FUJITSU

shaping tomorrow with you

LXCF: Tools for Dividing Host OS into Container with libvirt-LXC

May 20, 2014

Yasunori Goto / Hideyuki Niwa
Fujitsu Limited

Who are we?

FUJITSU

- Yasunori Goto (Speaker)
  - Leader for KVM and LXC development team of Fujitsu
    - developed Linux memory hotplug with community until 2008
    - joined customer support team to analyze users troubles about Linux kernel for several years
    - returned to the Linux development team last year
- Hideyuki Niwa
  - Author of LXCF
    - developed UNIX OS for super computer with vector processor
    - joined Linux division and has worked for Linux now

Agenda

FUJITSU

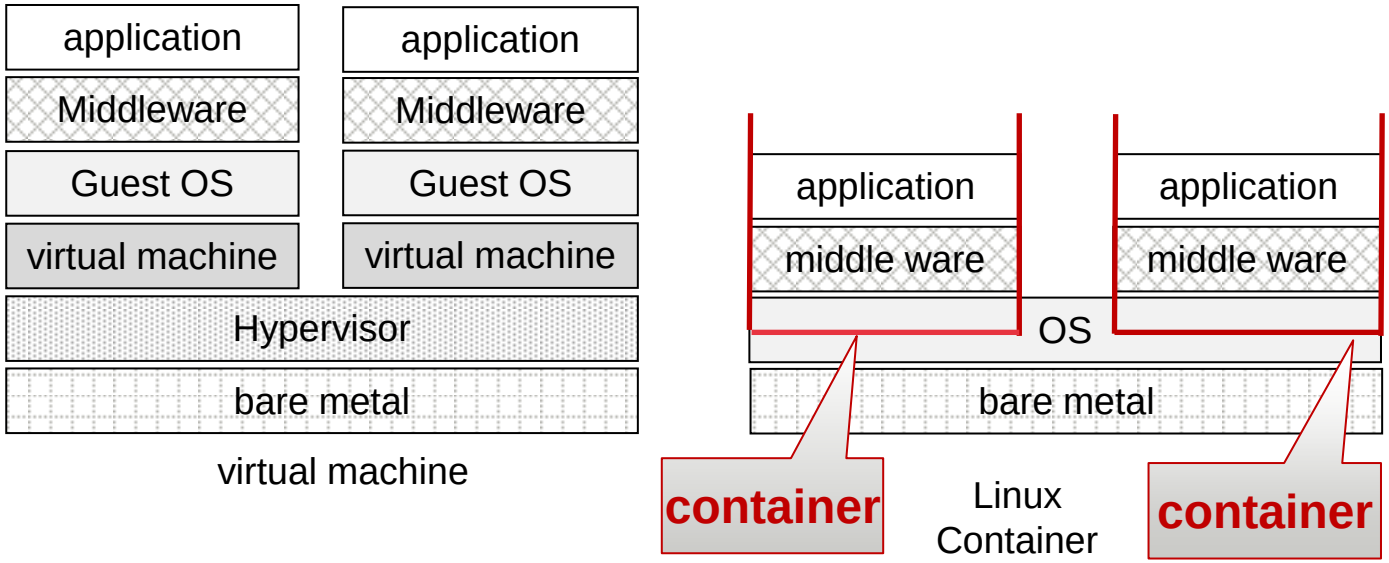
- Basis of LXC
- Why we are developing LXCF
- Design of LXCF
- How to use

FUJITSU

Basis of LXC

Linux Container

- Linux Container is an OS level virtualization method
  - It provides multiple isolated environments on one host
    - User can execute many services on the host
  - No need to emulate hardware like virtual machine
    - Performance is better than virtual machine



User-land tools of LXC

FUJITSU

- To use Linux Container, user-land tool is necessary
  - Kernel provides only...
    - system calls to manage namespace (clone(), unshare(), and setns())
    - interfaces for resource control (cgroup)
- There are many user-land tool set
  - LXC (userland toolset which is simply called as “LXC”)
    - well developed tool to use Linux Container feature
  - Libvirt-lxc
    - provides core feature for Linux Container with libvirt
  - Docker
    - “The hottest”
    - “Linux Container” + “git style image management”
    - for immutable system
  - etc
    - lmctfy, systemd-nspawn

Note in this presentation

FUJITSU

- The name of “LXC” has 2 meanings
  1. General features/APIs of Container
  2. Userland tool-set which is shown in the previous page
- To prevent confusing, I would like to use the following words in this presentation
  - “Linux Container” means the first meaning
  - “LXC” means the second one

