Solaris ZFS - Administrationshandbuch

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Teilenr.: 820–2313–14
Oktober 2009

091006@22749

Inhalt

Vorwort

Das *Solaris ZFS-Administrationshandbuch* enthält Informationen zum Einrichten und Verwalten von Solaris™ ZFS-Dateisystemen.

Dieses Handbuch enthält Informationen für SPARC®- und x86-basierte Systeme.

Hinweis – Dieses Solaris-Release unterstützt Systeme, die SPARC- und x86-Prozessorarchitekturen verwenden: UltraSPARC®, SPARC64, AMD64, Pentium und Xeon EM64T. Die unterstützten Systeme können Sie in der *Solaris 10 Hardware-Kompatibilitätsliste* unter http://www.sun.com/bigadmin/hcl nachlesen. Eventuelle Implementierungsunterschiede zwischen den Plattformtypen sind in diesem Dokument angegeben.

In diesem Handbuch werden x86-Begriffe wie folgt verwendet:

- „x86" bezeichnet die weitere Familie an Produkten, die mit 64-Bit- und 32-Bit-x86-Architekturen kompatibel sind.
- „x64" weist auf spezifische, für 64-Bit-Systeme geltende Informationen zu AMD64- bzw. EM64T-Systemen hin.
- „32-Bit x86" weist auf spezifische, für 32-Bit-Systeme geltende Informationen zu x86-basierten Systemen hin.

Informationen zu unterstützten Systemen finden Sie in der *Solaris 10 Hardware Compatibility List*.

Zielgruppe dieses Handbuchs

Dieses Handbuch richtet sich an alle Personen, die für die Einrichtung und Verwaltung von Solaris ZFS-Dateisystemen zuständig sind. Diese Personen sollten Erfahrung in der Arbeit mit dem Betriebssystem (BS) Solaris bzw. einer anderen UNIX®-Version haben.

Aufbau dieses Handbuchs

Die folgende Tabelle erläutert die Kapitel dieses Handbuchs.

Kapitel	Beschreibung
Kapitel 1, „Das ZFS-Dateisystem (Einleitung)“	Enthält einen Überblick über ZFS und seine Funktionen bzw. Vorteile. Hier werden auch einige Grundkonzepte sowie Begriffe behandelt.
Kapitel 2, „Erste Schritte mit ZFS“	Enthält Anleitungsschritte zum Einrichten einfacher ZFS-Konfigurationen mit einfachen Pools und Dateisystemen. Dieses Kapitel enthält darüber hinaus Informationen zu Hardware und Software, die zum Erstellen von ZFS-Dateisysteme erforderlich ist.
Kapitel 3, „Unterschiede zwischen ZFS und herkömmlichen Dateisystemen“	Erläutert wichtige Funktionen, durch die sich ZFS erheblich von herkömmlichen Dateisystemen unterscheidet. Durch ein Verständnis dieser Hauptunterschiede werden Unklarheiten bei der Zusammenarbeit von herkömmlichen Tools mit ZFS ausgeräumt.
Kapitel 4, „Verwalten von ZFS-Speicher-Pools“	Enthält ausführliche Informationen zum Erstellen und Verwalten von Speicher-Pools.
Kapitel 5, „Installieren und Booten eines ZFS-Root-Dateisystems“	Stellt dar, wie ein ZFS-Dateisystem installiert und gebootet wird. Darüber hinaus wird die Migration eines UFS-Root-Dateisystems in ein ZFS-Dateisystem mithilfe von Solaris Live Upgrade behandelt.
Kapitel 6, „Verwalten von ZFS-Dateisystemen“	Enthält ausführliche Informationen zum Verwalten von ZFS-Dateisystemen. Hier werden Konzepte wie die hierarchische Dateisystemstrukturierung, Eigenschaftsvererbung, die automatische Verwaltung von Einhängepunkten sowie die Interaktion zwischen Netzwerkdateisystemen behandelt.
Kapitel 7, „Arbeiten mit ZFS-Snapshots und -Klonen“	Enthält Informationen zum Erstellen und Verwalten von ZFS-Snapshots und -Klonen.
Kapitel 8, „Schützen von ZFS-Dateien mit Zugriffssteuerungslisten“	Enthält Informationen zum Arbeiten mit Zugriffssteuerungslisten. Solche Listen schützen ZFS-Dateien, indem im vergleich zu den UNIX-Standardzugriffsrechten feiner abgestimmte Zugriffsrechte definieren und anwenden.
Kapitel 9, „Delegierte ZFS-Administration“	Erläutert, wie Benutzern ohne ausreichende Zugriffsrechte mithilfe der delegierten ZFS-Administration die Durchführung von ZFS-Administrationsaufgaben ermöglicht werden kann.
Kapitel 10, „Fortgeschrittene ZFS-Themen“	Enthält Informationen zur Verwendung von ZFS-Volumes, die Verwendung von ZFS auf Solaris-Systemen mit installierten Zonen sowie alternativen Root-Pools.
Kapitel 11, „Problembehebung und Datenwiederherstellung in ZFS“	Enthält Informationen zum Erkennen von ZFS-Fehlermodi und Wiederherstellung der Normalfunktion nach dem Auftreten von Fehlersituationen. Darüber hinaus werden hier auch Maßnahmen zum Vermeiden von Fehlfunktionen beschrieben.

Verwandte Dokumentation

Die folgenden Handbücher enthalten Informationen zu allgemeinen Themen der Solaris-Systemadministration:

- *Solaris System Administration: Basic Administration*
- *Solaris System Administration: Advanced Administration*
- *Solaris System Administration: Devices and File Systems*
- *Solaris System Administration: Security Services*
- *Solaris Volume Manager Administration Guide*

Dokumentation, Support und Schulung

Auf der Sun-Website finden Sie Informationen zu den folgenden zusätzlichen Ressourcen:

- Dokumentation (`http://www.sun.com/documentation/`)
- Support (`http://www.sun.com/support/`)
- Schulung (`http://www.sun.com/training/`)

Ihre Meinung ist gefragt

Sun möchte seine Dokumentation laufend verbessern. Ihre Kommentare und Vorschläge sind daher immer willkommen. Wenn Sie Kommentare hinterlassen möchten, gehen Sie zu `http://docs.sun.com` und klicken Sie auf "Feedback".

Typografische Konventionen

In der folgenden Tabelle sind die in diesem Handbuch verwendeten typografischen Konventionen aufgeführt.

TABELLE P–1 Typografische Konventionen

Schriftart	Bedeutung	Beispiel
`AaBbCc123`	Die Namen von Befehlen, Dateien, Verzeichnissen sowie Bildschirmausgabe.	Bearbeiten Sie Ihre `.login`-Datei. Verwenden Sie `ls -a`, um eine Liste aller Dateien zu erhalten. `system% Sie haben eine neue Nachricht.`

TABELLE P–1 Typografische Konventionen *(Fortsetzung)*

Schriftart	Bedeutung	Beispiel
AaBbCc123	Von Ihnen eingegebene Zeichen (im Gegensatz zu auf dem Bildschirm angezeigten Zeichen)	`Computername%` **`su`** `Passwort:`
aabbcc123	Platzhalter: durch einen tatsächlichen Namen oder Wert zu ersetzen	Der Befehl zum Entfernen einer Datei lautet `rm` *Dateiname* .
AaBbCc123	Buchtitel, neue Ausdrücke; hervorgehobene Begriffe	Lesen Sie hierzu Kapitel 6 im *Benutzerhandbuch*. Ein *Cache* ist eine lokal gespeicherte Kopie. Diese Datei *nicht* speichern. **Hinweis:** Einige hervorgehobene Begriffe werden online fett dargestellt.

Shell-Eingabeaufforderungen in Befehlsbeispielen

Die folgende Tabelle zeigt die Standard-Systemeingabeaufforderung von UNIX und die Superuser-Eingabeaufforderung für die C-Shell, die Bourne-Shell und die Korn-Shell.

TABELLE P–2 Shell-Eingabeaufforderungen

Shell	Eingabeaufforderung
C-Shell	`system%`
C-Shell für Superuser	`system#`
Bourne-Shell und Korn-Shell	`$`
Bourne-Shell und Korn-Shell für Superuser	`#`

◆ ◆ ◆ KAPITEL 1

Das ZFS-Dateisystem (Einleitung)

Dieses Kapitel bietet einen Überblick über das ZFS-Dateisystem, seine Funktionen und Vorteile. Darüber hinaus werden die in diesem Handbuch verwendeten Grundbegriffe erläutert.

Dieses Kapitel enthält die folgenden Abschnitte:

Neuerungen in ZFS

In diesem Abschnitt sind die neuen Leistungsmerkmale des ZFS-Dateisystems zusammengefasst.

Unterstützung für ZFS- und Flash-Installation

Solaris 10 10/09-Release: In diesem Solaris-Release haben Sie die Möglichkeit, ein JumpStart-Profil zur Identifizierung eines Flash-Archivs eines ZFS-Root-Pools einzurichten. Weitere Informationen finden Sie unter „Installieren eines ZFS-Root-Dateisystems (Flash-Archiv-Installation)“ auf Seite 127.

ZFS-Benutzer- und Gruppenkontingente

Solaris 10 10/09-Release: In früheren Solaris-Versionen konnten Kontingente und Reservierungen zur Verwaltung und Reservierung von Speicherplatz auf ZFS-Dateisysteme angewendet werden.

In diesem Solaris-Release können Sie ein Kontingent für den Speicherplatz, der von zu einem bestimmten Benutzer oder einer bestimmten Gruppe gehörigen Dateien beansprucht wird, einrichten. Das Einrichten von Benutzer- oder Gruppenkontingenten ist in einer Umgebung mit vielen Benutzern oder Gruppen sinnvoll.

Sie können Benutzer- oder Gruppenkontingente unter Verwendung der Eigenschaften `zfs userquota` und `zfs groupquota` wie folgt einrichten:

```
# zfs set userquota@user1=5G tank/data
# zfs set groupquota@staff=10G tank/staff/admins
```

Sie können die aktuelle Einstellung eines Benutzer- oder Gruppenkontingents wie folgt anzeigen:

```
# zfs get userquota@user1 tank/data
NAME       PROPERTY         VALUE            SOURCE
tank/data  userquota@user1  5G               local
# zfs get groupquota@staff tank/staff/admins
NAME                PROPERTY          VALUE              SOURCE
tank/staff/admins  groupquota@staff  10G                local
```

Stellen Sie allgemeine Informationen über ein Kontingent wie folgt ein:

```
# zfs userspace tank/data
TYPE        NAME   USED  QUOTA
POSIX User  root     3K   none
POSIX User  user1    0    5G

# zfs groupspace tank/staff/admins
TYPE         NAME   USED  QUOTA
POSIX Group  root     3K   none
POSIX Group  staff    0    10G
```

Sie können individuelle Speicherplatzbelegung von Benutzern oder Gruppen über die Eigenschaften `userused@`*user* und `groupused@` *group* wie folgt anzeigen:

```
# zfs get userused@user1 tank/staff
NAME        PROPERTY        VALUE          SOURCE
tank/staff  userused@user1  213M           local
# zfs get groupused@staff tank/staff
NAME        PROPERTY         VALUE           SOURCE
tank/staff  groupused@staff  213M            local
```

Weitere Informationen zum Einrichten von Benutzerkontingenten finden Sie unter „Einstellen von ZFS-Kontingenten und -Reservierungen“ auf Seite 208.

ZFS-Zugriffssteuerungslistenvererbungsmodus "Pass Through" zur Ausführungsberechtigung

Solaris 10 10/09-Release: In früheren Solaris-Versionen konnte die Zugriffssteuerungslistenvererbung angewendet werden, sodass alle Dateien mit den

Zugriffsrechten `0664` oder `0666` erstellt wurden. Wenn Sie die Ausführungsberechtigung aus dem Dateierstellungsmodus optional in die vererbbare Zugriffssteuerungsliste einschließen möchten, können Sie in diesem Release den Vererbungsmodus "Pass Through" zur Ausführungsberechtigung verwenden.

Ist `aclinherit=passthrough-x` auf einem ZFS-Dataset aktiviert, so können Sie die Ausführungsberechtigung für eine aus dem Tool `cc` oder `gcc` erstellte Ausgabedatei einschließen. Beinhaltet die vererbbare Zugriffssteuerungsliste die Ausführungsberechtigung nicht, so ist die Ausgabe des Compilers erst dann ausführbar, wenn Sie mit dem Befehl `chmod` die Berechtigungen der Datei ändern.

Weitere Informationen finden Sie unter Beispiel 8–12.

Verbesserungen der ZFS-Eigenschaften

Solaris 10/09-Release: Die folgenden Verbesserungen der ZFS-Dateisysteme sind in diesem Release inbegriffen.

- **Einstellung der Eigenschaften von ZFS-Dateisystemen zum Zeitpunkt der Erstellung eines Pools** – Sie können die Eigenschaften des ZFS-Dateisystems bei Erstellung eines Pools einstellen. Im folgenden Beispiel ist die Komprimierung auf dem ZFS-Dateisystem, das bei Erstellung des Pools eingerichtet wird, aktiviert.

  ```
  # zpool create -O compression=on pool mirror c0t1d0 c0t2d0
  ```

- **Einstellung von Cache-Eigenschaften auf einem ZFS-Dateisystem** – In Solaris 10/09 werden zwei neue Eigenschaften von ZFS-Dateisystemen zur Verfügung gestellt, mit denen Sie kontrollieren können, was im primären Cache (ARC) und was im sekundären Cache (L2ARC) gespeichert wird. Die Cache-Eigenschaften sind folgendermaßen eingestellt:
  - `primarycache` – Kontrolliert, was im ARC gespeichert wird.
  - `secondarycache` – Kontrolliert, was im L2ARC gespeichert wird.
  - Mögliche Werte für beide Eigenschaften – `all`, `none` und `metadata`. Ist diese Eigenschaft auf `all` gesetzt, werden sowohl Benutzerdaten als auch Metadaten im Cache gespeichert. Ist diese Eigenschaft auf `none` gesetzt, werden weder Benutzerdaten noch Metadaten im Cache gespeichert. Ist diese Eigenschaft auf `metadata` gesetzt, werden nur Metadaten im Cache gespeichert. Die Standardeinstellung ist `all`.

  Sie können diese Eigenschaften für ein bereits vorhandenes Dateisystem oder bei der Erstellung des Dateisystems einstellen. Beispiel:

  ```
  # zfs set primarycache=metadata tank/datab
  # zfs create -o primarycache=metadata tank/newdatab
  ```

  Handelt es sich um ein bereits vorhandenes Dateisystem, werden nur neue E/A-Operationen auf der Basis des Werts dieser Eigenschaften im Cache gespeichert.

Bei einigen Datenbank-Umgebungen kann es von Vorteil sein, Benutzerdaten nicht im Cache zu speichern. Sie können selbst bestimmen, ob das Einstellen von Cache-Eigenschaften für Ihre Umgebung sinnvoll ist.

- **Eigenschaften der Speicherplatzverwaltung** – Mit neuen schreibgeschützten Dateisystem-Eigenschaften können Sie die Speicherplatzbelegung für Klone, Dateisysteme und Volumes identifizieren, allerdings nicht für Snapshots. Folgende Eigenschaften stehen zur Verfügung:
  - `usedbychildren` – Gibt den Speicherplatz an, der von untergeordneten Elementen dieses Datasets beansprucht wird, und der beim Löschen dieser untergeordneten Elemente frei werden würde. Die Abkürzung für die Eigenschaft lautet `usedchild`
  - `usedbydataset` – Gibt den Speicherplatz an, der vom Dataset selbst beansprucht wird und der beim Löschen des Datasets und vorherigem Löschen aller Snapshots und Entfernen von `refreservation` frei werden würde. Die Abkürzung für die Eigenschaft lautet `usededds`
  - `usedbyrefreservation` – Gibt den von einem `refreservation`-Set auf diesem Dataset beanspruchten Speicherplatz an, der beim Entfernen von `refreservation` frei werden würde. Die Abkürzung für die Eigenschaft lautet `usedrefreserv`
  - `usedbysnapshots` – Gibt den von Snapshots dieses Datasets beanspruchten Speicherplatz an. Insbesondere geht es dabei um den Speicherplatz, der beim Löschen aller Snapshots des Datasets frei werden würde. Beachten Sie, dass es sich dabei nicht einfach um die Summe der `used` Eigenschaften der Snapshots handelt, da Speicherplatz von mehreren Snapshots geteilt werden kann. Die Abkürzung für die Eigenschaft lautet `usedsnap`

  Diese neuen Eigenschaften teilen den Wert der `used` Eigenschaft in die verschiedenen Elemente auf, die Speicherplatz beanspruchen. Genauer betrachtet wird der Wert der `used` Eigenschaft folgendermaßen aufgeteilt:

```
used property = usedbychildren + usedbydataset + usedbyrefreservation + usedbysnapshots
```

Sie können diese Eigenschaften mit dem Befehl `zfs list - o space` genauer betrachten. Beispiel:

```
$ zfs list -o space
NAME                AVAIL   USED  USEDSNAP  USEDDS  USEDREFRESERV  USEDCHILD
rpool               25.4G  7.79G         0     64K              0      7.79G
rpool/ROOT          25.4G  6.29G         0     18K              0      6.29G
rpool/ROOT/snv_98   25.4G  6.29G         0   6.29G              0          0
rpool/dump          25.4G  1.00G         0   1.00G              0          0
rpool/export        25.4G    38K         0     20K              0        18K
rpool/export/home   25.4G    18K         0     18K              0          0
rpool/swap          25.8G   512M         0    111M           401M          0
```

Der obige Befehl entspricht dem Befehl `zfs list - o name,avail,used,usedsnap,usedds,usedrefreserv,usedchild -t filesystem,volume`.

- **Snapshots auflisten**– Die Pool-Eigenschaft `listsnapshots` steuert, ob Snapshot-Informationen mit dem Befehl `zfs list` angezeigt werden. Der Standardwert ist `on`, Snapshot-Informationen werden also standardmäßig angezeigt.

  Wenn Sie die Eigenschaft `listsnapshots` deaktivieren, können Sie den Befehl `zfs list -t snapshots` verwenden, Snapshot-Informationen werden angezeigt.

Wiederherstellung von ZFS-Protokolliergeräten

Solaris 10 10/09-Release: In diesem Release identifiziert ZFS Intent-Protokoll-Fehler durch den Befehl `zpool status`. Diese Fehler werden auch von.FMA gemeldet. Ein Intent-Protokoll-Fehler kann sowohl mit ZFS als auch mit FMA behoben werden.

Wenn das System beispielsweise plötzlich herunterfährt, bevor synchrone Schreibvorgänge auf einem Pool mit separatem Protokolliergerät abgeschlossen werden können, werden Meldungen dieser Art angezeigt:

```
# zpool status -x
  pool: pool
 state: FAULTED
status: One or more of the intent logs could not be read.
        Waiting for adminstrator intervention to fix the faulted pool.
action: Either restore the affected device(s) and run 'zpool online',
        or ignore the intent log records by running 'zpool clear'.
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        pool        FAULTED      0     0     0 bad intent log
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
        logs        FAULTED      0     0     0 bad intent log
          c0t5d0    UNAVAIL      0     0     0 cannot open
```

Nach dem Fehlschlag des Protokolliergeräts können Sie den Normalbetrieb folgendermaßen wiederherstellen:

- Ersetzen Sie das Protokolliergerät oder stellen Sie es wieder her. In diesem Beispiel handelt es sich um das Gerät `c0t5d0`.
- Nehmen Sie das Protokolliergerät wieder in Betrieb.

  ```
  # zpool online pool c0t5d0
  ```

- Setzen Sie den Fehlerzustand "Protokolliergerät fehlgeschlagen" zurück.

  ```
  # zpool clear pool
  ```

Wenn Sie den Normalbetrieb wiederherstellen möchten, ohne das Protokolliergerät zu ersetzen, können Sie ihn mit dem Befehl `zpool clear` löschen. In diesem Szenario läuft der Pool in eingeschränktem Modus, und die Protokolleinträge werden in den Haupt-Pool geschrieben, bis das separate Protokolliergerät ersetzt wurde.

Um die Auswirkungen des Fehlschlags eines Protokolliergeräts abzuschwächen, können Sie gespiegelte Protokolliergeräte verwenden.

Verwenden von Cache-Geräten im ZFS-Speicher-Pool

Solaris 10 10/09-Release: In diesem Solaris-Release können Sie Pools erstellen und *Cache-Geräte* angeben, die zur Speicherung von Speicher-Pool-Daten im Cache dienen.

Cache-Speicher·bietet zwischen Hauptspeicher und Festplatte eine zusätzliche Schicht zur Datenspeicherung. Die Verwendung von Cache-Speicher bietet für die Speicherung meist statischer Daten mithilfe von wahlfreiem Zugriff die größte Leistungsverbesserung.

Beim Erstellen eines Speicher-Pools können eines oder meherer Cache-Speichergeräte angegeben werden. Beispiel:

```
# zpool create pool mirror c0t2d0 c0t4d0 cache c0t0d0
# zpool status pool
  pool: pool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        pool        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
        cache
          c0t0d0    ONLINE       0     0     0

errors: No known data errors
```

Nach dem Erstellen von Cache-Speichergeräten werden diese nach und nach mit Daten aus dem Hauptspeicher gefüllt. Je nach Größe eines definierten Cache-Speichergeräts kann es bis zu über eine Stunde lang dauern, bis es voll ist. Die Kapazität und Lesevorgänge können mithilfe des Befehls `zpool iostat` wie folgt überwacht werden:

```
# zpool iostat -v pool 5
```

Nach der Erstellung eines Pools können zu ihm Cache-Speichergeräte hinzugefügt und wieder entfernt werden.

Weitere Informationen finden Sie unter „Erstellen eines ZFS-Speicher-Pools mit Cache-Geräten“ auf Seite 70 und Beispiel 4–4.

Zonenmigration in einer ZFS-Umgebung

Solaris 10 5/09: Dieses Release bietet eine erweiterte Unterstützung für das Migrieren von Zonen in einer ZFS-Umgebung mit Live Upgrade. Weitere Informationen finden Sie unter „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 5/09 und Solaris 10 10/09)“ auf Seite 147.

Die Solaris 10 5/09-Versionshinweise enthalten eine Liste der bekannten Probleme mit diesem Release.

Unterstützung für Installation und Booten von ZFS-Root-Dateisystemen

Solaris 10 10/08: In diesem Solaris-Release können ZFS-Root-Dateisysteme installiert und gebootet werden. Das Installieren eines ZFS-Root-Dateisystems ist sowohl mit der Erstinstallationsoption als auch mit der JumpStart-Funktion möglich. Alternativ kann ein UFS-Root-Dateisystem mithilfe der Live Upgrade-Funktion in ein ZFS-Root-Dateisystem migriert werden. Außerdem steht ZFS-Unterstützung für Swap- und Dump-Geräte zur Verfügung. Weitere Informationen finden Sie in Kapitel 5, „Installieren und Booten eines ZFS-Root-Dateisystems“.

Die Solaris 10 10/08-Versionshinweise enthalten eine Liste der bekannten Probleme mit diesem Release.

Wiederherstellen eines Datasets ohne Aushängen

Release Solaris 10 10/08: Dieses Release bietet die Möglichkeit, Datasets wiederherzustellen, ohne sie zuerst auszuhängen. Dieses Leistungsmerkmal bedeutet, dass es nicht mehr notwendig ist, mit der Option `zfs rollback -f` das Aushängen zu erzwingen. Die Option `-f` wird nicht mehr Unterstützt und wird bei Angabe ignoriert.

Verbesserungen des Befehls `zfs send`

Release Solaris 10 10/08: Dieses Release umfasst die folgenden Verbesserungen des Befehls `zfs send`.

- Senden aller inkrementellen Streams von einem Snapshot zu einem kumulativen Snapshot. Beispiel:

```
# zfs list
NAME                      USED  AVAIL  REFER  MOUNTPOINT
pool                      428K  16.5G    20K  /pool
pool/fs                    71K  16.5G    21K  /pool/fs
pool/fs@snapA              16K      -  18.5K  -
pool/fs@snapB              17K      -    20K  -
pool/fs@snapC              17K      -  20.5K  -
pool/fs@snapD                0      -    21K  -
# zfs send -I pool/fs@snapA pool/fs@snapD > /snaps/fs@combo
```

Alle inkrementellen Snapshots zwischen `fs@snapA` und `fs@snapD` werden nach `fs@combo` gesendet.

- Senden eines inkrementellen Streams vom ursprünglichen Snapshot, um einen Klon zu erstellen. Der ursprüngliche Snapshot muss auf der Empfangsseite bereits vorhanden sein, damit der inkrementelle Stream angenommen werden kann. Beispiel:

```
# zfs send -I pool/fs@snap1 pool/clone@snapA > /snaps/fsclonesnap-I
.
.
# zfs receive -F pool/clone < /snaps/fsclonesnap-I
```

- Senden eines Replikationsstreams aller abhängigen Dateisysteme zu den benannten Snapshots. Nach dem Empfang werden alle Eigenschaften, Snapshots, abhängigen Dateisysteme und Klone beibehalten. Beispiel:

```
zfs send -R pool/fs@snap > snaps/fs-R
```

Ein ausführlicheres Beispiel finden Sie in Beispiel 7–1.

- Senden eines inkrementellen Replikationsstreams.

```
zfs send -R -[iI] @snapA pool/fs@snapD
```

Ein ausführlicheres Beispiel finden Sie in Beispiel 7–1.

Weitere Informationen finden Sie unter „Senden und Empfangen komplexer ZFS-Snapshot-Datenstreams“ auf Seite 229.

ZFS-Kontingente und -Reservierungen ausschließlich für Dateisystemdaten

Release Solaris 10 10/08: Zusätzlich zu den vorhandenen ZFS-Funktionen für Kontingente und Reservierungen enthält diese Version in der Speicherplatzanzeige auch Dataset-Kontingente und -Reservierungen ohne abhängige Entitäten wie z. B. Snapshots oder Klone.

- Die Eigenschaft `refquota` beschränkt den Speicherplatz, den ein Dataset belegen kann. Sie erzwingt einen absoluten Grenzwert des belegbaren Speicherplatzes. Dieser absolute Grenzwert berücksichtigt jedoch nicht den von abhängigen Entitäten wie z. Snapshots oder Klonen belegten Speicherplatz.
- Die Eigenschaft `refreservation` legt den für ein Dataset minimal garantierten Speicherplatz (ohne Speicherplatz für abhängige Entitäten) fest.

So können Sie beispielsweise in `refquota` für `studentA` einen Wert von 10 GB festlegen, der für den von diesem Benutzer *belegten* Speicherplatz einen absoluten Grenzwert von 10 GB festlegt. Zum Erreichen einer zusätzlichen Flexibilität können Sie ein 20 GB-Quotum einstellen, mit dessen Hilfe Sie die Snapshots von `studentA` verwalten können.

```
# zfs set refquota=10g tank/studentA
# zfs set quota=20g tank/studentA
```

Weitere Informationen finden Sie unter „Einstellen von ZFS-Kontingenten und -Reservierungen“ auf Seite 208.

Eigenschaften von ZFS-Speicher-Pools

Release Solaris 10 10/08: Die Eigenschaften von ZFS-Speicher-Pools wurden mit einem vorigen Release eingeführt. In diesem Release stehen zusätzliche Informationen zu den Eigenschaften zur Verfügung. Beispiel:

```
# zpool get all mpool
NAME   PROPERTY     VALUE             SOURCE
mpool  size         33.8G             -
mpool  used         5.76G             -
mpool  available    28.0G             -
mpool  capacity     17%               -
mpool  altroot      -                 default
mpool  health       ONLINE            -
mpool  guid         2689713858991441653  -
mpool  version      10                default
mpool  bootfs       mpool/ROOT/zfsBE  local
mpool  delegation   on                default
mpool  autoreplace  off               default
mpool  cachefile    -                 default
mpool  failmode     continue          local
```

Eine Beschreibung dieser Eigenschaften können Sie Tabelle 4–1 entnehmen.

- Eigenschaft `cachefile` - **Solaris 10 10/08:** Diese Version enthält die neue Eigenschaft `cachefile` die festlegt, wo Informationen zur Poolkonfiguration im Cache-Speicher abgelegt werden. Alle Pools im Cache werden beim Booten des Systems automatisch

importiert. Es kann jedoch sein, dass Installations- und Cluster-Umgebungen diese Informationen an verschiedenen Stellen im Cache-Speicher ablegen müssen, sodass Pools nicht automatisch importiert werden.

Sie können diese Eigenschaft so einstellen, dass Poolkonfigurationen an einer anderen Stelle im Cache-Speicher abgelegt werden und später mithilfe des Befehls `zpool import c` importiert werden können. Für die meisten ZFS-Konfigurationen wird diese Eigenschaft nicht verwendet.

Die Eigenschaft `cachefile` ist nicht beständig und wird nicht auf Festplatte gespeichert. Diese Eigenschaft löst die Eigenschaft `temporary` ab, die in früheren Solaris-Versionen anzeigte, dass Poolinformationen nicht im Cache gespeichert werden sollten.

- Eigenschaft `failmode` - **Release Solaris 10 10/08:** Diese Version enthält die Eigenschaft `failmode`, mit der festgelegt wird, wie sich das System im Falle eines äußerst schwerwiegenden·Poolausfalls aufgrund von Unterbrechungen in der Gerätekonnektivität bzw. dem gleichzeitigen Ausfall aller Speichergeräte im Pool verhalten soll. Die Eigenschaft `failmode` kann auf die Werte `wait`, `continue` oder `panic` gesetzt werden. Der Standardwert ist `wait`. Dies bedeutet, dass Sie das ausgefallene Gerät neu in den Pool integrieren oder auswechseln und den Fehler danach mit dem Befehl `zpool clear` löschen müssen.

  Die Eigenschaft `failmode` wird wie andere einstellbare ZFS-Eigenschaften auch gesetzt. Dies kann vor oder nach dem Erstellen eines Pools geschehen. Beispiel:

```
# zpool set failmode=continue tank
# zpool get failmode tank
NAME  PROPERTY  VALUE     SOURCE
tank  failmode  continue  local

# zpool create -o failmode=continue users mirror c0t1d0 c1t1d0
```

  Eine Beschreibung aller Eigenschaften für ZFS-Pools finden Sie in Tabelle 4–1.

Verbesserungen des ZFS-Befehlsprotokolls (`zpool history`)

Release Solaris 10 10/08: Der Befehl `zpool history` wurde um die folgenden neuen Leistungsmerkmale erweitert:

- Informationen zu ZFS-Dateisystemereignissen werden angezeigt. Beispiel:

```
# zpool history
History for 'rpool':
2009-08-26.16:49:07 zpool create -f -o failmode=continue -R /a -m legacy -o cachefile=
/tmp/root/etc/zfs/zpool.cache rpool c1t1d0s0
2009-08-26.16:49:08 zfs set canmount=noauto rpool
2009-08-26.16:49:08 zfs set mountpoint=/rpool rpool
```

```
2009-08-26.16:49:09 zfs create -o mountpoint=legacy rpool/ROOT
2009-08-26.16:49:10 zfs create -b 8192 -V 2048m rpool/swap
2009-08-26.16:49:11 zfs create -b 131072 -V 1024m rpool/dump
2009-08-26.16:49:14 zfs create -o canmount=noauto rpool/ROOT/zfs1009BE
2009-08-26.16:49:15 zpool set bootfs=rpool/ROOT/zfs1009BE rpool
2009-08-26.16:49:15 zfs set mountpoint=/ rpool/ROOT/zfs1009BE
2009-08-26.16:49:16 zfs set canmount=on rpool
2009-08-26.16:49:17 zfs create -o mountpoint=/export rpool/export
2009-08-26.16:49:18 zfs create rpool/export/home
2009-08-28.08:17:59 zpool attach rpool c1t1d0s0 c1t0d0s0
```

- Die Option `-l` zum Anzeigen eines langen Formats mit Benutzernamen, Hostnamen und Angabe der Zone, in der die Operation ausgeführt wurde. Beispiel:

```
# zpool history -l rpool
History for 'rpool':
2009-08-26.16:49:07 zpool create -f -o failmode=continue -R /a -m legacy -o cachefile=
/tmp/root/etc/zfs/zpool.cache rpool c1t1d0s0 [user root on neo:global]
2009-08-26.16:49:08 zfs set canmount=noauto rpool [user root on neo:global]
2009-08-26.16:49:08 zfs set mountpoint=/rpool rpool [user root on neo:global]
2009-08-26.16:49:09 zfs create -o mountpoint=legacy rpool/ROOT [user root on neo:global]
2009-08-26.16:49:10 zfs create -b 8192 -V 2048m rpool/swap [user root on neo:global]
2009-08-26.16:49:11 zfs create -b 131072 -V 1024m rpool/dump [user root on neo:global]
2009-08-26.16:49:14 zfs create -o canmount=noauto rpool/ROOT/zfs1009BE [user root on neo:global]
2009-08-26.16:49:15 zpool set bootfs=rpool/ROOT/zfs1009BE rpool [user root on neo:global]
2009-08-26.16:49:15 zfs set mountpoint=/ rpool/ROOT/zfs1009BE [user root on neo:global]
2009-08-26.16:49:16 zfs set canmount=on rpool [user root on neo:global]
2009-08-26.16:49:17 zfs create -o mountpoint=/export rpool/export [user root on neo:global]
2009-08-26.16:49:18 zfs create rpool/export/home [user root on neo:global]
2009-08-28.08:17:59 zpool attach rpool c1t1d0s0 c1t0d0s0 [user root on neo:global]
```

- Die Option `-i` zum Anzeigen von Informationen zu internen Ereignissen. Diese sind für diagnostische Zwecke nutzbar. Beispiel:

```
# zpool history -i rpool
History for 'rpool':
2009-08-26.16:49:07 zpool create -f -o failmode=continue -R /a -m legacy -o cachefile=
/tmp/root/etc/zfs/zpool.cache rpool c1t1d0s0
2009-08-26.16:49:07 [internal property set txg:6] mountpoint=/ dataset = 16
2009-08-26.16:49:07 [internal property set txg:7] mountpoint=legacy dataset = 16
2009-08-26.16:49:08 [internal property set txg:8] canmount=2 dataset = 16
2009-08-26.16:49:08 zfs set canmount=noauto rpool
2009-08-26.16:49:08 [internal property set txg:10] mountpoint=/rpool dataset = 16
2009-08-26.16:49:08 zfs set mountpoint=/rpool rpool
2009-08-26.16:49:09 [internal create txg:12] dataset = 31
2009-08-26.16:49:09 [internal property set txg:13] mountpoint=legacy dataset = 31
2009-08-26.16:49:09 zfs create -o mountpoint=legacy rpool/ROOT
2009-08-26.16:49:09 [internal create txg:15] dataset = 37
2009-08-26.16:49:10 [internal property set txg:16] refreservation=2147483648 dataset = 37
```

```
2009-08-26.16:49:10 [internal refreservation set txg:16] 2147483648 dataset = 37
2009-08-26.16:49:10 zfs create -b 8192 -V 2048m rpool/swap
2009-08-26.16:49:10 [internal create txg:18] dataset = 43
2009-08-26.16:49:10 [internal property set txg:19] refreservation=1073741824 dataset = 43
2009-08-26.16:49:10 [internal refreservation set txg:19] 1073741824 dataset = 43
.
.
.
```

Weitere Informationen zur Verwendung des Befehls `zpool history` entnehmen Sie bitte dem Abschnitt „Erkennen von Problemen in ZFS“ auf Seite 295.

Upgrade von ZFS-Dateisystemen (zfs upgrade)

Release Solaris 10 10/08: Dieses Release bietet den Befehl `zfs upgrade`, mit dem Dateisysteme künftig auf neue ZFS-Dateisystemverbesserungen aufgerüstet werden können. ZFS-Speicherpools besitzen eine ähnliche Upgrade-Funktion, um vorhandene Speicherpools um neue Funktionalität zu erweitern.

Beispiel:

```
# zfs upgrade
This system is currently running ZFS filesystem version 3.

All filesystems are formatted with the current version.
```

Hinweis – Auf aufgerüstete Dateisysteme sowie aus diesen mit dem Befehl `zfs send` erstellte Datenstreams kann auf Systemen nicht zugegriffen werden, die unter einer älteren Softwareversion laufen.

Delegierte ZFS-Administration

Release Solaris 10 10/08: In diesem Release können fein abgestimmte Zugriffsrechte für die Durchführung von ZFS-Administrationsaufgaben an Benutzer ohne ausreichende Zugriffsrechte delegiert werden.

Zum Gewähren und Verweigern von Zugriffsrechten dienen die Befehle `zfs allow` und `zfs unallow`.

Mit der Speicherpool-Eigenschaft `delegation` kann die delegierte Administration aktiviert und deaktiviert werden. Beispiel:

```
# zpool get delegation users
NAME  PROPERTY    VALUE       SOURCE
users  delegation  on          default
```

```
# zpool set delegation=off users
# zpool get delegation users
NAME  PROPERTY    VALUE       SOURCE
users  delegation  off         local
```

Standardmäßig ist die Eigenschaft `delegation` aktiviert.

Weitere Informationen entnehmen Sie bitte Kapitel 9, „Delegierte ZFS-Administration“ und `zfs(1M)`.

Einrichten separater ZFS-Protokolliergeräte

Release Solaris 10 10/08: Das Protokoll ZIL (ZFS Intent Log) erfüllt die POSIX-Anforderungen für synchrone Transaktionen. So setzen Datenbanken bei der Rückkehr von Systemaufrufen beispielsweise oft voraus, dass Transaktionen auf stabilen Speichergeräten stattfinden. NFS und andere Anwendungen können zur Gewährleistung der Datenstabilität ebenfalls fsync() verwenden. Standardmäßig wird das ZIL aus Blöcken innerhalb des Hauptspeicherpools zugewiesen. Durch Verwendung getrennter Intent-Protokolliergeräte im ZFS-Speicher-Pool wie z. B. NVRAM oder eine speziell dafür vorgesehene Festplatte kann jedoch eine höhere Leistung erreicht werden.

Protokolliergeräte für das Intent-Protokoll von ZFS sind etwas Anderes als Datenbankprotokolldateien.

Die Einrichtung eines ZFS-Protokolliergeräts kann bei oder nach der Erstellung des Speicher-Pools erfolgen. Beispiele zum Einrichten von Protokolliergeräten finden Sie unter „Erstellen eines ZFS-Speicher-Pools mit Protokolliergeräten“ auf Seite 70und „Hinzufügen von Datenspeichergeräten zu einem Speicher-Pool“ auf Seite 77.

Sie können zur Datenspiegelung an ein vorhandenes Protokolliergerät ein weiteres Protokolliergerät anschließen. Dies entspricht dem Verbinden eines Speichergeräts in einem Speicher-Pool ohne Datenspiegelung.

Berücksichtigen Sie folgende Aspekte bei der Überlegung, ob die Einrichtung eines ZFS-Protokolliergeräts für Ihre Umgebung ratsam ist:

- Jegliche Leistungssteigerung durch die Implementierung eines separaten Protokolliergeräts ist von der Art des Geräts, der Hardwarekonfiguration des Speicher-Pools sowie von der Arbeitslast der Anwendung abhängig. Vorläufige Informationen zur Leistung finden Sie in diesem Blog:

  `http://blogs.sun.com/perrin/entry/slog_blog_or_blogging_on`
- Es ist möglich, Protokolliergeräte zu spiegeln oder deren Replikation aufzuheben. RAID-Z wird für Protokolliergeräte jedoch nicht unterstützt.
- Wenn ein separates Protokolliergerät nicht gespiegelt wurde und das Gerät mit den Protokollen ausfällt, wird durch Speicherung von Protokollblöcken auf den Speicher-Pool zurückgeschaltet.

- Protokolliergeräte können als Teil des Speicher-Pools hinzugefügt, ersetzt, angehängt, abgehängt und importiert und exportiert werden. Derzeit können Protokolliergeräte nicht entfernt werden.
- Die Mindestgröße für ein Protokolliergerät entspricht der Mindestgröße für jedes Gerät in einem Pool. Dies sind 64 MB. Auf einem Protokolliergerät können verhältnismäßig wenige Ausführungsdaten gespeichert werden. Bei Bestätigung der Protokolltransaktion (Systemaufruf) werden die Protokollblöcke geleert.
- Ein Protokolliergerät sollte nicht größer als rund die Hälfte des physischen Speichers sein, da dies die Höchstmenge an potenziellen Ausführungsdaten ist, die gespeichert werden kann. So sollten Sie beispielsweise für ein System mit 16 GB physischem Speicher ein Protokolliergerät von höchstens 8 GB Größe in Erwägung ziehen.

Erstellen intermediärer ZFS-Datasets

Release Solaris 10 10/08: Durch Verwendung der Option `-p` mit den Befehlen `zfs create`, `zfs clone` und `zfs rename` können Sie schnell ein intermediäres Dataset erstellen, falls es noch nicht vorhanden ist.

So können Sie beispielsweise ZFS-Datasets (`users/area51`) im Speicher-Pool `datab` erstellen.

```
# zfs list
NAME                        USED  AVAIL  REFER  MOUNTPOINT
datab                       106K  16.5G    18K  /datab
# zfs create -p -o compression=on datab/users/area51
```

Wenn während des Erstellungsvorgangs bereits ein intermediäres Dataset vorhanden ist, wird er ohne Fehlermeldung abgeschlossen.

Angegebene Eigenschaften gelten für das Ziel-Dataset und nicht für die intermediären Datasets. Beispiel:

```
# zfs get mountpoint,compression datab/users/area51
NAME                PROPERTY     VALUE                SOURCE
datab/users/area51  mountpoint   /datab/users/area51  default
datab/users/area51  compression  on                   local
```

Es wird ein intermediäres Dataset mit Standard-Einhängepunkt erstellt. Alle zusätzlichen Eigenschaften werden für dieses intermediäre Dataset deaktiviert. Beispiel:

```
# zfs get mountpoint,compression datab/users
NAME         PROPERTY     VALUE         SOURCE
datab/users  mountpoint   /datab/users  default
datab/users  compression  off           default
```

Weitere Informationen finden Sie in der Manpage `zfs(1M)`.

Verbesserungen der ZFS-Hotplug-Funktion

Release Solaris 10 10/08: In diesem Release reagiert ZFS effektiver auf Speichergeräte, die entfernt werden. Darüber hinaus steht ein Mechanismus zur automatischen Erkennung von neu eingesetzten Speichergeräten zur Verfügung, der folgende Verbesserungen bietet:

- Sie können ein Speichergerät durch ein anderes auswechseln, ohne dafür den Befehl `zpool replace` eingeben zu müssen.

  Die Eigenschaft `autoreplace` legt die Charakteristika des automatischen Erkennens ausgewechselter Geräte fest. Wenn diese Eigenschaft auf „off“ gesetzt ist, muss das Auswechseln von Speichergeräten vom Administrator mithilfe des Befehls `zpool replace` initiiert werden. Wenn diese Eigenschaft auf „on“ gesetzt ist, wird das neue Speichergerät an der physischen Adresse des vorherigen Speichergeräts im Pool automatisch formatiert und in den Pool eingebunden. Standardeinstellung ist „off“.
- Für die physische Entfernung eines Speichergeräts bzw. Hot-Spares bei laufendem System gibt es jetzt den Speicher-Pool-Status `REMOVED`. Falls verfügbar, wird ein Hot-Spare für das entfernte Speichergerät in den Pool eingebunden.
- Wenn ein Speichergerät entfernt und danach wieder eingesetzt wird, wird es online geschaltet. Wenn für das entfernte Speichergerät ein Hot-Spare eingebunden wurde, wird dieses bei Abschluss der Online-Schaltung wieder entfernt.
- Das automatische Erkennen entfernter und hinzugefügter Speichergeräte ist hardwareabhängig und wird nicht von allen Plattformen unterstützt. So werden USB-Speichergeräte beispielsweise beim Einfügen automatisch konfiguriert. SATA-Laufwerke müssen unter Umständen jedoch mit dem Befehl `cfgadm -c configure` konfiguriert werden.
- Hot-Spares werden regelmäßig überprüft, um sicherzustellen, dass sie online und verfügbar sind.

Weitere Informationen finden Sie in der Manpage `zpool(1M)`.

Rekursives Umbenennen von ZFS-Snapshots (zfs rename -r)-

Release Solaris 10 10/08: Der Befehl `zfs rename -r` ermöglicht es, alle untergeordneten ZFS-Snapshots rekursiv umzubenennen.

Erstellen Sie z. B. einen Snapshot von einer Reihe von ZFS-Dateisystemen.

```
# zfs snapshot -r users/home@today
# zfs list
NAME                    USED  AVAIL  REFER  MOUNTPOINT
users                   216K  16.5G    20K  /users
```

```
users/home                76K  16.5G    22K  /users/home
users/home@today            0      -    22K  -
users/home/markm          18K  16.5G    18K  /users/home/markm
users/home/markm@today      0      -    18K  -
users/home/marks          18K  16.5G    18K  /users/home/marks
users/home/marks@today      0      -    18K  -
users/home/neil           18K  16.5G    18K  /users/home/neil
users/home/neil@today       0      -    18K  -
```

Führen Sie am nächsten Tag eine Umbenennung der Snapshots durch.

```
# zfs rename -r users/home@today @yesterday
# zfs list
NAME                          USED  AVAIL  REFER  MOUNTPOINT
users                         216K  16.5G    20K  /users
users/home                     76K  16.5G    22K  /users/home
users/home@yesterday             0      -    22K  -
users/home/markm               18K  16.5G    18K  /users/home/markm
users/home/markm@yesterday       0      -    18K  -
users/home/marks               18K  16.5G    18K  /users/home/marks
users/home/marks@yesterday       0      -    18K  -
users/home/neil                18K  16.5G    18K  /users/home/neil
users/home/neil@yesterday        0      -    18K  -
```

Snapshots sind die einzigsten Daten, die rekursiv umbenannt werden können.

