Erledigt ZFS mit OpenZFS on OSX

Beitrag von „dietanu“ vom 29. Mai 2016, 13:46

ZFS. Das ist ein Thema, mit dem ich mich persönlich schon seit Jahren auseinander setze. Angefangen mit ZFS auf OpenSolaris, FreeBSD und dann später auf Nexenta, ZOL (ZFS On Linux).

Viele hier haben scheinbar schon Erfahrung mit ZFS gesammelt und so brauche ich nicht ganz so weit ausholen. Für die anderen hier nochmal die harten Fakten kurz gefasst:

- ZFS ist sicherer als andere Dateisysteme, weil es mit "Copy on write" arbeitet. Sprich: Die Chance sich beim einem Stromausfall die Daten zu versauen ist mit ZFS deutlich geringer, als bei einem Dateisystem wie HFS+, NTFS oder ext4.
- ZFS baut Prüfsummen aus den Dateien und begegnet so dem Phänomen namens "Bit Rot". Kippende Bits werden erkannt, die bei dann defekten Fotos z.B. so aussehen:

Das kennt sicher der eine oder andere. Zum Beispiel von alten CD-Rs, die man vor langer Zeit gebrannt hat und wo die organische Schicht beschädigt ist. Dies kann auf Festplatten ebenso passieren. Grund dafür sind Fehler in den Datenströmen, verursacht durch (bitte jetzt nicht Lachen) u.a. Sonnenstürme, Gewitter, aber auch durchaus durch schlechte SATA-Kabel. Das ist im Grunde für mich das Hauptargument FÜR ZFS. Ich möchte nicht, dass meine Fotos beschädigt werden, und neben dem Einsatz von solchen System hilft aus meiner Sicht (und das von mir praktizierte) Backup auf M-DISCs.

- ZFS enhält RAID-ähnliche Verbundmechanismen, die ähnlich wie RAID funktionieren. Man unterscheidet hier:
- ZFS mit mirror: Entspricht RAID1
- ZFS mit striping: Entspricht RAID0 (das ist meistens eine doofe Idee, ich weiß nicht genau wie das unter ZFS gemacht wird)

- ZFS mit RAIDz1: Entspricht so ungefähr RAID5, eine Platte darf ausfallen, ohne das Datenverlust auftritt (Paritätsverteilung).
- ZFS mit RAIDz2: Entspricht so ungefähr RAID6, zwei Platten dürfen ausfallen, ohne das Datenverlust auftritt (Paritätsverteilung).
- ZFS mit RAIDz3: Ist im Grunde RAID6+1 bzw. RAID5+2, also dürfen drei Platten ausfallen.

- ZFS kann Deduplizierung:
- Eine tolle Sache, man stelle sich vor, man hat einen Ordner mit Fotos, einmal die Originalen und einmal die bearbeiteten RAWs, diese liegen doppelt auf der Platte. Statt hier z.B. 10 GiB zu verbrauchen (2x 5 GiB), werden nur 5 GiB verbraucht, weil das ZFS Links erstellt. Dieses Feature ist allerdings für einen Desktop leider etwas zu "teuer", weil es sehr viel CPU-Power und noch mehr RAM frisst.

- ZFS kann Snapshots:
- Vielleicht von LVM oder virtuellen Maschinen bekannt, kann man mit ZFS Snapshots erstellen. Diese frieren quasi die Dateien in dem "Pool" ein und merken sich den Zustand. Man kann diese also wiederherstellen. Auch dieses tolle Feature ersetzt natürlich UNTER GAR KEINEN UMSTÄNDEN ein Backup

- ZFS ist für Server entwickelt worden:
- Ursprünglich wurde ZFS von der Firma Sun Microsystems für den Einsatz in Solaris entwickelt. Leider ist Sun pleite gegangen und wurde von (uhh) ORACLE gekauft, die nur noch halbherzig an ZFS rumentwicklen. Solaris gibt's mittlerweile zwar von ORACLE, aber OpenSolaris wurde z.B. eingestellt. ZFS wurde unter der CDDL-Lizenz (BSD) zur Verfügung gestellt, was erlaubte, das Dateisystem auch auf andere Betriebssysteme zu portieren. Oben hatte ich bereits FreeBSD als eines der ersten nicht-Solaris-basierten Betriebssysteme erwähnt. Illumos (ein Fork des OpenSolaris Kernels) findet dagegen ein neues Heim in Projekten wie OpenIndiana. Auf Linux (mein Primärsystem auf Servern), ist die Sache danke GNU nicht ganz so einfach. ZFS ist nicht mit der GPL kompatibel und DARF somit nicht direkt im Kernel angebunden - sprich: ausgeliefert werden. Ziemlich Banane irgendwie, aber die andere Seite der Medallie, wenn man Open Source Software verwenden möchte. Canonical setzt sich gerade über dieses Verbot hinweg (Ubuntu kann ab 16.04 nativ mit ZFS umgehen). Für alle anderen gibt es das ZOL (ZFS on Linux) Projekt, was recht einfach einzubinden ist. Via DKMS ist die Implemtation auch bei Kernel-Updates RELATIV sicher einzusetzen.

