

NAME

dracut.modules – dracut modules

DESCRIPTION

dracut uses a modular system to build and extend the initramfs image. All modules are located in `/usr/lib/dracut/modules.d` or in `<git-src>/modules.d`. The most basic dracut module is *80base*. In *80base* the initial shell script `init` is defined, which gets run by the kernel after initramfs loading. Although you can replace `init` with your own version of *80base*, this is not encouraged. Instead you should use, if possible, the hooks of dracut. All hooks, and the point of time in which they are executed, are described in **BOOT PROCESS STAGES**.

The main script, which creates the initramfs is dracut itself. It parses all arguments and sets up the directory, in which everything is installed. It then executes all `check`, `install`, `installkernel` scripts found in the modules, which are to be processed. After everything is installed, the `install` directory is archived and compressed to the final initramfs image. All helper functions used by `check`, `install` and `installkernel` are found in the file *dracut-functions*. These shell functions are available to all module installer (`install`, `installkernel`) scripts, without the need to source *dracut-functions*.

A module can check the preconditions for `install` and `installkernel` with the `check` script. Also dependencies can be expressed with `check`. If a module passed `check`, `install` and `installkernel` will be called to install all of the necessary files for the module. To split between kernel and non-kernel parts of the installation, all kernel module related parts have to be in `installkernel`. All other files found in a module directory are module specific and mostly are hook scripts and udev rules.

BOOT PROCESS STAGES

dracut modules can insert custom script at various points, to control the boot process. These hooks are plain directories containing shell scripts ending with ".sh", which are sourced by `init` or other scripts. Since hook scripts are sourced instead of directly executed, "return" must be used instead of "exit" at the end of a hook to avoid having dracut quit prematurely. Please unset any used variables (or, better yet, use local variables within functions) to avoid influencing subsequent code. Common used functions are in *dracut-lib.sh*, which can be sourced by any script.

dracut looks for custom hook scripts in subdirectories (`cmdline`, `pre-udev`, `pre-trigger`, `initqueue`, `pre-mount`, `mount`, `pre-pivot`, `cleanup`) of the following locations: `/var/lib/dracut/hooks`, `/etc/dracut/hooks`, and `/usr/lib/dracut/hooks`. The intended use of these locations is: standard, distribution shipped scripts are put to `/usr/lib/dracut/hooks`; `/etc/dracut/hooks` is used as a local override; dracut modules create (and remove) scripts in runtime in `/var/lib/dracut/hooks`. If a hook with the same name exists in multiple locations simultaneously, the most privileged location (`/var/lib/dracut/hooks`, then `/etc/dracut/hooks`, and then `/usr/lib/dracut/hooks`) wins and only one instance of the hook is executed.

Hook: cmdline

The *cmdline* hook is a place to insert scripts to parse the kernel command line and prepare the later actions, like setting up udev rules and configuration files.

In this hook the most important environment variable is defined: `root`. The second one is `rootok`, which indicates, that a module claimed to be able to parse the root defined. So for example, `root=iscsi:....` will be claimed by the `iscsi` dracut module, which then sets `rootok`.

Hook: pre-udev

This hook is executed right after the `cmdline` hook and a check if `root` and `rootok` were set. Here modules can take action with the final `root`, and before udev has been run.

Start Udev

Now udev is started and the logging for udev is setup.

Hook: pre-trigger

In this hook, you can set udev environment variables with `udevadm control --property=KEY=value` or control the further execution of udev with `udevadm`.

Trigger Udev

udev is triggered by calling `udevadm trigger`, which sends add events for all devices and subsystems.

Main Loop

In the main loop of dracut loops until udev has settled and all scripts in *initqueue/finished* returned true. In this loop there are three hooks, where scripts can be inserted by calling `/sbin/initqueue`.

Initqueue

This hook gets executed every time a script is inserted here, regardless of the udev state.

Initqueue settled

This hook (`initqueue/settled`) gets executed every time udev has settled.

Initqueue timeout

This hook (`initqueue/timeout`) gets executed, when the main loop counter becomes half of the `rd.retry` counter.

Initqueue online

This hook (`initqueue/online`) gets executed whenever a network interface comes online (that is, once it is up and configured by the configured network module).