Weitere Informationen zu Snapshots finden Sie unter „Überblick über ZFS-Snapshots“ auf Seite 217 und in folgendem Blog-Eintrag, in dem die Erstellung rotierender Snapshots beschrieben ist:

`http://blogs.sun.com/mmusante/entry/rolling_snapshots_made_easy`

GZIP-Komprimierung für ZFS

Release Solaris 10 10/08: In diesem Solaris-Release können ZFS-Dateisysteme zusätzlich zu `lzjb` auch mit `gzip` komprimiert werden. Sie können festlegen, dass die Komprimierung vom Typ `gzip` (Standardeinstellung) sein soll oder vom Typ `gzip-`*N*, wobei *N* den Wert 1 bis 9 haben kann. Beispiel:

```
# zfs create -o compression=gzip users/home/snapshots
# zfs get compression users/home/snapshots
NAME                  PROPERTY     VALUE            SOURCE
users/home/snapshots  compression  gzip             local
# zfs create -o compression=gzip-9 users/home/oldfiles
# zfs get compression users/home/oldfiles
NAME                  PROPERTY     VALUE           SOURCE
users/home/oldfiles   compression  gzip-9          local
```

Weitere Informationen zum Setzen von ZFS-Eigenschaften finden Sie unter „Setzen von ZFS-Eigenschaften“ auf Seite 196.

Speichern mehrerer Kopien von ZFS-Benutzerdaten

Release Solaris 10 10/08: Sofern möglich, werden ZFS-Dateisystem-Metadaten im Rahmen der Zuverlässigkeitsfunktionen automatisch auf mehreren Festplatten gespeichert. Dies wird als *ditto blocks* bezeichnet.

In dieser Version können Sie über den Befehl `zfs set copies` festlegen, dass mehrere Kopien der Benutzerdaten auch pro Dateisystem gespeichert werden. Beispiel:

```
# zfs set copies=2 users/home
# zfs get copies users/home
NAME        PROPERTY  VALUE       SOURCE
users/home  copies    2           local
```

Verfügbare Werte sind 1, 2 oder 3. Der Standardwert ist 1. Diese Kopien werden zusätzlich zu den von Redundanzfunktionen (Datenspiegelung bzw. RAID-Z) auf Pool-Ebene angelegten Sicherungskopien erstellt.

Die Speicherung mehrerer Kopien von ZFS-Benutzerdaten bringt die folgenden Vorteile mit sich:

- Verbesserte Datenaufbewahrung, da für alle ZFS-Konfigurationen die Wiederherstellung von nicht wiederherstellbaren Blocklesefehlern zugelassen wird, z. B. Datenträgerfehler (bit rot)
- Bietet Schutz der Daten auch wenn nur ein Laufwerk verfügbar ist
- Ermöglicht die Auswahl von Datenschutzrichtlinien auf Dateisystembasis jenseits der Grenzen des Speicher-Pools

Je nach Zuweisung der ditto blocks im Speicher-Pool ist es möglich, dass mehrere Kopien auf einer einzigen Festplatte abgelegt werden. Ein Ausfall einer ganzen Festplatte kann folglich bedeuten, dass sämtliche ditto blocks unverfügbar sind.

Die Verwendung von ditto blocks kann hilfreich sein, wenn Sie versehentlich einen nicht-redundanten Pool erstellen und Richtlinien für die Datenaufbewahrung festlegen müssen.

In folgendem Blog wird ausführlich darauf eingegangen, wie sich die Erstellung von Kopien auf einem System mit Einzelplatten-Pool oder Mehrplatten-Pool auf den Datenschutz im Allgemeinen auswirken kann:

http://blogs.sun.com/relling/entry/zfs_copies_and_data_protection

Weitere Informationen zum Setzen von ZFS-Eigenschaften finden Sie unter „Setzen von ZFS-Eigenschaften“ auf Seite 196.

Verbesserte Ausgabe von `zpool status`

Release Solaris 10 8/07: Mit dem Befehl `zpool status -v` können Sie sich eine Liste der Dateien mit dauerhaften Fehlern ausgeben lassen. In früheren Versionen mussten die Dateinamen mithilfe des Befehls `find -inum` anhand der Liste der angezeigten Knoten ermittelt werden.

Weitere Informationen zum Anzeigen einer Liste von Dateien mit dauerhaften Fehlern finden Sie unter „Reparatur beschädigter Dateien bzw. Verzeichnisse“ auf Seite 315.

Verbesserungen für ZFS mit Solaris iSCSI

Release Solaris 10 8/07: In diesem Solaris-Release können Sie durch Setzen der Eigenschaft `shareiscsi` im ZFS-Volume ein ZFS-Volume als ein Solaris iSCSI-Zielgerät erstellen. Mithilfe dieses Verfahrens können Solaris iSCSI-Zielgeräte schnell eingerichtet werden. Beispiel:

```
# zfs create -V 2g tank/volumes/v2
# zfs set shareiscsi=on tank/volumes/v2
# iscsitadm list target
Target: tank/volumes/v2
    iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
    Connections: 0
```

Nach dem Erstellen des iSCSI-Zielgeräts muss der iSCSI-Initiator definiert werden. Informationen zur Einrichtung eines Solaris iSCSI-Initiators finden Sie in Kapitel 14, „Configuring Solaris iSCSI Targets and Initiators (Tasks)“ in *System Administration Guide: Devices and File Systems*.

Weitere Informationen zur Verwaltung eines ZFS-Volumes als iSCSI-Zielgerät entnehmen Sie bitte dem Abschnitt „Verwendung von ZFS-Volumes als Solaris-iSCSI-Zielgerät“ auf Seite 281.

ZFS-Befehlsprotokoll (`zpool history`)

Release Solaris 10 8/07: In diesem Solaris-Release protokolliert ZFS automatisch erfolgreich ausgeführte Aufrufe der Befehle `zfs` und `zpool`, die Daten zu Pool-Zuständen ändern. Beispiel:

```
# zpool history
History for 'newpool':
2007-04-25.11:37:31 zpool create newpool mirror c0t8d0 c0t10d0
2007-04-25.11:37:46 zpool replace newpool c0t10d0 c0t9d0
2007-04-25.11:38:04 zpool attach newpool c0t9d0 c0t11d0
2007-04-25.11:38:09 zfs create newpool/user1
2007-04-25.11:38:15 zfs destroy newpool/user1
```

```
History for 'tank':
2007-04-25.11:46:28 zpool create tank mirror c1t0d0 c2t0d0 mirror c3t0d0 c4t0d0
```

Dank dieses Leistungsmerkmals können Sie oder Sun-Supportmitarbeiter *genau* feststellen, welche ZFS-Befehle bei der Behebung eines Fehlers ausgeführt wurden.

Spezifische Speicher-Pools können mit dem Befehl `zpool history` identifiziert werden. Beispiel:

```
# zpool history newpool
History for 'newpool':
2007-04-25.11:37:31 zpool create newpool mirror c0t8d0 c0t10d0
2007-04-25.11:37:46 zpool replace newpool c0t10d0 c0t9d0
2007-04-25.11:38:04 zpool attach newpool c0t9d0 c0t11d0
2007-04-25.11:38:09 zfs create newpool/user1
2007-04-25.11:38:15 zfs destroy newpool/user1
```

In diesem Solaris-Release werden`user-ID`, *hostname* und *zone-name* über den Befehl *zpool history* nicht aufgezeichnet. Weitere Informationen finden Sie unter „Verbesserungen des ZFS-Befehlsprotokolls (`zpool history`)“ auf Seite 25.

Weitere Informationen zur Behebung von ZFS-Problemen siehe „Erkennen von Problemen in ZFS“ auf Seite 295.

Verbesserungen der ZFS-Eigenschaften

ZFS-Eigenschaft `xattr`

Release Solaris 10 8/07: Mit der Eigenschaft `xattr` können Sie für ein bestimmtes ZFS-Dateisystem erweiterte Attribute deaktivieren oder aktivieren. Standardmäßig sind sie aktiviert. Eine Beschreibung von ZFS-Eigenschaften finden Sie unter „ZFS-Eigenschaften“ auf Seite 179.

ZFS-Eigenschaft `canmount`

Release Solaris 10 8/07: Mit der neuen Eigenschaft `canmount` können Sie festlegen, ob ein Dataset mithilfe des Befehls `zfs mount` eingehängt werden kann. Weitere Informationen dazu finden Sie unter „Die Eigenschaft `canmount`“ auf Seite 191.

Benutzerdefinierte ZFS-Eigenschaften

Release Solaris 10 8/07: Zusätzlich zu den nativen Standardeigenschaften, die zum Exportieren interner Statistiken oder Steuern des ZFS-Dateisystemverhaltens dienen, unterstützt ZFS benutzerdefinierte Eigenschaften. Benutzerdefinierte Eigenschaften wirken sich nicht auf das ZFS-Verhalten aus, können jedoch zum Versehen von Datasets mit Informationen, die für Ihre lokalen Gegebenheiten wichtig sind, verwendet werden.

Weitere Informationen finden Sie unter „Benutzerdefinierte ZFS-Eigenschaften“ auf Seite 192.

Setzen von Eigenschaften beim Erstellen von ZFS-Dateisystemen

Release Solaris 10 8/07: In diesem Solaris-Release können Sie Eigenschaften nicht nur nach, sondern auch bereits bei der Erstellung von Dateisystemen festlegen.

Die folgenden Beispiele zeigen die entsprechende Syntax:

```
# zfs create tank/home
# zfs set mountpoint=/export/zfs tank/home
# zfs set sharenfs=on tank/home
# zfs set compression=on tank/home

# zfs create -o mountpoint=/export/zfs -o sharenfs=on -o compression=on tank/home
```

Anzeigen aller ZFS-Dateisysteminformationen

Release Solaris 10 8/07: In diesem Solaris-Release können Sie sich mithilfe verschiedener Versionen des Befehls `zfs get` Informationen zu allen Datasets anzeigen lassen, wenn weder ein Dataset noch `all` angegeben wurde. Bisher war es nicht möglich, mit dem Befehl `zfs get` Informationen aller Datensätze anzuzeigen.

Beispiel:

```
# zfs get -s local all
tank/home               atime          off                    local
tank/home/bonwick       atime          off                    local
tank/home/marks         quota          50G                    local
```

Neue Option `F` für `-zfs receive`

Release Solaris 10 8/07: Sie können nun die neue Option `-F` für den Befehl `zfs receive` verwenden, um das Dateisystem auf den letzten Snapshot vor dem Empfang zurückzusetzen. Die Verwendung dieser Option kann erforderlich werden, wenn das Dateisystem zwischen dem Zeitpunkt des Rollbacks und des Beginns der receive-Operation geändert wurde.

Weitere Informationen finden Sie unter „Empfangen von ZFS-Snapshots“ auf Seite 227.

Rekursive ZFS-Snapshots

Release Solaris 10 11/06: Wenn Sie zum Erstellen eines Dateisystem-Snapshots den Befehl `zfs snapshot` verwenden, können Sie durch die Verwendung der Option `-r` erreichen, dass für alle

untergeordneten Dateisysteme rekursiv Snapshots erstellt werden. Darüber hinaus werden bei der Löschung eines Snapshots mit der Option `-r` alle nachfolgenden Snapshots rekursiv gelöscht.

Rekursive ZFS-Snapshots werden schnell als eine einzige Kernoperation erstellt. Snapshots werden entweder zusammen (d. h. alle auf einmal) oder gar nicht erstellt. Der Vorteil solcher simultaner Snapshots besteht darin, dass die Snapshot-Daten auch bei Dateisystemhierarchien zu einem einzigen konsistenten Zeitpunkt erstellt werden.

Weitere Informationen finden Sie unter „Erstellen und Löschen von ZFS-Snapshots“ auf Seite 218.

RAID-Z-Konfiguration mit doppelter Parität (`raidz2`)

Release Solaris 10 11/06: Redundante RAID-Z-Konfigurationen können jetzt einfache oder doppelte Parität besitzen. Das bedeutet, dass in einem System bis zu zwei Geräteausfälle ohne Datenverlust möglich sind. Eine RAID-Z-Konfiguration doppelter Parität kann mithilfe des Schlüsselworts `raidz2` angegeben werden. Entsprechend können Sie für eine RAID-Z-Konfiguration mit einfacher Parität eines der Schlüsselwörter `raidz` oder `raidz1` angeben.

Weitere Informationen finden Sie unter „Erstellen von RAID-Z-Speicher-Pools“ auf Seite 68 oder `zpool(1M)`.

Hot-Spares für ZFS-Speicher-Pools

Release Solaris 10 11/06: Mithilfe der ZFS-Hot-Spare-Funktion können Sie Datenträger ermitteln, die zum Ersetzen eines ausgefallenen bzw. fehlerhaften Geräts in einem bzw. mehreren Speicher-Pools verwendet werden können. Das Vorsehen eines Datenträgers als *Hot-Spare*-Gerät bedeutet, dass bei Ausfall eines aktiven Datenträgers im Pool das Hot-Spare-Gerät diesen automatisch ersetzt. Alternativ dazu können Sie Datenträger in einem Speicher-Pool auch manuell durch ein Hot-Spare-Gerät ersetzen.

Weitere Informationen finden Sie unter „Zuweisen von Hot-Spares im Speicher-Pool“ auf Seite 88 und `zpool(1M)`.

Ersetzen eines ZFS-Dateisystems durch einen ZFS-Klon (`zfs promote`)

Release Solaris 10 11/06: Der Befehl `zfs promote` ermöglicht es, ein vorhandenes ZFS-Dateisystem durch einen Klon desselben zu ersetzen. Diese Funktion ist hilfreich, wenn Sie an verschiedenen Versionen eines Dateisystems Tests ausführen und danach eine alternative Version des Dateisystems zum aktiven Dateisystem machen möchten.

Weitere Informationen finden Sie in „Ersetzen eines ZFS-Dateisystems durch einen ZFS-Klon“ auf Seite 224 und zfs(1M).

Aufrüsten von ZFS-Speicher-Pools (zpool upgrade)

Release Solaris 10 6/06: Mit dem Befehl zpool upgrade können Sie ein Upgrade eines Speicher-Pools vornehmen, um die neuesten Funktionen zu nutzen. Darüber hinaus wurde der Befehl zpool status so geändert, dass Sie jetzt darauf hingewiesen werden, wenn Pools mit älteren Versionen laufen.

Weitere Informationen finden Sie unter „Aufrüsten von ZFS-Speicher-Pools“ auf Seite 112 und zpool(1M).

Wenn Sie die ZFS-Administrationskonsole auf einem System mit einem Pool aus früheren Solaris-Versionen nutzen möchten, müssen Sie die Pools vor Verwendung der ZFS-Administrationskonsole aufrüsten. Mit dem Befehl zpool status lässt sich ermitteln, ob Pools aufgerüstet werden müssen. Informationen zur ZFS-Administrationskonsole finden Sie unter „Webbasierte ZFS-Verwaltung“ auf Seite 39.

ZFS-Befehle „backup“ und „restore“ wurden umbenannt

Release Solaris 10 6/06: In diesem Solaris-Release heißen die Befehle zfs backup und zfs restore nun zfs send und zfs receive. Diese Namen geben die Funktion der Befehle genauer wieder. Diese Befehle dienen zum Speichern und Abrufen von ZFS-Datenstreaminstanzen.

Weitere Informationen zu diesen Befehlen finden Sie unter „Senden und Empfangen von ZFS-Daten“ auf Seite 225.

Wiederherstellen gelöschter Speicher-Pools

Release Solaris 10 6/06: Dieses Solaris-Release enthält den Befehl zpool import -D. Damit können Sie Pools wiederherstellen, die zuvor mit dem Befehl zpool destroy gelöscht wurden.

Weitere Informationen dazu finden Sie unter „Wiederherstellen gelöschter ZFS-Speicher-Pools“ auf Seite 110.

Integration von ZFS mit Fault Manager

Release Solaris 10 6/06: Dieses Release enthält ein integriertes ZFS-Diagnoseprogramm, das Pool- und Datenträgerausfälle diagnostiziert und meldet. Darüber hinaus werden auch mit solchen Pool- bzw. Datenträgerausfällen im Zusammhang stehende Prüfsummen-, E/A-, Geräte- und Poolfehler gemeldet.

Das Diagnoseprogramm enthält keine Funktion zur Früherkennung von Prüfsummen- bzw. E/A-Fehlern und umfasst auch keine auf Fehleranalysen beruhenden proaktiven Operationen.

Bei einem ZFS-Ausfall wird von `fmd` in etwa die folgende Meldung angezeigt:

```
SUNW-MSG-ID: ZFS-8000-D3, TYPE: Fault, VER: 1, SEVERITY: Major
EVENT-TIME: Fri Aug 28 09:10:27 PDT 2009
PLATFORM: SUNW,Sun-Fire-T200, CSN: -, HOSTNAME: neo
SOURCE: zfs-diagnosis, REV: 1.0
EVENT-ID: d6725ad6-4546-6c48-fa16-eace4d371981
DESC: A ZFS device failed.  Refer to http://sun.com/msg/ZFS-8000-D3 for more information.
AUTO-RESPONSE: No automated response will occur.
IMPACT: Fault tolerance of the pool may be compromised.
REC-ACTION: Run 'zpool status -x' and replace the bad device.
```

Durch Überprüfen der empfohlenen Aktion, die nach den spezifischeren Anweisungen im Befehl `zpool status` angezeigt wird, können Sie die Fehlerursache schnell erkennen und beheben.

Ein Beispiel für die Wiederherstellung des Normalbetriebs nach einem aufgetretenen ZFS-Problem finden Sie unter „Reparieren eines ausgefallenen Geräts“ auf Seite 301.

Neuer Befehl `zpool clear`

Release Solaris 10 6/06: Dieses Release enthält den Befehl `zpool clear`, mit dem Fehlerzähler für Geräte- bzw. Poolausfälle zurückgesetzt werden können. In früheren Versionen wurden Fehlerzähler bei der Wiederinbetriebnahme eines Datenträgers im Pool mithilfe des Befehls `zpool online` zurückgesetzt. Weitere Informationen finden Sie unter `zpool(1M)` und „Löschen von Gerätefehlern im Speicher-Pool“ auf Seite 86.

Kompaktes Format von NFSv4-Zugriffssteuerungslisten

Release Solaris 10 6/06: In diesem Release sind für NFSv4-Zugriffssteuerungslisten drei Formate verfügbar: ausführlich, positional und kompakt. Mit den neuen kompakten und positionalen Zugriffssteuerungslistenformaten können Zugriffssteuerungslisten gesetzt und

angezeigt werden. Mit dem Befehl `chmod` können Sie alle drei Zugriffssteuerungslistenformate setzen. Mit dem Befehl `ls -V` können Sie Zugriffssteuerungslisten im kompakten und positionalen Format, mit `ls -v` im ausführlichen Format anzeigen.

Weitere Informationen finden Sie unter „Setzen und Anzeigen von Zugriffssteuerungslisten an ZFS-Dateien im Kompaktformat“ auf Seite 258, `chmod(1)` und `ls(1)`.

Dienstprogramm `fsstat` zum Überwachen von Dateisystemen

Release Solaris 10 6/06: Zum Melden von Operationen an Dateisystemen steht das neue Dienstprogramm `fsstat` zur Dateisystemüberwachung zur Verfügung. Aktivitäten können nach Einhängepunkt oder Dateisystemtypen protokolliert werden. Das folgende Beispiel zeigt eine allgemeine Aktivität eines ZFS-Dateisystems.

```
$ fsstat zfs
 new  name   name  attr  attr lookup rddir  read read  write write
 file remov  chng   get   set    ops   ops   ops bytes   ops bytes
7.82M 5.92M 2.76M 1.02G 3.32M  5.60G 87.0M  363M 1.86T 20.9M  251G zfs
```

Weitere Informationen finden Sie in der Manpage `fsstat(1M)`.

Webbasierte ZFS-Verwaltung

Release Solaris 10 6/06: Zum Ausführen vieler Administrationsaufgaben steht ein webbasiertes ZFS-Verwaltungsprogramm zur Verfügung. Mit diesem Programm können Sie folgende Aufgaben ausführen:

- Erstellen eines neuen Datenspeicher-Pools
- Erweitern der Kapazität eines vorhandenen Datenspeicher-Pools
- Verlagern (Exportieren) eines Datenspeicher-Pools auf ein anderes System
- Importieren eines zuvor exportierten Datenspeicher-Pools, um diesen auf einem anderen System verfügbar zu machen
- Anzeigen von Informationen zu Datenspeicher-Pools
- Erstellen von Dateisystemen
- Erstellen von Volumes
- Erstellen eines Snapshots von Dateisystemen bzw. Volumes
- Wiederherstellen eines früheren Snapshots eines Dateisystems

Sie können mithilfe eines sicheren Webbrowsers unter der folgenden URL auf die ZFS-Administrationskonsole zugreifen:

```
https://system-name:6789/zfs
```

Wenn Sie die entsprechende URL eingeben und die ZFS-Administrationskonsole nicht erreichen, kann es sein, dass deren Server nicht läuft. Geben Sie den folgenden Befehl ein, um den Server zu starten:

```
# /usr/sbin/smcwebserver start
```

Geben Sie den folgenden Befehl ein, wenn der Server beim Hochfahren des Systems automatisch gestartet werden soll:

```
# /usr/sbin/smcwebserver enable
```

Hinweis – Die Solaris Management Console (`smc`) kann nicht zur Verwaltung von ZFS-Speicher-Pools bzw. -Dateisystemen verwendet werden.

Was ist ZFS?

Das ZFS-Dateisystem ist ein revolutionäres neues Dateisystem, das die Verwaltung von Dateisystemen grundlegend ändert. Es besitzt Leistungsmerkmale und Vorteile, die in keinem anderen heutzutage verfügbaren Dateisystem zu finden sind. ZFS ist robust, skalierbar und einfach zu verwalten.

Speicher-Pools in ZFS

Die physische Datenspeicherung beruht bei ZFS auf dem Konzept der *Speicher-Pools*. Früher wurden Dateisysteme auf ein einziges physisches Datenspeichergerät aufsetzend konzipiert. Damit mehrere Datenspeichergeräte addressiert werden können und Datenredundanz möglich wird, entwickelte man sogenannte *Volume Manager*, durch die Dateisysteme nur einen Datenträger „sehen“, wo tatsächlich mehrere Datenträger vorhanden sind. So mussten Dateisysteme nicht modifiziert werden, damit sie mit einem System mehrerer Datenspeichergeräte arbeiten können. Dieses Design fügte ein neues Komplexitätsniveau hinzu und behinderte bestimmte Weiterentwicklungen an Dateisystemen, da das Dateisystem keine Kontrolle über die physische Speicherung von Daten auf den zu einem virtuellen System zusammengefassten einzelnen Datespeichergeräten (Volumes) hatte.

In ZFS ist diese Verwaltung einzelner Volumes nicht mehr erforderlich. Anstatt einer erzwungenen Erstellung virtueller Volume-Systeme fasst ZFS Datenspeichergeräte in so genannten Speicher-Pools zusammen. Ein solcher Speicher-Pool beschreibt die physischen Eigenschaften der Speicherung (Gerätestruktur, Datenredundanz usw.) und fungiert als flexibler Datenspeicher, aus dem Dateisysteme erstellt werden können. Dateisysteme sind nicht mehr auf individuelle Datenträger beschränkt und können den Speicherplatz im Pool

gemeinsam nutzen. Sie müssen die Kapazität eines Dateisystems nicht mehr vorher angeben, da diese automatisch innerhalb des im Speicher-Pool zugewiesenen Speicherplatzes erweitert werden. Wenn ein Pool um neuen Speicherplatz erweitert wird, können alle Dateisysteme dieses Pools diesen neuen Speicherplatz sofort nutzen, ohne dass dafür Konfigurationen geändert werden müssen. Ein Speicher-Pool verhält sich in vielerlei Hinsicht wie ein virtuelles Speichersystem. Wenn ein System um neue DIMM-Speicherchips erweitert wird, brauchen Sie auf Betriebssystemebene keine Befehle einzugeben, mit denen dieser neue Speicher konfiguriert und zu einzelnen Prozessen zugewiesen werden muss. Alle Prozesse des Systems verwenden automatisch diesen zusätzlichen Speicherplatz.

Transaktionale Semantik

ZFS ist ein transaktionales Dateisystem. Das bedeutet, dass der Dateisystemstatus auf dem Datenträger stets konsistent ist. Bei herkömmlichen Dateisystemen werden vorhandene Daten überschrieben. Dies kann dazu führen, dass sich ein Dateisystem in einem undefinierten Status befindet, wenn beispielsweise zwischen dem Zeitpunkt der Zuweisung eines Datenblocks und dem der Verknüpfung mit einem Verzeichnis ein Stromausfall aufgetreten war. Früher wurde dieses Problem durch den Befehl `fsck` behoben. Dieser Befehl überprüfte den Dateisystemstatus auf Konsistenz und versuchte, Dateisystembereiche in einem undefinierten Status zu reparieren. Diese Methode verursachte Systemadministratoren viele Unannehmlichkeiten und bot nicht die Garantie, alle Probleme zu beheben. Zuletzt wurden Dateisysteme entwickelt, die auf dem Konzept des so genannten *Journaling* beruhen. Beim Journaling werden Aktionen in einem eigenen „Journal“ festgehalten, das im Falle von Systemcrashs sicher eingespielt werden kann. Diese Methode verursachte jedoch unnötigen Datenverarbeitungsaufwand, da Daten zweimal geschrieben werden müssen. Darüber hinaus entstanden oft weitere Probleme, wenn beispielsweise das Journal nicht fehlerfrei gelesen werden konnte.

Transaktionale Dateisysteme verwalten Daten mithilfe der so genannten *Copy on Write-Semantik*. Bei diesem Verfahren werden niemals Daten überschrieben, und jegliche Folge von Vorgängen wird entweder vollständig ausgeführt oder ganz ignoriert. Das bedeutet, dass ein Dateisystem bei diesem Verfahren durch Stromausfälle oder Systemabstürze grundsätzlich nicht beschädigt werden kann. Deswegen ist ein Befehl der Art `fsck` nicht mehr erforderlich. Obwohl ganz zuletzt gespeicherte Daten unter Umständen verloren gehen können, bleibt das Dateisystem an sich stets konsistent. Darüber hinaus werden (mithilfe des `O_DSYNC`-Flags geschriebene) synchrone Daten stets vor der Rückkehr geschrieben, sodass sie niemals verloren gehen.

Prüfsummen und Daten mit Selbstheilungsfunktion

In ZFS werden alle Daten und Metadaten mit Prüfsummen versehen, die durch einen benutzerspezifisch auswählbaren Algorithmus berechnet werden. Wegen der zusätzlichen Schicht der Datenträgerverwaltung und des grundsätzlichen Aufbaus herkömmlicher

Dateisysteme führen solche Systeme mit Prüfsummenfunktionalität die Prüfsummenberechnung datenblockweise aus. Aufgrund des herkömmlichen Designs dieser Dateisysteme kann es vorkommen, dass bestimmte Fehlersituationen wie z. B. das Speichern eines Datenblocks an einem falschen Speicherort zu Daten mit korrekten Prüfsummen führen, obwohl die Daten tatsächlich fehlerhaft sind. ZFS-Prüfsummen werden so gespeichert, dass solche Fehlersituationen erkannt und der ordnungsgemäße Zustand des Dateisystems wiederhergestellt werden kann. Alle Prüfsummenberechnungen und Datenwiederherstellungen finden auf Dateisystemebene statt und sind so für Anwendungsprogramme sichtbar.

Darüber hinaus bietet ZFS Selbstheilungsfunktionen für Daten. ZFS unterstützt Speicher-Pools mit unterschiedlichen Stufen der Datenredundanz einschließlich Datenspiegelung und eine RAID-5-Variante. Bei Erkennung beschädigter Datenblöcke ruft ZFS die unbeschädigten Daten von einer redundanten Kopie ab und repariert die beschädigten Daten, indem es diese mit der unbeschädigten Kopie ersetzt.

Konkurrenzlose Skalierbarkeit

ZFS wurde von Grund auf mit dem Ziel entwickelt, eine für Dateisysteme bislang unerreichte Skalierbarkeit bereitzustellen. Das Dateisystem beruht auf der 128-Bit-Architektur, was die Speicherung von 256 Billiarden Zettabyte Daten ermöglicht. Alle Metadata werden dynamisch zugewiesen. Damit entfällt die Notwendigkeit, Inodes vorher zuweisen bzw. die Skalierbarkeit eines Dateisystems bei der ersten Erstellung anderweitig einschränken zu müssen. Alle Algorithmen wurden im Hinblick auf eine bestmögliche Skalierbarkeit entwickelt. Verzeichnisse können bis zu 2^{48} (256 Billionen) Einträge enthalten, und für die Anzahl der Dateisysteme bzw. die Anzahl der in einem Dateisystem enthaltenen Dateien bestehen keine Beschränkungen.

ZFS-Snapshots

Ein *Snapshot* ist eine schreibgeschützte Kopie eines Dateisystems bzw. Volumes. Snapshots können schnell und einfach erstellt werden. Anfänglich belegen Snapshots keinen zusätzlichen Speicherplatz im Pool.

Mit der Änderung von Daten innerhalb des aktiven Datasets belegt der Snapshot mehr Speicherplatz, da weiterhin Verweise auf die älteren Daten gespeichert sind. So verhindern Snapshots, dass der von den Daten belegte Speicherplatz für den Pool freigegeben wird.

Vereinfachte Administration

Eine der wichtigsten Eigenschaften von ZFS ist das erheblich vereinfachte Administrationsmodell. Durch hierarchische Dateisystemstrukturen, Eigenschaftsvererbung sowie automatische Verwaltung von Einhängepunkten und NFS-Netzwerksemantik können in

ZFS Dateisysteme auf einfache Weise erstellt und verwaltet werden, ohne dass dafür mehrere Befehle erforderlich sind oder Konfigurationsdateien bearbeitet werden müssen. Sie können auf einfache Weise Kontingente bzw. Reservierungen vornehmen, Datenkomprimierung aktivieren/deaktivieren oder Einhängepunkte für mehrere Dateisysteme mit einem einzigen Befehl verwalten. Datenspeichergeräte können überprüft bzw. repariert werden, ohne dass dafür Kenntnisse über einen separaten Volume Manager-Befehlssatz erforderlich sind. Sie können eine unbegrenzte Anzahl sofortiger Dateisystem-Snapshots erstellen sowie von einzelnen Dateisystemen Sicherungskopien erstellen und diese Dateisysteme aus Sicherungskopien wiederherstellen.

ZFS verwaltet Dateisysteme über eine Hierarchie, die eine vereinfachte Verwaltung von Eigenschaften wie z. B. Kontingenten, Reservierungen, Komprimierung und Einhängepunkten ermöglicht In diesem Modell werden Dateisysteme zur zentralen Verwaltungsstelle. Dateisysteme sind (vergleichbar mit einem neuen Verzeichnis) an sich sehr kostengünstig. Deswegen sollten Sie für jeden Benutzer und Arbeitsbereich sowie für jedes Projekt ein neues Dateisystem erstellen. Dieses Design ermöglicht sehr fein abgestimmte Verwaltungsknoten.

In ZFS verwendete Begriffe

In diesem Abschnitt werden die im vorliegenden Dokument verwendeten Begriffe erläutert:

Alternative Boot-Umgebung	Eine mit dem Befehl `lucreate` erstellte und möglicherweise mit dem Befehl `luupgrade` aktualisierte Boot-Umgebung, die derzeit weder die aktive noch die primäre Boot-Umgebung darstellt. Die alternative Boot-Umgebung (ABU) kann mithilfe des Befehls `luactivate` zur primären Boot-Umgebung (PBU) ernannt werden.
checksum	Mithilfe eines 256-Bit-Hash-Algorithmus verschlüsselte Daten eines Dateisystemblocks. Prüfsummenalgorithmen reichen vom einfachen und schnellen Fletcher2-Algorithmus (die Standardeinstellung) bis hin zu Hash-Algorithmen mit starker Verschlüsselung wie z. B. SHA256.
Klon	Ein Dateisystem, dessen anfänglicher Inhalt identisch mit dem Inhalt eines Snapshots ist. Informationen zu Klonen finden Sie in „Überblick über ZFS-Klone“ auf Seite 223.
Dataset	Ein allgemeiner Name für die folgenden ZFS-Entitäten: Klone, Dateisysteme, Snapshots oder Volumes (Datenträger). Jedes Dataset wird im ZFS-Namensraum durch einen eindeutigen Namen identifiziert. Datasets werden mithilfe des folgenden Formats benannt:

Pool/*Pfad*[*@Snapshot*]

pool	Der Name des Speicher-Pools, der das Dataset enthält
Pfad	Ein durch Schrägstriche begrenzter Pfadname für das Dataset-Objekt
Snapshot	Eine optionale Komponente, die den Snapshot eines Datasets identifiziert

Weitere Informationen zu Datasets finden Sie in Kapitel 6, „Verwalten von ZFS-Dateisystemen“.

Dateisystem — Ein ZFS-Dataset der Art `filesystem`, der im Standardnamensraum des Systems eingehängt ist und sich wie jedes andere Dateisystem verhält.

Weitere Informationen zu Dateisystemen finden Sie in Kapitel 6, „Verwalten von ZFS-Dateisystemen“.

Mirror — Eine Konfiguration mit virtuellen Geräten, in der identische Kopien von Daten auf zwei oder mehr Festplatten gespeichert werden. Wenn eine Festplatte im Datenspiegelungssystem ausfällt, stellt eine andere Festplatte dieses Systems die gleichen Daten bereit.

Pool — Eine logische Gruppe von Geräten, die das Layout und die physischen Merkmale des verfügbaren Speichers beschreibt. Datensätzen wird Speicher aus einem Pool zugewiesen.

Weitere Informationen zu Speicher-Pools finden Sie in Kapitel 4, „Verwalten von ZFS-Speicher-Pools“.

Primäre Boot-Umgebung — Eine Boot-Umgebung, aus der mit dem Befehl `lucreate` die alternative Boot-Umgebung erstellt wird. Standardmäßig ist die primäre Boot-Umgebung (PBU) die aktuelle Boot-Umgebung. Dieses Standardverhalten kann mit `lucreate` und der Option `-s` außer Kraft gesetzt werden.

RAID-Z — Ein virtuelles Gerät, das Daten und Paritäten auf mehreren Festplatten speichert (vergleichbar mit RAID-5). Weitere Informationen zu RAID-Z finden Sie unter „Speicher-Pools mit RAID-Z-Konfiguration“ auf Seite 64.

Resilvering — Den Vorgang des Übertragens von Daten von einem Datenspeichergerät auf ein anderes Gerät nennt man *Resilvering*. Beispiel: Wenn eine Speicherkomponente ersetzt oder offline genommen wird, werden die Daten von der

	aktuellen Spiegelkomponente in die neu wiederhergestellte Spiegelkomponente kopiert. Dieser Vorgang heißt in herkömmlichen Datenträgermanagement-Produkten *Neusynchronisierung der Datenspiegelung*. Weitere Informationen zum ZFS-Resilvering finden Sie in „Anzeigen des Resilvering-Status“ auf Seite 311.
snapshot	Ein schreibgeschütztes Abbild eines Dateisystems oder Volumes zu einem bestimmten Zeitpunkt. Weitere Informationen zu Snapshots finden Sie in „Überblick über ZFS-Snapshots“ auf Seite 217.
Virtuelles Gerät	Ein logisches Gerät in einem Pool. Dies kann ein physisches Datenspeichergerät, eine Datei oder ein Geräteverbund sein. Weitere Informationen zu virtuellen Geräten finden Sie unter „Anzeigen von Informationen zu virtuellen Geräten in Storage-Pools“ auf Seite 71.
volume	Ein Dataset, mit dem ein physisches Datenspeichergerät emuliert wird. Beispielsweise lässt sich ein ZFS-Volume als Swap-Gerät erstellen. Weitere Informationen zu ZFS-Volumes finden Sie unter „ZFS-Volumes“ auf Seite 279.

Konventionen für das Benennen von ZFS-Komponenten

Jede ZFS-Komponente muss nach den folgenden Regeln benannt werden:

- Leere Komponenten sind unzulässig
- Eine Komponente darf nur alphanumerische Zeichen sowie die folgenden vier Sonderzeichen enthalten:
  - Unterstrich (_)
  - Bindestrich (-)
  - Doppelpunkt (:)
  - Punkt (.)
- Pool-Namen müssen mit einem Buchstaben beginnen. Dabei gelten folgende Ausnahmen:
  - Die Zeichenfolge `c[0-9]` ist am Namensbeginn nicht erlaubt.
  - Der Name `log` ist reserviert.

- Namen, die mit `mirror`, `raidz` oder `spare` beginnen, sind nicht erlaubt, da diese Namen reserviert sind.

Darüber hinaus dürfen Pool-Namen kein Prozentzeichen (`%`) enthalten.

- Dataset-Namen müssen mit einem alphanumerischen Zeichen beginnen. Dataset-Namen dürfen kein Prozentzeichen (`%`) enthalten.

◆ ◆ ◆ KAPITEL 2

Erste Schritte mit ZFS

Dieses Kapitel enthält schrittweise Anleitungen zum Einrichten einfacher ZFS-Konfigurationen. Wenn Sie dieses Kapitel vollständig durchgearbeitet haben, sollte Ihnen das Funktionsprinzip von ZFS-Befehlen klar sein, und Sie können einfache Pools und Dateisysteme erstellen. Dieses Kapitel ist nicht als allumfassende Informationsquelle gedacht und enthält Verweise auf die nächsten Kapitel, die ausführlichere Informationen enthalten.

Dieses Kapitel enthält die folgenden Abschnitte:

Hardware- und Softwarevoraussetzungen und -Empfehlungen für ZFS

Überprüfen Sie vor dem Arbeiten mit der ZFS-Software die folgenden Hardware- und Software-Voraussetzungen und -Empfehlungen:

- Ein SPARC®- oder x86-System, auf dem Solaris 10 6/06 oder höher installiert ist.
- Die Mindestkapazität für Festplatten beträgt 128 MB. Der für einen Speicher-Pool mindestens benötigte Speicherplatz beträgt ca. 64 MB.
- Gegenwärtig beträgt der für die Installation eines Solaris-Systems mindestens erforderliche Speicherplatz 768 MB. Zum Erreichen einer optimalen ZFS-Leistung werden jedoch mindestens 1 GB Speicher oder mehr empfohlen.
- Für die Erstellung eines Festplattensystems mit Datenspiegelung werden mehrere Controller empfohlen.

Erstellen eines einfachen ZFS-Dateisystems

Die ZFS-Administration wurde unter Berücksichtigung größtmöglicher Einfachheit entwickelt. Eines der Ziele des ZFS-Designs besteht in der Verringerung der Anzahl der Befehle, die zum Erstellen eines funktionierenden Dateisystems erforderlich sind. Beim Erstellen eines neuen Pools wird automatisch ein neues ZFS-Dateisystem erstellt und eingehängt.

Das folgende Beispiel zeigt, wie mit einem einzigen Befehl einen einfachen gespiegelten Speicher-Pool namens `tank` und ein ZFS-Dateisystem mit dem Namen `tank` erstellt werden können. Es wird angenommen, dass die Datenträger `/dev/dsk/c1t0d0` und `/dev/dsk/c2t0d0` vollständig verfügbar sind.

```
# zpool create tank mirror c1t0d0 c2t0d0
```

Weitere Informationen zu ZFS-Pool-Konfigurationen finden Sie unter „Replikationsfunktionen eines ZFS-Speicher-Pools“ auf Seite 63.

Das neue ZFS-Dateisystem `tank` kann so viel Speicherplatz wie erforderlich belegen und wird automatisch als `/tank` eingehängt.

```
# mkfile 100m /tank/foo
# df -h /tank
Filesystem             size   used  avail capacity  Mounted on
tank                   80G   100M    80G     1%    /tank
```

Innerhalb eines Pools werden Sie höchstwahrscheinlich noch weitere Dateisysteme erstellen. Dateisysteme bieten Administrationsschnittstellen, mit deren Hilfe verschiedene Datengruppen innerhalb des gleichen Pools verwaltet werden können.

Das folgende Beispiel zeigt die Erstellung eines Dateisystems namens `fs` im Speicher-Pool `tank`.

```
# zfs create tank/fs
```

Das neue ZFS-Dateisystem `tank/fs` kann so viel Speicherplatz wie erforderlich belegen und wird automatisch als `/tank/fs` eingehängt.

```
# mkfile 100m /tank/fs/foo
# df -h /tank/fs
Filesystem             size   used  avail capacity  Mounted on
tank/fs                 80G   100M    80G     1%    /tank/fs
```

In den meisten Fällen werden Sie wahrscheinlich eine Dateisystemhierarchie erstellen, die die Organisationsstruktur Ihrer Einrichtung widerspiegelt. Weitere Informationen zum Erstellen einer Hierarchie von ZFS-Dateisystemen finden Sie unter „Erstellen einer ZFS-Dateisystemhierarchie“ auf Seite 50.

Erstellen eines ZFS-Speicher-Pools

Das vorherige Beispiel verdeutlicht die Einfachheit von ZFS. Im Folgenden wird ein umfassenderes Beispiel vorgestellt, das einer praktischen Anwendung schon relativ nahe kommt. Zuerst müssen Sie Speicheranforderungen definieren und einen Speicher-Pool erstellen. Der Pool beschreibt die physischen Eigenschaften des Speicherbedarfs und muss vor dem Erstellen von Dateisystemen angelegt werden.

▼ So definieren Sie Speicheranforderungen für einen ZFS-Speicher-Pool

1 Legen Sie verfügbare Datenspeichergeräte fest.

Vor dem Erstellen eines Speicher-Pools müssen Sie festlegen, auf welchen Datenspeichergeräten Daten gespeichert werden sollen. Solche Datenspeichergeräte müssen eine Kapazität von mindestens 128 MB besitzen und dürfen nicht von anderen Bereichen des Betriebssystems verwendet werden. Die Datenspeichergeräte können einzelne Bereiche eines vorformatierten Datenträgers oder gesamte Datenträger sein, die ZFS als einzelnen größeren Bereich formatiert.

Bei dem Speicherbeispiel unter „So erstellen Sie einen ZFS-Speicher-Pool“ auf Seite 50 wird angenommen, dass die gesamten Datenträger `/dev/dsk/c1t0d0` und `/dev/dsk/c2t0d0` verfügbar sind.

Weitere Informationen zu Datenträgern und ihrer Verwendung und Bezeichnung finden Sie unter „Verwenden von Datenträgern in einem ZFS-Speicher-Pool“ auf Seite 59.

2 Wählen Sie die Art der Datenreplikation.

ZFS unterstützt mehrere Datenreplikationsverfahren. Diese legen fest, gegen welche Hardware-Ausfälle ein Pool gewappnet ist. ZFS unterstützt nicht redundante Konfigurationen (Stripes) sowie Datenspiegelung und RAID-Z (eine RAID-5-Variante).

Beim Speicherbeispiel unter „So erstellen Sie einen ZFS-Speicher-Pool“ auf Seite 50 wird ein einfache Datenspiegelung zweier verfügbarer Festplatten eingesetzt.

Weitere Informationen zur ZFS-Datenreplikation finden Sie unter „Replikationsfunktionen eines ZFS-Speicher-Pools“ auf Seite 63.

▼ So erstellen Sie einen ZFS-Speicher-Pool

1 Melden Sie sich als Root oder in einer gleichberechtigten Rolle (mithilfe des entsprechenden Zugriffsrechtssprofils von ZFS) an.

Weitere Informationen zu ZFS-Zugriffsrechtsprofilen finden Sie unter „ZFS-Zugriffsrechtsprofile“ auf Seite 290.

2 Wählen Sie einen Pool-Namen aus.

Dieser Name dient zur Identifizierung des Speicher-Pools bei Verwendung der Befehle `zpool` bzw. `zfs`. Bei den meisten Systemen muss nur ein einziger Pool erstellt werden, deswegen können Sie einen beliebigen Namen auswählen. Beachten Sie dabei jedoch die unter „Konventionen für das Benennen von ZFS-Komponenten“ auf Seite 45 aufgeführten Einschränkungen.

3 Erstellen Sie den Pool.

Sie können beispielsweise einen Pool namens `tank` mit Datenspiegelung erstellen.

```
# zpool create tank mirror c1t0d0 c2t0d0
```

Wenn Datenspeichergeräte andere Dateisysteme enthalten oder anderweitig belegt sind, kann der Befehl den Pool nicht erstellen.

Weitere Informationen zum Erstellen von Speicher-Pools finden Sie unter „Erstellen eines ZFS-Speicher-Pools“ auf Seite 66.

Weitere Informationen darüber, wie Sie herausfinden können, ob Datenspeichergeräte belegt sind oder nicht, finden Sie unter „Erkennen belegter Geräte“ auf Seite 73.

4 Überprüfen Sie die Ergebnisse.

Mit dem Befehl `zpool list` können Sie überprüfen, ob der Pool erfolgreich erstellt wurde.

```
# zpool list
NAME                    SIZE    USED   AVAIL    CAP  HEALTH     ALTROOT
tank                     80G    137K     80G     0%  ONLINE     -
```

Weitere Informationen zum Anzeigen des Pool-Status finden Sie unter „Abfragen des Status von ZFS-Speicher-Pools“ auf Seite 95.

Erstellen einer ZFS-Dateisystemhierarchie

Nach dem Erstellen eines Speicher-Pools zum Speichern von Daten können Sie mit dem Anlegen der Dateisystemhierarchie beginnen. Mit Hierarchien können Informationen auf einfache und gleichzeitig leistungsfähige Weise organisiert werden. Jedem, der bereits mit einem Dateisystem gearbeitet hat, sind diese vertraut.

In ZFS können Dateisysteme in beliebigen Hierarchien organisiert werden, wobei jedes Dateisystem nur ein übergeordnetes System besitzt. Der Pool-Name ist stets der Name der obersten Hierarchieebene. ZFS erweitert diese Hierarchie durch Eigenschaftsvererbung, sodass gemeinsame Eigenschaften schnell und einfach an gesamten Dateisystemhierarchien gesetzt werden können.

▼ So legen Sie eine ZFS-Dateisystemhierarchie fest

1 Wählen Sie die Dateisystemgranularität aus.

ZFS-Dateisysteme spielen für die Administration eine zentrale Rolle. Sie sind kompakt und können schnell erstellt werden. Eine gute Faustregel ist die Erstellung eines Dateisystems pro Benutzer bzw. Projekt, da ein solches Modell die benutzer- bzw. projektweise Kontrolle von Eigenschaften, Snapshots und Sicherungskopien ermöglicht.

