Btrfs and Rollback

How It Works and How to Avoid Pitfalls

Thorsten Kukuk
Senior Architect SUSE Linux Enterprise Server
kukuk@suse.com

rm -rf / ?

Agenda:

- Btrfs / Copy-on-Write / Subvolumes
- Rollback on SUSE Linux Enterprise Server 12
- Grub2 and rollback
- Caveats and risks
- Managing subvolumes

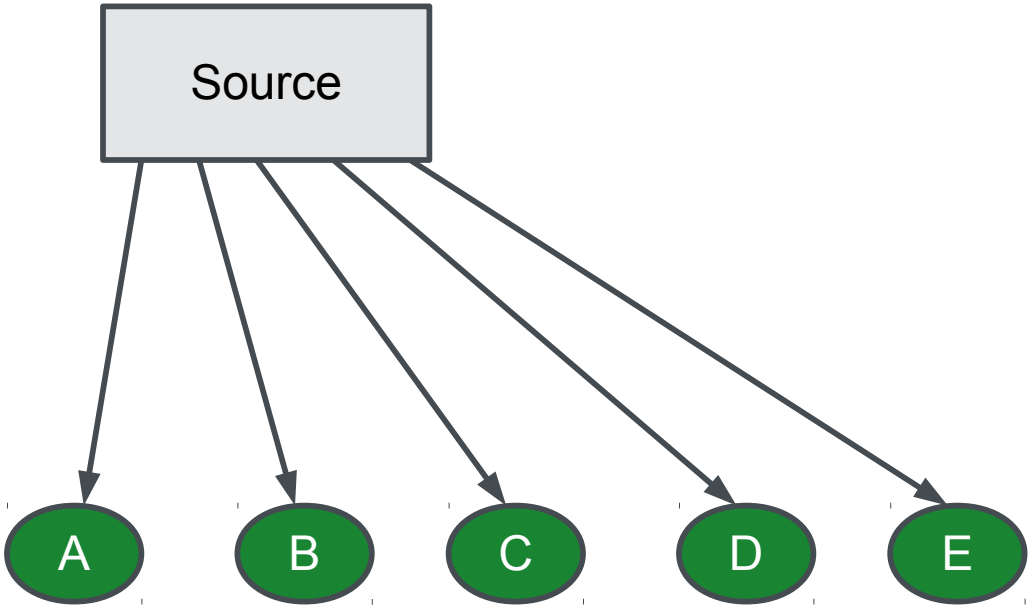
Btrfs / CoW

- Copy on Write (CoW) general purpose file system
- Trees for
  - Data
  - Metadata
- Snapshots
  - Every snapshot is again a subvolume
  - Can be mounted and accessed like every other subvolume
  - Snapshots can be created read-only

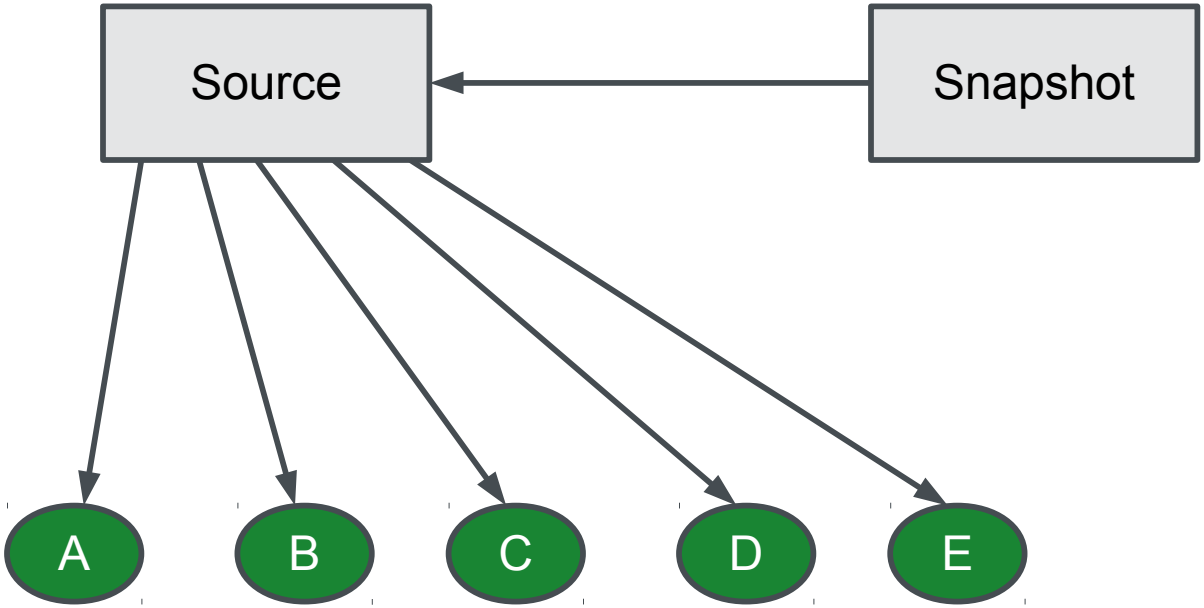
Btrfs / Subvolumes

- Not like a LVM logical volume
- Are hierarchical
- Can be accessed in two ways:
  - From the parent subvolume (like a directory)
  - Separate mounted filesystem (using subvol/subvolid)
- Every btrfs filesystem has a default, top-level subvolume with id 5
- Snapshots are subvolumes, which shares its data with other subvolumes (snapshots)
- Only subvolumes can be the source for snapshots

