

RK3568开发笔记-buildroot挂载overlay联合文件系统

前言

AI提问 评论 笔记

在本篇博客中，我们将探讨如何使用 [Buildroot](#) 构建定制的 [Linux](#) 发行版，并通过挂载 [Overlay](#) 联合文件系统来实现在 RK3568 处理器上运行一个精简的 Linux 环境。本文主要介绍在只读的buildroot文件系统中如何搭建overlay联合文件系统，在不改变底层跟文件系统的前提下提供一个上层可读写的环境，overlay联合文件系统应用典型就是openwrt系统，在保护根文件系统的基础上可以在上层文件系统进行读写操作，并支持系统一键还原操作。对于需要时常还原的服务器有着广泛的应用。

一、什么是overlay?

Overlay 文件系统是一种在 Linux 中实现文件系统层叠的机制，允许将一个或多个文件系统层叠在另一个文件系统之上。这种机制允许您在不修改原始根文件系统的情况下，向其添加、修改或覆盖文件和目录。Overlay 文件系统通常用于构建[嵌入式系统](#)中的可定制化和动态变化的部分。

以下是 Overlay 文件系统的一些重要特点和优势：

- 不侵入性：** Overlay 文件系统不会修改原始根文件系统。它通过在挂载点层叠一个或多个文件系统来实现变更，因此不会影响原始文件系统的完整性。
- 层叠结构：** Overlay 文件系统由多个层组成，每个层都可以包含文件、目录和符号链接。这些层可以被逐层叠加，最终形成一个虚拟文件系统。
- 只读和读写层：** 通常，Overlay 文件系统由只读的底层（通常是原始根文件系统）和读写的上层（用于覆盖和添加内容）组成。
- 变更隔离：** 由于只读底层不会受到修改的影响，因此您可以随时替换或更新只读底层，而不会影响上层的定制内容。
- 运行时修改：** Overlay 文件系统允许在系统运行时添加、修改或删除文件和目录，而无需重新启动。
- 快照和回滚：** 由于层叠结构，您可以轻松地创建 Overlay 文件系统的快照，并在需要时进行回滚。
- 动态定制化：** Overlay 文件系统非常适合嵌入式系统，因为它允许根据不同的用例和应用需求，在同一[硬件](#)平台上创建不同的定制化配置。
- 资源节省：** Overlay 文件系统不会占用大量的额外存储空间，因为只有更改部分被添加到上层。
- 应用场景：** Overlay 文件系统常用于嵌入式系统中，例如在根文件系统上添加用户自定义配置、应用程序数据或插件功能。

二、挂载操作

在buildroot文件系统/etc/init.d/文件夹下新建S22overfs.sh脚本，实现overlay文件系统自动挂载，脚本内容如下所示：

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:      mount-overlayfs
```

```

# Default-Start:  S
# Default-Stop:
# Description:    Mount Overlayfs
### END INIT INFO

case "$1" in
    start|"")
        if [ ! -d /home/data/upper ];then
            mkdir -p /home/data/upper
        fi
        if [ ! -d /home/data/work ];then
            mkdir -p /home/data/work
        fi

