Btrfs

Linux-Dateisystem der Zukunft ?

Proseminar „Speicher- und Dateisystem“

Wintersemster 2010/2011

Schriftliche Ausarbeitung

Von: Thomas Schöbel

Studiengang: Wirtschaftsinformatik

Matrikel-Nummer: 6112130

Inhaltsverzeichnis

1. Geschichte

Btrfs ist die Abkürzung für **B**-**Tr**ee **F**ile **S**ystem und wird im Allgemeinen auch „Butter FS“ genannt. Es wird seit 2007 von der Firma Oracle Corporation entwickelt.

Btrfs kann als Linux-Analogon – sprich: Gegenstück – zum ZFS angesehen werden. Obwohl ZFS bereits Jahre zuvor entwickelt wurde, damals noch von Sun Microsystems, ist es Aufgrund der Lizensierung für die Verwendung unter Linux ungeeignet. Im April 2009 wurde Sun Microsystems von der Oracle Corporation übernommen.

Die Kernstruktur von Btrfs ist die Copy-on-Write B-Baum-Struktur. Diese wurde ursprünglich vom IBM-Forscher Ohad Rodeh auf einer Tagung bei der USENIX 2007 vorgestellt. Zusätzlich schlug er vor, dass man Referenzierungszähler für Speicherblöcke hinzufügen sollte sowie bestimmte Lockerungen der Balancing-Algorithmen in B-Bäumen. Dies soll die B-Bäume für Hochleistungsspeicherlösungen mit Copy-on-Wirte-Snapshots tauglich machen.

Im gleichen Jahr wurde Chris Mason, damaliger ReiserFS-Entwickler bei SUSE, bei der Oracle Corporation eingestellt und begann dort seine Arbeit an einem neuen Dateisystem. Dieses neue Dateisystem soll fast ausschließlich B-Bäume versenden. Nicht nur für Meta- und Nutzdateien, sondern auch zur Verfolgung der Speicherzuteilung der Bäume selbst. Der Vorteil hinter diese Idee ist es, dass man nur eine Routine braucht um sämtliche Operationen abwickeln zu können.

1.1 Ziele und Heute

Das Btrfs soll sich nicht nur vom aktuellem Linux-Standarddateisystem ext3/ext4 abheben, sondern auch von ähnlichen Dateisystemen wie ZFS, XFS oder JFS.

Aktuell ist Btrfs im Entwicklungsstadium und soll auf Wunsch des Entwicklers vorerst nur für Berichte und Leistungstests (Benchmarks) genutzt werden. Ein Produktiv-Einsatz wird ganz klar verneint, da es wohlmöglich noch zu vielen Fehlern kommen könnte, die ggf. die Datensicherheit und deren Konsistenz stören würde.

Am 09. Januar 2009 wurde das Btrfs erstmals im Linux Kernel 2.6.29 aufgenommen, was dem Btrfs sicherlich in Zukunft immer weiter zum Erfolg bringen dürfte.

2. Funktionsumfang

2.1 Erweiterter Speicherbereich

Btrfs hat einen so genannten erweiterten Speicherbereich im Vergleich mit den aktuellen Linux-Dateisystemen. Btrfs kann bis zu 2^{64} Byte adressieren. Um sich 2^{64} Byte besser vorstellen zu können, hier eine kleine Veranschaulichung:

2^{64} Byte entsprechen **16 Exabyte** oder **16.384 Petabyte** oder **16.777.216 Terabyte**

Um diese noch immer sehr großen Zahlen noch besser zu verstehen, können wir uns die Masse in aktuell verfügbaren Festplatten vorstellen. Wir gehen davon aus, dass die aktuell größten Festplatten 3 Terabyte speichern können:

16 Exabyte entsprechen **ca. 5,59 Millionen Festplatten** à 3 Terabyte

Ein Turm wäre, wenn wir die Festplatten stapeln würden, somit rund **167,7 Kilometer** hoch. Unabhängig von den ganzen Kabeln und Controllern die man bräuchte, wäre so ein Turm aktuell nicht realisierbar.

Das aktuelle Standard Linux-Dateisystem ext4 kann theoretisch maximal 1 Exabyte adressieren, ist jedoch durch e2fsprogs Beschränkt auf 16 Terabyte.