Im Beispiel unter „So erstellen Sie ZFS-Dateisysteme“ auf Seite 52 werden zwei ZFS-Dateisysteme (`bonwick` und `billm`) erstellt.

Weitere Informationen zur Verwaltung von Dateisystemen finden Sie in Kapitel 6, „Verwalten von ZFS-Dateisystemen“.

2 Fassen Sie ähnliche Dateisysteme zu Gruppen zusammen.

In ZFS können Dateisysteme in Hierarchien organisiert werden, was die Gruppierung ähnlicher Dateisysteme ermöglicht. Dieses Modell bietet eine zentrale Administrationsschnittstelle zur Kontrolle von Eigenschaften und Verwaltung von Dateisystemen. Ähnliche Dateisysteme sollten unter einem gemeinsamen Namen erstellt werden.

Im Beispiel unter „So erstellen Sie ZFS-Dateisysteme“ auf Seite 52 befinden sich die beiden Dateisysteme in einem Dateisystem namens `home`.

3 Wählen Sie die Dateisystemeigenschaften.

Die meisten Charakteristika von Dateisystemen werden mit einfachen Eigenschaften eingestellt. Diese Eigenschaften legen verschiedene Verhaltensaspekte fest, so z. B. den Einhängepunkt eines Dateisystems, ob und wie sie über Netzwerk zugänglich sind, ob sie Datenkomprimierung verwenden und ob für ein Dateisystem Kontingente gelten.

Im Beispiel unter „So erstellen Sie ZFS-Dateisysteme“ auf Seite 52 sind alle home-Verzeichnisse als `/export/zfs/`*Benutzer* eingehängt, mithilfe von NFS über Netzwerk zugänglich und nutzen Datenkomprimierung. Darüber hinaus gilt für `bonwick` ein Kontingent von 10 GB.

Weitere Informationen zu Eigenschaften finden Sie unter „ZFS-Eigenschaften“ auf Seite 179.

▼ So erstellen Sie ZFS-Dateisysteme

1 **Melden Sie sich als Root oder in einer gleichberechtigten Rolle (mithilfe des entsprechenden Zugriffsrechtssprofils von ZFS) an.**

Weitere Informationen zu ZFS-Zugriffsrechtsprofilen finden Sie unter „ZFS-Zugriffsrechtsprofile“ auf Seite 290.

2 **Erstellen Sie die gewünschte Hierarchie.**

In diesem Beispiel wird ein Dateisystem erstellt, das als Container für untergeordnete Dateisysteme dienen soll.

```
# zfs create tank/home
```

Als Nächstes werden einzelne Dateisysteme im Dateisystem `home` des Pools `tank` gruppiert.

3 **Legen Sie die vererbten Eigenschaften fest.**

Nach dem Erstellen der Dateisystemhierarchie sollten Eigenschaften festgelegt werdem, die für alle Benutzer gleich sein sollen:

```
# zfs set mountpoint=/export/zfs tank/home
# zfs set sharenfs=on tank/home
# zfs set compression=on tank/home
# zfs get compression tank/home
NAME             PROPERTY       VALUE                      SOURCE
tank/home        compression    on                         local
```

Jetzt ist eine neue Funktion verfügbar, mit der Sie beim Erstellen eines Dateisystems Dateisystemeigenschaften festlegen können. Beispiel:

```
# zfs create -o mountpoint=/export/zfs -o sharenfs=on -o compression=on tank/home
```

Weitere Informationen zu Eigenschaften und deren Vererbung finden Sie unter „ZFS-Eigenschaften“ auf Seite 179.

4 **Erstellen Sie die einzelnen untergeordneten Dateisysteme.**

Bitte beachten Sie, dass diese Dateisysteme bereits auf der `home`-Ebene hätten erstellt (und die Eigenschaften eingestellt) werden können. Alle Eigenschaften können dynamisch geändert werden, wenn Dateisysteme in Betrieb sind.

```
# zfs create tank/home/bonwick
# zfs create tank/home/billm
```

Diese Dateisysteme erben ihre Eigenschaftswerte von ihrem übergeordneten Dateisystem. Deswegen werden sie automatisch als `/export/zfs/`*Benutzer* eingehängt und mithilfe von NFS über Netzwerk zugänglich. Sie brauchen die Datei `/etc/vfstab` bzw. `/etc/dfs/dfstab` nicht zu bearbeiten.

Weitere Informationen zum Erstellen von Dateisystemen finden Sie unter „Erstellen eines ZFS-Dateisystems“ auf Seite 176.

Weitere Informationen zum Einhängen von Dateisystemen und Freigeben von Dateisystemen für den Netzwerkzugang finden Sie in „Einhängen und Freigeben von ZFS-Dateisystemen“ auf Seite 201.

5 Legen Sie die dateisystemspezifischen Eigenschaften fest.

In diesem Beispiel gilt für den Benutzer `bonwick` ein Kontingent von 10 GB. Diese Eigenschaft beschränkt unabhängig davon, wieviel Speicherplatz im Pool verfügbar ist, den durch diesen Benutzer zu belegenden Speicherplatz.

```
# zfs set quota=10G tank/home/bonwick
```

6 Überprüfen Sie die Ergebnisse.

Mit dem Befehl `zfs list` können verfügbare Dateisysteminformationen angezeigt werden:

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank                  92.0K  67.0G   9.5K  /tank
tank/home             24.0K  67.0G     8K  /export/zfs
tank/home/billm          8K  67.0G     8K  /export/zfs/billm
tank/home/bonwick        8K  10.0G     8K  /export/zfs/bonwick
```

Beachten Sie, dass der Benutzer `bonwick` maximal 10 GB Speicherplatz belegen darf, der Benutzer `billm` dagegen die gesamte Pool-Kapazität (67 GB) nutzen kann.

Weitere Informationen zum Anzeigen des Dateisystemstatus finden Sie unter „Abfragen von ZFS-Dateisysteminformationen“ auf Seite 194.

Weitere Informationen zur Verwendung und Berechnung von Speicherplatz finden Sie unter „ZFS-Speicherplatzverwaltung“ auf Seite 56.

◆ ◆ ◆ KAPITEL 3

Unterschiede zwischen ZFS und herkömmlichen Dateisystemen

Dieses Kapitel erläutert einige wichtige Unterschiede zwischen ZFS und herkömmlichen Dateisystemen. Durch ein Verständnis dieser Hauptunterschiede werden Unklarheiten bei der Zusammenarbeit von herkömmlichen Tools mit ZFS ausgeräumt.

Dieses Kapitel enthält die folgenden Abschnitte:

Granularität von ZFS-Dateisystemen

Früher waren Dateisysteme auf ein Datenspeichergerät beschränkt, womit die Kapazität dieser Dateisysteme auch automatisch auf die Kapazität des betreffenden Datenträgers beschränkt war. Das Erstellen und Neuerstellen herkömmlicher Dateisysteme aufgrund von Kapazitätsbeschränkungen ist zeitaufwändig und in einigen Fällen kompliziert. Dabei half spezielle Software zur Verwaltung herkömmlicher Dateisysteme.

Da ZFS-Dateisysteme nicht auf spezielle Geräte beschränkt sind, können sie ähnlich wie Verzeichnisse schnell und einfach erstellt werden. ZFS-Dateisysteme werden innerhalb der für einen Speicher-Pool verfügbaren Kapazität automatisch vergrößert.

Anstatt ein Dateisystem (z. B. `/export/home`) zum Verwalten von Benutzerverzeichnissen zu erstellen, können Sie pro Benutzer ein Dateisystem anlegen. Darüber hinaus besitzt ZFS eine Dateisystemhierarchie. So können Sie viele Dateisystem auf einfache Weise einrichten und verwalten, indem Sie Eigenschaften setzen, die von Dateisystemen innerhalb einer Hierarchie geerbt werden.

Ein Beispiel zum Erstellen einer Dateisystemhierarchie finden Sie unter „Erstellen einer ZFS-Dateisystemhierarchie“ auf Seite 50.

ZFS-Speicherplatzverwaltung

ZFS beruht auf dem Konzept der Datenspeicherung in Pools. Im Gegensatz zu herkömmlichen, physischen Datenträgern direkt zugewiesenen Dateisystemen teilen sich ZFS-Dateisysteme in einem Pool den in diesem Pool verfügbaren Speicherplatz. Somit kann sich der Wert des von Dienstprogrammen wie z. B. `df` gemeldeten freien Speicherplatzes auch dann ändern, wenn das interessierende Dateisystem inaktiv ist, da andere Dateisysteme im Pool Speicherplatz belegen bzw. freigeben. Bitte beachten Sie, dass die maximal mögliche Dateisystemkapazität durch die Zuteilung von Kontingenten beschränkt werden kann. Weitere Informationen zu Kontingenten finden Sie unter „Setzen von Kontingenten für ZFS-Dateisysteme“ auf Seite 209. Durch Reservierungen kann einem Dateisystem Speicherplatz garantiert werden. Weitere Informationen zu Reservierungen finden Sie unter „Setzen von Reservierungen für ZFS-Dateisysteme“ auf Seite 213. Dieses Modell gleicht dem NFS-Modell sehr; dort werden mehrere Verzeichnisse vom gleichen Dateisystem (z. B. `/home`) eingehängt.

Alle Metadaten werden in ZFS dynamisch zugewiesen. In herkömmlichen Dateisystemen erfolgt die Zuweisung von Metadaten meist vorher. Deswegen muss für diese Metadaten schon beim Erstellen des Dateisystems zusätzlicher Speicherplatz reserviert werden. Dieses Verhalten bedeutet weiterhin, dass die mögliche Gesamtdateianzahl eines Dateisystems schon vorher festgelegt ist. Da ZFS Metadaten nach Bedarf zuweist, muss vorher kein zusätzlicher Speicherplatz reserviert werden, und die Dateianzahl wird lediglich durch den verfügbaren Speicherplatz begrenzt. Die Ausgabe des Befehls `df -g` ist bei ZFS anders als bei herkömmlichen Dateisystemen zu interpretieren. Die unter `total files` ausgegebene Dateinanzahl ist lediglich ein Schätzwert, der auf dem Betrag des im Pool verfügbaren Speicherplatzes beruht.

ZFS ist ein transaktionales Dateisystem. Das bedeutet, dass die meisten Dateisystemmodifikationen in Transaktionsgruppen zusammengefasst und asynchron auf dem Datenträger ausgeführt werden. Diese Modifikationen gelten solange als *anstehende Änderungen*, bis sie auf dem Datenträger abgeschlossen sind. Der von einer Datei bzw. einem Dateisystem belegte, verfügbare und referenzierte Speicherplatz berücksichtigt keinen durch solche anstehenden Änderungen belegten Speicherplatz. Anstehende Änderungen werden im Allgemeinen innerhalb weniger Sekunden abgeschlossen. Auch nach Abschluss eines Schreibvorgangs auf dem Datenträger durch `fsync(3c)` oder `O_SYNC` werden die Informationen zum belegten Speicherplatz nicht notwendigerweise sofort aktualisiert.

Zusätzliche Informationen über den von `du` und `df` gemeldeten ZFS-Speicherplatzverbrauch finden Sie über den folgenden Link:

`http://opensolaris.org/os/community/zfs/faq/#whydusize`

Verhalten bei ungenügendem Speicherplatz

Snapshots von Dateisystemen sind in ZFS äußerst einfach und ohne hohen Aufwand zu erstellen. Snapshots kommen in den meisten ZFS-Umgebungen sehr häufig vor. Weitere Informationen zu ZFS-Snapshots finden Sie in Kapitel 7, „Arbeiten mit ZFS-Snapshots und -Klonen“.

Das Vorhandensein von Snapshots kann beim Freigeben von Speicherplatz ein unvorgesehenes Verhalten hervorrufen. Normalerweise können Sie mit den entsprechenden Zugriffsrechten eine Datei aus einem vollständigen Dateisystem löschen, und durch diese Aktion wird im Dateisystem mehr Speicherplatz verfügbar. Wenn die zu löschende Datei jedoch in einem Snapshot eines Dateisystems vorhanden ist, wird durch das Löschen dieser Datei kein Speicherplatz freigegeben. Die von dieser Datei belegten Datenblöcke werden weiterhin vom Snapshot referenziert.

Deswegen kann infolge des Löschens einer Datei mehr Speicherplatz belegt werden, da eine neue Verzeichnisversion erstellt werden muss, die den neuen Namensplatzstatus widerspiegelt. Dadurch können beim Löschen von Dateien unvorhergesehene `ENOSPC`- oder `EDQUOT`-Fehler auftreten.

Einhängen von ZFS-Dateisystemen

ZFS soll die Komplexität verringern und die Administration vereinfachen. Bei herkömmlichen Dateisystemen müssen Sie beispielsweise bei jedem Hinzufügen eines neuen Dateisystems die Datei `/etc/vfstab` entsprechend ändern. In ZFS ist das nicht mehr erforderlich, da Dateisysteme je nach den Dataset-Eigenschaften automatisch ein- und ausgehängt werden. ZFS-Einträge müssen also nicht in der Datei `/etc/vfstab` verwaltet werden.

Weitere Informationen zum Einhängen von Dateisystemen und Freigeben von ZFS-Dateisystemen für den Netzwerkzugang finden Sie unter „Einhängen und Freigeben von ZFS-Dateisystemen“ auf Seite 201.

Herkömmliche Datenträgerverwaltung

Wie bereits unter „Speicher-Pools in ZFS“ auf Seite 40 erwähnt, ist für ZFS kein spezielles Dienstprogramm zur Datenträgerverwaltung erforderlich. ZFS arbeitet mit im raw-Modus betriebenen Geräten. Deswegen können Speicher-Pools erstellt werden, die aus (software- oder hardwarebasierten) Logical Volumes bestehen. Diese Konfiguration wird jedoch nicht empfohlen, da ZFS am besten mit im raw-Modus betriebenen Geräten arbeitet. Die Verwendung von Logical Volumes kann sich negativ auf die Leistung und die Zuverlässigkeit auswirken und sollte deswegen vermieden werden.

Neues Solaris-Modell für Zugriffssteuerungslisten

Frühere Versionen des Betriebssystems Solaris unterstützten eine auf der POSIX-Spezifikation beruhende Implementierung von Zugriffssteuerungslisten. POSIX-basierte Zugriffssteuerungslisten dienen zum Schutz von UFS-Dateien. Zum Schutz von ZFS-Dateien wird ein neues, auf der NFSv4-Spezifikation beruhendes Zugriffssteuerungslistenmodell verwendet.

Die Hauptunterschiede des neuen Solaris-Zugriffssteuerungslistenmodells bestehen in Folgendem:

- Das neue Modell basiert auf der NFSv4-Spezifikation und ähnelt NT-Zugriffssteuerungslistenmodellen,
- enthält einen feiner abstimmbaren Satz an Zugriffsrechten,
- wird mit den Befehlen `chmod` und `ls` anstatt `setfacl` und `getfacl` eingestellt und angezeigt,
- enthält eine reichhaltigere Vererbungssemantik, um festlegen zu können, wie Zugriffsrechte von über- und untergeordneten Verzeichnissen geregelt werden, usw.

Weitere Informationen zu Zugriffssteuerungslisten mit ZFS-Dateien finden Sie in Kapitel 8, „Schützen von ZFS-Dateien mit Zugriffssteuerungslisten“.

◆ ◆ ◆ KAPITEL 4

Verwalten von ZFS-Speicher-Pools

Dieses Kapitel erläutert, wie Sie ZFS-Speicher-Pools erstellen und verwalten können.

Dieses Kapitel enthält die folgenden Abschnitte:

Komponenten eines ZFS-Speicher-Pools

Die folgenden Abschnitte enthalten ausführliche Informationen zu den folgenden Komponenten eines Speicher-Pools:

Verwenden von Datenträgern in einem ZFS-Speicher-Pool

Das grundlegende Element eines Speicher-Pools ist eine physische Komponente zur Datenspeicherung. Zur physischen Datenspeicherung können alle Blockeinheiten mit mindestens 128 MB Speicherkapazität verwendet werden. Normalerweise handelt es sich dabei um Festplatten, die vom System im Verzeichnis `/dev/dsk` erkannt werden.

Ein Datenspeichergerät kann eine gesamte Festplatte (z. B. `c1t0d0`) oder ein einzelner Plattenbereich (z. B. `c0t0d0s7`) sein. Es wird empfohlen, als Datenspeichergeräte gesamte Festplatten zu verwenden, da in diesem Fall die betreffende Festplatte nicht besonders formatiert werden muss. ZFS formatiert solche Datenträger mit einem EFI-Label als einzelnen, großen Bereich. In diesem Fall wird die vom Befehl `format` ausgegebene Partitionstabelle in etwa wie folgt angezeigt:

```
Current partition table (original):
Total disk sectors available: 17672849 + 16384 (reserved sectors)

Part      Tag    Flag     First Sector         Size         Last Sector
  0        usr    wm               256       8.43GB          17672849
  1 unassigned    wm                 0          0               0
  2 unassigned    wm                 0          0               0
  3 unassigned    wm                 0          0               0
  4 unassigned    wm                 0          0               0
  5 unassigned    wm                 0          0               0
  6 unassigned    wm                 0          0               0
  8   reserved    wm          17672850       8.00MB          17689233
```

Bei der Verwendung gesamter Festplatten müssen diese der Benennungskonvention `/dev/dsk/cXtXdX` entsprechen. Einige Treiber von Drittanbietern nutzen andere Benennungskonventionen oder hängen Datenträger nicht im Verzeichnis `/dev/dsk`, sondern anderswo ein. Damit solche Datenträger verwendet werden können, müssen Sie diese manuell benennen und für ZFS einen Bereich verfügbar machen.

ZFS verwendet EFI-Label, wenn Sie Speicher-Pools mit gesamten Festplatten erstellen. Weitere Informationen zu EFI-Labels finden Sie unter „EFI Disk Label“ in *System Administration Guide: Devices and File Systems*.

Eine für einen ZFS-Root-Pool gedachte Festplatte muss mit einem SMI-Label, nicht mit einem EFI-Label erstellt werden. Mit dem Befehl `format -e` können Sie eine Festplatte im Nachhinein mit einem SMI-Label versehen.

Datenträger können mit dem vollständigen Pfad (z. B. `/dev/dsk/c1t0d0` oder in einer Kurzschreibweise, die aus dem Gerätenamen im Verzeichnis `/dev/dsk` besteht (z. B. `c1t0d0`), angegeben werden. Im Folgenden sind beispielsweise gültige Datenträgernamen aufgeführt:

- `c1t0d0`
- `/dev/dsk/c1t0d0`
- `c0t0d6s2`
- `/dev/foo/disk`

Die Verwendung gesamter physischer Datenträger ist die einfachste Methode zum Erstellen von ZFS-Speicher-Pools. ZFS-Konfigurationen werden in Sachen Verwaltung, Zuverlässigkeit und Leistung erheblich komplexer, wenn Sie Pools aus Datenträgerbereichen, LUNs in Hardware-RAID-Arrays oder Volumes, die von Software zur Datenträgerverwaltung verfügbar

gemacht werden, aufbauen. Bei der Entscheidung, ob Sie ZFS mit anderen hardware- bzw. softwarebasierten Speicherlösungen konfigurieren, sollten Sie die folgenden Aspekte berücksichtigen:

- Wenn Sie ZFS-Konfigurationen mit LUNs aus Hardware-RAID-Arrays aufbauen, muss Ihnen die Beziehung zwischen den Redundanzfunktionen von ZFS und denen des Arrays klar sein. Bestimmte Konfigurationen bieten ausreichende Redundanz und Leistung, bei anderen braucht dies aber nicht der Fall zu sein.
- Sie können logische Geräte für ZFS mithilfe von Volumes aufbauen, die von Software zur Datenträgerverwaltung wie z. B. Solaris™ Volume Manager (SVM) oder Veritas Volume Manager (VxVM) bereitgestellt werden. Solche Konfigurationen werden jedoch nicht empfohlen. Obwohl ZFS mit solchen Konfigurationen ordnungsgemäß funktioniert, ist die Leistung in diesen Fällen oftmals nicht optimal.

Zusätzliche Informationen zu Empfehlungen für Speicher-Pool-Konfigurationen finden Sie auf der „Best Practices"-Website für ZFS unter:

`http://www.solarisinternals.com/wiki/index.php/ZFS_Best_Practices_Guide`

Datenträger werden nach Pfad und Geräte-ID (falls verfügbar) identifiziert. Mit dieser Methode können Datenspeichergeräte eines Systems neu konfiguriert werden, ohne dass der ZFS-Status aktualisiert werden muss. Beim Umschalten eines Datenträgers zwischen Controller 1 und Controller 2 erkennt ZFS anhand der Geräte-ID, dass der Datenträger verschoben wurde und auf diesen jetzt über Controller 2 zugegriffen werden muss. Geräte-IDs werden von der Firmware eines Laufwerks eindeutig vergeben. Obwohl dies relativ unwahrscheinlich ist, kann es vorkommen, dass durch Firmware-Aktualisierungen Geräte-IDs geändert werden. In solchen Fällen kann ZFS auf das betreffende Gerät immer noch mithilfe des Pfads zugreifen und die gespeicherte Geräte-ID automatisch aktualisieren. Wenn der Pfad und die ID des Gerätes versehentlich geändert werden, müssen Sie den Pool exportieren und wieder importieren, damit es verwendet werden kann.

Verwenden von Bereichen in einem ZFS-Speicher-Pool

Datenträger können mit einem herkömmlichen Solaris VTOC (SMI)-Label benannt werden, wenn Sie Speicher-Pools mit Festplattenbereichen erstellen.

Für einen boot-fähigen ZFS-Root-Pool müssen die Festplatten im Pool Bereiche enthalten und ein SMI-Label haben. Die einfachste Konfiguration würde darin bestehen, die gesamte Festplattenkapazität in Bereich 0 (Slice 0) zu legen und diesen Bereich für den Root-Pool zu verwenden.

Auf einem SPARC-System verfügt eine 72-GB-Festplatte über 68 GB nutzbaren Speicher in Bereich 0, wie in der folgenden Ausgabe von `format` zu sehen ist.

```
# format
.
.
.
Specify disk (enter its number): 4
selecting c1t1d0
partition> p
Current partition table (original):
Total disk cylinders available: 14087 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders         Size            Blocks
  0       root    wm       0 - 14086       68.35GB    (14087/0/0) 143349312
  1 unassigned    wm       0                0         (0/0/0)             0
  2     backup    wm       0 - 14086       68.35GB    (14087/0/0) 143349312
  3 unassigned    wm       0                0         (0/0/0)             0
  4 unassigned    wm       0                0         (0/0/0)             0
  5 unassigned    wm       0                0         (0/0/0)             0
  6 unassigned    wm       0                0         (0/0/0)             0
  7 unassigned    wm       0                0         (0/0/0)             0
```

Auf einem x86-System verfügt eine 72-GB-Festplatte über 68 GB nutzbaren Speicher in Bereich 0, wie in der folgenden Ausgabe von `format` zu sehen ist. Einige Boot-Infomationen sind in Bereich 8 enthalten. Bereich 8 erfordert keine Verwaltung und kann nicht verändert werden.

```
# format
.
.
.
selecting c1t0d0
partition> p
Current partition table (original):
Total disk cylinders available: 49779 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders         Size            Blocks
  0       root    wm       1 - 49778       68.36GB    (49778/0/0) 143360640
  1 unassigned    wu       0                0         (0/0/0)             0
  2     backup    wm       0 - 49778       68.36GB    (49779/0/0) 143363520
  3 unassigned    wu       0                0         (0/0/0)             0
  4 unassigned    wu       0                0         (0/0/0)             0
  5 unassigned    wu       0                0         (0/0/0)             0
  6 unassigned    wu       0                0         (0/0/0)             0
  7 unassigned    wu       0                0         (0/0/0)             0
  8       boot    wu       0 -     0        1.41MB    (1/0/0)          2880
  9 unassigned    wu       0                0         (0/0/0)             0
```

Falls Sie den Einsatz von Bereichen für einen ZFS-Speicher-Pool erwägen, das kein boot-fähiger ZFS-Root-Pool ist, beachten Sie die folgenden Bedingungen, unter welchen die Verwendung von Bereichen notwendig sein kann:

- Das Gerät besitzt keinen Standardnamen.
- Ein Datenträger wird von ZFS und einem anderen Dateisystem wie z. B. UFS gemeinsam genutzt.

Verwenden von Dateien in einem ZFS-Speicher-Pool

In ZFS können Sie UFS-Dateien als virtuelle Geräte im Speicher-Pool verwenden. Dieses Merkmal dient vorrangig zum Testen und Aktivieren einfacher experimenteller Konfigurationen und ist nicht für den Praxiseinsatz unter Produktionsbedingungen gedacht. Dies ist darin begründet, dass **jede Verwendung von Dateien auf der Konsistenz des zugrunde liegenden Dateisystems beruht**. Wenn Sie einen ZFS-Pool mit Dateien aus einem UFS-Dateisystem erstellen, verlassen sie sich stillschweigend darauf, dass UFS ordnungsgemäße und synchrone Semantik gewährleistet.

Dateien können jedoch nützlich sein, wenn Sie ZFS zum ersten Mal testen oder mit komplexeren Strukturen experimentieren, wenn nicht genügend physische Datenspeichergeräte zur Verfügung stehen. Alle Dateien müssen mit dem vollständigen Pfad angegeben werden und mindestens 64 MB groß sein. Beim Entfernen bzw. Umbenennen einer Datei muss der Pool exportiert und neu importiert werden, damit er verwendet werden kann, da ansonsten den betreffenden Dateien keine Geräte-IDs zugewiesen sind, mit deren Hilfe sie erkannt werden können.

Replikationsfunktionen eines ZFS-Speicher-Pools

ZFS bietet in Konfigurationen mit Datenspiegelung und RAID-Z Datenredundanz und Selbstheilungsfunktionen.

- „Speicher-Pools mit Datenspiegelung“ auf Seite 63
- „Speicher-Pools mit RAID-Z-Konfiguration“ auf Seite 64
- „Selbstheilende Daten in einer redundanten Konfiguration“ auf Seite 65
- „Dynamisches Striping in einem Speicher-Pool“ auf Seite 65
- „ZFS-Hybrid-Speicher-Pool“ auf Seite 65

Speicher-Pools mit Datenspiegelung

Für Speicher-Pools mit Datenspiegelung sind mindestens zwei Datenträger erforderlich, auf die optimalerweise von zwei verschiedenen Controllern zugegriffen werden sollte. Für eine Datenspiegelung können viele Datenträger verwendet werden. Darüber hinaus können Sie in einem Pool mehrere Datenspiegelungen erstellen. Eine einfache Datenspiegelungskonfiguration würde im Prinzip ungefähr wie folgt aussehen:

```
mirror c1t0d0 c2t0d0
```

Eine komplexere Datenspiegelungskonfiguration würde im Prinzip ungefähr wie folgt aussehen:

```
mirror c1t0d0 c2t0d0 c3t0d0 mirror c4t0d0 c5t0d0 c6t0d0
```

Informationen zum Erstellen von Speicher-Pools mit Datenspiegelung finden Sie unter „Erstellen eines Speicher-Pools mit Datenspiegelung“ auf Seite 67.

Speicher-Pools mit RAID-Z-Konfiguration

Neben Speicher-Pools mit Datenspiegelung bietet ZFS RAID-Z-Konfigurationen mit Fehlertoleranzen einfacher oder doppelter Parität. RAID-Z mit einfacher Parität ist mit RAID-5 vergleichbar. RAID-Z mit doppelter Parität ähnelt RAID-6.

Alle herkömmlichen RAID-5-Algorithmen (z. B. RAID-4, RAID-6, RDP und EVEN-ODD) besitzen ein Problem, das als sog. „RAID-5 Write Hole“ bekannt ist. Wenn nur ein Teil eines RAID-5-Stripe geschrieben wird und ein Stromausfall auftritt, bevor alle Datenblöcke erfolgreich auf den Datenträger geschrieben wurden, ist die Parität nicht mehr mit den Daten synchronisiert und deswegen dauerhaft nutzlos, es sei denn, ein nachfolgender vollständiger Stripe-Schreibvorgang überschreibt sie wieder. Bei RAID-Z verwendet ZFS RAID-Stripes veränderlicher Länge, sodass alle Schreibvorgänge vollständige Stripes speichern. Dieses Design ist nur möglich, weil ZFS die Verwaltung von Dateisystemen und Datenspeichergeräten so integriert, dass die Metadaten eines Dateisystems genügend Informationen zum zugrunde liegenden Datenredundanzmodell besitzen und sie somit RAID-Stripes veränderlicher Länge verarbeiten können. RAID-Z ist die weltweit erste nur auf Software basierte Lösung, die das RAID-5 Write Hole eliminiert.

Eine RAID-Z-Konfiguration mit N Datenträgern der Kapazität X mit P Paritätsdatenträgern kann ca. (N-P)*X Byte speichern und ist gegen einen Ausfall von P Geräten ohne Gefährdung der Datenintegrität geschützt. Für eine RAID-Z-Konfiguration mit einfacher Parität sind mindestens zwei, für eine RAID-Z-Konfiguration mit doppelter Parität mindestens drei Datenträger erforderlich. Wenn sich beispielsweise in einer RAID-Z-Konfiguration mit einfacher Parität drei Datenträger befinden, belegen Paritätsdaten einen Speicherplatz, der der Kapazität eines der drei Datenträger entspricht. Zum Erstellen einer RAID-Z-Konfiguration ist darüber hinaus keine spezielle Hardware erforderlich.

Eine RAID-Z-Konfiguration mit drei Datenträgern würde im Prinzip ungefähr wie folgt aussehen:

```
raidz c1t0d0 c2t0d0 c3t0d0
```

Eine komplexere RAID-Z-Konfiguration würde im Prinzip ungefähr wie folgt aussehen:

```
raidz c1t0d0 c2t0d0 c3t0d0 c4t0d0 c5t0d0 c6t0d0 c7t0d0 raidz c8t0d0 c9t0d0 c10t0d0 c11t0d0
c12t0d0 c13t0d0 c14t0d0
```

Beim Erstellen einer RAID-Z-Konfiguration wie in diesem Beispiel sollten Sie eine RAID-Z-Konfiguration mit 14 Datenträgern besser in zwei Gruppen mit je 7 Datenträgern aufteilen. RAID-Z-Konfigurationen mit weniger als zehn Datenträgern erreichen eine optimalere Leistung.

Informationen zum Erstellen von RAID-Z-Speicher-Pools finden Sie unter „Erstellen von RAID-Z-Speicher-Pools“ auf Seite 68.

Weitere Informationen zu auf Leistungs- und Speicherplatzanforderungen basierenden Auswahlkriterien zwischen Datenspiegelungs- oder RAID-Z-Konfiguration finden Sie im folgenden Blog:

`http://blogs.sun.com/roller/page/roch?entry=when_to_and_not_to`

Zusätzliche Informationen zu Empfehlungen für RAID-Z-Speicher-Pools finden Sie auf der Best Practices-Website für ZFS unter:

`http://www.solarisinternals.com/wiki/index.php/ZFS_Best_Practices_Guide`

ZFS-Hybrid-Speicher-Pool

Bei dem in der Produktreihe Sun Storage 7000 verfügbaren ZFS-Hybrid-Speicher-Pool handelt es sich um einen speziellen Speicher-Pool, der DRAM, SSDs und HDDs zur Erhöhung von Leistung und Kapazität bei gleichzeitiger Verringerung des Energieverbrauchs kombiniert. Sie können die ZFS-Redundanzkonfiguration des Speicher-Pools auswählen und einfach andere Konfigurationsoptionen mit dieser Verwaltungsschnittstelle des Produkts verwalten.

Weitere Informationen zu diesem Produkt finden Sie im *Sun Storage Unified Storage System Administration Guide* .

Selbstheilende Daten in einer redundanten Konfiguration

ZFS bietet in Konfigurationen mit Datenspiegelung und RAID-Z Selbstheilungsfunktionen.

Bei Erkennung beschädigter Datenblöcke ruft ZFS nicht nur die unbeschädigten Daten von einer redundanten Kopie ab, sondern repariert die beschädigten Daten auch, indem es diese mit der unbeschädigten Kopie ersetzt.

Dynamisches Striping in einem Speicher-Pool

Für jedes virtuelle Gerät, das zum Pool hinzugefügt wird, verteilt ZFS Daten dynamisch mithilfe des Stripe-Verfahrens auf verfügbaren Datenspeichergeräten. Die Entscheidung, wo Daten gespeichert werden, wird zur Zeit des Speichervorgangs getroffen.

Beim Hinzufügen virtueller Geräte zum Pool weist ZFS dem neuen Gerät zur Einhaltung von Leistungs- und Speicherplatzzuweisungsrichtlinien schrittweise Daten zu. Jedes virtuelle Gerät kann auch für die Datenspiegelung oder als RAID-Z-Gerät, das andere Datenträgergeräte bzw. Dateien enthält, eingesetzt werden. Mit dieser Konfiguration wird die Kontrolle der Fehlercharaketristika eines Pools flexibler. So können Sie beispielsweise aus vier Datenträgern die folgenden Konfigurationen bilden:

- vier Datenträger mit dynamischem Striping
- eine vierfache RAID-Z-Konfiguration
- zwei zweifache Geräte für die Datenspiegelung, die dynamisches Striping verwenden

Obwohl ZFS die Kombination verschiedener Arten virtueller Geräte im gleichen Pool unterstützt, wird dies nicht empfohlen. Sie können beispielsweise einen Pool mit einer zweifachen Datenspiegelungs- und einer dreifachen RAID-Z-Konfiguration erstellen. Die Fehlertoleranz des Gesamtsystems ist jedoch nur so gut wie die die Geräts mit der schlechtesten Fehlertoleranz, in diesem Fall RAID-Z. Im Allgemeinen ist es üblich, virtuelle Geräte der obersten Hierarchieebene zu verwenden, die das gleiche Redundanzniveau besitzen.

Erstellen und Entfernen von ZFS-Speicher-Pools

In den folgenden Abschnitten werden unterschiedliche Situationen zum Erstellen und Entfernen von ZFS-Speicher-Pools beschrieben.

- „Erstellen eines ZFS-Speicher-Pools“ auf Seite 66
- „Behandlung von Fehlern beim Erstellen von ZFS-Speicher-Pools“ auf Seite 73
- „Löschen von ZFS-Speicher-Pools“ auf Seite 76
- „Anzeigen von Informationen zu virtuellen Geräten in Storage-Pools“ auf Seite 71

Das Erstellen und Entfernen von Pools ist im Prinzip schnell und einfach. Sie sollten jedoch beim Ausführen solcher Operationen äußerste Vorsicht walten lassen. Obwohl Überprüfungen durchgeführt werden, die die Verwendung von Geräten verhindern, die bereits von einem anderen Pool benutzt werden, ist es ZFS nicht immer bekannt, ob ein Datenspeichergerät bereits belegt ist oder nicht. Das Entfernen von Pools ist noch einfacher. Benutzen Sie den Befehl `zpool destroy` mit Vorsicht. Dabei handelt es sich um einen einfachen Befehl mit erheblichen Konsequenzen.

Erstellen eines ZFS-Speicher-Pools

Speicher-Pools können mit dem Befehl `zpool create` erstellt werden. Dieser Befehl akzeptiert einen Pool-Namen und eine beliebige Anzahl virtueller Geräte als Argumente. Der Pool-Name muss den unter „Konventionen für das Benennen von ZFS-Komponenten“ auf Seite 45 aufgeführten Namenskonventionen genügen.

Erstellen eines einfachen Speicher-Pools

Mit dem folgenden Befehl wird ein Pool namens `tank` erstellt, das aus den beiden Datenträgern `c1t0d0` und `c1t1d0` besteht:

```
# zpool create tank c1t0d0 c1t1d0
```

Diese gesamten Datenträger befinden sich im Verzeichnis `/dev/dsk` und werden von ZFS als einzelner, großer Bereich gekennzeichnet. Daten werden mithilfe von Striping dynamisch über beide Datenträger verteilt.

Erstellen eines Speicher-Pools mit Datenspiegelung

Verwenden Sie zum Erstellen eines Pools mit Datenspiegelung das Schlüsselwort `mirror`, gefolgt von einer bestimmten Anzahl von Datenspeichergeräten, aus denen das Datenspiegelungssystem bestehen soll Mehrere Datenspiegelungssysteme können durch Wiederholen des Schlüsselworts `mirror` in der Befehlszeile erstellt werden. Mit dem folgenden Befehl wird ein Pool mit zwei zweifachen Datenspiegelungen erstellt:

```
# zpool create tank mirror c1d0 c2d0 mirror c3d0 c4d0
```

Das zweite Schlüsselwort `mirror` spezifiziert ein neues virtuelles Gerät der obersten Hierarchieebene. Daten werden mithilfe von Striping dynamisch über beide Datenträger verteilt, wobei zwischen jedem Datenträger eine jeweilige Datenredundanz auftritt.

Weitere Informationen zu empfohlenen Konfigurationen mit Datenspiegelung finden Sie unter:

`http://www.solarisinternals.com/wiki/index.php/ZFS_Best_Practices_Guide`

Gegenwärtig werden für ZFS-Datenspiegelungskonfigurationen die folgenden Operationen unterstützt:

- Hinzufügen eines weiteren Festplattensatzes zu einer vorhandenen Datenspiegelungskonfiguration, um eine neue obere `vdev`-Schicht zu erhalten. Weitere Informationen dazu finden Sie unter „Hinzufügen von Datenspeichergeräten zu einem Speicher-Pool“ auf Seite 77.
- Hinzufügen weiterer Festplatten zu einer vorhandenen Datenspiegelungskonfiguration bzw. Hinzufügen weiterer Festplatten zu einer nicht replizierten Konfiguration, um eine Datenspiegelungskonfiguration zu erstellen. Weitere Informationen dazu finden Sie unter „Verbinden und Trennen von Geräten in einem Speicher-Pool“ auf Seite 82.
- Ersetzen einer oder mehrerer Festplatten in einer vorhandenen Datenspiegelungskonfiguration, solange die Kapazität der neuen Festplatten größer oder gleich der Kapazität der zu ersetzenden Festplatten ist. Weitere Informationen dazu finden Sie unter „Austauschen von Geräten in einem Speicher-Pool“ auf Seite 86.

- Abtrennen einer oder mehrerer Festplatten in einer Datenspiegelungskonfiguration, solange die verbleibenden Speichergeräte für die Konfiguration noch eine ausreichende Redundanz gewährleisten. Weitere Informationen dazu finden Sie unter „Verbinden und Trennen von Geräten in einem Speicher-Pool“ auf Seite 82.

Die folgenden Operationen werden gegenwärtig für ZFS-Datenspiegelungskonfigurationen nicht unterstützt:

- Ein Speichergerät kann nicht direkt aus einem Speicher-Pool mit Datenspiegelung entfernt werden. Dafür wurde ein RFE eingereicht.
- Eine Datenspiegelungskonfiguration kann nicht zum Zwecke des Erstellens von Sicherungskopien aufgesplittet bzw. aufgebrochen werden. Dafür wurde ein RFE eingereicht.

Erstellen eines ZFS-Root-Pools

In aktuellen Solaris-Releases können Sie von einem ZFS-Root-Dateisystem installieren und booten. Beachten Sie die folgenden Root-Pool-Konfigurationsinformationen:

- Für den Root-Pool verwendete Datenträger müssen ein VTOC (SMI)-Label aufweisen, und der Pool muss mit Festplattenbereichen erstellt werden.
- Ein Root-Pool muss als gespiegelte Konfiguration oder als einzelne Festplattenkonfiguration erstellt werden. Sie können keine zusätzlichen Festplatten hinzufügen, um mehrere gespiegelte vdevs zu erstellen, Sie können jedoch eine gespiegelte vdev mit dem Befehl `zpool attach` erweitern.
- RAID-Z- oder Stripes-Konfigurationen werden nicht unterstützt.
- Ein Root-Pool kann über kein separates Protokolliergerät verfügen.
- Bei dem Versuch, eine nicht unterstützte Pool-Konfiguration für einen Root-Pool zu verwenden, erscheint eine Meldung folgender Art:

```
ERROR: ZFS pool <pool-name> does not support boot environments

# zpool add -f rpool log c0t6d0s0
cannot add to 'rpool': root pool can not have multiple vdevs or separate logs
```

Weitere Informationen zum Installieren und Booten eines ZFS-Dateisystems können Sie dem Abschnitt Kapitel 5, „Installieren und Booten eines ZFS-Root-Dateisystems“ entnehmen.

Erstellen von RAID-Z-Speicher-Pools

Das Erstellen eines RAID-Z-Pools mit einfacher Parität läuft ähnlich ab wie das Erstellen eines Pools mit Datenspiegelung. Der einzige Unterschied besteht darin, dass anstatt des Schlüsselworts `mirror` das Schlüsselwort `raidz` bzw. `raidz1` verwendet wird. Das folgende Beispiel zeigt die Erstellung eines Pools mit einem RAID-Z-Gerät, das aus fünf Datenträgern besteht:

```
# zpool create tank raidz c1t0d0 c2t0d0 c3t0d0 c4t0d0 /dev/dsk/c5t0d0
```

Dieses Beispiel zeigt, dass Datenträger mit dem vollständigen Pfad angegeben werden können. Das Gerät `/dev/dsk/c5t0d0` ist das gleiche Gerät wie `c5t0d0`.

Eine ähnliche Konfiguration kann mit Datenträgerbereichen erstellt werden. Beispiel:

```
# zpool create tank raidz c1t0d0s0 c2t0d0s0 c3t0d0s0 c4t0d0s0 c5t0d0s0
```

Die Datenträger müssen jedoch so vorformatiert werden, dass sie einen Bereich 0 mit angemessener Kapazität besitzen.

Eine RAID-Z-Konfiguration doppelter Parität kann beim Erstellen eines Pools mithilfe des Schlüsselworts `raidz2` angelegt werden. Beispiel:

```
# zpool create tank raidz2 c1t0d0 c2t0d0 c3t0d0
# zpool status -v tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME          STATE     READ WRITE CKSUM
        tank          ONLINE       0     0     0
          raidz2      ONLINE       0     0     0
            c1t0d0    ONLINE       0     0     0
            c2t0d0    ONLINE       0     0     0
            c3t0d0    ONLINE       0     0     0

errors: No known data errors
```

Gegenwärtig werden für ZFS RAID-Z-Konfigurationen die folgenden Operationen unterstützt:

- Hinzufügen eines weiteren Festplattensatzes zu einer vorhandenen RAID-Z-Konfiguration, um eine neue obere `vdev`-Schicht zu erhalten. Weitere Informationen dazu finden Sie unter „Hinzufügen von Datenspeichergeräten zu einem Speicher-Pool“ auf Seite 77.
- Ersetzen einer oder mehrerer Festplatten in einer vorhandenen RAID-Z-Konfiguration, solange die Kapazität der neuen Festplatten größer oder gleich der Kapazität der zu ersetzenden Festplatten ist. Weitere Informationen dazu finden Sie unter „Austauschen von Geräten in einem Speicher-Pool“ auf Seite 86.

Die folgenden Operationen werden gegenwärtig für RAID-Z-Konfigurationen nicht unterstützt:

- Hinzufügen einer weiteren Festplatte zu einer vorhandenen RAID-Z-Konfiguration.
- Abtrennen einer Festplatte von einer RAID-Z-Konfiguration.
- Ein Speichergerät kann nicht direkt aus einer RAID-Z-Konfiguration entfernt werden. Dafür wurde ein RFE eingereicht.

Weitere Informationen zu RAID-Z-Konfigurationen finden Sie unter „Speicher-Pools mit RAID-Z-Konfiguration“ auf Seite 64.

Erstellen eines ZFS-Speicher-Pools mit Protokolliergeräten

Standardmäßig wird das ZIL aus Blöcken innerhalb des Haupt-Pools zugewiesen. Durch Verwendung getrennter Intent-Protokolliergeräte wie z. B. NVRAM oder eine speziell dafür vorgesehene Festplatte kann jedoch eine höhere Leistung erreicht werden. Weitere Informationen zu ZFS-Protokolliergeräten finden Sie unter „Einrichten separater ZFS-Protokolliergeräte“ auf Seite 28.

Die Einrichtung eines ZFS-Protokolliergeräts kann bei oder nach der Erstellung des Speicher-Pools erfolgen.

Erstellen Sie beispielsweise einen Speicher-Pool mit Datenspiegelung und gespiegelten Protokolliergeräten.

```
# zpool create datap mirror c1t1d0 c1t2d0 mirror c1t3d0 c1t4d0 log mirror c1t5d0 c1t8d0
# zpool status
  pool: datap
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        datap       ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0
            c1t4d0  ONLINE       0     0     0
        logs        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c1t5d0  ONLINE       0     0     0
            c1t8d0  ONLINE       0     0     0

errors: No known data errors
```

Informationen zum Wiederherstellen des Normalbetriebs nach dem Fehlschlag eines Protokolliergeräts finden Sie unter Beispiel 11–2.

Erstellen eines ZFS-Speicher-Pools mit Cache-Geräten

Sie können mit Cache-Geräten einen Pool erstellen, um Speicher-Pool-Daten im Cache zu speichern. Beispiele:

```
# zpool create tank mirror c2t0d0 c2t1d0 c2t3d0 cache c2t5d0 c2t8d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0
        cache
          c2t5d0    ONLINE       0     0     0
          c2t8d0    ONLINE       0     0     0
```

Beachten Sie folgende Aspekte, wenn Sie die Erstellung eines ZFS-Speicher-Pools mit Cache-Geräten in Betracht ziehen:

- Die Verwendung von Cache-Speicher bietet für die Speicherung meist statischer Daten mithilfe von wahlfreiem Zugriff die größte Leistungsverbesserung.
- Die Kapazität und Lesevorgänge können mithilfe des Befehls `zpool iostat` überwacht werden.
- Einzelne oder mehrere Cache-Geräte können sowohl während als auch nach der Erstellung des Pools hinzugefügt und entfernt werden. Weitere Informationen finden Sie unter Beispiel 4–4.
- Cache-Geräte können nicht gespiegelt werden oder Teil einer RAID-Z-Konfiguration sein.
- Wird auf einem Cache-Gerät ein Lesefehler entdeckt, so werden diese E/A-Leseoperationen wieder dem ursprünglichen Speicher-Pool-Gerät zugewiesen, was Teil einer gespiegelten oder RAID-Z-Konfiguration sein kann. Der Inhalt der Cache-Geräte wird als flüchtig betrachtet, wie auch bei anderen Systemcaches.

Anzeigen von Informationen zu virtuellen Geräten in Storage-Pools

Jeder Speicher-Pool besteht aus einem oder mehreren virtuellen Geräten. Unter einem *virtuellen Gerät* versteht man eine interne Darstellung eines Speicher-Pools, das die Struktur der physischen Datenspeicherung und seiner Fehlercharakteristika beschreibt. Somit repräsentiert ein virtuelles Gerät die Datenträger bzw. Dateien, die zum Erstellen eines Speicher-Pools verwendet werden. Ein Pool kann beliebig viele virtuelle Geräte auf oberster Konfigurationsebene aufweisen. Dieses werden *root vdev* genannt.

Datenredundanz wird von zwei virtuellen Geräten der Root- oder obersten Hierarchieebene bereitgestellt: datengespiegelten und virtuellen RAID-Z-Geräten. Diese virtuellen Geräte bestehen aus Datenträgern, Datenträgerbereichen oder Dateien. Als Spare werden spezielle vdevs bezeichnet, die die verfügbaren Hot-Spares für einen Pool überwachen.

Das folgende Beispiel veranschaulicht die Erstellung eines Pools aus zwei Root-vdevs, die jeweils eine Datenspiegelungskonfiguration aus zwei Festplatten darstellen.

