btrfs and snapper

The Next Steps From Pure filesystem Features to Management Integration and Compliance

SUSECon 2013, Orlando, Florida

Gábor Nyers
Sr. System Engineer @SUSE
gnyers@suse.com

Arvin Schnell
Sr. Software Developer
aschnell@suse.com

Abstract

"btrfs" as a filesystem has been getting a lot of attention over the past few years. While it is interesting from its feature set alone (checksums, copy on write, snapshots, Volume Management integration), some of these features are not directly useful for customers without a proper integration management infrastructure and compliance processes.

This session will help you understand the features of btrfs and how snapper can be used for snapshot management in SUSE Linux Enterprise. We also will provide an outlook for future functionality.

Agenda

- Introduction to Btrfs
- Btrfs in SUSE distro's Snapper
- Btrfs Use cases
- Summary and Questions

- Btrfs specs
- Features and Concepts
- Current limitations
- Support from distributions

What People Say About Btrfs...

Chris Mason (lead developer Btrfs)

- General purpose filesystem that scales to very large storage
- Focused on features that no other Linux filesystems have
- Easy administration and fault tolerant operation

Ted Tso (lead developer Ext4)

- (Btrfs is) “... the way forward”

Others:

- “Next generation Linux filesystem”
- “Btrfs is the Linux answer to ZFS”

Why Another Linux filesystem?

- Solve Storage Challenges
  - Scalability
  - Data Integrity
  - Dynamic Resources (expand and shrink)
  - Storage Management
  - Server, Cloud – Desktop, Mobile
- Compete with and exceed the filesystem capabilities of other Operating Systems

Btrfs Specs

- Max volume size : 16 EB (2^64 byte)
- Max file size : 16 EB
- Max file name size : 255 bytes
- Characters in file name : any, except 0x00
- Directory lookup algorithm : B-Tree
- Filesystem check : on- and off-line
- Compatibility
  - POSIX file owner/permission
    Access Control Lists (ACLs)
    Asynchronous and Direct I/O
    Hard- and symbolic links,
    Extended Attributes (xattrs),
    Sparse files

Btrfs Feature Summary 1/2

- Extents
  - Use only what's needed
  - Contiguous runs of disk blocks
- Copy-on-write
  - Never overwrite data!
- Snapshots
  - Light weight
  - At file system level
  - RO / RW
- Multi-device Management
  - mixed size and speed
  - on-line add and remove devs
- Object level RAID:
  - 0, 1, 10
- Efficient small file storage
- SSD support (optimizations, trim)

Btrfs Feature Summary 2/2

- Checksums on data and meta data
- On-line:
  - Balancing
  - Grow and shrink(!)
  - Scrub
  - Defragmentation
- Transparent compression (gzip, lzo)
- In-place conversion from Ext[34] to Btrfs
- Quota groups
- Send/Receive
  - Similar to ZFS' send/receive function
- Seed devices
  - Overlay a RW file system on top of an RO
- Data de-duplication:
  - Background de-dup process (see also bedup)

Btrfs Feature Support – SLES 11 SP3

Supported

- Snapshots
- Copy-on-Write
- Subvolumes
- Metadata Integrity
- Data Integrity
- Online metadata scrubbing
- Manual defrag
- Manual deduplication (soon)
- Multiple devices
- QGroups

Unsupported

- Inode cache
- Auto Defrag
- RAID
- Compression
- Send / Receive
- Hot add / remove disks
- Seeding devices
- Big Metadata

Btrfs Planned Features

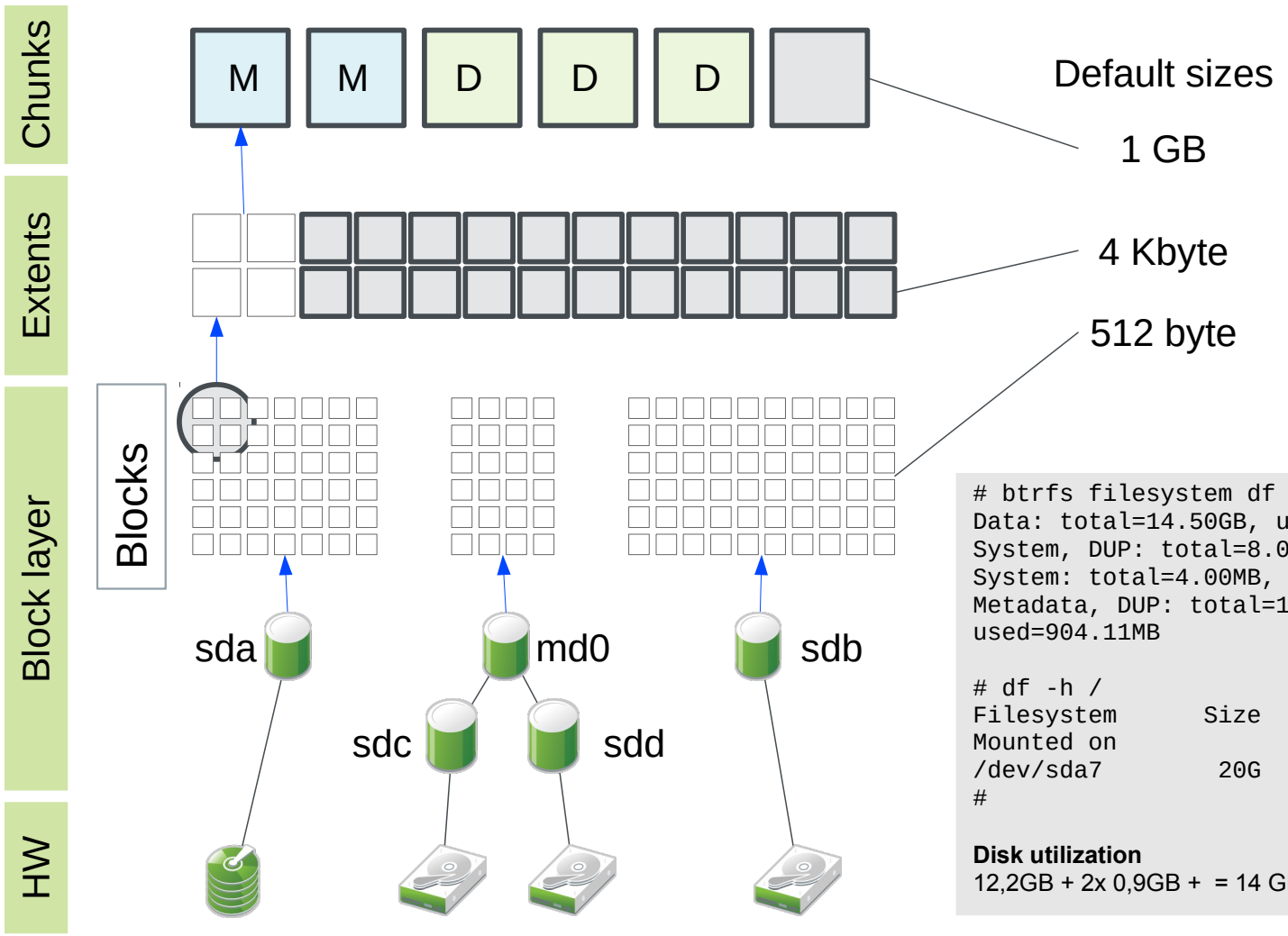
- Object-level RAID 5, 6
- Data de-duplication:
  - On-line de-dup during writes
- Tiered storage
  - Frequently used “hot” data on SDD(s)
  - “Archive” on HDD(s)