Die Daten des Teilchenbeschleuniger LHC vom Forschungszentrum Cern speichert seine Daten auf einem Peta-Grid. Das ist ein Verbund aus diversen Rechenzentren weltweit, die Speicherplatz zur Verfügung stellen. Das Peta-Grid umfasst aktuell 20 Petabyte.

2.2 Effizientes Speichern

Durch die besondere B-Taum Struktur wird ein effizientes Speichern von kleinen Dateien und Ordnern gewährleistet indem alle Daten zusammen gepackt werden und nicht einzelne Blöcke aufgeteilt werden (Vgl. Kapitel X.X.X).

2.3 Snapshots und Subvolumes

Ein Snapshot ist eine Momentaufnahme. Im Btrfs können per einfachen Befehl Snapshots von einzelnen Dateien oder ganzen Ordnern oder ganzen Subvolumes erstellt werden. Subvolumes kann man sich als virtuelle Partitionen vorstellen, die jeweils ein eigenes Dateisystem beherbergen.

Unter Btrfs erfolgt die Erstellung eines Snapshots in Bruchteilen einer Sekunde. Ein Snapshot ist so aufgebaut, dass er zunächst wenig bis gar keinen Speicherplatz belegt, der lediglich einen Verweis auf die gleichen Datenblöcke legt. Wird nun eine Datei verändert wird dies vom Snapshot-System erkannt, egal ob im Original oder im Snapshot, und erst jetzt wird die Veränderung gespeichert bzw. eine Kopie vom Original im Snapshot hinterlegt.

2.4 Integriertes RAID

Das was eigene RAID-Controller machen oder in anderen Linux-Dateisystemen eigene Software, wie z.B. mdadm, kann hier von Btrfs direkt gemacht werden. Es wurde ein sogenannten Multi-Device-RAID implementiert, das ein RAID nicht nur aus gleich großen Festplatten oder Partitionen erstellen kann, sondern auch als Misch-Verbund aus Festplatten, Partitionen oder Subvolumes.

Aktuell sind die RAID-Modi 0, 1 und 10 implementiert. Weitere wie 5, 6 und 60 sollen folgenden.

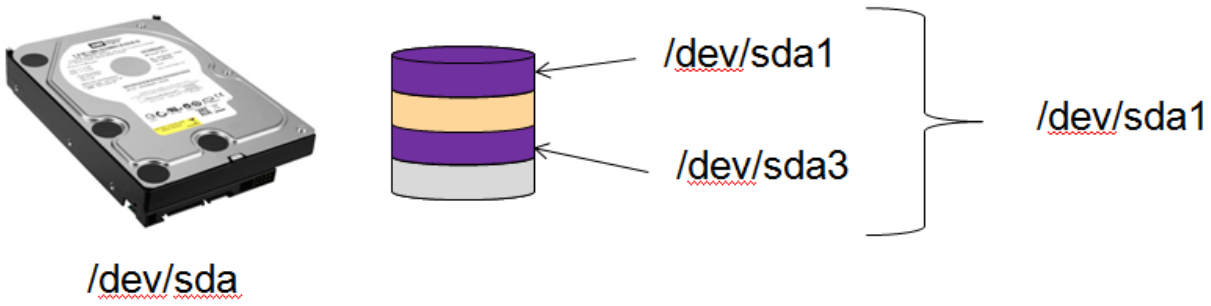

Abbildung 1

Die **Abbildung 1** zeigt wie wir eine vorhandene Partition mit einer weiteren Partition erweitern können. Sollte z.B. /dev/sda1 voll sein, so können wir mit wenigen Befehlen binnen weniger Sekunden die freie Partition /dev/sda3 einfach zu /dev/sda1 hinzufügen bzw. damit erweitern.

2.5 Weitere Funktionen

Btrfs hat, anders als die meisten anderen Linux-Dateisysteme, dynamische Inodes. Somit können eine unbegrenzte Anzahl an verschiedenen Metadaten gespeichert werden. Um die Integrität zu verbessern werden zusätzlich Checksummen für Dateien und Metadaten eingeführt.

