/shared --fd=3 -f --
thread-pool-size=1 -o announce_submounts
[pid3]     1  /opt/ecr/bin/qemu-system-aarch64 -name sandbox-{sid}
```

目录

- KT_ROOT=/run/kata-containers/shared/sandboxes
  - KT_ROOT/sid/mounts, 持有所有主机/客户机共享的目录，包括volume等
    - KT_ROOT/sid/mounts/cid/rootfs, 容器根目录
  - KT_ROOT/sid/shared/, 主机/客户机共享目录(virtiofs 源目录)，bind 从 sid/mounts (ro) 挂载
  - KT_ROOT/sid/private/, 用于在创建只读挂载时持有所有临时私有挂载的目录
- RUN=/run/vc/
  - RUN/vm/ 记录多个 socket，包括有 vhost-fs，qmp，console。
  - RUN/sbs/sid 记录 pod 设备，卷，dns配置信息
- G_ROOT=/run/kata-containers 虚拟机内目录
  - G_ROOT/cid/rootfs, 虚拟机内的容器根目录

启动



- rpc 请求实现位于 src/runtime/pkg/containerd-shim-v2/service.go，这里是入口调用
  - 加载配置文件，优先级顺序为 annotation > option > env > default
    - 当前default 位置是 /etc/ecr/con..toml (src/runtime/pkg/katautils/config-settings.go.in)
    - 具体可以配置内容见 docs/how-to/how-to-set-sandbox-config-kata.md
  - 准备及 mount 到 rootfs
  - 对于sandbox 和 container 会在此分流，并执行 katautil
- katautil 位于 src/runtime/pkg/katautils/[opt]，是全局函数，和 container 有关的内容
  - 执行 oci hook：prestart 和 createRuntime
  - check 必须用容器网络
  - 调用接口 vci (src/runtime/virtcontainers/implementation.go)
- sandbox 实现位于 src/runtime/virtcontainers/sandbox.go，管理所有 hyper,agent 等等
  - 准备后端 hypervisor，通常是 qemu
  - 准备 fsshare 管理器，即目录 KT_ROOT
  - 准备 PersistDriver 管理， 即目录 /run/vc 下内容
  - 准备 device 管理，是 qemu 块设备驱动类型设置
  - 准备 agent 管理，主要用于准备 vsock
- 从 计算/存储/网络 分别介绍
  - 网络部分在虚拟机内是容器共用的模式，虚拟机外禁止使用主机网络
  - 存储包括 guest rootfs， container rootfs，

计算

- 遵循 oci spec 内容来分析
- hook 当前执行内容均在 katautil 中，是 qemu 外执行
  - prestart，createRuntime 在启动 qemu 前执行
  - postStart / postStop 在启动 container 之后执行
- 因为没有在 qemu 中执行hook方式，因此 kata 设置 guest_hook_path 来在 guest 中执行
  - 执行时机是在 startVM 之后，即启动 container 之前
- linux 类型，主要有namespace，devices, xxPaths, resource{cpu/mem/device}
  - xxPath 包括三种类型 cgroupPath，maskPath，readonlyPath
  - linux.devices：容器可见性控制
  - linux.namespace： 新建或继承已有的命名空间控制
  - resource 包括 device(容器可访问) cgroup

- cgroupPath：主要是控制 container 资源使用，但 kata 现在有 qemu，virtiofsd 额外进程，如此限制有些不公平，因此除 runtimeClass.overhead 还有 sandbox_cgroup_only 配置
  - 和 guest 不同的 ns 有 ipc,mnt,pid,uts，说明下
    - IPC 隔离Posix消息队列
    - MNT 隔离文件根挂载点
    - PID 隔离进程号，如此可以使用相同的进程id
    - UTS 隔离主机名和域名
- qemu 启动时需配置 cpu，memory，当前启动使用默认值，容器的资源则通过热加载完成，这类资源热加载需要 qemu 和 guest kernel 支持
  - 参考 docs/how-to/how-to-hotplug-memory-arm64.md，docs/design/vcpu-handling-runtime-go.md，how-to-use-virtio-mem-with-kata.md ，根据以上文件 vcpus 热加载为 ceiling(quota / period)
  - 问题 pull#47, EAS-58498
  - 可以配置 static_sandbox_resource_mgmt 跳过 hotplug 配置
  - 配置项
    - default_vcpus = 1 默认方式，会根据container 重新计算
    - default_maxvcpus = 0 最大只，为0则不限制
    - default_memory = 2048 默认值，
    - default_maxmemory = 0 最大值，为0则不限制
- 设备管理包括 vfio(直通) 和 块设备，而字符设备通过 virtiofsd 处理
  - vfio 设备主要是 pci 设备在虚拟机内驱动类型，通过 hotplug 加入，有关配置
    - vfio_mode(guest_kernel, vfio 之一)
    - hotplug_vfio_on_root_bus
  - 字符设备 IO 和块设备对比即其他问题，涉及cache和driver类型，参考 EAS-111745, EAS-68959

调试

- 打开配置文件 /etc/ecr/configure.toml 中 debug; journalctl -t kata
- 打开 containerd debug ，设置debug.level = debug

runtime

- 使用 ecr-runtime 可以直接进入虚拟机内操作，以前则需要通过 agnet-ctl ，如下
- 启动时配置vsock
  - -device vhost-vsock-pci,disable-modern=false,vhostfd=3,id=vsock-774180885,guest-cid=774180885
- 使用 ss 可以查询主机上已有的 vsock

    - ss --vsock
- 执行查询命令
    - kata-agent-ctl connect -n --bundle-dir {bundle} --server-address "vsock://{gustid}:1024" -c ListInterfaces
    - 其中 bundle 是 sandboxID；guestid 是 ss 查询到的 Peer ID