SYSTEMD-NSPAWN 101

Lukáš Nykrýn <lnykryn@redhat.com>

http://redhat.slides.com/lnykryn/systemd-nspawn-101

SYSTEMD ECOSYSTEM

WHAT DO YOU HAVE FOR ME?

SYSTEMD ECOSYSTEM

2010
LENNART POETTERING

HTTPS://WWW.FREEDESKTOP.ORG/WIKI/SOFTWARE/SYSTEMD/
HTTPS://GITHUB.COM/SYSTEMD/SYSTEMD

SYSTEMD = SYSTEM MANAGER

SET OF TOOLS FOR MANAGING SYSTEM

STANDARD BASE FOR LINUX USER-SPACE

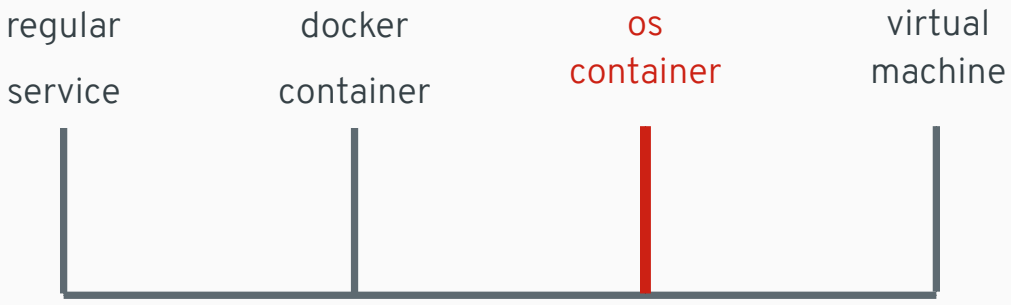
SYSTEMD ECOSYSTEM

SYSTEMD-NSPAWN

DESIGNED AS TOOL FOR TESTING SYSTEMD

CURRENTLY USED AS A BACKEND FOR RKT

FOCUSES ON OS CONTAINERS



SYSTEMD ECOSYSTEM

SYSTEMD / SYSTEMCTL

starting the containers on boot, init inside the container

SYSTEMD-MACHINED / MACHINECTL

registration of containers, working with images and containers

SYSTEMD-JOURNALD / JOURNALCTL

aggregation of logs

SYSTEMD-IMPORTD

downloading images

SYSTEMD-NETWORKD

network setup

CREATING IMAGE

FOR THE START, I PROBABLY NEED TO DOWNLOAD SOMETHING, RIGHT?

CREATING A CONTAINER

SUPPORTED FORMATS

PLAIN DIRECTORY

BTRFS SUBVOLUME

TAR

RAW IMAGE

CREATING A CONTAINER

DOWNLOADING A IMAGE

tar images

```
[root@localhost ~]# machinectl pull-tar \
https://cloud-images.ubuntu.com/trusty/\
current/trusty-server-cloudimg-amd64-root.tar.gz
```

raw images

```
[root@localhost ~]# machinectl pull-raw --verify=no \
https://download.fedoraproject.org/pub/fedora/\
linux/releases/24/CloudImages/x86_64/images/Fedora-Cloud-Base-24-\
1.2.x86_64.raw.xz
```

pull-* will also download .nspawn file if it is present on the server.

CREATING A CONTAINER

USING PACKAGE MANAGER

dnf (Fedora)

```
[root@localhost ~]# dnf -y --releasever=25 \
--installroot=/srv/fedora-tree --disablerepo='*' \
--enablerepo=fedora --enablerepo=updates install \
systemd passwd dnf fedora-release vim-minimal
```

debootstrap (Debian)

```
[root@localhost ~]# debootstrap --arch=amd64 unstable /srv/debian-tree/
```

pacstrap (Arch Linux)

```
[root@localhost ~]# pacstrap -c -d /srv/arch-tree/ base
```

CREATING A CONTAINER

MKOSI

https://github.com/systemd/mkosi

```
[root@localhost ~]# mkosi
[root@localhost ~]# mkosi -t directory -o fedora
[root@localhost ~]# mkosi -t raw_btrfs --bootable -o foobar.raw*
```

* You can run this image inside VM or install it on physical machine (UEFI required)

STARTING A CONTAINER

WELL, I HAVE AN IMAGE READY, WHAT NOW?