A Few Btrfs Concepts

- Extents and Chunks
- B-Tree
- Raw data
- Meta data
- Subvolume
- Snapshot

Btrfs Concepts:

Extents and Storage Organization



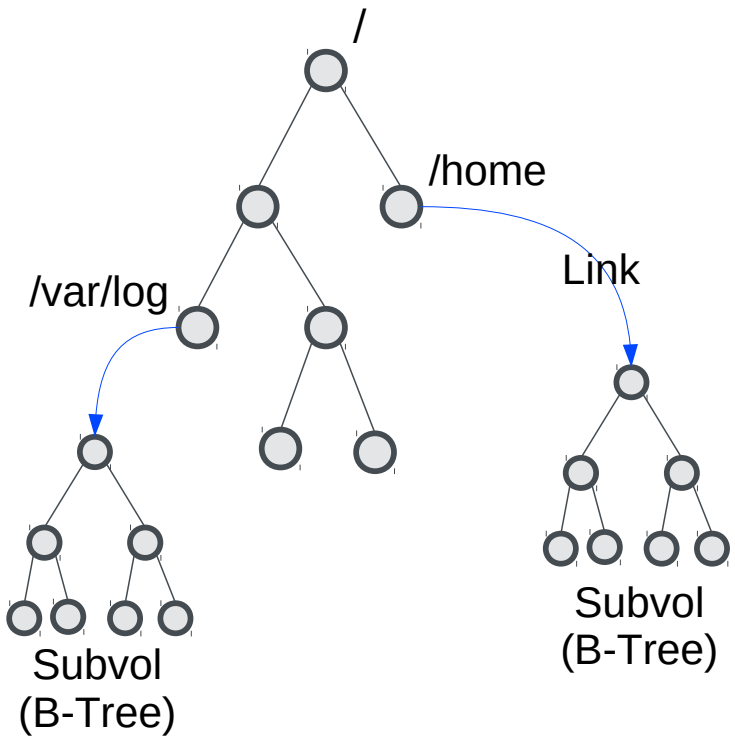

```
# btrfs filesystem df /
Data: total=14.50GB, used=12.20GB
System, DUP: total=8.00MB, used=12.00KB
System: total=4.00MB, used=0.00
Metadata, DUP: total=1.75GB,
used=904.11MB

# df -h /
Filesystem      Size  Used Avail Use%
Mounted on
/dev/sda7        20G   14G  4.3G  77% /
#
```

Disk utilization
12,2GB + 2x 0,9GB + = 14 GB

Btrfs Concepts:

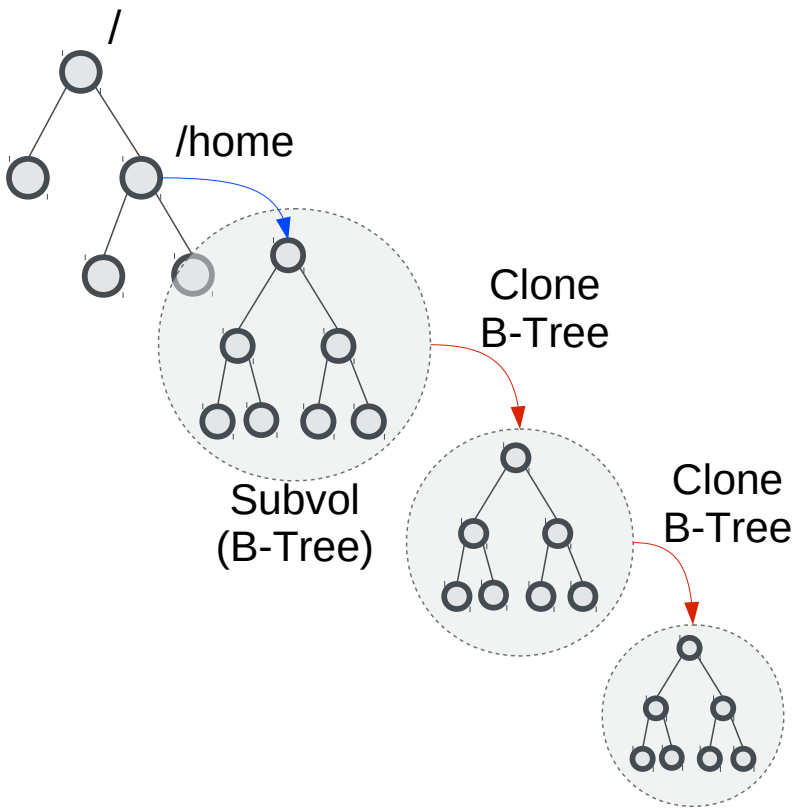
Subvolumes



- Independent B-Tree linked to some directory of the root subvolume
- A part of the file system
- Appears on file system as a directory
- Independently mountable
- Subvols on a Btrfs file system share the same device pool
- Independently snapshotable (copy B-Tree)

Btrfs Concepts:

Snapshots



- A record of the state of a subvolume
- CoW copy of another subvolume
- After creation, snapshot shares all raw data and metadata with parent
- (practically) unlimited in number
- Read Only, Writable and Nested (= “snapshot of a snapshot”)
- Snapshots on the file system level

Btrfs file system check, recovery and repair

- Status of btrfsck
  - Released in SLES 11 SP2 and OL6 with UEK2
  - Off-line filesystem repair
- Btrfs-restore
  - Read-only recovery tool
- Auto recovery at mount
  - mount -o recovery

Btrfs Features:

Current limitations (Oct 2013)

- Full featured off-line fsck repair tool, however:
  - Implementation of off-line fsck already available
  - On-line repair options with btrfs scrub
  - Recovery mount option
  - btrfs-restore utility
- Limited bootloader support (GRUB2 only)
- RAID 5 and 6 (patch)
- Quality technical documentation

Btrfs Support Status – Distros

Supported

- SUSE® Linux Enterprise Server 11 SP2+
- Oracle Linux 6 with UEK2+

Available

- OpenSUSE 11.4+
- Debian 6+
- Ubuntu 11.04+

Unsupported (Technology Preview)

- Red Hat Enterprise Linux
- Fedora
- ...