```
# zpool create tank mirror c1d0 c2d0 mirror c3d0 c4d0
```

Das folgende Beispiel veranschaulicht die Erstellung eines Pools, das aus einem Root-vdev mit vier Festplatten besteht.

```
# zpool create mypool raidz2 c1d0 c2d0 c3d0 c4d0
```

Mit dem Befehl `zpool add` kann diesem Pool ein weiteres Root-vdev hinzugefügt werden. Beispiel:

```
# zpool add mypool raidz2 c2d0 c3d0 c4d0 c5d0
```

Festplatten, Festplattenbereiche oder Dateien, die in nicht-redundanten Pools verwendet werden, fungieren selbst als virtuelle Geräte der obersten Hierarchieebene. Speicher-Pools enthalten normalerweise in der obersten Hierarchieebene mehrere virtuelle Geräte. ZFS verteilt Daten dynamisch mithilfe des sog. Stripe-Verfahrens über alle in der obersten Hierarchieebene befindlichen virtuellen Geräte eines Pools.

Virtuelle Geräte und die physischen Geräte, die in einem ZFS-Speicher-Pool enthalten sind, werden mit dem Befehl `zpool status` angezeigt. Beispiel:

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

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0
```

Behandlung von Fehlern beim Erstellen von ZFS-Speicher-Pools

Fehler beim Erstellen von Pools können in verschiedenen Situationen auftreten. Einige Fehlergründe sind sofort offensichtlich, wenn beispielsweise ein angegebenes Gerät nicht existiert, währenddessen andere Gründe nicht so einfach nachzuvollziehen sind.

Erkennen belegter Geräte

Vor dem Formatieren eines Datenspeichergeräts versucht ZFS herauszufinden, ob dieser Datenträger bereits von ZFS oder anderen Bereichen des Betriebssystems verwendet wird. Wenn der Datenträger bereits verwendet wird, sehen Sie in etwa folgende Fehlermeldung:

```
# zpool create tank c1t0d0 c1t1d0
invalid vdev specification
use '-f' to override the following errors:
/dev/dsk/c1t0d0s0 is currently mounted on /. Please see umount(1M).
/dev/dsk/c1t0d0s1 is currently mounted on swap. Please see swap(1M).
/dev/dsk/c1t1d0s0 is part of active ZFS pool zeepool. Please see zpool(1M).
```

Einige dieser Fehler können mithilfe der Option `-f` ignoriert werden, die meisten Fehler jedoch nicht. Die folgenden Fehler können mithilfe der Option `-f` nicht ignoriert werden, und Sie müssen diese manuell beheben:

Mounted file system (eingehängtes Dateisystem)
: Der Datenträger oder einige seiner Bereiche enthalten ein gegenwärtig eingehängtes Dateisystem. Sie müssen diesen Fehler mit dem Befehl `umount` beheben.

File system in /etc/vfstab (Dateisystem in /etc/vfstab)
: Der Datenträger enthält ein Dateisystem, das in der Datei `/etc/vfstab` aufgeführt, jedoch gegenwärtig nicht eingehängt ist. Zur Behebung dieses Fehlers müssen Sie die entsprechende Zeile aus der Datei `/etc/vfstab` entfernen bzw. diese Zeile auskommentieren.

Dedicated dump device (reserviertes Dump-Gerät)
: Dieser Datenträger ist das reservierte Dump-Gerät für dieses System. Sie müssen diesen Fehler mit dem Befehl `dumpadm` beheben.

Part of a ZFS pool (Teil eines ZFS-Pools)
: Der Datenträger bzw. die Datei gehört zu einem aktiven ZFS-Speicher-Pool. Sofern nicht mehr benötigt, entfernen Sie den anderen Pool mit dem Befehl `zpool destroy`. Oder verwenden Sie den Befehl `zpool detach`, um die Festplatte aus dem anderen Pool zu trennen. Sie können nur eine Festplatte aus einem Speicher-Pool mit Datenspiegelung trennen.

Die folgenden Überprüfungen auf Datenträgerbelegungen dienen als nützliche Warnungen und können mithilfe der Option `-f` manuell übergangen werden, um den Pool zu erstellen:

Contains a file system (Enthält ein Dateisystem)
Der Datenträger enthält ein bekanntes Dateisystem, obwohl es nicht eingehängt ist und nicht in Gebrauch zu sein scheint.

Part of volume (Teil eines Volumes)
Der Datenträger gehört zu einem SVM-Volume.

Live Upgrade
Der Datenträger dient als alternative Boot-Umgebung für Solaris Live Upgrade.

Part of a exported ZFS pool (Teil eines exportierten ZFS-Pools)
Der Datenträger gehört zu einem Speicher-Pool, das exportiert bzw. manuell aus einem System entfernt wurde. Im letzteren Fall wird der Pool als `potenziell aktiv` gemeldet, da der Datenträger unter Umständen über das Netzwerk von einem anderen System belegt sein kann. Überprüfen Sie den Status vor beim Überschreiben eines potenziell aktiven Datenträgers äußerst sorgfältig.

Das folgende Beispiel zeigt die Verwendung der Option `-f`:

```
# zpool create tank c1t0d0
invalid vdev specification
use '-f' to override the following errors:
/dev/dsk/c1t0d0s0 contains a ufs filesystem.
# zpool create -f tank c1t0d0
```

Sie sollten im Normalfall jedoch Fehler beheben, anstatt die Option `-f` zu verwenden.

Inkongruente Replikationsmethoden

Die Erstellung von Pools mit unterschiedlichen Replikationsmethoden wird nicht empfohlen. Der Befehl `zpool` verhindert, dass Sie versehentlich einen Pool mit inkongruenten Redundanzebenen erstellen. Wenn Sie versuchen, einen Pool mit einer solchen Konfiguration zu erstellen, wird in etwa die folgende Fehlermeldung ausgegeben:

```
# zpool create tank c1t0d0 mirror c2t0d0 c3t0d0
invalid vdev specification
use '-f' to override the following errors:
mismatched replication level: both disk and mirror vdevs are present
# zpool create tank mirror c1t0d0 c2t0d0 mirror c3t0d0 c4t0d0 c5t0d0
invalid vdev specification
use '-f' to override the following errors:
mismatched replication level: 2-way mirror and 3-way mirror vdevs are present
```

Sie können diese Fehler mit der Option `-f` ignorieren, obwohl dies nicht empfohlen wird. Der Befehl warnt Sie darüber hinaus auch, wenn Sie versuchen, einen Pool mit Datenspiegelung bzw. RAID-Z zu erstellen, das Datenträger unterschiedlicher Kapazität verwendet. Obwohl eine solche Konfiguration zulässig ist, verursachen inkongruente Redundanzebenen auf dem Datenträger höherer Kapazität ungenutzten Speicherplatz, und die entsprechende Warnung muss mit der Option `-f` ignoriert werden.

Ausführen eines Testlaufs für die Erstellung eines Speicher-Pools

Da die Erstellung eines Pools auf unterschiedliche Weise fehlschlagen und das Formatieren von Datenträgern erhebliche Konsequenzen haben kann, besitzt der Befehl `zpool create` die zusätzliche Option `-n`, die die Erstellung des Pools simuliert, ohne dass das Gerät dabei tatsächlich beschrieben wird. Diese Option führt die Belegungsüberprüfung von Datenträgern sowie die Evaluierung der Replikationsmethoden durch und meldet im Verlauf dieses Vorgangs eventuell auftretende Fehler. Wenn keine Fehler gefunden wurden, wird in etwa die folgende Meldung ausgegeben:

```
# zpool create -n tank mirror c1t0d0 c1t1d0
would create 'tank' with the following layout:

        tank
          mirror
            c1t0d0
            c1t1d0
```

Einige Fehler werden jedoch erst angezeigt, wenn der Pool tatsächlich erstellt wird. Ein Beispiel für einen solchen häufig auftretenden Fehler ist die mehrmalige Angabe eines Datenträgers in der gleichen Konfiguration. Dieser Fehler kann erst beim tatsächlichen Schreiben von Daten zuverlässig erkannt werden. Aus diesem Grund kann es sein, dass der Befehl `create -n` beim Testlauf fehlerfrei funktioniert und beim tatsächlichen Erstellen des Pools Fehler meldet.

Standard-Einhängepunkt für Speicher-Pools

Bei der Erstellung eines Speicher-Pools ist der Standard-Einhängepunkt für das Root-Dataset */pool-name*. Dieses Verzeichnis darf noch nicht vorhanden oder muss leer sein. Falls das Verzeichnis noch nicht existiert, wird es autoamtisch erstellt. Wenn das Verzeichnis leer ist, wird das Root-Dataset in dieses vorhandene Verzeichnis eingehängt. Mithilfe der Option `-m` des Befehls `zpool create` können Sie Pools mit anderen Standard-Einhängepunkten erstellen:

```
# zpool create home c1t0d0
default mountpoint '/home' exists and is not empty
use '-m' option to specify a different default
# zpool create -m /export/zfs home c1t0d0

# zpool create home c1t0d0
default mountpoint '/home' exists and is not empty
use '-m' option to provide a different default
# zpool create -m /export/zfs home c1t0d0
```

Dieser Befehl erstellt den neuen Pool `home`, und das Dataset `home` wird als `/export/zfs` eingehängt.

Weitere Informationen zu Einhängepunkten finden Sie unter „Verwalten von ZFS-Einhängepunkten“ auf Seite 202.

Löschen von ZFS-Speicher-Pools

Pools werden mit dem Befehl `zpool destroy` gelöscht. Dieser Befehl löscht einen Pool auch dann, wenn er eingehängte Datasets enthält.

```
# zpool destroy tank
```

Achtung – Seien Sie beim Löschen von Pools äußerst vorsichtig. Vergewissern Sie sich, dass der richtige Pool gelöscht wird und Sie von den Daten dieses Pools entsprechende Sicherungskopien erstellt haben. Falls versehentlich der falsche Pool gelöscht wurde, kann er eventuell wiederhergestellt werden. Weitere Informationen dazu finden Sie unter „Wiederherstellen gelöschter ZFS-Speicher-Pools“ auf Seite 110.

Löschen eines Pools mit fehlerhaften Geräten

Nach dem Löschen eines Pools muss den Daten, die auf Datenträgern gespeichert werden sollen, mitgeteilt werden, dass der betreffende Pool nicht mehr verfügbar ist. Diese Statusinformationen verhindern, dass beim Durchführen von Imports diese Datenspeichergeräte als potenziell verfügbar angezeigt werden. Auch wenn Datenspeichergeräte nicht verfügbar sind, kann der Pool gelöscht werden. Die erforderlichen Statusinformationen werden in diesem Fall jedoch nicht auf den beschädigten Datenträgern gespeichert.

Nach einer entsprechenden Reparatur werden dieser Datenspeichergeräte bei der Erstellung neuer Pools als *potenziell aktiv* gemeldet und erscheinen für zu importierende Pools als zulässige Geräte. Wenn ein Pool so viele fehlerhafte Datenspeichergeräte enthält, dass der Pool selbst als fehlerhaft erkannt wird (d.h. das Gerät der obersten Hierarchieebene ist fehlerhaft), gibt der Befehl eine Warnmeldung aus und kann ohne die Option `-f` nicht fortgesetzt werden. Diese Option ist erforderlich, da der Pool nicht geöffnet werden kann und somit nicht bekannt ist, ob dort Daten gespeichert sind. Beispiel:

```
# zpool destroy tank
cannot destroy 'tank': pool is faulted
use '-f' to force destruction anyway
# zpool destroy -f tank
```

Weitere Informationen zum Funktionsstaus von Pools und Datenspeichergeräten finden Sie unter „Ermitteln des Funktionsstatus von ZFS-Speicher-Pools“ auf Seite 101.

Weitere Informationen zum Importieren von Pools finden Sie unter „Importieren von ZFS-Speicher-Pools“ auf Seite 109.

Verwalten von Datenspeichergeräten in ZFS-Speicher-Pools

Die meisten grundlegenden Informationen zu Datenspeichergeräten sind in „Komponenten eines ZFS-Speicher-Pools“ auf Seite 59 enthalten. Nach dem Erstellen eines Pools können Sie zum Verwalten der zum Pool gehörenden Datenspeichergeräte verschiedene Aufgaben ausführen.

Hinzufügen von Datenspeichergeräten zu einem Speicher-Pool

Durch Hinzufügen eines neuen Gerätes der obersten Hierarchieebene können Sie einen Pool dynamisch um Speicherplatz erweitern. Der zusätzliche Speicherplatz steht allen im Pool enthaltenen Datasets sofort zur Verfügung. Mit dem Befehl `zpool add` fügen Sie einem Pool ein neues virtuelles Gerät hinzu. Beispiel:

```
# zpool add zeepool mirror c2t1d0 c2t2d0
```

Das Format zum Angeben dieser virtuellen Geräte entspricht dem Format für den Befehl `zpool create`, und es gelten die gleichen Regeln. Es wird geprüft, ob die betreffenden Datenspeichergeräte belegt sind, und der Befehl kann die Redundanzebene ohne die Option `-f` nicht ändern. Der Befehl unterstützt auch die Option `-n` zum Durchführen eines Testlaufs. Beispiel:

```
# zpool add -n zeepool mirror c3t1d0 c3t2d0
would update 'zeepool' to the following configuration:
      zeepool
        mirror
            c1t0d0
            c1t1d0
        mirror
            c2t1d0
            c2t2d0
        mirror
            c3t1d0
            c3t2d0
```

Diese Befehlssyntax fügt zur vorhandenen Konfiguration von `zeepool` die datengespiegelten Geräte `c3t1d0` und `c3t2d0` hinzu.

Weitere Informationen zur Überprüfung von virtuellen Geräten finden Sie unter „Erkennen belegter Geräte“ auf Seite 73.

BEISPIEL 4–1 Hinzufügen von Festplatten in eine ZFS-Konfiguration mit Datenspiegelung

In folgendem Beispiel wird eine ZFS-Konfiguration mit Datenspiegelung auf einem Sun Fire x4500-System um eine weitere Spiegelplatte ergänzt.

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

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0

errors: No known data errors
# zpool add tank mirror c0t3d0 c1t3d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0

errors: No known data errors
```

BEISPIEL 4–2 Hinzufügen von Datenträgern zu einer RAID-Z-Konfiguration

Datenträger können in ähnlicher Weise auch zu einer RAID-Z-Konfiguration hinzugefügt werden. Das folgende Beispiel zeigt, wie ein Speicher-Pool mit einem aus drei Festplatten bestehenden RAID-Z-Gerät in ein Speicher-Pool mit zwei aus je drei Datenträgern bestehenden RAID-Z-Geräten umgewandelt werden kann.

```
# zpool status
  pool: rpool
 state: ONLINE
 scrub: none requested
config:
        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          raidz1      ONLINE       0     0     0
            c1t2d0    ONLINE       0     0     0
            c1t3d0    ONLINE       0     0     0
            c1t4d0    ONLINE       0     0     0

errors: No known data errors
# zpool add rpool raidz c2t2d0 c2t3d0 c2t4d0
# zpool status
  pool: rpool
 state: ONLINE
 scrub: none requested
config:
        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          raidz1      ONLINE       0     0     0
            c1t2d0    ONLINE       0     0     0
            c1t3d0    ONLINE       0     0     0
            c1t4d0    ONLINE       0     0     0
          raidz1      ONLINE       0     0     0
            c2t2d0    ONLINE       0     0     0
            c2t3d0    ONLINE       0     0     0
            c2t4d0    ONLINE       0     0     0

errors: No known data errors
```

BEISPIEL 4–3 Hinzufügen eines gespiegelten Protokolliergeräts zu einem ZFS-Speicher-Pool

Das folgende Beispiel zeigt, wie einem Speicher-Pool mit Datenspiegelung ein gespiegeltes Protokolliergerät hinzugefügt werden kann.Weitere Informationen zum Einsatz von Protokolliergeräten in Speicher-Pools finden Sie unter „Einrichten separater ZFS-Protokolliergeräte“ auf Seite 28.

BEISPIEL 4–3 Hinzufügen eines gespiegelten Protokolliergeräts zu einem ZFS-Speicher-Pool *(Fortsetzung)*

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

        NAME         STATE     READ WRITE CKSUM
        newpool      ONLINE       0     0     0
          mirror     ONLINE       0     0     0
            c1t9d0   ONLINE       0     0     0
            c1t10d0  ONLINE       0     0     0

errors: No known data errors
# zpool add newpool log mirror c1t11d0 c1t12d0
# zpool status newpool
  pool: newpool
 state: ONLINE
 scrub: none requested
config:

        NAME         STATE     READ WRITE CKSUM
        newpool      ONLINE       0     0     0
          mirror     ONLINE       0     0     0
            c1t9d0   ONLINE       0     0     0
            c1t10d0  ONLINE       0     0     0
        logs         ONLINE       0     0     0
          mirror     ONLINE       0     0     0
            c1t11d0  ONLINE       0     0     0
            c1t12d0  ONLINE       0     0     0

errors: No known data errors
```

Sie können zur Datenspiegelung an ein vorhandenes Protokolliergerät ein weiteres Protokolliergerät anschließen. Dies entspricht dem Verbinden eines Speichergeräts in einem Speicherpool ohne Datenspiegelung.

BEISPIEL 4–4 Cache-Geräte zu ZFS-Speicher-Pools hinzufügen bzw. sie von diesen entfernen

Sie können Cache-Geräte zu ZFS-Speicher-Pools hinzufügen bzw. sie von ihnen entfernen

Verwenden Sie den Befehl `zpool add` zum Hinzufügen von Cache-Geräten. Beispiele:

```
# zpool add tank cache c2t5d0 c2t8d0
# zpool status tank
  pool: tank
```

BEISPIEL 4–4 Cache-Geräte zu ZFS-Speicher-Pools hinzufügen bzw. sie von diesen entfernen *(Fortsetzung)*

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

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0
        cache
          c2t5d0    ONLINE       0     0     0
          c2t8d0    ONLINE       0     0     0

errors: No known data errors
```

Cache-Geräte können nicht gespiegelt werden oder Teil einer RAID-Z-Konfiguration sein.

Verwenden Sie den Befehl `zpool remove` zum Entfernen von Cache-Geräten. Beispiel:

```
# zpool remove tank c2t5d0 c2t8d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0

errors: No known data errors
```

Der Befehl `zpool remove` unterstützt derzeit nur das Entfernen von Hot-Spares und Cache-Geräten. Geräte, die zur primären gespiegelten Pool-Konfiguration gehören, können mit dem Befehl `zpool detach` entfernt werden. Geräte ohne Redundanz und RAID-Z-Geräte können nicht aus einem Pool entfernt werden.

Weitere Informationen zur Verwendung von Cache-Geräten in einem ZFS-Speicher-Pool finden Sie unter „Erstellen eines ZFS-Speicher-Pools mit Cache-Geräten“ auf Seite 70.

Verbinden und Trennen von Geräten in einem Speicher-Pool

Zusätzlich zum Befehl `zpool add` können Sie mit dem Befehl `zpool attach` zu einem Gerät mit oder ohne Datenspiegelung ein neues Datenspeichergerät hinzufügen.

Wenn Sie eine Festplatte in einem ZFS-Root-Pool hinzufügen und entfernen, lesen Sie unter „So ersetzen Sie eine Festplatte im ZFS-Root-Pool“ auf Seite 167 nach.

BEISPIEL 4–5 Umwandlung eines Speicher-Pools mit zweifacher Datenspiegelung in einen Speicher-Pool mit dreifacher Datenspiegelung

In diesem Befehl ist `zeepool` eine vorhandene zweifache Datenspiegelungskonfiguration, die durch Verbinden des Geräts `c2t1d0` mit dem vorhandenen Gerät `c1t1d0` in eine dreifache Datenspiegelungskonfiguration umgewandelt wird.

```
# zpool status
  pool: zeepool
 state: ONLINE
 scrub: none requested
config:
        NAME        STATE     READ WRITE CKSUM
        zeepool     ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
errors: No known data errors
# zpool attach zeepool c1t1d0 c2t1d0
# zpool status
  pool: zeepool
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Fri Aug 28 14:11:33 2009
config:

        NAME        STATE     READ WRITE CKSUM
        zeepool     ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0 73.5K resilvered
```

Wenn das betreffende Gerät zu einer zweifachen Datenspiegelungskonfiguration gehört, wird durch Verbinden des neuen Geräts eine dreifache Datenspiegelungskonfiguration erstellt usw. In jedem Fall wird sofort das Resilvering durchgeführt (d. h. die gespiegelten Daten auf das neue Gerät in der Datenspiegelungskonfiguration übertragen).

BEISPIEL 4–6 Umwandeln eines ZFS-Speicherpools ohne Redundanz in einen ZFS-Speicher-Pool mit Datenspiegelung

Darüber hinaus können Sie mithilfe des Befehls `zpool attach` ein Speicher-Pool ohne Redundanz in ein Pool mit Redundanz umwandeln. Beispiel:

```
# zpool create tank c0t1d0
# zpool status
  pool: tank
 state: ONLINE
 scrub: none requested
config:
        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          c0t1d0    ONLINE       0     0     0

errors: No known data errors
# zpool attach tank c0t1d0 c1t1d0
# zpool status
  pool: tank
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Fri Aug 28 14:13:12 2009
config:
        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0 73.5K resilvered
```

Mit dem Befehl `zpool detach` kann ein Datenspeichergerät aus einem Pool mit Datenspiegelungskonfiguration abgetrennt werden. Beispiel:

```
# zpool detach zeepool c2t1d0
```

Diese Operation ist allerdings nicht ausführbar, wenn keine anderen Replikationen der betreffenden Daten vorhanden sind. Beispiel:

```
# zpool detach newpool c1t2d0
cannot detach c1t2d0: only applicable to mirror and replacing vdevs
```

In- und Außerbetriebnehmen von Geräten in einem Speicher-Pool

Einzelne Datenspeichergeräte können in ZFS in und außer Betrieb genommen werden. Wenn Hardwarekomponenten unzuverlässig sind und nicht ordnungsgemäß funktionieren, schreibt ZFS trotzdem weiter Daten auf das betreffende Gerät bzw. liest sie von diesem und nimmt an,

dass diese Situation nur zeitweise auftritt. Falls dies nicht der Fall ist und diese Situation ein Dauerzustand ist, können Sie ZFS durch Außerbetriebnehmen des Geräts mitteilen, dass es ignoriert werden soll. ZFS sendet dann keine Anforderungen an das außer Betrieb genommene Datenspeichergerät.

Hinweis – Zum Austauschen von Datenspeichergeräten müssen diese vorher nicht außer Betrieb genommen werden.

Mit dem Befehl `offline` können Sie Datenspeichergeräte zeitweilig außer Betrieb nehmen. Wenn Sie beispielsweise ein Array aus einer Gruppe Fibre Channel-Switches herausnehmen und dieses ein andere andere Gruppe anschließen wollen, können Sie die LU-Nummern aus dem Array außer Betrieb nehmen, das in ZFS-Speicher-Pools verwendet wurde. Nachdem Sie das Array neu angeschlossen haben und dieses in der neuen Gruppe von Fibre Channel-Swicthes funktioniert, können Sie die betreffenden LU-Nummern wieder in Betrieb nehmen. Daten, die in der Zeit, in der die LU-Nummern außer Betrieb waren, in den Speicher-Pools gespeichert wurden, werden nach der Wiederinbetriebnahme dieser LU-Nummern durch Resilvering auf diese übertragen.

Diese Situation ist möglich, da angenommen wird, dass die Datenträger nach dem Anschließen an die neuen FC-Switches für die betreffenden Systeme sichtbar sind. Dies kann unter Umständern über andere Controller als zuvor geschehen, und die betreffenden Pools sind mit Datenspiegelung oder RAID-Z konfiguriert.

Außerbetriebnehmen eines Geräts

Datenspeichergeräte werden mit dem Befehl `zpool offline` außer Betrieb genommen. Wenn es sich bei dem Datenspeichergerät um eine Festplatte handelt, kann es mit dem vollständigen Pfad oder einer Kurzbezeichnung angegeben werden. Beispiel:

```
# zpool offline tank c1t0d0
bringing device c1t0d0 offline
```

Beim Außerbetriebnehmen von Datenspeichergeräten sollten Sie folgende Aspekte berücksichtigen:

- Pools können nicht außer Betrieb genommen werden, wenn dadurch die Datensicherheit beeinträchtigt wird. So können Sie beispielsweise zwei Geräte nicht aus einer RAID-Z-Konfiguration oder kein virtuelles Gerät der obersten Hierrachieebene außer Betrieb nehmen.

  ```
  # zpool offline tank c1t0d0
  cannot offline c1t0d0: no valid replicas
  ```

- Der Offline-Status ist dauerhaft, d. h. das Gerät bleibt auch bei einem Systemneustart außer Betrieb.

Mit dem Befehl `zpool offline -t` können Sie ein Datenspeichergerät zeiweilig außer Betrieb nehmen. Beispiel:

```
# zpool offline -t tank c1t0d0
 bringing device 'c1t0d0' offline
```

Bei einem Systemneustart wird dieses Gerät dann automatisch in den `ONLINE`-Status gebracht.

- Wenn ein Gerät außer Betrieb genommen wurde, wird es nicht automatisch vom Speicher-Pool getrennt. Wenn Sie versuchen, das betreffende Gerät für ein anderes Pool zu verwenden (selbst wenn dieses Pool gelöscht wurde), wird in etwa die folgende Meldung angezeigt:

  device `is part of exported or potentially active ZFS` *pool*`. Please see zpool(1M)`

  Wenn Sie das außer Betrieb genommene Gerät für ein anderes Pool verwenden wollen, wenn das ursprüngliche Pool gelöscht werden soll, müssen Sie das Gerät zunächst wieder in Betrieb nehmen und dann das ursprüngliche Pool löschen.

  Eine andere Methode zur Nutzung eines Geräts eines anderen Speicher-Pools, wenn das ursprüngliche Pool beibehalten werden soll, besteht darin, das vorhandene Gerät im ursprünglichen Pool durch ein anderes, vergleichbares Gerät zu ersetzen. Weitere Informationen zum Austauschen von Geräten finden Sie unter „Austauschen von Geräten in einem Speicher-Pool“ auf Seite 86.

Außer Betrieb genommene Geräte werden bei der Abfrage des Pool-Status unter `OFFLINE` angezeigt. Weitere Informationen zum Abfragen des Pool-Status finden Sie unter „Abfragen des Status von ZFS-Speicher-Pools“ auf Seite 95.

Weitere Informationen zum Funktionsstatus von Datenspeichergeräten finden Sie unter „Ermitteln des Funktionsstatus von ZFS-Speicher-Pools“ auf Seite 101.

Inbetriebnehmen eines Gerätes

Nach dem Außerbetriebnehmen eines Datenspeichergerätes kann es mit dem Befehl `zpool online` wieder in Betrieb genommen werden.

```
# zpool online tank c1t0d0
bringing device c1t0d0 online
```

Nach dem Inbetriebnehmen des Gerätes werden alle Daten, die im Pool gespeichert wurden, mit dem neu verfügbaren Gerät synchronisiert. Bitte beachten Sie, dass Datenspeichergeräte nicht durch Inbetriebnehmen ausgetauscht werden können. Wenn Sie ein Gerät außer Betrieb nehmen, es austauschen und versuchen, es wieder in Betrieb zu nehmen, verursacht das einen Fehlerzustand.

Wenn Sie versuchen, ein fehlerhaftes Gerät in Betrieb zu nehmen, wird in etwa die folgende Meldung angezeigt:

```
# zpool online tank c1t0d0
warning: device 'c1t0d0' onlined, but remains in faulted state
use 'zpool replace' to replace devices that are no longer present
```

Möglicherweise wird von `fmd` auch die Meldung "fehlerhafte Festplatte" angezeigt.

```
SUNW-MSG-ID: ZFS-8000-D3, TYPE: Fault, VER: 1, SEVERITY: Major
EVENT-TIME: Fri Aug 28 14:08:39 MDT 2009
PLATFORM: SUNW,Sun-Fire-T200, CSN: -, HOSTNAME: neo2
SOURCE: zfs-diagnosis, REV: 1.0
EVENT-ID: 9da778a7-a828-c88a-d679-c9a7873f4808
DESC: A ZFS device failed.  Refer to http://sun.com/msg/ZFS-8000-D3 for more information.
AUTO-RESPONSE: No automated response will occur.
IMPACT: Fault tolerance of the pool may be compromised.
REC-ACTION: Run 'zpool status -x' and replace the bad device.
```

Weitere Informationen zum Austauschen fehlerhafter Geräte finden Sie unter „Reparieren eines ausgefallenen Geräts“ auf Seite 301.

Löschen von Gerätefehlern im Speicher-Pool

Wenn ein Datenspeichergerät aufgrund von Fehlern, die in der Ausgabe von `zpool status` aufgeführt werden, außer Betrieb genommen wird, können Sie diese Fehler mithilfe des Befehls `zpool clear` löschen.

Wenn keine Argumente angegeben werden, löscht der Befehl alle Gerätefehler eines Pools. Beispiel:

```
# zpool clear tank
```

Wenn ein Gerät oder mehrere Geräte angegeben werden, löscht der Befehl nur die zu den betreffenden Geräten gehörenden Fehler. Beispiel:

```
# zpool clear tank c1t0d0
```

Weitere Informationen zum Löschen von `zpool`-Fehlern finden Sie unter „Löschen vorübergehender Fehler“ auf Seite 305.

Austauschen von Geräten in einem Speicher-Pool

Mit dem Befehl `zpool replace` können Sie Datenspeichergeräte in einem Speicher-Pool austauschen.

Wenn Sie in einem Pool mit Redundanz ein Datenspeichergerät an der gleichen Stelle durch ein anderes Datenspeichergerät ersetzen, brauchen Sie nur das ersetzte Datenspeichergerät

anzugeben. ZFS erkennt, dass sich an der betreffenden Stelle jetzt ein anderes Datenspeichergerät befindet. Wenn Sie beispielsweise eine ausgefallene Festplatte (`c1t1d0`) durch Auswechseln an der gleichen Stelle ersetzen wollen, sollten Sie eine Syntax ähnlich der folgenden verwenden:

```
# zpool replace tank c1t1d0
```

Wenn Sie ein Datenspeichergerät in einem Speicher-Pool ohne Redundanz, das nur ein Datenspeichergerät enthält, auswechseln wollen, müssen Sie beide Datenspeichergeräte angeben. Beispiel:

```
# zpool replace tank c1t1d0 c1t2d0
```

Wenn Sie eine Festplatte in einem ZFS-Root-Pool ersetzen, lesen Sie unter „So ersetzen Sie eine Festplatte im ZFS-Root-Pool“ auf Seite 167 nach.

Zum Austauschen von Datenträgern gehören folgende grundlegenden Schritte:

- Nehmen Sie den Datenträger wenn nötig mit dem Befehl `zpool offline` außer Betrieb.
- Bauen Sie die zu erstetzende Festplatte aus.
- Setzen Sie die Ersatzfestplatte ein.
- Führen Sie den Befehl `zpool replace` aus. Beispiel:

  ```
  # zpool replace tank c1t1d0
  ```

- Nehmen Sie den Datenträger mit dem Befehl `zpool online` in Betrieb.

Bei manchen Systemen, wie etwa bei Sun Fire x4500, muss eine Festplatte vor der Außerbetriebnahme dekonfiguriert werden. Wenn Sie bei diesem System lediglich eine Festplatte an ein und derselben Steckplatzposition austauschen, genügt es, den Befehl `zpool replace` wie oben beschrieben auszuführen.

Ein Beispiel für das Austauschen eines Datenträgers auf diesem System finden Sie in Beispiel 11–1.

Beachten Sie beim Auswechseln von Datenspeichergeräten in einem ZFS-Speicher-Pool Folgendes:

- Wenn Sie die Eigenschaft `autoreplace` des Pools auf „On“ setzen, wird jedes neue Datenspeichergerät, das sich an der physischen Stelle eines zuvor zum Pool gehörenden Datenspeichergeräts befindet, automatisch formatiert und ohne den Befehl `zpool replace` ersetzt. Dieses Leistungsmerkmal ist möglicherweise nicht auf jeder Art von Hardware verfügbar.
- Die Kapazität des Austauschgeräts muss mindestens gleich der Mindestkapazität aller Datenspeichergeräte in einer Datenspiegelungs- bzw. RAID-Z-Konfiguration sein.

- Wenn die Kapazität des Austauschgeräts höher ist, wird nach Abschluss des Austausches die Pool-Kapazität erhöht. Gegenwärtig müssen Sie das Pool exportieren und wieder importieren, wenn die Kapazitätserhöhung ersichtlich sein soll. Beispiel:

```
# zpool list tank
NAME   SIZE   USED  AVAIL    CAP  HEALTH  ALTROOT
tank  16.8G    94K  16.7G     0%  ONLINE  -
# zpool replace tank c0t0d0 c0t4d0
# zpool list tank
NAME   SIZE   USED  AVAIL    CAP  HEALTH  ALTROOT
tank  16.8G   112K  16.7G     0%  ONLINE  -
# zpool export tank
# zpool import tank
# zpool list tank
NAME   SIZE   USED  AVAIL    CAP  HEALTH  ALTROOT
tank  33.9G   114K  33.9G     0%  ONLINE  -
```

  Weitere Informationen zum Importieren und Exportieren von Pools finden Sie unter „Migrieren von ZFS-Speicher-Pools“ auf Seite 105.

- Derzeit müssen auch beim Vergrößern einer vorhandenen, zu einem Speicher-Pool gehörenden LUN die Export- und Importschritte ausgeführt werden, um die erweiterte Festplattenkapazität sichtbar zu machen.
- Das Austauschen mehrerer Festplatten in einem großen Pool ist zeitaufwändig, da die Daten mithilfe von Resilvering auf die neuen Festplatten aufgespielt werden müssen. Darüber hinaus sollten Sie zwischen dem Austausch von Festplatten den Befehl `zpool scrub` ausführen, um sicherzustellen, dass die Austauschgeräte ordnungsgemäß funktionieren und Daten korrekt geschrieben wurden.
- Wenn eine ausgefallene Festplatte automatisch durch eine Hot-Spare-Festplatte ersetzt wurde, müssen Sie die Hot-Spare-Festplatte möglicherweise nach dem Ersetzen der ausgefallenen Festplatte abtrennen. Weitere Informationen zum Abtrennen von Hot-Spares finden Sie in „Aktivieren und Deaktivieren von Hot-Spares im Speicher-Pool“ auf Seite 90.

Weitere Informationen zum Austauschen von Geräten finden Sie unter „Reparieren eines ausgefallenen Geräts“ auf Seite 301 sowie „Ersetzen oder Reparieren eines beschädigten Geräts“ auf Seite 303.

Zuweisen von Hot-Spares im Speicher-Pool

Mithilfe der Hot-Spare-Funktion können Sie Datenträger ermitteln, die zum Ersetzen eines ausgefallenen bzw. fehlerhaften Geräts in einem bzw. mehreren Speicher-Pools verwendet werden können. Das Vorsehen eines Datenspeichergeräts als *Hot-Spare* bedeutet, dass das betreffende Gerät im Pool nicht aktiv ist, sondern bei Ausfall eines aktiven Datenspeichergeräts im Pool dieses automatisch ersetzt).

Datenspeichergeräte können mit den folgenden Methoden als Hot-Spares vorgesehen werden:

- bei der Erstellung eines Pools mit dem Befehl `zpool create`,
- nach der Erstellung eines Pools mit dem Befehl `zpool add`,
- Hot-Spares können von mehreren Pools gemeinsam genutzt werden.

Sie können Datenspeichergeräte bei der Erstellung eines Pools als Hot-Spares zuweisen. Beispiel:

```
# zpool create zeepool mirror c1t1d0 c2t1d0 spare c1t2d0 c2t2d0
# zpool status zeepool
pool: zeepool
 state: ONLINE
 scrub: none requested
config:

        NAME         STATE     READ WRITE CKSUM
        zeepool      ONLINE       0     0     0
          mirror     ONLINE       0     0     0
            c1t1d0   ONLINE       0     0     0
            c2t1d0   ONLINE       0     0     0
        spares
          c1t2d0     AVAIL
          c2t2d0     AVAIL
```

Sie können Hot-Spares nach der Erstellung eines Pools durch Hinzufügen zum Pool zuweisen. Beispiel:

```
# zpool add zeepool spare c1t3d0 c2t3d0
# zpool status zeepool
pool: zeepool
 state: ONLINE
 scrub: none requested
config:

        NAME         STATE     READ WRITE CKSUM
        zeepool      ONLINE       0     0     0
          mirror     ONLINE       0     0     0
            c1t1d0   ONLINE       0     0     0
            c2t1d0   ONLINE       0     0     0
        spares
          c1t3d0     AVAIL
          c2t3d0     AVAIL
```

Als Hot-Spares vorgesehene Datenspeichergeräte können von mehreren Pools gemeinsam genutzt werden. Beispiel:

```
# zpool create zeepool mirror c1t1d0 c2t1d0 spare c1t2d0 c2t2d0
# zpool create tank raidz c3t1d0 c4t1d0 spare c1t2d0 c2t2d0
```

Hot-Spares können mit dem Befehl `zpool remove` aus einem Speicher-Pool entfernt werden. Beispiel:

```
# zpool remove zeepool c1t2d0
# zpool status zeepool
pool: zeepool
 state: ONLINE
 scrub: none requested
config:

        NAME         STATE     READ WRITE CKSUM
        zeepool      ONLINE       0     0     0
          mirror     ONLINE       0     0     0
            c1t1d0   ONLINE       0     0     0
            c2t1d0   ONLINE       0     0     0
        spares
          c1t3d0     AVAIL
```

Ein Hot-Spare kann nur entfernt werden, wenn es gegenwärtig nicht vom Speicher-Pool verwendet wird.

Beachten Sie beim Arbeiten mit ZFS-Hot-Spares Folgendes:

- Zurzeit können mit dem Befehl `zpool remove` nur Hot-Spares und Cache-Geräte entfernt werden.
- Es sollten nur Festplatten als Hot-Spares zu einem Pool hinzugefügt werden, deren Kapazität gleich oder größer als die größte Festplatte im Pool ist. Sie können Festplatten mit geringerer Kapazität zwar hinzufügen, sollten jedoch beachten, dass bei Aktivierung der Festplatte mit der geringeren Kapazität (automatisch oder mithilfe des Befehls `zpool replace`) der Vorgang mit einer Fehlermeldung wie der Folgenden abbricht:

  `cannot replace` *disk3* `with` *disk4*`: device is too small`
- Hot-Spares können von mehreren Pools gemeinsam genutzt werden. Es ist jedoch nicht möglich, ein Pool mit einem belegten, freigegebenen Hot-Spare zu exportieren, es sei denn, sie verwenden die Export-Option `f- (force)`. Dieses Verhalten verhindert eine potenzielle Datenbeschädigung durch Export eines Pools mit benutztem, freigegebenem Hot-Spare und einen versuchten Zugriff eines anderen Pools auf das gemeinsame Hot-Spare aus dem exportierten Pool. Wenn Sie ein Pool mit einem belegten, freigegebenen Hot-Spare mithilfe der Option `-f` exportieren, müssen Sie sich darüber im Klaren sein, dass diese Situation zu Datenbeschädigung führen kann, falls ein anderes Pool das belegte, gemeinsame Hot-Spare zu aktivieren versucht.

Aktivieren und Deaktivieren von Hot-Spares im Speicher-Pool

Hot-Spares können mit einer der folgenden Methoden aktiviert werden:

- Manueller Austausch - Austausch eines ausgefallenen Geräts im Speicher-Pool durch ein Hot-Spare mit dem Befehl `zpool replace`.
- Automatischer Austausch – Beim Empfang einer Ausfallsmeldung überprüft ein FMA-Agent das Pool, um zu sehen, ob in ihm Hot-Spares zur Verfügung stehen. Wenn dies der Fall ist, ersetzt er das fehlerhafte Gerät durch ein verfügbares Ersatzgerät.

  Falls ein gegenwärtig in Verwendung befindliches Hot-Spare ausfällt, trennt der Agent das Ersatzgerät ab und macht den Austausch somit rückgängig. Dann versucht der Agent, das Datenspeichergerät mit einem anderen Hot-Spare zu ersetzen, falls dies möglich ist. Diese Funktion ist gegenwärtig Beschränkungen unterworfen, da das ZFS-Diagnoseprogramm nur Fehler meldet, wenn ein Datenspeichergerät aus dem System verschwindet.

  Wenn Sie ein ausgefallenes Datenspeichergerät physisch durch ein aktives Hot-Spare ersetzen, können Sie das ursprüngliche, aber ersetzte Gerät wieder aktivieren, indem Sie das Hot-Spare mithilfe des Befehls `zpool detach` abtrennen. Wenn Sie die Eigenschaft `autoreplace` des Pools auf „On“ setzen, wird das Hot-Spare automatisch wieder zum Spare-Pool abgetrennt, sobald das neue Datenspeichergerät eingesetzt und die Inbetriebnahme abgeschlossen ist.

Mit dem Befehl `zpool replace` können Sie ein Datenspeichergerät manuell durch ein Hot-Spare ersetzen. Beispiel:

```
# zpool replace zeepool c2t1d0 c2t3d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Fri Aug 28 14:16:04 2009
config:

        NAME           STATE     READ WRITE CKSUM
        zeepool        ONLINE       0     0     0
          mirror       ONLINE       0     0     0
            c1t2d0     ONLINE       0     0     0
            spare      ONLINE       0     0     0
              c2t1d0   ONLINE       0     0     0
              c2t3d0   ONLINE       0     0     0  76.5K resilvered
        spares
          c2t3d0      INUSE     currently in use

errors: No known data errors
```

Ein fehlerhaftes Datenspeichergerät wird automatisch ausgetauscht, wenn ein Hot-Spare verfügbar ist. Beispiel:

```
# zpool status -x
  pool: zeepool
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
```

```
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-D3
 scrub: resilver completed after 0h12m with 0 errors on Fri Aug 28 14:18:16 2009
config:

        NAME                 STATE     READ WRITE CKSUM
        zeepool              DEGRADED     0     0     0
          mirror             DEGRADED     0     0     0
            c1t2d0           ONLINE       0     0     0
            spare            DEGRADED     0     0     0
              c2t1d0         UNAVAIL      0     0     0  cannot open
              c2t3d0         ONLINE       0     0     0  58.5K resilvered
        spares
          c2t3d0             INUSE     currently in use

errors: No known data errors
```

Gegenwärtig können Hot-Spares mit drei Methoden deaktiviert werden:

- Entfernen des Hot-Spares durch Löschen aus dem Speicher-Pool
- Ersetzen des ursprünglichen Datenspeichergeräts durch ein Hot-Spare
- Dauerhaftes „Einlagern" (Swap-in) des Hot-Spares

Mit dem Befehl `zpool detach` können Sie nach dem Austauschen eines fehlerhaften Datenspeichergeräts das Hot-Spare wieder dem verfügbaren Hot-Spare-Satz zuweisen. Beispiel:

```
# zpool detach zeepool c2t3d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: resilver completed with 0 errors on Fri Aug 28 14:21:02 2009
config:

        NAME               STATE     READ WRITE CKSUM
        zeepool            ONLINE       0     0     0
          mirror           ONLINE       0     0     0
            c1t2d0         ONLINE       0     0     0
            c2t1d0         ONLINE       0     0     0
        spares
          c2t3d0           AVAIL

errors: No known data errors
```

Eigenschaften von ZFS-Speicher-Pools

Mit dem Befehl `zpool get` können Sie Informationen zu den Pool-Eigenschaften abrufen. Beispiel:

```
# zpool get all mpool
NAME   PROPERTY       VALUE                  SOURCE
mpool  size           33.8G                  -
mpool  used           5.91G                  -
mpool  available      27.8G                  -
mpool  capacity       17%                    -
mpool  altroot        -                      default
mpool  health         ONLINE                 -
mpool  guid           17361998391267837263   -
mpool  version        10                     default
mpool  bootfs         mpool/ROOT/zfs2BE      local
mpool  delegation     on                     default
mpool  autoreplace    on                     local
mpool  cachefile      -                      default
mpool  failmode       continue               local
mpool  listsnapshots  on                     default
```

Mit dem Befehl `zpool set` lassen sich die Pool-Eigenschaften festlegen. Beispiel:

```
# zpool set autoreplace=on mpool
# zpool get autoreplace mpool
NAME  PROPERTY     VALUE    SOURCE
mpool autoreplace  on       default
```

TABELLE 4–1 Beschreibungen der Eigenschaften für ZFS-Pools

Eigenschaft	Typ	Standardwert	Beschreibung
`altroot`	Zeichenkette	off	Das alternative Root-Verzeichnis. Ist diese Eigenschaft gesetzt, wird dieses Verzeichnis jedem Einhängepunkt innerhalb des Pools vorangestellt. Diese Eigenschaft ist nützlich, um ein unbekanntes Pool zu untersuchen, wenn die Einhängepunkte nicht vertrauenswürdig sind oder, wenn in einer alternativen Boot-Umgebung die üblichen Pfade nicht gültig sind.
`available`	Zahl	entf.	Schreibgeschützter Wert, der die Menge des nicht belegten Speicherplatzes im Pool angibt. Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`avail`) referenziert werden.

TABELLE 4–1 Beschreibungen der Eigenschaften für ZFS-Pools *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`autoreplace`	Boolesch	`off`	Regelt den automatischen Austausch von Speichergeräten. Wenn diese Eigenschaft auf „off“ gesetzt ist, muss das Auswechseln von Speichergeräten vom Administrator mithilfe des Befehls `zpool replace` initiiert werden. Wenn diese Eigenschaft auf „on“ gesetzt ist, wird das neue Speichergerät an der physischen Adresse des vorherigen Speichergeräts im Pool automatisch formatiert und in das Pool eingebunden. Standardeinstellung ist „off“. Diese Eigenschaft kann auch über ihre Kurzbezeichnung (`replace`) referenziert werden.
`bootfs`	Boolesch	entf.	Das boot-fähige Standard-Dataset für den Root-Pool. Diese Eigenschaft wird in der Regel vom Installations- bzw. Upgrade-Programm gesetzt.
`capacity`	Zahl	entf.	Schreibgeschützter Wert, der die Menge des belegten Speicherplatzes im Pool als Verhältnis zur Gesamtkapazität in Prozent angibt. Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`cap`) referenziert werden.
`delegation`	Boolesch	`on`	Bestimmt, ob einem Benutzer ohne ausreichende Berechtigungen die für das Dataset festgelegten Zugriffsrechte erteilt werden können. Weitere Informationen finden Sie in Kapitel 9, „Delegierte ZFS-Administration“.
`failmode`	Zeichenkette	`wait`	Regelt das Systemverhalten im Fall eines Pool-Desasters. Dieser Umstand ist in der Regel das Ergebnis eines Konnektivitätsverlusts mit den zugrunde liegenden Speichergeräten oder eines Ausfalls sämtlicher Geräte im Pool. Das Verhalten in einem solchen Fall wird durch einen der folgenden Werte bestimmt: ■ `wait` – Blockiert sämtliche E/A-Zugriffe, bis die Gerätekonnektivität wiederhergestellt ist und die Fehler mit dem Befehl `zpool clear` zurückgesetzt wurden. Dies ist das Standardverhalten. ■ `continue` – Gibt bei jeder neuen E/A-Schreibanfrage einen E/A-Fehler zurück, lässt aber Lesezugriffe auf die übrigen fehlerfreien Speichergeräte zu. Schreibanfragen, die noch an die Festplatte übergeben werden müssen, werden blockiert. Nach Wiederherstellung der Verbindung oder Ersetzen des Geräts müssen die Fehler mit dem Befehl `zpool clear` zurückgesetzt werden. ■ `panic` – Gibt eine Meldung an die Konsole aus und generiert einen Systemspeicherabzug.

TABELLE 4–1 Beschreibungen der Eigenschaften für ZFS-Pools *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`guid`	Zeichenkette	entf.	Schreibgeschützte Eigenschaft und eindeutige Kennzeichnung des Pools.
`health`	Zeichenkette	entf.	Schreibgeschützte Eigenschaft, die den aktuellen Zustand des Pools angibt. Dabei bestehen folgende Möglichkeiten: ONLINE, DEGRADED, FAULTED, OFFLINE, REMOVED oder UNAVAIL.
`listsnapshots`	Zeichenkette	off	Kontrolliert, ob Snapshot-Informationen, die mit diesem Pool in Verbindung stehen, mit dem Befehl `zfs list` angezeigt werden. Wenn diese Eigenschaft deaktiviert wird, können die Snapshot-Informationen mit dem Befehl `zfs list -t snapshot` angezeigt werden. Der Standardwert ist `on`.
`size`	Zahl	entf.	Schreibgeschützte Eigenschaft, die die Gesamtkapazität des Speicher-Pools angibt.
`used`	Zahl	entf.	Schreibgeschützter Wert, der die Menge des belegten Speicherplatzes im Pool angibt.
`version`	Zahl	entf.	Die aktuelle Version des Pools. Der Wert dieser Eigenschaft kann erhöht, niemals aber verringert werden. Zum Aktualisieren von Pools wird grundsätzlich die Methode mit dem Befehl `zpool upgrade` empfohlen. Diese Eigenschaft bietet sich dann an, wenn aus Gründen der Abwärtskompatibilität eine spezifische Version benötigt wird. Diese Eigenschaft kann auf jede Zahl zwischen 1 und der aktuellen Version laut Ausgabe des Befehls `zpool upgrade -v` gesetzt werden. Der Wert `current` ist ein Alias für die neueste unterstützte Version.

Abfragen des Status von ZFS-Speicher-Pools

Mithilfe des Befehls `zpool list` können Sie mit unterschiedlichen Methoden Informationen zum Pool-Status abrufen. Die verfügbaren Informationen unterteilen sich im Allgemeinen in drei Kategorien: grundlegende Informationen zur Auslastung, E/A-Statistiken und Informationen zum Funktionsstatus. In diesem Abschnitt werden alle drei Kategorien dieser Informationen zu Speicher-Pools behandelt.

- „Anzeigen grundlegender Informationen zu ZFS-Speicher-Pools“ auf Seite 96
- „Anzeigen von E/A-Statistikinformationen zu ZFS-Speicher-Pools“ auf Seite 99
- „Ermitteln des Funktionsstatus von ZFS-Speicher-Pools“ auf Seite 101

Anzeigen grundlegender Informationen zu ZFS-Speicher-Pools

Mit dem Befehl `zpool list` können Sie grundlegende Pool-Informationen anzeigen.

Anzeigen von Informationen zu allen Speicher-Pools

Ohne Argumente zeigt der Befehl alle Felder für alle im System verfügbaren Pools an. Beispiel:

```
# zpool list
NAME                    SIZE    USED   AVAIL    CAP  HEALTH     ALTROOT
tank                   80.0G   22.3G   47.7G    28%  ONLINE     -
dozer                   1.2T    384G    816G    32%  ONLINE     -
```

Diese Ausgabe zeigt die folgenden Informationen an:

`NAME` Der Name des Pools.

`SIZE` Die Gesamtkapazität des Pools, entspricht der Summe der Speicherkapazität aller virtuellen Geräte der obersten Hierarchieebene.

`USED` Der von allen Datasets und internen Metadaten belegte Speicherplatz. Bitte beachten Sie, dass sich dieser Wert von dem auf Dateisystemebene gemeldeten Speicherplatz unterscheidet.

Weitere Informationen zum Ermitteln des verfügbaren Dateisystemspeicherplatzes finden Sie unter „ZFS-Speicherplatzverwaltung“ auf Seite 56.

`AVAIL` Der Wert des nicht belegten Speicherplatzes im Pool.

`CAPACITY (CAP)` Der Wert des belegten Speicherplatzes als Verhältnis zur Gesamtkapazität, in Prozent.

`HEALTH` Der gegenwärtige Funktionsstatus des Pools.

Weitere Informationen zum Pool-Status finden Sie unter „Ermitteln des Funktionsstatus von ZFS-Speicher-Pools“ auf Seite 101.

`ALTROOT` Das alternative Root-Verzeichnis des Pools (falls vorhanden).

Weitere Informationen zu Speicher-Pools mit alternativem Root-Verzeichnis finden Sie unter „Verwenden von ZFS-Speicher-Pools mit alternativem Root-Verzeichnis“ auf Seite 288.

Durch Angabe eines Pool-Namens können Sie sich auch Statistiken zu diesem bestimmten Pool anzeigen lassen. Beispiel:

```
# zpool list tank
NAME                    SIZE    USED   AVAIL    CAP  HEALTH     ALTROOT
tank                   80.0G   22.3G   47.7G    28%  ONLINE     -
```

Anzeigen spezifischer Speicher-Pool-Statistikinformationen

Mithilfe der Option `-o` können Sie sich spezifische Statistikinformationen anzeigen lassen. Damit können Sie benutzerdefinierte Berichte erstellen oder sich gewünschte Informationen anzeigen lassen. Mit der folgenden Syntax wird beispielsweise nur der Name und die Speicherkapazität jedes Pools angezeigt:

```
# zpool list -o name,size
NAME                    SIZE
tank                   80.0G
dozer                   1.2T
```

Die Spaltentitel sind unter „Anzeigen von Informationen zu allen Speicher-Pools“ auf Seite 96 erläutert.

Verwenden von Ausgaben von ZFS-Speicher-Pools für Skripten

Die Standardausgabe des Befehls `zpool list` dient der Lesbarkeit am Bildschirm und ist für Shell-Skripten nicht zu verwenden. Zur Unterstützung programmatischer Verwendungen dieses Befehls kann mithilfe der Option `-H` die Ausgabe der Spaltentitel unterdrückt werden, und die einzelnen Felder werden durch Leerzeichen statt durch Tabulatoren getrennt. So rufen Sie beispielsweise eine einfache Liste aller Pool-Namen im System ab:

```
# zpool list -Ho name
tank
dozer
```

Hier ist ein weiteres Beispiel:

```
# zpool list -H -o name,size
tank   80.0G
dozer  1.2T
```

Anzeige des Befehlsprotokolls von ZFS-Speicher-Pools

ZFS protokolliert automatisch `zfs`- und `zpool`-Befehle, durch die Pool-Zustandsinformationen geändert werden. Diese Informationen können mit dem Befehl `zpool history` angezeigt werden.

Die folgende Syntax zeigt beispielsweise die Befehlsausgabe für den Root-Pool.

```
# zpool history
History for 'rpool':
2009-05-07.13:51:00 zpool create -f -o failmode=continue -R /a -m legacy -o cachefile=
/tmp/root/etc/zfs/zpool.cache rpool c1t0d0s0
2009-05-07.13:51:01 zfs set canmount=noauto rpool
2009-05-07.13:51:02 zfs set mountpoint=/rpool rpool
2009-05-07.13:51:02 zfs create -o mountpoint=legacy rpool/ROOT
2009-05-07.13:51:03 zfs create -b 8192 -V 2048m rpool/swap
2009-05-07.13:51:04 zfs create -b 131072 -V 1024m rpool/dump
2009-05-07.13:51:09 zfs create -o canmount=noauto rpool/ROOT/snv_114
2009-05-07.13:51:10 zpool set bootfs=rpool/ROOT/snv_114 rpool
2009-05-07.13:51:10 zfs set mountpoint=/ rpool/ROOT/snv_114
2009-05-07.13:51:11 zfs set canmount=on rpool
2009-05-07.13:51:12 zfs create -o mountpoint=/export rpool/export
2009-05-07.13:51:12 zfs create rpool/export/home
```

Sie können auf Ihrem System eine ähnliche Ausgabe verwenden, um den *genauen* Satz der ZFS-Befehle zu identifizieren, der bei der Behebung eines Fehlers ausgeführt wurde.

Das Verlaufsprotokoll zeichnet sich durch die folgenden Merkmale aus:

- Das Protokoll kann nicht deaktiviert werden.
- Das Protokoll wird persistent gespeichert, es wird also neustartübergreifend geführt.
- Das Protokoll wird in Form eines Ringpuffers implementiert. Die Mindestgröße beträgt 128 Kbytes. Die maximale Größe beträgt 32 Mbytes.
- Für kleinere Pools ist die maximale Größe auf 1 % der Pool-Größe beschränkt, wobei die *Größe* zum Zeitpunkt der Pool-Erstellung bestimmt wird.
- Erfordert keine Verwaltung, d. h., eine Anpassung der Protokollgröße bzw. eine Änderung des Speicherorts sind nicht erforderlich.

Verwenden Sie zur Identifizierung des Befehlsprotokolls eines bestimten Speicher-Pools in etwa folgende Syntax:

```
# zpool history mypool
History for 'mypool':
2009-06-02.10:56:54 zpool create mypool mirror c0t4d0 c0t5d0
2009-06-02.10:57:31 zpool add mypool spare c0t6d0
2009-06-02.10:57:54 zpool offline mypool c0t5d0
2009-06-02.10:58:02 zpool online mypool c0t5d0
```

Verwenden Sie die Option `-l` zum Anzeigen eines langen Formats mit Benutzernamen, Hostnamen und Angabe der Zone, in der die Operation ausgeführt wurde. Beispiel:

```
# zpool history -l mypool
History for 'mypool':
2009-06-02.10:56:54 zpool create mypool mirror c0t4d0 c0t5d0 [user root on neo:global]
```

```
2009-06-02.10:57:31 zpool add mypool spare c0t6d0 [user root on neo:global]
2009-06-02.10:57:54 zpool offline mypool c0t5d0 [user root on neo:global]
2009-06-02.10:58:02 zpool online mypool c0t5d0 [user root on neo:global]
```

Verwenden Sie die Option `-i` zum Anzeigen interner Ereignisinformationen, die bei der Diagnose behilflich sein können. Beispiel:

```
# zpool history -i mypool
History for 'mypool':
2009-06-02.10:56:54 zpool create mypool mirror c0t4d0 c0t5d0
2009-06-02.10:57:31 zpool add mypool spare c0t6d0
2009-06-02.10:57:54 zpool offline mypool c0t5d0
2009-06-02.10:58:02 zpool online mypool c0t5d0
2009-06-02.11:02:20 [internal create txg:23] dataset = 24
2009-06-02.11:02:20 [internal property set txg:24] mountpoint=/data dataset = 24
2009-06-02.11:02:20 zfs create -o mountpoint=/data mypool/data
2009-06-02.11:02:34 [internal create txg:26] dataset = 30
2009-06-02.11:02:34 zfs create mypool/data/datab
```

Anzeigen von E/A-Statistikinformationen zu ZFS-Speicher-Pools

Mit dem Befehl `zpool iostat` können Sie E/A-Statistikinformationen für einen Pool bzw. ein virtuelles Datenspeichergerät abrufen. Dieser Befehl zeigt ähnlich wie der Befehl `iostat` eine statische „Momentaufnahme" aller bisherigen E/A-Aktivitäten sowie für jedes bestimmte Zeitintervall aktualisierte Statistikinformationen an. Es werden die folgenden Statistikinformationen ausgegeben:

`USED CAPACITY` — Die Kapazität der gegenwärtig im Pool bzw. Gerät gespeicherten Daten. Aufgrund interner Implementierungsaspekte unterscheidet sich dieser Wert von dem für eigentliche Dateisysteme verfügbaren Speicherplatz um einen geringen Betrag.

Weitere Informationen zum Unterschied zwischen Pool- und Dataset-Speicherplatz finden Sie in „ZFS-Speicherplatzverwaltung" auf Seite 56.

`AVAILABLE CAPACITY` — Die Kapazität des im Pool bzw. Gerät verfügbaren Speicherplatzes. Wie beim Parameter `USED CAPACITY` unterscheidet sich dieser Wert von dem für Datasets verfügbaren Speicherplatz um einen geringen Betrag.

`READ OPERATIONS` — Die Anzahl der zum Pool bzw. Gerät gesendeten E/A-Leseoperationen einschließlich Metadaten-Operationen.

`WRITE OPERATIONS`	Die Anzahl der zum Pool bzw. Gerät gesendeten E/A-Schreiboperationen.
`READ BANDWIDTH`	Die Bandbreite aller Leseoperationen (einschließlich Metadaten) in Einheiten pro Sekunde.
`WRITE BANDWIDTH`	Die Bandbreite aller Schreiboperationen in Einheiten pro Sekunde.

Anzeigen globaler Pool-Statistikinformationen

Ohne Optionen zeigt der Befehl `zpool iostat` die bisher aufgelaufenen Statistikinformationen für alle Pools im System seit dem Hochfahren des Systems an. Beispiel:

```
# zpool iostat
               capacity     operations    bandwidth
pool         used  avail   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
tank         100G  20.0G   1.2M   102K   1.2M  3.45K
dozer       12.3G  67.7G   132K  15.2K  32.1K  1.20K
```

Da die angezeigten Statistikinformationen seit dem Hochfahren des Systems aufgelaufen sind, kann es sein, dass die Bandbreite bei relativ geringer Auslastung des Pools gering erscheint. Durch Angabe eines Zeitintervalls erhalten Sie ein realistischeres Bild der aktuellen Bandbreite. Beispiel:

```
# zpool iostat tank 2
               capacity     operations    bandwidth
pool         used  avail   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
tank         100G  20.0G   1.2M   102K   1.2M  3.45K
tank         100G  20.0G    134      0  1.34K      0
tank         100G  20.0G     94    342  1.06K   4.1M
```

In diesem Beispiel zeigt der Befehl alle zwei Sekunden lang die Statistikinformationen zur Auslastung für den Pool `tank` solange an, bis Sie die Tastenkombination CTRL-C drücken. Als Alternative können Sie einen zusätzlichen Zählparameter (`count`) angeben, der die Ausführung des Befehls nach einer bestimmten Durchlaufanzahl beendet. So gibt `zpool iostat 2 3` beispielsweise dreimal alle zwei Sekunden (also insgesamt sechs Sekunden lang) eine Übersicht aus. Wenn nur ein einziger Pool vorhanden ist, werden die Statistikinformationen in aufeinander folgenden Zeilen angezeigt. Sind mehrere Pools vorhanden, werden die Werte für die einzelnen Pools zur besseren Lesbarkeit durch gestrichelte Linien getrennt.

Anzeigen von Statistikinformationen zu virtuellen Geräten

Neben den globalen E/A-Statistikinformationen für einen Pool kann der Befehl `zpool iostat` außerdem Statistikinformationen für bestimmte virtuelle Datenspeichergeräte anzeigen. Mit

diesem Befehl können Sie unverhältnismäßig langsame Datenspeichergeräte identifizieren oder die Verteilung der von ZFS generierten E/A-Operationen überprüfen. Zum Abrufen der vollständigen virtuellen Gerätestruktur sowie aller E/A-Statistikinformationen können Sie den Befehl `zpool iostat -v` nutzen. Beispiel:

```
# zpool iostat -v
               capacity     operations    bandwidth
tank         used  avail   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
mirror      20.4G  59.6G      0     22      0  6.00K
  c1t0d0        -      -      1    295  11.2K   148K
  c1t1d0        -      -      1    299  11.2K   148K
----------  -----  -----  -----  -----  -----  -----
total       24.5K   149M      0     22      0  6.00K
```

Bitte beachten Sie zwei wichtige Aspekte, wenn Sie E/A-Statistikinformationen für virtuelle Geräte abrufen:

- Informationen zur Speicherplatzbelegung sind nur für virtuelle Geräte der obersten Hierarchieebene verfügbar. Die Art und Weise der Speicherplatzzuweisung bei virtuellen Geräten mit Datenspiegelung und RAID-Z ist implementierungsspezifisch und kann nicht in Form einer einzelnen Zahl ausgedrückt werden.
- Zweitens kann es sein, dass die einzelnen Werte nicht die erwartete Summe ergeben. Insbesondere sind Werte bei Geräten mit Datenspiegelung und RAID-Z nicht genau gleich. Diese Unterschiede sind besonders nach der Erstellung eines Pools bemerkbar, da ein erheblicher Teil von E/A-Operationen infolge der Pool-Erstellung direkt an den Datenträgern ausgeführt wird, was auf der Datenspiegelungsebene nicht berücksichtigt wird. Mit der Zeit gleichen sich diese Wert schrittweise an; defekte, nicht reagierende bzw. außer Betrieb genommen Geräte können die Symmetrie jedoch beeinträchtigen.

Bei der Untersuchung von Statistikinformationen zu virtuellen Geräten können Sie die gleichen Optionen (Zeitintervall und Zählparameter) verwenden.

Ermitteln des Funktionsstatus von ZFS-Speicher-Pools

ZFS bietet eine integrierte Methode zur Untersuchung der ordnungsgemäßen Funktion von Pools und Datenspeichergeräten. Der Funktionsstatus eines Pools wird aus dem Funktionsstatus aller seiner Datenspeichergeräte ermittelt. Diese Statusinformationen werden mit dem Befehl `zpool status` angezeigt. Darüber hinaus werden potenzielle Pool- und Geräteausfälle von `fmd` gemeldet und auf der Systemkonsole angezeigt (und in der Datei `/var/adm/messages` protokolliert). In diesem Abschnitt wird die Ermittlung des Funktionsstatus von Pools und Datenspeichergeräten erläutert. Dieses Kapitel enthält jedoch keine Informationen zur Reparatur fehlerhafter Pools bzw. Wiederherstellen des

Normalbetriebs eines Pools. Weitere Informationen zur Fehlerbehebung und Datenwiederherstellung finden Sie in Kapitel 11, „Problembehebung und Datenwiederherstellung in ZFS“.

Datenspeichergeräte können sich in einem der folgenden Zustände befinden:

`ONLINE`	Das Datenspeichergerät arbeitet normal. Obwohl zeitweilige Übergangsfehler auftreten können, arbeitet das Gerät sonst einwandfrei.
`DEGRADED`	Am virtuellen Gerät ist ein Fehler aufgetreten, es funktioniert jedoch noch Dieser Zustand tritt am Häufigsten auf, wenn in einer RAID-Z-Konfiguration ein oder mehrere Datenspeichergeräte nicht mehr verfügbar sind. Die Fehlertoleranz des Pools kann beeinträchtigt werden, da ein Ausfall eines weiteren Geräts nicht behebbar sein könnte.
`FAULTED`	Auf das virtuelle Gerät kann nicht zugegriffen werden. Dieser Status gibt normalerweise einen Totalausfall des Geräts an, bei dem ZFS mit dem Gerät keine Daten mehr austauschen kann. Wenn sich ein virtuelles Gerät der obersten Hierarchieebene in diesem Status befindet, kann auf den gesamten Pool nicht mehr zugegriffen werden.
`OFFLINE`	Das virtuelle Gerät wurde vom Administrator außer Betrieb genommen.
`UNAVAILABLE`	Mit dem Gerät bzw. virtuellen Gerät kann nicht kommuniziert werden. In einigen Fällen gehen Pools mit Geräten im Status `UNAVAILABLE` in den Status `DEGRADED`. Wenn sich ein virtuelles Gerät der obersten Hierarchieebene in diesem Status befindet, können von diesem Pool keine Daten abgerufen werden.
`REMOVED`	Das Gerät wurde bei laufendem Systembetrieb physisch ausgebaut. Die Erkennung ausgebauter Geräte ist von der jeweiligen Hardware abhängig und wird möglicherweise nicht auf allen Plattformen unterstützt.

Der Funktionsstatus eines Pools wird aus dem Funktionsstatus aller seiner Datenspeichergeräte der obersten Hierarchieebene ermittelt. Befinden sich alle virtuellen Geräte eines Pools im Status `ONLINE`, besitzt der Pool ebenfalls den Status `ONLINE`. Befindet sich eines der virtuellen Geräte eines Pools im Status `DEGRADED` bzw. `UNAVAILABLE`, besitzt der Pool den Status `DEGRADED`. Wenn sich ein virtuelles Gerät der obersten Hierarchieebene im Status `FAULTED` bzw. `OFFLINE` befindet, besitzt der Pool den Status `FAULTED`. Auf einen Pool in diesem Status kann nicht zugegriffen werden. Es können erst wieder Daten abgerufen werden, wenn erforderliche Datenspeichergeräte verbunden bzw. repariert werden. Ein Pool im Status DEGRADED arbeitet zwar weiter, es kann jedoch sein, dass nicht das gleiche Niveau an Datenredundanz bzw. Datendurchsatz erreicht wird, wie wenn der Pool ordnungsgemäß arbeitet.

Grundlegender Funktionsstatus eines Speicher-Pools

Die einfachste Weise zum schnellen Abrufen des Funktionsstatus von Pools ist die Verwendung des Befehls `zpool status`:

```
# zpool status -x
all pools are healthy
```

Durch Anhängen eines Pool-Namens an den Befehl können Sie den Funktionsstatus dieses Pools überprüfen. Pools, die sich nicht im Status `ONLINE` befinden, sollten auf potenzielle Probleme untersucht werden (siehe folgender Abschnitt).

Ausführliche Informationen zum Funktionsstatus

Mithilfe der Option `-v` können Sie ausführlichere Informationen zum Funktionsstatus abrufen. Beispiel:

```
# zpool status -v tank
  pool: tank
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist
        for the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: none requested
config:

        NAME                STATE     READ WRITE CKSUM
        tank                DEGRADED     0     0     0
          mirror            DEGRADED     0     0     0
            c1t0d0          FAULTED      0     0     0  cannot open
            c1t1d0          ONLINE       0     0     0
errors: No known data errors
```

Diese Ausgabe enthält eine vollständige Beschreibung darüber, warum sich der Pool in seinem gegenwärtigen Funktionsstatus befindet. Es findet sich auch eine lesbare Erläuterung des Problems und ein Verweis auf einen Artikel in der Sun Knowledge Base, wenn Sie weitere Informationen dazu benötigen. Die Artikel der Sun Knowledge Base enthalten die aktuellsten Informationen zur Behebung eines bestimmten Problems. Mithilfe der aufgeführten ausführlichen Konfigurationsinformationen sollten Sie in der Lage sein, zu emitteln, welches Datenspeichergerät defekt ist und wie Sie den Pool reparieren können.

Im obigen Beispiel muss das defekte Datenspeichergerät ausgewechselt werden. Nach dem Austauschen des Geräts können Sie es mit dem Befehl `zpool online` wieder in Betrieb nehmen. Beispiel:

```
# zpool online tank c1t0d0
Bringing device c1t0d0 online
# zpool status -x
all pools are healthy
```

Wenn in einem Pool ein außer Betrieb genommenes Gerät vorhanden ist, kann es mithilfe der vom Befehl ausgegebenen Informationen identifiziert werden. Beispiel:

```
# zpool status -x
  pool: tank
 state: DEGRADED
status: One or more devices has been taken offline by the adminstrator.
        Sufficient replicas exist for the pool to continue functioning in a
        degraded state.
action: Online the device using 'zpool online' or replace the device with
        'zpool replace'.
 scrub: none requested
config:

        NAME         STATE     READ WRITE CKSUM
        tank         DEGRADED     0     0     0
          mirror     DEGRADED     0     0     0
            c1t0d0   ONLINE       0     0     0
            c1t1d0   OFFLINE      0     0     0

errors: No known data errors
```

In den Spalten `READ` und `WRITE` wird die Anzahl der für das betreffende Gerät gezählten E/A-Fehler angezeigt, und die Spalte `CKSUM` enthält die Anzahl der am Gerät aufgetretenen nicht behebbaren Prüfsummenfehler. Beide Fehlerzähler weisen auf einen wahrscheinlichen Geräteausfall hin, und Sie sollten Maßnahmen zur Behebung dieses Problems einleiten. Wenn für ein virtuelles Gerät der obersten Hierarchieebene Fehlerwerte größer als null angezeigt werdn, kann es sein, dass auf Daten teilweise nicht mehr zugegriffen werden kann Die Fehlerzähler weisen auf bekannte Datenfehler hin.

In der Befehlsausgabe des obigen Beispiels verursacht das außer Betrieb genommene Gerät keine Datenfehler.

Weitere Informationen zum Auffinden von Fehlern und Reparieren fehlerhafter Pools und Daten finden Sie in Kapitel 11, „Problembehebung und Datenwiederherstellung in ZFS“.

Migrieren von ZFS-Speicher-Pools

Gelegentlich kann es vorkommen, dass Speicher-Pools zwischen Rechnern transferiert werden müssen. Dafür müssen Sie die Datenspeichergeräte aus dem ursprünglichen Rechnersystem herausnehmen und diese an das neue System anschließen. Dies kann durch Neuverkabelung der betreffenden Geräte bzw. die Verwendung von Geräten mit mehreren Anschlüssen, z. B. Geräte in einem Speichernetzwerk (SAN), bewerkstelligt werden. Mit ZFS können Sie einen Pool von einem Rechner exportieren und diesen in den neuen Rechner importieren. Dies ist auch möglich, wenn beide Rechnerarchitekturen unterschiedliche Bitbreiten besitzen. Informationen zum Replizieren bzw. Migrieren von Dateisystemen zwischen verschiedenen Speicher-Pools, die auf unterschiedlichen Rechnersystemen installiert sind, finden Sie unter „Senden und Empfangen von ZFS-Daten“ auf Seite 225.

Vorbereiten der Migration eines ZFS-Speicher-Pools

Speicher-Pools sollten explizit migriert werden, damit klar ist, dass sie zur Migration bereit sind. Diese Operation schließt alle noch nicht geleerten Datenpuffer durch Speichern der dort enthaltenen Daten auf dem Datenträger, kopiert alle Daten auf den Datenträger und schließt den Export damit ab. Dann werden alle „Spuren” des Pools vom System beseitigt.

Wenn Sie den Pool nicht explizit exportieren, sondern die Datenträger manuell entfernen, kann der resultierende Pool trotzdem noch in ein anderes System importiert werden. Es kann jedoch sein, dass die in den allerletzen Sekunden ausgeführten Datentransaktionen verloren gehen und der betreffende Pool auf dem ursprünglichen Rechnersystem als fehlerhaft angezeigt wird, da die Datenspeichergeräte nicht mehr vorhanden sind. Standardmäßig kann ein Pool, der nicht explizit exportiert wurde, nicht importiert werden. Dies ist erforderlich, um zu verhindern, dass aktive Pools, die über Netzwerk zugänglichen und von anderen Systemen belegten Speicherplatz enthalten, versehentlich importiert werden.

Exportieren eines ZFS-Speicher-Pools

Speicher-Pools können mit dem Befehl `zpool export` exportiert werden. Beispiel:

```
# zpool export tank
```

Nach der Ausführung dieses Befehls ist der Pool `tank` im System nicht mehr sichtbar. Vor dem Fortfahren versucht der Befehl alle innerhalb des Pools eingehängten Dateisysteme auszuhängen. Falls das Aushängen von Dateisystemen fehlschlägt, können Sie mithilfe der Option `-f` ein Aushängen erzwingen. Beispiel:

```
# zpool export tank
cannot unmount '/export/home/eschrock': Device busy
# zpool export -f tank
```

Falls Datenspeichergeräte zum Zeitpunkt des Exports nicht verfügbar sind, können die betreffenden Datenträger nicht als „sauber" exportiert eingestuft werden. Wenn ein solches Datenspeichergerät später ohne die funktionierenden Datenspeichergeräte mit einem System verbunden wird, erscheint das betreffende Gerät als potenziell aktiv." Wenn im Pool ZFS-Volumes vorhanden sind, kann der Pool auch nicht mithilfe der Option `-f` exportiert werden. Zum Exportieren eines Pools mit ZFS-Volume müssen Sie sich zunächst vergewissern, dass keine Ressource, die das Volume belegt hat, noch aktiv ist.

Weitere Informationen zu ZFS-Volumes finden Sie unter „ZFS-Volumes" auf Seite 279.

Ermitteln verfügbarer Speicher-Pools für den Import

Nachdem ein Pool (durch Export bzw. erzwungenes Entfernen von Datenspeichergeräten) aus einem System entfernt wurde, müssen die betreffenden Geräte mit dem Zielsystem verbunden werden. Obwohl ZFS Situationen behandeln kann, in denen nur ein Teil der Geräte verfügbar ist, müssen alle zu einem Pool gehörenden Datenspeichergeräte zum neuen System transferiert werden. Geräte müssen jedoch nicht notwendigerweise unter dem gleichen Gerätenamen verbunden werden. ZFS erkennt verschobene bzw. umbenannte Geräte und passt die Konfiguration entsprechend an. Führen Sie zum Ermitteln importierbarer Pools den Befehl `zpool export` aus. Beispiel:

```
# zpool import
  pool: tank
    id: 3778921145927357706
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        tank        ONLINE
          mirror    ONLINE
            c1t0d0  ONLINE
            c1t1d0  ONLINE
```

In diesem Beispiel kann der Pool `tank` in ein Zielsystem importiert werden. Jeder Pool wird durch einen Namen und einen eindeutigen nummerischen Bezeichner identifiziert. Wenn im

Falle mehrerer importierbarer Pools einige Pools den gleichen Namen besitzen, können sie mithilfe des nummerischen Bezeichners unterschieden werden.

Wie beim Befehl `zpool status` verweist auch der Befehl `zpool import` auf einen Artikel der Sun Knowledge Base auf dem Internet, der die aktuellsten Informationen und Reparaturhinweise zu Problemen enthält, die das Importieren von Pools verhindern. In diesem Fall kann das Importieren eines Pools erzwungen werden. Das Importieren eines Pools, der gegenwärtig über Netzwerkzugriff von einem anderen System verwendet wird, kann Daten beschädigen und auf beiden Systemen zu Abstürzen führen, wenn diese Systeme Daten auf das gleiche Datenspeichergerät schreiben. Wenn einige Geräte eines Pools nicht verfügbar sind, zum Zusammenstellen eines funktionierenden Pools jedoch genügend Redundanz vorhanden ist, geht der Pool in den Status `DEGRADED`. Beispiel:

```
# zpool import
  pool: tank
    id: 3778921145927357706
 state: DEGRADED
status: One or more devices are missing from the system.
action: The pool can be imported despite missing or damaged devices.  The
        fault tolerance of the pool may be compromised if imported.
   see: http://www.sun.com/msg/ZFS-8000-2Q
config:

        tank         DEGRADED
          mirror     DEGRADED
            c1t0d0   UNAVAIL   cannot open
            c1t1d0   ONLINE
```

In diesem Beispiel ist der erste Datenträger beschädigt oder nicht vorhanden, der Pool kann aber trotzdem importiert werden, da die gespiegelten Daten noch verfügbar sind. Wenn zuviele fehlerhafte Datenspeichergeräte vorhanden sind bzw. Geräte fehlen, kann der Pool nicht importiert werden. Beispiel:

```
# zpool import
  pool: dozer
    id: 12090808386336829175
 state: FAULTED
action: The pool cannot be imported. Attach the missing
        devices and try again.
   see: http://www.sun.com/msg/ZFS-8000-6X
config:
        raidz               FAULTED
          c1t0d0    ONLINE
          c1t1d0    FAULTED
          c1t2d0    ONLINE
          c1t3d0    FAULTED
```

In diesem Beispiel fehlen in einem virtuellen RAID-Z-Gerät zwei Datenträger, was bedeutet, dass zum Rekonstruieren des Pools nicht genügend Redundanz verfügbar ist. In einigen Fällen kann es auch sein, dass zum Ermitteln der vollständigen Konfiguration nicht genügend Datenspeichergeräte vorhanden sind. In einem solchen Fall „weiß" ZFS nicht, welche anderen Geräte zu diesem Pool gehört haben, obwohl ZFS zu dieser Situation soviele Informationen wie möglich meldet. Beispiel:

```
# zpool import
pool: dozer
    id: 12090808386336829175
 state: FAULTED
status: One or more devices are missing from the system.
action: The pool cannot be imported. Attach the missing
        devices and try again.
   see: http://www.sun.com/msg/ZFS-8000-6X
config:
        dozer          FAULTED   missing device
          raidz       ONLINE
            c1t0d0    ONLINE
            c1t1d0    ONLINE
            c1t2d0    ONLINE
            c1t3d0    ONLINE
        Additional devices are known to be part of this pool, though their
        exact configuration cannot be determined.
```

Auffinden von ZFS-Speicher-Pools aus anderen Verzeichnissen

Standardmäßig durchsucht der Befehl `zpool import` nur im Verzeichnis `/dev/dsk` enthaltene Datenspeichergeräte. Wenn Geräte in einem anderen Verzeichnis vorhanden sind oder Sie auf Dateien aufsetzende Pools verwenden, müssen Sie mithilfe der Option `-d` andere Verzeichnisse durchsuchen. Beispiel:

```
# zpool create dozer mirror /file/a /file/b
# zpool export dozer
# zpool import -d /file
  pool: dozer
    id: 10952414725867935582
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        dozer        ONLINE
          mirror     ONLINE
```

```
            /file/a  ONLINE
            /file/b  ONLINE
# zpool import -d /file dozer
```

Sie können die Option `-d` mehrmals angeben, wenn Datenspeichergeräte in mehreren Verzeichnissen vorhanden sind.

Importieren von ZFS-Speicher-Pools

Wenn ein Pool für den Import ermittelt wurde, können Sie es durch Angabe des Namens bzw. seines nummerischen Bezeichners als Argument für den Befehl `zpool import` importieren. Beispiel:

```
# zpool import tank
```

Wenn mehrere Pools den gleichen Namen besitzen, können Sie mithilfe des nummerischen Bezeichners angeben, welcher Pool importiert werden soll. Beispiel:

```
# zpool import
  pool: dozer
    id: 2704475622193776801
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        dozer       ONLINE
          c1t9d0    ONLINE

  pool: dozer
    id: 6223921996155991199
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        dozer       ONLINE
          c1t8d0    ONLINE
# zpool import dozer
cannot import 'dozer': more than one matching pool
import by numeric ID instead
# zpool import 6223921996155991199
```

Wenn ein Pool-Name mit einem bereits vorhandenen Pool-Namen in Konflikt steht, können Sie den Pool unter einem anderen Namen importieren. Beispiel:

```
# zpool import dozer zeepool
```

Dieser Befehl importiert den exportierten Pool `dozer` unter dem neuen Namen `zeepool`. Wenn ein Pool nicht ordnungsgemäß exportiert wurde, benötigt ZFS das `-f`-Flag, um zu verhindern, dass versehentlich ein Pool importiert wird, der noch von einem anderen System benutzt wird. Beispiel:

```
# zpool import dozer
cannot import 'dozer': pool may be in use on another system
use '-f' to import anyway
# zpool import -f dozer
```

Pools können mithilfe der Option `-R` in ein anderes Root-Verzeichnis importiert werden. Weitere Informationen zu Speicher-Pools mit alternativem Root-Verzeichnis finden Sie unter „Verwenden von ZFS-Speicher-Pools mit alternativem Root-Verzeichnis“ auf Seite 288.

Wiederherstellen gelöschter ZFS-Speicher-Pools

Gelöschte Speicher-Pools können mit dem Befehl `zpool import -D` wiederhergestellt werden. Beispiel:

```
# zpool destroy tank
# zpool import -D
pool: tank
    id: 3778921145927357706
 state: ONLINE (DESTROYED)
action: The pool can be imported using its name or numeric identifier.  The
        pool was destroyed, but can be imported using the '-Df' flags.
config:

        tank        ONLINE
          mirror    ONLINE
            c1t0d0  ONLINE
            c1t1d0  ONLINE
```

In der obigen Ausgabe des Befehls `zpool import` kann der betreffende Pool aufgrund der folgenden Statusinformationen als gelöscht erkannt werden:

```
state: ONLINE (DESTROYED)
```

Führen Sie zum Wiederherstellen des gelöschten Pools den Befehl `zpool import -D` am wiederherzustellenden Pool erneut aus. Beispiel:

```
# zpool import -D tank
# zpool status tank
  pool: tank
 state: ONLINE
```

```
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0

errors: No known data errors
```

Wenn eines der Datenspeichergeräte im gelöschten Pool fehlerhaft oder nicht verfügbar ist, können Sie den gelöschten Pool unter Umständen mit der Option `-f` trotzdem wiederherstellen. Importieren Sie in einer solchen Situation den im eingeschränkten Zustand befindlichen Pool und versuchen Sie dann, den Geräteausfall zu beheben. Beispiel:

```
# zpool destroy dozer
# zpool import -D
pool: dozer
    id:
 state: DEGRADED (DESTROYED)
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: resilver completed after 0h0m with 0 errors on Fri Aug 28 09:33:56 2009
config:

        NAME         STATE     READ WRITE CKSUM
        dozer        DEGRADED     0     0     0
          raidz2     DEGRADED     0     0     0
            c2t8d0   ONLINE       0     0     0
            c2t9d0   ONLINE       0     0     0
            c2t10d0  ONLINE       0     0     0
            c2t11d0  UNAVAIL      0    35     1  cannot open
            c2t12d0  ONLINE       0     0     0

errors: No known data errors
# zpool import -Df dozer
# zpool status -x
 pool: dozer
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: resilver completed after 0h0m with 0 errors on Fri Aug 28 09:33:56 2009
```

```
config:

        NAME         STATE     READ WRITE CKSUM
        dozer        DEGRADED     0     0     0
          raidz2     DEGRADED     0     0     0
            c2t8d0   ONLINE       0     0     0
            c2t9d0   ONLINE       0     0     0
            c2t10d0  ONLINE       0     0     0
            c2t11d0  UNAVAIL      0    37     0  cannot open
            c2t12d0  ONLINE       0     0     0

errors: No known data errors
# zpool online dozer c2t11d0
Bringing device c2t11d0 online
# zpool status -x
all pools are healthy
```

Aufrüsten von ZFS-Speicher-Pools

Wenn in Ihrem System ZFS-Speicher-Pools aus einer früheren Solaris-Version wie z. B. Solaris 10 6/06 vorhanden sind, können Sie diese Pools mit dem Befehl `zpool upgrade` aufrüsten, um die Pool-Funktionen von Solaris 10 11/06 nutzen zu können. Darüber hinaus wurde der Befehl `zpool status` so geändert, dass Sie jetzt darauf hingewiesen werden, wenn Pools mit älteren Versionen laufen. Beispiel:

```
# zpool status
  pool: test
 state: ONLINE
status: The pool is formatted using an older on-disk format.  The pool can
        still be used, but some features are unavailable.
action: Upgrade the pool using 'zpool upgrade'.  Once this is done, the
        pool will no longer be accessible on older software versions.
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        test        ONLINE       0     0     0
          c1t27d0   ONLINE       0     0     0

errors: No known data errors
```

Mithilfe der folgenden Syntax können Sie zusätzliche Informationen zu einer bestimmten Version und unterstützten Releases ermitteln.

```
# zpool upgrade -v
This system is currently running ZFS version 3.