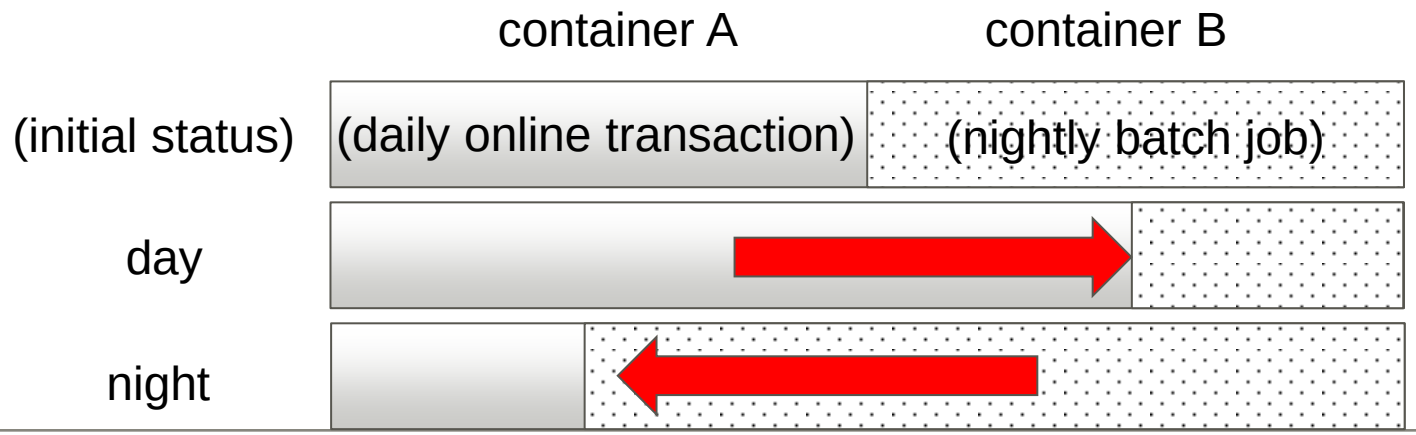
FUJITSU

Why we are developing LXCF

Motivation of LXCF

FUJITSU

- To provide “a system divided into some containers” for Enterprise customers
  - For example, execute “daily online transaction job” on a container and “nightly batch job” on another container in one host
  - Easy to tune resources of each container
    - Making “dynamic resource control” easy on the above example



Purpose of LXCF

FUJITSU

- Divide a system into some containers
  - Put several services/applications together on a host
  - Provide likely full OS environment by Linux Container
    - systemd works in each container
    - User can operate/maintain services like bare metal environment
  - Same runtime/ libraries in each container with host
    - Application / Middleware works easily in LXCF container by providing similar environment of bare metal
- Easy resource tuning
  - For dynamic resource control
- Long life container
  - need update software which user specified  (next page)
    - not only host's software, but also containers' one

What long life means?

FUJITSU

- Customers' demand on enterprise system
  - "Long life support" is required
    - Users require 10 years support
      - The most extreme example, one user required 20 years support
        Since a "law" for his factory required him to keep the line of factory for 20 years, he requested us for same length support for the factory's system
  - want to update only packages which have problems
    - Maintenance time for operating system is very limited
    - They do not update software as much as possible
      - update means that users have to test the system again
    - Even if update is necessary, users tend to avoid changing whole of image to minimize the influence of update
    - Blue/green deployment is not popular yet

Existing userland toolsets

FUJITSU

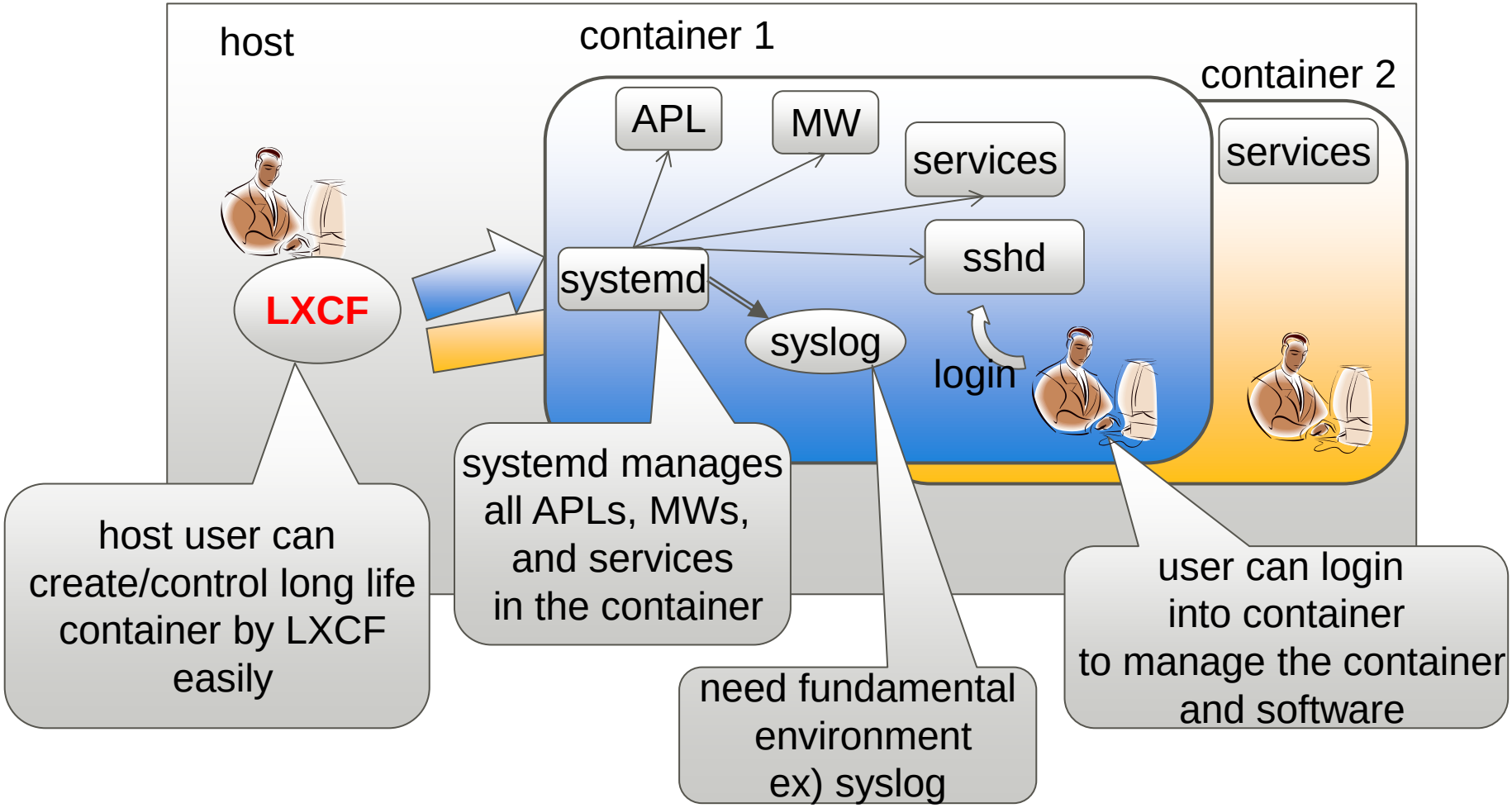
- From our point of view
  - LXC
    - RHEL / CentOS may not merge LXC :-P
    - If runtime is shared among containers by bind-mount, its update will fail
      - (I will describe why update fails for bind-mounted directory later...)
  - Libvirt-lxc
    - only provide core features / interfaces
    - User has to customize to create a new container
      - It is hard to use
  - Docker
    - force users to change their operation and maintenance way drastically
    - Enterprise users have to change policy of test
    - Though Docker has promising future, it need much time for enterprise users to accept Docker's concept

