ZFS

ZFS 101—Understanding ZFS storage and performance

https://lists.debian.org/debian-user/2012/05/msg01026.html

Features:

- data pools (tanks) are abstraction aggregate block devices (simple, mirror, raidz, spares, etc)
- data set is created on data pool or another (parent) data set.
- whole data pool space is shared between dataset (no fixed partition size problem). Size of data set (and its descendants) can be limited using quota
- compression
- block level deduplication (not usable for emails with attachment, where attachment are shifted to different offset)

OpenZFS2.0.0 (Dec 20) https://github.com/openzfs/zfs/releases/tag/zfs-2.0.0:

- Sequential resilver (rebuild only used by data portions)
- Persistent L2ARC cache (survives between reboots)
- ZSTD
- Redacted replication (replicate with some data excluded)
- FreeBSD and Linux unification

Proposed use case: POOL created on encrypted LUKS block device.

```
POOL
  |-- /filer (quota)
  |     |- foto
  |     |- mp3 (dedup)
  |     |- movies
  |     +- backup (copies=2, compression)
  |
  |-- /home (compression, dedup, quota)
  +-- /var (quota)
        +- log (compression)
```

ZFS implementations

ZFS-Fuse 0.7 is using old pool version 23, where ZFSonLinux is using pool version 28. zfs-fuse vs. zfsonlinux

Creating ZFS dataset

```
zpool create INBOX /dev/loop0 /dev/loop1 /dev/loop2 /dev/loop3
```

```
# zpool list
NAME    SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
INBOX   780M   448K   780M     0%  1.00x  ONLINE  -
```

```
# zpool status
  pool: INBOX
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        INBOX       ONLINE       0     0     0
          loop0     ONLINE       0     0     0
          loop1     ONLINE       0     0     0
          loop2     ONLINE       0     0     0
          loop3     ONLINE       0     0     0

errors: No known data errors
```

Dataset “INBOX” is also automatically created based on zpool name “INBOX”. It is mounted as /INBOX

```
# zfs list
NAME    USED  AVAIL  REFER  MOUNTPOINT
INBOX   400K   748M   112K  /INBOX
```

Mount dataset

```
zfs mount INBOX
```

Create more datasets in pool

```
zfs create <pool name>/<data set name>
```

Add new block device (disc) to online pool

```
zpool add INBOX /dev/loop4
```

Deduplication

```
zfs set dedup=on INBOX
```

New attributed applies only to newly written data.

Tests For test I was using 3 files 16MB each of random data (/dev/urandom): B1, B2 and B3 Above 3 files takes 38,6M on disc:

```
# zdb -S INBOX
Simulated DDT histogram:

bucket              allocated                       referenced
______   ______________________________   ______________________________
refcnt   blocks   LSIZE   PSIZE   DSIZE   blocks   LSIZE   PSIZE   DSIZE
------   ------   -----   -----   -----   ------   -----   -----   -----
     1      309   38.6M   38.6M   38.6M      309   38.6M   38.6M   38.6M
 Total      309   38.6M   38.6M   38.6M      309   38.6M   38.6M   38.6M

dedup = 1.00, compress = 1.00, copies = 1.00, dedup * compress / copies =
1.00
```

Additionaly one big file with content B1|B2|B3 was added to filesystem:

```
# zdb -S INBOX
Simulated DDT histogram:

bucket              allocated                       referenced
______   ______________________________   ______________________________
refcnt   blocks   LSIZE   PSIZE   DSIZE   blocks   LSIZE   PSIZE   DSIZE
------   ------   -----   -----   -----   ------   -----   -----   -----
     2      384     48M     48M     48M      768     96M     96M     96M
 Total      384     48M     48M     48M      768     96M     96M     96M

dedup = 2.00, compress = 1.00, copies = 1.00, dedup * compress / copies =
2.00
```

Additionaly one big file with content B1|B2|B3|B1|B2|B3 was added to filesystem:

```
# zdb -S INBOX
Simulated DDT histogram:

bucket              allocated                       referenced
______   ______________________________   ______________________________
refcnt   blocks   LSIZE   PSIZE   DSIZE   blocks   LSIZE   PSIZE   DSIZE
------   ------   -----   -----   -----   ------   -----   -----   -----
     4      384     48M     48M     48M    1.50K    192M    192M    192M
 Total      384     48M     48M     48M    1.50K    192M    192M    192M

dedup = 4.00, compress = 1.00, copies = 1.00, dedup * compress / copies =
4.00
```

Next, new file with content 0|B1|B2|B3 (one dummy byte plus B1|B2|B3) was added:

```
# zdb -S INBOX
Simulated DDT histogram:
```

```
bucket              allocated                     referenced
______   ______________________________   ______________________________
refcnt   blocks   LSIZE   PSIZE   DSIZE   blocks   LSIZE   PSIZE   DSIZE
------   ------   -----   -----   -----   ------   -----   -----   -----
     1      385   48.1M   48.1M   48.1M      385   48.1M   48.1M   48.1M
     4      384     48M     48M     48M    1.50K    192M    192M    192M
 Total      769   96.1M   96.1M   96.1M    1.88K    240M    240M    240M

dedup = 2.50, compress = 1.00, copies = 1.00, dedup * compress / copies =
2.50
```

So ZFS cannot match shifted data and make deduplication!

Additional simple test. Two files:

0	B1	0	B2	0	B3	0
0	B1	B2	B3			

Only beginning of both files |0|B1| was deduplicated (16MB saved)

ZFS provides block level deduplication based on block checksums which we got almost for free.

Compression

Enable compression and dedupliaction in parent dataset (will be inherited by childs)

```
zfs set compression=on INBOX
```

Possible parameters for compression=on | off | lzjb | gzip | gzip-[1-9] | zle New attributed applies only to newly written data. For test data I was using Maildir with some huge e-mails.

compression	**logical size**	**physical size**	**ratio**
off	702 MB	703 MB	1.0
on = lzjb	702 MB	531 MB	1.32
gzip-1	702 MB	374 MB	1.87
gzip=gzip-6	702 MB	359 MB	1.95
gzip-9	702 MB	353 MB	1.96
squashfs		365 MB	

zdb -S INBOX zdb -b INBOX

```
zfs get compressratio
```

References:

http://docs.oracle.com/cd/E19253-01/819-5461/6n7ht6qu6/index.html
https://wiki.freebsd.org/ZFSQuickStartGuide

http://www.funtoo.org/ZFS_Fun

http://constantin.glez.de/blog/2011/07/zfs-dedupe-or-not-dedupe

http://www.oracle.com/technetwork/articles/servers-storage-admin/o11-113-size-zfs-dedup-1354231.html