STARTING A CONTAINER

CHROOT-LIKE ENVIRONMENT

```
[root@localhost machines]# systemd-nspawn -D /var/lib/machines/fedora/
Spawning container fedora on /var/lib/machines/fedora.
Press ^] three times within 1s to kill container.
-bash-4.3# ps
  PID TTY          TIME CMD
    1 ?        00:00:00 bash
   16 ?        00:00:00 ps
-bash-4.3#
```

STARTING A CONTAINER

PID 2 (MY SERVICE CAN'T RUN AS INIT)

```
[root@localhost machines]# systemd-nspawn --as-pid2 \
-M fedora ps -A
Spawning container fedora on /var/lib/machines/fedora.
Press ^] three times within 1s to kill container.
  PID TTY          TIME CMD
    1 ?        00:00:00 STUBINIT
    2 ?        00:00:00 ps
Container fedora exited successfully.
```

STARTING A CONTAINER

BOOTING A CONTAINER

```
[root@localhost machines]# systemd-nspawn -b -D /var/lib/machines/fedora/
Spawning container fedora on /var/lib/machines/fedora.
Press ^] three times within 1s to kill container.
systemd 231 running in system mode. (+PAM +AUDIT +SELINUX +IMA -APPARMOR
+SMACK +SYSVINIT +UTMP +LIBCRYPTSETUP +GCRYPT +GNUTLS +ACL +XZ +LZ4 +SECC
OMP +BLKID +ELFUTILS +KMOD +IDN)
Detected virtualization systemd-nspawn.
Detected architecture x86-64.

Welcome to Fedora 25 (Twenty Five)!
[  OK  ] Listening on /dev/initctl Compatibility Named Pipe
[  OK  ] Started Dispatch Password Requests to Console Directory Watch.
...

Fedora 25 (Twenty Five)
Kernel 4.8.0-0.rc2.git3.1.fc25.x86_64 on an x86_64 (console)

fedora login:
```

STARTING A CONTAINER

RUNNING ON BACKGROUND

```
[root@localhost machines]# machinectl start fedora
[root@localhost machines]#
[root@localhost machines]# machinectl enable fedora
Created symlink /etc/systemd/system/machines.target.wants/sys
temd-nspawn@fedora.service → /usr/lib/systemd/system/systemd

-nspawn@.service.
```

WORKING WITH IMAGES

IMAGES SHOULD BE IN /VAR/LIB/MACHINES.
WHAT CAN I DO WITH THEM THERE?

WORKING WITH IMAGES

LIST INSTALLED IMAGES

```
[root@localhost ~]# machinectl list-images
NAME          TYPE      RO  USAGE CREATED                          MODIFIED
centos        directory no  n/a   n/a                              n/a
fedora        directory no  n/a   n/a                              n/a
fedora-btrfs subvolume no  n/a   Sun 2016-09-25 15:33:08 CEST n/a

3 images listed.
```

WORKING WITH IMAGES

GET INFORMATION ABOUT IMAGE

```
[root@localhost ~]# machinectl image-status fedora-btrfs
fedora-btrfs
             Type: subvolume
             Path: /var/lib/machines/fedora-btrfs
               RO: writable
          Created: Sun 2016-09-25 15:33:08 CEST; 14min ago
```

WORKING WITH IMAGES

GET INFORMATION ABOUT IMAGE

(scriptable edition)

```
[root@localhost ~]# machinectl show-image fedora-btrfs
Name=fedora-btrfs
Path=/var/lib/machines/fedora-btrfs
Type=subvolume
ReadOnly=no
CreationTimestamp=Sun 2016-09-25 15:33:08 CEST
Usage=18446744073709551615
Limit=18446744073709551615
UsageExclusive=18446744073709551615
LimitExclusive=18446744073709551615
```

WORKING WITH IMAGES

OTHER STUFF

Create clone of image

```
[root@localhost ~]# machinectl clone fedora-btrfs fedora-clone
```

Rename image

```
[root@localhost ~]# machinectl rename fedora-btrfs fedora-base
```

Set image read-only

```
[root@localhost ~]# machinectl read-only fedora-base true
```

CONTROLLING MACHINES

OK, WE NOW KNOW HOW TO START CONTAINER BY HAND.

HOW DO WE START IT IN BACKGROUND?

HOW DO WE STOP IT? HOW CAN I KNOW WHAT IS RUNNING THERE?

