基于StratoVirt的——
安全容器系统构建与技术应用实践

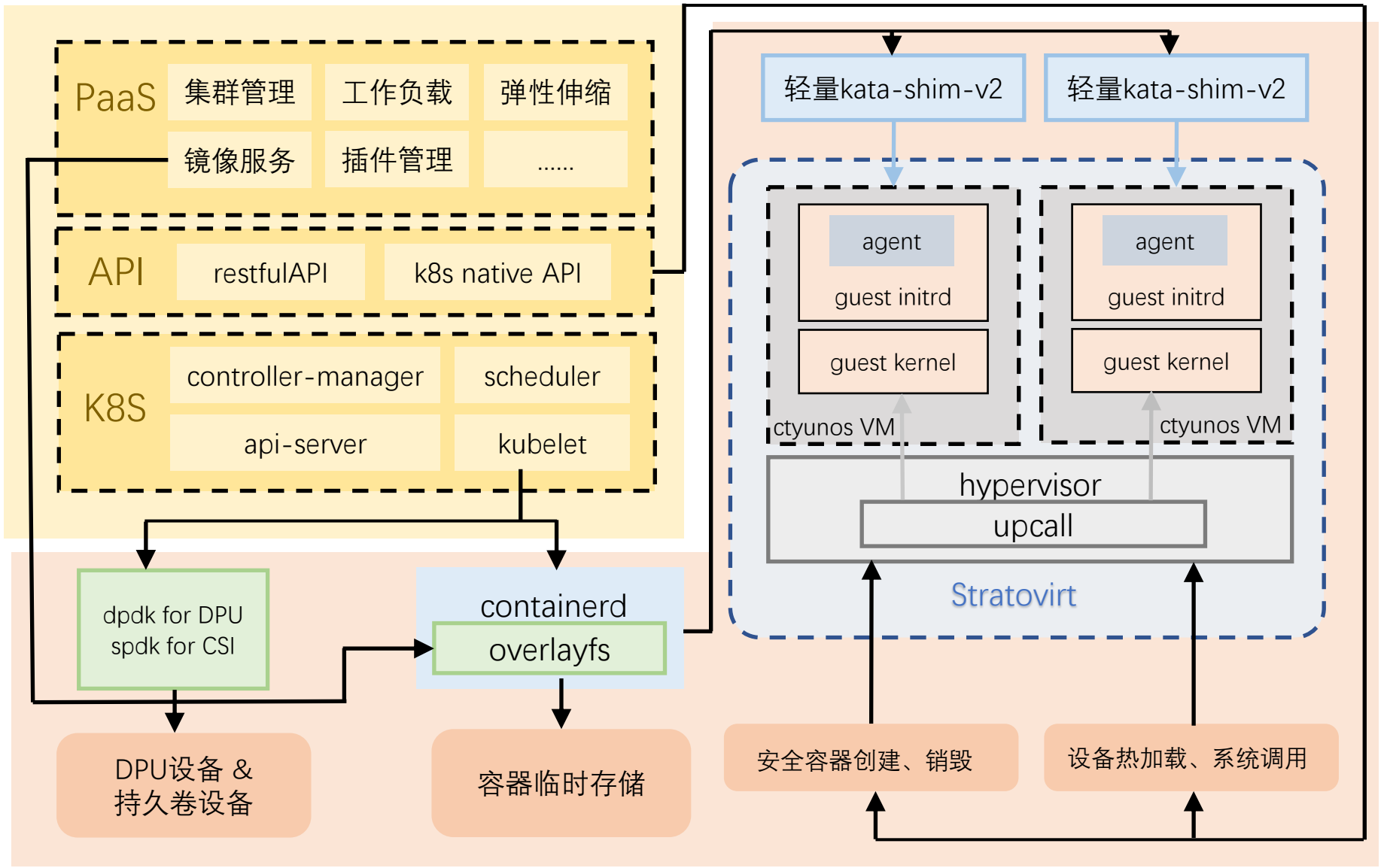
王　麟
白耀伟

天翼云 China Telecom Cloud

开放原子开源基金会 OPENATOM FOUNDATION | OpenEuler

系统总体架构

- 新增upcall
- 定制的hypervisor
- ctyunos microVM
- 定制kata-shim
- DPU和持续卷使能
- k8s-api开发
- PaaS业务适配



