Building initrd images from rpms

Zbigniew Jędrzejewski-Szmek

Red Hat

zbyszek@in.waw.pl

Nest with Fedora, 7.8.2021

Why do we need an initrd?

The initrd is a small file system that the boot loader passes to the kernel.

Why do we need an initrd?

The initrd is a small file system that the boot loader passes to the kernel.

The primary purpose of the initrd is to mount the real root file system.

Why do we need an initrd?

The initrd is a small file system that the boot loader passes to the kernel.

The primary purpose of the initrd is to mount the real root file system.

initrd == initramfs

Status quo — dracut

Dracut — “generic initramfs intrastructure”

- configuration mechanism for deciding what is available in the initrd image (also a dependency mechanism with `check()`, `depends()`)
- create the image from files on the host (`instmods()`, `dracut_install()`, `inst()`, `inst_hook()`, `inst_rules()`)
- event-driven execution queue in initrd
- helpers to do various things in the initrd

Status quo — dracut

Dracut — “generic initramfs intrastructure”

- configuration mechanism for deciding what is available in the initrd image (also a dependency mechanism with `check()`, `depends()`)
- create the image from files on the host (`instmods()`, `dracut_install()`, `inst()`, `inst_hook()`, `inst_rules()`)
- event-driven execution queue in initrd
- helpers to do various things in the initrd

Dracut (optionally) installs systemd and udev into the initrd.

Status quo — dracut

Dracut — “generic initramfs intrastructure”

- configuration mechanism for deciding what is available in the initrd image (also a dependency mechanism with `check()`, `depends()`)
- create the image from files on the host (`instmods()`, `dracut_install()`, `inst()`, `inst_hook()`, `inst_rules()`)
- event-driven execution queue in initrd
- helpers to do various things in the initrd

Dracut (optionally) installs systemd and udev into the initrd.
Systemd is written to support running in the initrd.
Systemd provides services to do all the things that the initrd does.

Status quo — dracut

Dracut — “generic initramfs intrastructure”

- configuration mechanism for deciding what is available in the initrd image (also a dependency mechanism with `check()`, `depends()`)
- create the image from files on the host (`instmods()`, `dracut_install()`, `inst()`, `inst_hook()`, `inst_rules()`)
- event-driven execution queue in initrd
- helpers to do various things in the initrd

Dracut (optionally) installs systemd and udev into the initrd.
Systemd is written to support running in the initrd.
Systemd provides services to do all the things that the initrd does.

Why wrap systemd in another execution queue?

One step back: why is the initrd so special?

As far as the kernel is concerned, the initrd is just another file system (`/init`, `/etc/initrd-release`)

One step back: why is the initrd so special?

As far as the kernel is concerned, the initrd is just another file system (`/init`, `/etc/initrd-release`)

(We switch back into the initrd image for shutdown.)

One step back: why is the initrd so special?

As far as the kernel is concerned, the initrd is just another file system (`/init`, `/etc/initrd-release`)

(We switch back into the initrd image for shutdown.)

Anything we would do in the initrd, we also need to do in the host:
storage
degraded storage
networking
network-based file systems and storage (nfs, iscsi, clevis)
fsck
emergency mode

One step back: why is the initrd so special?

As far as the kernel is concerned, the initrd is just another file system (`/init`, `/etc/initrd-release`)

(We switch back into the initrd image for shutdown.)

Anything we would do in the initrd, we also need to do in the host:
storage
degraded storage
networking
network-based file systems and storage (nfs, iscsi, clevis)
fsck
emergency mode

Nowadays all this functionality is implemented either using daemons and/or systemd units and/or various helpers.

Overview of the new scheme

Dracut:

- configuration mechanism...
- create the image from files on the host
- event-driven execution queue...
- helpers ... in the initrd

Overview of the new scheme

Dracut:

- configuration mechanism...
- create the image from files on the host
- event-driven execution queue...
- helpers ... in the initrd

- (some list of rpms)

Overview of the new scheme

Dracut:

- configuration mechanism…
- create the image from files on the host
- event-driven execution queue…
- helpers … in the initrd

- (some list of rpms)
- `dnf --installroot=… && cpio --create …`

Overview of the new scheme

Dracut:

- configuration mechanism…
- create the image from files on the host
- event-driven execution queue…
- helpers … in the initrd