Bisher war es so, dass Dateisysteme oft nur im Offline-Modus, dass heißt wenn Sie nicht gemountet waren bzw. nicht in Benutzung sind, überprüft werden konnten. Unter Btrfs ist es möglich, auch während des eingehängten Zustandes das Dateisystem der Festplatte zu überprüfen.

Mit der bereits erläuterten Snapshot und Subvolumes Funktionalität, wurde zusätzlich auch die Möglichkeit eines inkrementellen Backups implementiert. Vorteil dieses inkrementellen Backups ist es, dass nur Veränderungen mit abgespeichert werden. So kann man im späteren Verlauf an Speicherpunkte (Datum und Uhrzeit) im Backup zurück springen.

2.6 Vergleich mit anderen Dateisystemen

Um einen anschaulichen Vergleich/Übersicht mit anderen Dateisystemen zu erhalten, wurden folgende Main-Funktionen ausgewählt: maximaler Speicherbereich, Snapshots, Verschlüsselung, Subvolumes, Online Dateisystemüberprüfung.

	max. Speicherbereich	Snapshots	Verschlüsselung	Subvolumes	Online Systemprüfung
Btrfs	16 EiB	Ja	Geplant	Ja	Ja
Ext3	8 TiB	Nein	Nein	Nein	Nein
Ext4	16 TiB *	Nein	Nein	Nein	Nein
XFS	8 EiB	Ja	Nein	Nein	Nein
ZFS	16 EiB	Ja	Ja	Ja	Ja

* Theoretisch bis 1 EiB, jedoch durch e2fsprogs Beschränkt auf maximal 16 TiB.

2.7 Erste Betriebssystemimplementierungen

Wie bereits zu Beginn erwähnt, wurde Btrfs bereits 2009 im Linux Kernel integriert. Es gibt aber bereits eine Linux-Distribution die trotz Entwicklungsstadium Btrfs als Standard-Dateisystem verwendet: MeeGo.

MeeGo ist eine Linux-Distribution die speziell für Handys und Tablett-Lösungen entwickelt wurde. Gerade durch die Effizienz der Speicherung von kleinen Dateien und Ordnern, sowie die schnelle Suchfolge, ist Btrfs interessant für solche Distributionen wie MeeGo.

Auch durchaus größere Distributionen wie Fedorda oder Ubuntu planen Btrfs in einer der kommenden Versionen als Standard-Dateisystem zu implementieren. Aktuell kann es bereits als Zusatz-Dateisystem ausgewählt werden. Da Btrfs leider nach wie vor im Entwicklungsstadium ist, wird es wohl auch noch ein wenig dauern, ehe Fedora, Ubuntu & Co auf Btrfs umstellen werden.

3. Technischer Aufbau

3.1 B+-Bäume

Wie in der Geschichte zu Btrfs kurz angerissen, basiert das Btrfs-Dateisystem auf besonders angepassten B-Bäumen, die man auch als B^+-Bäume bezeichnen bzw. finden kann. Diese B^+-Bäume Kennzeichen sich besonders dadurch aus, dass diese nicht wie andere B-Bäume eine große Tiefe bzw. Höhe und dafür weit gefächert sind, sondern wird hier versucht einen B^+-Baum möglichst so „flach“ wie möglich zu halten. Dafür wird jedoch in die „breite“ gegangen. Das hat den Vorteil, dass beim suchen nicht endlos in die Tiefe gegangen werden muss, ehe das zu Suchende gefunden wurde.

Das Dateisystem baut darauf auf, dass alles was gespeichert werden soll in eigenen B-Bäumen landet. Somit werden Subvolumes, der Superblock, Metadaten, etc. in eigenen Bäumen gespeichert.