Ihr merkt, viel von Desktop ist hier nicht zu lesen. Das liegt vor allem an einem: ZFS besteht quasi auf ECC Speicher. Ich habe viele Videos von Leuten gesehen (wie Paul), die FreeNAS (ein

NAS-System auf Basis von FreeBSD und damit kommt es mit ZFS), die Non-ECC Speicher trotzt oder gerade eben mit ZFS einsetzen. Ich persönlich halte das für eine nicht so brilliante Idee, zumindest, wenn es sich um den primären Speicher handelt.

Die meisten von Euch werden, so wie ich, vermutlich im PC/Hackintosh KEINEN ECC-Speicher haben (es sei denn Ihr habt eine Workstation auf Xeon Basis und MÜSST evtl. sogar ECC Speicher verwenden). Achso - ECC, das ist "normaler" RAM, der allerdings eine Fehlerkorrektur enthält. Man findet diesen häufig in Servern. In einem NAS würde ich auch immer auch ECC Speicher setzen. Mein neues NAS hat z.B. 24 GiB ECC RAM DDR3 bekommen. Allerdings müssen CPU und Mainboard dies unterstützen.

Es ist eine Glaubsfrage, ob man ZFS auf einem Desktop einsetzt oder nicht. Selbst Sun Mitarbeiter haben früher ZFS auf Non-ECC Laptops eingesetzt, zu Demozwecken. Es lässt sich vortrefflich drüber streiten.

In meinem Fall möchte ich gerne einen RAID5 in meinem Hackintosh haben. Der Grund ist relativ banal: Ich möchte eine zweite Backup-Ebene haben. Mein primäres Backup erfolgt auf zwei USB 3.0 3 TiB Festplatten am Server, die dann rotierend angeschlossen und "offsite" geschafft werden. Da passt aber nicht alles drauf, aber das wichtigste wie Fotos und Dokumente.

Also schaute ich mich um, wie ich einen RAID5 im Hackintosh realisieren kann. Erstmal: Gar nicht. RAID5 für einen Hackintosh per Software kann ich vergessen. Wenn, dann müsste ich mir einen Hardware-RAID-Controller kaufen ODER die Software SoftRAID (für über 180$). Nee. Dann stolperte ich über einen Artikel, der von OpenZFS on OSX schrieben.

Hier nun der Installationsprozess, der denkbar einfach ist.

Ladet Euch die aktuelle Version von OpenZFS for OSX hier herunter: https://openzfsonosx.org/wiki/Downloads

Nach der Installation steht Euch ZFS bereits zur Verfügung. Loggt Euch ins Terminal ein und tippt:

Code

```
1. $ sudo zpool status
```

Bekommt Ihr hier keine Fehlermeldung, sondern nur die Information, dass noch kein Pool angelegt wurde, ist alles sauber installiert.

Für Backups möchte ich das von mir frisch entdeckte Borg Backup nicht verzichtenm deswegen installiere ich es direkt.

Für Borg Backup braucht Ihr Homebrew. Der Anleitung auf der Seite ist zu folgen. Eine tolle Sache übrigens, Ihr müsste allerdings am besten Xcode installieren, um auf alle Developer Module die benötigt werden, zugreifen zu können.

Ich habe Homebrew immer installiert, weil ich auf Linux/UNIX Tools wie htop, dfc, screenfetch  oder aktuelle Versionen von vim nicht verzichten möchte.