CTyunos microVM制作

ctyunos microVM制作

内核编译

```
wget https://                                                    /Packages/kernel-5.10.0-136.12.0.86.ctl3.src.rpm
rpm -ivh kernel-5.10.0-136.12.0.86.ctl3.src.rpm
tar -xf rpmbuild/SOURCES/kernel.tar.gz -C ~/.
```

将config文件(kata社区官方config文件)放到源码目录下并命名为.config，执行make olddefconfig：

```
mv config .config
make olddefconfig
          64

# 编译
make -j

# 编译完成后，生成stratovirt支持的vminux-bin文件
objcopy -O binary vmlinux vmlinux.bin
```

ctyunos microVM制作

ctyunos microVM kernel精简分析

对比kata默认kernel选项与标准ctyunos3 kernel选项，精简的ctyunos microVM kernel禁用了许多配置，例如：

1. 禁用与Xen虚拟化平台相关的Linux内核配置选项
2. 禁用Linux内核中的一些性能事件监控相关的选项。
3. 禁用与非易失性内存（NVDIMM）和系统健康检查相关的 Linux 内核配置选项。
4. 禁用许多TCP拥塞控制算法以及Linux内核中的 IPv6 协议相关功能。
5. 禁用NVMe块设备支持
6. 禁用 Linux 内核的电源管理和系统休眠的支持。
7. 禁用了与 ACPI 相关的多个功能，其中包括 ACPI 电源适配器、电池、风扇、可插拔（Dock）等。
8. 禁用Linux 内核中的 kexec（kernel exec）相关。
9. 禁用内存随机化（ASLR）特性。
10. 禁用了linux 内核中的 PCI（Peripheral Component Interconnect）子系统和相关功能配置。

ctyunos microVM制作

initrd制作

参考仓库：https://gitee.com/src-openeuler/kata_integration

```
# create a temp dir to store rootfs
rm -rf ${ROOTFS_DIR}
mkdir -p ${ROOTFS_DIR}/lib \
      ${ROOTFS_DIR}/lib64 \
      ${ROOTFS_DIR}/lib/modules

mkdir -m 0755 -p ${ROOTFS_DIR}/dev \
      ${ROOTFS_DIR}/sys \
      ${ROOTFS_DIR}/sbin \
      ${ROOTFS_DIR}/bin \
      ${ROOTFS_DIR}/tmp \
      ${ROOTFS_DIR}/proc

if [ ! -f "${BUILD_PATH}/kata-agent" ];then
    echo "kata-agent doesn't exist!"
    exit 1
fi

# busybox
cp /sbin/busybox ${ROOTFS_DIR}/sbin/
cp ${BUILD_PATH}/kata-agent ${ROOTFS_DIR}/init
# ipvs
cp /usr/sbin/ipvsadm ${ROOTFS_DIR}/sbin
# conntrack-tools
cp /usr/sbin/conntrack ${ROOTFS_DIR}/sbin
# quota
cp /usr/bin/quota* ${ROOTFS_DIR}/bin
cp /usr/bin/quotasync ${ROOTFS_DIR}/bin
# glibc-devel glibc
cp /lib64/libnss_dns* ${ROOTFS_DIR}/lib64
cp /lib64/libnss_files* ${ROOTFS_DIR}/lib64
```

```
# cp run request files in initrd
cat $rpmlist | while read rpm
do
    if [ "${rpm:0:1}" != "#" ]; then
        rpm -ql $rpm > /dev/null 2>&1
        if [ $? -ne 0 ]; then
            continue
        fi
        array=($(rpm -ql $rpm| grep -v "share" | grep -v ".build-id"))
        for file in ${array[@]};
        do
            source=$file
            dts_file=${ROOTFS_DIR}$file
            dts_folder=${dts_file%/*}
            if [ ! -d "$dts_folder" ];then
                mkdir -p $dts_folder
            fi
            cp -r -f -d $source $dts_folder
        done
    fi
done
```

```
#create symlinks to busybox
BUSYBOX_BINARIES=(/usr/bin/sh /usr/bin/mount /usr/bin/umount /usr/bin/ls /usr/bin/ps /usr/bin/file
/usr/bin/ldd /usr/bin/tar /usr/bin/hwclock /usr/sbin/modprobe /usr/sbin/depmod /usr/bin/ip
/usr/bin/modinfo /usr/bin/insmod /usr/bin/rmmod /usr/bin/free)
for bin in ${BUSYBOX_BINARIES[@]}
do
    mkdir -p ${ROOTFS_DIR}/`dirname ${bin}`
    ln -sf /sbin/busybox ${ROOTFS_DIR}/${bin}
done

LDD_BINARIES=(/init /sbin/busybox /sbin/conntrack /sbin/ipvsadm)
for bin in ${LDD_BINARIES[@]}
 do
     ldd ${ROOTFS_DIR}${bin} | while read line
     do
        arr=(${line// / })

        for lib in ${arr[@]}
        do
            echo $lib
            if [ "${lib:0:1}" = "/" ]; then
                dir=${ROOTFS_DIR}`dirname $lib`
                mkdir -p "${dir}"
                cp -f $lib $dir
            fi
        done
    done
 done

(cd ${ROOTFS_DIR} && find . | cpio -H newc -o | gzip -9 ) > ${BUILD_PATH}/${IMAGE_NAME}
```