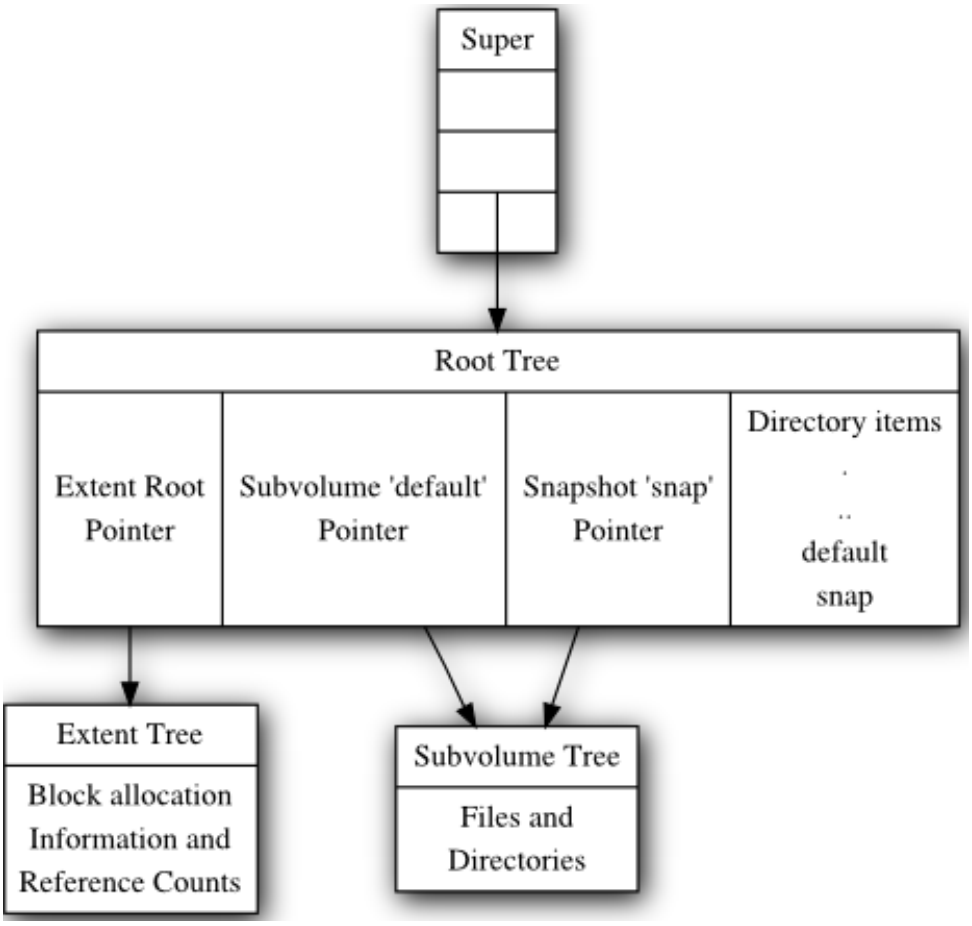

Abbildung 2

Wie in **Abbildung 2** zu erkennen, verweist der Superblock auf den Root-Tree, den „Wurzel-Baum“. Im Wurzel-Baum selbst liegen dann die Bäume für die einzelnen Subvolumes, Snapshots, Verweise zu den Metadaten und die Daten selbst. Insofern ist gut zu erkennen, dass die Bäume eher den drang haben sich auszubreiten, als in die Tiefe zu gehen.

3.2 Dateisystem

Gängige Dateisysteme speichern Ihre Daten in einzelnen Systemblöcken. Sind nun Daten kleiner als ein Systemblock selbst, geht dadurch ungenutzter Speicherplatz verloren der solange nicht adressiert werden kann, eher die Daten in dem Superblock gelöscht wurden.