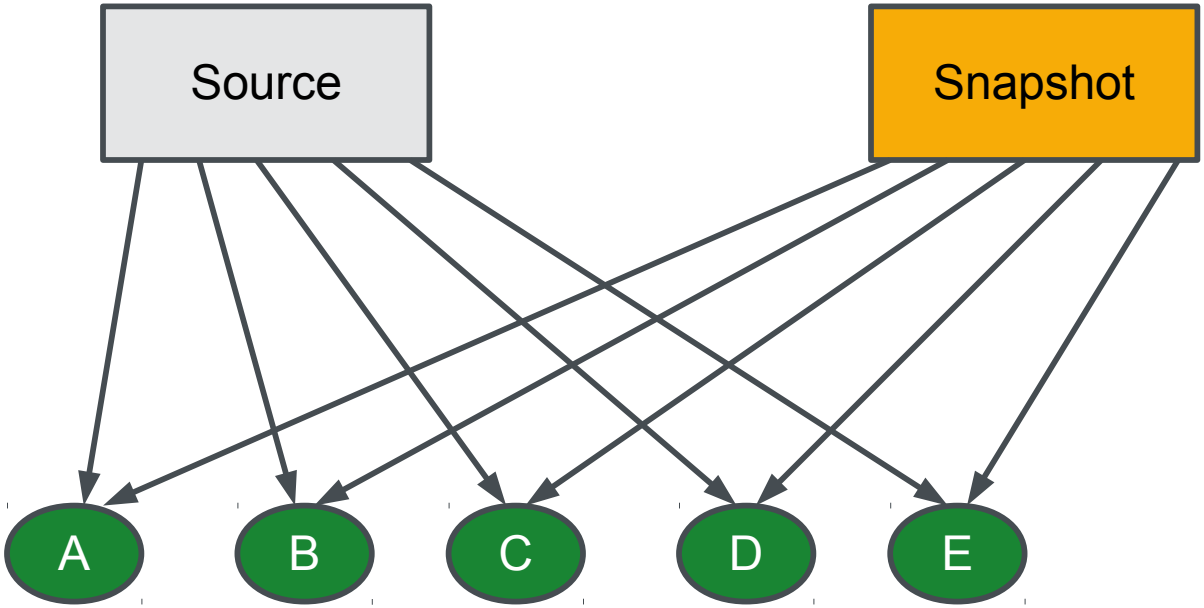
Btrfs / Copy-on-Write (1/4)



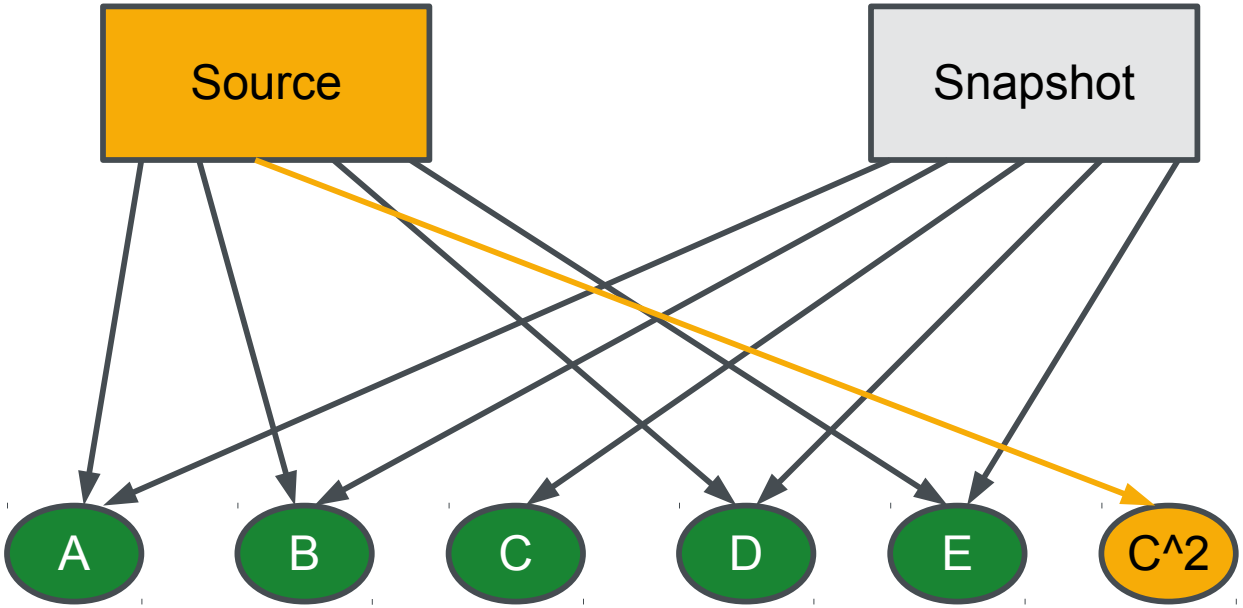
Btrfs / Copy-on-Write (2/4)



Btrfs / Copy-on-Write (3/4)



Btrfs / Copy-on-Write (4/4)



Btrfs Snapshots and Disk Usage

- How much disk space does a snapshot need?

The answer nobody likes: It depends!

- Initial snapshot: few Bytes for Metadata
- Growing over time when original data changes
- At the end: same amount as original data
- Worst case: Lot of snapshots and no common blocks between them.