FUJITSU

Design of LXCF

Image of container with LXCF

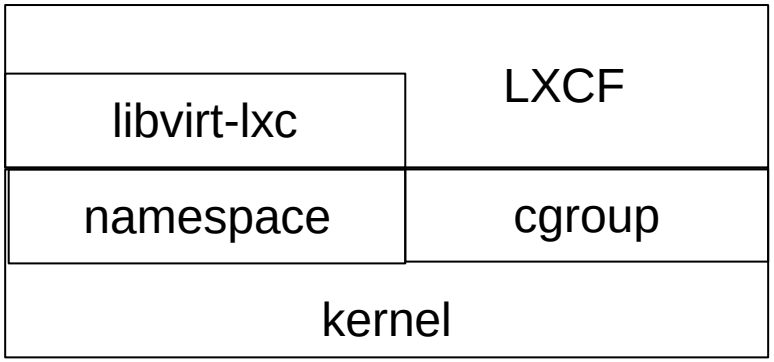
- LXCF helps making/control containers



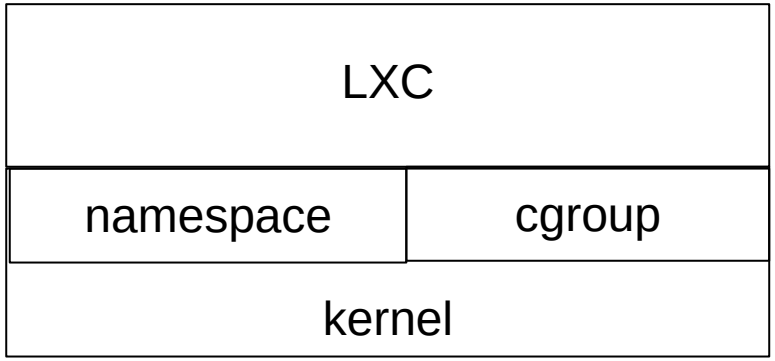
Software Stack of LXCF

FUJITSU

- LXCF depends on libvirt-lxc
  - But, LXCF accesses cgroup interfaces directly now.
    - Cgroup interfaces are under reconstruction currently
    - If libvirt (or systemd) will support all of cgroup interfaces, lxcf would like to use it.



LXCF



LXC

2 Models of LXCF

FUJITSU

- LXCF supports 2 type use-case by the following models
  - Separate model
  - Joint model

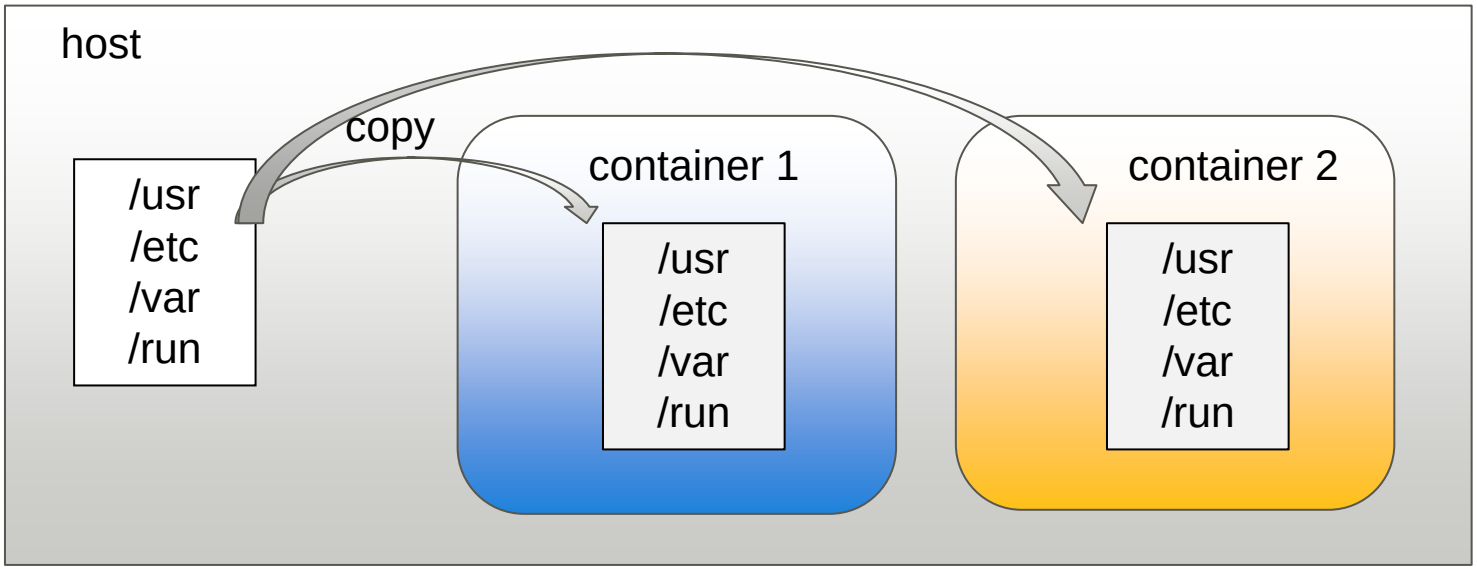
- This setting is selected by config file of LXCF on host

```
[root@localhost /]# cat /etc/lxcf/lxcf.conf
#
# model = joint or separate
#
[Model]
model=joint    <----- here!
```

