Btrfs
Current Status and Future Prospects

Oct 13 2014
Satoru Takeuchi <takeuchi_satoru@jp.fujitsu.com>
Fujitsu LTD.
Agenda

- Background
- Core Features
- Developments Statistics
- Future Prospects
Agenda

- Background
- Core Features
- Developments Statistics
- Future Prospects
Background

- Fujitsu has developed Btrfs for Mission Critical (MC) systems since 2010

Requirements of MC systems

- High robustness
  - Don’t crash: data duplication
  - Error detection: checksum
  - Repair, recovery: snapshot, backup/restore, repairing tools

- High availability: Should work 365 days/24h
  - Limited maintenance time: enlarge storage size and backup online

- Btrfs is designed for such the requirements
Agenda

- Background
- Core Features
- Developments Statistics
- Future Prospects
Core Features

- Multi-volumes
- Copy-on-Write Style Update
- Data/Metadata Checksum
- Subvolume
- Snapshot
- Transparent Compression
Multi-volumes

- Btrfs file system can consists of multiple volumes
  - Low layered and low overhead than LVM
  - Many features: RAID, online \{add/remove/replace\} devices

```
# mkfs.btrfs /dev/sd{a,b,c}1
```

![Diagram showing Btrfs file system and its components]
Copy-on-Write (CoW) style update

- Btrfs uses CoW style data/metadata update
  - Safer than overwrite style update by design

- Overwrite style: Update the data in place
  - System crash => data become inconsistent

- CoW style: Copy, update, and replace pointer
  - System crash => data keep consistency
CoW versus Overwrite

1,000 surprising power failure test
- Linux File System Analysis for IVI system, Mitsuharu Ito, Fujitsu

Result
- Ext4: Metadata was corrupted
- Btrfs: Worked fine without any problem
- In my internal similar testing, XFS corrupted too.
Data/Metadata Checksum

- Btrfs has checksum for each data/metadata extent to detect and repair the broken data.
- When Btrfs reads a broken extent, it detects checksum inconsistency.
  - With mirroring: RAID1/RAID10
    - Read a correct copy
    - Repair a broken extent with a correct copy
  - Without mirroring
    - Dispose a broken extent and return EIO
- With “btrfs scrub”, Btrfs traverses all extents and fix incorrect ones.
  - Online background job.
Subvolume

- A subvolume is a file system inside file system
  - Can be treated as a file system root
    - Mountable: most mount options are shared
    - Own inode namespace and quota limit
  - Efficient: Available space is shared

```
# btrfs subvolume create sub
```
Snapshot

- Copy of a subvolume
  - Far faster than LVM
    - Not a full copy, but only update metadata in CoW style
- Readonly snapshot: with \texttt{--r} option
- Incremental snapshot: snapshot of snapshot

```bash
# btrfs subvolume snapshot [-r] ./sub ./snap
```

Reference count

Capture a snapshot

Update data C in a snapshot
Performance of Snapshot: Btrfs versus LVM

1. Copy the following data to a volume
   - Consists of 100 directories and 100 files for each directory
     - File size: 1MB

2. Capture a snapshot of the volume

<table>
<thead>
<tr>
<th>Hardware Environment</th>
<th>Software Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>• PRIMERGY RX300 S6</td>
<td>• Red Hat Enterprise Linux 7.0</td>
</tr>
<tr>
<td>• CPU: Intel Xeon X5690 3.47GHz x12 core</td>
<td>• File systems</td>
</tr>
<tr>
<td>• Memory: 16GiB</td>
<td>• Btrfs</td>
</tr>
<tr>
<td>• Storages: 100GB HDD x 2</td>
<td>• Data/metadata: RAID1</td>
</tr>
<tr>
<td></td>
<td>• Other options: default</td>
</tr>
<tr>
<td></td>
<td>• XFS: default options</td>
</tr>
<tr>
<td></td>
<td>• Volume manager for XFS</td>
</tr>
<tr>
<td></td>
<td>• dm-thinp: chunksize is 256KiB</td>
</tr>
<tr>
<td></td>
<td>• LVM: RAID1</td>
</tr>
</tbody>
</table>
**Result**

- **Copy: Btrfs > LVM >>> dm-thinp**
- **Snapshot: Btrfs > dm-thinp >>> LVM**

<table>
<thead>
<tr>
<th>Volume type</th>
<th>Copy</th>
<th>Snapshot</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Without page cache</td>
</tr>
<tr>
<td>Btrfs</td>
<td>106s</td>
<td>0.126s</td>
</tr>
<tr>
<td>XFS on dm-thinp</td>
<td>209s</td>
<td>0.260s</td>
</tr>
<tr>
<td>XFS on LVM</td>
<td>133s</td>
<td>1.03s</td>
</tr>
</tbody>
</table>
**Transparent compression**

- Automatically compress/expand file data on I/O
- Low space consumption and high I/O performance
  - Need some extra CPU time
- Usage: `mount -o compress={lzo,zlib} <device> <mnt point>`
  - Can also be enabled/disabled for each file

---

![Diagram showing page cache and compress/expand system and storage with and without compression](image-url)
Agenda

- Background
- Core Features
- Developments statistics
- Future Prospects
Developments statistics

- Patch statistics
- Performance
- Summary
Patch Statistics

![Bar chart showing patch statistics for different versions (v3.0 to v3.17). The chart indicates the number of bugfixes, functional enhancements, performance enhancements, and cleanup entries per version.]

