ulm university universität
uulm

ZFS - Verschlüsselung und andere Neuigkeiten

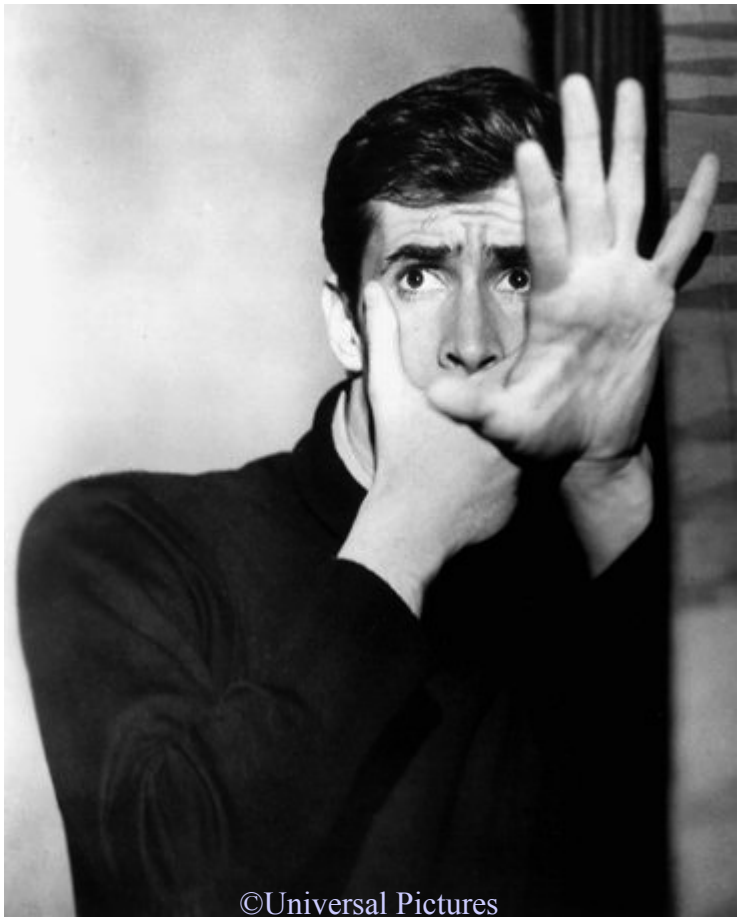
Thomas Nau, kiz (thomas.nau@uni-ulm.de)

Kommunikations- und Informationszentrum (kiz)

- Die Aufgaben der "Abteilung Infrastruktur" umfassen
  - Cluster basierende universitätsweite Mail-, LDAP-, Portal-, Datenbank- und File-Services
  - Betreuung von ca. 600 Desktop und Laptop Arbeitsplätzen
    - 25% Linux, 75% Windows
  - Backup Service für die Universitäten Konstanz und Ulm
  - HPC Cluster für die Universitäten Konstanz, Stuttgart und Ulm
  - 4 lokale Netzwerke plus flächendeckendes Campus WLAN und MAN im Ulmer Stadtbereich
  - Telefonanlage mit ca. 14.000 Anschlüssen unter Einsatz von VoIP und 2-Draht Technik
  - Azubi Ausbildung

Vorspann

Warnung, wir sind manchmal paranoid ...

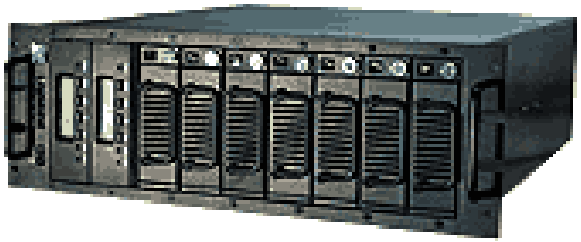


... wenn es darum geht ...

- nichts von Fehlern zu wissen die in den Daten bzw. dem Filesystem bereits vorhanden sind
- keinen Schutz gegen "fat-fingers" zu haben
- stundenlang *fsck(1M)* in Filesystemen laufen zu lassen, ohne zu wissen wie lange es wirklich dauert
- den File-, Mail-, Datenbank-, ... Server vom Backup zu restaurieren
  - dabei im Durchschnitt die letzten 12 Stunden zu verlieren
  - nicht zu wissen welche Mails bzw. Dateien betroffen sind

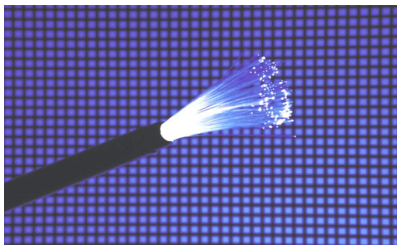
OK, alles nur eine Frage des Vertrauens ...

in Platten Caches
deren Firmware
„winzige“ Prüfsummen



RAID Controller
mit mehr Firmware
Caches und Batterien
interner Verkabelung

Betriebssysteme
noch mehr Firmware
mehr Kabel, Stecker usw.
jede Menge Mikroelektronik
"Anwendung"



Glasfasern, Kabel und Stecker

Was schief gehen kann wird schief gehen!