Initqueue finished

This hook (`initqueue/finished`) is called after udev has settled and if all scripts herein return 0 the main loop will be ended. Arbitrary scripts can be added here, to loop in the `initqueue` until something happens, which a dracut module wants to wait for.

Hook: pre-mount

Before the root device is mounted all scripts in the hook `pre-mount` are executed. In some cases (e.g. NFS) the real root device is already mounted, though.

Hook: mount

This hook is mainly to mount the real root device.

Hook: pre-pivot

This hook is called before cleanup hook, This is a good place for actions other than cleanups which need to be called before pivot.

Hook: cleanup

This hook is the last hook and is called before `init` finally switches root to the real root device. This is a good place to clean up and kill processes not needed anymore.

Cleanup and switch_root

`Init` (or `systemd`) kills all udev processes, cleans up the environment, sets up the arguments for the real `init` process and finally calls `switch_root`. `switch_root` removes the whole filesystem hierarchy of the `initramfs`, `chroot()`s to the real root device and calls `/sbin/init` with the specified arguments.

To ensure all files in the `initramfs` hierarchy can be removed, all processes still running from the `initramfs` should not have any open file descriptors left.

NETWORK INFRASTRUCTURE

FIXME

WRITING A MODULE

A simple example module is *90kernel-modules*, which `modprobes` a kernel module after udev has settled and the basic device drivers have been loaded.

All module installation information is in the file `module-setup.sh`.

The following are the most commonly used shell variables used inside dracut modules. These shell variables are always declared and assigned values (the assigned value might be an empty string). There is a compatibility promise on these variables between dracut releases. Module should only read (and not write) these variables. `* $DRACUT_ARCH` (not an empty string) `* $hostonly` (could be an empty string) `*`

```
$hostonly_mode (could be an empty string) * $hostonly_cmdline (not an empty string) *
$moddir (not an empty string) * $hookdir (not an empty string)
```

First we create a `check()` function, which just exits with 0 indicating that this module should be included by default.

```
check():
```

```
return 0
```

Then we create the `install()` function, which installs a `cmdline` hook with priority number 20 called `parse-insmodpost.sh`. It also installs the `insmodpost.sh` script in `/sbin`.

```
install():
```

```
inst_hook cmdline 20 "$moddir/parse-insmodpost.sh"
inst_simple "$moddir/insmodpost.sh" /sbin/insmodpost.sh
```

The `parse-insmodpost.sh` parses the kernel command line for a argument `rd.driver.post`, blacklists the module from being autoloaded and installs the hook `insmodpost.sh` in the `initqueue/settled`.

```
parse-insmodpost.sh:
```

```
for p in $(getargs rd.driver.post=); do
    echo "blacklist $p" >> /run/modprobe.d/initramfsblacklist.conf
    _do_insmodpost=1
done
```

```
[ -n "$_do_insmodpost" ] && /sbin/initqueue --settled --unique --onetime /sbin/insmo
unset _do_insmodpost
```

`insmodpost.sh`, which is called in the `initqueue/settled` hook will just `modprobe` the kernel modules specified in all `rd.driver.post` kernel command line parameters. It runs after `udev` has settled and is only called once (`--onetime`).

```
insmodpost.sh:
```

```
. /lib/dracut-lib.sh
```

```
for p in $(getargs rd.driver.post=); do
    modprobe $p
done
```

module-setup.sh: check()

`check()` is called by dracut to evaluate the inclusion of a dracut module in the `initramfs`.

```
$hostonly
```

If the `$hostonly` variable is set, then the module `check()` function should be in "hostonly" mode, which means, that the `check()` should only return 0, if the module is really needed to boot this specific host.

Avoid making decisions inside this function based on the value of the `$hostonly_mode` variable. This principle ensures the inclusion of the same dracut modules in both sloppy and strict modes.

`check()` should return with:

0

Include the dracut module in the initramfs.

1

Do not include the dracut module. The requirements are not fulfilled (missing tools, etc.)

255

Only include the dracut module, if another module requires it or if explicitly specified in the config file or on the argument list.

module-setup.sh: depends()

The function depends() should echo all other dracut module names the module depends on.

module-setup.sh: cmdline()

This function should print the kernel command line options needed to boot the current machine setup. It should start with a space and should not print a newline.

module-setup.sh: install()

The install() function is called to install everything non-kernel related. To install binaries, scripts, and other files, you can use the functions mentioned in [creation].