1. 暂时删除了支持gpu和InfiniBand的驱动
2. 修复kata-exec无法进入vm shell问题（创建busybox软链接，/bin/目录修改为/usr/bin）
3. 新增了free top du等命令
4. 精简了make-initrd-rpm.list

ctyunos microVM制作

make-initrd-rpm.list精简分析

tcp_wrappers-libs

有状态连接的特定服务进行安全检测并实现访问控制，凡是包含有libwrap.so库文件的的程序就可以受TCP_Wrappers的安全控制。它的主要功能就是控制谁可以访问，常见的程序有rpcbind、vsftpd、sshd，telnet。

libverto-tevent

提供了异步API接口，允许其他库暴露异步接口给应用程序并来启动或停止应用程序的主循环。

libtirpc

libtirpc是一个RPC库，提供上层应用与NFS客户端和服务端交互的通信服务。

libevent

事件通知库，适用于windows、linux、bsd等多种平台，内部使用select、epoll、kqueue、等系统调用管理事件机制。

libcom_err

处理错误代码和错误消息的库。注：是yum相关的依赖包，随意的更改版本或者删除，会导致yum命令无法使用

libbasicobjects

与nfs部署相关

gssproxy

GSSAPI（通用安全服务 API）是一个符合 RFC 标准的接口，用于希望使用安全库的应用程序。通常情况下，GSSAPI 用作与Kerberos通信的接口，但也提供了其他机制

kata-stratovirt编译安装与安全容器性能指标

kata-stratovirt编译安装与容器启动流程

1. 确保编译环境中已安装go，推荐golang版本：1.19.3

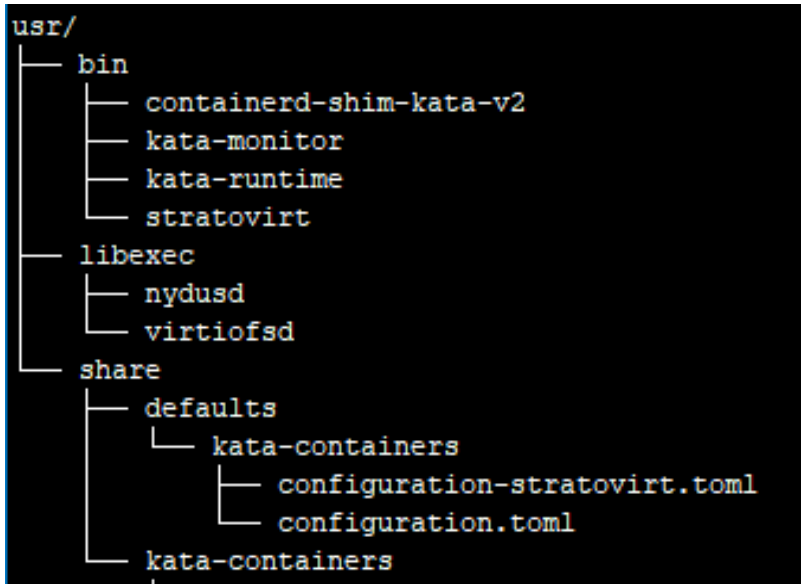
```
wget https://golang.google.cn/dl/go1.19.3.linux-amd64.tar.gz
rm -rf /usr/local/go && tar -C /usr/local -xzf go1.19.3.linux-amd64.tar.gz
vi /etc/profile
export PATH=$PATH:/usr/local/go/bin
source /etc/profile

# 设置代理环境变量
go env -w GOPROXY=https://goproxy.cn,direct
```

