Cicero LawPack en ZFS

Waarom?

In de eerste plaats omdat ZFS het enige filesysteem is dat gegarandeerd bescherming kan bieden tegen stille data corruptie. ZFS detecteert en repareert corruptie, is sneller in het vinden van defecte disks dan S.M.A.R.T. en is het enige systeem dat bescherming biedt tegen bit rot.

ZFS snapshots beschermen tegen ongewenste wijzigingen, tonen vorige versies van documenten en zorgen voor veilige upgrades van Cicero.

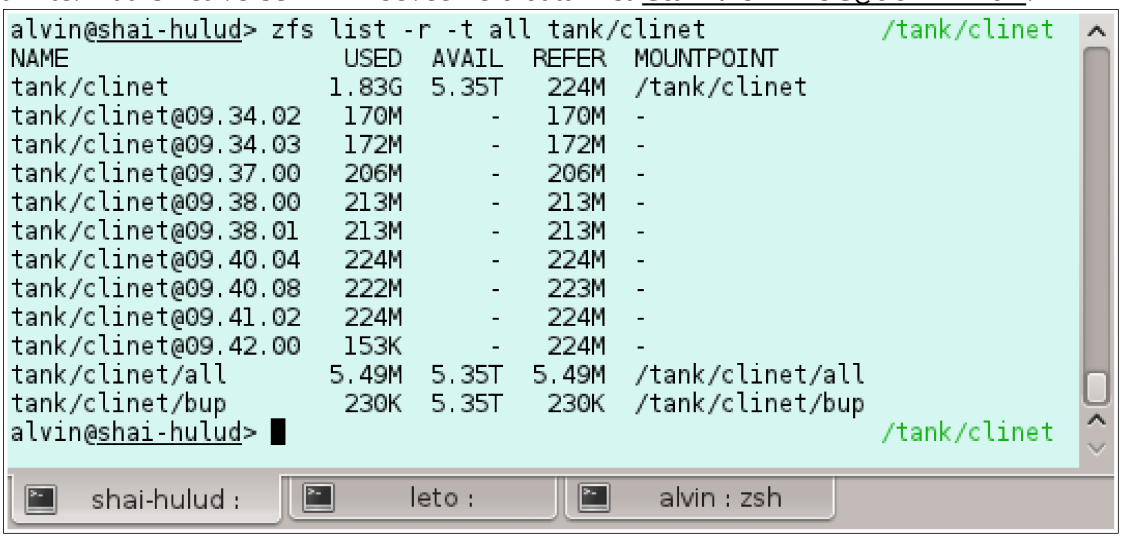
De ZFS `send` en `receive` commando's kunnen gebruikt worden om snapshots op te slaan als files of om remote replicatie te doen. Synchroniseren van Cicero data duurt doorgaans erg lang. ZFS send/receive hoeft niet alle files te overlopen en kan dit proces zo herleiden tot enkele seconden.

Daarnaast is het een erg flexibel storage systeem en is het gewoon leuk om te kunnen zeggen dat je data op een 128-bit filesysteem staat. Zoek de voordelen maar op!