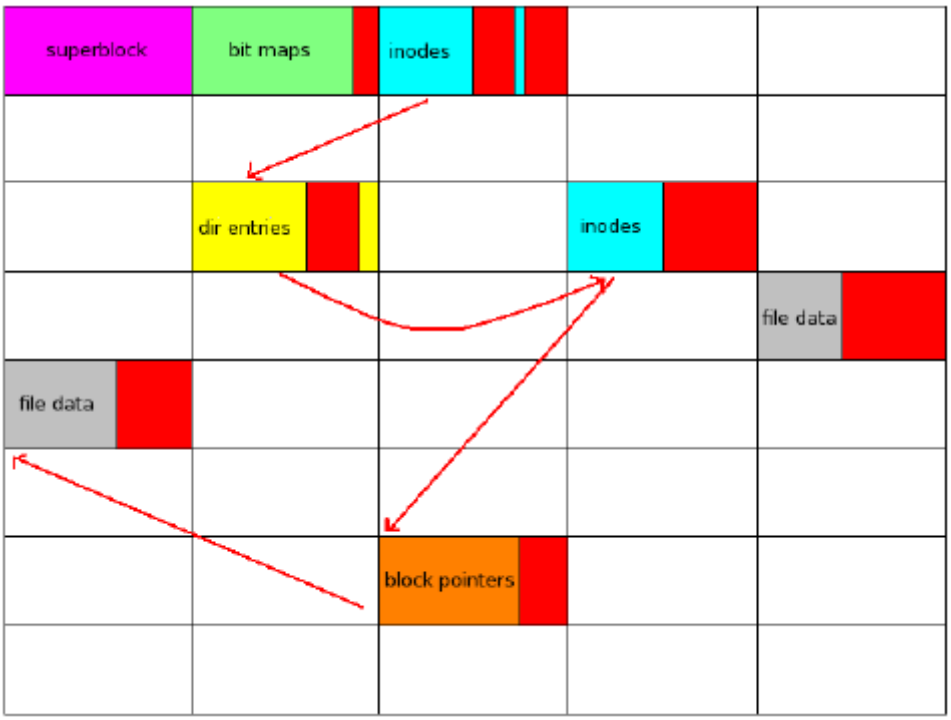

Abbildung 3

In der **Abbildung 3** ist ungenutzter Speicherplatz in Rot markiert. Es ist so z.B. zu sehen, dass die Positionen der Daten selbst wie „gewürfelt“ in die Superblocks geschrieben werden. Die roten Pfeile deuten eine Suchreihen folge an. Nehmen wir an, wir wollen die Datei aus der 5.ten Zeilen in der 1. Spalte finden. So müssen wir vorne in Zeile 1, Spalte 1 beginnen zu suchen. Wir springen somit zunächst von dem Inode der die Informationen über das Verzeichnis gespeichert hat, indem sich unsere zu suchenden Daten befinden, selbst in das Verzeichnis. Danach holen wir uns die Inode Daten für unsere eigentlichen Daten, laden jedoch zunächst auf einem Block-Pointer der uns zu der eigentlichen Positionen unserer gesuchten Daten springen lässt. Wir haben somit erst nach 4-Schritten unsere eigentlichen Daten gefunden und können diese nun aufrufen. Dies geschieht natürlich in wenigen Bruchteilen einer Sekunde, kann aber je größer und voller ein Dateisystem ist immer länger dauern.

Beim Btrfs System werden die Daten nicht in Superblöcken geschrieben, sondern alle sortiert hintereinander gereiht.

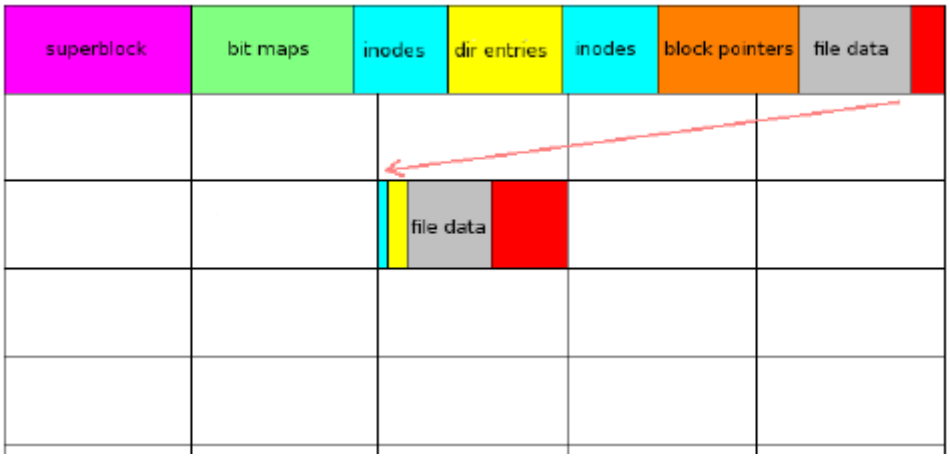

Abbildung 4

Wie in **Abbildung 4** zu erkennen, müssen wir in unserem Beispiel hier auch in der Zeile 1, Spalte 1 beginnen zu suchen, brauchen jedoch nur eine Suchfolge auszuführen, da alle relevanten Daten passend aneinander gestellt wurden.