2. kata组件编译

```
git clone https://                    /kata-containers.git
cd kata-containers
git checkout 7794/StratoVirt_VMM_support
cd src/runtime
make
```

kata-stratovirt-static 安装后目录结构如下:



注：其余引用kata社区release包

1. 在src/runtime下会编译出containerd-shim-kata-v2, kata-runtime, kata-monitor二进制。
2. 在src/runtime/config下会编译出configuration-stratovirt.toml配置文件。

- 注意：containerd-shim-kata-v2编译环境host内核版本须与最终测试/运行kata容器的host

关于kata-containers的定制后边有详细解析

kata-stratovirt编译安装与容器启动流程

3. 确保编译环境中已安装rust，推荐rust版本：1.72.0

```
# 执行安装文档的下载脚本
curl https://sh.rustup.rs -sSf | sh
export PATH="$PATH:~/.cargo/bin"

# 安装指定版本
rustup install 1.72.0
```

4. stratovirt编译

```
git clone https://gitee.com/openeuler/stratovirt.git
cd stratovirt

# 静态编译stratovirt二进制
cargo build --release --bin stratovirt --target=x86_64-unknown-linux-musl
```

5. 配置configuration-stratovirt.toml

- stratovirt二进制路径
- guest kernel路径
- guest initrd路径

```
[hypervisor.stratovirt]
path = "/opt/kata-stratovirt-static-20.03-v2/usr/bin/stratovirt"
kernel = "/opt/kata-stratovirt-static-20.03-v2/usr/share/kata-containers/vmlinux.container"
#image = "/opt/kata-stratovirt-static-20.03-v2/usr/share/kata-containers/kata-containers.img"
initrd = "/opt/kata-stratovirt-static-20.03-v2/usr/share/kata-containers/kata-containers-initrd.img"
machine_type = "microvm"
```

```
[root@localhost kata-containers]# ll
total 1.1G
-rwxr-xr-x. 1 root root  19M Nov  1 16:43 ctyun-vmlinux.bin
-rw-r--r--. 1 root root 9.5M Nov  2 16:59 kata-alpine-3.15.initrd
lrwxrwxrwx. 1 root root   24 Nov 18 09:59 kata-containers.img -> kata-ubuntu-latest.image
-rw-r--r--. 1 root root 9.7M Nov  2 16:53 kata-containers-initrd_alpine_3.1.0_agent_0f83291926.initrd
lrwxrwxrwx. 1 root root   18 Nov 18 10:00 kata-containers-initrd.img -> kata-ctyun-new.img
-rw-r--r--. 1 root root  25M Nov 18 09:56 kata-ctyun.initrd
-rw-r--r--. 1 root root  12M Nov 18 09:54 kata-ctyun-new.img
-rw-r--r--. 1 root root  12M Nov  2 14:18 kata-openeuler.initrd
-rw-r--r--. 1 root root 739M Nov  2 16:59 kata-static-3.2.0-amd64.tar
-rw-r--r--. 1 1000 1000 256M Sep 18 13:25 kata-ubuntu-latest.image
-rw-r--r--. 1 root root 1.1K Nov 25 11:43 test.sh
-rw-r--r--. 1 1000 1000  22M Sep 18 13:25 vmlinux-bin-6.1.38-112
lrwxrwxrwx. 1 root root   17 Nov 18 09:51 vmlinux.container -> ctyun-vmlinux.bin
```