Btrfs
on SUSE Linux Enterprise Server 12

- Default root filesystem for root partition (incl. /boot)
- Advantage: rollback including kernel
- Bootloader can boot from btrfs
- Rollback per subvolume
- Rollback integrated into bootloader
- No consistent snapshots cross partition boundaries
- No automatic snapshots per cron for root filesystem

Full System Rollback

Rollback per Subvolume (1/2)

How it works

- Instead of the original subvolume, the snapshot is mounted with the options “subvol=<name>”
  - Remember: snapshots are subvolumes

- “*btrfs subvolume set-default ...*” for permanent assignments

→ Implemented in Snapper as “rollback”

Rollback per Subvolume (2/2)

- Benefits
  - “atomic” operation
  - Very fast
- Disadvantages
  - Additional complexity
    - Requires explicit mounting of subvolumes
    - Subvolumes can prevent snapshots from being deleted
  - “Disk space leaks”
    - Initial installation needs to be done into an extra subvolume

Reboot *Later* Mode

- Administrator is in a current read-write filesystem, but wants to rollback
- “snapper list” to view and select a snapshot
- Call “snapper rollback <number>”, which will:
  - Create a new read-only snapshot of the currently running system
  - Create a new read-write snapshot of the snapshot <number>, linearly after the just recently created read-only snapshot
  - “setdefault” to the new read-write snapshot
- Then reboot

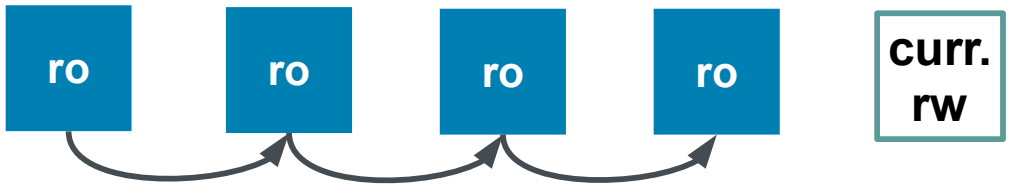
Reboot *Now* Mode

- Boot into an existing read-only snapshot
- Text console and some services should work
  - Because most data is in writeable subvolumes
- To continue to work in this snapshot, the admin should call "*snapper rollback*". This will:
  - Create a new read-only snapshot of the old read-write one
  - Create a new read-write snapshot of the current read-only one
  - All linearly after the last existing snapshot
  - "setdefault" to the new read-write snapshot
- Then reboot

User View on Snapshot History

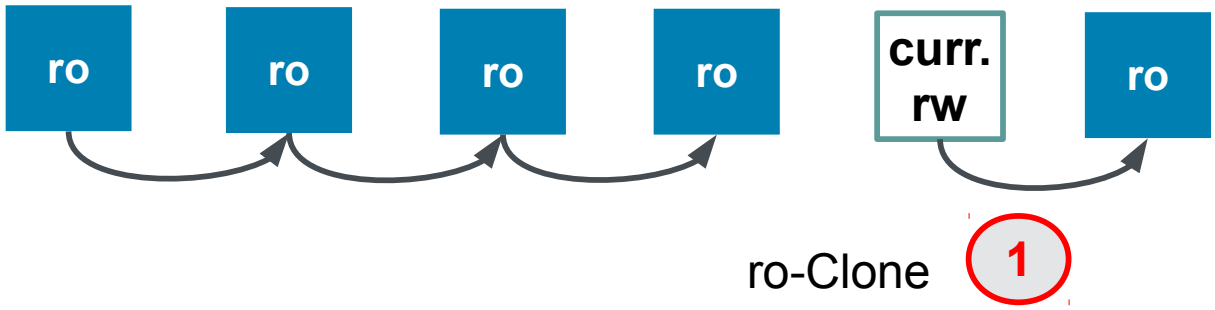
Snapshot / Rollback

User View on Snapshot History (1)



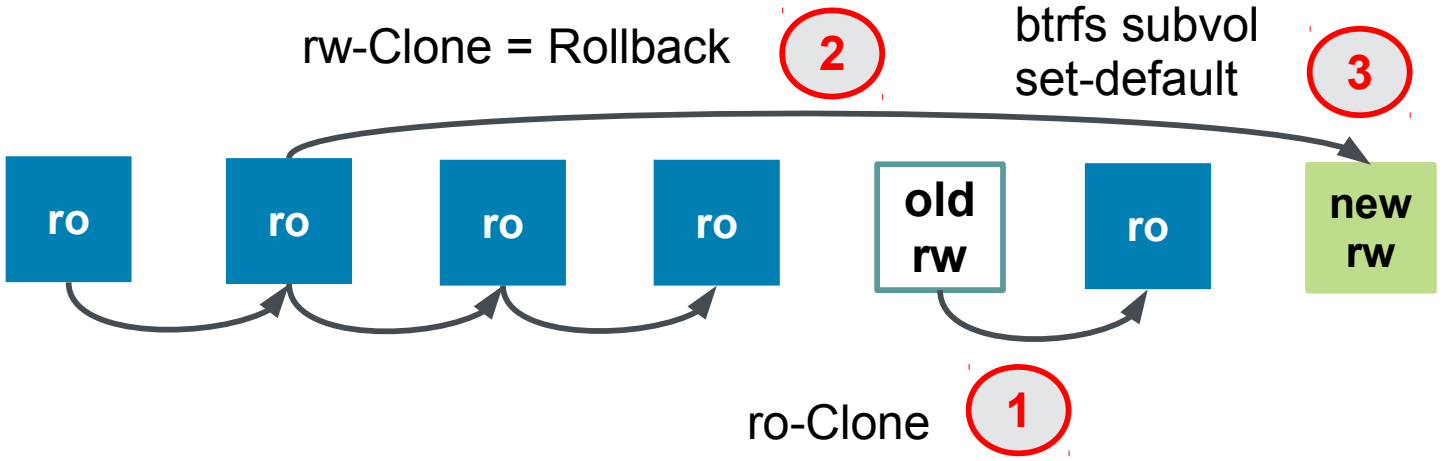
Snapshot / Rollback

User View on Snapshot History (2)



