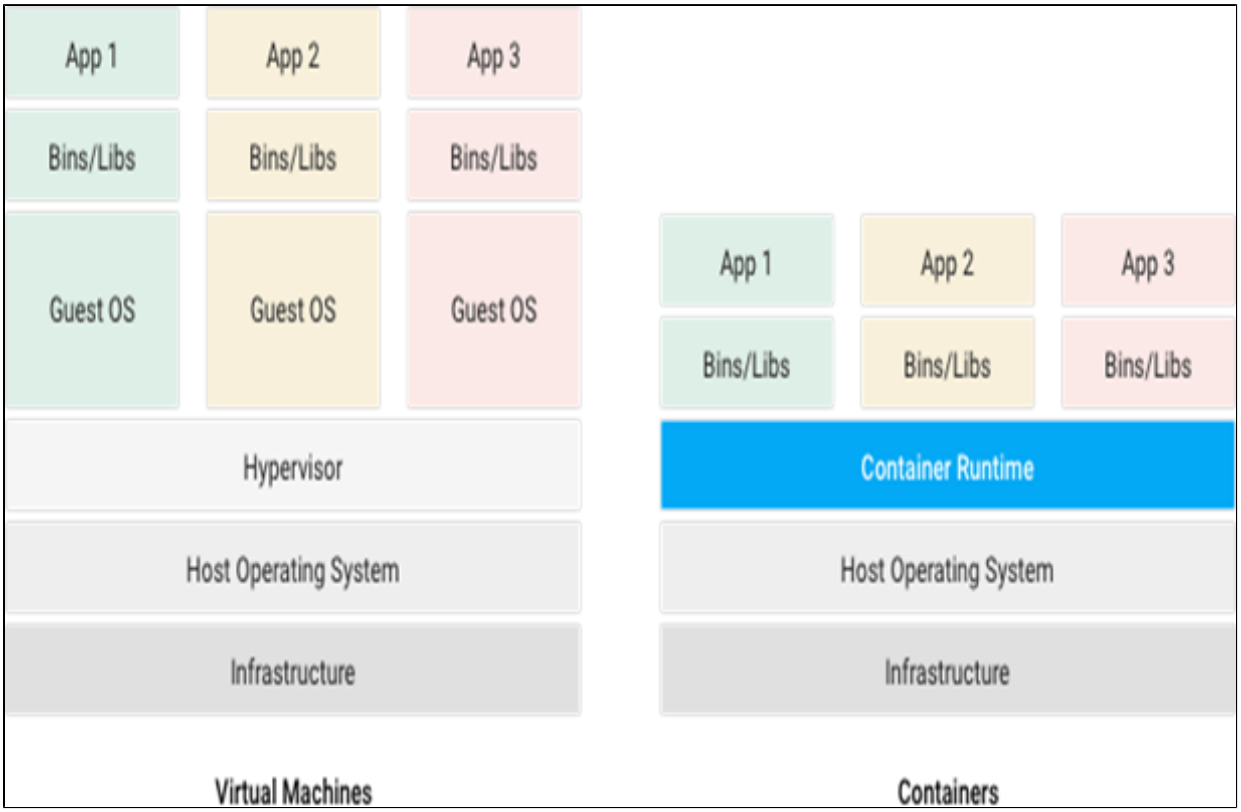
RDK-V Emulator containers

Introduction

Containers offer a logical packaging mechanism in which applications can be abstracted from the environment in which they run. Containers are often compared with virtual machines (VMs). a guest operating system such as Linux or Windows runs on top of a host operating system with virtual access to the underlying hardware. Like virtual machines, containers allow to package your application together with libraries and other dependencies, providing isolated environments for running your software services.



Why Containers?

- Instead of virtualizing the hardware stack, containers virtualize at the operating system level, with multiple containers running atop the OS kernel directly which means containers are more lightweight: they share the OS kernel, start much faster, and use a fraction of the memory compared to booting an entire OS.

- Container consists of an entire run-time environment: an application, plus all its dependencies, libraries and other binaries, and configuration files needed to run it, bundled into one package. By containerizing the application platform and its dependencies, differences in OS distributions and underlying infrastructure are abstracted away.