6. 启动安全容器

```
[secure@gzinf-computer-55e235e17e38 kata-containers]$ sudo time ctr run docker.io/library/busybox:latest test-kata echo "."
.
0.01user 0.02system 0:00.10elapsed 39%CPU (0avgtext+0avgdata 41580maxresident)k
0inputs+0outputs (0major+2460minor)pagefaults 0swaps
```

性能指标

系统占用测试

- 内存/cpu占用（虚机分配内存256M）

VM type	单个容器used内存占用	%Cpu
ctyunos microVM	25M	0.0us 0.0sy
未优化前虚拟机	114M	0.5us 0.3sy

top输出值为0;
无业务时不占用CPU,
符合serverless预期

```
bash-5.0# free -m
              total        used        free      shared  buff/cache   available
Mem:            232          25         167          35          38         166
Swap:             0           0           0
```

```
[root@localhost ~]# systemd-analyze
Startup finished in 996ms (kernel) + 2.600s (initrd) + 7.758s (userspace) = 11.355s
multi-user.target reached after 3.505s in userspace
[root@localhost ~]# free -h
               total        used        free      shared  buff/cache   available
Mem:           455Mi       114Mi       168Mi       2.8Mi       173Mi       326Mi
Swap:          511Mi          0B       511Mi
[root@localhost ~]#
```

- 磁盘使用情况

VM type	磁盘占用
ctyunos microVM	39.7M
未优化前虚拟机	617M

```
bash-5.0# sbin/busybox du -sh /
39.7M    /
```

```
[root@StratoVirt ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       9.8G  617M  8.7G   7% /
devtmpfs        107M     0  107M   0% /dev
tmpfs           110M     0  110M   0% /dev/shm
tmpfs            44M  784K   44M   2% /run
tmpfs           4.0M     0  4.0M   0% /sys/fs/cgroup
tmpfs           110M     0  110M   0% /tmp
```

性能指标

速度测试

- 容器创建时间（ctr启动容器时带-d参数）

VM type	单个容器	100个容器串行	100个容器并发
Stratovirt + ctyunos microVM	100ms	213.4538s	3.3506s

```
[secure@gzinf-computer-55e235e17e38 kata-containers]$ ./test_time
ctr: failed to create shim: Could not bind mount /run/kata-containers/shared/sandboxes/uu9/mounts to /run/kata-containers/shared/sand
boxes/uu9/shared: no such file or directory: unknown
ctr: failed to create shim: Could not bind mount /run/kata-containers/shared/sandboxes/uu38/mounts to /run/kata-containers/shared/san
dboxes/uu38/shared: no such file or directory: unknown
213.4538
```

```
[secure@gzinf-computer-55e235e17e38 kata-containers]$ ./test_time
ctr: failed to create shim: Could not bind mount /run/kata-containers/shared/sandboxes/uu38/mounts to /run/kata-containers/shared/san
dboxes/uu38/shared: no such file or directory: unknown
ctr: failed to create shim: Could not bind mount /run/kata-containers/shared/sandboxes/uu9/mounts to /run/kata-containers/shared/sand
boxes/uu9/shared: no such file or directory: unknown
3.3506
```