CONTROLLING MACHINES

START CONTAINER

```
[root@localhost ~]# machinectl start fedora
```

CONTROLLING MACHINES

STATUS OF CONTAINER

```
[root@localhost ~]# machinectl status fedora
fedora(d5e8fae516ab454d8c5da64f698bf6ba)
           Since: Sun 2016-09-25 16:57:48 CEST; 1h 14min ago
          Leader: 2910 (systemd)
         Service: systemd-nspawn; class container
            Root: /var/lib/machines/fedora
           Iface: ve-fedora
              OS: Fedora 25 (Twenty Five)
            Unit: systemd-nspawn@fedora.service
                  |-2888 /usr/bin/systemd-nspawn --quiet --keep-unit --boot --link-journal=try-guest --networ
k-veth -U --settings=override --machine=fedora
                  |-init.scope
                  | `-2910 /usr/lib/systemd/systemd
                  `-system.slice
                    |-console-getty.service
                    | `-2950 /sbin/agetty --noclear --keep-baud console 115200,38400,9600 vt220
                    |-dbus.service
                    | `-2947 /usr/bin/dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-a
ctivation
                    |-systemd-journald.service
                    | `-2929 /usr/lib/systemd/systemd-journald
                    `-systemd-logind.service
                      `-2949 /usr/lib/systemd/systemd-logind

Sep 25 16:57:48 localhost.localdomain systemd-nspawn[2888]: [  OK  ] Reached target Login Prompts.
Sep 25 16:57:48 localhost.localdomain systemd-nspawn[2888]: [  OK  ] Started Login Service.
Sep 25 16:57:48 localhost.localdomain systemd-nspawn[2888]: [  OK  ] Reached target Multi-User System.
Sep 25 16:57:48 localhost.localdomain systemd-nspawn[2888]: [  OK  ] Reached target Graphical Interface.
Sep 25 16:57:48 localhost.localdomain systemd-nspawn[2888]:          Starting Update UTMP about System Runlev
el Changes...
Sep 25 16:57:48 localhost.localdomain systemd-nspawn[2888]: [  OK  ] Started Update UTMP about System Runleve
l Changes.
Sep 25 16:57:49 localhost.localdomain systemd-nspawn[2888]: [2B blob data]
Sep 25 16:57:49 localhost.localdomain systemd-nspawn[2888]: Fedora 25 (Twenty Five)
Sep 25 16:57:49 localhost.localdomain systemd-nspawn[2888]: Kernel 4.8.0-0.rc7.git0.1.fc25.x86_64 on an x86_6
4 (console)
Sep 25 16:57:49 localhost.localdomain systemd-nspawn[2888]: [1B blob data]
```

CONTROLLING MACHINES

STOPPING CONTAINER

Clean shutdown

```
[root@localhost ~]# machinectl poweroff fedora
```

Immediate termination

```
[root@localhost ~]# machinectl terminate fedora
```

Send signal to leader/all processes

```
[root@localhost ~]# machinectl kill --signal=SIGTERM\
--kill-who=all fedora
```

GETTING INTO MACHINES

SO, I KNOW HOW TO START A CONTAINER IN BACKGROUND.

BUT WHAT IF I WANT TO DO SOMETHING IN IT?

GETTING INTO MACHINES

SYSTEMD INSIDE SYSTEMCTL OUTSIDE

```
[root@localhost ~]# systemctl -M fedora-apache enable --now httpd.service
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service -> /usr/li
b/systemd/system/httpd.service.

[root@localhost ~]# systemctl -M fedora-apache status httpd.service
* httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: di
sabled)
   Active: active (running) since Sun 2016-09-25 21:39:32 CEST; 26s ago
 Main PID: 37
   Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served
/sec:   0 B/sec"
   CGroup: /machine.slice/systemd-nspawn@fedora-apache.service/system.slice/httpd.se
rvice
           |-37 /usr/sbin/httpd -DFOREGROUND
           |-38 /usr/sbin/httpd -DFOREGROUND
           |-39 /usr/sbin/httpd -DFOREGROUND
           |-40 /usr/sbin/httpd -DFOREGROUND
           |-41 /usr/sbin/httpd -DFOREGROUND
           `-42 /usr/sbin/httpd -DFOREGROUND
```

GETTING INTO MACHINES

GETTING SHELL

```
[root@localhost ~]# machinectl shell fedora
Connected to machine fedora. Press ^] three times within 1s to exit
 session.
sh-4.3# logout
```

GETTING INTO MACHINES

LOG IN

```
[root@localhost ~]# machinectl login fedora
Connected to machine fedora. Press ^] three times within 1s to e
xit session.

Fedora 25 (Twenty Five)
Kernel 4.8.0-0.rc7.git0.1.fc25.x86_64 on an x86_64 (pts/0)

fedora login:
```

GETTING INTO MACHINES

RUNNING COMMAND

```
[root@localhost ~]# systemd-run -M fedora /bin/true
Running as unit: run-u9.service
```

INTEGRATION WITH JOURNAL

I THINK MY CONTAINER IS UP TO SOMETHING.
HOW CAN I CHECK WHAT IT HAS BEEN DOING?