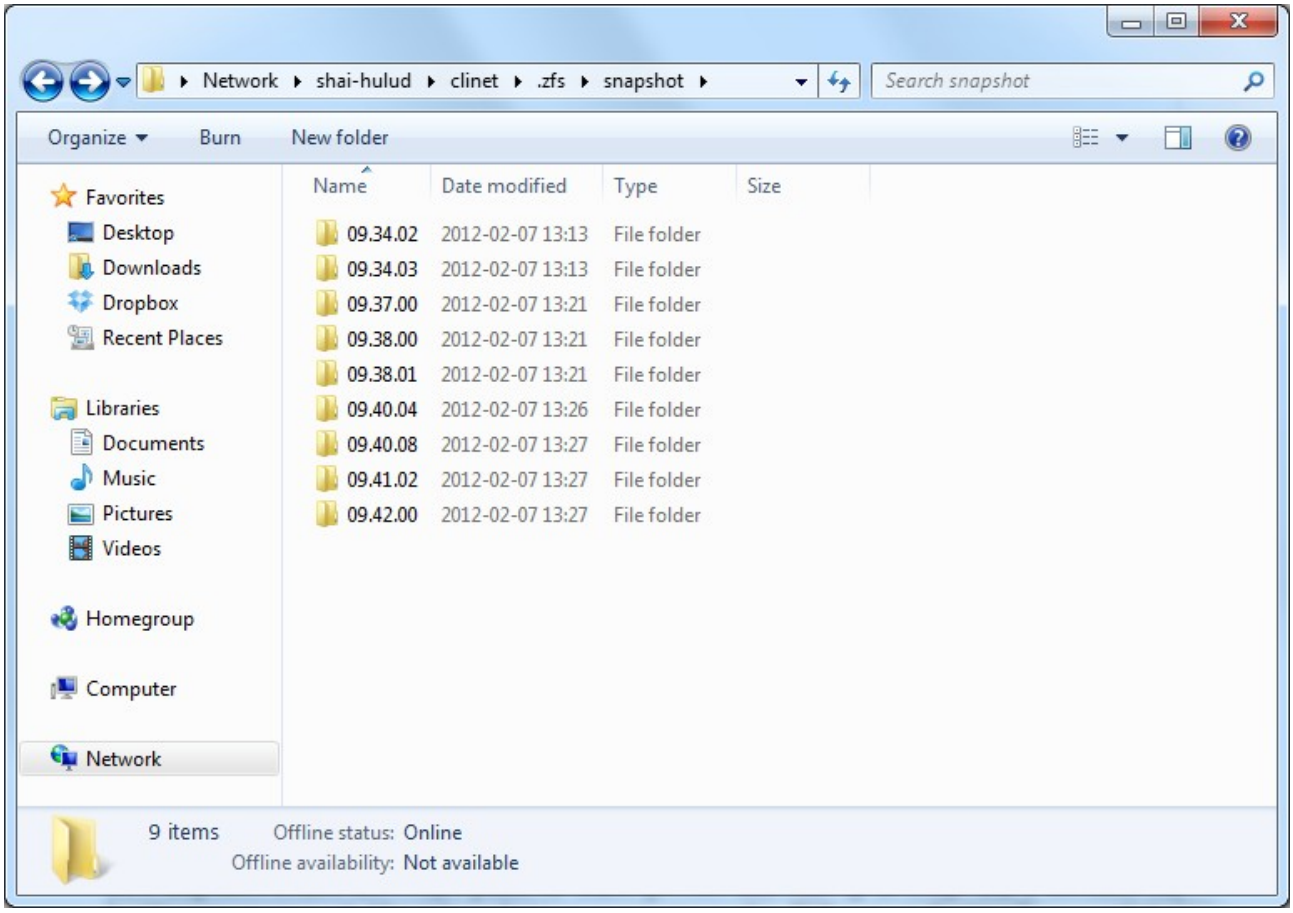
Gebruik van snapshots

Neem als voorbeeld een doorsnee upgrade. Voor de upgrade wordt een back-up van de database genomen (in `/bup`) en een snapshot van de clinet share. Indien iets misloopt tijdens of na de upgrade kan er met 1 commando een restore van de data worden uitgevoerd. Tijdens dit proces blijft data in `/bup` en `/all` intact.

Dagelijks, of zelfs elk uur, kan een snapshot genomen worden van de data in `/all`. Gebruikers kunnen voorgaande versies van hun documenten raadplegen of kopiëren vanuit `all/.zfs/snapshot/`. ZFS snapshots nemen nauwelijks plaats in. Enkel gewiste data staat op de schijf. Dat is allemaal een beetje wennen voor wie klassieke volume managers gewend is. Kijk naar onderstaande screenshot. De snapshot `tank/clinet@09.42.00` gebruikt 224M schijfruimte. Dat is het verschil in hoeveelheid data met `tank/clinet@09.41.02`.