Separate Model (1/2)

FUJITSU

- Basic usage
  - Each container has its own directory
    - many files are copied from host



  - User has to install/update applications and MW for each containers
  - Container private installation is allowed
    - E.g.) for DB containers, for web server containers, etc

Separate model (2/2)

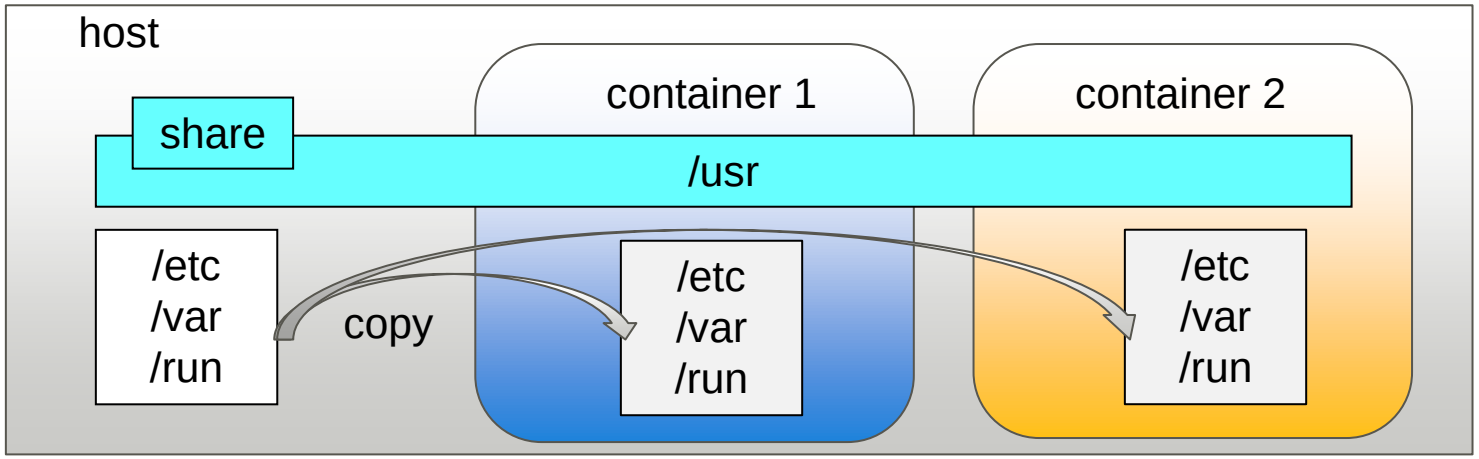
FUJITSU

- How to create new container's directories
  - Copied from host to each container
    - /usr, /etc, /var
  - Symbolic linked
    - /bin , /sbin, /lib, /lib64
  - Automatically created when the container start
    - /dev, /proc, /tmp, /sys, /run
  - Empty directories are created
    - /boot, /home, /media, /mnt, /opt, /srv
  - .bashrc is only copied from host
    - /root

Joint Model (1/2)

FUJITSU

- Extended usage
  - Runtime is shared
    - Each container bind-mounts /usr
      - (/bin, /sbin and /lib are symbolic-linked to /usr)
      - other files are copied to each container

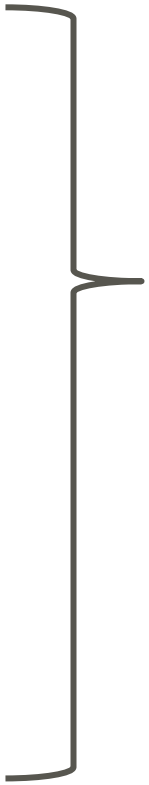


  - Each container users don't have to update software on each container
    - When user update software of host, user of each container can use it
      It will reduce update cost very much
    - no private installation

Joint model (2/2)

FUJITSU

- How to create new container's directories
  - Bind-mounted to host's directory
    - /usr
  - Copied from host
    - /etc, /var
  - Symbolic link
    - /bin, /sbin, /lib, /lib64,
  - Automatically created when the container start
    - /dev, /proc, /tmp, /sys, /run
  - Create empty directories
    - /boot, /home, /media, /mnt, /opt, /srv
  - .bashrc is only copied from host
    - /root