- 创建+删除时间（ctr启动容器时带--rm参数）

VM type	单个容器	10个容器串行	10个容器并发
Stratovirt + ctyunos microVM	170ms	20.3166s	1.1034s

注：同时启动多个容器时容易发生失败，须在生产中进行规避

HyperVisor与业务定制

kata-container定制总结

kata-container原仓库地址：https://github.com/kata-containers/kata-containers/releases

kata-stratovirt在kata社区的kata-containers基础上，**新增了Stratovirt作为支持的hypervisor。**具体地：

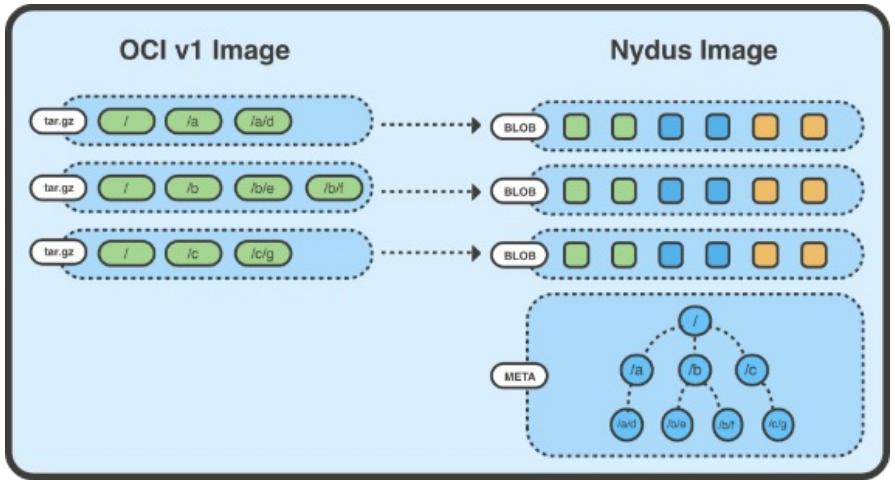
1. src/runtime/virtcontainers中新增stratovirt.go文件：**为 StratoVirt 构建虚拟设备，并生成启动虚拟机时所需的参数，包括磁盘设备、网络设备、文件系统设备等**，这些设备可以用于配置虚拟机的运行环境。这些参数将用于启动 StratoVirt 虚拟机实例，使其能够运行在指定的虚拟化环境中。
2. src/runtime/hypervisor.go文件中**新增stratovirt相关配置。**
3. 在src/runtime/Makefile、src/runtime/arch/arm64-options.mk、src/runtime/arch/amd64-options.mk文件中新增了用于配置和定义与 Stratovirt 相关的一些**参数和选项**
4. 新增了src/runtime/config/configuration-stratovirt.toml.in**配置文件**
5. 在 **kata-deploy**中增加对 StratoVirt 的支持，包括：

- 新增tools/packaging/static-build/stratovirt/build-static-stratovirt.sh文件，该脚本的主要功能是从 Stratovirt 的发布版本中下载预编译的二进制文件，并将其解压到指定目录。
- 新增tools/packaging/kata-deploy/runtimeclasses/kata-stratovirt.yaml文件，这是 Kubernetes 中的一个 YAML 文件，定义了一个名为 kata-stratovirt 的 RuntimeClass。

6. 为 StratoVirt hypervisor 添加测试。

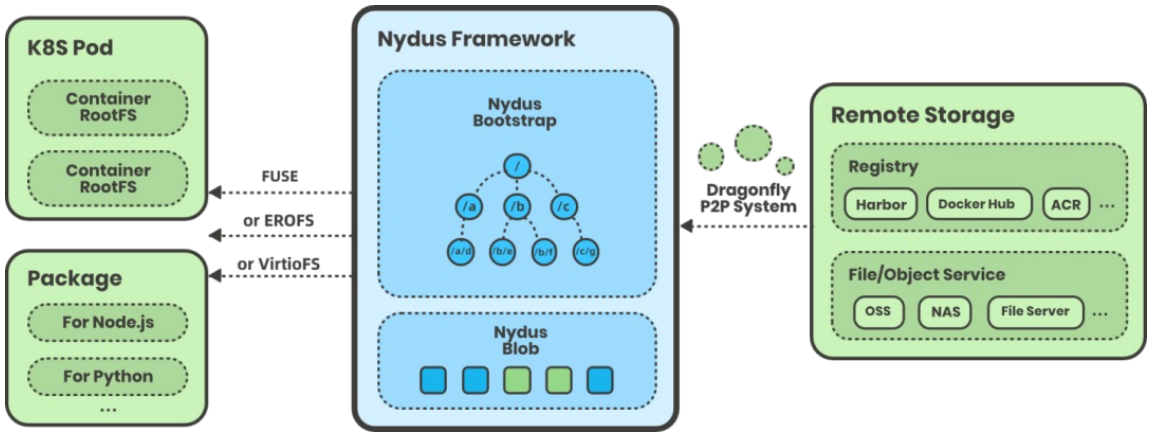
- 新增tests/metrics/cmd/checkmetrics/ci_worker/checkmetrics-json-stratovirt-kata-metric8.toml文件。定义了一系列用于**性能测试的度量标准**
- 新增src/runtime/virtcontainers/stratovirt_test.go文件作为 Stratovirt的**自测试代码**，包括创建 Stratovirt 的配置对象，创建虚拟机功能，启动沙箱功能，清理虚拟机功能，添加虚拟设备的功能等。