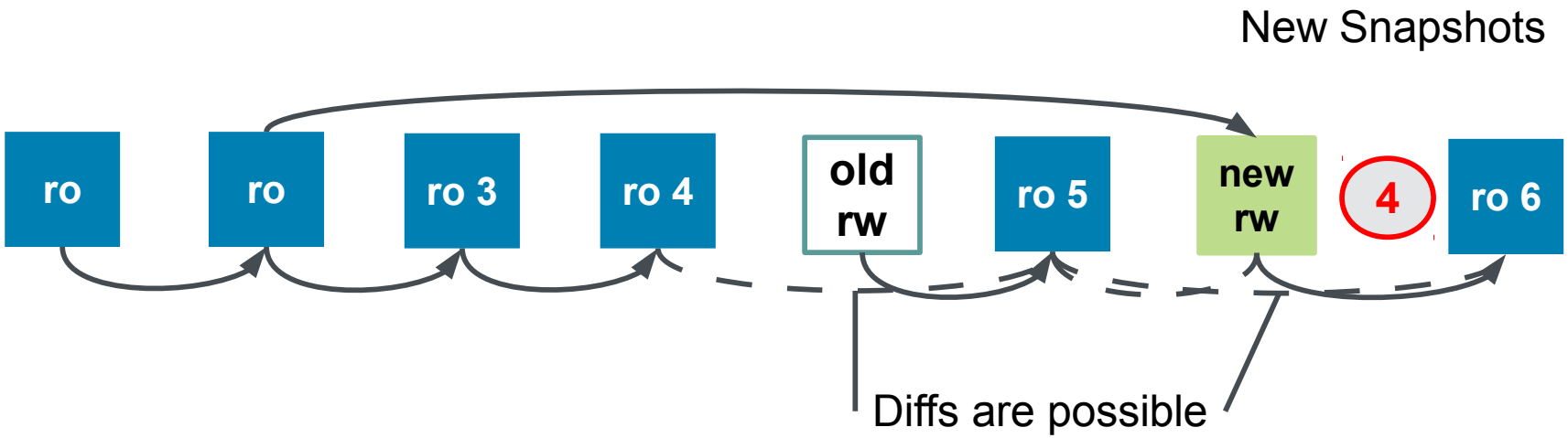
Snapshot / Rollback

User View on Snapshot History (3)



Snapshot / Rollback

User View on Snapshot History (4)



User View on Snapshot History (5)

Condensed view

What happens,
if we rollback again?

Caveat: this does not reflect 1:1 what happens technically.

User View On grub2 Interface

Snapper Headers

- **Type**: [Pre | Post | Single]
- **#**: Nr of snapshot
- **Pre #**: if type is “Post” the matching Pre nr.
- **Date**: timestamp
- **Cleanup**: cleanup algorithm for this snapshot
- **Description**: A fitting description of the snapshot (free text)
- **Userdata**: key=value pairs to record all sorts of useful information about the snapshot in an easily parsable format

Important Snapshots

Snapshots are marked as important (“*”), if a package affecting the boot process is updated:

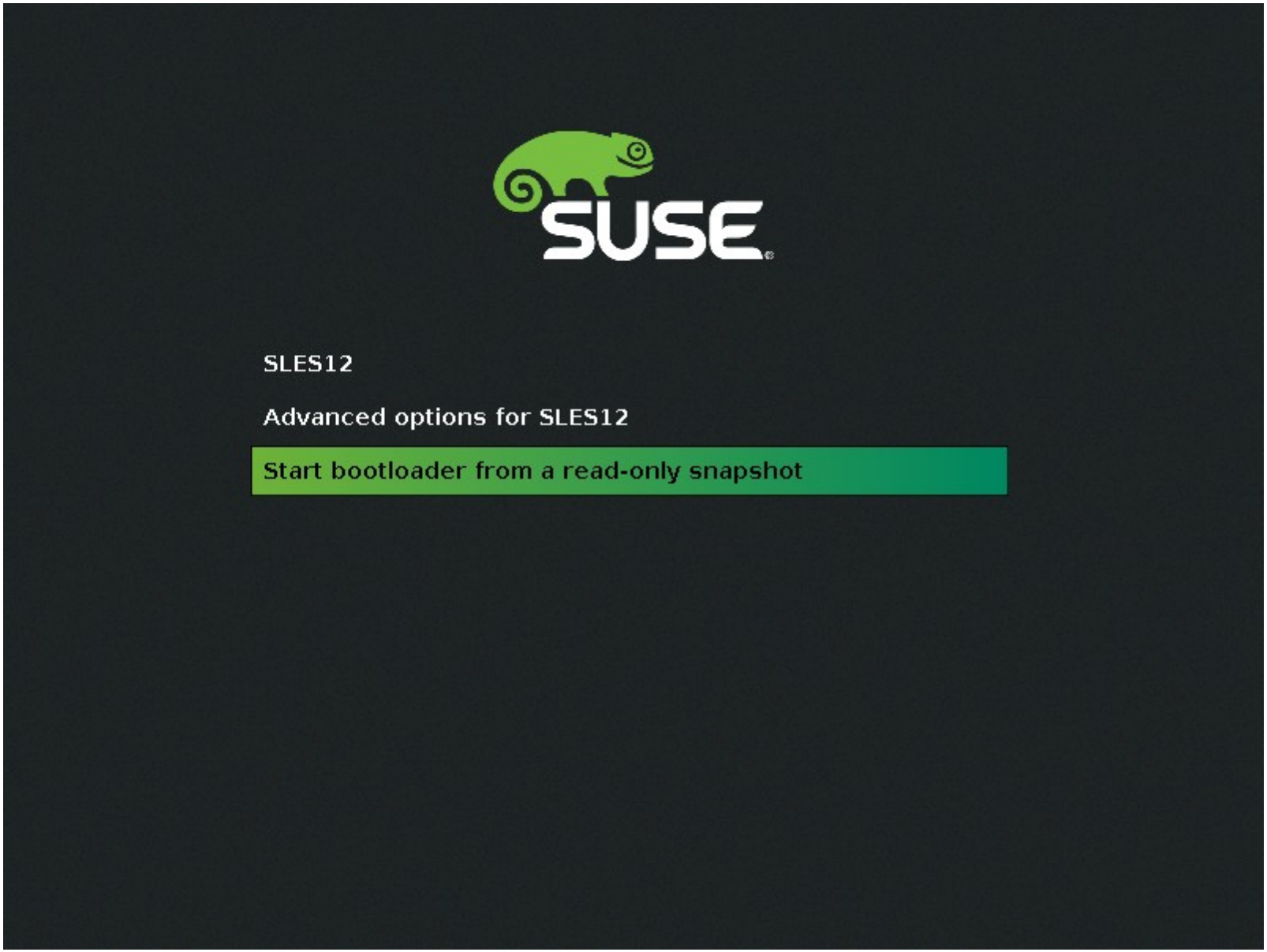
- kernel
- dracut
- glibc
- systemd
- udev

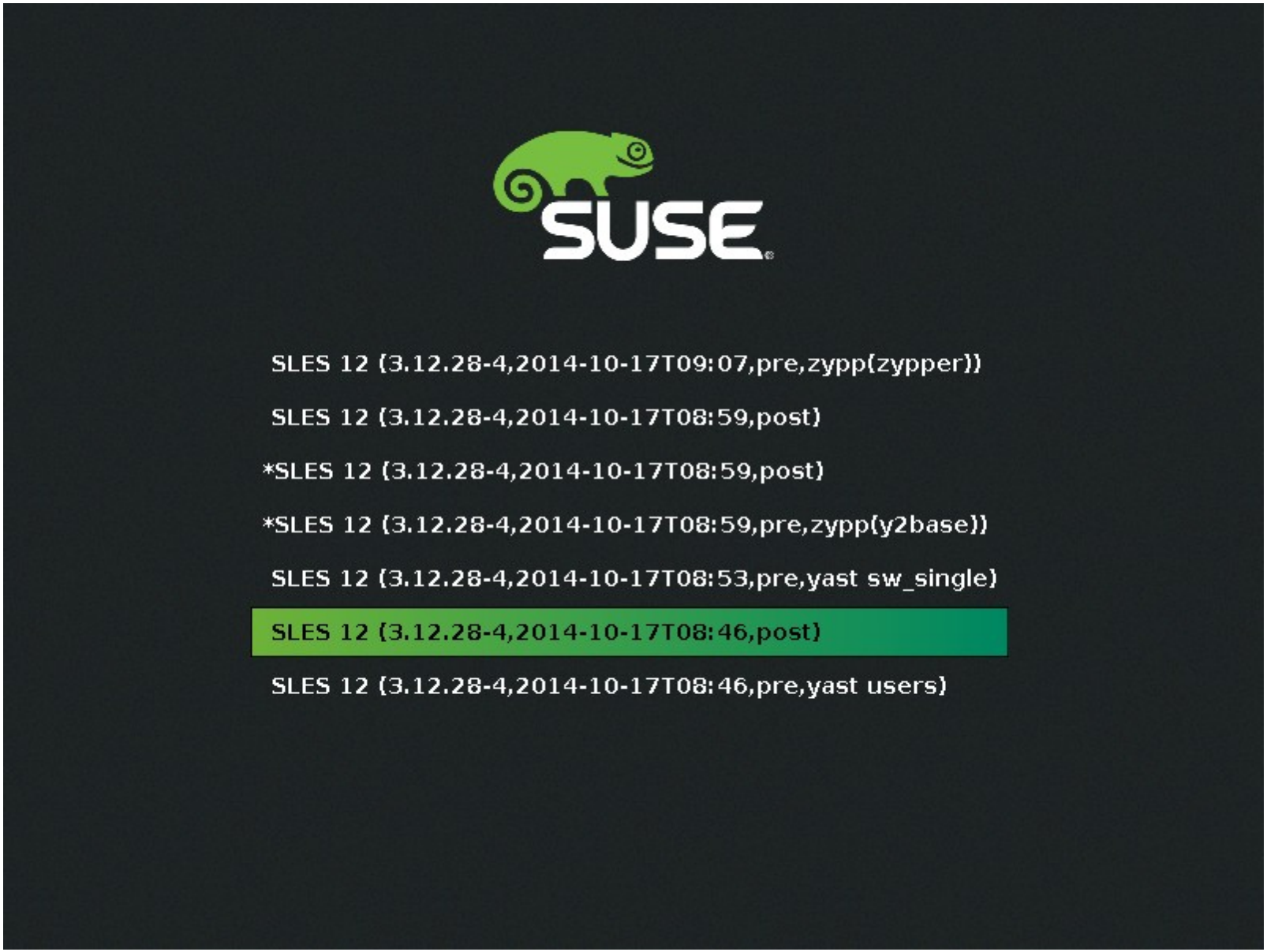
You can configure that in /etc/snapper/zypp-plugin.conf