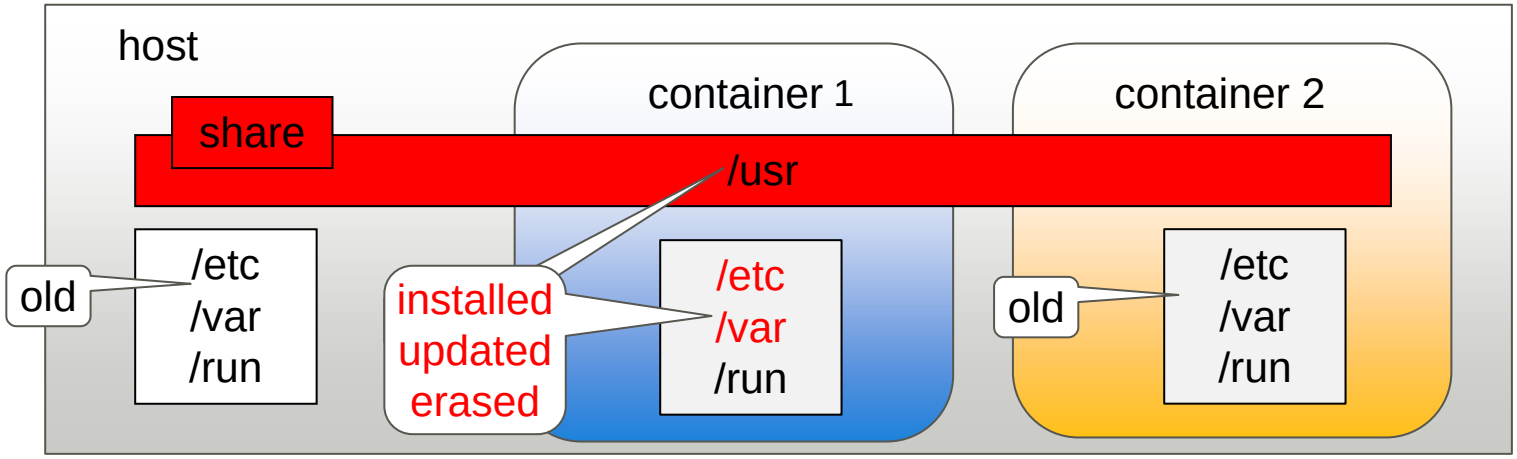
Pros and cons of 2 models

FUJITSU

- SEPARATE model
  - pros
    - Private installation is allowed for each container
  - cons
    - Creating new container is slower due to the time for runtime copy
      - runtime:5.4GiB, other files: 0.9GiB, total files : 6.3GiB (Fedora default install)
- JOINT model
  - pros
    - Creating new container is faster than SEPARATE model
      - No runtime copy (5.4GiB)
  - cons
    - Due to same binaries are shared, private installation is disallowed
    - Stop all containers at software update

Problem of runtime sharing

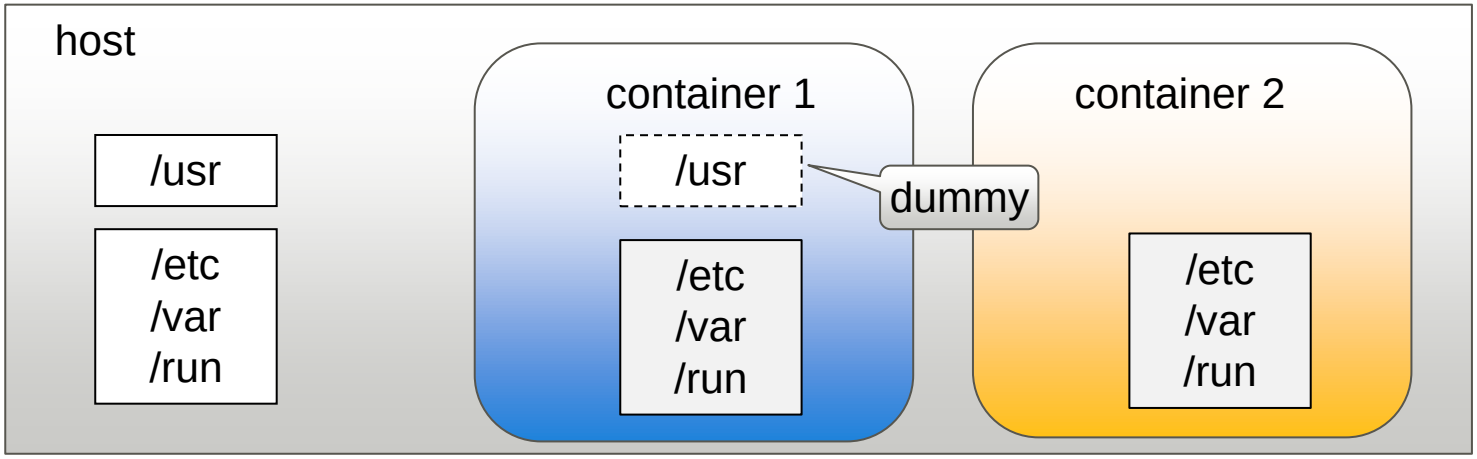
- Runtime is shared at Joint model, but how to install/update/erase a package?
  - For example, if user execute rpm/yum command in container, shared runtime affects host and other containers, but other files are still old



How LXCF maintain runtime on Joint model (1/4)

- LXCF have special command to maintain software packages of host and guest
- Here is the step of the command