支持Nydus的虚机模板



Nydus容器镜像格式

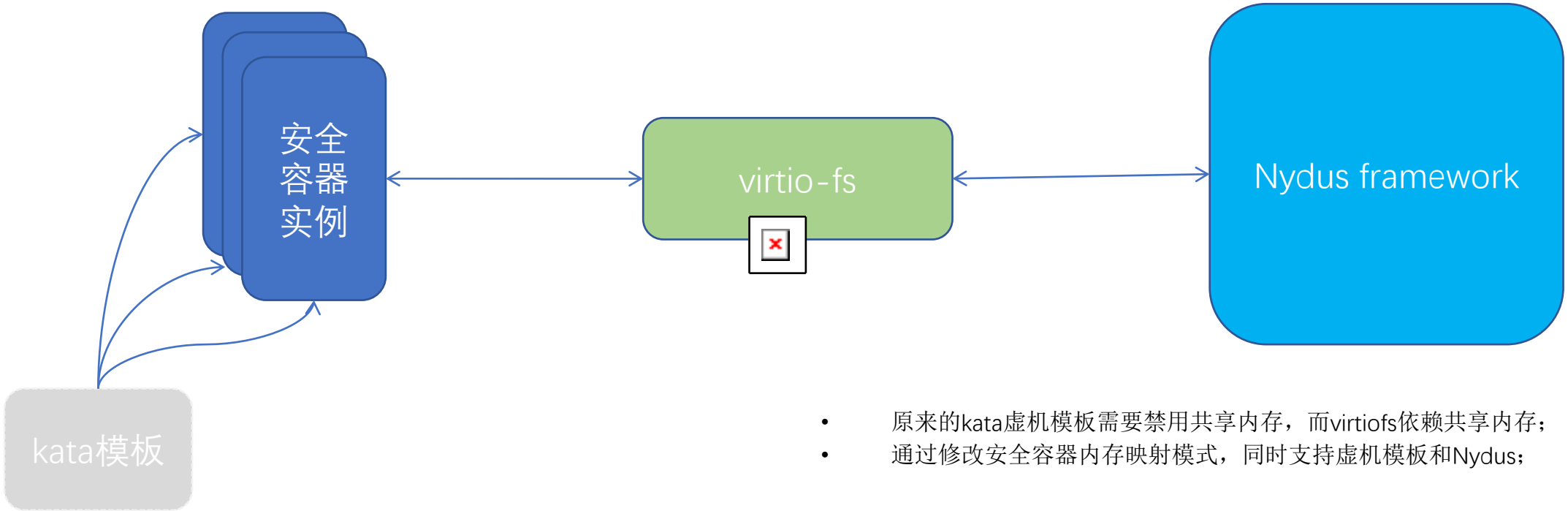
- 元数据和数据分离，用户态按需加载与解压；
- 更细粒度的块级别数据切割与去重；
- 扁平化元数据层（移除中间层），直接呈现文件系统视图；
- 端到端的文件系统元数据树与数据校验；



Nydus架构

- 通过VirtioFS承载FUSE协议，为安全容器提供容器镜像按需加载能力；
- 存储后端可以对接OCI兼容的Registry（例如harbor），也可以对接对象存储、网络文件系统；

支持Nydus的虚机模板



- 原来使用mmap，大文件有一些性能优势
- 现在使用userfaultfd，允许在用户空间实现on-demand paging

- 原来的kata虚机模板需要禁用共享内存，而virtiofs依赖共享内存；
- 通过修改安全容器内存映射模式，同时支持虚机模板和Nydus；