OS Containers

Michal Sekletár
msekleta@redhat.com

June 14, 2017

whoami

- Senior Software Engineer @ Red Hat
- systemd and udev maintainer
- Free/Open Source Software contributor

Agenda

- Introduction to OS containers
- Implementations of OS Containers
  - FreeBSD
  - Solaris
  - Linux
    - Linux namespaces
    - Control Groups

Introduction

Q: What is a container?

Introduction

A: Virtualized application execution environment.

Container Technologies

- chroot
- BSD Jails
- Solaris Zones
- Linux VServer
- namespaces
- cgroups
- OpenVZ
- LXC
- systemd-nspawn
- Docker
- rkt

chroot(2)

- System call used to change filesystem root of a process
- First appeared in 4.1BSD in early 80's (March 18 1982 according to SCCS log)

```
chroot()
{
  if (suser())
        chdirec(&u.u_rdir);
}
```

Figure: Original implementation of chroot system call

BSD Jails

- FreeBSD - 4.0 (1998)
- Legacy UNIX privilege delegation model is basically non-existent
- Authors of jails wanted to solve problem of "omnipotent root"
- jail (2) syscall
- `struct prison`
- Once process is put to a jail it can't get out anymore
- Jails heavily leverage chroot (2)
- Some functionality is disabled in a jail (mounting filesystems, creating SOCK_RAW sockets)

BSD Jails

Each jail is characterized by 4 main properties,

- Directory subtree
- Hostname
- IP address (usually an alias address on existing interface)
- A command to run

Jails DEMO

Solaris Zones

- Early 2000's (2004)
- Inspired by jails work in FreeBSD
- Zones were trying to solve problems related to workload consolidation
- Sun Microsystems was selling boxes too big to run only a single application
- Primitives baked into an operating system
- `zone_t`
- 2 kinds of zones
  - Global
  - Non-global
- Global zone is a first (default zone) in the system
- Zone must be installed and configured, then it is booted and user can log into it
- ICMP is allowed but RAW sockets are still not allowed in a non-global zone
- zoneadm, zonecfg commands

Zones DEMO

Linux Containers

- First implemented as part of Virtuozzo
- OpenVZ in 2005
- Implementation of underlying technologies was later included upstream
- Namespaces
- Cgroups

Linux Namespaces

- Feature provided by Linux
- Used to virtualize various global system resources
  - mount
  - PID
  - user
  - uts
  - network
  - IPC
  - cgroup
- System calls used to manipulate namespaces,
  - `clone`
  - `unshare`
  - `setns`

Linux Namespaces

```
# ls -l /proc/self/ns
total 0
lrwxrwxrwx. 1 root root 0 Nov  6 09:09 cgroup -> ’cgroup:[4026531835]’
lrwxrwxrwx. 1 root root 0 Nov  6 09:09 ipc -> ’ipc:[4026531839]’
lrwxrwxrwx. 1 root root 0 Nov  6 09:09 mnt -> ’mnt:[4026531840]’
lrwxrwxrwx. 1 root root 0 Nov  6 09:09 net -> ’net:[4026531969]’
lrwxrwxrwx. 1 root root 0 Nov  6 09:09 pid -> ’pid:[4026531836]’
lrwxrwxrwx. 1 root root 0 Nov  6 09:09 user -> ’user:[4026531837]’
lrwxrwxrwx. 1 root root 0 Nov  6 09:09 uts -> ’uts:[4026531838]’
```

Mount Namespace

- Virtualization of a filesystem view
- `unshare -m /bin/bash`
- Oldest namespace
- `clone(2)` argument `CLONE_NEWNS`
- Mount point propagation,
  - private
  - shared
  - slave
  - unchanged

PID Namespace

- Virtualization of process identifiers,
- `CLONE_NEWPID`
- `unshare -p --fork --mount-proc /bin/bash`
- `init` process in PID namespace
- Reaps zombie processes within namespaces
- Same signal handling exceptions applies as for real PID 1
- When `init` exits all other processes in a namespace get SIGKILL from kernel
- PID namespace of a process can't be changed
- It is possible to nest PID namespaces

User Namespace

- Virtualization of user and group databases and capabilities
- `unshare -U --map-root /bin/bash`
- Mapping of users between a container and a host system (created by writing to `/proc/[pid]/uid, gid_map`)
- User namespaces can be nested

Network Namespace

- `unshare -n /bin/bash`
- Virtualization of network related system resources,
  - Interfaces
  - IPv4 stack
  - IPv6 stack
  - Routing tables
  - Ports
- `veth` pair to create tunnel between namespaces

Other Kernel Namespaces

- IPC
  - Isolation of SystemV IPC resources and POSIX message queues
  - `unshare -i /bin/bash`
- UTS
  - Virtualization of hostname and NIS domain name
  - `unshare -u /bin/bash`
- Cgroups
  - Virtualization of a cgroup tree view
  - `unshare -C /bin/bash`

Linux Cgroups

- Subsystems used for process aggregation and resource limiting
- Named hierarchies
- Cgroup controllers,
  - `cpuset`
  - `cpu`
  - `cpuacct`
  - `blkio`
  - `memory`
  - `devices`
  - `freezer`
  - `net_cls`
  - `net_prio`
  - `pids`
- Orthogonal hierarchies
- Unified hierarchy in cgroup fs v2