        #mount -n /dev/mmcblk0p6 /overlay/
        mount --bind -n / /overlay
        mount -t overlay overlay -o
lowerdir=/overlay/,upperdir=/home/data/upper/,workdir=/home/data/work/ /mnt/
        mount -n /proc --move /mnt/proc
        pivot_root /mnt /mnt/overlay
        mount -n /overlay/dev --move /dev
        mount -n /overlay/tmp --move /tmp
        mount -n /overlay/run --move /run
        mount -n /overlay/mnt/ramfs --move /mnt/ramfs
        mount -n /overlay/sys --move /sys
        mount -n /overlay/home/user --move /home/user
        mount -n /overlay/mnt/sconf --move /mnt/sconf
        mount -n /overlay/home/data --move /home/data
        mount -n /overlay/overlay --move /overlay
        #mount -t overlay overlay -o
lowerdir=/,upperdir=/userdata/upper,workdir=/userdata/work /
        ;;
    restart|reload|force-reload)
        # No-op
        ;;
    stop|status)
        # No-op
        ;;
    *)
        echo "Usage: start" >&2
        exit 3
        ;;
esac

```

脚本解释如下：

1、mount --bind -n /overlay

这个命令使用 `mount` 命令中的 `--bind` 选项来创建一个镜像挂载，将根文件系统上的 `/` 目录挂载到 `/overlay` 目录。这种操作通常用于在同一文件系统中创建文件或目录的镜像，以便在不影响原始文件系统的情况下进行修改。

- `mount`: 这是 Linux 中用于挂载文件系统的命令。
- `--bind`: 这是一个选项，表示创建一个镜像挂载。这将使目标目录中的内容镜像为源目录的内容，无论您对哪个目录进行修改，都会反映在另一个目录中。
- `-n`: 这是一个选项，表示以 "noexec" 模式挂载文件系统。这意味着在挂载的文件系统上不会执行任何可执行文件。
- `/`: 这是源目录，即根文件系统的根目录。
- `/overlay`: 这是目标目录，表示将源目录挂载到的位置。

通过执行这个命令，您将会在 `/overlay` 目录中创建一个文件系统的镜像，该镜像包含根文件系统的内容。这使得您可以在 `/overlay` 目录中进行修改，而不会影响原始根文件系统

2、mount -t overlay overlay -o lowerdir=/overlay/,upperdir=/home/data/upper/,workdir=/home/data/work/ /mnt/

这个命令使用 Overlay 文件系统来挂载一个 Overlay 层到 `/mnt/` 目录。它将三个目录 (`/overlay/`、`/home/data/upper/` 和 `/home/data/work/`) 合并为一个 Overlay 文件系统。

- `mount`: 这是 Linux 中用于挂载文件系统的命令。
- `-t overlay`: 这是指定挂载类型为 Overlay 文件系统。
- `overlay`: 这是目标挂载点，表示要将 Overlay 文件系统挂载到 `/mnt/` 目录。

- `-o`: 这是一个选项，用于指定挂载选项。
 - `lowerdir=/overlay/`: 这是底层只读层的目录，即基本的根文件系统。
 - `upperdir=/home/data/upper/`: 这是上层读写层的目录，其中的更改将覆盖底层。
 - `workdir=/home/data/work/`: 这是工作目录，Overlay 文件系统需要一个临时工作目录来处理一些操作。

通过执行这个命令，您将创建一个 Overlay 文件系统，其中 `/overlay/` 目录的内容将成为底层只读层，`/home/data/upper/` 目录的内容将成为上层读写层。任何对 `/mnt/` 目录的访问都将在这两个层上进行合并，并且上层读写层的更改将覆盖底层。

Overlay 文件系统通常用于在嵌入式系统中进行系统定制、应用数据存储等。这个命令的效果是在 `/mnt/` 目录中展现了底层和上层目录的内容的合并视图。要确保底层和上层目录已经存在并准备好，并且要理解 Overlay 文件系统的概念和运作机制。

3、mount -n /proc --move /mnt/proc

这个命令是将已经挂载的 `/proc` 文件系统移动到 `/mnt/proc` 目录。这种操作可以在初始化或启动过程中用于重新组织文件系统的布局。

- `mount`: 这是 Linux 中用于挂载文件系统的命令。
- `-n`: 这是一个选项，表示以 "noexec" 模式挂载文件系统。这意味着在挂载的文件系统上不会执行任何可执行文件。
- `/proc`: 这是要移动的原始挂载点，表示 `/proc` 文件系统。
- `--move`: 这是一个选项，表示执行一个移动操作，将已挂载的文件系统从一个位置移动到另一个位置。
- `/mnt/proc`: 这是目标挂载点，表示 `/proc` 文件系统将会被移动到 `/mnt/proc` 目录。

通过执行这个命令，您将 `/proc` 文件系统的内容从默认的挂载点（通常是根文件系统的一部分）移动到了 `/mnt/proc` 目录。这在某些情况下可能有用，例如在某些系统初始化阶段，您可能想要重新组织文件系统的结构以便更好地管理和控制不同的文件系统层。

4、pivot_root /mnt /mnt/overlay

这个命令用于在 Linux 中执行 `pivot_root` 操作，将当前根文件系统更改为 `/mnt` 目录，并将 `/mnt/overlay` 目录设置为新的根文件系统。

- `pivot_root`: 这是 Linux 中的一个命令，用于执行根文件系统的切换。这个操作通常用于修改根文件系统的内容，以及在启动过程中更改文件系统的结构。
- `/mnt`: 这是新的根文件系统目录。在执行 `pivot_root` 后，这将成为新的根文件系统。
- `/mnt/overlay`: 这是要作为新根文件系统的挂载点的目录。在执行 `pivot_root` 后，该目录中的内容将变成新的根文件系统的一部分。

通过执行这个命令，您将会进行根文件系统的切换。具体操作的影响和后果会因为文件系统结构和用例的不同而有所不同。这种操作需要非常小心，确保您对文件系统的结构和操作有充分的理解。在实际使用中，需要进行充分的测试，并确保在操作之前备份重要的数据。

三、挂载示例

在RK3568平台buildroot只写文件系统挂载overlay文件系统如下所示：