- (some list of rpms)
- `dnf --installroot=… && cpio --create …`
- `systemd FTW!`

Overview of the new scheme

Dracut:

configuration mechanism...	(some list of rpms)
create the image from files on the host	`dnf --installroot=... && cpio --create ...`
event-driven execution queue...	`systemd FTW!`
helpers ... in the initrd	POOR-solution

Overview of the new scheme

Dracut:

- configuration mechanism…
- create the image from files on the host
- event-driven execution queue…
- helpers … in the initrd

- (some list of rpms)
- `dnf --installroot=… && cpio --create …`
- `systemd FTW!`
- POOR-solution
  plain old ordinary rpms

Some questions that could in principle be asked frequently

Q: My rpm is too large?!

Some questions that could in principle be asked frequently

Q: My rpm is too large?!

A: Split it up!

Container folks, cloud users, live cd creators, flatpak consumers, IoT engineers will all thank you.

Some questions that could in principle be asked frequently

Q: My rpm is too large?!

A: Split it up!
Container folks, cloud users, live cd creators, flatpak consumers, IoT engineers will all thank you.

Q: My rpm has too many dependencies?!

A: …
`https://docs.fedoraproject.org/en-US/minimization/`

Some questions that could in principle be asked frequently

Q: My rpm is too large?!
A: Split it up!
Container folks, cloud users, live cd creators, flatpak consumers, IoT engineers will all thank you.

Q: My rpm has too many dependencies?!
A: …
`https://docs.fedoraproject.org/en-US/minimization/`

Q: My rpm requires special setup / doesn't work in the initrd…
A: …

Why I think it's good to build the image directly from packages

- reliable installation: rpm is very good at doing what it does
- normal dependency mechanism
- we don't pull files from the host
- images are reproducible
- developers don't need to learn another system
- bash helpers $\rightarrow$ compiled programs
- clear ownership of bugs
- any improvements are immediately shared

Current implementation — ...

Current implementation — mkosi-initrd

mkosi is a python program to build images from rpms
mkosi-initrd uses mkosi to create a .cpio.zstd archive

Current implementation — mkosi-initrd

mkosi is a python program to build images from rpms
mkosi-initrd uses mkosi to create a .cpio.zstd archive

Some alternatives:

- osbuild
- kiwi-ng

Demo?

```
sudo mkosi -f -o initrd.cpio.zstd sumary
sudo mkosi -f -o initrd.cpio.zstd
```

Demo?

```
sudo mkosi -f -o initrd.cpio.zstd sumary
sudo mkosi -f -o initrd.cpio.zstd

KVER=5.13.0-0.rc2.19.fc35.x86_64
sudo mkosi --build-env=KERNEL_VERSION=$KVER -f -o initrd-$
```

Size comparison

```
$ du -sh dracut-*.cpio.* mkosi-*.cpio.*
34M     dracut-5.13.4-200.fc34.x86_64.cpio.xz
62M     mkosi-5.13.4-200.fc34.x86_64.cpio.zstd
```

Size comparison

```
$ du -sh dracut-*.cpio.* mkosi-*.cpio.*
34M     dracut-5.13.4-200.fc34.x86_64.cpio.xz
62M     mkosi-5.13.4-200.fc34.x86_64.cpio.zstd
$ du -sh dracut-*.d/ mkosi-*.d/
77M     dracut-5.13.4-200.fc34.x86_64.d
165M    mkosi-5.13.4-200.fc34.x86_64.d
```

Size comparison

```
$ du -sh dracut-*.cpio.* mkosi-*.cpio.*
34M     dracut-5.13.4-200.fc34.x86_64.cpio.xz
62M     mkosi-5.13.4-200.fc34.x86_64.cpio.zstd
$ du -sh dracut-*.d/ mkosi-*.d/
77M     dracut-5.13.4-200.fc34.x86_64.d
165M    mkosi-5.13.4-200.fc34.x86_64.d
```

Some differences:
/lib/modules 5 MB vs. 37 MB
/usr/bin 8 MB vs. 18 MB
/usr/sbin 10 MB vs. 14 MB
/usr/lib64 41 MB vs. 51 MB
/usr/share 0.5 MB vs. 11 MB
(…/licenses 3 MB, …/zoneinfo 5 MB, …/pki 1 MB, …/terminfo 1 MB)
/etc 0.5 MB vs. 12 MB
(…/udev/hwdb.bin 9MB, …/pki 1 MB)

Host-specific and Generic images

We build images on the host to be able to customize.
Locally-built images are incompatible with centralized signing for Secure Boot.