Modify grub2 menu

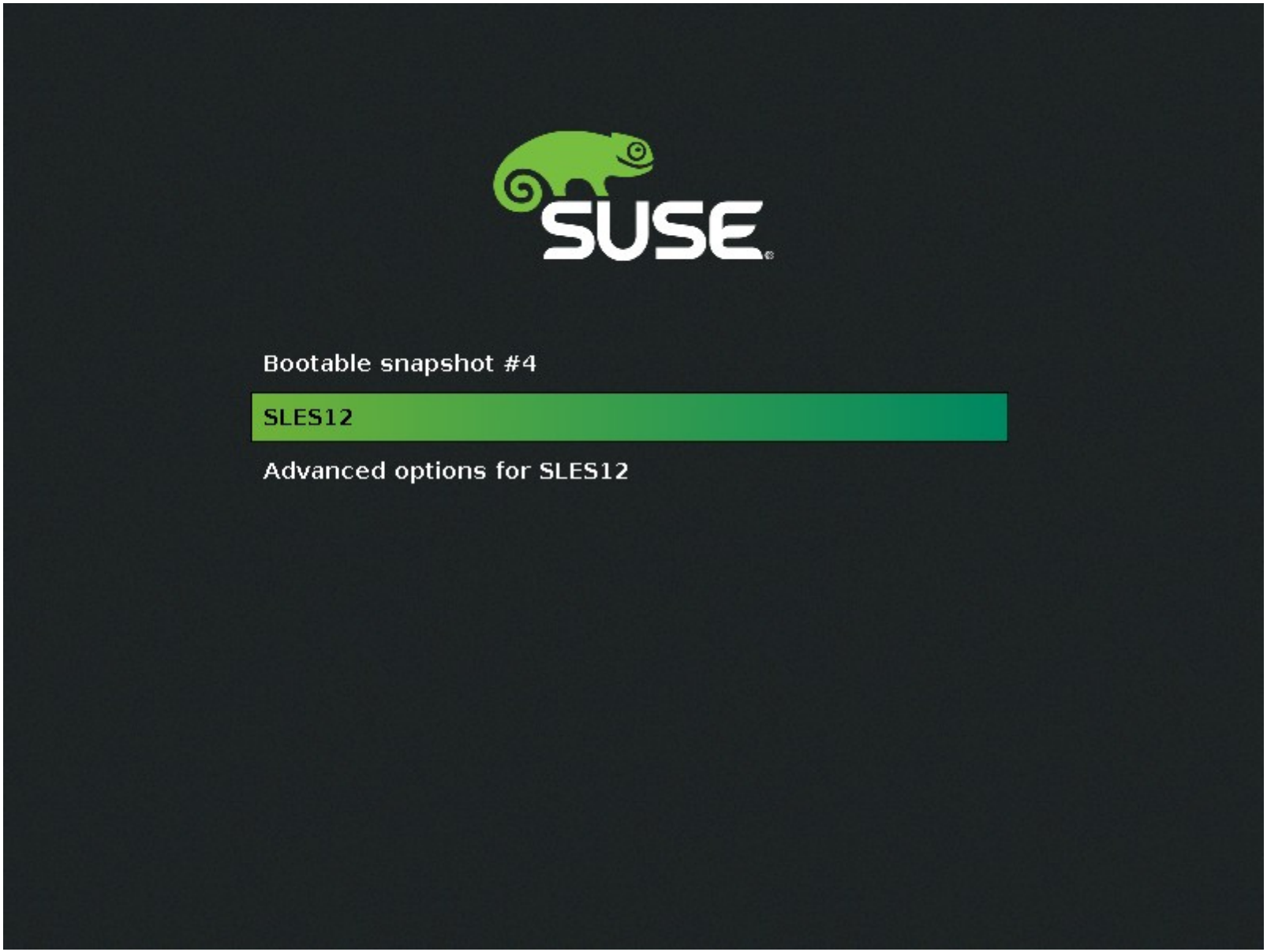
The text in the grub2 menu can be set by the admin:

