IEN-006 - Stage 5 — Mounting the Root Filesystem

After the initrd (or initramfs) has completed its work, the system finally reaches one of the most important milestones in the entire boot process:

> **The real root filesystem can now be mounted.**

Everything that happened before this point—from firmware initialization to loading storage drivers—had one ultimate goal:

Make the root filesystem accessible.

Only after the root filesystem is mounted can Linux begin executing the programs that make up the operating system itself.

What Is the Root Filesystem?

Every Linux system has a directory called:

```
/
```

This directory is known as the **root filesystem**.

It is the highest level of the Linux directory hierarchy.

Everything else exists beneath it.

For example:

```
/
├── bin
├── boot
├── dev
├── etc
├── home
├── lib
├── proc
├── root
├── run
├── sbin
├── sys
├── tmp
├── usr
└── var
```

Every application, configuration file, shared library, and executable ultimately resides somewhere under this hierarchy.

Without the root filesystem, Linux has nowhere to find:

- `/bin/bash`
- `/etc/fstab`
- `/usr/bin`
- `/lib`
- `/sbin/init`

At this stage, these files still do **not** exist in the running environment.

The system is still operating from the temporary initrd.

Temporary Root vs Real Root

One concept that often confuses Linux administrators is the existence of **two different root filesystems** during boot.

The first is the temporary root filesystem provided by initrd.

The second is the permanent root filesystem stored on disk.

The temporary root exists only to make the permanent root accessible.

Once the permanent root is ready, the temporary one is discarded.

Engineering Insight

Many engineers assume that the `/` directory they see after logging in has existed since the kernel started.

In reality, the first `/` belongs to the initrd environment. The root filesystem you interact with after the system boots is a completely different filesystem that replaces the temporary one through the `switch_root` process.

Understanding this transition makes it much easier to troubleshoot early boot failures.

The Virtual Filesystem (VFS)

Before mounting any filesystem, Linux uses an abstraction layer called the **Virtual Filesystem (VFS)**.

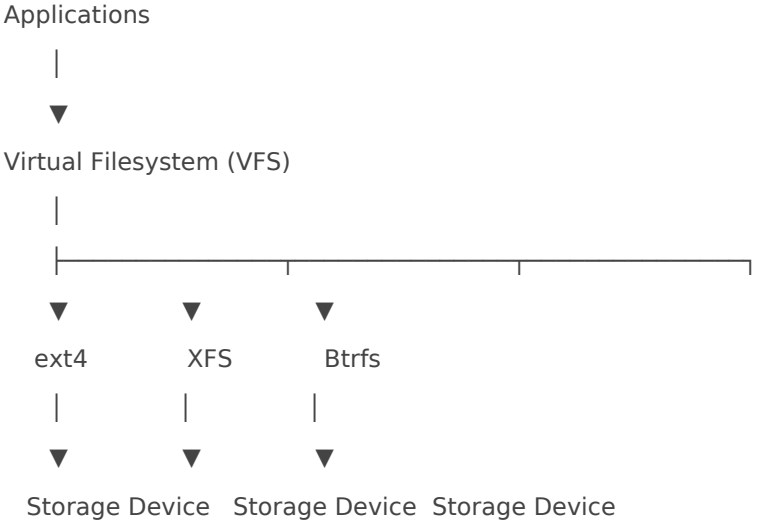
The VFS allows the kernel to interact with different filesystem types through a common interface.

Whether the underlying filesystem is:

- ext4
- XFS
- Btrfs
- ext3
- ext2

the kernel uses the same internal API.

A simplified architecture looks like this:



This abstraction allows Linux to support many filesystem types without requiring applications to know which one is in use.

Finding the Root Filesystem

At this point, the initrd environment has already activated storage devices such as:

- LVM
- RAID
- Multipath

- SAN
- NVMe

The kernel now attempts to locate the root filesystem specified by the kernel command line.

