ish

How to install FreeBSD 7.0 under ZFS

ZFS

ZFS is an exciting new file system developed by Sun and recently ported to FreeBSD. Many people are excited by the possibilities of ZFS (including us) as it promises to simplify a great many things. It offers:

- increased reliability through checksums, multiple copies of data and self-healing RAID
- elimination of that dreaded "oh, if only I'd made the /var partition larger" feeling. Partitions can now be resized at any time and in fact can each be allocated up to the full size of the storage media.
- built-in compression and encryption
- built-in NFS file sharing
- clean, easy to use toolset for creating storage pools, volumes and much more
- snapshots and rollbacks for backups

Read more about some of the powerful commands here

Installing ZFS on FreeBSD

Since it is early days for ZFS on BSD, the installer doesn't yet support ZFS natively. So there are a few tricks to getting it up and running. Also, you cannot boot directly from a ZFS partition since adding that functionality to the boot loader in FreeBSD is a huge undertaking.

Step One: installing FreeBSD

- Boot up on the FreeBSD Current 7.0 CD
- Choose Country and Locale
- Go Custom install

- Partition fdisk with Auto option (one slice, whole disk)
- Disklabel with options

```
A: 512Mb    UFS2  /
B: swap
D: rest of disk
```

To create D you'll need to enter any mount point you want and then use the M option to clear it. This ensures that it will not mount or be created as a file system.
* Distribution choose Minimal install
* Media Select CD/DVD

System will install to the small 512Mb UFS root partition you created.

Step Two: creating the ZFS pool

- Once installation is complete, remove CD and reboot into our new FreeBSD system.
- Boot into FreeBSD partition and select "4" for single user mode.
- Hit ENTER to accept /bin/sh shell.

```
# mount -w /
```

Now create a Disk pool using the D label we prepared during the install. In this example we have a SATA disk at ad4.

```
# zpool create tank /dev/ad4s1d
```

Create some extra/common mountpoints

```
# zfs create tank/usr
# zfs create tank/var
# zfs create tank/tmp
```

Have a look to what we have done

```
#df -h
```

```
#zfs list
```

Beautiful isn't it? :)

Edit /etc/rc.conf and enable ZFS

```
# echo 'zfs_enable="YES"' >> /etc/rc.conf
```

Now copy the UFS bootable slice to the ZFS mountpoint. This gives us a workable FreeBSD installation under ZFS.

```
# find -x / | cpio -pmd /tank
```

Step Three: solving the ZFS boot problem

The problem which remains is that FreeBSD will not be able to
boot directly into ZFS since the bootloader doesn't know anything
about ZFS. So a little trick is that we put the kernel onto /boot
which lives on the UFS partition. This gets the system running to
the point where ZFS can be mounted and the rest of the boot
proceeds.

Remove the /tank/boot just copied over from the UFS system:

```
# rm -rf /tank/boot
```

Now make the directory in which our UFS partition will be
mounted. This will be useful later on when we want to update the
contents of that slice from a running system. We also need to
make sure that when the ZFS is booting up it can see the UFS
bootdir.

```
# mkdir /tank/bootdir
# cd /tank
# ln -s bootdir/boot boot
```

Now we tell the loader on the UFS slice to load and boot from the
contents of the ZFS volume:

```
# echo 'zfs_load="YES"' >>  /boot/loader.conf
# echo 'vfs.root.mountfrom="zfs:tank"'  >>  /boot/loader.conf
```

Edit /tank/etc/fstab so our UFS slice is mounted in the right
location for when ZFS boots.

```
/dev/ad4s1a   /bootdir          ufs      rw      1        1
```

When we reboot in a minute, we want the ZFS tank to mount in
/var /usr /tmp and/ and not within the /tank location it is now. So
set the true mountpoints:

```
# zfs set mountpoint=/tmp tank/tmp
# zfs set mountpoint=/usr tank/usr
# zfs set mountpoint=/var tank/var
```

Set root mount point to 'legacy' so ZFS won't try to mount it
automatically. It should already have been mounted by the loader:

```
# zfs set mountpoint=legacy tank
```

All Done!