Dies wird dadurch erreicht, dass hier alle Daten, egal ob Datei, Ordner oder Metadaten, mit einer Object-ID und einem Type versehen werden. Einfach gesehen sieht so eine Markierung wie folgt aus:

<1, Ordner>

Eine Datei in unserem Ordner würde die Markierung beinhalten, dass die Datei „A“ in dem Ordner „Test“ liegt.

4. Benchmarks

Da das Btrfs System aktuell nur für Reviews und Benchmarks freigegeben ist, gibt es natürlich bereits eine Vielzahl an Benchmarks. Das Problem ist jedoch, man kann vieles objektiv Vergleichen mit anderen Dateisystemen, jedoch muss man im Hinterkopf behalten das Btrfs noch nicht „stable“ ist.

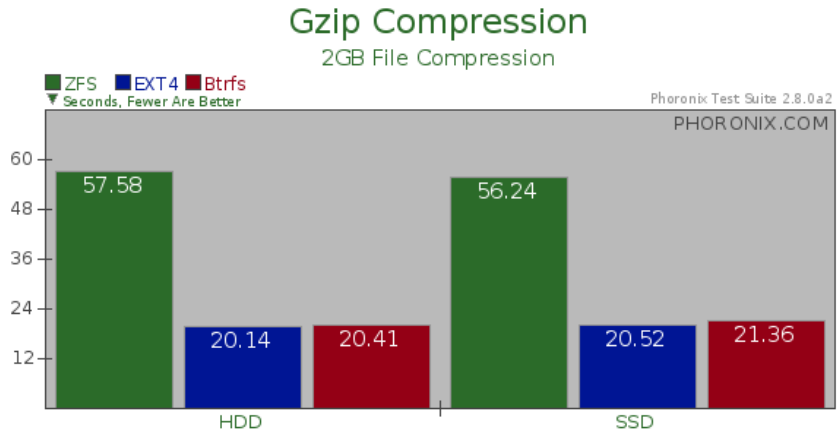

Abbildung 5

Abbildung 5 zeigt einen Benchmark von Btrfs zusammen mit EXT4 und ZFS das über Fuse implementiert wurde. Die Fuse-Implementierung hat sicherlich die schlechten Resultate zur Folge. Wie gut zu erkennen ist, schlägt sich Btrfs bereits ganz gut gegen Ext4 durch und kann dort mithalten.

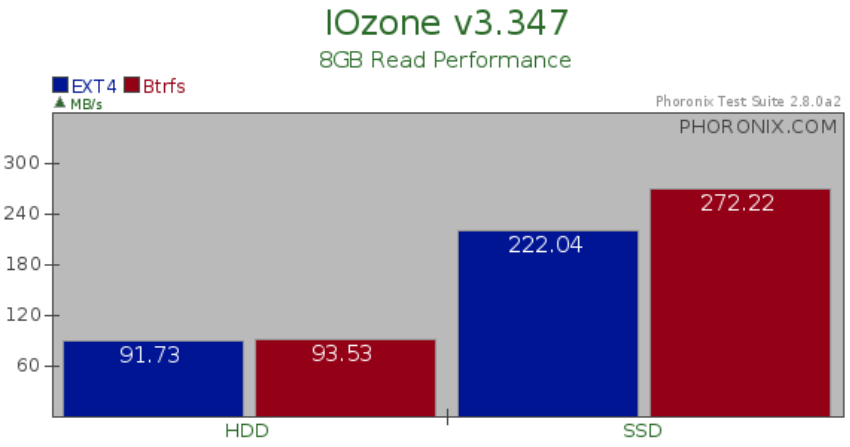

Abbildung 6

Mit Hilfe IOzone, einem Benchmarktool, wurde gemessen, wie lange es dauern würde eine 8GB große Datei zu lesen. **Abbildung 6** zeigt, dass Ext4 und Btrfs auf einer normalen Festplatte (HDD) ähnliche Performancewerte haben. Wird hingegen eine SSD genutzt, ist die Performance von Btrfs Signifikat höher. Es ist davon auszugehen, dass bei kommenden Releases die Performance auch für Festplatten (HDD) angepasst werden.