Anfang 2006 (pre ZFS) … PANIC

- System panic nach “freeing free inode”;
  „UFS logging“ hilft in diesem Falle nichts
- der notwendige Filesystem check *fsck(1m)* dauerte >10 Stunden ohne die Möglichkeit auf die Daten zuzugreifen; *fsck(1m)* erfolgreich beendet, Problem gelöst?
- kurz nach Aktivierung des mail Systems kommt es erneut zur "panic"
- nach einem weiteren, ebenfalls nicht erfolgreichen, *fsck(1m*) Versuch wurden alle Daten auf ZFS kopiert; zum Glück konnte lesend auf die Daten zugegriffen werden
- *rsync()* dauerte ca. 40 Stunden und ist nach dieser Erfahrung nicht das Tool der Wahl für solche Szenarien

Analyse und offene Fragen

- wir nehmen an, dass der Auslöser des Problems in einem ca. 4 Wochen zurückliegenden teilweisen Stromausfall und dem damit verbundenen Ausfall von FC-AL Ports gefolgt von einem RAID-System Absturz mit Cache-Verlust zu suchen ist
- eine der ungeklärten Fragen
  - kann *fsck(1m)* im Zusammenspiel mit dem „round-robin“ Zugriff des Solaris Volume Managers (oder aller anderen) Probleme überhaupt zuverlässig erkennen und beheben?

Mission Critical: wie früh erkenne ich Fehler?

- RAID Systeme decken nicht den kompletten Pfad der Daten ab, etwa keine FC-HBAs, …
- Volume Manager können im Allgemeinen nicht entscheiden welcher Spiegel korrekte Daten enthält
- Daten sind unterschiedlich wichtig
  - etwa Metadaten vs. Bilddaten oder MP3, letztere sind "by design" Verlust behaftet

Die derzeit einzige echte Lösung: ZFS

- erkennt Fehler anhand "starker" Prüfsummen und korrigiert diese, sofern mit Redundanz konfiguriert
- integriert den Volume Manager
  - kein "round-robin" Problem
- hat Wissen über die Relevanz von Daten
  - Pool Metadaten vs. "normale" Metadaten vs. Daten
- senkt den Stresspegel der Administratoren

ZFS Grundlagen aus 10.000m Höhe

ZFS Sicherheitsmechanismen

- ZFS verwendet starke Prüfsummen (fletcher4, sha256) für alle Daten, nicht nur für Metadaten
  - diese werden nicht zusammen mit den Daten sondern im „Pointer“ Bereich abgelegt
  - damit ist die Entscheidung welcher Teil des Spiegels die gültigen Daten enthält einfach zu treffen; alle anderen können ggf. korrigiert werden
- „zpool scrub“ liest alle Daten, berechnet die Prüfsummen neu und korrigiert ggf. fehlerhafte Komponenten
  - Prüfsummenfehler sind „normal“
  - Tipp: scrub often; neue Versionen verwenden scrub-throttle

Scrubbing eines 15TB Pools

```
kronos:.../~# zpool status -v data
  pool: data
 state: ONLINE
  scan: scrub repaired 580K in 7h26m with 0 errors
        on Sun Oct 23 23:56:46 2011
config:

  NAME          STATE     READ WRITE CKSUM
  data          ONLINE       0     0     0
    raidz2-0    ONLINE       0     0     0
      c3t1d0    ONLINE       0     0     0
      c4t1d0    ONLINE       0     0     3   212.5 repaired
      c5t1d0    ONLINE       0     0     0
...
```

ZFS Sicherheitsmechanismen

- IO wird in Transaktionen mittels „copy-on-write“ (COW) durchgeführt
  - „Snapshots“ sind eine freie Zugabe
  - damit lassen sich große Datenmengen wie etwa Mail Verzeichnisse einfach und aus Sicht der Anwendung konsistent sichern
- gültige Daten werden niemals überschrieben; kein fsck(1m)
  - ZFS garantiert nicht, dass die Daten auf die Platte geschrieben werden sondern „nur“ die Integrität (alles oder nichts)