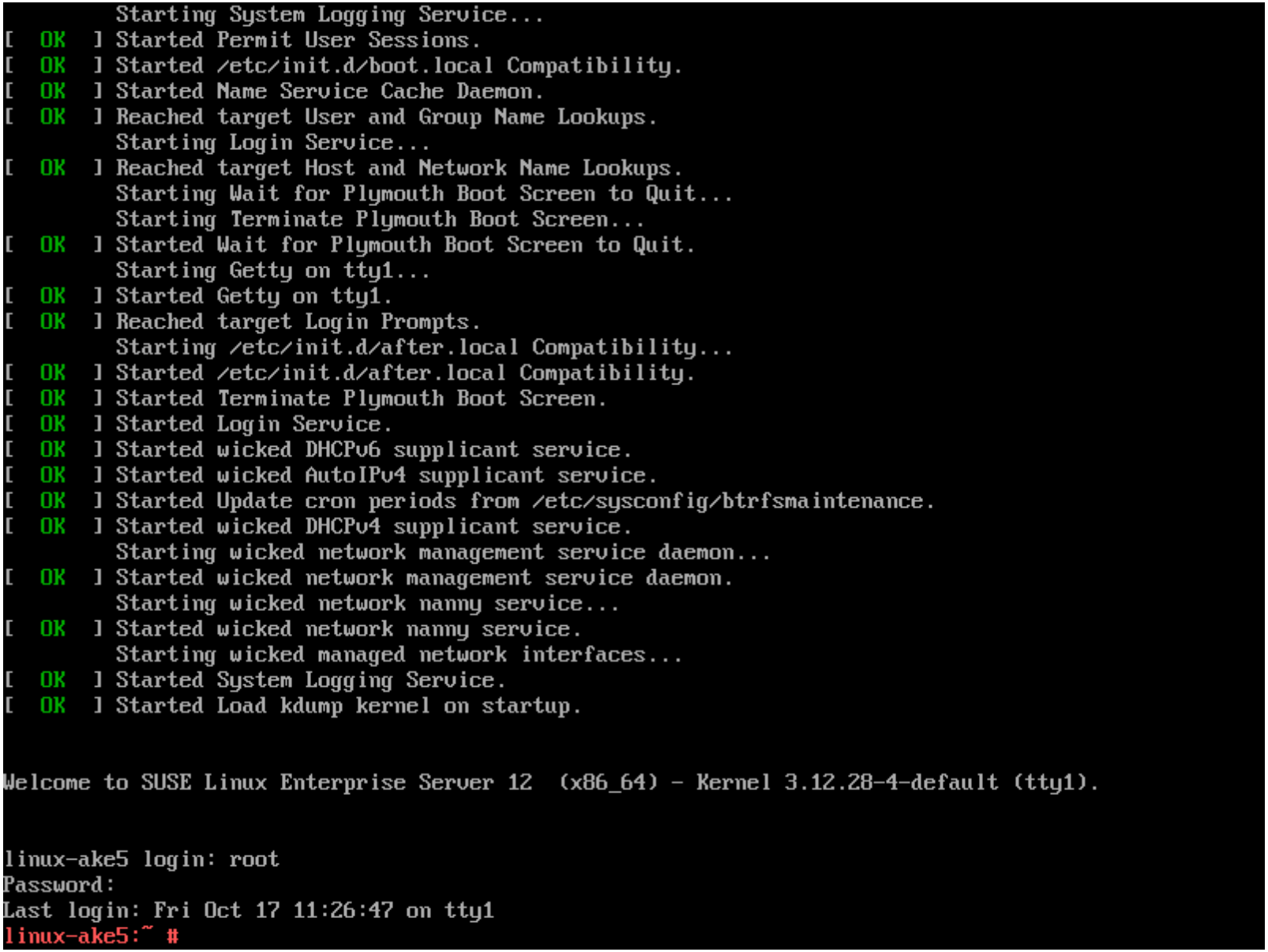
- snapper modify --userdata="bootloader=foo bar" [number]
  - [number] = number of the snapshot


SUSE
SLES12
Advanced options for SLES12
Start bootloader from a read-only snapshot


SLES 12 (3.12.28-4,2014-10-17T09:07,pre,zypp(zypper))
SLES 12 (3.12.28-4,2014-10-17T08:59,post)
*SLES 12 (3.12.28-4,2014-10-17T08:59,post)
*SLES 12 (3.12.28-4,2014-10-17T08:59,pre,zypp(y2base))
SLES 12 (3.12.28-4,2014-10-17T08:53,pre,yast sw_single)
SLES 12 (3.12.28-4,2014-10-17T08:46,post)
SLES 12 (3.12.28-4,2014-10-17T08:46,pre,yast users)

```
g227:~ # snapper list
Type   | #  | Pre # | Date                     | User | Cleanup | Description    | Userdata
-------+----+-------+--------------------------+------+---------+----------------+--------------
single | 0  |       |                          | root |         | current        |
pre    | 3  |       | Fri Oct 17 10:46:22 2014 | root | number  | yast users     |
post   | 4  | 3     | Fri Oct 17 10:46:49 2014 | root | number  |                |
pre    | 5  |       | Fri Oct 17 10:53:14 2014 | root | number  | yast sw_single |
pre    | 6  |       | Fri Oct 17 10:59:12 2014 | root | number  | zypp(y2base)   | important=yes
post   | 7  | 6     | Fri Oct 17 10:59:16 2014 | root | number  |                | important=yes
post   | 8  | 5     | Fri Oct 17 10:59:19 2014 | root | number  |                |
pre    | 9  |       | Fri Oct 17 11:07:17 2014 | root | number  | zypp(zypper)   | important=no
post   | 10 | 9     | Fri Oct 17 11:07:18 2014 | root | number  |                | important=no
g227:~ # _
```