The following versions are supported:

VER  DESCRIPTION
---  --------------------------------------------------------
 1   Initial ZFS version
 2   Ditto blocks (replicated metadata)
 3   Hot spares and double parity RAID-Z

For more information on a particular version, including supported releases, see:

http://www.opensolaris.org/os/community/zfs/version/N

Where 'N' is the version number.
```

Dann können Sie zum Aufrüsten aller Pools den Befehl `zpool upgrade` ausführen. Beispiel:

```
# zpool upgrade -a
```

Hinweis – Wenn Sie den Pool auf eine neuere ZFS-Version aktualisieren, ist er auf einem System, auf dem eine ältere ZFS-Version ausgeführt wird, nicht verfügbar.

◆ ◆ ◆ KAPITEL 5

Installieren und Booten eines ZFS-Root-Dateisystems

In diesem Kapitel erfahren Sie, wie ein ZFS-Dateisystem installiert und gebootet wird. Darüber hinaus wird die Migration eines UFS-Root-Dateisystems in ein ZFS-Dateisystem mithilfe von Solaris Live Upgrade behandelt.

Dieses Kapitel enthält die folgenden Abschnitte:

Die Solaris 10 10/09 -Versionshinweise enthalten eine Liste der bekannten Probleme mit diesem Release.

Stets aktuelle Informationen zur Fehlerbehebung finden Sie unter:

`http://www.solarisinternals.com/wiki/index.php/ZFS_Troubleshooting_Guide`

Installieren und Booten eines ZFS-Root-Dateisystems (Übersicht)

Ab dem Release Solaris 10 10/08 stehen folgende Methoden zum Installieren und Booten eines ZFS-Root-Dateisystems zur Verfügung:

- Sie können eine Erstinstallation durchführen und dabei ZFS als Root-Dateisystem auswählen.
- Sie können ein UFS-Root-Dateisystem mithilfe der Solaris Live Upgrade-Funktion in ein ZFS-Root-Dateisystem überführen. Darüber hinaus können Sie mit Solaris Live Upgrade Folgendes vornehmen:
  - Erstellen einer neuen Boot-Umgebung innerhalb eines vorhandenen ZFS-Root-Pools
  - Erstellen einer neuen Boot-Umgebung in einem neuen ZFS-Root-Pool
- Sie können ein Solaris JumpStart-Profil zur automatischen Installation eines Systems mit einem ZFS-Root-Dateisystem verwenden.
- Das Release Solaris 10 10/09 ermöglicht Ihnen die automatische Installation eines Systems mit ZFS-Flash-Archiv mithilfe eines JumpStart-Profils.

Nach der Installation eines ZFS-Root-Dateisystems oder der Migration in ein ZFS-Root-Dateisystem auf einem SPARC- oder x86-System startet das System automatisch aus dem ZFS-Root-Dateisystem. Weitere Informationen zum Ändern des Boot-Verhaltens finden Sie unter „Booten aus einem ZFS-Root-Dateisystem“ auf Seite 161.

Leistungsmerkmale für die ZFS-Installation

In diesem Solaris-Release stehen die folgenden Leistungsmerkmale für die ZFS-Installation zur Verfügung:

- Die interaktive Textmodus-Installationsoption von Solaris ermöglicht die Installation eines UFS- oder ZFS-Root-Dateisystems. In diesem Solaris-Release ist UFS weiterhin das Standarddateisystem. Auf die interaktive Textmodus-Installationsoption können Sie wie folgt zugreifen:
  - Syntax für SPARC-Systeme und die Solaris-Installations-DVD:

    ```
    ok boot cdrom - text
    ```

  - Syntax für SPARC-Systeme beim Booten über das Netzwerk:

    ```
    ok boot net - text
    ```

  - Bei x86-Systemen wählen Sie die Textmodus-Installationsoption, wenn diese zur Auswahl steht.
- Die benutzerdefinierter JumpStart™-Installation bietet folgende Funktionen:

  - Sie können ein Profil zum Erstellen eines ZFS-Speicher-Pools anlegen und ein bootfähiges ZFS-Dateisystem benennen.
  - Sie können ein Profil zur Identifizierung eines Flash-Archivs eines ZFS-Root-Pools anlegen.
- Sie können ein UFS-Root-Dateisystem mithilfe der Solaris Live Upgrade-Funktion in ein ZFS-Root-Dateisystem überführen. Die Befehle `lucreate` und `luactivate` wurden um Unterstützung für ZFS-Pools und -Dateisysteme erweitert. Die Befehle `lustatus` und `ludelete` verhalten sich genau wie in vorherigen Solaris-Versionen.
- Sie können bei der Installation zwei Festplatten auswählen und einen gespiegelten Speicher-Pool mit ZFS-Root-Dateisystem (im Folgenden „ZFS-Root-Pool“) einrichten. Alternativ lassen sich auch nach der Installation zusätzliche Festplatten anhängen, um einen gespiegelten ZFS-Root-Pool herzustellen.
- Swap- und Dump-Geräte werden automatisch auf ZFS-Volumes im ZFS-Root-Pool erstellt.

Die folgenden Installationsfunktionen stehen in diesem Release nicht zur Verfügung:

- Die GUI-Installationsfunktion steht derzeit nicht zum Installieren eines ZFS-Root-Dateisystems zur Verfügung.
- Die Funktion Solaris™ Flash-Installation zum Installieren eines ZFS-Root-Dateisystems ist durch Auswahl der Option "Flash-Installation" aus der Erstinstallationsoption nicht verfügbar. Sie können jedoch ein JumpStart-Profil zur Identifizierung eines Flash-Archivs eines ZFS-Root-Pools erstellen. Weitere Informationen finden Sie unter „Installieren eines ZFS-Root-Dateisystems (Flash-Archiv-Installation)“ auf Seite 127.
- Ein Upgrade des UFS-Root-Dateisystems zu einem ZFS-Root-Dateisystem ist nicht mithilfe des Standard-Upgrade-Programms möglich. Wenn mindestens ein boot-fähiger UFS-Bereich vorhanden ist, sollte die Standard-Upgrade-Option verfügbar sein. Wenn ein boot-fähiger ZFS-Pool, aber kein boot-fähiger UFS-Bereich vorhanden ist, ist ein Upgrade ausschließlich mit Live Upgrade möglich. Das Standard-Upgrade-Programm kann in diesem Fall nicht verwendet werden. Wenn sowohl ein boot-fähiger ZFS-Pool als auch ein boot-fähiger UFS-Bereich vorhanden ist, sollte die Standard-Upgrade-Option verfügbar sein, aber nur der UFS-Bereich für ein Update zur Verfügung stehen.

Solaris-Installation und Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung

Vergewissern Sie sich vor dem Versuch, ein ZFS-Root-Dateisystem auf einem System zu installieren oder ein UFS- in ein ZFS-Root-Dateisystem zu migrieren, dass folgende Voraussetzungen erfüllt sind:

Voraussetzungen für das Solaris-Release

Sie haben folgende Möglichkeiten, ein ZFS-Root-Dateisystem zu installieren und zu booten oder in ein ZFS-Root-Dateisystem zu migrieren:

- Installieren eines ZFS-Root-Dateisystems – ab Solaris 10 10/08 verfügbar.
- Migrieren aus einem UFS-Root-Dateisystem in ein ZFS-Root-Dateisystem mit Solaris Live Upgrade – Sie müssen Solaris 10 10/08 oder höher installiert oder auf Solaris 10 10/08 oder höher aktualisiert haben.

Allgemeine Voraussetzungen für einen ZFS-Speicher-Pool

Lesen Sie die folgenden Abschnitte über den Speicherplatz für ZFS-Root-Pools und Konfigurationsvoraussetzungen.

Für ZFS-Speicher-Pools erforderlicher Speicherplatz

Ein ZFS-Root-Dateisystem benötigt mehr Speicherplatz im Pool als ein UFS-Root-Dateisystem, da Swap- und Dump-Geräte in einer ZFS-Root-Umgebung separate Speichergeräte sein müssen. In UFS-Root-Dateisystemen sind Swap- und Dump-Geräte standardmäßig dasselbe Gerät.

Wenn auf einem System ein ZFS-Root-Dateisystem installiert oder ein Upgrade darauf vorgenommen wird, hängen die Größe des Swap-Bereichs und des Dump-Geräts von der physischen Speicherkapazität ab. Der mindestens erforderliche Speicherplatz im Pool für ein boot-fähiges ZFS-Root-Dateisystem richtet sich nach der Größe des physischen Speichers, dem freien Speicherplatz auf der Festplatte und der Anzahl der zu erstellenden Boot-Umgebungen (BUs).

Machen Sie sich mit folgenden Voraussetzungen bezüglich des Speicherplatzes für ZFS-Speicher-Pools vertraut:

- Für die Installation eines ZFS-Root-Dateisystems sind mindestens 768 MB Arbeitsspeicher erforderlich.
- Für eine bessere ZFS-Gesamtleistung wird 1 GB Arbeitsspeicher empfohlen.
- Empfohlen werden mindestens 16 GB freier Speicherplatz auf der Festplatte. Der Speicherplatz wird wie folgt belegt:
  - **Swap-Bereich und Dump-Gerät** – Die Solaris-Installationsprogramme erstellen Swap- und Dump-Volumes in den folgenden Standardgrößen:
    - **Erstinstallation von Solaris** – Die Größe des Standard-Swap-Volume ist in der neuen ZFS-BU halb so groß wie der physische Speicher, im Allgemeinen zwischen 512 MB und 2 GB. Die Größe des Swap-Volume kann während einer Erstinstallation angepasst werden.

- Die Dump-Volume-Größe wird vom Kernel basierend auf den `dumpadm`-Informationen und der Größe des physikalischen Speichers berechnet. Die Größe des Dump-Volume kann während einer Erstinstallation angepasst werden.
- **Solaris Live Upgrade** – Wenn ein UFS-Root-Dateisystem auf ein ZFS-Root-Dateisystem umgestellt wird, wird die Größe des Standard-Swap-Volume für die ZFS-Boot-Umgebung (ZFS-BU) als Größe des Swap-Geräts der UFS-BU berechnet. Bei der Berechnung der Größe des Standard-Swap-Volume werden einfach die Größen aller Swap-Geräte in der UFS-BU addiert, und es wird ein ZFS-Volume der entsprechenden Größe in der ZFS-BU erstellt. Wenn keine Swap-Geräte in der UFS-BU definiert sind, wird die Größe des Standard-Swap-Volume auf 512 MB gesetzt.
- Die Größe des Standard-Dump-Volume ist in der ZFS-BU halb so groß wie der physische Speicher, zwischen 512 MB und 2 GB.

Sie können die Größen der Swap- und Dump-Volumes ändern, sofern die neuen Größen den Betrieb des Systems unterstützen. Weitere Informationen finden Sie unter „Anpassen der Größe von ZFS-Swap- und Dump-Geräten“ auf Seite 159.

- **Boot-Umgebung (BU)** - Zusätzlich zu dem für neue Swap- und Dump-Geräte erforderlichen oder nachträglich geänderten Speicherplatz werden für eine ZFS-BU bei Migration von einer UFS-BU ungefähr 6 GB benötigt. Für ZFS-BUs, die aus anderen ZFS-BUs geklont werden, ist kein zusätzlicher Speicherplatz erforderlich. Beachten Sie jedoch, dass die BU-Größe durch die Installation von Patches zunimmt. Alle ZFS-BUs in demselben Root-Pool greifen auf dieselben Swap- und Dump-Geräte zu.

Beispielsweise könnte ein System mit 12 GB Festplattenplatz zu klein für eine boot-fähige ZFS-Umgebung sein, da 2 GB je Swap- und Dump-Gerät und rund 6 GB für die ZFS-BU benötigt werden, die aus einer UFS-BU migriert wird.

Voraussetzungen für die Konfiguration des ZFS-Speicher-Pools

Machen Sie sich mit folgenden Voraussetzungen bezüglich der Konfiguration von ZFS-Speicher-Pools vertraut:

- Der als Root-Pool bestimmte Pool muss ein SMI-Label haben. Diese Anforderung sollte bei Pools erfüllt sein, die aus Festplattenbereichen erstellt werden.
- Der Pool muss entweder auf einem Festplattenbereich oder auf gespiegelten Festplattenbereichen vorhanden sein. Bei dem Versuch, eine nicht unterstützte Pool-Konfiguration bei einer Live Upgrade-Migration zu verwenden, erscheint eine Meldung folgender Art:

```
ERROR: ZFS pool name does not support boot environments
```

Weitere Informationen zu unterstützten Konfigurationen von ZFS-Root-Pools finden Sie unter „Erstellen eines ZFS-Root-Pools“ auf Seite 68.

- Auf einem x86-System muss die Festplatte eine Solaris-`fdisk`-Partition enthalten. Eine Solaris-`fdisk`-Partition wird bei der Installation eines x86-Systems automatisch installiert. Weitere Informationen zu Solaris-`fdisk`-Partitionen finden Sie in „Guidelines for Creating an fdisk Partition“ in *System Administration Guide: Devices and File Systems*.
- Datenträger, die für das Booten in einem ZFS-Root-Pool bestimmt sind, dürfen auf SPARC- sowie auf x86-Systemen nicht mehr als 1 TB umfassen.
- Auf dem Root-Pool kann die Komprimierung aktiviert werden, allerdings erst nach Installation des Root-Pools. Während der Installation eines Root-Pools kann die Komprimierung nicht aktiviert werden. Der Komprimierungs-Algorithmus `gzip` wird auf Root-Pools nicht unterstützt.

Installieren eines ZFS-Root-Dateisystems (Erstinstallation)

In diesem Solaris-Release können Sie eine Erstinstallation durchführen und mit der interaktiven Textmodus-Installationsoption von Solaris einen ZFS-Speicher-Pool erstellen, das ein boot-fähiges ZFS-Root-Dateisystem enthält. Wenn Sie einen bereits vorhandenen ZFS-Speicher-Pool für das ZFS-Root-Dateisystem verwenden möchten, müssen Sie das vorhandene UFS-Root-Dateisystem mit Solaris Live Upgrade in ein ZFS-Root-Dateisystem in einem vorhandenen ZFS-Speicher-Pool migrieren. Weitere Informationen finden Sie unter „Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem (Solaris Live Upgrade)“ auf Seite 134.

Wenn Sie nach der Erstinstallation eines ZFS-Root-Dateisystems Zonen konfigurieren und das System patchen bzw. aktualisieren möchten, lesen Sie unter „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 141 oder „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 5/09 und Solaris 10 10/09)“ auf Seite 147 nach.

Wenn auf Ihrem System bereits ZFS-Speicher-Pools vorhanden sind, werden diese mit der folgenden Meldung angezeigt. Sie bleiben jedoch unverändert, wenn Sie zur Installation des neuen Speicher-Pools nicht die Festplatten auswählen, auf denen sich die vorhandenen Pools befinden.

```
There are existing ZFS pools available on this system.  However, they can only be upgraded
using the Live Upgrade tools.  The following screens will only allow you to install a ZFS root system,
not upgrade one.
```

Achtung – Vorhandene Pools werden überschrieben, falls einige ihrer Datenträger für den neuen Pool ausgewählt werden.

Bevor Sie mit der Erstinstallation für die Erstellung eines ZFS-Speicher-Pools beginnen, lesen Sie den Abschnitt „Solaris-Installation und Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung “ auf Seite 117.

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems

Der interaktive, textgestützte Solaris-Installationsprozess unterscheidet sich im Wesentlichen nicht von den vorherigen Solaris-Releases, außer darin, dass Sie gefragt werden, ob ein UFS- oder ZFS-Root-Dateisystem erstellt werden soll. UFS ist auch in diesem Release weiterhin das Standarddateisystem. Wenn Sie ein ZFS-Root-Dateisystem wählen, werden Sie dazu aufgefordert, einen ZFS-Speicher-Pool zu erstellen. Die Installation eines ZFS-Root-Dateisystems umfasst die folgenden Schritte:

1. Wählen Sie die interaktive Solaris-Installationsmethode, da die Solaris Flash-Installation zum Erstellen eines boot-fähigen ZFS-Root-Dateisystems nicht verfügbar ist. Sie können jedoch ein ZFS-Flash-Archiv zur Verwendung während einer JumpStart-Installation erstellen. Weitere Informationen finden Sie unter „Installieren eines ZFS-Root-Dateisystems (Flash-Archiv-Installation)“ auf Seite 127.

   Ab der Solaris-Version 10 10/08 können Sie ein UFS-Root-Dateisystem in ein ZFS-Root-Dateisystem migrieren. Hierfür muss Solaris 10 10/08 oder eine neuere Version bereits installiert sein. Weitere Informationen zur Migration in ein ZFS-Root-Dateisystem finden Sie unter „Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem (Solaris Live Upgrade)“ auf Seite 134.

2. Wenn Sie ein ZFS-Root-Dateisystem erstellen möchten, wählen Sie die ZFS-Option. Beispiel:

```
Choose Filesystem Type

  Select the filesystem to use for your Solaris installation


            [ ] UFS
            [X] ZFS
```

3. Nach der Auswahl der zu installierenden Software werden Sie zur Auswahl der Festplatten aufgefordert, auf denen der ZFS-Speicher-Pool installiert werden soll. Dieser Bildschirm sieht ähnlich aus wie in vorherigen Solaris-Versionen:

```
Select Disks

  On this screen you must select the disks for installing Solaris software.
  Start by looking at the Suggested Minimum field; this value is the
  approximate space needed to install the software you've selected. For ZFS,
  multiple disks will be configured as mirrors, so the disk you choose, or the
  slice within the disk must exceed the Suggested Minimum value.
  NOTE: ** denotes current boot disk

  Disk Device                                              Available Space
============================================================================
  [X] ** c1t1d0                                          69994 MB
  [ ]    c1t2d0                                          69994 MB  (F4 to edit)
```

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

```
                                    Maximum Root Size:  69994 MB
                                    Suggested Minimum:   7466 MB
```

Sie können die Festplatte(n) auswählen, auf der/denen ein ZFS-Root-Pool installiert werden soll. Wenn Sie zwei Festplatten auswählen, wird für den Root-Pool eine aus zwei Platten bestehende Konfiguration mit Datenspiegelung ausgewählt. Optimal ist ein gespiegelter Pool mit zwei oder drei Festplatten. Wenn acht Festplatten vorhanden sind und Sie alle acht auswählen, werden diese für den Root-Pool als ein großer Mirror verwendet. Diese Konfiguration ist nicht optimal. Alternativ können Sie einen gespiegelten Root-Pool nach der Erstinstallation erstellen. RAID-Z-Konfigurationen werden für den Root-Pool nicht unterstützt. Weitere Informationen zur Konfiguration von ZFS-Speicher-Pools finden Sie unter „Replikationsfunktionen eines ZFS-Speicher-Pools“ auf Seite 63.

4. Zur Auswahl von zwei Festplatten zur Erstellung eines gespiegelten Root-Pools wählen Sie die zweite Festplatte mit dem Cursor aus. Es werden zum Beispiel sowohl `c1t1d1` als auch `c0t2d0` als Root-Pool-Festplatten ausgewählt. Beide Festplatten müssen ein SMI-Label und den Bereich 0 haben. Falls die Festplatten kein SMI-Label haben oder Bereiche enthalten, verlassen Sie das Installationsprogramm und verwenden Sie das Dienstprogramm `format` zur Umbenennung und Neupartitionierung der Festplatten. Beginnen Sie dann die Installation erneut.

```
Select Disks

  On this screen you must select the disks for installing Solaris software.
  Start by looking at the Suggested Minimum field; this value is the
  approximate space needed to install the software you've selected. For ZFS,
  multiple disks will be configured as mirrors, so the disk you choose, or the
  slice within the disk must exceed the Suggested Minimum value.
  NOTE: ** denotes current boot disk

 Disk Device                                                Available Space
================================================================================
  [X] ** c1t1d0                                           69994 MB
  [X]    c1t2d0                                           69994 MB  (F4 to edit)

                                    Maximum Root Size:  69994 MB
                                    Suggested Minimum:   7466 MB
```

Wenn in der Spalte "Verfügbarer Speicherplatz" 0 MB angezeigt werden, bedeutet dies in der Regel, dass die Festplatte ein EFI-Label hat.

5. Nach der Auswahl eines bzw. mehrerer Datenträger für den ZFS-Speicher-Pool erscheint ein Bildschirm wie dieser:

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

```
Configure ZFS Settings

  Specify the name of the pool to be created from the disk(s) you have chosen.
  Also specify the name of the dataset to be created within the pool that is
  to be used as the root directory for the filesystem.


            ZFS Pool Name: rpool
    ZFS Root Dataset Name: s10s_u8wos_08a
    ZFS Pool Size (in MB): 69994
Size of Swap Area (in MB): 2048
Size of Dump Area (in MB): 1024
      (Pool size must be between 6442 MB and 69995 MB)

                      [X] Keep / and /var combined
                      [ ] Put /var on a separate dataset
```

In diesem Bildschirm können Sie den Namen des ZFS-Pools, den Dataset-Namen, die Pool-Größe sowie die Größe der Swap- und Dump-Geräte ändern. Verschieben Sie hierzu den Cursor in den Einträgen und ersetzen Sie die Standardwerte durch die gewünschten Werte. Anderenfalls können Sie die Standardwerte übernehmen. Darüber hinaus können Sie das Verfahren ändern, mit dem das Dateisystem `/var` erstellt und eingehängt wird.

In diesem Beispiel wird der Name des Root-Datasets in `zfs1009BE` abgeändert.

```
      ZFS Pool Name: rpool
    ZFS Root Dataset Name: zfs1009BE
    ZFS Pool Size (in MB): 69994
Size of Swap Area (in MB): 2048
Size of Dump Area (in MB): 1024
      (Pool size must be between 6442 MB and 69995 MB)

                      [X] Keep / and /var combined
                      [ ] Put /var on a separate dataset
```

6. Dieser letzte Installationsbildschirm bietet die Möglichkeit, das Installationsprofil zu ändern. Beispiel:

```
Profile

  The information shown below is your profile for installing Solaris software.
  It reflects the choices you've made on previous screens.

  ============================================================================

               Installation Option: Initial
```

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

```
                  Boot Device: c1t1d0
          Root File System Type: ZFS
                Client Services: None

                        Regions: North America
                  System Locale: C ( C )

                       Software: Solaris 10, Entire Distribution
                      Pool Name: rpool
          Boot Environment Name: zfs1009BE
                      Pool Size: 69994 MB
                Devices in Pool: c1t1d0
                                 c1t2d0
```

Überprüfen Sie nach abgeschlossener Installation die Informationen zum erstellten ZFS-Speicher-Pool und Dateisystem. Beispiel:

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

        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          mirror      ONLINE       0     0     0
            c1t1d0s0  ONLINE       0     0     0
            c1t2d0s0  ONLINE       0     0     0

errors: No known data errors
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
rpool                 6.66G  60.3G    97K  /rpool
rpool/ROOT            4.66G  60.3G    21K  legacy
rpool/ROOT/zfs1009BE  4.66G  60.3G  4.66G  /
rpool/dump            1.00G  60.3G  1.00G  -
rpool/export            44K  60.3G    23K  /export
rpool/export/home       21K  60.3G    21K  /export/home
rpool/swap               1G  61.3G    16K  -
```

In der Beispielausgabe von `zfs list` sind die Root-Pool-Komponenten wie z. B. das Verzeichnis `rpool/ROOT` aufgeführt, auf das standardmäßig nicht zugegriffen werden kann.

Wenn Sie den ZFS-Speicher-Pool zunächst mit einem Datenträger erstellt haben, können Sie es nach Abschluss der Installation in eine gespiegelte Konfiguration umformen, indem Sie mithilfe des Befehls `zpool attach` eine verfügbare Festplatte einbinden. Beispiel:

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

```
# zpool attach rpool c1t1d0s0 c1t2d0s0
# zpool status
  pool: rpool
 state: ONLINE
status: One or more devices is currently being resilvered.  The pool will
        continue to function, possibly in a degraded state.
action: Wait for the resilver to complete.
 scrub: resilver in progress for 0h0m, 5.03% done, 0h13m to go
config:

        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          mirror      ONLINE       0     0     0
            c1t1d0s0  ONLINE       0     0     0
            c1t2d0s0  ONLINE       0     0     0

errors: No known data errors
```

Das Schreiben der Daten auf die neue Festplatte nimmt einige Zeit in Anspruch, während der der Pool jedoch weiterhin verfügbar ist.

Wenn Sie das Booten auf den anderen Festplatten in der Datenspiegelungskonfiguration ermöglichen möchten, müssen Sie, solange CR 6668666 nicht behoben ist, die Boot-Informationen auf den zusätzlich eingebundenen Festplatten mithilfe des Befehls `installboot` oder `installgrub` installieren. Dieser Schritt ist unnötig, wenn Sie zur Erstellung des gespiegelten ZFS-Root-Pools die Erstinstallationsmethode verwenden. Weitere Informationen zum Installieren von Boot-Informationen finden Sie unter „Booten von einer alternativen Festplatte in einem gespiegelten ZFS-Root-Pool“ auf Seite 161.

Weitere Informationen zum Hinzufügen oder Einbinden von Festplatten stehen unter „Verwalten von Datenspeichergeräten in ZFS-Speicher-Pools“ auf Seite 77 zur Verfügung.

Um eine weitere ZFS-Boot-Umgebung (BU) in demselben Speicher-Pool zu erstellen, verwenden Sie den Befehl `lucreate`. Im folgenden Beispiel wird eine neue BU namens `zfs10092BE` erstellt. Die aktuelle BU `zfs509BE`, die in der Ausgabe von `zfs list` angezeigt wird, wird in der Ausgabe von `lustatus` erst nach Erstellung der neuen BU angegeben.

```
# lustatus
ERROR: No boot environments are configured on this system
ERROR: cannot determine list of all boot environment names
```

Zum Erstellen einer neuen ZFS-BU in demselben Pool verwenden Sie folgende Syntax:

```
# lucreate -n zfs10092BE
Analyzing system configuration.
No name for current boot environment.
```

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

```
INFORMATION: The current boot environment is not named - assigning name <zfs1009BE>.
Current boot environment is named <zfs1009BE>.
Creating initial configuration for primary boot environment <zfs1009BE>.
The device </dev/dsk/c1t1d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <zfs1009BE> PBE Boot Device </dev/dsk/c1t1d0s0>.
Comparing source boot environment <zfs1009BE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs10092BE>.
Source boot environment is <zfs1009BE>.
Creating boot environment <zfs10092BE>.
Cloning file systems from boot environment <zfs1009BE> to create boot environment <zfs10092BE>.
Creating snapshot for <rpool/ROOT/zfs1009BE> on <rpool/ROOT/zfs1009BE@zfs10092BE>.
Creating clone for <rpool/ROOT/zfs1009BE@zfs10092BE> on <rpool/ROOT/zfs10092BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs10092BE>.
Population of boot environment <zfs10092BE> successful.
Creation of boot environment <zfs10092BE> successful.
```

Beim Erstellen einer ZFS-BU innerhalb desselben Pools kommen die Klon- und Snapshot-Funktionen von ZFS zum Einsatz und die BU wird sofort erstellt. Weitere Informationen zur ZFS-Root-Migration mithilfe von Solaris Live Upgrade finden Sie unter „Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem (Solaris Live Upgrade)“ auf Seite 134.

Überprüfen Sie anschließend die neuen Boot-Umgebungen. Beispiel:

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfs1009BE                  yes      yes    yes       no     -
zfs10092BE                 yes      no     no        yes    -
# zfs list
NAME                             USED  AVAIL  REFER  MOUNTPOINT
rpool                           6.66G  60.3G    97K  /rpool
rpool/ROOT                      4.66G  60.3G    21K  legacy
rpool/ROOT/zfs10092BE             93K  60.3G  4.66G  /
rpool/ROOT/zfs1009BE            4.66G  60.3G  4.66G  /
rpool/ROOT/zfs1009BE@zfs10092BE  81.5K      -  4.66G  -
rpool/dump                      1.00G  60.3G  1.00G  -
rpool/export                      44K  60.3G    23K  /export
rpool/export/home                 21K  60.3G    21K  /export/home
rpool/swap                         1G  61.3G    16K  -
```

BEISPIEL 5–1 Erstinstallation eines boot-fähigen ZFS-Root-Dateisystems *(Fortsetzung)*

Zum Booten aus einer alternativen BU verwenden Sie den Befehl `luactivate`. Nachdem Sie die BU auf einem SPARC-basierten System aktiviert haben, können Sie mithilfe des Befehls `boot -L` verfügbare BUs identifizieren, wenn das Boot-Gerät einen ZFS-Speicher-Pool enthält. Beim Booten eines x86-Systems ermitteln Sie die zu bootende BU über das GRUB-Menü.

Geben Sie also beispielsweise bei einem SPARC-System den Befehl `boot -L` ein, um eine Liste der verfügbaren BUs anzuzeigen. Zum Booten aus der neuen BU `zfs5092BE` wählen Sie Option 2. Geben Sie dann den angezeigten Befehl `boot -Z` ein.

```
ok boot -L
Executing last command: boot -L
Boot device: /pci@1f,0/pci@1/scsi@8/disk@0,0:a  File and args: -L
1 zfs1009BE
2 zfs10092BE
Select environment to boot: [ 1 - 2 ]: 2

To boot the selected entry, invoke:
boot [<root-device>] -Z rpool/ROOT/zfs10092BE
ok boot -Z rpool/ROOT/zfs10092BE
```

Weitere Informationen zum Booten eines ZFS-Dateisystems können Sie dem Abschnitt „Booten aus einem ZFS-Root-Dateisystem“ auf Seite 161 entnehmen.

Installieren eines ZFS-Root-Dateisystems (Flash-Archiv-Installation)

In Solaris 10 10/09 kann ein Flash-Archiv auf einem System erstellt werden, auf dem ein UFS-Root-Dateisystem oder ein ZFS-Root-Dateisystem ausgeführt wird. Ein Flash-Archiv eines ZFS-Root-Pools beinhaltet die gesamte Pool-Hierarchie außer Swap- und Speicherabzug-Volumes und jeglichen ausgeschlossenen Datensätzen. Die Swap- und Speicherabzug-Volumes werden bei der Installation des Flash-Archivs erstellt. Sie können bei der Flash-Archiv-Installation wie folgt vorgehen:

- Erstellen Sie ein Flash-Archiv, das zur Installation und zum Starten eines Systems mit einem ZFS-Root-Dateisystem verwendet werden kann
- Führen Sie eine JumpStart-Installation eines Systems aus, indem Sie ein ZFS-Flash-Archiv verwenden. Durch die Erstellung eines ZFS-Flash-Archivs wird ein ganzer Root Pool geklont, nicht nur einzelne Boot-Umgebungen. Einzelne Datensätze innerhalb des Pools können mit der Option `flarcreate` und `flar` command's `-D` ausgeschlossen werden.

Beachten Sie folgende Einschränkungen, bevor Sie ein ZFS-Flash-Archiv auf einem System installieren:

- Bei der Installation eines ZFS-Flash-Archivs wird nur die JumpStart-Installation unterstützt. Sie können die interaktive Installationsoption eines Flash-Archivs nicht zur Installation eines ZFS-Root-Dateisystems auf einem System verwenden. Des Weiteren ist es nicht möglich, ein Flash-Archiv zur Installation einer ZFS-BU mit Solaris Live Upgrade zu verwenden.
- Sie können mit einem ZFS-Flash-Archiv nur ein System mit der gleichen Architektur installieren. Beispielsweise kann ein auf einem `sun4u`-System erstelltes Archiv nicht auf einem `sun4v` -System installiert werden.
- Es wird nur die vollständige Erstinstallation eines ZFS-Flash-Archivs unterstützt. Sie können weder ein anderes Flash-Archiv eines ZFS-Root-Dateisystems noch ein Hybrid-UFS/ZFS-Archiv installieren.
- Vorhandene UFS-Flash-Archive können weiterhin nur zur Installation eines UFS-Root-Dateisystems verwendet werden. Das ZFS-Flash-Archiv kann nur zur Installation eines ZFS-Root-Dateisystems verwendet werden.
- Auch wenn der gesamte Root-Pool, ausdrücklich ausgeschlossene Datasets ausgenommen, archiviert und installiert wird, ist nur die ZFS-BU, die bei Erstellung des Archivs gebootet wird, nach Installation des Flash-Archivs verwendbar. Jedoch können mit der Befehlsoption `flar` oder `flarcreate -R` *rootdir* archivierte Pools zur Archivierung eines anderen als des aktuell gebooteten Root-Pools verwendet werden.
- Ein mit einem Flash-Archiv erstellter Name eines ZFS-Root-Pools muss mit dem Namen des Master-Root-Pools übereinstimmen. Der Name des Root-Pools, der zur Erstellung des Flash-Archivs verwendet wird, wird dem neu erstellten Pool zugewiesen. Der Pool-Name kann nicht verändert werden.
- Die Befehlsoptionen `flarcreate` und `flar` zum Ein- und Ausschließen einzelner Dateien werden in einem ZFS-Flash-Archiv nicht unterstützt. Sie können nur komplette Datasets aus einem ZFS-Flash-Archiv ausschließen.
- Der Befehl `flar info` wird für ein ZFS-Flash-Archiv nicht unterstützt. Beispiel:

```
# flar info -l zfs10u8flar
ERROR: archive content listing not supported for zfs archives.
```

Nachdem Solaris 10 10/09 auf einem Master-System installiert oder darauf aktualisiert wurde, können Sie ein ZFS-Flash-Archiv zur Installation eines Zielsystems verwenden. Grundsätzlich sieht die Vorgehensweise folgendermaßen aus:

- Installieren Sie Solaris 10 10/09 auf dem Master-System oder aktualisiern Sie darauf. Nehmen Sie ggf. benutzerdefinierte Einstellungen vor.
- Erstellen Sie das ZFS-Flash-Archiv mit dem Befehl `flarcreate` auf dem Master-System. Alle Datasets im Root-Pool außer Swap- und Dump-Volumes werden im ZFS-Flash-Archiv beinhaltet.
- Erstellen Sie ein JumpStart-Profil, um die Flash-Archiv-Information auf dem Installationsserver zu beinhalten.

- Installieren Sie das ZFS-Flash-Archiv auf dem Zielsystem.

Die folgenden Archivierungsoptionen werden zur Installation eines Flash-Archivs in einem ZFS-Root-Pool unterstützt:

- Verwenden Sie den Befehl `flarcreate` oder `flar` zur Erstellung eines Flash-Archivs aus dem angegebenen ZFS-Root-Pool. Falls kein ZFS-Root-Pool angegeben ist, wird ein Flash-Archiv aus dem Standard-Root-Pool erstellt.
- Verwenden Sie `flarcreate -D` *dataset* zum Ausschließen der angegebenen Datasets aus dem Flash-Archiv. Diese Option kann mehrmals zum Ausschließen mehrerer Datasets verwendet werden.

Nach Installation eines ZFS-Flash-Archivs ist das System folgendermaßen konfiguriert:

- Die komplette auf dem System, auf dem das Flash-Archiv erstellt wurde, vorhandene Dataset-Hierarchie wird auf dem Zielsystem neu erstellt, ausgenommen jegliche Datasets, die bei der Erstellung des Archivs ausgeschlossen wurden. Die Swap- und Dump-Volumes werden im Flash-Archiv nicht beinhaltet.
- Der Root-Pool hat den gleichen Namen wie der Pool, aus dem das Archiv erstellt wurde.
- Die zum Zeitpunkt der Erstellung des Flash-Archivs aktive Boot-Umgebung ist die aktive und standardmäßige BU auf den verwendeten Systemen.

BEISPIEL 5–2 Installieren eines ZFS-Flash-Archivs auf einem System

Nachdem Sie Solaris 10 10/09 auf dem Master-System installiert oder darauf aktualisiert haben, erstellen Sie ein Flash-Archiv aus dem ZFS-Root-Pool. Beispiel:

```
# flarcreate -n zfs10u8BE zfs10u8flar
Full Flash
Checking integrity...
Integrity OK.
Running precreation scripts...
Precreation scripts done.
Determining the size of the archive...
The archive will be approximately 4.94GB.
Creating the archive...
Archive creation complete.
Running postcreation scripts...
Postcreation scripts done.

Running pre-exit scripts...
Pre-exit scripts done.
```

Erstellen Sie ein JumpStart-Profil auf dem System, das als Installationsserver dienen soll, so wie Sie es bei der Installation jedes Systems tun würden. Das folgende Profil wird beispielsweise für die Installation des `zfs10u8flar`-Archivs verwendet.

```
install_type flash_install
archive_location nfs system:/export/jump/zfs10u8flar
partitioning explicit
pool rpool auto auto auto mirror c0t1d0s0 c0t0d0s0
```

Installieren eines ZFS-Root-Dateisystems (JumpStart-Installation)

Für die Installation eines ZFS- oder UFS-Root-Dateisystems können Sie ein JumpStart-Profil erstellen. Bei einem für die Installation eines UFS-Root-Dateisystems eingerichteten Profil verhalten sich sämtliche Schlüsselwörter wie in vorherigen Solaris-Versionen.

ZFS-spezifische Profile müssen das neue Schlüsselwort `pool` enthalten. Dadurch wird ein neuer Root-Pool installiert und standardmäßig eine neue Boot-Umgebung erstellt. Sie haben die Möglichkeit, einen Namen für die Boot-Umgebung anzugeben und ein separates `/var` -Dataset zu erstellen. Hierzu dienen die Schlüsselwörter `bootenv installbe` und die Optionen `bename` und `dataset`.

Allgemeine Informationen zu den JumpStart-Funktionen finden Sie im *Solaris 10 Installationshandbuch: Benutzerdefinierte JumpStart-Installation und komplexe Installationsszenarien*.

Wenn Sie nach einer·JumpStart-Installation eines ZFS-Root-Dateisystems Zonen konfigurieren und das System patchen bzw. aktualisieren möchten, lesen Sie unter „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 141 nach.

Beispiele für ZFS-JumpStart-Profile

Dieser Abschnitt enthält einige Beispiele für ZFS-spezifische JumpStart-Profile.

Das folgende Profil führt eine Erstinstallation (angegeben mit `install_type` *initial_install*) in einem neuen Pool (angegeben mit `pool` *newpool*) durch, dessen Größe gemäß dem Schlüsselwort `auto` automatisch auf die Größe der angegebenen Festplatten bemessen wird. Swap-Bereich und Dump-Gerät werden durch Angabe des Schlüsselworts `auto` automatisch in einer Festplattenkonfiguration mit Datenspiegelung (mit Schlüsselwort `mirror` und Angabe der Datenträger als *c0t0d0s0* und *c0t1d0s0*) gesetzt. Die Merkmale der Boot-Umgebung werden mit dem Schlüsselwort `bootenv` festgelegt, eine neue BU mit dem Schlüsselwort `installbe` installiert und der `bename` mit dem Wert *s10up-xx* wird erstellt.

```
install_type initial_install
pool newpool auto auto auto mirror c0t0d0s0 c0t1d0s0
bootenv installbe bename s10up-xx
```

Das folgende Profil führt eine Erstinstallation mit dem Schlüsselwort `install_type` *initial_install* des Metaclusters `SUNWCall` in einem neuen Pool namens `newpool` und mit der Größe 80 GB durch. Dieser Pool wird mit einem Swap-Volume von 2 GB und einem Dump-Volume von ebenfalls 2 GB in einer gespiegelten Konfiguration zwei beliebiger verfügbarer Speichergeräte erstellt, die groß genug sind, um einen Pool mit einer Kapazität von 80 GB zu bilden. Sollten zwei derartige Geräte nicht verfügbar sein, schlägt die Installation fehl. Die Merkmale der Boot-Umgebung werden mit dem Schlüsselwort `bootenv` festgelegt, eine neue BU mit dem Schlüsselwort `installbe` installiert und der `bename` mit dem Wert *s10up-xx* wird erstellt.

```
install_type initial_install
cluster SUNWCall
pool newpool 80g 2g 2g mirror any any
bootenv installbe bename s10up-xx
```

Die JumpStart-Installationssyntax unterstützt die Fähigkeit, ein UFS-Dateisystem auf einem Datenträger, der auch einen ZFS-Root-Pool enthält, zu erhalten oder zu erstellen. Diese Konfiguration wird nicht für Produktionssysteme empfohlen, kann aber für die Erfordernisse der Umstellung kleinerer Systeme (beispielsweise eines Laptops) verwendet werden.

Schlüsselwörter für ZFS-JumpStart

Die folgenden Schlüsselwörter sind in ZFS-spezifischen Profilen zulässig:

`auto` Legt die Größe der Bereiche für Pool, Swap-Volume oder Dump-Volume automatisch fest. Die Größe der Festplatte wird auf Einhaltung der Mindestgröße überprüft. Wenn die Mindestgröße gegeben ist, wird die maximale Pool-Größe zugewiesen, die die Einschränkungen (Größe der Festplatten, reservierte Bereiche usw.) ermöglichen.

Wenn Sie beispielsweise `c0t0d0s0` und eines der Schlüsselwörter `all` oder `auto` angeben, wird der Bereich so groß wie möglich erstellt. Anderenfalls können Sie für den Bereich bzw. das Swap- oder Dump-Volume eine bestimmte Größe vorgeben.

Im Zusammenhang mit einem ZFS-Root-Pool verhält sich das Schlüsselwort `auto` wie das Schlüsselwort `all`, da bei Pools das Konzept des nicht belegten·Speicherplatzes nicht gültig ist.

`bootenv` Mit diesem Schlüsselwort werden die Merkmale der Boot-Umgebung festgelegt.

Das Schlüsselwort `bootenv` war bereits vorhanden, es wurden jedoch neue Optionen definiert. Zum Erstellen einer boot-fähigen ZFS-Root-Umgebung verwenden Sie das Schlüsselwort `bootenv` mit der folgenden Syntax:

`bootenv installbe bename` *BU-Name* [`dataset` *Einhängepunkt*]

`installbe` — Erstellt eine neue, durch die Option `bename` und den Eintrag *BU-Name* näher bezeichnete BU und installiert sie.

`bename` *BU-Name* — Gibt den *BU-Namen* für die Installation an.

Wenn Sie `bename` nicht zusammen mit dem Schlüsselwort `pool` angeben, wird eine Standard-BU erstellt.

`dataset` *Einhängepunkt* — Das optionale Schlüsselwort `dataset` ermöglicht es, ein vom Root-Dataset separates `/var`-Dataset anzugeben. Der Wert für *Einhängepunkt* ist derzeit auf `/var` beschränkt. So würde beispielsweise eine `bootenv`-Syntaxzeile für ein separates `/var`-Dataset wie folgt aussehen:

```
bootenv installbe bename zfsroot dataset /var
```

`pool` — Definiert den neuen, zu erstellenden Root-Pool. Die folgende Schlüsselwortsyntax muss verwendet werden:

poolname poolsize swapsize dumpsize vdevlist

Pool-Name — Gibt den Namen des zu erstellenden Pools an. Der Pool wird mit der angegebenen *Größe* und den angegebenen physischen Geräten (*vdev*s) erstellt. Geben Sie mit `Pool-Name` nicht den Namen eines vorhandenen Pools an, sonst wird der vorhandene Pool überschrieben.

Pool-Größe — Gibt die Größe des zu erstellenden Pools an. Der Wert kann `auto` oder `existing` sein. Der Wert `auto` bedeutet, dass die maximale Pool-Größe zugewiesen wird, die die Einschränkungen (Größe der Festplatten, reservierte Bereiche usw.) ermöglichen. Der Wert `existing` bewirkt, dass die Grenzen vorhandener Bereiche mit diesem Namen beibehalten und überschrieben werden. Sofern nicht `g` (GB) angegeben wird, gilt für die Größenangabe die Einheit MB.

Swap-Größe — Gibt die Größe des zu erstellenden Swap-Volumes an. Der Wert kann `auto` (die Standard-Swap-Größe wird verwendet) oder *Größe* sein (Sie geben eine Größe an). Sofern nicht `g` (GB) angegeben wird, gilt für die Größenangabe die Einheit MB.

Dump-Größe — Gibt die Größe des zu erstellenden Dump-Volumes an. Der Wert kann `auto` (die Standard-Dump-Größe wird verwendet) oder

Größe sein (Sie geben eine Größe an). Sofern nicht `g` (GB) angegeben wird, gilt für die Größenangabe die Einheit MB.

vdev-Liste Gibt eines oder mehrere Speichergeräte an, aus denen der Pool gebildet wird. Das Format der *vdev-Liste* ist identisch mit dem Format für den Befehl `zpool create`. Derzeit werden bei der Angabe mehrerer Geräte nur gespiegelte Konfigurationen unterstützt. Bei den Geräten in der *vdev-Liste* muss es sich um Bereiche für den Root-Pool handeln. Die Zeichenkette `any` bedeutet, dass die Installationssoftware ein passendes Gerät wählt.

Es können beliebig viele Festplatten gespiegelt werden, doch die Größe des Pools, das erstellt wird, richtet sich nach der kleinsten der angegebenen Festplatten. Weitere Informationen zum Erstellen von gespiegelten Speicher-Pools finden Sie unter „Speicher-Pools mit Datenspiegelung“ auf Seite 63.

Probleme mit ZFS-JumpStart

Vor Beginn einer JumpStart-Installation eines boot-fähigen ZFS-Root-Dateisystems sind folgende Probleme zu berücksichtigen.

- Ein vorhandener ZFS-Speicher-Pool kann nicht für eine JumpStart-Installation zum Erstellen eines boot-fähigen ZFS-Root-Dateisystems verwendet werden. Sie müssen einen neuen ZFS-Speicher-Pool mit Syntax wie der folgenden erstellen:

  ```
  pool rpool 20G 4G 4G c0t0d0s0
  ```

  Da kein bereits vorhandener Pool verwendet werden kann, muss die gesamte `pool`-Schlüsselwortzeile angegeben werden. Beispiel:

  ```
  install_type initial_install
  cluster SUNWCall
  pool rpool 20G 4g 4g any
  bootenv installbe bename newBE
  ```

- Sie müssen den Pool mit Festplattenbereichen anstatt mit gesamten Festplatten erstellen. Siehe hierzu „Solaris-Installation und Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung “ auf Seite 117. So ist beispielsweise die fett gedruckte Syntax unzulässig:

  ```
  install_type initial_install
  cluster SUNWCall
  pool rpool all auto auto mirror c0t0d0 c0t1d0
  bootenv installbe bename newBE
  ```

  Diese fett gedruckte Syntax ist zulässig:

```
install_type initial_install
cluster SUNWCall
pool rpool all auto auto mirror c0t0d0s0 c0t1d0s0
bootenv installbe bename newBE
```

Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem (Solaris Live Upgrade)

Die Solaris Live Upgrade-Funktionen aus früheren Versionen sind verfügbar und verhalten sich in Bezug auf UFS-Komponenten unverändert.

Folgende Leistungsmerkmale sind verfügbar:

- Für die Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem muss mit der Option `-p` ein vorhandener ZFS-Speicher-Pool bestimmt werden.
- Wenn das UFS-Root-Dateisystem Komponenten auf verschiedenen Bereichen umfasst, werden diese in den ZFS-Root-Pool überführt.
- Sie können ein System mit Zonen zwar migrieren, die unterstützten Konfigurationen sind in Solaris 10 10/08 jedoch beschränkt. Ab Solaris 10 5/09 werden mehr Zonenkonfigurationen unterstützt. Weitere Informationen finden Sie in den folgenden Abschnitten:
  - „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 141
  - „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 5/09 und Solaris 10 10/09)“ auf Seite 147

  Lesen Sie unter „Migration in ein ZFS-Root-Dateisystem mit Solaris Live Upgrade (ohne Zonen) “ auf Seite 136 nach, wenn Sie ein ZFS-Root-Dateisystem ohne Zonen migrieren möchten.
- Beim Erstellen einer ZFS-BU in demselben Pool können die ZFS-Snapshot- und -Klon-Funktionen mit Solaris Live Upgrade verwendet werden. Dadurch wird die BU-Erstellung wesentlich schneller als in vorherigen Solaris-Versionen.

Ausführliche Informationen zu den Solaris-Installations- und Solaris Live Upgrade-Funktionen finden Sie in *Solaris 10 Installationshandbuch: Solaris Live Upgrade und Planung von Upgrades*.

Die Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem läuft in folgenden Grundschritten ab:

- Installation von Solaris 10 10/08, Solaris 10 5/09 oder Solaris 10 10/09 oder Upgrade von einem vorherigen Solaris 10-Release mit dem Standard-Upgrade-Programm auf einem beliebigen unterstützten SPARC- oder x 86-System.
- Wenn Solaris 10 10/08, Solaris 10 5/09 oder Solaris 10 10/09 ausgeführt wird, erstellen Sie, falls erforderlich, einen ZFS-Speicher-Pool für das ZFS-Root-Dateisystem.
- Überführen des UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem mithilfe von Solaris Live Upgrade.
- Aktivieren der ZFS-BU mit dem Befehl `luactivate`.

Informationen zu Voraussetzungen für ZFS und Solaris Live Upgrade finden Sie unter „Solaris-Installation und Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung “ auf Seite 117.

Probleme bei der ZFS-Migration mit Solaris Live Upgrade

Für die Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem mithilfe von Solaris Live Upgrade sind folgende Probleme zu berücksichtigen:

- Die standardmäßige Upgrade-Option der grafischen Solaris-Installationsoberfläche ist für die Migration von einem UFS- in ein ZFS-Root-Dateisystem nicht verfügbar. Hierzu müssen Sie Solaris Live Upgrade verwenden.
- Vor der Solaris Live Upgrade-Operation müssen Sie den ZFS-Speicher-Pool erstellen, der zum Booten verwendet werden soll. Aufgrund von derzeitigen Boot-Einschränkungen muss der ZFS-Root-Pool außerdem mit Bereichen anstatt mit ganzen Festplatten erstellt werden. Beispiel:

  ```
  # zpool create rpool mirror c1t0d0s0 c1t1d0s0
  ```

  Vergewissern Sie sich vor dem Erstellen des neuen Pools, dass die für den Pool vorgesehenen Festplatten ein SMI-Label (VTOC) anstatt eines EFI-Labels aufweisen. Bei einer Neubezeichnung der Festplatte mit einem SMI-Label ist darauf zu achten, dass durch die Bezeichnung keine Änderungen am Partitionierungsschema erfolgt sind. In den meisten Fällen sollte ein Großteil der Festplattenkapazität in den für den Root-Pool vorgesehenen Bereichen liegen.
- Die Erstellung einer UFS-BU aus einer ZFS-BU ist mit Solaris Live Upgrade nicht möglich. Wenn Sie eine UFS-BU in eine ZFS-BU migrieren und die UFS-BU beibehalten, kann sowohl aus der UFS-BU als auch der ZFS-BU gebootet werden.

- Benennen Sie die ZFS-BUs nicht mit dem Befehl `zfs rename` um, da Solaris Live Upgrade diese Namensänderung nicht erkennt. Nachfolgende Befehle wie z. B. `ludelete` schlagen fehl. Sehen Sie davon ab, ZFS-Pools oder -Dateisysteme umzubenennen, wenn Sie bereits vorhandene BUs weiterhin verwenden möchten.
- Solaris Live Upgrade erstellt die Datasets für die BU- und ZFS-Volumes für den Swap-Bereich und das Dump-Gerät, berücksichtigt aber keine Änderungen der Eigenschaften vorhandener Datasets. Folglich müssen Sie, um eine Dataset-Eigenschaft in der neuen BU zu aktivieren, diese Eigenschaft vor der `lucreate`-Operation setzen. Beispiel:

  ```
  # zfs set compression=on rpool/ROOT
  ```

- Wenn Sie eine alternative BU erstellen, die ein Klon der primären BU ist, können Sie nicht auf die Optionen `-f`, `-x`, `-y`, `-Y` und `-z` zum Einbinden oder Ausschließen von Dateien in die bzw. aus der primären BU zurückgreifen. Diese Optionen zum Einbinden und Ausschließen können weiterhin in folgenden Fällen verwendet werden:

  ```
  UFS -> UFS
  UFS -> ZFS
  ZFS -> ZFS (different pool)
  ```

- Während ein Upgrade eines UFS-Root-Dateisystems auf ein ZFS-Root-Dateisystem mit Solaris Live Upgrade möglich ist, können Sie mit Solaris Live Upgrade kein Upgrade von Nicht-Root- oder freigegebenen Dateisystemen vornehmen.
- Der Befehl `lu` kann nicht zum Erstellen bzw. Migrieren eines ZFS-Root-Dateisystems verwendet werden.

Migration in ein ZFS-Root-Dateisystem mit Solaris Live Upgrade (ohne Zonen)

Die folgenden Beispiele veranschaulichen die Migration eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem.

Wenn Sie ein System mit Zonen migrieren oder aktualisieren, lesen Sie folgende Abschnitte:

- „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 141
- „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 5/09 und Solaris 10 10/09)“ auf Seite 147

BEISPIEL 5–3 Überführen eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem mithilfe von Solaris Live Upgrade

Das folgende Beispiel zeigt, wie eine BU eines ZFS-Root-Dateisystems aus einem UFS-Root-Dateisystem erstellt wird. Die aktuelle BU `ufs509BE`, die ein UFS-Root-Dateisystem

BEISPIEL 5–3 Überführen eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem mithilfe von Solaris Live Upgrade *(Fortsetzung)*

enthält, wird mit der Option `-c` angegeben. Wenn Sie die Option `-c` nicht angeben, wird als BU-Name standardmäßig der Gerätename verwendet. Die neue BU `zfs1009BE` wird durch die Option `-n` identifiziert. Vor der `lucreate`-Operation muss bereits ein ZFS-Speicher-Pool vorhanden sein.

Ein ZFS-Speicherpool muss auf·Datenträgerbereichen·(Slices) und darf nicht auf einer gesamten Festplatte erstellt werden, damit es bootfähig ist und später aufgerüstet werden kann. Vergewissern Sie sich vor dem Erstellen des neuen Pools, dass die für den Pool vorgesehenen Festplatten ein SMI-Label (VTOC) anstatt eines EFI-Labels aufweisen. Bei einer Neubezeichnung der Festplatte mit einem SMI-Label ist darauf zu achten, dass durch die Bezeichnung keine Änderungen am Partitionierungsschema erfolgt sind. In den meisten Fällen sollte ein Großteil der Festplattenkapazität in den für den Root-Pool vorgesehenen Bereichen liegen.