Automatisierte Snapshots gegen fat-fingers"

- Cronjob / Daemon sorgt für regelmäßige Snapshots von ZFS Filesystemen und löscht alte Snapshots
- es existieren mehrere vordefinierte *smf(5)* Instanzen
  - frequent, hourly, daily, weekly und monthly
  - Anzahl ist konfigurierbar, bei uns 24 stündliche und 7 tägliche
- Clones sind schreibbare Kopien eines Snapshots
  - nützlich für konsistente Backups sofern die Backup-Software, etwa TSM, nicht mit Snapshots zusammenarbeitet
- erlauben Backup Paradigmenwechsel
  - unsere Backup Clients benötigen teilweise >16 Stunden um ein Filesystem zu durchsuchen

ZFS Sicherheitsmechanismen

- der integrierte Volume Manager kennt die Art der Daten, ca. 2% sind Metadaten
- “ditto blocks”
  - Kopien, zwei- oder dreifach, wichtiger Metadaten
    - neuere ZFS Versionen (OpenIndiana, Solaris 11 Express, ...) unterstützen auch Kopien von Datenblöcken (2 oder 3 fach)
    - poor mans mirror
  - werden über die Platten (vdevs) verteilt da Sektoren im Allgemeinen in größeren Gruppen lokal beieinander liegend ausfallen
- im Fehlerfall ist dadurch die Navigation im Filesystem (cd, ls, ...) meist möglich

Neuigkeiten in Solaris 11 und Update 1

Verschlüsselung

- "starke Verschlüsselung" von ZFS Filesystemen und Volumes ist Bestandteil von Solaris 11
  - AES-128-CCM ist die Voreinstellung
    - http://tools.ietf.org/html/rfc3610
    - http://en.wikipedia.org/wiki/CCM_mode
- Hardware-Beschleunigung wird mittels "Solaris crypto framework" automatisch genutzt
  - crypto engine in T4
- aktuell noch keine Verschlüsselung des root Filesystems der globalen Zone

Verschlüsselung

- setzt mindestens Pool Version #30 voraus
  - aktuelle Versionen von FreeBSD, Illumos, … bieten keine Verschlüsselung
- Bindung "für's Leben"
  - Aktivierung kann nur im Rahmen eines "`zfs create`" erfolgen
  - spätere Deaktivierung ist nicht möglich
  - der gewählte Algorithmus ist nicht änderbar
  - nachgeordnete Filesysteme, snapshots und clones, sind ebenfalls zwingend verschlüsselt

Verschlüsseltes Filesystem anlegen

```
obi-wan# zfs create -o encryption=aes-256-ccm pool/thomas
Enter passphrase for 'pool/thomas':
Must be at least 8 characters.
Enter passphrase for 'pool/thomas':
Enter again:



obi-wan# zfs get all pool/thomas
NAME          PROPERTY         VALUE                      SOURCE
...
pool/thomas   encryption       aes-256-ccm                local
pool/thomas   keychangedate    Sat Sep 15 18:03 2012      local
pool/thomas   keysource        passphrase,prompt          local
pool/thomas   keystatus        available                  -
pool/thomas   rekeydate        Sat Sep 15 18:03 2012      local
...
```

Schlüsselverwaltung

- zweistufige Schlüsselverwaltung
  - wrapping-key
    - passphrase, hex-string oder Bytefolge
  - dient dazu den eigentlichen zufällig generierten encryption-key zu schützen
    - bei einer Änderung des Schlüssels wird nur der encryption-key neu verschlüsselt, nicht der Datenbestand
  - auch der encryption-key kann mittels Kommando neu (zufällig) generiert werden
    - NIST (National Institute of Standards and Technology) Empfehlung: alle 2 Jahre
    - auch hier bleibt der alte Datenbestand unberührt

Encryption Key Verwaltung

```
obi-wan# zfs get keychangedate,rekeydate pool/thomas
NAME          PROPERTY        VALUE                    SOURCE
pool/thomas   keychangedate   Sat Sep 15 18:03 2012   local
pool/thomas   rekeydate       Sat Sep 15 18:03 2012   local


obi-wan# zfs key -K pool/thomas


obi-wan# zfs get keychangedate,rekeydate pool/thomas
NAME          PROPERTY        VALUE                    SOURCE
pool/thomas   keychangedate   Sat Sep 15 18:03 2012   local
pool/thomas   rekeydate       Sat Sep 15 18:06 2012   local
```