Agenda

- Introduction to Btrfs
- Btrfs in SUSE distro's Snapper
- Btrfs Use cases
- Summary and Questions

- Btrfs integration in SLES and openSUSE
- Partitioner
- Planned features
- Snapshot management with Snapper
- Filesystem recommendations

Btrfs integration in SLE 11 SP2 and openSUSE 12.1

Basic integration into

- Installer
  - Btrfs as root file system
  - Recommendation for subvolume layout
- Partitioner
  - Create Btrfs
  - Create subvolumes

Tools

- Btrfs support in AutoYaST
- Snapper
  - Manage snapshots
  - Automatically create snapshots
  - Display differences between snapshots
  - Roll-back

A Few Recommendations 1/2

Filesystem size:

- Starting: ~30% filling
- Operation: <90% filling

Subvolumes layout

- Directories containing logs to avoid rolling back logs
- Directories w/ high volume write I/O, like:

```
/tmp, /srv,
/var/spool, /var/log,
/var/run, /var/tmp,
/opt:
```

should go on different subvolume, or a non-btrfs filesystem;
Espcially on rotating disks!

A Few Recommendations 2/2
A maintenance Plan Proposal

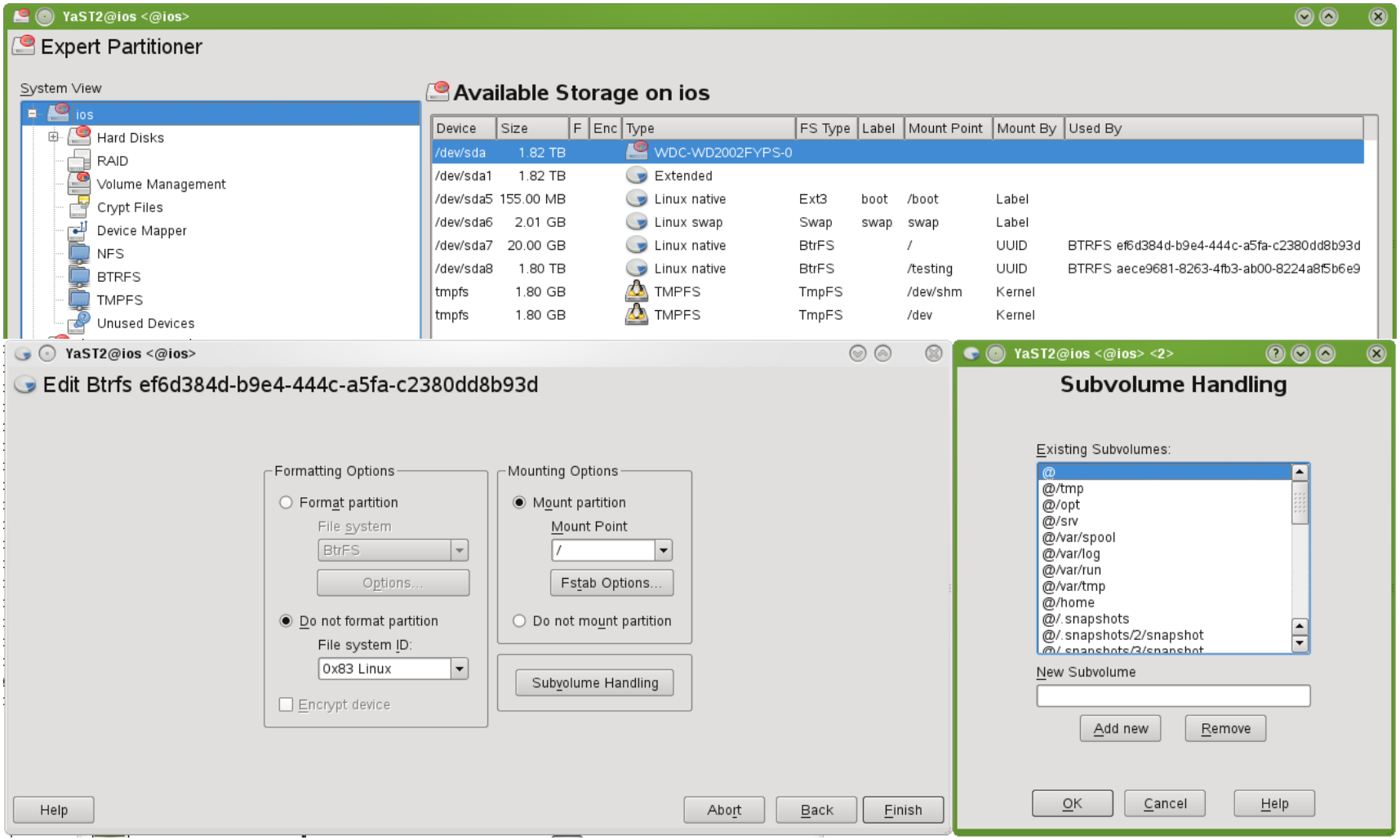
- Getting a baseline (ca. 1 week – 1 month)
  - Locate “Hot zones”
    - using `filefrag` check fragmentation of files

- Daily maintenance
  - Monitor free space
  - Monitor performance impact of “Hot zones”
  - Scheduled Scrub
- Weekly maintenance
  - Scheduled / Manual defrag (if Btrfs on HDD, instead of on SSD)
  - Check / remove unneeded snapshots
- Monthly maintenance
  - re-evaluate need for filesystem extension
  - `btrfs file balance`
  - re-evaluate subvolume layout

Demo 1

- Convert exitsting Ext3 to Btrfs
- On-line resize Btrfs
  - Grow
  - Shrink

Btrfs Integration in YaST Partitioner



Future plans

- YaST partitioner support for:
  - Built-in multi-volume handling and RAID
  - Transparent compression
- Bootloader support for /boot on btrfs (SLE12)
- Snapshot integration into bootloader