INTEGRATION WITH JOURNAL

READING JOURNAL FROM OUTSIDE

```
[root@localhost ~]# journalctl -M fedora
```

Although this only works for running containers.

INTEGRATION WITH JOURNAL

LINKING JOURNAL

Store the journal files of the container on the host and symlink them to container.

```
[root@localhost ~]# systemd-nspawn --link-journal=host -b -M fedora
```

Or just symlink them from the container to the host.

```
[root@localhost ~]# systemd-nspawn --link-journal=guest -b -M fedora
```

And from the host you can now see the logs from the container.

```
[root@localhost ~]# journalctl -m _HOSTNAME=fedora
```

PLAYING WITH FILESYSTEM

WHAT ABOUT READ-ONLY CONTAINERS, ...?

PLAYING WITH FILESYSTEM

EPHEMERAL MODE

Creates a temporary btrfs snapshot on top of image.

```
[root@localhost ~]# systemd-nspawn -x -M fedora
```

PLAYING WITH FILESYSTEM

VOLATILE CONTAINERS

Runs with container as read-only and mounts /var as tmpfs.

```
[root@localhost ~]# systemd-nspawn --volatile=state -M fedora
```

Runs with container as read-only and mounts /var and /etc as tmpfs.

/etc is populated form /usr/share/factory, users from sysusers and systemd-firstboot can be run.

```
[root@localhost ~]# systemd-nspawn --volatile=yes -M fedora
```

PLAYING WITH FILESYSTEM

OVERLAYSFS

```
[root@localhost ~]# mkdir -p /ofs/a
[root@localhost ~]# mkdir -p /ofs/b
[root@localhost ~]# touch /ofs/a/xxx
[root@localhost ~]# touch /ofs/b/yyy
[root@localhost ~]# systemd-nspawn --overlay=/ofs/a:/ofs/b:/root\
-M fedora ls /root
Spawning container fedora on /var/lib/machines/fedora.
Press ^] three times within 1s to kill container.
xxx  yyy
Container fedora exited successfully.
```

PLAYING WITH FILESYSTEM

WHAT WILL THIS DO? (BTRFS ONLY)

```
[root@localhost ~]# systemd-nspawn -D / -xb
```

RUNNING NETWORK SERVICES

I HAVE THIS SERVICE THAT WOULD LIKE TO SPEAK TO OTHERS …

RUNNING NETWORK SERVICES

BY DEFAULT CONTAINER SHARES NETWORK WITH HOST

```
[root@localhost ~]# systemd-nspawn -M net
Spawning container net on /var/lib/machines/net.
Press ^] three times within 1s to kill container.
-bash-4.3# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP grou
p default qlen 1000
    link/ether 52:54:00:ed:37:bf brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.201/24 brd 192.168.122.255 scope global dynamic ens3
       valid_lft 2402sec preferred_lft 2402sec
    inet6 fe80::8ae6:3be:f308:ef73/64 scope link
       valid_lft forever preferred_lft forever
```

RUNNING NETWORK SERVICES

WE CAN DISABLE NETWORK COMPLETELY

```
[root@localhost ~]# systemd-nspawn --private-network -M net
Spawning container net on /var/lib/machines/net.
Press ^] three times within 1s to kill container.
-bash-4.3# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
```

RUNNING NETWORK SERVICES

VIRTUAL DEVICE INSIDE CONTAINER AND ADD IT TO BRIDGE

```
[root@localhost ~]# systemd-nspawn --network-veth -M net ip a
Spawning container net on /var/lib/machines/net.
Press ^] three times within 1s to kill container.
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN grou
p default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: host0@if6: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group
 default qlen 1000
    link/ether 42:9b:d3:19:4a:55 brd ff:ff:ff:ff:ff:ff link-netnsid 0
```

But it will not assign the ip address.