Copyright 2014 FUJITSU LIMITED
Patch Statistics: Tips of v3.17

Rejected by Linus
Patch statistics: Main changes

- auto defrag
- scrub
- replace subcommand
- quota
- send/receive
- btrfsck
- repair
- Improve sync write ~60%
- RAID5/6
- Inode properties
- offline dedup
- Improve error handling
- Inode properties
- performance enhancement
- functional enhancement
- cleanup
Fujitsu’s contribution

- btrfsck, error handling
- fast {random/async} write
- LZO compression
- read only snapshot
- random Bug fixes
- enrich xfstests
Fujitsu’s contribution: btrfs-progs

- fsck
- error handling
- random bug fixes
- enrich xfstests
- documentation
## Performance measurement

<table>
<thead>
<tr>
<th>Hardware Environment</th>
<th>Software Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>• PRIMERGY TX300 S6</td>
<td>• Benchmark software: filebench</td>
</tr>
<tr>
<td>• CPU: Xeon x5670 x 2</td>
<td>• Kernel: 3.14.11, 3.15.4, 3.16.3, and 3.17-rc2</td>
</tr>
<tr>
<td>• 12 core</td>
<td>• I/O scheduler: deadline</td>
</tr>
<tr>
<td>• HT is disabled</td>
<td>• File systems: Btrfs(single volume), XFS, and ext4</td>
</tr>
<tr>
<td>• Memory: 4GB</td>
<td>• default mkfs options and mount options</td>
</tr>
<tr>
<td>• HDD: 300GB x 1</td>
<td></td>
</tr>
<tr>
<td>• MegaRAID SAS, HITACHI HUS156030VLS600</td>
<td></td>
</tr>
</tbody>
</table>
The result: Compare with other file systems

Kernel version: v3.17-rc2

- btrfs
- ext4
- xfs
The result: Compare with old Btrfses
VFS has also improved performance

Accomplished by VFS layer performance enhancement
Summary

- Ready to use without RAID5/6
  - Performance: OK
  - Stability: OK
    - # of new features has decreased
    - Test coverage has increased
- Features: almost OK
  - RAID5/6: Lack of scrub and replace subcommands
- RAID1 and RAID10 are the best choice
  - Especially safe and stable
Agenda

- Background
- Core Features
- Developments statistics
- Future Prospects
Future Prospects: Fujitsu’s plan

- **RAID 5/6 enhancement**
  - Add scrub and replace subcommands
    - We’re testing patches now and will post it to linux-btrfs ML soon
  - Add five tests for these features to xfstests

- **Further enhancement of robustness and performance**
  - Repairing tools and so on

- **Education and documents for this purpose**
  - Operation know-how
    - Btrfs operations are different from other file systems
      - e.g. Btrfsの基礎 part1 機能編(It’s in Japanese. Now translating to English…)
        [http://www.slideshare.net/fj_staoru_takeuchi/btrfs-part1](http://www.slideshare.net/fj_staoru_takeuchi/btrfs-part1)

- **File system structure**

- **Code logic**
Future Prospects: Btrfs users are increasing

- Will be used by OpenSuSE13.2 as its default
- Supported by Ubuntu
- Available with RHEL7 as tech-preview
- Will be used for In Vehicle Infortaiment(IVI) system

  - Linux File System Analysis for IVI system, Mitsuharu Ito, Fujitsu
Conclusion

- Please try Btrfs
- It’s ready to use
  - RAID1/10 are the best choice
  - RAID5/6 need some more work
  - Recommend the newest stable kernel
References

- **Linux File System Analysis for IVI system, Mitsuharu Ito, Fujitsu**
  

- **Btrfsの基礎 part1 機能編**
  
  [http://www.slideshare.net/fj_staoru_takeuchi/btrfs-part1](http://www.slideshare.net/fj_staoru_takeuchi/btrfs-part1)

- **Linux-btrfs ML**
  
  [linux-btrfs@vger.kernel.org](mailto:linux-btrfs@vger.kernel.org)

- **Btrfs wiki**
  
shaping tomorrow with you