```
# zpool create mpool mirror c1t2d0s0 c2t1d0s0
# lucreate -c ufs1009BE -n zfs1009BE -p mpool
Analyzing system configuration.
No name for current boot environment.
Current boot environment is named <ufs1009BE>.
Creating initial configuration for primary boot environment <ufs1009BE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <ufs1009BE> PBE Boot Device </dev/dsk/c1t0d0s0>.
Comparing source boot environment <ufs1009BE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
The device </dev/dsk/c1t2d0s0> is not a root device for any boot environment; cannot get BE ID.
Creating configuration for boot environment <zfs1009BE>.
Source boot environment is <ufs1009BE>.
Creating boot environment <zfs1009BE>.
Creating file systems on boot environment <zfs1009BE>.
Creating <zfs> file system for </> in zone <global> on <mpool/ROOT/zfs1009BE>.
Populating file systems on boot environment <zfs1009BE>.
Checking selection integrity.
Integrity check OK.
Populating contents of mount point </>.
Copying.
Creating shared file system mount points.
Creating compare databases for boot environment <zfs1009BE>.
Creating compare database for file system </mpool/ROOT>.
Creating compare database for file system </>.
Updating compare databases on boot environment <zfs1009BE>.
```

BEISPIEL 5–3 Überführen eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem mithilfe von Solaris Live Upgrade *(Fortsetzung)*

```
Making boot environment <zfs1009BE> bootable.
Creating boot_archive for /.alt.tmp.b-qD.mnt
updating /.alt.tmp.b-qD.mnt/platform/sun4u/boot_archive
Population of boot environment <zfs1009BE> successful.
Creation of boot environment <zfs1009BE> successful.
```

Prüfen Sie nach Abschluss der `lucreate`-Operation den BU-Status mithilfe des Befehls `lustatus`. Beispiel:

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
ufs1009BE                  yes      yes    yes       no     -
zfs1009BE                  yes      no     no        yes    -
```

Kontrollieren Sie dann die Liste der ZFS-Komponenten. Beispiel:

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
mpool                 7.17G  59.8G  95.5K  /mpool
mpool/ROOT            4.66G  59.8G    21K  /mpool/ROOT
mpool/ROOT/zfs1009BE  4.66G  59.8G  4.66G  /
mpool/dump               2G  61.8G    16K  -
mpool/swap             517M  60.3G    16K  -
```

Aktivieren Sie dann mit dem Befehl `luactivate` die neue ZFS-BU. Beispiel:

```
# luactivate zfs1009BE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfs1009BE>.

**********************************************************************

The target boot environment has been activated. It will be used when you
reboot. NOTE: You MUST NOT USE the reboot, halt, or uadmin commands. You
MUST USE either the init or the shutdown command when you reboot. If you
do not use either init or shutdown, the system will not boot using the
target BE.

**********************************************************************
.
.
.
Modifying boot archive service
```

BEISPIEL 5–3 Überführen eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem mithilfe von Solaris Live Upgrade *(Fortsetzung)*

```
Activation of boot environment <zfs1009BE> successful.
```

Als Nächstes starten Sie das System in der ZFS-BU neu.

```
# init 6
```

Vergewissern Sie sich, dass die ZFS-BU aktiv ist:

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
ufs1009BE                  yes      no     no        yes    -
zfs1009BE                  yes      yes    yes       no     -
```

Wenn Sie wieder auf die UFS-BU zurückschalten, müssen Sie während des Bootens der ZFS-BU erstellte ZFS-Speicher-Pools zurückimportieren, da sie nicht automatisch in der UFS-BU verfügbar sind.

Wird die UFS-BU nicht mehr benötigt, kann Sie mit dem Befehl `ludelete` gelöscht werden.

BEISPIEL 5–4 Erstellen einer ZFS-BU aus einer ZFS-BU mit Solaris Live Upgrade

Das Erstellen einer ZFS-BU aus einer ZFS-BU in demselben Pool ist eine sehr schnelle Angelegenheit, da für diesen Vorgang die ZFS-Snapshot- und -Klon-Funktionen verwendet werden. Wenn die aktuelle BU in demselben ZFS-Pool, beispielsweise `mpool`, residiert, wird die Option `-p` nicht berücksichtigt.

Sind mehrere ZFS-BUs auf einem SPARC-System vorhanden, können mit dem Befehl `boot -L` die verfügbaren BUs ermittelt und mit dem Befehl `boot -Z` die BU ausgewählt werden, aus der gebootet werden soll. Auf x86-Systemen kann die BU über das GRUB-Menü gewählt werden. Weitere Informationen finden Sie in Beispiel 5–9.

```
# lucreate -n zfs10092BE
Analyzing system configuration.
Comparing source boot environment <zfs1009BE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs10092BE>.
Source boot environment is <zfs1009BE>.
Creating boot environment <zfs10092BE>.
```

BEISPIEL 5–4 Erstellen einer ZFS-BU aus einer ZFS-BU mit Solaris Live Upgrade *(Fortsetzung)*

```
Cloning file systems from boot environment <zfs1009BE> to create boot environment <zfs10092BE>.
Creating snapshot for <mpool/ROOT/zfs1009BE> on <mpool/ROOT/zfs1009BE@zfs10092BE>.
Creating clone for <mpool/ROOT/zfs1009BE@zfs10092BE> on <mpool/ROOT/zfs10092BE>.
Setting canmount=noauto for </> in zone <global> on <mpool/ROOT/zfs10092BE>.
Population of boot environment <zfs10092BE> successful.
Creation of boot environment <zfs10092BE> successful.
```

BEISPIEL 5–5 Upgrade der ZFS-BU (`luupgrade`)

Sie können die ZFS-BU um zusätzliche Packages oder Patches aufrüsten.

Dieser Vorgang läuft im Wesentlichen wie folgt ab:

- Erstellen Sie mit dem Befehl `lucreate` eine alternative BU.
- Aktivieren Sie die alternative BU und booten Sie.
- Fügen Sie der primären ZFS-BU mit dem Befehl `luupgrade` die Packages oder Patches hinzu.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfs1009BE                  yes      no     no        yes    -
zfs10092BE                 yes      yes    yes       no     -
# luupgrade -p -n zfs1009BE -s /net/system/export/s10u8/Solaris_10/Product SUNWchxge

Validating the contents of the media </net/system/export/s10u8/Solaris_10/Product>.
Mounting the BE <zfs1009BE>.
Adding packages to the BE <zfs1009BE>.

Processing package instance <SUNWchxge> from </net/install/export/s10u8/Solaris_10/Product>

Chelsio N110 10GE NIC Driver(sparc) 11.10.0,REV=2006.02.15.20.41
Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

This appears to be an attempt to install the same architecture and
version of a package which is already installed.  This installation
will attempt to overwrite this package.

Using </a> as the package base directory.
## Processing package information.
## Processing system information.
   4 package pathnames are already properly installed.
```

BEISPIEL 5–5 Upgrade der ZFS-BU (`luupgrade`) *(Fortsetzung)*

```
## Verifying package dependencies.
## Verifying disk space requirements.
## Checking for conflicts with packages already installed.
## Checking for setuid/setgid programs.

This package contains scripts which will be executed with super-user
permission during the process of installing this package.

Do you want to continue with the installation of <SUNWchxge> [y,n,?] y

Installing Chelsio N110 10GE NIC Driver as <SUNWchxge>

## Installing part 1 of 1.
## Executing postinstall script.

Installation of <SUNWchxge> was successful.
Unmounting the BE <zfs1009BE>.
The package add to the BE <zfs1009BE> completed.
```

Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)

Sie können ein System mit Zonen mithilfe von Solaris Live Upgrade zwar migrieren, die unterstützten Konfigurationen sind in Solaris 10 10/08 jedoch beschränkt. Nach der Installation von oder dem Upgrade auf Solaris 10 5/09 werden mehr Zonenkonfigurationen unterstützt. Weitere Informationen finden Sie unter „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 5/09 und Solaris 10 10/09)“ auf Seite 147.

In diesem Abschnitt wird beschrieben, wie Sie ein System mit Zonen so installieren und konfigieren müssen, dass es mit Solaris Live Upgrade aufgerüstet·und gepatcht werden kann. Lesen Sie unter „Migration in ein ZFS-Root-Dateisystem mit Solaris Live Upgrade (ohne Zonen) “ auf Seite 136 nach, wenn Sie ein ZFS-Root-Dateisystem ohne Zonen migrieren möchten.

Wenn Sie ein System mit Zonen migrieren möchten bzw. die Konfiguration eines Systems mit Zonen in Betracht ziehen, berücksichtigen·Sie in Solaris 10 10/08 die folgenden Vorgehensweisen:

- „So migrieren Sie ein UFS-Root-Dateisystem mit Zonen-Roots auf UFS auf ein ZFS-Root-Dateisystem (Solaris 10 10/08)“ auf Seite 142

- „So konfigurieren Sie ein ZFS-Root-Dateisystem mit Zonen-Roots auf ZFS (Solaris 10 10/08)“ auf Seite 144
- „So aktualisieren bzw. patchen Sie ein ZFS-Root-Dateisystem mit Zonen-Roots auf ZFS (Solaris 10 10/08)“ auf Seite 145
- „Lösen von Problemen mit ZFS-Einhängepunkten, die ein erfolgreiches Booten verhindern“ auf Seite 165

Halten Sie sich an die empfohlenen Vorgehensweisen zum Einrichten von Zonen mit einem ZFS-Root-Dateisystem, damit Sie für dieses System Solaris Live Upgrade verwenden können.

▼ So migrieren Sie ein UFS-Root-Dateisystem mit Zonen-Roots auf UFS auf ein ZFS-Root-Dateisystem (Solaris 10 10/08)

Gehen Sie wie folgt vor, um ein UFS-Root-Dateisystem mit installierten Zonen in ein ZFS-Root-Dateisystem zu migrieren, sodass die Konfiguration der ZFS-Zonen-Root aufgerüstet·bzw. gepatcht werden kann.

In den nach dem Beispiel aufgeführten Schritten ist der Name des Beispiel-Pools `rpool`, und der Name der aktuell aktiven Boot-Umgebung ist `S10BE*` .

1 Rüsten·Sie das System auf Solaris 10 10/08 auf, falls darauf ein früheres Solaris 10-Release installiert ist.

Weitere Informationen zum Aufrüsten·eines Systems, auf dem Solaris 10 installiert ist, finden Sie in *Solaris 10 Installationshandbuch: Solaris Live Upgrade und Planung von Upgrades*.

2 Erstellen Sie den Root-Pool.

Informationen zu Voraussetzungen für den Root-Pool finden Sie unter „Solaris-Installation und Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung “ auf Seite 117.

3 Überprüfen Sie, ob die Zonen aus der UFS-Umgebung gebootet wurden.

4 Erstellen Sie die neue Boot-Umgebung.

```
# lucreate -n S10BE2 -p rpool
```

Dieser Befehl erstellt im Root-Pool Datasets für die neue Boot-Umgebung und kopiert die aktuelle Boot-Umgebung einschließlich der Zonen in diese Datasets.

5 Aktivieren Sie die neue Boot-Umgebung.

```
# luactivate s10BE2
```

Jetzt besitzt das System ein ZFS-Root-Dateisystem, die Zonen-Roots auf UFS befinden sich jedoch noch im UFS-Root-Dateisystem. Als nächster Schritt müssen die UFS-Zonen vollständig in eine unterstützte ZFS-Konfiguration migriert werden.

6 Starten Sie das System neu.

```
# init 6
```

7 Migrieren Sie die Zonen in eine ZFS-BU.

a. Booten Sie die Zonen.

b. Erstellen Sie eine weitere BU innerhalb des Pools.

```
# lucreate S10BE3
```

c. Aktivieren Sie die neue Boot-Umgebung.

```
# luactivate S10BE3
```

d. Starten Sie das System neu.

```
# init 6
```

Dieser Schritt stellt sicher, dass die ZFS-BU und die Zonen gebootet werden können.

8 In diesem Solaris-Release müssen Sie alle potenziellen Probleme mit Einhängepunkten lösen.

Aufgrund eines Bugs in der Live Upgrade-Funktion kann es sein, dass das Booten der nichtaktiven Boot-Umgebung fehlschlägt, da ein ZFS-Dataset bzw. das ZFS-Dataset einer Zone in der Boot-Umgebung einen ungültigen Einhängepunkt besitzt.

a. Überprüfen Sie die Ausgabe des Befehls `zfs list`.

Suchen Sie ungültige temporäre Einhängepunkte. Beispiel:

```
# zfs list -r -o name,mountpoint rpool/ROOT/s10u6

NAME                                MOUNTPOINT
rpool/ROOT/s10u6                    /.alt.tmp.b-VP.mnt/
rpool/ROOT/s10u6/zones              /.alt.tmp.b-VP.mnt//zones
rpool/ROOT/s10u6/zones/zonerootA    /.alt.tmp.b-VP.mnt/zones/zonerootA
```

Der Einhängepunkt für die ZFS-Root-BU (`rpool/ROOT/s10u6`) muss `/` sein.

b. Setzen Sie die Einhängepunkte für die ZFS-BU und ihre Datasets zurück.

Beispiel:

```
# zfs inherit -r mountpoint rpool/ROOT/s10u6
# zfs set mountpoint=/ rpool/ROOT/s10u6
```

c. Starten Sie das System neu.

Wählen Sie die Boot-Umgebung aus, deren Einhängepunkte Sie gerade korrigiert haben, wenn im GRUB-Menü bzw. in der Eingabeaufforderung des OpenBoot-PROM die Option zum Booten einer spezifischen Boot-Umgebung angezeigt wird.

▼ So konfigurieren Sie ein ZFS-Root-Dateisystem mit Zonen-Roots auf ZFS (Solaris 10 10/08)

Gehen Sie wie folgt vor, wenn Sie ein ZFS-Root-Dateisystem und eine Konfiguration der ZFS-Zone-Root einrichten möchten, die aufgerüstet·bzw. gepatcht werden können. In dieser Konfiguration werden die ZFS-Zonen-Roots als ZFS-Datasets erstellt.

In den nach dem Beispiel aufgeführten Schritten ist der Name des Beispiel-Pools `rpool`, und der Name der aktuell aktiven Boot-Umgebung ist `S10BE*` .

1 Installieren Sie mithilfe des interaktiven Erstinstallationsverfahrens oder des Installationsverfahrens·Solaris JumpStart ein System mit ZFS-Root.

Weitere Informationen zur Installation eines ZFS-Root-Dateisystems mithilfe des Erstinstallationsverfahrens bzw. mit Solaris JumpStart finden Sie unter „Installieren eines ZFS-Root-Dateisystems (Erstinstallation)“ auf Seite 120 bzw. „Installieren eines ZFS-Root-Dateisystems (JumpStart-Installation)“ auf Seite 130.

2 Booten Sie das System vom neu erstellten Root-Pool.

3 Erstellen Sie ein Dataset zum Gruppieren der Zonen-Roots.

Beispiel:

```
# zfs create -o canmount=noauto rpool/ROOT/S10be/zones
```

Der Name für das Zonen-Dataset kann ein beliebiger zulässiger Dataset-Name sein. In den nachfolgend aufgeführten Schritten ist der Dataset-Name `zones`.

Das Setzen des Wertes `noauto` der Eigenschaft `canmount` garantiert, dass das Dataset nur von der speziellen Aktion von Solaris Live Upgrade und dem Startup-Code des Systems eingehängt werden kann.

4 Hängen Sie das neu erstellte Zonen-Container-Dataset ein.

```
# zfs mount rpool/ROOT/S10be/zones
```

Das Dataset wird unter `/zones` eingehängt.

5 Erstellen Sie für jede Zonen-Root ein·Dataset und hängen Sie dieses ein.

```
# zfs create -o canmount=noauto rpool/ROOT/S10be/zones/zonerootA
# zfs mount rpool/ROOT/S10be/zones/zonerootA
```

6 Setzen Sie für das Zonen-Root-Verzeichnis die entsprechenden Zugriffsrechte.

```
# chmod 700 /zones/zonerootA
```

7 Konfigurieren Sie die Zone und setzen Sie den Zonen-Pfad wie folgt:

```
# zonecfg -z zoneA
    zoneA: No such zone configured
    Use 'create' to begin configuring a new zone.
    zonecfg:zoneA> create
    zonecfg:zoneA> set zonepath=/zones/zonerootA
```

Mithilfe der folgenden Syntax können Sie festlegen, dass die Zonen beim Booten des Systems automatisch gebootet werden sollen:

```
zonecfg:zoneA> set autoboot=true
```

8 Installieren Sie die Zone.

```
# zoneadm -z zoneA install
```

9 Booten Sie die Zone.

```
# zoneadm -z zoneA boot
```

▼ So aktualisieren bzw. patchen Sie ein ZFS-Root-Dateisystem mit Zonen-Roots auf ZFS (Solaris 10 10/08)

Gehen Sie wie folgt vor, wenn Sie ein ZFS-Root-Dateisystem mit Zonen-Roots auf ZFS aufrüsten bzw. patchen müssen. Bei solchen Aktualisierungen·kann es sich um eine Systemaufrüstung oder das Anwenden von Patches handeln.

In den nachfolgenden Schritten ist `newBE` der Beispielname der Boot-Umgebung, die aufgerüstet·bzw. gepatcht werden soll.

1 Erstellen Sie die Boot-Umgebung, die aufgerüstet·bzw. gepatcht werden soll.

```
# lucreate -n newBE
```

Die vorhandene Boot-Umgebung wird mit allen Zonen geklont. Für jedes Dataset der ursprünglichen Boot-Umgebung wird ein neues Dataset erstellt. Die neuen Datasets werden im gleichen Pool erstellt, in dem sich auch der aktuelle Root-Pool befindet.

2 Wählen Sie eines der folgenden Verfahren aus, um das System aufzurüsten oder Patches auf die neue Boot-Umgebung anzuwenden.

- Rüsten·Sie das System auf.

  ```
  # luupgrade -u -n newBE -s /net/install/export/s10u7/latest
  ```

  Die Option `-s` gibt an, wo sich der Solaris-Installationsdatenträger befindet.
- Wenden Sie auf die neue Boot-Umgebung Patches an.

```
# luupgrade -t -n newBE -t -s /patchdir 139147-02 157347-14
```

3 Aktivieren Sie die neue Boot-Umgebung, wenn alle Aktualisierungen der neuen Boot-Umgeung abgeschlossen sind.

```
# luactivate newBE
```

4 Booten Sie das System von der neu aktivierten Boot-Umgebung.

```
# init 6
```

5 In Solaris 10/08 müssen Sie alle potenziellen Probleme mit Einhängepunkten lösen.

Aufgrund eines Bugs in der Live Upgrade-Funktion kann es sein, dass das Booten der nichtaktiven Boot-Umgebung fehlschlägt, da ein ZFS-Dataset bzw. das ZFS-Dataset einer Zone in der Boot-Umgebung einen ungültigen Einhängepunkt besitzt.

a. Überprüfen Sie die Ausgabe des Befehls `zfs list`.

Suchen Sie ungültige temporäre Einhängepunkte. Beispiel:

```
# zfs list -r -o name,mountpoint rpool/ROOT/newBE

NAME                                MOUNTPOINT
rpool/ROOT/newBE                    /.alt.tmp.b-VP.mnt/
rpool/ROOT/newBE/zones              /.alt.tmp.b-VP.mnt//zones
rpool/ROOT/newBE/zones/zonerootA    /.alt.tmp.b-VP.mnt/zones/zonerootA
```

Der Einhängepunkt für die ZFS-Root-BU (`rpool/ROOT/newBE`) muss `/` sein.

b. Setzen Sie die Einhängepunkte für die ZFS-BU und ihre Datasets zurück.

Beispiel:

```
# zfs inherit -r mountpoint rpool/ROOT/newBE
# zfs set mountpoint=/ rpool/ROOT/newBE
```

c. Starten Sie das System neu.

Wählen Sie die Boot-Umgebung aus, deren Einhängepunkte Sie gerade korrigiert haben, wenn im GRUB-Menü bzw. in der Eingabeaufforderung des OpenBoot-PROM die Option zum Booten einer spezifischen Boot-Umgebung angezeigt wird.

Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 5/09 und Solaris 10 10/09)

Ab Solaris 10 10/08 können Sie Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen verwenden. Zusätzliche Sparse- und Gesamtzonen-Konfigurationen werden durch Live Upgrade ab Solaris 10 5/09 unterstützt.

In diesem Abschnitt wird beschrieben, wie Sie ein System mit Zonen so konfigurieren und installieren, dass es mit Solaris Live Upgrade ab Solaris 10 5/09 aktualisiert·und gepatcht werden kann. Lesen Sie unter „Migration in ein ZFS-Root-Dateisystem mit Solaris Live Upgrade (ohne Zonen) “ auf Seite 136 nach, wenn Sie ein ZFS-Root-Dateisystem ohne Zonen migrieren möchten.

Berücksichtigen Sie folgende Punkte, wenn Sie Live Upgrade für ZFS und Zonen ab Solaris 10 5/09 verwenden.

- Wenn Sie Live Upgrade für ab Solaris 10 5/09 unterstützte Zonenkonfigurationen verwenden, müssen Sie zunächst mit dem standardmäßigen Upgrade-Programm ein Upgrade zu Solaris 10 5/09 oder Solaris 10 10/09 durchführen.
- Dann können Sie mit Live Upgrade Ihr UFS-Root-Dateisystem mit Zonen-Roots auf ein ZFS-Root-Dateisystem migrieren oder ein Upgrade oder Patch auf Ihr ZFS-Root-Dateisystem und Zonen-Roots anwenden.
- Nicht unterstützte Zonenkonfigurationen aus einem früheren Solaris 10-Release können nicht auf Solaris 10 5/09 oder Solaris 10 10/09 migriert werden.

Wenn Sie ein System mit Zonen migrieren möchten bzw. die Konfiguration eines Systems mit Zonen in Betracht ziehen, berücksichtigen·Sie ab Solaris 10 5/09 die folgenden Informationen:

- „Unterstütztes ZFS mit Zonen-Root-Konfigurationsinformationen (Solaris 10 5/09 oder Solaris 10 10/09)“ auf Seite 147
- „So erstellen Sie eine ZFS-BU mit einem ZFS-Root-Dateisystem und einer Zonen-Root (Solaris 10 5/09 oder Solaris 10 10/09)“ auf Seite 149
- „So aktualisieren bzw. patchen Sie ein ZFS-Root-Dateisystem mit Zonen-Roots (Solaris 10 5/09 oder Solaris 10 10/09)“ auf Seite 151
- „So migrieren Sie ein UFS-Root-Dateisystem mit Zonen-Roots auf ein ZFS-Root-Dateisystem (Solaris 10 5/09 oder Solaris 10 10/09)“ auf Seite 155

Unterstütztes ZFS mit Zonen-Root-Konfigurationsinformationen (Solaris 10 5/09 oder Solaris 10 10/09)

Überprüfen Sie die unterstützten Zonenkonfigurationen, bevor Sie Live Upgrade zur Migration oder zum Upgrade eines Systems mit Zonen verwenden.

- **Migrieren eines UFS-Root-Dateisystems in ein ZFS-Root-Dateisystem** – Die folgenden Zonen-Root-Konfigurationen werden unterstützt:
  - In einem Verzeichnis des UFS-Root-Dateisystems
  - In einem Unterverzeichnis eines Einhängepunkts im UFS-Root-Dateisystem
  - UFS-Root-Dateisystem mit einer Zonen-Root (wie oben beschrieben) und einem ZFS-Pool außerhalb der Root mit Zonen-Root

  Die folgende UFS/Zonenkonfiguration wird nicht unterstützt:
  - UFS-Root-Dateisystem mit einer Zonen-Root als Einhängepunkt
- **Migrieren oder Aktualisieren eines UFS-Root-Dateisystems auf ein ZFS-Root-Dateisystem** – Die folgenden Zonen-Root-Konfigurationen werden ununterstützt:
  - In einem Dataset im ZFS-Root-Pool. In einigen Fällen erstellt Live Upgrade ein Dataset für die Zonen-Root (`zoneds`), wenn vor dem Live Upgrade-Vorgang keines vorhanden ist.
  - In einem Unterverzeichnis des ZFS-Root-Dateisystems
  - In einem Dataset außerhalb des ZFS-Root-Dateisystems
  - In einem Unterverzeichnis eines Dataset außerhalb des ZFS-Root-Dateisystems
  - In einem Dataset in einem Pool außerhalb der Root. `zonepool/zones` ist beispielsweise ein Dataset, das die Zonen-Roots beinhaltet, `rpool` beinhaltet die ZFS-BU.

    ```
    zonepool
    zonepool/zones
    zonepool/zones/myzone
    rpool
    rpool/ROOT
    rpool/ROOT/myBE
    ```

    Mit der folgenden Syntax erstellt Live Upgrade Snapshots und Klone in `zonepool` und der BU `rpool`:

    ```
    # lucreate -n newBE
    ```

    Die BU `newBE` in `rpool/ROOT/newBE` wird erstellt und bietet bei Aktivierung Zugriff auf die `zonepool`-Komponenten.

    Wenn im obigen Beispiel `/zonepool/zones` ein Unterverzeichnis und kein getrenntes Dataset wäre, dann würde Live Upgrade es als Komponente des Root-Pools `rpool` migrieren.

- **Informationen zur Migration oder zum Upgrade von UFS- und ZFS-Zonen** – Berücksichtigen Sie folgende Informationen zu einer Migration oder einem Upgrade einer UFS- oder ZFS-Umgebung:
  - Wenn Sie Ihre Zonen wie unter „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 141 in Solaris 10 10/08 konfiguriert und ein Upgrade auf Solaris 10 5/09 oder Solaris 10 10/09 ausgeführt haben, sollten Sie in ein ZFS-Root-Dateisystem migrieren oder Live Upgrade für ein Upgrade auf Solaris 10 5/09 oder Solaris 10 10/09 verwenden können.
  - Erstellen Sie keine Zonen-Roots in verschachtelten Verzeichnissen wie `zones/zone1` und `zones/zone1/zone2`, andernfalls kann das Einhängen während des Boot-Vorgangs fehlschlagen.

▼ So erstellen Sie eine ZFS-BU mit einem ZFS-Root-Dateisystem und einer Zonen-Root (Solaris 10 5/09 oder Solaris 10 10/09)

Nach der Erstinstallation von Solaris 10 5/09 oder Solaris 10 10/09 zur Erstellung eines ZFS-Root-Dateisystems oder nach dem Upgrade eines ZFS-Root-Dateisystems auf Solaris 10 5/09 oder Solaris 10 10/09 mit der Funktion `luupgrade`, gehen Sie wie folgt vor. Eine mit dieser Vorgehensweise erstellte ZFS-BU kann aktualisiert oder gepatcht werden.

Das in den folgenden Schritten verwendete Solaris 10 10/09-Beispielsystem besitzt ein ZFS-Root-Dateisystem und ein Zonen-Root-Dataset in `/rpool/zones`. Eine ZFS-BU mit dem Namen `zfs10092BE` wird erstellt, die aktualisiert oder gepatcht werden kann.

1 Überprüfen Sie die vorhandenen ZFS-Dateisysteme. Beispiel:

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
rpool                 7.26G  59.7G    98K  /rpool
rpool/ROOT            4.64G  59.7G    21K  legacy
rpool/ROOT/zfs1009BE  4.64G  59.7G  4.64G  /
rpool/dump            1.00G  59.7G  1.00G  -
rpool/export            44K  59.7G    23K  /export
rpool/export/home       21K  59.7G    21K  /export/home
rpool/swap               1G  60.7G    16K  -
rpool/zones            633M  59.7G   633M  /rpool/zones
```

2 Stellen Sie sicher, dass die Zonen installiert und gebootet sind. Beispiel:

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 zfszone          running    /rpool/zones                   native   shared
```

3 Erstellen Sie die ZFS-BU. Beispiel:

```
# lucreate -n zfs10092BE
Analyzing system configuration.
Comparing source boot environment <zfs1009BE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs10092BE>.
Source boot environment is <zfs1009BE>.
Creating boot environment <zfs10092BE>.
Cloning file systems from boot environment <zfs1009BE> to create boot environment <zfs10092BE>.
Creating snapshot for <rpool/ROOT/zfs1009BE> on <rpool/ROOT/zfs1009BE@zfs10092BE>.
Creating clone for <rpool/ROOT/zfs1009BE@zfs10092BE> on <rpool/ROOT/zfs10092BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs10092BE>.
Creating snapshot for <rpool/zones> on <rpool/zones@zfs10092BE>.
Creating clone for <rpool/zones@zfs10092BE> on <rpool/zones-zfs10092BE>.
Population of boot environment <zfs10092BE> successful.
Creation of boot environment <zfs10092BE> successful.
```

4 Aktivieren Sie die ZFS-BU.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfs1009BE                  yes      yes    yes       no     -
zfs10092BE                 yes      no     no        yes    -
# luactivate zfs10092BE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfs10092BE>.
.
.
.
# init 6
```

5 Überprüfen Sie, ob die ZFS-Dateisysteme und Zonen in der neuen BU erstellt wurden. Beispiel:

```
# zfs list
NAME                                 USED  AVAIL  REFER  MOUNTPOINT
rpool                               7.38G  59.6G    98K  /rpool
rpool/ROOT                          4.72G  59.6G    21K  legacy
rpool/ROOT/zfs10092BE               4.72G  59.6G  4.64G  /
rpool/ROOT/zfs10092BE@zfs10092BE    74.0M      -  4.64G  -
rpool/ROOT/zfs1009BE                5.45M  59.6G  4.64G  /.alt.zfs1009BE
rpool/dump                          1.00G  59.6G  1.00G  -
rpool/export                          44K  59.6G    23K  /export
rpool/export/home                     21K  59.6G    21K  /export/home
rpool/swap                             1G  60.6G    16K  -
```

```
rpool/zones                       17.2M  59.6G   633M  /rpool/zones
rpool/zones-zfs1009BE              653M  59.6G   633M  /rpool/zones-zfs1009BE
rpool/zones-zfs1009BE@zfs10092BE  19.9M      -   633M  -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   - zfszone          installed  /rpool/zones                   native   shared
```

▼ So aktualisieren bzw. patchen Sie ein ZFS-Root-Dateisystem mit Zonen-Roots (Solaris 10 5/09 oder Solaris 10 10/09)

Gehen Sie wie folgt vor, wenn Sie ein ZFS-Root-Dateisystem mit Zonen-Roots in Solaris 10 5/09 oder Solaris 10 10/09 aktualisieren bzw. patchen müssen. Bei solchen Aktualisierungen·kann es sich um eine Systemaufrüstung oder das Anwenden von Patches handeln.

In den nachfolgenden Schritten ist `zfs10093BE` der Beispielname der Boot-Umgebung, die aktualisiert·bzw. gepatcht werden soll.

1 Überprüfen Sie die vorhandenen ZFS-Dateisysteme. Beispiel:

```
# zfs list
NAME                               USED  AVAIL  REFER  MOUNTPOINT
rpool                             7.38G  59.6G   100K  /rpool
rpool/ROOT                        4.72G  59.6G    21K  legacy
rpool/ROOT/zfs10092BE             4.72G  59.6G  4.64G  /
rpool/ROOT/zfs10092BE@zfs10092BE  75.0M      -  4.64G  -
rpool/ROOT/zfs1009BE              5.46M  59.6G  4.64G  /
rpool/dump                        1.00G  59.6G  1.00G  -
rpool/export                        44K  59.6G    23K  /export
rpool/export/home                   21K  59.6G    21K  /export/home
rpool/swap                          1G   60.6G    16K  -
rpool/zones                       22.9M  59.6G   637M  /rpool/zones
rpool/zones-zfs1009BE              653M  59.6G   633M  /rpool/zones-zfs1009BE
rpool/zones-zfs1009BE@zfs10092BE  20.0M      -   633M  -
```

2 Stellen Sie sicher, dass die Zonen installiert und gebootet sind. Beispiel:

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   5 zfszone          running    /rpool/zones                   native   shared
```

3 Erstellen Sie die ZFS-BU, die aktualisiert oder gepatcht werden soll. Beispiel:

```
# lucreate -n zfs10093BE
Analyzing system configuration.
Comparing source boot environment <zfs10092BE> file systems with the file
```

```
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs10093BE>.
Source boot environment is <zfs10092BE>.
Creating boot environment <zfs10093BE>.
Cloning file systems from boot environment <zfs10092BE> to create boot environment <zfs10093BE>.
Creating snapshot for <rpool/ROOT/zfs10092BE> on <rpool/ROOT/zfs10092BE@zfs10093BE>.
Creating clone for <rpool/ROOT/zfs10092BE@zfs10093BE> on <rpool/ROOT/zfs10093BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs10093BE>.
Creating snapshot for <rpool/zones> on <rpool/zones@zfs10093BE>.
Creating clone for <rpool/zones@zfs10093BE> on <rpool/zones-zfs10093BE>.
Population of boot environment <zfs10093BE> successful.
Creation of boot environment <zfs10093BE> successful.
```

4 Wählen Sie eines der folgenden Verfahren aus, um das System aufzurüsten oder Patches auf die neue Boot-Umgebung anzuwenden.

- Rüsten·Sie das System auf. Beispiel:

  ```
  # luupgrade -u -n zfs10093BE -s /net/install/export/s10uX/combined.s10s_uXwos/latest
  ```

  Die Option `-s` gibt an, wo sich der Solaris-Installationsdatenträger befindet.

  Dieser Vorgang kann viel Zeit beanspruchen.

  Ein vollständiges Beispiel für den `luupgrade`-Vorgang finden Sie in Beispiel 5–6.

- Wenden Sie auf die neue Boot-Umgebung Patches an. Beispiel:

  ```
  # luupgrade -t -n zfs10093BE -t -s /patchdir patch-id-02 patch-id-04
  ```

5 Aktivieren Sie die neue Boot-Umgebung, wenn alle Aktualisierungen der neuen Boot-Umgeung abgeschlossen sind.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfs1009BE                  yes      no     no        yes    -
zfs10092BE                 yes      yes    yes       no     -
zfs10093BE                 yes      no     no        yes    -
# luactivate zfs10093BE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfs10093BE>.
.
.
.
```

6 Booten Sie das System von der neu aktivierten Boot-Umgebung.

```
# init 6
```

Beispiel 5–6 Upgrade eines ZFS-Root-Dateisystems mit Zonen-Root auf ein ZFS-Root-Dateisystem in Solaris 10 10/09

In diesem Beispiel wird eine auf einem Solaris 10 5/09-System erstellte ZFS-BU (`zfs509BE`) mit einem ZFS-Root-Dateisystem und einer Zonen-Root in einem Pool außerhalb der Root auf Solaris 10 10/09 aktualisiert. Dieser Vorgang kann viel Zeit beanspruchen. Dann wird die aktualisierte BU (`zfs10092BE`) aktiviert. Stellen Sie vor dem Migrationsversuch sicher, dass die Zonen installiert und gebootet wurden.

In diesem Beispiel werden der Pool `zonepool`, das Dataset `/zonepool/zones` und `zfszone` wie folgt erstellt:

```
# zpool create zonepool mirror c2t1d0 c2t5d0
# zfs create zonepool/zones
# chmod 700 zonepool/zones
# zonecfg -z zfszone
zfszone: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:zfszone> create
zonecfg:zfszone> set zonepath=/zonepool/zones
zonecfg:zfszone> verify
zonecfg:zfszone> exit
# zoneadm -z zfszone install
cannot create ZFS dataset zonepool/zones: dataset already exists
Preparing to install zone <zfszone>.
Creating list of files to copy from the global zone.
Copying <8960> files to the zone.
.
.
.

# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 zfszone          running    /zonepool/zones                native   shared

# lucreate -n zfs1009BE
.
.
.
# luactivate zfs1009BE
.
.
```

```
.

# init 6
# luupgrade -u -n zfs1009BE -s /net/install/export/s10u8/combined.s10s_u8wos/latest
40410 blocks
miniroot filesystem is <lofs>
Mounting miniroot at </net/system/export/s10u8/latest/Solaris_10/Tools/Boot>
Validating the contents of the media </net/system/export/s10u8//latest>.
The media is a standard Solaris media.
The media contains an operating system upgrade image.
The media contains <Solaris> version <10>.
Constructing upgrade profile to use.
Locating the operating system upgrade program.
Checking for existence of previously scheduled Live Upgrade requests.
Creating upgrade profile for BE <zfs1009BE>.
Determining packages to install or upgrade for BE <zfs1009BE>.
Performing the operating system upgrade of the BE <zfs1009BE>.
CAUTION: Interrupting this process may leave the boot environment unstable
or unbootable.
Upgrading Solaris: 100% completed
Installation of the packages from this media is complete.
Updating package information on boot environment <zfs1009BE>.
Package information successfully updated on boot environment <zfs1009BE>.
Adding operating system patches to the BE <zfs1009BE>.
The operating system patch installation is complete.
INFORMATION: The file </var/sadm/system/logs/upgrade_log> on boot
environment <zfs1009BE> contains a log of the upgrade operation.
INFORMATION: The file </var/sadm/system/data/upgrade_cleanup> on boot
environment <zfs1009BE> contains a log of cleanup operations required.
INFORMATION: Review the files listed above. Remember that all of the files
are located on boot environment <zfs1009BE>. Before you activate boot
environment <zfs1009BE>, determine if any additional system maintenance is
required or if additional media of the software distribution must be
installed.
The Solaris upgrade of the boot environment <zfs1009BE> is complete.
Installing failsafe
Failsafe install is complete.
# luactivate zfs1009BE
# init 6
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfs509BE                   yes      no     no        yes    -
zfs1009BE                  yes      yes    yes       no     -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
```

```
   0 global          running    /                            native   shared
   - zfszone         installed  /zonepool/zones              native   shared
```

▼ So migrieren Sie ein UFS-Root-Dateisystem mit Zonen-Roots auf ein ZFS-Root-Dateisystem (Solaris 10 5/09 oder Solaris 10 10/09)

Gehen Sie wie folgt vor, um ein System mit einem UFS-Root-Dateisystem und einer Zonen-Root auf Solaris 10 5/09 oder Solaris 10 10/09 zu aktualisieren. Erstellen Sie dann mit Live Upgrade eine ZFS-BU.

In den folgenden Schritten lautet der Name der UFS-Beispiel-BU `c0t1d0s0`, die UFS-Zonen-Root heißt `zonepool/zfszone` und die ZFS-Root-BU `zfs1009` .

1 Aktualisieren·Sie das System auf Solaris 10 5/09 oder Solaris 10 10/09, falls darauf ein früheres Solaris 10-Release installiert ist.

Weitere Informationen zum Aufrüsten·eines Systems, auf dem Solaris 10 installiert ist, finden Sie in *Solaris 10 Installationshandbuch: Solaris Live Upgrade und Planung von Upgrades*.

2 Erstellen Sie den Root-Pool.

Informationen zu Voraussetzungen für den Root-Pool finden Sie unter „Solaris-Installation und Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung “ auf Seite 117.

3 Überprüfen Sie, ob die Zonen aus der UFS-Umgebung gebootet wurden. Beispiel:

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 zfszone          running    /zonepool/zones                native   shared
```

4 Erstellen Sie die neue ZFS-Boot-Umgebung. Beispiel:

```
# lucreate -c c1t1d0s0 -n zfs1009 -p rpool
```

Dieser Befehl erstellt im Root-Pool Datasets für die neue Boot-Umgebung und kopiert die aktuelle Boot-Umgebung einschließlich der Zonen in diese Datasets.

5 Aktivieren Sie die neue ZFS-Boot-Umgebung. Beispiel:

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
c1t1d0s0                   yes      yes    yes       no     -
zfs1009BE                  yes      no     no        yes    -
# luactivate zfs1009BE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfs1009BE>.
.
```

.
.

6 Starten Sie das System neu.

```
# init 6
```

7 Überprüfen Sie, ob die ZFS-Dateisysteme und Zonen in der neuen BU erstellt wurden. Beispiel:

```
# zfs list
NAME                                USED  AVAIL  REFER  MOUNTPOINT
rpool                              6.17G  60.8G    98K  /rpool
rpool/ROOT                         4.67G  60.8G    21K  /rpool/ROOT
rpool/ROOT/zfs1009BE               4.67G  60.8G  4.67G  /
rpool/dump                         1.00G  60.8G  1.00G  -
rpool/swap                          517M  61.3G    16K  -
zonepool                            634M  7.62G    24K  /zonepool
zonepool/zones                      270K  7.62G   633M  /zonepool/zones
zonepool/zones-c1t1d0s0             634M  7.62G   633M  /zonepool/zones-c1t1d0s0
zonepool/zones-c1t1d0s0@zfs1009BE   262K      -   633M  -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   - zfszone          installed  /zonepool/zones                native   shared
```

Beispiel 5–7 Migrieren eines UFS-Root-Dateisystems mit Zonen-Root in ein ZFS-Root-Dateisystem

In diesem Beispiel wird ein Solaris 10 10/09-System mit einer UFS-Root, einer Zonen-Root (`/uzone/ufszone`), einem ZFS-Pool außerhalb der Root (`pool`) und einer Zonen-Root (`/pool/zfszone`) auf ein ZFS-Root-Dateisystem migriert. Stellen Sie vor dem Migrationsversuch sicher, dass der ZFS-Root-Pool erstellt sowie die Zonen installiert und gebootet wurden.