Wrapping Key

- können nach Belieben mit diversen Formaten und Mechanismen geändert werden
- Formate

raw	Bytefolge
hex	Hexadezimal-String
passphrase	Passwort aus dem der Schlüssel generiert wird

- Mechanismen

prompt	Abfrage bei Erzeugung oder mount
file:///filename	Schlüssel wird aus Datei (USB-Stick) gelesen
pkcs11	Schlüssel wird aus einem PKCS#11 token gelesen
https://location	Schlüssel ist auf einem Web-Server abgelegt

Wrapping Key

- **Achtung:** das Aushängen eines verschlüsselten Filesystems mit `umount(1m)` sperrt den Schlüssel nicht, ein re-mount ist ohne wrapping key also bis zum reboot möglich
- Abhilfe schafft das manuelle Entfernen aus dem Kernel

```
zfs key -u <filesystem>
```

- `pam(3PAM)` unterstützt die Verschlüsselung von Home-Directories; wrapping key stammt vom Passwort ab

  https://blogs.oracle.com/darren/entry/user_user_home_directory_encryption

Kommandozeile

```
obi-wan# zfs key -c pool/thomas
Enter new passphrase for 'pool/thomas':
Enter again:

obi-wan# zfs key -l pool/thomas
Enter passphrase for 'pool/thomas':

obi-wan# echo "MySecret" > $HOME/KEY
obi-wan# chmod 400 $HOME/KEY
obi-wan# zfs key -u pool/thomas
obi-wan# zfs set keysource=passphrase,file://$HOME/KEY \
            pool/thomas
obi-wan# zfs key -l pool/thomas
```

Immutable Zones mit Verschlüsselung

- neu in Solaris 11
- schützen root Filesystem vor Manipulation
  - strikter oder flexibler Schutz
  - letzterer erlaubt Änderungen von Dateien in /var, $HOME, …
- kann gut mit Verschlüsselung kombiniert werden

  https://blogs.oracle.com/darren/entry/immutable_zones_on_encrypted_zfs

Zusammenspiel

- der Ablauf von Komprimierung, Deduplizierung und Verschlüsselung stellt sich beim Schreiben wie folgt dar
  - Komprimierung der Daten
  - Verschlüsselung der komprimierten Daten
  - Prüfsummenbildung
  - Deduplizierung
- da der encryption key im Allgemeinen für Volumes unterschiedlich ist kann die pool-weite Deduplizierung leiden

Migration mittels "shadow migration"

- "shadow migration" transferiert lokale oder via NFS gemountete ZFS und UFS Filesysteme in neue, lokale ZFS
  - transparent für Nutzer
  - Quell-Filesystem muss read-only gemountet sein
  - Ziel-Filesystem muss leer sein
- damit lassen sich z.B. Home-Directories in verschlüsselte ZFS migrieren

"shadow migration"

```
obi-wan# echo "My_Passphrase" > /root/KEY_TN
obi-wan# zfs create -o encryption=aes-256-ccm \
            -o keysource=passphrase,file:///root/KEY_TN \
            rpool/home/tn_enc

obi-wan# ls -l /home/tn_enc
total 0


obi-wan# zfs set shadow=file:///home/tn rpool/home/tn_enc

obi-wan# ls -l /home/tn_enc
total 4
drwx------   2 nau      kizinfra       2 Sep 10 15:08 bin
drwx------   2 nau      kizinfra       2 Jul 10 08:04 doc
drwx------   2 nau      kizinfra       2 Oct 27  2002 src
drwx--x--x   2 nau      kizinfra       2 Sep 13 13:04 tmp
```

CIFS Server: ZFS für die Windows Welt

- Fileserver für SMBv1 Protokoll
  - workgroup oder Active Directory Modus
  - ID-mapping UNIX UID/GID ↔ Windows SID
  - Windows ACL aware
  - unterstützt Microsoft Shadow Copies via ZFS snapshots
  - derzeit noch kein SMBv2
  - kein Support für Microsoft Druckdienste
  - im Gegensatz zu NFS nur in globaler Zone möglich

COMSTAR: ZFS Volumes für alle

- drastische Performance Verbesserungen gegenüber Solaris 10 iSCSI Target Code
  - jetzt vollständig in Kernel integriert
- COMSTAR unterstützt auch FC oder FCoE
- unsere Citrix XenServer Infrastruktur nutzt so ZFS als Storage Pool via iSCSI und multipathing

Verschiedenes ...

- deutlich verbesserter ARC caching code in Solaris 11 dank neuem VM System
- bis zu 1MB Blocksize
- viele kleine Verbesserungen, patches, ...

Danke für's Zuhören!