For example:

```
root=/dev/sda2
```

or

```
root=/dev/mapper/vg_root-lv_root
```

or

```
root=UUID=31fd66b5-dfd6-4d96-9c65-aab7c6d79e5d
```

The kernel resolves this parameter to an actual block device.

If the device exists and the required filesystem driver is available, the filesystem can be mounted.

Why UUID Is Preferred

Modern Linux systems usually avoid using device names such as:

```
/dev/sda1
```

Instead they use:

```
UUID=31fd66b5-dfd6-4d96-9c65-aab7c6d79e5d
```

or

```
LABEL=rootfs
```

Why?

Because device names are not guaranteed to remain consistent.

Adding another disk may change:

```
/dev/sda
```

into

```
/dev/sdb
```

which could prevent the system from booting.

UUIDs remain stable regardless of device ordering.

Mounting the Filesystem

Once the root device has been located, Linux mounts it.

Initially, many distributions mount the filesystem as:

```
read-only
```

This allows the kernel to perform consistency checks before permitting write operations.

Later during the boot process, it is remounted as:

```
read-write
```

This transition often occurs before system services begin starting.

The switch_root Operation

After the root filesystem has been mounted successfully, Linux reaches another critical milestone.

The temporary initrd environment has fulfilled its purpose.

Execution must now continue from the real operating system.

This transition is performed by:

```
switch_root
```

The sequence looks like this:

```
Kernel
    |
    ▼
Temporary Root
    |
    ▼
Mount Real Root
    |
    ▼
switch_root
    |
    ▼
Execute /sbin/init
```

The `switch_root` operation replaces the temporary root filesystem with the permanent one.

From this point onward:

- `/etc`
- `/usr`
- `/var`
- `/home`
- `/bin`

all come from the real operating system installed on disk.

The initrd environment disappears completely.

What Happens If Mounting Fails?

Failure at this stage is one of the most common causes of boot problems.

Typical reasons include:

- Incorrect `root=` parameter.
- Missing filesystem driver.
- Corrupted filesystem.
- Inactive LVM volume.
- Missing RAID device.
- Damaged storage.
- Incorrect UUID.

Depending on the failure, the system may display messages such as:

```
VFS: Cannot open root device
```

or

```
Kernel panic - not syncing:
VFS: Unable to mount root fs
```

or

```
Waiting for root device...
```

Although these messages mention the kernel, the underlying issue is frequently related to storage discovery rather than a defect in the kernel itself.

Engineering Insight

One of the easiest mistakes to make during recovery is focusing only on the final error message.

For example, if the system reports **"Unable to mount root filesystem"**, it is tempting to assume the filesystem is corrupted.

However, the filesystem may never have been reached.

The real issue could be:

- the LVM volume was never activated,
- the RAID array was incomplete,
- the storage driver was missing,
- or the kernel was looking for the wrong root device.

Always verify that the expected block device actually exists before attempting filesystem repair.

Verifying the Root Device

During recovery or rescue mode, several commands can help confirm that the expected root device is available.

```
lsblk
```

Displays the hierarchy of block devices and mounted filesystems.

```
blkid
```

Shows UUIDs and filesystem types.

```
cat /proc/cmdline
```

Displays the kernel command line, including the `root=` parameter used during boot.

```
mount
```

Shows which filesystems are currently mounted.

```
cat /proc/mounts
```

Lists active mount points from the kernel's perspective.

These commands are often the first step when investigating boot issues involving storage or root filesystem discovery.

Summary

The goal of the previous stages was never simply to start the kernel—it was to make the real root filesystem available.

Once the root filesystem has been mounted, Linux leaves the temporary initrd environment behind and transitions to the permanent operating system through `switch_root`.

From this moment onward, the kernel can begin executing programs stored on the real filesystem, starting with the very first userspace process.

This process, known as **PID 1**, is responsible for initializing the rest of the operating system and will be the focus of the next stage.

Revision #1
Created 4 July 2026 04:18:50 by Kapak Maut
Updated 4 July 2026 04:20:20 by Kapak Maut