Borg Backup installiere ich per:

Code

```
1. $ brew cask install borgbackup
```

Weitere Programme z.B. per:

Code

```
1. $ brew install tmux vim dfc htop screenfetch
```

Zurück zu ZFS. In meinem Rechner habe ich 5 Platten aus meinem früheren Medienserver verbaut, die teilweise schon eher älteren Datums sind, aber zum Wegschmeißen sind sie zu schade und als Zweitbackup durchaus noch zu gebrauchten. Hier ein Foto von der Rückseite meines Hackintoshs:

Ich weiß, dass ich die Platten ruhigen Gewissens überschreiben kann. SEIT EUCH SICHER, WAS NUN KOMMT, LÖSCHT ALLE DATEN AUF DIESEN PLATTEN! Auch solltet Ihr GENAU drauf achten, wie die Platte heißt!

Mit:

Code

1. Shiro-Usagi:~ root# diskutil list
2. /dev/disk0 (internal, physical):
3. #: TYPE NAME SIZE IDENTIFIER
4. 0: GUID_partition_scheme *240.1 GB disk0
5. 1: EFI EFI 209.7 MB disk0s1
6. 2: Apple_HFS Mac SSD 239.1 GB disk0s2
7. 3: Apple_Boot Recovery HD 650.0 MB disk0s3
8. /dev/disk1 (internal, physical):
9. #: TYPE NAME SIZE IDENTIFIER
10. 0: GUID_partition_scheme *1.5 TB disk1
11. 1: ZFS 1.5 TB disk1s1
12. 2: 6A945A3B-1DD2-11B2-99A6-080020736631 8.4 MB disk1s9
13. /dev/disk2 (internal, physical):
14. #: TYPE NAME SIZE IDENTIFIER

15. 0: GUID_partition_scheme *1.5 TB disk2
16. 1: ZFS 1.5 TB disk2s1
17. 2: 6A945A3B-1DD2-11B2-99A6-080020736631 8.4 MB disk2s9
18. /dev/disk3 (internal, physical):
19. #: TYPE NAME SIZE IDENTIFIER
20. 0: GUID_partition_scheme *1.5 TB disk3
21. 1: ZFS 1.5 TB disk3s1
22. 2: 6A945A3B-1DD2-11B2-99A6-080020736631 8.4 MB disk3s9
23. /dev/disk4 (internal, physical):
24. #: TYPE NAME SIZE IDENTIFIER
25. 0: GUID_partition_scheme *2.0 TB disk4
26. 1: ZFS 2.0 TB disk4s1
27. 2: 6A945A3B-1DD2-11B2-99A6-080020736631 8.4 MB disk4s9
28. /dev/disk5 (internal, physical):
29. #: TYPE NAME SIZE IDENTIFIER
30. 0: GUID_partition_scheme *1.5 TB disk5
31. 1: ZFS 1.5 TB disk5s1
32. 2: 6A945A3B-1DD2-11B2-99A6-080020736631 8.4 MB disk5s9
33. /dev/disk6 (external, physical):
34. #: TYPE NAME SIZE IDENTIFIER
35. 0: FDisk_partition_scheme *31.9 GB disk6
36. 1: Windows_FAT_32 NO NAME 31.9 GB disk6s1

Alles anzeigen

Erhalte ich eine Übersicht über meine Festplatten. Anhand der Größe identifiziere ich diese. Interessant sind in meinem Falle die Platten /dev/disk1,2,3,4 und 5. 0 ist die System-SSD und 6 ist eine SD-Card.

Ich bin mir GANZ SICHER, dass ich die richtigen Platten gewählt habe und starte den Bau des ZFS RAIDz1 (Ihr erinnert Euch? Das ist das RAID5-Equivalent):

Code

1. # zpool create -f pool raidz /dev/disk1 /dev/disk2 /dev/disk3 /dev/disk4 /dev/disk5

Beachtet, dass Ihr hier als root arbeiten MÜSST oder ein "sudo" davorhängt.

Der "pool" (so der Name) wird mit "-f", also mit "force" Permission erstellt. Das kann nötig sein, wenn die Platten vorher z.B. in einem Windows System waren und GPT nicht sauber geschrieben wurde etc.

Das Erstellen des Pools dauert wirklich nicht lange, wenige Sekunden (ein großer Unterschied zu anderen Software, aber auch Hardware-RAIDs). Zum Vergleich: Mein NAS hat einen "mdadm" basiertes RAID5, das "Initialisieren" hat gut und gerne 8h gedauert. Das liegt einfach daran, dass ein RAID mit Blöcken umgeht, ein ZFS allerdings arbeitet auf Dateiebene.