Reboot and login as root. df -h and zfs list and you will see everything.

```
# df -h
Filesystem          Size     Used    Avail Capacity   Mounted o
tank                282G     454M     281G      0%     /
devfs               1.0K     1.0K       0B    100%     /dev
/dev/ad4s1a         496M     317M     139M     69%     /bootdir
tank/tmp            281G     1.1M     281G      0%     /tmp
tank/usr            287G     5.7G     281G      2%     /usr
tank/var            281G      76M     281G      0%     /var

# zfs list
NAME        USED   AVAIL   REFER   MOUNTPOINT
tank        6.27G  281G     454M    legacy
tank/tmp    1.10M  281G    1.10M    /tmp
tank/usr    5.75G  281G    5.75G    /usr
tank/var    75.5M  281G    75.5M    /var
```

All is done. :) Basic install completed. At this point you might want to install a more complete version of FreeBSD, install the ports system or anything else you normally do.

Creating a Mirror

That was pretty exciting.

Now let's say that we have some important data on that drive and we are pretty keen not to lose it. So, let's add another drive as a mirror under ZFS. In some ways this is even superior to a hardware RAID setup since ZFS is able to monitor the checksums on the disk and automatically detect which of the two drives has corrupted a block of data, transparently using the other and repairing if necessary.

- Install the HDD and boot up on the system.
- Login as root
- Run sysinstall
- Enter Custom Install
- Partition fdisk with Auto option (one slice, who disk) and press 'w' to write changes
- Label with options

```
A: 512Mb   /
B: swap
D: rest of disk empty (for zfs later)
```

and once again, press 'w' to save changes and state 'YES' to the question it asks you about applying now. You'll need to make this identical to the labels on the first drive.
* Exit sysinstall.

Check out what we have:

```
#zpool status
```

Add the 2nd drive to our mirror (our second drive comes up as ad6)

```
# zpool attach tank ad4s1d ad6s1d
# zpool status
```

and it will now show the two HDD's in a mirror

```
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            ad4s1d  ONLINE       0     0     0
            ad6s1d  ONLINE       0     0     0

errors: No known data errors
```

Now we have a mirror, but if we lose the first drive we will need the contents of the UFS slice copied to the second drive in order to reboot.

```
# newfs /dev/ad6s1a
# mkdir /mnt/bootdir
# mount /dev/ad6s1a /mnt/bootdir
# find -x /bootdir | cpio -pmd /mnt
```

This gives us another ready to use system on the 2nd HDD in case we have a failure on the 1st HDD with all the necessary zfs tools to move forward.

Disaster recovery

With the way we have setup our system if the ZFS fails to boot or function for whatever reason we can still boot off the original minimal install at any time. To do this do the following:

- Boot up
- Choose #6 (Escape to Loader Prompt) at the boot menu.
- Suppress the mounting of the ZFS volumes like this:

```
# unset vfs.root.mountfrom
# boot
```

- Login

```
# df -h
```

```
# zpool list
```

You will notice that it has not picked up the ZFS drives or the pool. Never fear we can import it!

```
#zpool import -d /boot/zfs
```

This loads the zpool data so the libraries can find the information it needs, now we need to import our pool using an alternate root.

```
#zpool import -f -R /tank tank
```

And check out that it has worked.

```
# df -h
# zpool list
```

Useful Commands

Create a mirror

```
#zpool create -m /usr/local/www/ mirror <mirror name> ad4s1g ad
```

Show available pool/resources

```
#zfs list
```

Show Status of Disk/pool

```
#zfs status
```

Changing Mount points for created resources

```
#zfs set mountpoint=/<mountpoint> <pool resources>
```

Data Limiting

```
# zfs set quota=10g testpool/testfs/dir1
```

(logically limits space)

```
# zfs set reservation=20g testpool/testfs/dir2
```

(logically preallocates space)

Links and References

FreeBSD ZFS progress
FreeBSD quickstart guide
FreeBSD ZFS on root

Solutions

- Case Studies
- Examples
- Testimonials
- Articles
    - FreeBSD on ZFS
    - Sendmail and OS X