Advantages of Containers

- A container may be only tens of megabytes in size, whereas a virtual machine with its own entire operating system may be several gigabytes in size.
- Containerization allows for greater modularity. Rather than run an entire complex application inside a single container, the application can be split in to modules.

Implementation details

Containers layer - meta-rdk-containers:

- Consists of main container image(rdk-generic-hybrid-lxc-image).
- Latest "lxc-container-generator" has been added for container generation at do_rootfs stage.
- Distro feature and latest lxc version updated in qemux86hybsecure.conf.

Emulator layer - meta-rdk-bsp-emulator:

- Added emulator specific package groups and plugins to the container image.

New Container generation process:

This subsection describes how the new container generation process is replacing the earlier process.

- In this process containers will be generated using "lxc-container-generator" recipe, which will use corresponding .xml files to generate containers.
- All dependencies(such as required binaries,libraries,script files) will be provided in each container .XML file.
- For permissions of files "add-users-groups-file-owners-and-permissions.inc" file has been added.
- At rootfs stage containers will be generated in /container path of rootfs.
- Each container will consists of corresponding script (.sh) file for launching that particular container.
- Every process will be launched from corresponding component service file. Single (or) multiple processes can be launched/attached to container.

XML and conf files:

- All required XML and configuration files are placed along with lxc-container-generator recipe in meta-rdk-bsp-emulator layer.

Service files:

- **In platformcontrol container:**

  Three service files added for launching corresponding processes inside container (sysmgr.service, irmgr.service and dsmgr.service) .

- **In rmfstreamer container**:

  rmfstreamer.service file has been added.

- **In rdkbrowser2 container**:

  rdkbrowser2.service file has been added.

Implemented containers

Platformcontrol

- runs sysmgr,irmgr and dsmgr processes inside container.
- sysmgr will be launched in new container using lxc-execute.
- irmgr and dsmgr processes has been attached to same container using lxc-attach.

Rmfstreamer

- runs rmfstreamer inside container.
- rmfstreamer will be launched in new container using lxc-execute.

Rdkbrowser2

- runs rdkbrowser2 browser application inside container.
- westeros will be launched in new container using lxc-execute.
- rdkbrowser2 will be attached to the same container using lxc-attach.

Note: As we are in the plan of bringing APPmanager as default application we are not running rdkbrowser2 service file on boot-up.

Building procedure

- repo init -u https://code.rdkcentral.com/r/manifests -b rdk-next -m rdkv-asp-extsrc.xml
- repo sync --no-tags
- source meta-cmf-bsp-emulator/setup-environment
- meta-rdk-containers/conf/machine/qemux86hybsecure.conf
- bitbake rdk-generic-hybrid-lxc-image

Container verification

- pstree can be used to track the list of containers running as below.

```
root@qemux86hybsecure:/lib/systemd/system# cd
root@qemux86hybsecure:~#
root@qemux86hybsecure:~#
root@qemux86hybsecure:~# pstree
systemd-+-IARMDaemonMain
        |-agetty
        |-audiocapturemgr
        |-dbus-daemon
        |-diskMgrMain
        |-dnsmasq
        |-dropbear---sh---pstree
        |-ipIfaceMonitor.---sleep
        |-klogd
        |-lighttpd
        |-2*[login---sh]
        |-lxc-attach---dsMgrMain
        |-lxc-attach---irMgrMain
        |-lxc-execute---init.lxc.static---sysMgrMain
        |-mfrMgrMain
        |-netsrvmgr
        |-nlmon
        |-ntpd
        |-platformcontrol---usleep
        |-pwrMgrMain
        |-rdkbrowser2.sh---lxc-execute---init.lxc.static---westeros-init.s---westeros---westeros
        |-rdkbrowser2.sh---lxc-attach---rdkbrowser2.sh---rdkbrowser2-+-WPENetworkProce
        |                                                            `-WPEWebProcess
        |-rmfstreamer.sh---lxc-execute---init.lxc.static---rmfStreamer
        |-snmpd
        |-start.sh---dimclient
        |-storageMgrMain
        |-syslogd
        |-systemd-journal
        |-systemd-logind
        |-systemd-timesyn
        |-systemd-udevd
        |-tr69hostif
        |-2*[udhcpc]
        |-v86d
        |-xcal-device
        `-xdiscovery
root@qemux86hybsecure:~#
```