Wenn der Pool nun erstellt wurde, kann man sich diesen mit dem Befehl **zpool status** anschauen:

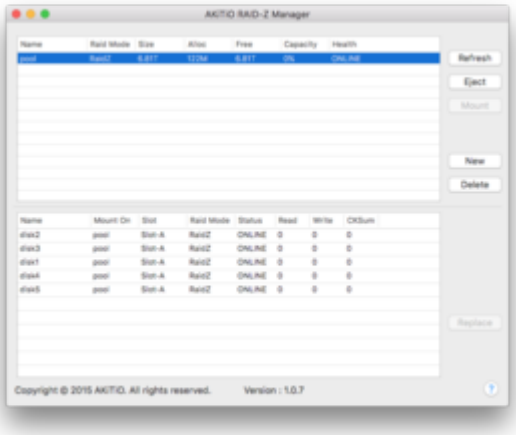
Code

1. Shiro-Usagi:~ root# zpool status
2. pool: pool
3. state: ONLINE
4. scan: none requested
5. config:
6.
7.
8. NAME STATE READ WRITE CKSUM
9. pool ONLINE 0 0 0
10. raidz1-0 ONLINE 0 0 0
11. media-9FA01B58-9828-7546-8EDB-29B60F2FD701 ONLINE 0 0 0
12. media-9F4C8B6A-766E-6845-A425-1A58FE9091BB ONLINE 0 0 0
13. media-D549541D-6EE6-5C48-AE7E-94CBE666CD53 ONLINE 0 0 0
14. media-465A7E0D-4B15-C342-A63B-F9DF63B971F1 ONLINE 0 0 0
15. media-E415B577-3489-1A49-B429-54FA3E7A4759 ONLINE 0 0 0
16.
17.
18. errors: No known data errors

Alles anzeigen

Hier ist nun alles in Ordnung

(wer es lieber grafisch mag, der kann auch zum kostenfreien Akitio RAID-Z Manager greifen):



Zu guter Letzt sind die Dateirechte wichtig. Sehr wichtig, denn aktuell seht Ihr zwar das Volume im Finder, aber der Zugriff ist nur für "root" (also den Admin-User) gestattet.

Lasst und das ändern:

Code

1. # chmod -R 744 /Volumes/pool/
2. # chmod -R -N /Volumes/pool/
3. # chown -R [USER]:staff /Volumes/pool/

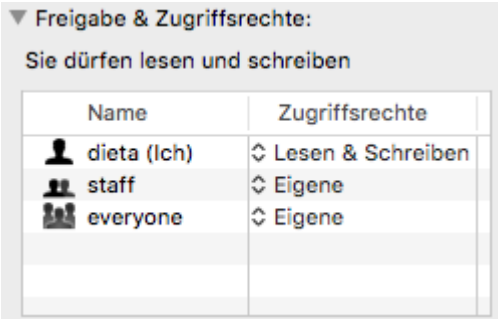
Nun sollte auch per Finder das Lesen UND Schreiben möglich sein.

Per Terminal könnt Ihr dies abfragen:

Code

1. $ ls -lisa /Volumes/
2. 2 4 drwxr--r--@ 6 [USER] staff 6 29 Mai 12:41 pool

Oder auch per CMD+I im Finder auf das Laufwerk:



Ich wünsche Euch viel Spaß mit ZFS, sofern Ihr es denn einsetzen möchtet.

Bitte bedenkt beim Einsatz von ZFS auf Non-ECC Systemen, dass dies nicht ideal ist. Das ist Euer Risiko. Die ganze Anleitung/Tipp hier sind aus meiner privaten Anwendung heraus entstanden und NATÜRLICH ist es EURE Sache, was Ihr mit Euren Daten macht. Ohne Backup solltet Ihr über so etwas nicht einmal nachdenken. Ich übernehme keine Garantie, dass Ihr Euch mit ZFS Daten abschießt.

Leider musste ich diesen letzten Absatz auf Grund eines anderen Threads einfach drunterschreiben

---

Beitrag von „al6042“ vom 29. Mai 2016, 14:05

Wow,

vielen Dank für den tollen Beitrag...

Ich denke ich sollte meine 3x 3TB HDDs mal leeren und damit neu zusammenbauen...

Beitrag von „dietanu“ vom 29. Mai 2016, 14:10

> Zitat von al6042
>
> Wow,
>
> vielen Dank für den tollen Beitrag...
> <img src="https://www.hackintosh-forum.de/wcf/images/smilies/neu/danke-2.gif" alt="
> 
>
> " />
>
> Ich denke ich sollte meine 3x 3TB HDDs mal leeren und damit neu zusammenbauen...

Aber bitte vorher ein sicheres Backup machen