Bootable snapshot #4
SLES12
Advanced options for SLES12


Starting System Logging Service...
[OK] Started Permit User Sessions.
[OK] Started /etc/init.d/boot.local Compatibility.
[OK] Started Name Service Cache Daemon.
[OK] Reached target User and Group Name Lookups.
Starting Login Service...
[OK] Reached target Host and Network Name Lookups.
Starting Wait for Plymouth Boot Screen to Quit...
Starting Terminate Plymouth Boot Screen...
[OK] Started Wait for Plymouth Boot Screen to Quit.
Starting Getty on tty1...
[OK] Started Getty on tty1.
[OK] Reached target Login Prompts.
Starting /etc/init.d/after.local Compatibility...
[OK] Started /etc/init.d/after.local Compatibility.
[OK] Started Terminate Plymouth Boot Screen.
[OK] Started Login Service.
[OK] Started wicked DHCPv6 supplicant service.
[OK] Started wicked AutoIPv4 supplicant service.
[OK] Started Update cron periods from /etc/sysconfig/btrfsmaintenance.
[OK] Started wicked DHCPv4 supplicant service.
Starting wicked network management service daemon...
[OK] Started wicked network management service daemon.
Starting wicked network nanny service...
[OK] Started wicked network nanny service.
Starting wicked managed network interfaces...
[OK] Started System Logging Service.
[OK] Started Load kdump kernel on startup.
Welcome to SUSE Linux Enterprise Server 12 (x86_64) - Kernel 3.12.28-4-default (tty1).
linux-ake5 login: root
Password:
Last login: Fri Oct 17 11:26:47 on tty1
linux-ake5:~ #

Caveats and Risks

Snapshotting “/” – Challenges

- Consistent system / “atomic”
  - We cannot do that cross partition boundaries
- Kernel and initrd / initramfs = “/boot”
  - /boot not as extra partition
- Different stages of bootloader needs to match
  - Exclude /boot/grub2/<grub2 arch> from snapshot
  - Grub2 configuration is part of the snapshot

  →new grub2 needs to be able to read old configs

Data and Rollback

I made a rollback, but …

… what happens with my new data???

System Integrity and Compliance

Don't allow to roll back certain log-files, databases etc.

Solution: subvolumes instead of directories for

- /boot/grub2/*
- /opt
- /srv
- /tmp
- /usr/local
- /var/crash
- /var/lib/{mailman,named,pgsql} (No mysql!)
- /var/log
- /var/opt
- /var/spool
- /var/tmp

What Can Be Broken After a Rollback?

- /home/<user> exists
  - but no entry in /etc/passwd
- /opt can contain Add-Ons
  - but dependencies are no longer fulfilled
- /srv can contain web applications
  - but wrong php/ruby on rails version installed
- Database was not in a subvolume or extra partition
  - all data after creating the snapshot for rollback is lost

→ Copy modified data from snapshot of old root subvolume into new root subvolume

Cleanup of snapshots

Automatic Cleanup of Snapshots

- Old snapshots are automatically deleted
  - Depending on “Cleanup” field (snapper list)
  - Cron job – once a day
- Snapshots containing subvolumes cannot be deleted
  - Have one subvolume and create all other subvolumes in it
- Root snapshots/subvolumes are excluded
  - Even old ones after rollback
  - Prevent deletion by accident

Automatic Cleanup of Snapshots

- Timeline Snapshots
  - Disabled by default for root partition
  - First snapshot of the last 10 days/months/years are kept
- Installation Snapshots/Administration Snapshots
  - New snapshot when calling YaST or zypper
  - Last 10 important snapshots are kept
  - Last 10 “regular” snapshots are kept

Managing Subvolumes

Handling of Subvolumes

Subvolumes are automatically mounted with their parent volume

- Past: Only the root subvolume in /etc/fstab
- SLES 12: All subvolumes are listed in /etc/fstab

Why?

- After rollback, old subvolumes are not part of the new parent subvolume!

How to Create a Subvolume

- Move old directory (/path/name) away
- Mount “original” root and create subvolume:
  - *Mount /dev/sda2 -o subvol=@ /mnt*
  - *btrfs subvolume create /mnt/path/name*
  - *umount /mnt*
- Add new subvolume to /etc/fstab and mount it:
  - *echo “/dev/sda2 /path/name btrfs subvol=@/path/name 0 0” >> /etc/fstab*
  - *mkdir /path/name*
  - *mount /path/name*
- Move old data back into new subvolume

How to Delete a Subvolume

- Create temporary directory /path/name.tmp
- Copy data into this directory
  - cp -a –reflink <src dir> <dst subvolume>
- Delete subvolume:
  - *btrfs subvolume delete /path/name*
- Remove /path/name from /etc/fstab
- Move temporary directory to original name:
  - *mv /path/name.tmp /path/name*

Missing Answer

rm -rf /

Is it now safe?

Missing Answer

rm -rf /

Is it now safe?

No!

Missing Answer

rm -rf /

Is it now safe?

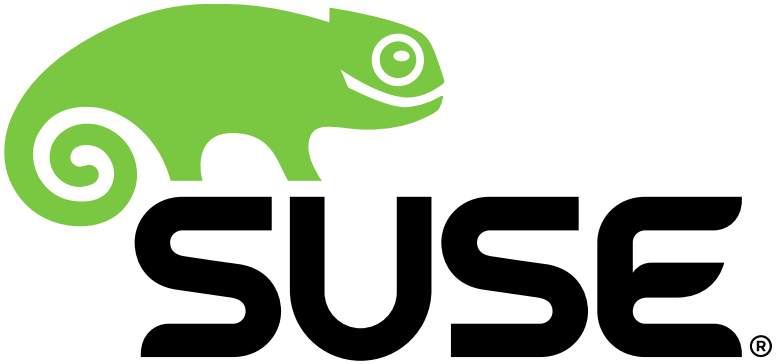
No!

Why not?

It will not stop on subvolumes.

Questions?

Thank you.


SUSE®
We adapt. You succeed.