Btrfs References

Publications

- Btrfs wiki (and mirror)
- Josef Bacik's article on Btrfs
- Arne Jansen's paper on qgroups (quota support)
- Oloh Rodeh - B-trees, Shadowing, and Clones, IBM Research paper
- LWN - “A short history of btrfs”article
- Wikipedia - Btrfs article

Video's

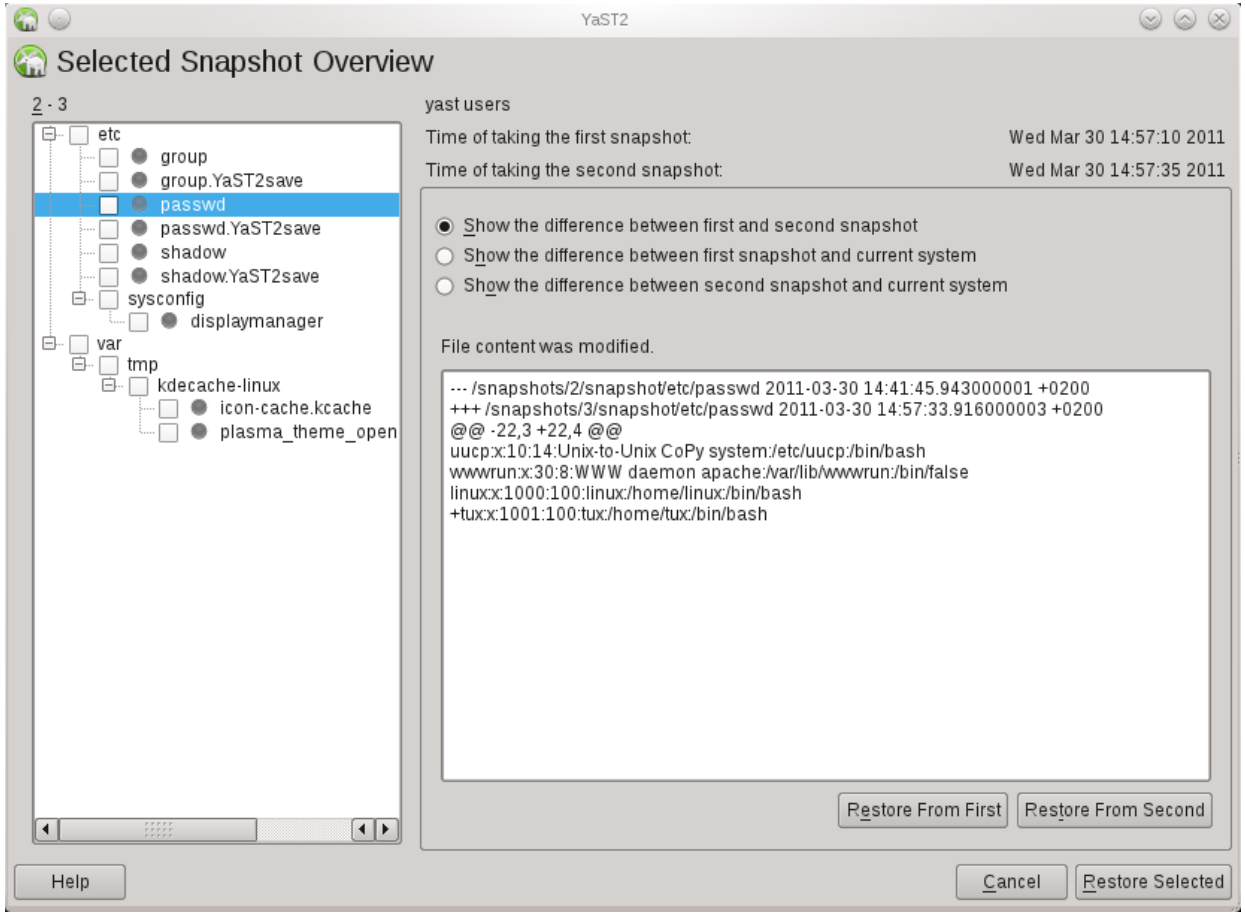
- Matthias Eckerman: Why btrfs is the Bread and Butter of Filesystems, LinuxCon 2013, New Orleans (49min, link)
- Chris Mason: Introduction to Btrfs (26min, link)
- Chris Mason: Btrfs Filesystem: Status and New Features, (May 2012, link)
- Avi Miller's Btrfs talk at LinuxConf AU (49min, Jan 2012)
  - Demo of “mount -o recovery”
  - Animations of disk usage on Ext3, XFS and Btrfs
- Douglas Fuller's talk (24min, Apr 2011)
  - Nice performance demo's