1. It makes “dummy” /usr directory for a container

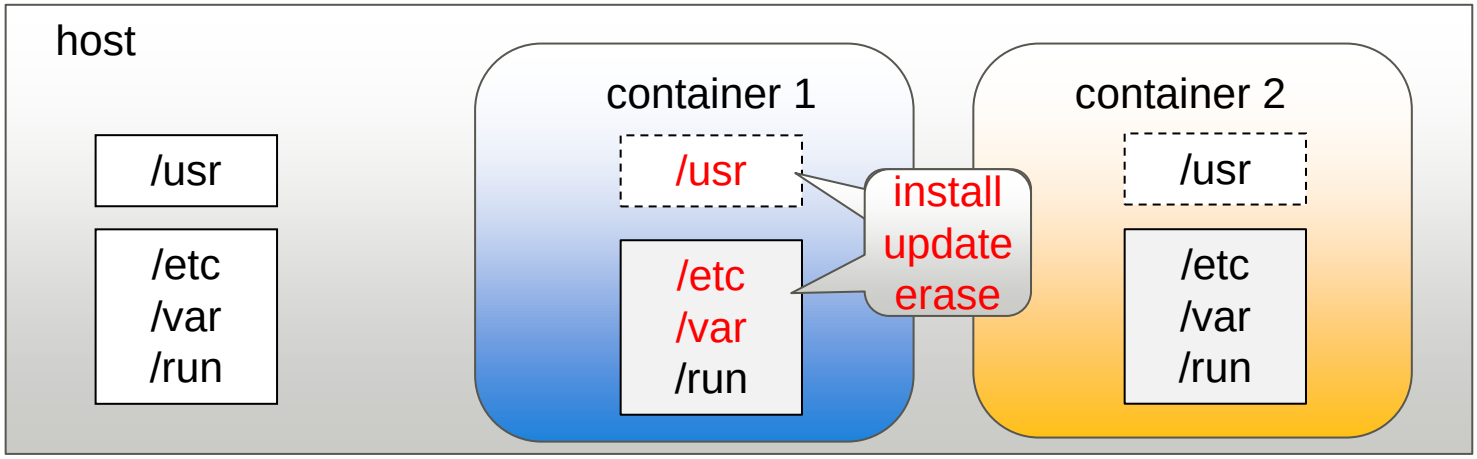


(cont.)

How LXCF maintain runtime on Joint model (2/4)

FUJITSU

2. It executes package maintenance command (yum/rpm) for the container
   - It installs/updates runtime of the “dummy” /usr directory on the container and config files in /etc, and so on if necessary
   - If erase is specified, it removes them

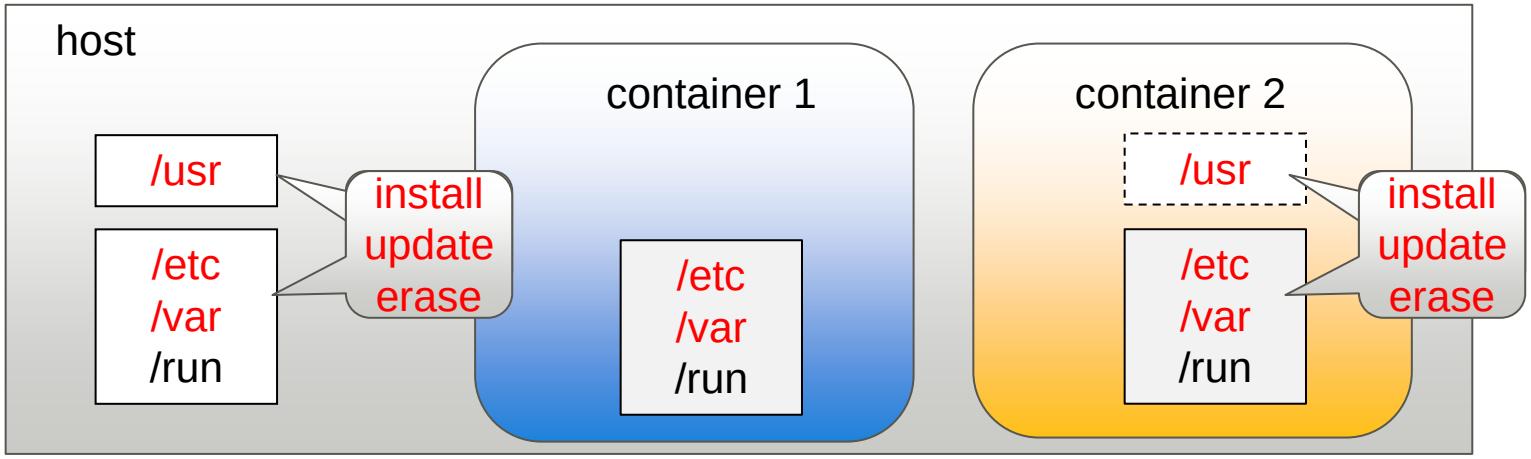


(cont.)