Host-specific and Generic images

We build images on the host to be able to customize.
Locally-built images are incompatible with centralized signing for Secure Boot.

With images built from rpms it makes even less sense:
everybody does the work, the result is always the same.

Host-specific and Generic images

We build images on the host to be able to customize.
Locally-built images are incompatible with centralized signing for Secure Boot.

With images built from rpms it makes even less sense:
everybody does the work, the result is always the same.

We could easily build one huge image with “everything”, but it would be slow and boot partitions are small.

We need a mechanism to extend/customize images.

Digression: systemd-sysext

DEMO!!!

Building sysexts (with mkosi?)

1. Mount an initramfs image somewhere
2. Mount an OverlayFS over it (upper layer empty)
3. `dnf install --installroot=… <packages for sysext>`
4. Create a file system image with upper layer only
5. (Optionally create partition dm-verity hash for it)
6. (Optionally sign the whole thing)

~~Host-specific and~~ Generic and Host-customized images

We need a mechanism to extend/customize images.

~~Host-specific and~~ Generic and Host-customized images

We need a mechanism to extend/customize images.

sysexts can be loaded into the initramfs image.

~~Host-specific and~~ Generic and Host-customized images

We need a mechanism to extend/customize images.

sysexts can be loaded into the initramfs image.

Distro-side: the distro builds the kernel, and the initrd,
and some set of sysexts
(all three are signed)

~~Host-specific and~~ Generic and Host-customized images

We need a mechanism to extend/customize images.

sysexts can be loaded into the initramfs image.

Distro-side: the distro builds the kernel, and the initrd,
and some set of sysexts
(all three are signed)

The boot loader / firmware verifies the kernel+initrd combo
The initrd checks and loads sysexts
The kernel verifies sysext images using dm-verity
(plenty of details TBD)

Should `initrd-5.13.4-200.fc34.x86_64.rpm` specify
`Requires:systemd>=<version used>`?

What works?

OK:
Fedora Server in QEmu with direct kernel boot
My laptop (Fedora Workstation Lenovo X1)
LVM
LUKS
emergency mode without authentication
resume

Never tested:
iscsi, fcoe, nfs, nbd, kdump, network syntax (ip=/ifname=/rd.route=/...) supported by dracut and systemd-network-generator, plymouth, network, raid, sshd, bluetooth, netconsole

Requires future work:
integration with systemd-repart?
switching back to initramfs for shutdown
firmware

TODO list

mkosi: cpio and zstd (#728) ✓
RemoveFiles= (#744) ✓
Release (v10) ✓

various systemd fixes (249+ should be OK)

split modules out of `kernel-core.rpm`

figure out how to deal with extensions
figure out how to authenticate root for troubleshooting

`kernel-install` plugin to call mkosi-initrd

mkosi support of sysext image creation

TODO list: minimization

Requires:dbus→Recommends:dbus in systemd

Prune from the initrd:
`pcre` (we have `pcre2`),
`libcap` (we have `libcap-ng`),
`shadow-utils` (users are pre-created),
`util-linux` (we have `util-linux-core`),
`fedora-repos` (images are static),
`alternatives` (wtf?),
`tzdata`

Port `systemd` over to `openssl`, drop `libgcrypt`, `libgpg-error`
Drop `polkit`?
Do something with identical license texts?

Summary

Build initramfs images directly from system packages
Let systemd do the heavy lifting in the initrd
Do things in the initrd like on the host
Extend the initrd image using systemd-ext/OverlayFS
(Build initrd images and extensions in koji)
(Sign and verify all individual compoments)

Links

https://github.com/systemd/mkosi
https://github.com/keszybz/mkosi-initrd
https://www.freedesktop.org/software/systemd/man/systemd-sysext.html
https://gitlab.com/cryptsetup/cryptsetup/-/wikis/DMVerity
https://www.kernel.org/doc/html/latest/admin-guide/device-mapper/verity.html
https://www.kernel.org/doc/html/latest/filesystems/overlayfs.html
These slides:
https://github.com/keszybz/mkosi-initrd-talk