Snapper

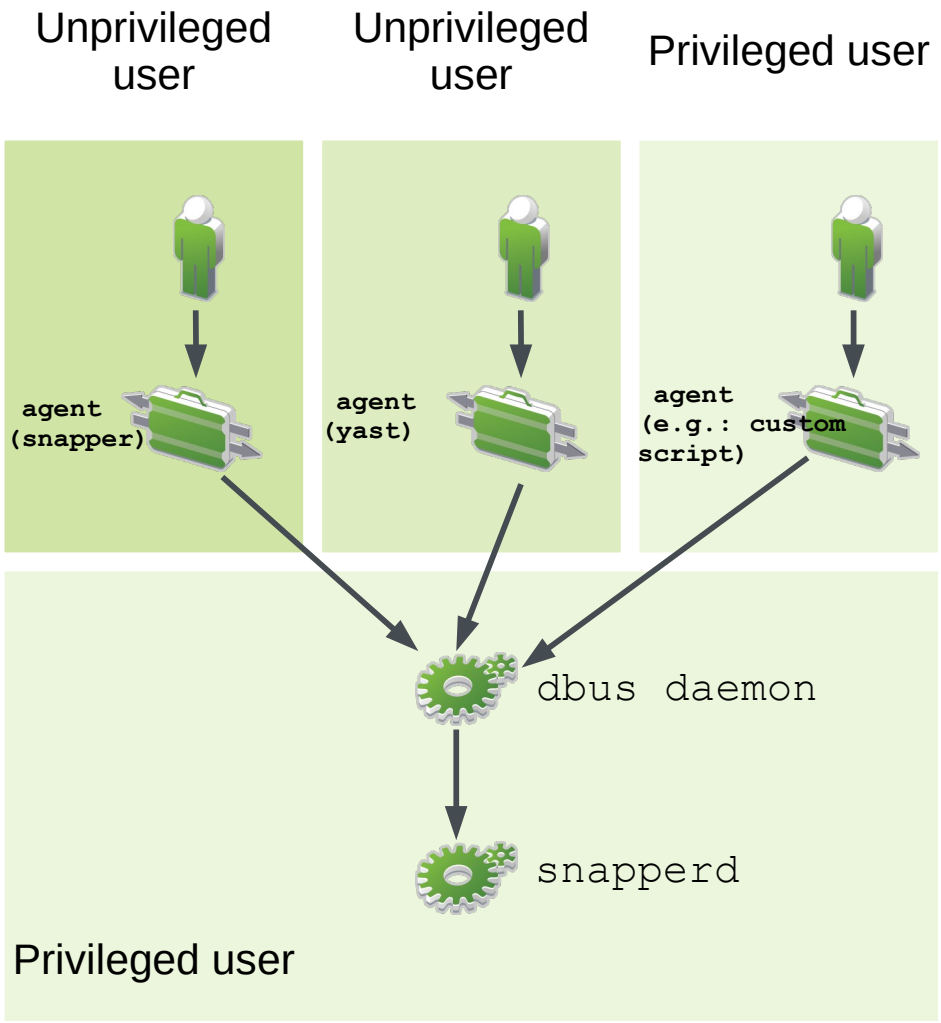
Snapshot Management With Snapper

Functions

- Automatic snapshots
- Integration with YaST and Zypp
- Rollback
- Integration points
- User created snapshots through DBus



Snapper DBus Support



- Snapper is split up:
  - `snapper` (client)
  - `snapperd` (server)
- Authorized users submit request through DBus
- `snapperd` performs actions on behalf of users
- Authorization scheme
  - Users
  - Agents

Snapper Configuration

- YaST configures snapper for the root filesystem
- `/etc/snapper/configs/` contains a file for each btrfs subvolume:
  - cleanup algorithms
  - snapshot creating permissions
- Cleanup Algorithms:
  - NUMBER_LIMIT="10"
  - TIMELINE_LIMIT_DAILY="5"
- Permissions:
  - ALLOW_USERS="tux"
  - chgrp users /home/tux

Snapper – Metadata

Meta information stored with each snapshot:

- **Type** : [Pre | Post | Single]
- **#** : Nr of snapshot
- **Pre #** : Matching “Pre” number, if type is “Post”
- **Date** : Timestamp
- **User** : User who created the snapshot
- **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 (e.g.: easily parsing from scripts)

Distro Support Status - Snapper

Supported

- SUSE® Linux Enterprise Server 11 SP2+

Available

- openSUSE 12.1 - 13.1
- Fedora
- Debian
- Ubuntu

Snapper – Planned Features

- Keep track of space usage by snapshots, utilizing qgroups

Demo 1

Snapper

- Snapper module for YaST
- Snapper integration with YaST
- Snapper command line tool
- Snapper as non-root

Agenda

- Introduction to Btrfs
- Btrfs in SUSE distro's Snapper
- Btrfs Use cases
- Summary and Questions

- On-line data migration
- Filesystem changes after software installation
- Snapper and ITIL
- Server side copy with Samba

Use Case: Filesystem Changes After Installation of Some Software

- Step 1: Create “Pre” snapshot

```
snapper create --type pre
--description "Before
installation”
```

- Step 2: Install application
- Step 3: Create “Post” snapshot

```
snapper create --type post
--pre-number $PreNR
--description “After
installation”
```

- Step 4: Compare “Pre” and “Post” situation

```
# snapper status $PreNR..$PostNr
c... /etc/ld.so.cache
+... /usr/lib/libXi.so.6
+... /usr/lib/libXi.so.6.1.0
c... /var/cache/ldconfig/aux-cache
[...]
```

Use Case:
Snapper and ITIL Change Management

 # @Begin of implementation Change:

```
snapper create \
 --type pre \
 --description "ChgMgt Work order: Upgrade syslog configuration
to forward log entries to central log server" \
 --userdata \
"WorkOrder=20120125303000012-1,
State=InProgress,Agent=jdoe@example.com"
```

 # @End of implementation Change:

```
snapper create \
 --type post --pre-number 240 \
 --description "Done: ChgMgt Work order: Upgrade syslog
configuration to forward log entries to central log server" \
 --userdata "WorkOrder=20120125303000012-1, State=Closed,
Agent=jdoe@example.com"
```

Use Case:
Server Side Copy With Samba

- Samba 4.1 supports
  Server side copy
- Client support:
  - Windows Server 2012
  - Samba?

Demo: How To Find Out The Level Of Fragmentation

Specifically for HDDs, file fragmentation caused by the CoW feature may impact disk I/O performance. To prevent performance degradation, regular de-fragmentation is advisable.

- Find the top 10 most fragmented files in the current directory:

```
filefrag * | sort -nr -k 2  |
head -10
```

- Defrag
  - files:

```
btrfs file defrag $file
```

  - directories:

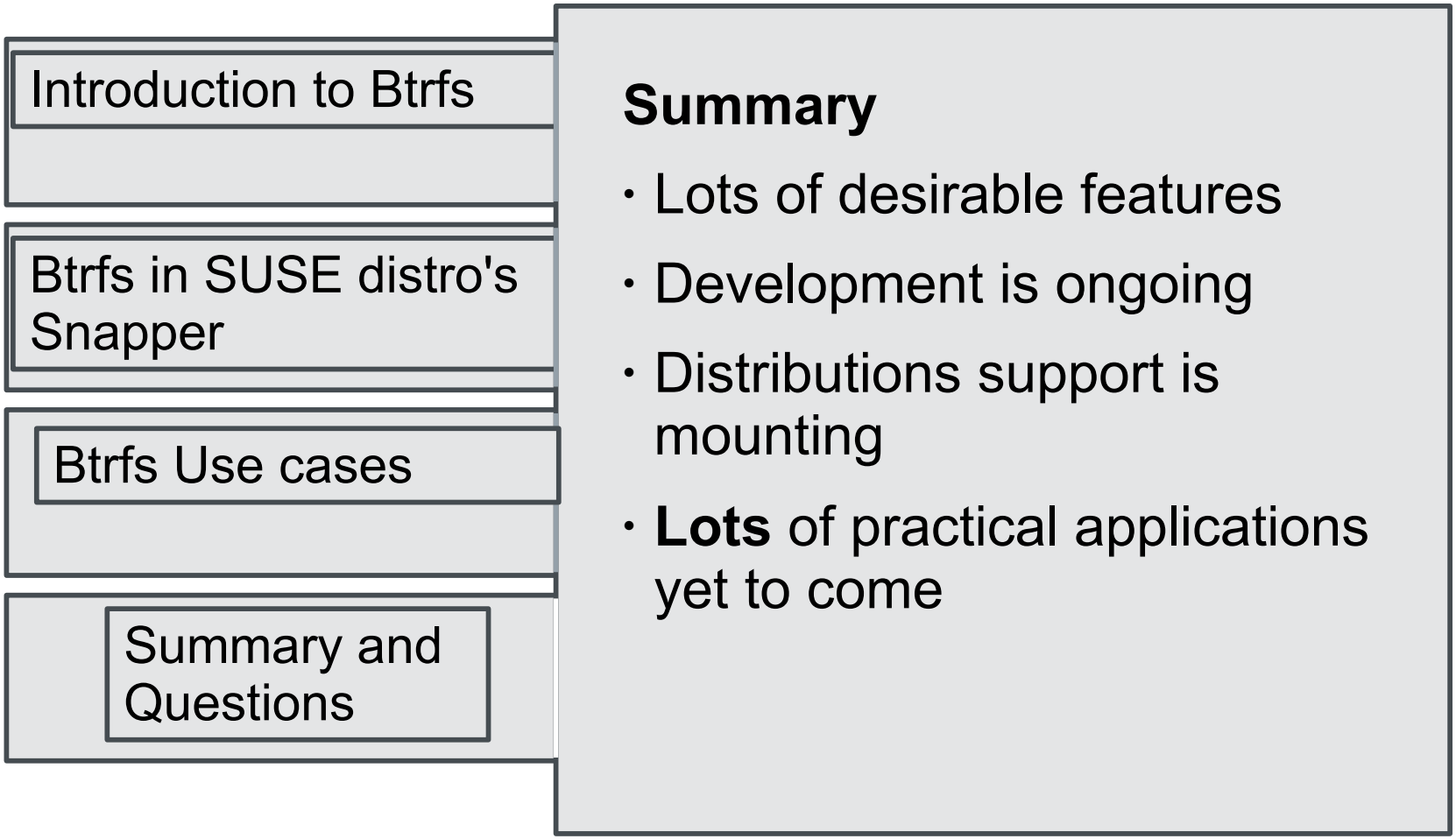
```
btrfs file defrag /var/log/
```

  - whole filesystem:

```
btrfs file defrag /
```

- Hot zones:
  - system specific: /var/log/journal/ /tmp
  - user specific: /home/

Agenda


Introduction to Btrfs
Btrfs in SUSE distro's Snapper
Btrfs Use cases
Summary and Questions
Summary
• Lots of desirable features
• Development is ongoing
• Distributions support is mounting
• Lots of practical applications yet to come

For more information please visit our website:
www.suse.com

Thank you.

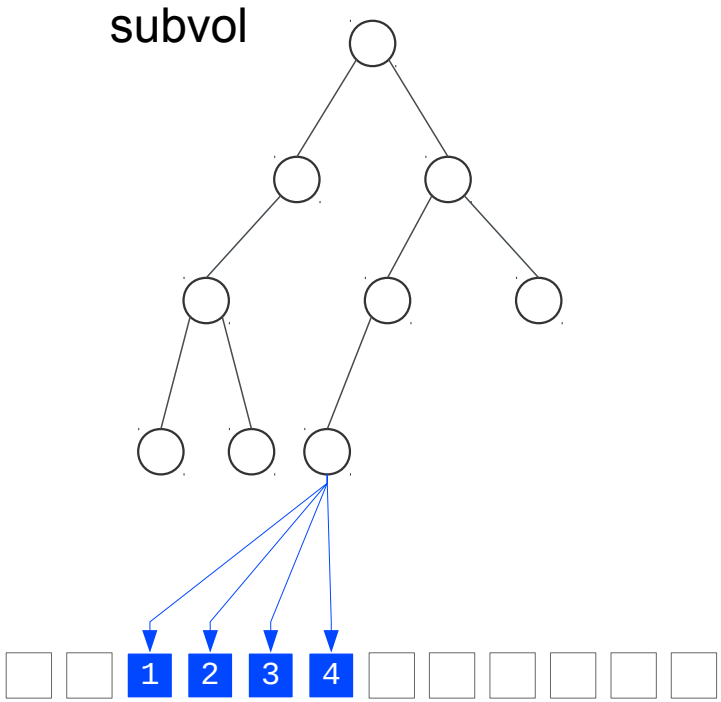
Reserve Slides: Other Use Cases

Reserve Slides: Btrfs CoW In-Depth

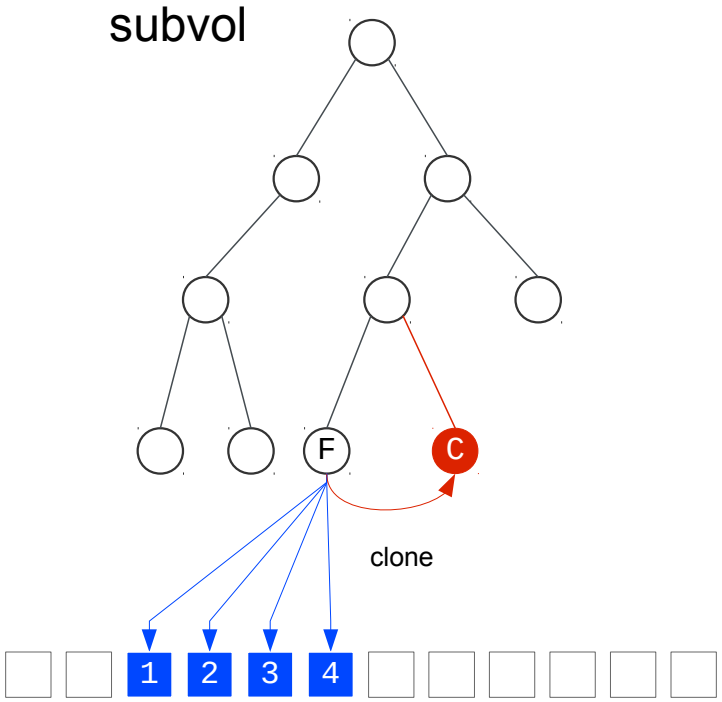
Btrfs Features:

Copy On Write Explained 1/4

1



2



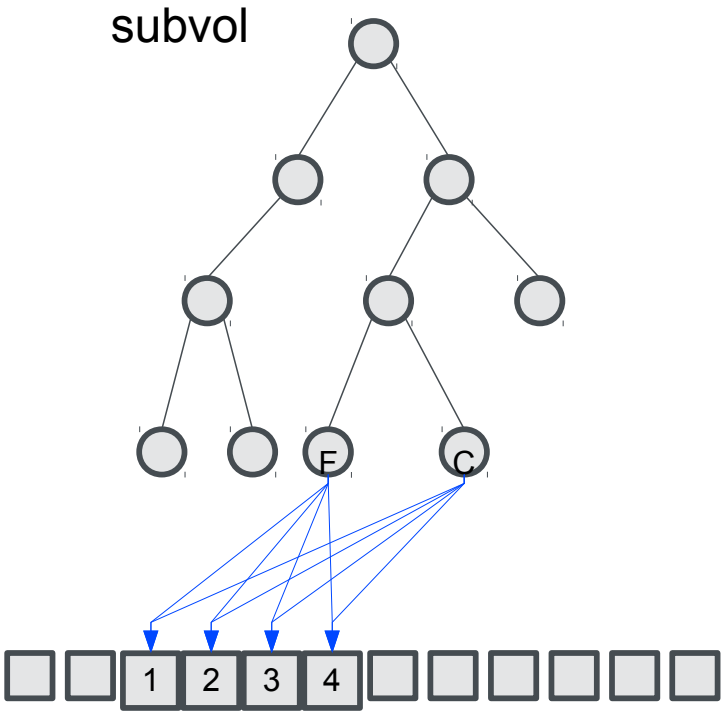
Copy Clone

```
cp --reflink=always F C
```

Btrfs Features:

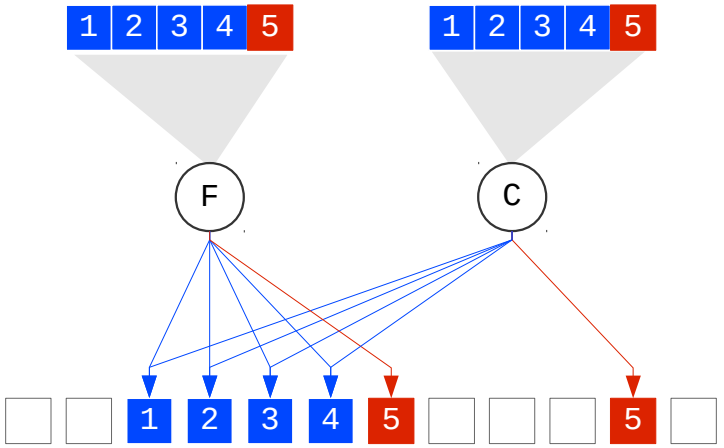
Copy On Write Explained 2/4

3



4

Append to files:
F: +1 extent
C: +1 extent

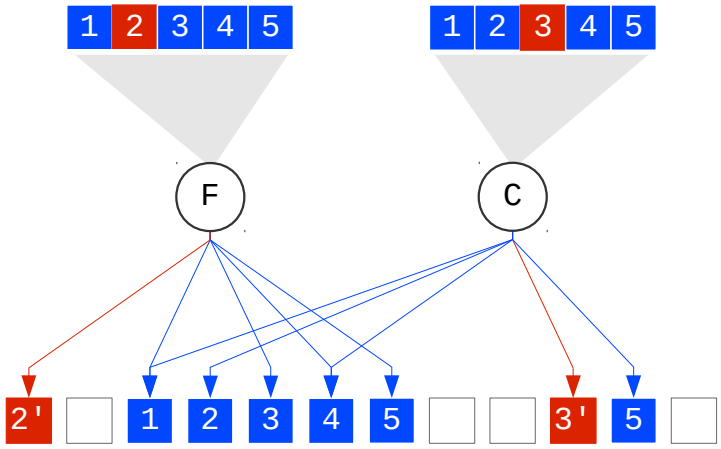


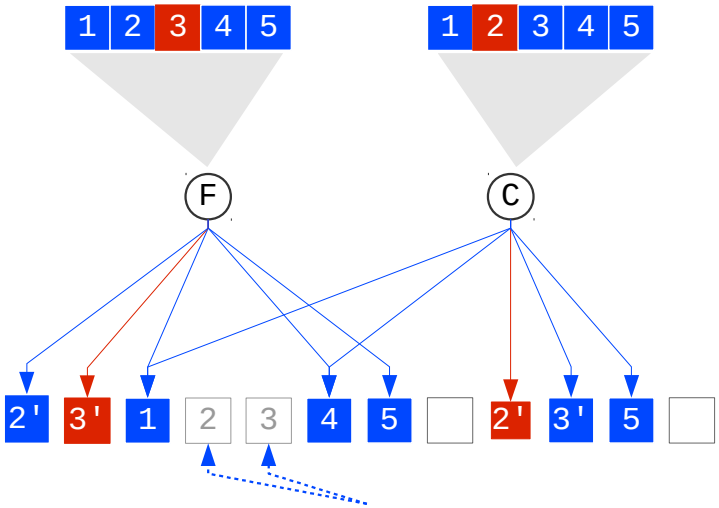
Btrfs Features:

Copy On Write Explained 3/4

5 Modify extent:
F: 2
C: 3

6 Modify extent:
F: 3
C: 2





Btrfs Features:

Copy On Write Explained 4/4

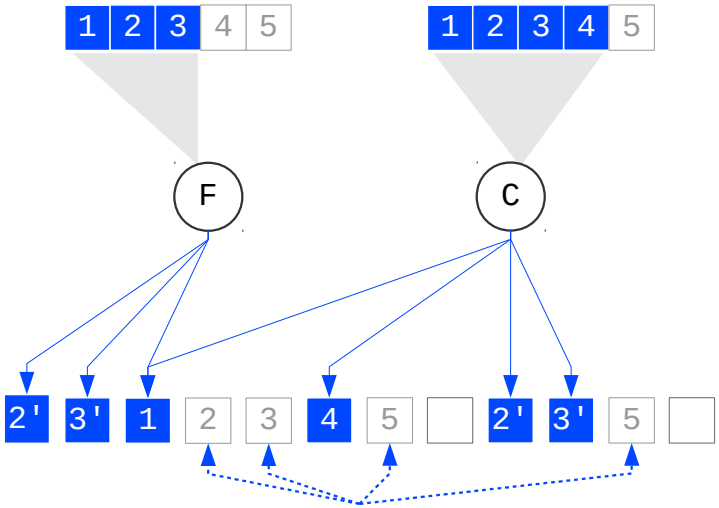
7

Truncate files:
F: -2 extent
C: -1 extent

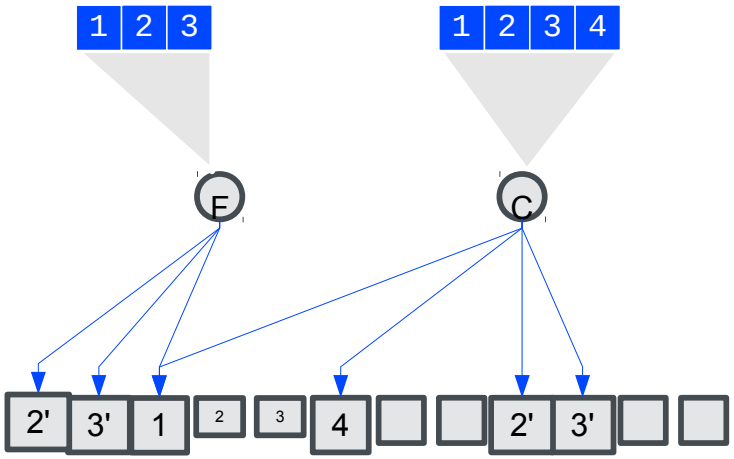
8

Trim command:

```
ATA : man 8 fstrim
SCSI: man 8 sg_unmap
```



extents needing “trimming”
('discard' mount option)



In-Depth Btrfs: Send/Receive

- available with kernel 3.6
- Allows to save the difference between subvolumes
- Use-case 1: Daily backup
  - `btrfs subvolume snapshot -r /orig /orig/Thu`
  - `btrfs send -p /orig/Wed /orig/Thu > Wed-Thu.btrfs`
  - `btrfs receive /backup < Wed-Thu.btrfs`
  - The file Wed-Thu.btrfs constains a stream of create, rename, clone, mkdir etc. commands
- Use-case 2: Speed up comparison of snapshots for Snapper

Btrfs Operations 1/2

- mkfs.btrfs
  - Different RAID algorithm for data and metadata
  - Different sized disks
- btrfs-convert
  - In-place conversion of Ext3 or Ext4 to Btrfs
  - Reversible
- Balance
  - Read all extents
  - Pass data through balancer
- Scrub
  - Identify and repair data corruption
  - Read all extents and verify checksum
  - In case of problem restore block from mirror (if avail.)
- Defrag
  - Re-allocate files to
  - Mount option `autodefrag`
  - Batch defrag

Btrfs Operations 2/2

- Create subvolume
  - `btrfs subvolume create /home`
- Create RO snapshot
  - `btrfs subvol snap -r /home/home.`date -I`
- Roll-back entire snapshot
  - "All-or-nothing"
  - mount -o subvol=
  - Atomic operation
  - For / fs boot parameter:

    `rootflags=subvol=@/.snapshots/mysnap`
- Roll-back files
  - Copy single files from snapshot to "main" filesystem
  - No atomic roll-back

Demo 2

- Make filesystem
- Btrfs utility:
  - Create subvolume
  - Create snapshot
  - Start scrub
- Mount subvolume and snapshot

Use Cases For Btrfs

Basic Btrfs HOWTOs

- How to create RAID
- Snapshots and subvols
- Grow / shrink

A few others

- System's management:
  - System snapshot and roll-back
  - Pre-patching
- Virtualization
  - Cloned VMs and containers
- Data center processes
  - Auditing
  - Change Mgt

A Few Recommendations

Btrfs on HDD

- Mount options:
  - autodefrag
  - noatime (whenever possible)
- Without “autodefrag” manually defrag on a regular basis!

Btrfs on SSD

- Mount options:
  - discard
  - ssd
  - noatime (whenever possible)
- Disk scheduler: noop
- Never defragment! → wears out SSD

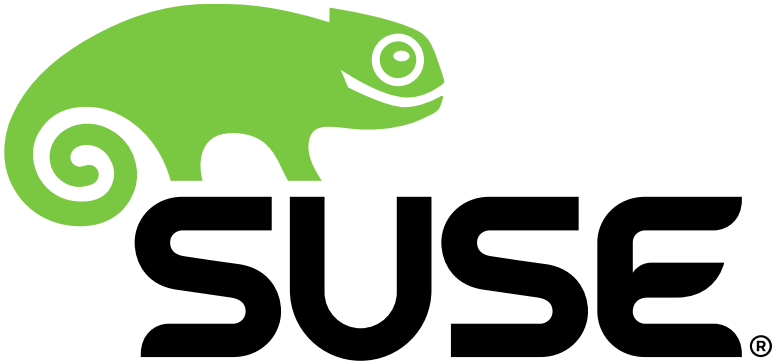
A Few Recommendations 3/3

Filesystem layout

- Depending on system purpose
- Non-mission-critical system:
  - /boot Ext3
  - / Btrfs
  - /db Ext3, ASM, raw
  - /home XFS, Ext3, Btrfs
  - /tmp tmpfs
  - /var Ext3
  - /vmstore XFS, Ext3, Btrfs

Performance

- Simple test: sustained read/write
  - SSD and HDD
  - Write test
    dd if=/dev/zero of=btrfs-demo-seq-write1 bs=1M count=4096 conv=fsync
  - Read test
    dd if=btrfs-demo-seq-write1 of=/dev/null bs=1M count=4096 iflag=nocache
- Results SSD:
  - Seq Write raw: 220 MB/s
  - Seq Write Btrfs: 200 MB/s
  - Seq Read raw: 225 MB/s
  - Seq Read Btrfs: 220 MB/s
- Results HDD:
  - Seq Write Btrfs: 32 MB/s
- For more benchmarking info see:
  - Chris Mason's Btrfs Intro
  - Avi Miller's LinuxConf AU talk
  - Douglas Fuller's talk



Corporate Headquarters
Maxfeldstrasse 5
90409 Nuremberg
Germany

+49 911 740 53 0 (Worldwide)
www.suse.com

Join us on:
www.opensuse.org