```
[root@RK356X:~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root        384M  384M    0 100% /overlay
devtmpfs         976M    0  976M   0% /dev
tmpfs            985M    0  985M   0% /dev/shm
tmpfs            985M  308K  984M   1% /tmp
tmpfs            985M  272K  984M   1% /run
tmpfs            985M    0  985M   0% /mnt/ramfs
/dev/mmcblk0p7   119M   4.4M  106M   4% /home/user
/dev/mmcblk0p8    2.8M   25K   2.5M   1% /mnt/sconf
/dev/mmcblk0p9    6.1G   2.5M   5.8G   1% /home/data
overlay          6.1G   2.5M   5.8G   1% /
[root@RK356X:~]#
```

CSDN @flypig吙啦啦

```

[ root@RK356X:~ ]# mount
/dev/mmcblk0p6 on /overlay type squashfs (ro,relatime)
devtmpfs on /dev type devtmpfs (rw,relatime,size=998664k,nr_inodes=249666,mode=755)
proc on /proc type proc (rw,relatime)
devpts on /dev/pts type devpts (rw,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /dev/shm type tmpfs (rw,relatime,size=1007688k,nr_inodes=251922,mode=777)
tmpfs on /tmp type tmpfs (rw,relatime,size=1007688k,nr_inodes=251922)
tmpfs on /run type tmpfs (rw,nosuid,nodev,relatime,size=1007688k,nr_inodes=251922,mode=755)
tmpfs on /mnt/ramfs type tmpfs (rw,nosuid,nodev,relatime,size=1007688k,nr_inodes=251922,mode=755)
sysfs on /sys type sysfs (rw,relatime)
debug on /sys/kernel/debug type debugfs (rw,relatime)
pstore on /sys/fs/pstore type pstore (rw,relatime)
/dev/mmcblk0p7 on /home/user type ext4 (rw,relatime)
/dev/mmcblk0p8 on /mnt/sconf type ext4 (rw,relatime)
/dev/mmcblk0p9 on /home/data type ext4 (rw,relatime)
/dev/mmcblk0p6 on /overlay type squashfs (ro,relatime)
overlay on / type overlay (rw,relatime,lowerdir=/overlay/,upperdir=/home/data/upper/,workdir=/home/data/work/)
none on /sys/kernel/config type configfs (rw,relatime)
adb on /dev/usb-lun0 type functionfs (rw,relatime)
[ root@RK356X:~ ]#

```

CSDN @flypig哇啦啦

一键还原系统：只需要删除/home/data/upper文件下所有内容即可还原初始系统。如 `rm -rf /home/data/upper/*`：

```

[ root@RK356X:/home/data/upper ]# ls -l
total 16
drwxr-xr-x 5 root root 4096 Jul  6 16:18 etc
drwx----- 2 root root 4096 Jul  6 16:39 root
drwxr-xr-x 4 root root 4096 Jul  6 16:18 userdata
drwxr-xr-x 4 root root 4096 Jul  5 06:43 usr
[ root@RK356X:/home/data/upper ]#
[ root@RK356X:/home/data/upper ]# rm -rf /home/data/upper/*

```

CSDN @flypig哇啦啦

总结

通过在 RK3568 上使用 Buildroot 构建定制的 [Linux 发行版](#)^Q，并挂载 Overlay 联合文件系统，您可以轻松地进行系统定制和配置，以满足您的应用需求。这种方法可以帮助您创建一个精简且高度定制的嵌入式 Linux 环境，使您的开发过程更加灵活和高效。