5. Zukunftsmusik

Für dieses Jahr war im Januar das erste „Stable“-Release 1.0 geplant, was jedoch bis heute noch nicht veröffentlicht wurde. Leider gibt es seitdem auch keine weiteren konkreten Äußerungen oder Ankündigungen, wann mit dem Release gerechnet werden kann. Es wird öffentlich in einigen Foren vermutet, dass wohlmöglich im März bis Mai mit dem Release gerechnet werden kann.

Es ist damit zu rechnen, dass Btrfs als Standard-Dateisystem in weiteren Distributionen wie Ubuntu und Fedora im Laufe der kommenden Jahre implementiert wird.

Parallel wird aktuell ein neues Netzwerkdateiprotokoll mit dem Namen CRFS entwickelt, dass auf Btrfs aufbauen ist. Es bleibt hier abzuwarten welchen Funktionsumfang es haben wird und welches bereits vorhandenen Netzwerkdateiprotokoll, wie z.B. NFS, es in Zukunft ablösen könnte.

6. Fazit

Um die Einleitungsfrage zu beantworte, ob Btrfs das Linux-Dateisystem der Zukunft wird: **Ja!**

Es ist davon auszugehen, dass bei dem aktuellem schon implementiertem Umfang an Funktionen und den geplanten, es über kurz oder lang das aktuelle Standard-Dateisystem Ext3/Ext4 ablösen wird. Alleine vom Interesse aktueller großer Distributionen, ist davon auszugehen.

Was in Zukunft spannend werden wird, sind Benchmarks mit den ersten „Stable“-Releases. Diese werden zeigen ob Btrfs auch als Performance-Wunder gilt, wie es an einigen Stellen zu lesen ist. Die hier gezeigten Benchmarks zeigen schon positiven Werte, sind aber noch ausbau fähig. Natürlich muss unabhängig von den Benchmarks auch in der Praxis geguckt werden, ob sich dort ein Performance-Schub spüren lässt.

Beim Wechsel von Ext3/Ext4 muss jedoch auch für Systemadministratoren ein reibungsloser Übergang gewährleistet sein. Denn schließlich darf Btrfs und ein Umzug nicht mehr Arbeit verschlingen, als der eigentliche Nutzen von Btrfs.

7. Quellen

Aus folgenden Text-Quellen stammt meine Ausarbeitung:

- http://dclug.tux.org/200908/BTRFS-DCLUG.pdf
- http://lwn.net/Articles/342892/
- http://www.heise.de/open/meldung/MeeGo-Projekt-waehlt-Btrfs-zum-Standard-Dateisystem-998662.html
- http://de.wikipedia.org/wiki/Btrfs
- https://btrfs.wiki.kernel.org/index.php/
- https://help.ubuntu.com/community/btrfs
- http://event.on24.com/event/23/31/61/rt/1/documents/slidepdf/btrfsweb.pdf
- http://www.pro-linux.de/artikel/2/1456/btrfs-linux-dateisystem-der-zukunft.html
- Div. Benchmarks von http://www.phoronix.com
- http://www.heise.de/open/artikel/Snapshots-Subvolumes-Performance-224650.html
- http://en.wikipedia.org/wiki/Comparison_of_file_systems
- http://www.thevarguy.com/2010/08/02/ubuntu-10-10s-new-file-system-btrfs/
- http://oss.oracle.com/projects/crfs/

Folgende Abbildungsquellen wurden genutzt:

Abbildung 1 http://www.itespresso.de/files/2007/images/Alternate_Festplatte_SATA.jpg

Abbildung 2 https://btrfs.wiki.kernel.org/images-btrfs/d/d7/Design-roots.png

Abbildung 3 http://lwn.net/images/ns/kernel/btrfs/oldskool.png

Abbildung 4 http://lwn.net/images/ns/kernel/btrfs/newskool.png

Abbildung 5 http://www.phoronix.com/data/img/results/btrfs_zfs_ssd/1.png

Abbildung 6 http://www.phoronix.com/data/img/results/btrfs_zfs_ssd/4.png