- ps -Af | grep lxc also lists the current running containers.

```
root@qemux86hybsecure:~#
root@qemux86hybsecure:~#
root@qemux86hybsecure:~#
root@qemux86hybsecure:~#
root@qemux86hybsecure:~# ps -Af | grep lxc
root       263   252  0 09:11 ?        00:00:00 /usr/bin/lxc-execute -n RDKBROWSER2 -f /container/RDKBROWSER
2/conf/lxc.conf -- /usr/bin/westeros-init.sh
rdkbrow+   286   263  0 09:11 ?        00:00:00 /init.lxc.static --gid 706 --uid 706 -- /usr/bin/westeros-in
it.sh
root       415     1  0 09:11 ?        00:00:00 /usr/bin/lxc-execute -n PLATFORMCONTROL -f /container/PLATFO
RMCONTROL/conf/lxc.conf -- /usr/bin/sysMgrMain --debugconfig /etc/debug.ini
root       417     1  0 09:11 ?        00:00:00 /usr/bin/lxc-attach -n PLATFORMCONTROL -f /container/PLATFOR
MCONTROL/conf/lxc.conf -u 704 -g 704 -- /usr/bin/dsMgrMain --debugconfig /etc/debug.ini
root       419     1  0 09:11 ?        00:00:00 /usr/bin/lxc-attach -n PLATFORMCONTROL -f /container/PLATFOR
MCONTROL/conf/lxc.conf -u 704 -g 704 -- /usr/bin/irMgrMain --debugconfig /etc/debug.ini
platfor+   422   415  0 09:11 ?        00:00:00 /init.lxc.static --gid 704 --uid 704 -- /usr/bin/sysMgrMain
--debugconfig /etc/debug.ini
root       707   703  0 09:11 ?        00:00:00 /usr/bin/lxc-execute -n RMFSTREAMER -f /container/RMFSTREAME
R/conf/lxc.conf -- /usr/bin/rmfStreamer --config /etc/rmfconfig.ini --debugconfig /etc/debug.ini
rmfstre+   709   707  0 09:11 ?        00:00:00 /init.lxc.static --gid 705 --uid 705 -- /usr/bin/rmfStreamer
 --config /etc/rmfconfig.ini --debugconfig /etc/debug.ini
root      2526  2525  0 09:13 ?        00:00:00 /usr/bin/lxc-attach -n RDKBROWSER2 -f /container/RDKBROWSER2
/conf/lxc.conf -u 706 -g 706 -- /usr/bin/rdkbrowser2.sh
root     16706  1198  0 10:20 pts/0    00:00:00 grep lxc
root@qemux86hybsecure:~#
```

DEBUG Logs

- strace can give more debug information about containers:

Example:
strace -f -o lxc-execute.log /usr/bin/lxc-attach -n PLATFORMCONTROL -f /container/PLATFORMCONTROL/conf/lxc.conf

-u 704 -g 704 -- /usr/bin/dsMgrMain

- lxc-execute.log for debugging purpose.

Test cases

- RMFAPP can be used to verify rmfstreamer container.
  Example: play http://192.168.2.68:8080/vldms/tuner?ocap_locator=ocap://0x125d
- RDKBROWSER2 can be used to launch any URL.
  Example: systemctl enable rdkbrowser2.service
  systemctl start rdkbrowser2.service - user can see the webpage in rdkbrowser2.

  If user wants to change URL, then we need to enter into this container and need to change rdkbrowser2.sh binary as below:
  systemctl stop rdkbrowser2.service
  systemctl stop westeros-setup.service\
  use command:
  /usr/bin/lxc-execute -n RDKBROWSER2   -f /container/RDKBROWSER2/conf/lxc.conf – /bin/sh
  and then change url in /usr/bin/rdkbrowser2.sh file inside this container environment.

Reference

- https://cloud.google.com/containers/
- https://www.cio.com/article/2924995/software/what-are-containers-and-why-do-you-need-them.html
- https://linuxcontainers.org/
- https://help.ubuntu.com/lts/serverguide/lxc.html
- https://linuxcontainers.org/lxc/documentation/