How LXCF maintain runtime on Joint model (3/4)

FUJITSU

3. Step 2 is repeated for other containers and host

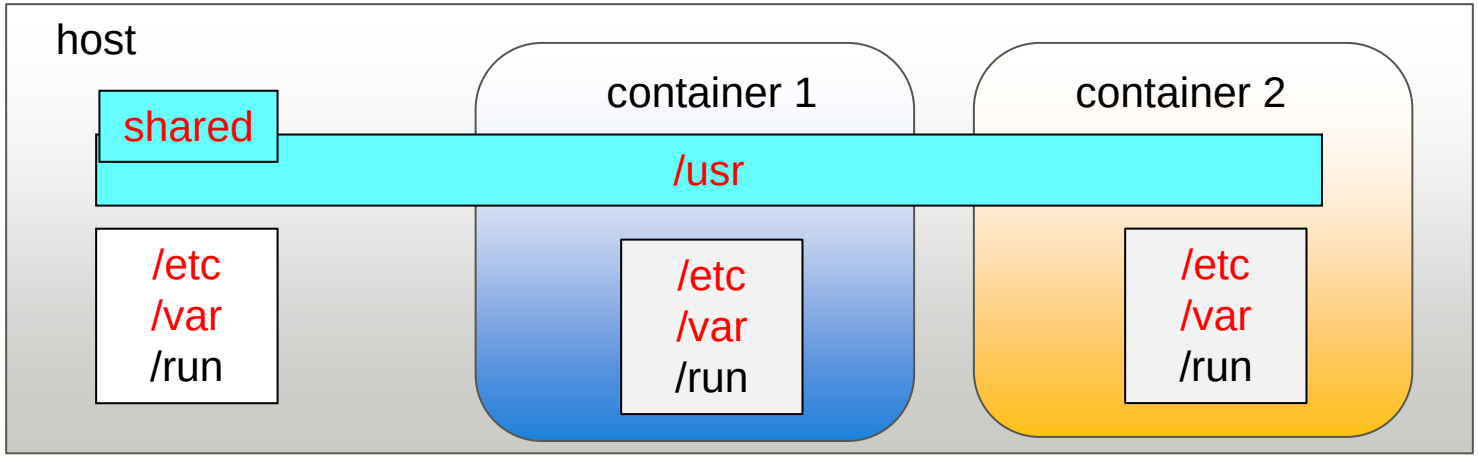


(cont.)

How LXCF maintain runtime on Joint model (4/4)

FUJITSU

4. runtime of host is shared again



Intended limitations of LXCF

FUJITSU

- LXCF provides same environment container with the host
  - same distribution, and same version
  - If different environment is provided, LXCF can not maintain those containers
- Basically LXCF assume that host admin and containers' admins are same
  - To simplify security consideration
- Nested container is not supported
  - Necessary?

FUJITSU

How to use LXCF

Basis of LXCF commands

FUJITSU

- lxcf <subcommand> [ARGs]
  - Example
    - create and start new container
      - lxcf sysgen <new container name>
    - stop and erase the container
      - lxcf erase <container name>
    - start the container
      - lxcf start <container name>
    - stop the container
      - lxcf stop <container name>
- Only root user can execute lxcf commands on host

Create and start new container

FUJITSU

- To create and start new container
  - # lxcf sysgen <new container's name>
- New container's root directory
  - /opt/lxc/<new container's name>/
  - LXCF make directories depends on the setting of Separate or Joint model as I mentioned
- LXCF containers can start automatically when host boots up with the following command
  - # lxcf autostart <container's name>
- Disable the above autostart setting
  - # lxcf autostart -d <container's name>

Network setting for new container

FUJITSU

- When LXCF package is installed, it creates lxcfnet1(virbr1) on host
  - NAT (192.168.125.0/24, gateway: 192.168.125.1)
- LXCF assigns a “fix” IP address for new container
  - It registers to /etc/hosts file of host with new container’s name automatically

More commands to manage containers

FUJITSU

- Clone a container
  - # lxcf clone <src container name> <new container name>
- Copy file or directory from host to each containers
  - # lxcf deploy <file name | directory name>
- check containers status
  - # lxcf list

Resource control

FUJITSU

Easy setting for resource control

Examples

- Limit cpu usage rate for a container
  - # lxcf cpurate <container's name> <cpu usage ratio>
- Limit the amount of available memory for a container
  - # lxcf memlimit <container's name> <amount of memory>
- Limit container's numa node
  - # lxcf numa <container's name> <numa-node numbers>
- Limit device's throughput of read for a container
  - # lxcf blkiops_r <container's name> <read limit>
- Confirm resource control settings
  - # lxcf show <container's name>