To address a file in the current module directory, use the variable "\$moddir".

module-setup.sh: installkernel()

In installkernel() all kernel related files should be installed. You can use all of the functions mentioned in [creation] to install files.

Creation Functions

inst_dir <dir>

installs a directory <dir> (but not its content) to the same place in the initramfs image.

inst [-H] <src> [<dst>]

inst_binary <src> [<dst>]

inst_script <src> [<dst>]

installs *one* file <src> either to the same place in the initramfs or to an optional <dst>, and also its dependencies and .hmac file (if it exists). inst with more than two arguments is treated the same as inst_multiple, all arguments are treated as files to install and none as install destinations.

options:

-H

log all installed files to */lib/dracut/hostonly-files*, so they can be removed if **rd.hostonly** is passed on the kernel command line.

inst_simple [-H] <src> [<dst>]

installs *one* file <src> either to the same place in the initramfs or to an optional <dst>, but without installing its dependencies or .hmac file.

options:

-H

log all installed files to */lib/dracut/hostonly-files*, so they can be removed if **rd.hostonly** is passed on the kernel command line.

inst_multiple [-H] [-o] <file> [<file> ...]

installs multiple binaries and files. If executables are specified without a path, dracut will search the path PATH=/usr/sbin:/sbin:/usr/bin:/bin for the binary.

options:

-H
log all installed files to `/lib/dracut/hostonly-files`, so they can be removed if **rd.hostonly** is passed on the kernel command line.

-o
optional, a missing file does not lead to an error.

inst_hook <hookdir> <prio> <src>
installs an executable/script <src> in the dracut hook <hookdir> with priority <prio>.

inst_rules <udevrule> [<udevrule> ...]
installs one or more udev rules. Non-existent udev rules are reported, but do not let dracut fail.

inst_sysusers <sysuserconf>
installs one sysuser config files. Non-existent sysuser config file is reported, but does not let dracut fail.

inst_libdir_dir <dir> [<dir>...]
installs multiple directories (but not their content) located on a library directory to the initramfs image.

inst_libdir_file [-n <pattern>] <file> [<file>...]
installs multiple files located on a library directory to the initramfs image.

options:

-n <pattern>
install all library files matching a pattern.

instmods [-c] [-s] <kernelmodule> [<kernelmodule> ...]
instmods should be used only in the `installkernel()` function.

instmods installs one or more kernel modules in the initramfs. <kernelmodule> can also be a whole subsystem, if prefixed with a "=", like "=drivers/net/team".

instmods will not install the kernel module, if `$hostonly` is set and the kernel module is not currently needed by any `/sys/.../uevent MODALIAS`. To install a kernel module regardless of the hostonly mode use the form:

```
hostonly='' instmods <kernelmodule>
```

a common pattern to call instmods is to use the following form:

```
'hostonly=$(optional_hostonly) instmods <kernelmodule>'
```

This statement installs the kernel module in either of the following conditions: * in non-hostonly mode (regardless of the host state) * in sloppy hostonly mode (regardless of the host state) * in hostonly mode if the kernel module is already loaded in the host

options:

-c
check that kernel modules exists and can be installed (i.e., not optional).

-s
reduce verbosity.

dracut_need_initqueue

Obsolete function as of dracut v108 that expresses a dependency on the initqueue. Instead of using this function, add a dracut module dependency for the initqueue dracut module. This function is maintained only to provide compatibility for modules aimed at older dracut releases.

require_binaries

Helper function for check() in module-setup.sh to check for required installed binaries. All of the binaries listed in the arguments of this function should exist in the host. Issues a standardized warning message.

require_any_binary

Helper function for check() in module-setup.sh to check for required installed binaries. At least one of the binaries listed in the arguments of this function should exist in the host. Issues a standardized warning message.

require_kernel_modules

Helper function for check() in module-setup.sh to check for required kernel modules. Issues a standardized warning message.

Initramfs Functions**determine_kernel_image <kversion>**

Helper function for determining the path for the kernel image file.

Network Modules

FIXME

AUTHOR

Harald Hoyer

SEE ALSO

man:dracut[8]