`zfs diff` kan vertellen welke bestanden gewijzigd zijn, welke verdwenen of aangemaakt zijn en welke hernoemd werden.
De gebruiker benadert snapshots via directories. De inhoud van vorige versies is eenvoudig beschikbaar via de verkenner.



Aanmaken van de filesystemen

Maak een filesysteem om als share te gebruiken. In dit voorbeeld heet het filesysteem *clinet* en heet de zpool *tank*. Ik ga er van uit dat er al een *zpool* gemaakt werd. Dit artikel geeft slechts voorbeelden. Voor het echte werk is het lezen van de ZFS Administration Guide een must. Omdat Oracle nogal graag de links in hun documentatie breekt is het hoofdstuk uit het FreeBSD handboek misschien beter geschikt.

Verschillende filesystemen worden gecreëerd op basis van verschillende eigenschappen.
tank/clinet: Inhoud wijzigt enkel bij het veranderen van versie.
tank/clinet/all: Inhoud wijzigt, maar is niet afhankelijk van wijzigingen in versie.
tank/clinet/bup: Inhoud wijzigt, maar is niet afhankelijk van wijzigingen in versie.

```
# zfs create -o compression=on -o atime=off -o exec=off -o setuid=off tank/clinet
# zfs create -o compression=gzip-9 tank/clinet/bup
# zfs create -o copies=2 tank/clinet/all
```

Opties

compression=on: Wordt overal gebruikt. clinet/all laat zich doorgaans erg goed comprimeren en database back-up scripts dienen zelf geen compressie te voorzien voor database dumps in `clinet/bup`. Dat is vooral handig omdat MS SQL Server Express zelf geen compressie toe laat. Het verschil tussen on (LZJB) en gzip-9 is dat LZJB sneller is, en gzip-9 soms betere compressie kan bieden tegenover een zwaardere belasting.

atime=off: Atimes uitzetten zorgt ervoor dat er niet geschreven wordt naar disk wanneer een file gelezen wordt. Cicero heeft ze niet nodig en dit levert een hoop snelheidswinst op.

exec= off: Omdat het niet nodig is executables uit te voeren op clinet. (.exe files zullen wel nog steeds starten indien geopend via Samba, dus echt invloed heeft dit niet.)

setuid=off: De setuid bit is niet bruikbaar. Executables kunnen niet uitgevoerd worden als root. Net zoals exec is het hier vooral het principe dat telt.

copies=2: De standaard waarde is 1, maar 3 is eveneens mogelijk. Als de data erg belangrijk is kan dit nog extra redundancy geven, bovenop RAIDZ, maar zelfs op een enkele disk (wat niet aan te raden is). Naar gelang de waarde van *copies* zal ZFS kopieën van de data verspreiden over de schijven en/of vdevs. Zie http://blogs.oracle.com/relling/entry/zfs_copies_and_data_protection. Deze instelling gebruikt uiteraard 2x of 3x meer schijfruimte.

Verder is er nog de mogelijkheid tot block-level deduplicatie. Ik heb het zelf nog niet geprobeerd, maar vermoed dat het voor Cicero niet erg veel ruimte zal besparen. De **dedup** optie vereist daarenboven erg veel RAM geheugen.

Sharing

Ook file sharing (**sharesmb**) is ingebouwd in ZFS, maar enkel bij gebruik van Solaris of afgeleiden van OpenSolaris, zoals Illumos. Op FreeBSD kan er probleemloos gebruik gemaakt worden van Samba.

Systeemvereisten

Een multicore CPU en zoveel RAM geheugen als mogelijk. De exacte hoeveelheid hangt af van opties zoals *dedup* en gebruik van de server voor andere zaken. ZFS storage pools schalen goed en kunnen gebruik maken van SSD cache.