RUNNING NETWORK SERVICES

VIRTUAL DEVICE INSIDE CONTAINER AND ADD IT TO BRIDGE + NETWORKD INSIDE

How to get ip

```
[root@localhost ~]# machinectl status net | grep Address
     Address: 10.0.0.9
```

Or just use nss_mymachines

```
[root@localhost ~]# grep mymachines /etc/nsswitch.conf
hosts:      mymachines files mdns4_minimal [NOTFOUND=return] dns myhostname
[root@localhost ~]# curl fedora-apache
```

RUNNING NETWORK SERVICES

ACCESSING THE SERVICE IN CONTAINER FROM OUTSIDE

```
[root@localhost ~]# systemd-nspawn -b -M fedora-apache \
--network-veth --port 'tcp:80:80'
```

Or use macvlan

```
[root@localhost ~]# systemd-nspawn -b -M fedora-apache --network-macvlan=ens3
```

Although this also needs a new configuration file for network in the container.

RUNNING NETWORK SERVICES

SOCKET ACTIVATION

SYSTEMD ON THE HOST LISTEN FOR INCOMING CONNECTION

ON INCOMING CONNECTION SYSTEMD STARTS THE CONTAINER

SYSTEMD ON THE HOST HANDS OVER THE CONNECTION TO SYSTEMD INSIDE THE CONTAINER

SYSTEMD IN THE CONTAINER START THE SERVICE

SYSTEMD IN THE CONTAINER HANDS OVER THE CONNECTION TO THE SERVICE

COMPILING WITH NSPAWN

YOU'VE MENTIONED THAT YOU USE NSPAWN FOR DEVELOPING AND TESTING?

COMPILING WITH NSPAWN

DUMMY PROJECT

```c
#include <stdlib.h>
#include <stdio.h>

int main (void) {
        printf("Hello world!\n");
        return EXIT_SUCCESS;
}
```

```
#!/bin/bash
make all
make install
hello
make clean
```

```
hello: hello.c
        gcc -o hello hello.c

all: hello

install: hello
        mkdir -p $(DESTDIR)/usr/bin/
        install -m755 hello $(DESTDIR)/usr/bin/

clean:
        rm hello
```

COMPILING WITH NSPAWN

QUICK&DIRTY

```
[root@localhost ~]# systemd-nspawn -M centos-gcc \
--bind /project\
--as-pid2\
--chdir=/project\
/project/test.sh
Spawning container centos-gcc on /var/lib/machines/centos-gcc.
Press ^] three times within 1s to kill container.
make: Nothing to be done for `all'.
mkdir -p /usr/bin/
install -m755 hello /usr/bin/hello
Hello world!
rm hello
Container centos-gcc exited successfully.
```

COMPILING WITH NSPAWN

MKOSI - SLOW&CLEAN

```
mkosi.default:
[Distribution]
Distribution=fedora
Release=25

[Output]
Format=raw_btrfs
Bootable=yes

[Packages]
BuildPackages=make gcc
```

```
mkosi.build:
#!/bin/sh
make all
make install
```

```
[root@localhost project]# mkosi
```

Go for coffee.

```
[root@localhost project]# systemd-nspawn -i image.raw
```

*.NSPAWN CONFIGURATION FILES

YOU KNOW THAT TYPING LONG COMMANDS TO TERMINAL IS BORING, RIGHT?

*.NSPAWN CONFIGURATION FILES

/etc/systemd/nspawn/*machine*.nspawn
/run/systemd/nspawn/*machine*.nspawn
/var/lib/machines/*machine*.nspawn

So instead systemd-nspawn -M centos-gcc --bind /project --as-pid2 --chdir=/project /project/test.sh

```
/etc/systemd/nspawn/centos-gcc.nspawn:
[Exec]
ProcessTwo=yes
Parameters=/project/test.sh
WorkingDirectory=/project

[Files]
Bind=/project
```

EXPORTING IMAGES

I HAVE THIS AWESOME IMAGE.
HOW CAN I SHARE IT WITH OTHERS?

EXPORTING IMAGES

Directories and btrfs subvolumes

```
[root@localhost ~]# machinectl export-tar fedora ~/fedora.tar.gz
```

Raw images

```
[root@localhost ~]# machinectl export-raw fedora-raw ~/fedora.raw
```

"Migrating" image

```
[root@localhost ~]# machinectl export-tar foo | \
ssh someothermachine machinectl import-tar foo
```

THANKS FOR YOUR ATTENTION!