Maintenance

FUJITSU

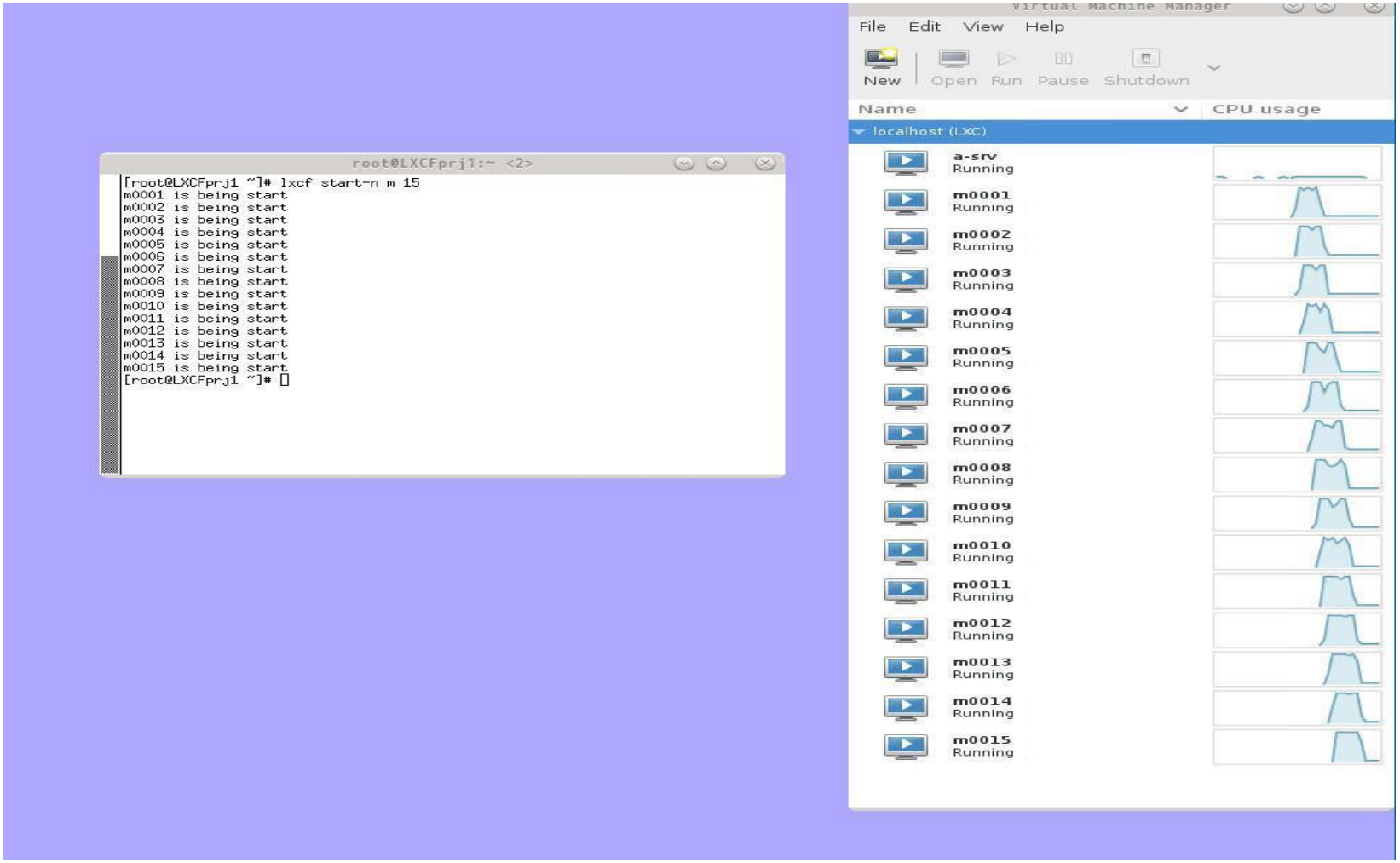
- As I described, LXCF has special command for maintain containers
  - # lxcf update <maintenance commands>

  Example)
  #lxcf update yum update -y <*packcage_name*>
  or
  #lxcf deploy <rpm package name>
  #lxcf update rpm -Uvh <rpm package name>

Demo

FUJITSU

LXCF Screenshot on GUI



Version of LXCF, distribusion and licence

FUJITSU

- Current version is 0.5
- LXCF works on Fedora 19/20
  - LXCF is mainly developed on Fedora 20.
  - Trying to merge the LXCF package into Fedora21
- LXCF will be able to run on CentOS 7 / RHEL7
- License of LXCF is GPL v2

Todo

FUJITSU

- Btrfs support
  - Use Copy On Write feature of Btrfs
- More flexible network setting
- Share runtime under /opt in Joint model
  - ISV’s application/middleware installed under it
- REST API?

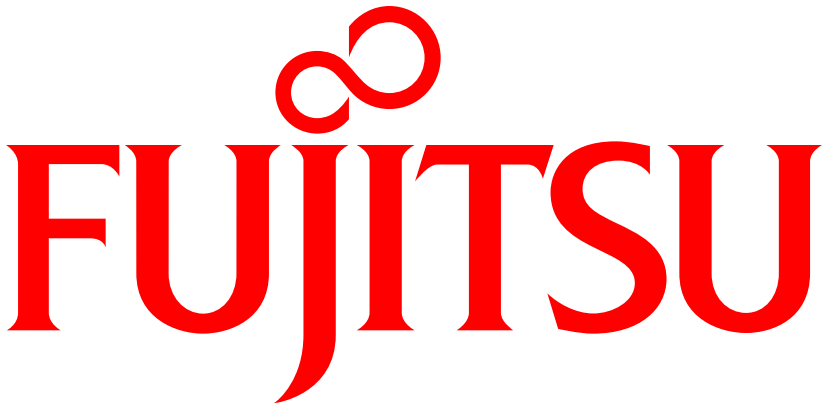
Summary

- Introduce LXCF
  - Why LXCF is created
  - Design of LXCF
  - How to use LXCF
  - Todo

Refernces

FUJITSU

- Web page
  - http://sourceforge.net/projects/lxcfacility
- Mailing List
  - https://lists.sourceforge.net/lists/listinfo/lxcfacility-allmail1
- New patch/proposal is welcome



shaping tomorrow with you

Why copy the files into container?

FUJITSU

- Q: Other userland tools make container's system volume via network (apt-get, yum, or download images). Why does LXCF copy files from host?
- A:
  1. Mission critical system is often disconnected from internet due to user's policy
     - User may not be able to install/update software via network
  2. The result of focusing same environment between host and guest as much as possible

Other userland tools

FUJITSU

- lmctfy
  - Currently, it focuses only easy resource control and isolation in host
  - no description about long life
- systemd-nspawn
  - man page says “for debugging, testing, and building”