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 ufszone          running    /uzone/ufszone                 native   shared
   3 zfszone          running    /pool/zones/zfszone            native   shared
```

```
# lucreate -c ufs1009BE -n zfs1009BE -p rpool
Analyzing system configuration.
No name for current boot environment.
Current boot environment is named <ufs509BE>.
Creating initial configuration for primary boot environment <ufs509BE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <ufs1009BE> PBE Boot Device </dev/dsk/c1t0d0s0>.
```

```
Comparing source boot environment <ufs1009BE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
The device </dev/dsk/c1t1d0s0> is not a root device for any boot environment; cannot get BE ID.
Creating configuration for boot environment <zfs1009BE>.
Source boot environment is <ufs1009BE>.
Creating boot environment <zfs1009BE>.
Creating file systems on boot environment <zfs1009BE>.
Creating <zfs> file system for </> in zone <global> on <rpool/ROOT/zfs1009BE>.
Populating file systems on boot environment <zfs1009BE>.
Checking selection integrity.
Integrity check OK.
Populating contents of mount point </>.
Copying.
Creating shared file system mount points.
Copying root of zone <ufszone> to </.alt.tmp.b-EYd.mnt/uzone/ufszone>.
Creating snapshot for <pool/zones/zfszone> on <pool/zones/zfszone@zfs1009BE>.
Creating clone for <pool/zones/zfszone@zfs1009BE> on <pool/zones/zfszone-zfs1009BE>.
Creating compare databases for boot environment <zfs1009BE>.
Creating compare database for file system </rpool/ROOT>.
Creating compare database for file system </>.
Updating compare databases on boot environment <zfs1009BE>.
Making boot environment <zfs1009BE> bootable.
Creating boot_archive for /.alt.tmp.b-DLd.mnt
updating /.alt.tmp.b-DLd.mnt/platform/sun4u/boot_archive
Population of boot environment <zfs1009BE> successful.
Creation of boot environment <zfs1009BE> successful.
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
ufs1009BE                  yes      yes    yes       no     -
zfs1009BE                  yes      no     no        yes    -
# luactivate zfs1009BE
.
.
.
# init 6
.
.
.
# zfs list
NAME                                 USED  AVAIL  REFER  MOUNTPOINT
pool                                 628M  66.3G    19K  /pool
pool/zones                           628M  66.3G    20K  /pool/zones
```

```
pool/zones/zfszone                     75.5K  66.3G   627M  /pool/zones/zfszone
pool/zones/zfszone-ufs1009BE            628M  66.3G   627M  /pool/zones/zfszone-ufs1009BE
pool/zones/zfszone-ufs1009BE@zfs1009BE   98K      -   627M  -
rpool                                  7.76G  59.2G    95K  /rpool
rpool/ROOT                             5.25G  59.2G    18K  /rpool/ROOT
rpool/ROOT/zfs1009BE                   5.25G  59.2G  5.25G  /
rpool/dump                             2.00G  59.2G  2.00G  -
rpool/swap                              517M  59.7G    16K  -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   - ufszone          installed  /uzone/ufszone                 native   shared
   - zfszone          installed  /pool/zones/zfszone            native   shared
```

ZFS-Unterstützung für Swap- und Dump-Geräte

Während einer Erstinstallation oder eines Solaris Live Upgrades in einem UFS-Dateisystem wird auf einem ZFS-Volume im ZFS-Root-Pool ein Swap-Bereich erstellt. Beispiel:

```
# swap -l
swapfile                  dev    swaplo   blocks     free
/dev/zvol/dsk/mpool/swap 253,3        16  8257520  8257520
```

Während einer Erstinstallation oder einem Solaris Live Upgrade in einem UFS-Dateisystem wird auf einem ZFS-Volume im ZFS-Root-Pool ein Dump-Gerät erstellt. Nach der Einrichtung ist keine Verwaltung des Dump-Geräts erforderlich. Beispiel:

```
# dumpadm
      Dump content: kernel pages
       Dump device: /dev/zvol/dsk/mpool/dump (dedicated)
Savecore directory: /var/crash/t2000
  Savecore enabled: yes
```

Informationen zu den von den Installationsprogrammen erstellten Swap- und Dump-Volume-Größen finden Sie unter „Solaris-Installation und Solaris Live Upgrade: Voraussetzungen für die ZFS-Unterstützung “ auf Seite 117.

Die Swap- und Dump-Volume-Größe kann während und nach der Installation geändert werden. Weitere Informationen finden Sie unter „Anpassen der Größe von ZFS-Swap- und Dump-Geräten“ auf Seite 159.

Berücksichtigen Sie bei der Arbeit mit ZFS-Swap- und Dump-Geräten die folgenden Probleme:

- Für Swap-Bereich und Dump-Gerät müssen separate ZFS-Volumes verwendet werden.
- Derzeit wird die Verwendung von Swap-Dateien in ZFS-Dateisystemen nicht unterstützt.

- Aufgrund von CR 6724860 müssen Sie bei Verwendung eines ZFS-Dump-Volumes `savecore` manuell ausführen, um einen Speicherabzug zu speichern.
- Wenn Sie den Swap-Bereich oder das Dump-Gerät nach der Installation oder dem Upgrade des Systems ändern müssen, benutzen Sie hierzu die Befehle `swap` und `dumpadm` wie in vorherigen Solaris-Versionen. Weitere Informationen finden Sie in Kapitel 20, „Configuring Additional Swap Space (Tasks)“ in *System Administration Guide: Devices and File Systems* und Kapitel 17, „Managing System Crash Information (Tasks)“ in *System Administration Guide: Advanced Administration*.

Anpassen der Größe von ZFS-Swap- und Dump-Geräten

Da bei einer ZFS-Root-Installation anders mit der Größenbemessung für Swap- und Dump-Geräte verfahren wird, müssen Sie unter Umständen vor, nach oder während der Installation diese Größen ändern.

- Die Leistung des ZFS-Dump-Volume verbessert sich, wenn das Volume mit einer Blockgröße von 128 KB erstellt wird
- Die Größe der Swap- und Dump-Volumes kann während einer Erstinstallation angepasst werden. Weitere Informationen finden Sie unter Beispiel 5–1.
- Sie können vor Ausführung von Solaris Live Upgrade Swap- und Dump-Volumes erstellen und deren Größe festlegen. Beispiel:

  1. Erstellen Sie den Speicher-Pool.

     ```
     # zpool create rpool mirror c0t0d0s0 c0t1d0s0
     ```

  2. Erstellen Sie das Dump-Gerät.

     ```
     # zfs create -V 2G rpool/dump
     ```

  3. Wählen Sie eine der folgenden Optionen, um den Swap-Bereich zu erstellen:

     - SPARC-Systeme: Erstellen Sie den Swap-Bereich. Stellen Sie die Blockgröße auf 8 KB ein.

       ```
       # zfs create -V 2G -b 8k rpool/swap
       ```

     - x86-Systeme: Erstellen Sie den Swap-Bereich. Stellen Sie die Blockgröße auf 4 KB ein.

       ```
       # zfs create -V 2G -b 4k rpool/swap
       ```

  4. Wenn ein Swap-Gerät hinzugefügt oder verändert wird, müssen Sie den Swap-Bereich aktivieren.

Solaris Live Upgrade bietet keine Möglichkeit, die Größe vorhandener Swap- und Dump-Volumes zu ändern.

- Sie können nach der Installation des Systems die Eigenschaft `volsize` des Dump-Geräts zurücksetzen. Beispiel:

```
# zfs set volsize=2G rpool/dump
# zfs get volsize rpool/dump
NAME        PROPERTY  VALUE       SOURCE
rpool/dump  volsize   2G          -
```

- Sie können die Größe des Swap-Volume ändern, bis zur Integration von CR 6765386 wird jedoch empfohlen, das Swap-Gerät zunächst zu entfernen. Erstellen Sie es dann neu. Beispiel:

```
# swap -d /dev/zvol/dsk/rpool/swap
# zfs volsize=2G rpool/swap
# swap -a /dev/zvol/dsk/rpool/swap
```

  Informationen zum Entfernen eines Swap-Geräts auf einem aktiven System finden Sie auf:

  http://www.solarisinternals.com/wiki/index.php/ZFS_Troubleshooting_Guide

- Mit folgender Profilsyntax lässt sich die Größe der Swap- und Dump-Volumes in einem JumpStart-Profil anpassen:

```
install_type initial_install
cluster SUNWCXall
pool rpool 16g 2g 2g c0t0d0s0
```

  In diesem Profil werden der Swap-Bereich und das Dump-Gerät mit den Einträgen `2g` und `2g` auf je 2 GB bemessen.

- Wenn Sie mehr Swap-Speicherplatz auf einem bereits installierten System benötigen, fügen Sie einfach ein weiteres Swap-Volume hinzu. Beispiel:

```
# zfs create -V 2G rpool/swap2
```

  Aktivieren Sie dann das neue Swap-Volume. Beispiel:

```
# swap -a /dev/zvol/dsk/rpool/swap2
# swap -l
swapfile                  dev  swaplo   blocks   free
/dev/zvol/dsk/rpool/swap  256,1      16 1058800 1058800
/dev/zvol/dsk/rpool/swap2 256,3      16 4194288 4194288
```

Booten aus einem ZFS-Root-Dateisystem

Sowohl bei SPARC- als auch bei x86-Systemen kommt die neue Art des Bootens mit Boot-Archiv zum Einsatz. Dabei handelt es sich um ein Dateisystemabbild, in dem sich die zum Booten benötigten Dateien befinden. Beim Booten aus einem ZFS-Root-Dateisystem werden die Pfadnamen des Archivs und der Kernel-Datei in dem für das Booten ausgewählten Root-Dateisystem aufgelöst.

Wenn das System für die Installation gebootet wird, dient während des gesamten Installationsvorgangs ein RAM-Datenträger als Root-Dateisystem. Es ist also nicht mehr notwendig, von einem Wechseldatenträger zu booten.

Sollten Sie eine Erstinstallation von Solaris 10 10/08 oder Solaris 10 5/09 oder mithilfe von Solaris Live Upgrade eine Migration auf ein ZFS-Root-Dateisystem in diesem Release vornehmen, können Sie sowohl auf einem SPARC- als auch auf einem x 86-System aus einem ZFS-Root-Dateisystem booten.

Das Booten aus einem ZFS-Dateisystem unterscheidet sich vom Booten aus einem UFS-Dateisystem dadurch, dass ein Gerätebezeichner bei ZFS kein einzelnes Root-Dateisystem sondern einen Speicher-Pool kennzeichnet. Ein Speicher-Pool kann mehrere *boot-fähige Datasets* oder ZFS-Root-Dateisysteme enthalten. Beim Booten aus ZFS müssen Sie ein Boot-Gerät und ein Root-Dateisystem innerhalb des Pools angeben, das durch das Boot-Gerät identifiziert ist.

Standardmäßig ist der zum Booten ausgewählte Dataset der mit der Eigenschaft `bootfs` des Pools bezeichnete Dataset. Diese Standardauswahl kann durch Angabe eines alternativen boot-fähigen Datasets im Befehl `boot -Z` außer Kraft gesetzt werden.

Booten von einer alternativen Festplatte in einem gespiegelten ZFS-Root-Pool

Bei der Installation des Systems haben Sie die Möglichkeit, einen ZFS-Root-Pool mit Datenspiegelung zu erstellen oder eine Festplatte einzubinden, um nach der Installation ein solches zu erstellen. Beachten Sie die folgenden bekannten Probleme im Zusammenhang mit gespiegelten ZFS-Root-Pools:

- CR 6668666 - Wenn Sie das Booten auf den anderen Festplatten in der Datenspiegelungskonfiguration ermöglichen möchten, müssen Sie die Boot-Informationen auf den zusätzlich eingebundenen Datenträgern mithilfe des Befehls `installboot` oder `installgrub` installieren. Dieser Schritt ist unnötig, wenn Sie zur Erstellung des gespiegelten ZFS-Root-Pools die Erstinstallationsmethode verwenden. Wenn beispielsweise den Datenträger `c0t1d0s0` als Zweiten der Datenspiegelungskonfiguration hinzugefügt wurde, lautet der Befehl `installboot` bzw. `installgrub` wie folgt:

```
sparc# installboot -F zfs /usr/platform/‘uname -i‘/lib/fs/zfs/bootblk /dev/rdsk/c0t1d0s0
```

```
x86# installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c0t1d0s0
```

- Es kann von verschiedenen Speichergeräten in einem gespiegelten ZFS-Root-Pool gebootet werden. Je nach der Hardwarekonfiguration kann es erforderlich sein, eine Aktualisierung von PROM oder BIOS vorzunehmen, um ein anderes Boot-Gerät angeben zu können.

  So ist es in diesem Pool beispielsweise möglich, von jedem Datenträger (`c1t0d0s0` oder `c1t1d0s0`) zu booten.

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

        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          mirror      ONLINE       0     0     0
            c1t0d0s0  ONLINE       0     0     0
            c1t1d0s0  ONLINE       0     0     0
```

  Auf einem SPARC-System geben Sie an der Eingabeaufforderung `ok` die alternative Festplatte ein.

```
ok boot /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@1
```

  Rufen Sie nach dem Neustart des Systems eine Bestätigung des aktiven Boot-Geräts ab. Beispiel:

```
SPARC# prtconf -vp | grep bootpath
        bootpath:  ’/pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@1,0:a’
```

  Verwenden Sie auf einem x86-System etwa folgende Syntax:

```
x86# prtconf -v|sed -n ’/bootpath/,/value/p’
        name=’bootpath’ type=string items=1
            value=’/pci@0,0/pci8086,25f8@4/pci108e,286@0/disk@0,0:a’
```

- Auf einem x86-System wählen Sie im entsprechenden BIOS-Menü eine alternative Festplatte im gespiegelten ZFS-Root-Pool aus.

Booten aus einem ZFS-Root-Dateisystem auf einem SPARC-System

Auf SPARC-Systemen mit mehreren ZFS-Boot-Umgebungen können Sie mithilfe des Befehls `luactivate` von jeder Boot-Umgebung booten. Nach dem Aktivieren einer BU können Sie sich mithilfe der Boot-Option `-L` eine Liste der Boot-Umgebungen anzeigen lassen, deren Boot-Geräte ZFS-Speicherpools enthalten.

Während der Installation und des Verfahrens mit Solaris Live Upgrade wid dem ZFS-Root-Dateisystem automatisch die Eigenschaft `bootfs` zugewiesen.

Innerhalb eines Pools können mehrere boot-fähige Datasets vorhanden sein. Standardmäßig bezieht sich die Eigenschaft `bootfs` des Pools auf den Eintrag für das boot-fähige Dataset in der Datei */Pool-Name*`/boot/menu.lst` . Der Eintrag `menu.lst` kann jedoch den Befehl `bootfs` enthalten, der ein alternatives Dataset im Pool angibt. So ist es möglich, dass die Datei `menu.lst` Einträge für mehrere Root-Dateisysteme im Pool enthält.

Wenn auf einem System ein ZFS-Root-Dateisystem installiert oder eine Migration in ein ZFS-Root-Dateisystem vorgenommen wird, wird die Datei `menu.lst` um einen Eintrag wie diesen erweitert:

```
title zfs1009BE
bootfs rpool/ROOT/zfs1009BE
title zfs509BE
bootfs rpool/ROOT/zfs509BE
```

Bei Erstellung einer neuen BU wird die Datei `menu.lst` automatisch aktualisiert.

Auf SPARC-Systemen stehen zwei neue Boot-Optionen zur Verfügung:

- Mit dem Befehl boot `-L` können Sie eine Liste der boot-fähigen Datasets innerhalb eines ZFS-Pools anzeigen lassen. Anschließend können Sie eines der boot-fähigen Datasets in der Liste auswählen. Es werden ausführliche Anweisungen zum Booten dieses Datasets angezeigt. Befolgen Sie diese Anweisungen zum Booten des ausgewählten Datasets. Diese Option ist nur verfügbar, wenn das Boot-Gerät einen ZFS-Speicher-Pool enthält.
- Zum Booten eines bestimmten ZFS-Datasets verwenden Sie den Befehl boot `-Z` *Dataset*.

BEISPIEL 5–8 Booten aus einer bestimmten ZFS-Boot-Umgebung

Wenn in einem ZFS-Speicher-Pool auf dem Boot-Gerät des Systems mehrere ZFS-BUs vorhanden sind, können Sie mit dem Befehl `luactivate` eine Standard-BU angeben.

So stehen beispielsweise die folgenden ZFS-BUs wie in der Ausgabe des Befehls `lustatus` beschrieben zur Verfügung:

BEISPIEL 5–8 Booten aus einer bestimmten ZFS-Boot-Umgebung *(Fortsetzung)*

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfs1009BE                  yes      yes    yes       no     -
zfs509BE                   yes      no     no        yes    -
```

Sind auf einem SPARC-System mehrere ZFS-BUs vorhanden, können Sie den Befehl `boot -L` verwenden. Beispiel:

```
ok boot -L
Rebooting with command: boot -L
Boot device: /pci@8,600000/SUNW,qlc@2/fp@0,0/disk@w500000e01082bbd1,0:a  File and args: -L
1 zfs1009BE
2 zfs509BE
Select environment to boot: [ 1 - 2 ]: 2
ok boot -Z rpool/ROOT/zfs509BE
```

BEISPIEL 5–9 SPARC: Booten eines ZFS-Dateisystems im Failsafe-Modus

Auf SPARC-Systemen kann aus dem in `/platform/`‘`uname -i`‘`/failsafe` befindlichen Failsafe-Archiv gebootet werden. Gehen Sie dazu wie folgt vor: Beispiel:

```
ok boot -F failsafe
```

Wenn Sie ein Failsafe-Archiv aus einem bestimmten boot-fähigen ZFS-Dataset booten möchten, verwenden Sie folgende Syntax:

```
ok boot -Z rpool/ROOT/zfs1009BE -F failsafe
```

Booten aus einem ZFS-Root-Dateisystem auf einem x86-System

Während der Installation oder des Vorgangs mit Solaris Live Upgrade werden der Datei */Pool-Name* `/boot/grub/menu.lst` die folgenden Einträge hinzugefügt, damit ZFS automatisch gebootet wird:

```
title Solaris 10 10/09 s10x_u8wos_07b X86
findroot (pool_rpool,0,a)
kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS
module /platform/i86pc/boot_archive
title Solaris failsafe
```

```
findroot (pool_rpool,0,a)
kernel /boot/multiboot kernel/unix -s -B console=ttya
module /boot/x86.miniroot-safe
```

Enthält das von GRUB als Boot-Gerät identifizierte Speichergerät einen ZFS-Speicher-Pool, wird auf Grundlage der Datei `menu.lst` das GRUB-Menü erstellt.

Auf x86-Systemen mit mehreren ZFS-BUs kann die BU über das GRUB-Menü gewählt werden. Ist das diesem Menüeintrag entsprechende Root-Dateisystem ein ZFS-Dataset, wird die folgende Option hinzugefügt.

```
-B $ZFS-BOOTFS
```

BEISPIEL 5–10 x86: Booten eines ZFS-Dateisystems

Beim Booten aus einem ZFS-Dateisystem wird das Root-Gerät durch den Boot-Parameter `-B $ZFS-BOOTFS` in der Zeile `kernel` oder `module` des GRUB-Menüeintrags angegeben. Ebenso wie alle anderen über die Option `-B` angegebenen Parameter wird dieser Wert von GRUB an den Kernel übergeben. Beispiel:

```
title Solaris 10 10/09 s10x_u8wos_07b X86
findroot (pool_rpool,0,a)
kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS
module /platform/i86pc/boot_archive
title Solaris failsafe
findroot (pool_rpool,0,a)
kernel /boot/multiboot kernel/unix -s -B console=ttya
module /boot/x86.miniroot-safe
```

BEISPIEL 5–11 x86: Booten eines ZFS-Dateisystems im Failsafe-Modus

Das Failsafe-Archiv für x86-Systeme lautet `/boot/x86.miniroot-safe` und kann durch Auswahl des Solaris-Failsafe-Eintrags im GRUB-Menü gebootet werden. Beispiel:

```
title Solaris failsafe
findroot (pool_rpool,0,a)
kernel /boot/multiboot kernel/unix -s -B console=ttya
module /boot/x86.miniroot-safe
```

Lösen von Problemen mit ZFS-Einhängepunkten, die ein erfolgreiches Booten verhindern

Die beste Methode zum Ändern der aktiven Boot-Umgebung ist der Befehl `luactivate`. Wenn das Booten von der aktiven Boot-Umgebung wegen eines ungültigen Patches oder eines

Konfigurationsfehlers fehlschlägt, können Sie nur von einer anderen Boot-Umgebung booten, wenn Sie diese zur Boot-Zeit auswählen. Bei x86-System können Sie aus dem GRUB-Menü eine alternative Boot-Umgebung auswählen oder diese (bei SPARC-Systemen) explizit vom PROM booten.

Aufgrund eines Bugs in der Live Upgrade-Funktion in Solaris 10 10/08 kann es sein, dass das Booten der nichtaktiven Boot-Umgebung fehlschlägt, da die·ZFS-Datasets bzw. das ZFS-Dataset der Zone in der Boot-Umgebung einen ungültigen Einhängepunkt besitzt. Der gleiche Bug verhindert auch das Einhängen einer BU, wenn sie ein separates `/var`-Dataset besitzt.

Wenn ein Zonen-Dataset einen ungültigen Einhängepunkt besitzt, kann dieser mit den folgenden Schritten korrigiert werden.

▼ So lösen Sie Probleme mit ZFS-Einhängepunkten

1 Booten Sie das System von einem Failsafe-Archiv.

2 Importieren Sie den Pool.

Beispiel:

```
# zpool import rpool
```

3 Überprüfen Sie nach dem Importieren des Pools die Ausgabe des Befehls `zfs list`.

Suchen Sie ungültige temporäre Einhängepunkte. Beispiel:

```
# zfs list -r -o name,mountpoint rpool/ROOT/s10u6

    NAME                              MOUNTPOINT
    rpool/ROOT/s10u6                  /.alt.tmp.b-VP.mnt/
    rpool/ROOT/s10u6/zones            /.alt.tmp.b-VP.mnt//zones
    rpool/ROOT/s10u6/zones/zonerootA  /.alt.tmp.b-VP.mnt/zones/zonerootA
```

Der Einhängepunkt für die Root-BU (`rpool/ROOT/s10u6`) muss `/` sein.

Wenn der Boot-Vorgang wegen Einhängeproblemen von `/var` fehlschlägt, sollten Sie einen ähnlichen unzulässigen Einhängepunkt für das `/var`-Dataset suchen.

4 Setzen Sie die Einhängepunkte für die ZFS-BU und ihre Datasets zurück.

Beispiel:

```
# zfs inherit -r mountpoint rpool/ROOT/s10u6
# zfs set mountpoint=/ rpool/ROOT/s10u6
```

5 Starten Sie das System neu.

Wählen Sie die Boot-Umgebung aus, deren Einhängepunkte Sie gerade korrigiert haben, wenn im GRUB-Menü bzw. in der Eingabeaufforderung des OpenBoot-PROM die Option zum Booten einer spezifischen Boot-Umgebung angezeigt wird.

Wiederherstellen von ZFS-Root-Pools oder Root-Pool-Snapshots

In den folgenden Abschnitten werden diese Vorgehensweisen beschrieben:

- „So ersetzen Sie eine Festplatte im ZFS-Root-Pool“ auf Seite 167
- „ So erstellen Sie Root-Pool-Snapshots“ auf Seite 169
- „ So erstellen Sie einen ZFS-Root-Pool neu und stellen Root-Pool-Snapshots wieder her“ auf Seite 170
- „ So erstellen Sie nach dem Booten im Failsafe-Modus ein Dateisystem im Zustand eines früheren Snapshots wieder her“ auf Seite 172

▼ So ersetzen Sie eine Festplatte im ZFS-Root-Pool

Das Ersetzen einer Festplatte im Root-Pool kann aus folgenden Gründen erforderlich sein:

- Die Root-Pool-Festplatte ist zu klein und Sie möchten sie durch eine größere Festplatte ersetzen.
- Die Root-Pool-Festplatte ist ausgefallen. Wenn Sie nach dem Festplattenausfall das System nicht mehr booten können, müssen Sie von einem anderen Medium wie einer CD oder dem Netzwerk booten, bevor Sie die Root-Pool-Festplatte ersetzen können.

Geben Sie die Pfadnamen des Boot-Geräts der aktuellen und der neuen Festplatte an, sodass Sie das Booten von der Ersatzfestplatte testen und manuell von der vorhandenen Festplatte booten können, falls das Booten von der Ersatzfestplatte fehlschlägt. In dem folgenden Beispiel lautet der Name der aktuellen Root-Pool-Festplatte `c1t10d0s0`:

```
/pci@8,700000/pci@3/scsi@5/sd@a,0
```

In dem folgenden Beispiel lautet der Name der Ersatzfestplatte zum Booten`c4t0d0s0`:

```
/pci@8,700000/pci@3/scsi@5/sd@9,0
```

1 Verbinden Sie die Ersatzfestplatte physisch.

2 Vergewissern Sie sich, dass die (neue) Ersatzfestplatte ein SMI-Label und Bereich 0 hat.

Informationen zum Umbenennen einer für den Root-Pool bestimmten Festplatte finden Sie auf:

http://www.solarisinternals.com/wiki/index.php/ZFS_Troubleshooting_Guide

3 Verbinden Sie die neue Festplatte mit dem Root-Pool.

Beispiel:

```
# zpool attach rpool c1t10d0s0 c4t0d0s0
```

4 Überprüfen Sie den Status des Root-Pools.

Beispiel:

```
# zpool status rpool
  pool: rpool
 state: ONLINE
status: One or more devices is currently being resilvered.  The pool will
        continue to function, possibly in a degraded state.
action: Wait for the resilver to complete.
 scrub: resilver in progress, 25.47% done, 0h4m to go
config:

        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          mirror      ONLINE       0     0     0
            c1t0d0s0  ONLINE       0     0     0
            c1t5d0s0  ONLINE       0     0     0

errors: No known data errors
```

5 Wenden Sie nach Abschluss des Resilvering die Boot-Blöcke auf die neue Festplatte an.

Beispiel:

Auf einem SPARC-System:

```
# installboot -F zfs /usr/platform/‘uname -i‘/lib/fs/zfs/bootblk /dev/rdsk/c1t5d0s0
```

Auf einem x86-System:

```
# installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c1t5d0s0
```

6 Überprüfen Sie, ob Sie von der neuen Festplatte booten können.

Beispiel für ein SPARC-System:

```
ok boot /pci@8,700000/pci@3/scsi@5/sd@9,0
```

7 **Wenn das System von der neuen Festplatte bootet, entfernen Sie die alte Festplatte.**

Beispiel:

```
# zpool detach rpool c1t0d0s0
```

8 **Richten Sie das System zum automatischen Booten von der neuen Festplatte ein, indem Sie den Befehl `eeprom`, den Befehl `setenv` am SPARC-Boot-PROM verwenden, oder das PC-BIOS neu konfigurieren.**

▼ So erstellen Sie Root-Pool-Snapshots

Erstellen Sie Root-Pool-Snapshots zur Wiederherstellung. Der beste Weg zum Erstellen von Root-Pool-Snapshots besteht darin, einen rekursiven Snapshot des Root-Pools zu erstellen.

Mit der unten beschriebenen Vorgehensweise wird ein rekursiver Root-Pool-Snapshot erstellt und in einer Datei innerhalb eines Pools auf einem entfernten System gespeichert. Im Falle des Fehlschlags eines Root-Pools kann das entfernte Dataset durch Verwendung von NFS eingehängt und die Snapshot-Datei im wiederhergestellten Pool gespeichert werden. Sie können Root-Pool-Snapshots auch als die eigentlichen Snapshots in einem Pool auf einem entfernten System speichern. Das Senden und Empfangen von Snapshots über ein entferntes System ist etwas komplexer, da Sie `ssh` konfigurieren oder `rsh` verwenden müssen, während das zu reparierende System von der Miniroot des Solaris-Betriebssystems gebootet wird.

Informationen zum Speichern und Wiederherstellen von Root-Pool-Snapshots auf entfernten Systemen und die aktuellsten Informationen zur Wiederherstellung von Root-Pools finden Sie auf:

`http://www.solarisinternals.com/wiki/index.php/ZFS_Troubleshooting_Guide`

Das Validieren von Snapshots, die auf entfernten Systemen als Dateien oder Snapshots gespeichert sind, ist ein wichtiger Schritt bei der Wiederherstellung von Root-Pools. Bei beiden Methoden sollten Snapshots regelmäßig wiederhergestellt werden, zum Beispiel, wenn sich die Konfiguration des Pools verändert oder das Solaris-Betriebssystem aktualisiert wird.

Im folgenden Beispiel wird das System von der Boot-Umgebung `zfs1009BE` gebootet.

1 **Erstellen Sie einen Bereich zum Speichern von Snapshots auf einem entfernten System.**

Beispiel:

```
remote# zfs create rpool/snaps
```

2 Geben Sie den Bereich für das lokale System frei.

Beispiel:

```
remote# zfs set sharenfs='rw=local-system,root=local-system' rpool/snaps
# share
-@rpool/snaps   /rpool/snaps   sec=sys,rw=local-system,root=local-system   ""
```

3 Erstellen Sie einen rekursiven Snapshot des Root-Pools.

```
local# zfs snapshot -r rpool@0804
local# zfs list
NAME                        USED  AVAIL  REFER  MOUNTPOINT
rpool                      6.17G  60.8G    98K  /rpool
rpool@0804                    0      -    98K  -
rpool/ROOT                 4.67G  60.8G    21K  /rpool/ROOT
rpool/ROOT@0804               0      -    21K  -
rpool/ROOT/zfs1009BE       4.67G  60.8G  4.67G  /
rpool/ROOT/zfs1009BE@0804   386K      -  4.67G  -
rpool/dump                 1.00G  60.8G  1.00G  -
rpool/dump@0804               0      -  1.00G  -
rpool/swap                  517M  61.3G    16K  -
rpool/swap@0804               0      -    16K  -
```

4 Senden Sie die Root-Pool-Snapshots an das entfernte System.

Beispiel:

```
local# zfs send -Rv rpool@0804 > /net/remote-system/rpool/snaps/rpool.0804
sending from @ to rpool@0804
sending from @ to rpool/swap@0804
sending from @ to rpool/ROOT@0804
sending from @ to rpool/ROOT/zfs1009BE@0804
sending from @ to rpool/dump@0804
```

▼ So erstellen Sie einen ZFS-Root-Pool neu und stellen Root-Pool-Snapshots wieder her

In diesem Beispiel wird von den folgenden Voraussetzungen ausgegangen:

- Der ZFS-Root-Pool kann nicht wiederhergestellt werden.
- Die ZFS-Root-Pool-Snapshots wurden auf einem entfernten System gespeichert und über NFS freigegeben.

Alle folgenden Schritte werden auf dem lokalen System ausgeführt.

1. **Booten Sie das System über CD/DVD oder das Netzwerk.**

   Wählen Sie auf einem SPARC-System eine der folgenden Boot-Methoden:

   ```
   ok boot net -s
   ok boot cdrom -s
   ```

   Wenn Sie die Option `-s` nicht verwenden, müssen Sie das Installationsprogramm beenden.

   Wählen Sie auf einem x86-System die Option zum Booten über DVD oder das Netzwerk. Beenden Sie das Installationsprogramm.

2. **Hängen Sie das entfernte Snapshot-Dataset ein.**

   Beispiel:

   ```
   # mount -F nfs remote-system:/rpool/snaps /mnt
   ```

   Wenn Ihre Netzwerkservices nicht konfiguriert sind, müssen Sie eventuell die IP-Adresse des *remote-system* angeben.

3. **Wenn die Root-Pool-Festplatte ersetzt wird und keine Festplattenbezeichnung enthält, die von ZFS verwendet werden kann, müssen Sie die Festplatte umbenennen.**

   Weitere Informationen zum Umbenennen der Festplatte finden Sie auf der folgenden Site:

   `http://www.solarisinternals.com/wiki/index.php/ZFS_Troubleshooting_Guide`

4. **Erstellen Sie den Root-Pool neu.**

   Beispiel:

   ```
   # zpool create -f -o failmode=continue -R /a -m legacy -o cachefile=
   /etc/zfs/zpool.cache rpool c1t1d0s0
   ```

5. **Stellen Sie die Root-Pool-Snapshots wieder her.**

   Dieser Schritt kann etwas Zeit beanspruchen. Beispiel:

   ```
   # cat /mnt/rpool.0804 | zfs receive -Fdu rpool
   ```

6. **Überprüfen Sie, ob die Root-Pool-Datasets wiederhergestellt wurden.**

   Beispiel:

   ```
   # zfs list
   NAME                        USED  AVAIL  REFER  MOUNTPOINT
   rpool                      6.17G  60.8G    98K  /a/rpool
   rpool@0804                     0      -    98K  -
   rpool/ROOT                 4.67G  60.8G    21K  /legacy
   rpool/ROOT@0804                0      -    21K  -
   rpool/ROOT/zfs1009BE       4.67G  60.8G  4.67G  /a
   rpool/ROOT/zfs1009BE@0804   398K      -  4.67G  -
   rpool/dump                 1.00G  60.8G  1.00G  -
   rpool/dump@0804                0      -  1.00G  -
   ```

```
rpool/swap                  517M  61.3G    16K  -
rpool/swap@0804                0      -    16K  -
```

7 Legen Sie die Eigenschaft `bootfs` in der Root-Pool-BU fest.

Beispiel:

```
# zpool set bootfs=rpool/ROOT/zfs1009BE rpool
```

8 Installieren Sie auf der neuen Festplatte die Boot-Blöcke.

Auf einem SPARC-System:

```
# installboot -F zfs /usr/platform/‘uname -i‘/lib/fs/zfs/bootblk /dev/rdsk/c1t5d0s0
```

Auf einem x86-System:

```
# installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c1t5d0s0
```

9 Starten Sie das System neu.

```
# init 6
```

▼ So erstellen Sie nach dem Booten im Failsafe-Modus ein Dateisystem im Zustand eines früheren Snapshots wieder her

Für dieses Verfahren wird vorausgesetzt, dass Root-Pool-Snapshots verfügbar sind. In diesem Beispiel befinden sich die Root-Pool-Snapshots auf dem lokalen System. Beispiel:

```
# zfs snapshot -r rpool@0804
# zfs list
NAME                        USED  AVAIL  REFER  MOUNTPOINT
rpool                      6.17G  60.8G    98K  /rpool
rpool@0804                    0      -    98K  -
rpool/ROOT                 4.67G  60.8G    21K  /rpool/ROOT
rpool/ROOT@0804               0      -    21K  -
rpool/ROOT/zfs1009BE       4.67G  60.8G  4.67G  /
rpool/ROOT/zfs1009BE@0804   398K      -  4.67G  -
rpool/dump                 1.00G  60.8G  1.00G  -
rpool/dump@0804               0      -  1.00G  -
rpool/swap                  517M  61.3G    16K  -
rpool/swap@0804               0      -    16K  -
```

1 Fahren Sie das System herunter und booten Sie im Failsafe-Modus.

```
ok boot -F failsafe
Multiple OS instances were found. To check and mount one of them
```

```
read-write under /a, select it from the following list. To not mount
any, select 'q'.

 1  /dev/dsk/c1t1d0s0              Solaris 10 10/09 s10s_u8wos_04 SPARC
 2  rpool:11306141908645873833     ROOT/zfs10092BE

Please select a device to be mounted (q for none) [?,??,q]: 2
mounting rpool on /a

Starting shell.
```

2 Stellen Sie die einzelnen Root-Pool-Snapshots wieder her.

```
# zfs rollback -rf rpool@0804
# zfs rollback -rf rpool/ROOT@0804
# zfs rollback -rf rpool/ROOT/zfs1009BE@0804
```

3 Booten Sie erneut im Mehrbenutzer-Modus.

```
# init 6
```

◆ ◆ ◆ KAPITEL 6

Verwalten von ZFS-Dateisystemen

Dieses Kapitel enthält ausführliche Informationen zum Verwalten von ZFS-Dateisystemen. Hier werden Konzepte wie die hierarchische Dateisystemstrukturierung, Eigenschaftsvererbung, die automatische Verwaltung von Einhängepunkten sowie die Interaktion zwischen Netzwerkdateisystemen behandelt.

ZFS-Dateisysteme setzen auf einem Speicher-Pool auf. Dateisysteme können dynamisch erstellt und gelöscht werden, ohne dass dafür zugrunde liegender Speicherplatz zugewiesen oder formatiert werden muss. Da Dateisysteme so kompakt und der Dreh- und Angelpunkt der ZFS-Administration sind, werden Sie wahrscheinliche sehr viele davon erstellen.

ZFS-Dateisysteme werden mit dem Befehl `zfs` verwaltet. Der Befehl `zfs` enthält eine Reihe von Unterbefehlen, die spezifische Operationen an Dateisystemen ausführen. In diesem Kapitel werden diese Unterbefehle ausführlich beschrieben. Snapshots, Volumes und Klone werden von diesem Befehl ebenfalls verwaltet; die Leistungsmerkmale werden jedoch erst später behandelt. Ausführliche Informationen zu Snapshots und Klonen finden Sie in Kapitel 7, „Arbeiten mit ZFS-Snapshots und -Klonen“. Ausführliche Informationen zu ZFS-Volumes finden Sie unter „ZFS-Volumes“ auf Seite 279.

Hinweis – Der Begriff *Dataset* wird in diesem Kapitel als Oberbegriff für Dateisysteme, Snapshots, Klone und Volumes verwendet.

Dieses Kapitel enthält die folgenden Abschnitte:

Erstellen und Entfernen von ZFS-Dateisystemen

ZFS-Dateisysteme können mit den Befehlen `zfs create` bzw. `zfs destroy` erstellt und gelöscht werden.

Erstellen eines ZFS-Dateisystems

ZFS-Dateisysteme werden mit dem Befehl `zfs create` erstellt. Der Befehl `create` erfordert als einziges Argument den Namen des zu erstellenden Dateisystems. Der Name des Dateisystems wird als Pfadname beginnend vom Namen des Pools angegeben:

Pool-Name/[Dateisystemname/]Dateisystemname

Der Pool-Name und die anfänglichen Dateisystemnamen im Pfad geben an, wo das neue Dateisystem in der Hierarchie erstellt wird. Alle dazwischen angegebenen Dateisystemnamen müssen bereits im Pool vorhanden sein. Der letzte Name im Pfad ist der Name des zu erstellenden Dateisystems. Der Dateisystemname muss den unter „Konventionen für das Benennen von ZFS-Komponenten“ auf Seite 45 aufgeführten Benennungskonventionen genügen.

Im folgenden Beispiel wird das Dateisystem `bonwick` im Dateisystem `tank/home` erstellt.

```
# zfs create tank/home/bonwick
```

ZFS hängt das neue Dateisystem bei fehlerfreier Erstellung automatisch ein. Dateisysteme werden standardmäßig als */Dataset* eingehängt; dabei wird der im Unterbefehl `create` angegebene Pfad verwendet. In diesem Beispiel befindet sich das neu erstellte Dateisystem `bonwick` unter `tank/home/bonwick`. Weitere Informationen zu automatisch verwalteten Einhängepunkten finden Sie unter „Verwalten von ZFS-Einhängepunkten“ auf Seite 202.

Weitere Informationen zum Befehl `zfs create` finden Sie in der Man Page `zfs(1M)`.

Sie können Dateisystemeigenschaften beim Erstellen des Dateisystems festlegen.

Im folgenden Beispiel wird ein Einhängepunkt für `/export/zfs` angegeben und für das Dateisystem `tank/home` erstellt.

```
# zfs create -o mountpoint=/export/zfs tank/home
```

Weitere Informationen zu Eigenschaften von Dateisystemen finden Sie unter „ZFS-Eigenschaften“ auf Seite 179.

Löschen eines ZFS-Dateisystems

ZFS-Dateisysteme werden mit dem Befehl `zfs destroy` gelöscht. Das gelöschte Dateisystem wird automatisch für den Netzwerkzugriff gesperrt und ausgehängt. Weitere Informationen zur automatischen Verwaltung von Einhängepunkten und gemeinsam genutzten Objekten finden Sie unter „Automatische Einhängepunkte“ auf Seite 202.

Im folgenden Beispiel wird das Dateisystem `tabriz` gelöscht.

```
# zfs destroy tank/home/tabriz
```

Achtung – Beim Ausführen des Unterbefehls `destroy` wird keine Bestätigung des Löschvorgangs angefordert. Verwenden Sie diesen Befehl deshalb mit äußerster Vorsicht.

Wenn das zu löschende Dateisystem noch von Ressourcen verwendet wird und deswegen nicht ausgehängt werden kann, schlägt der Befehl `zfs destroy` fehl. Aktive Dateisysteme werden mit der Option `-f` gelöscht. Sie sollten diese Option mit Sorgfalt verwenden, da sie aktive Dateisysteme aushängt, für den Netzwerkzugriff sperrt und löscht und somit unvorgesehenes Systemverhalten verursachen kann.

```
# zfs destroy tank/home/ahrens
cannot unmount 'tank/home/ahrens': Device busy

# zfs destroy -f tank/home/ahrens
```

Der Befehl `zfs destroy` schlägt darüber hinaus fehl, wenn in einem Dateisystem untergeordnete Dateisysteme vorhanden sind. Zum rekursiven Löschen von Dateisystemen und allen untergeordneten Dateisystemen dient die Option `-r`. Bitte beachten Sie, dass beim rekursiven Löschen auch Snapshots des Dateisystems gelöscht werden; Sie sollten diese Option deshalb mit äußerster Vorsicht verwenden.

```
# zfs destroy tank/ws
cannot destroy 'tank/ws': filesystem has children
use '-r' to destroy the following datasets:
tank/ws/billm
tank/ws/bonwick
tank/ws/maybee

# zfs destroy -r tank/ws
```

Wenn das zu löschende Dateisystem indirekte untergeordnete Dateisysteme besitzt, schlägt auch der oben beschriebene rekursive Löschbefehl fehl. Wenn Sie das Löschen *aller* untergeordneten Objekte einschließlich geklonter Dateisysteme außerhalb der Zielhierarchie erzwingen wollen, müssen Sie die Option `-R` verwenden. Verwenden Sie diese Option mit äußerster Vorsicht.

```
# zfs destroy -r tank/home/schrock
cannot destroy 'tank/home/schrock': filesystem has dependent clones
use '-R' to destroy the following datasets:
tank/clones/schrock-clone

# zfs destroy -R tank/home/schrock
```

Achtung – Beim Verwenden der Optionen `-f`, `-r` und `-R` wird keine Bestätigung des Löschvorgangs angefordert; deswegen sollten Sie diese Option mit äußerster Vorsicht verwenden.

Weitere Informationen zu Snapshots und Klonen finden Sie in Kapitel 7, „Arbeiten mit ZFS-Snapshots und -Klonen“.

Umbenennen eines ZFS-Dateisystems

Dateisysteme können mit dem Befehl `zfs rename` umbenannt werden. Der Unterbefehl `rename` führt die folgenden Operationen aus:

- Ändern des Namens eines Dateisystems
- Verlagern eines Dateisystems innerhalb der ZFS-Hierarchie
- Ändern des Namens eines Dateisystems und Verlagern dieses Systems innerhalb der ZFS-Hierarchie

Im folgenden Beispiel wird mithilfe des Unterbefehls `rename` ein einfaches Umbenennen eines Dateisystems ausgeführt:

```
# zfs rename tank/home/kustarz tank/home/kustarz_old
```

In diesem Beispiel wird das Dateisystem `kustarz` in `kustarz_alt` umbenannt.

Das folgende Beispiel zeigt die Verwendung des Befehls `zfs rename` zum Verlagern eines ZFS-Dateisystems.

```
# zfs rename tank/home/maybee tank/ws/maybee
```

In diesem Beispiel wird das Dateisystem `maybee` von `tank/home` nach `tank/ws` verlagert. Beim Verlagern von Dateisystemen mithilfe des Umbenennungsbefehls muss sich der neue Bestimmungsort innerhalb des gleichen Pools befinden, und dort muss für das Dateisystem genügend Speicherplatz verfügbar sein. Wenn der neue Bestimmungsort nicht genügend Speicherplatz besitzt (z. B. weil er das ihm zugeteilte Kontingent erreicht hat), schlägt das Verlagern mithilfe des Umbenennungsbefehls fehl.

Weitere Informationen zu Kontingenten finden Sie unter „Einstellen von ZFS-Kontingenten und -Reservierungen“ auf Seite 208.

Durch die Umbenennung wird das betreffende Dateisystem mit allen seinen untergeordneten Dateisystemen ausgehängt und wieder neu eingehängt. Die Umbenennung schlägt fehl, wenn ein aktives Dateisystem nicht ausgehängt werden kann. Wenn dieses Problem auftritt, müssen Sie das Aushängen des Dateisystems erzwingen.

Informationen zum Umbenennen von Snapshots finden Sie unter „Umbenennen von ZFS-Snapshots“ auf Seite 219.

ZFS-Eigenschaften

Mithilfe von Eigenschaften kann das Verhalten von Dateisystemen, Volumes, Snapshots und Klonen gesteuert werden Sofern nicht anders angegeben, gelten die in diesem Abschnitt beschriebenen Eigenschaften für alle Dataset-Typen.

Eigenschaften werden in zwei Typen (native und benutzerdefinierte Eigenschaften) eingeteilt. Native Eigenschaften exportieren interne Statistikinformationen und steuern das Systemverhalten von ZFS-Dateisystemen. Darüber hinaus können native Eigenschaften konfigurierbar oder schreibgeschützt sein. Benutzerdefinierte Eigenschaften wirken sich nicht auf das Verhalten von ZFS-Dateisystemen aus, können jedoch zum Versehen von Datasets mit Informationen, die für Ihre lokalen Gegebenheiten wichtig sind, verwendet werden. Weitere Informationen zu benutzerdefinierten Eigenschaften finden Sie unter „Benutzerdefinierte ZFS-Eigenschaften“ auf Seite 192.

Die meisten konfigurierbaren Eigenschaften sind vererbbar. Vererbbare Eigenschaften werden von einem übergeordneten Dateisystem an alle seine untergeordneten Dateisysteme weitergegeben.

Alle vererbbaren Eigenschaften besitzen eine zugehörige Quelle. Diese Quelle gibt an, wie eine Eigenschaft erhalten wurde. Die Eigenschaftsquelle kann die folgenden Werte besitzen:

`local`	Der Wert `local` gibt an, dass der Wert der betreffenden Eigenschaft mithilfe des Befehls `zfs set` (siehe „Setzen von ZFS-Eigenschaften“ auf Seite 196) explizit am Dataset eingestellt wurde.
`inherited from` *Dataset-Name*	Der Wert `inherited from` *Dataset-Name* bedeutet, dass die Eigenschaft von einem übergeordneten Objekt geerbt wurde.
`default`	Der Wert `default` gibt an, dass der Wert der betreffenden Eigenschaft weder geerbt noch lokal gesetzt wurde. Dieser Typ wird gesetzt, wenn ein Objekt kein übergeordnetes

Objekt besitzt und Eigenschaften nicht vom Typ `local` sind.

In der folgenden Tabelle sind schreibgeschützte und konfigurierbare native Eigenschaften von ZFS-Dateisystemen aufgeführt. Schreibgeschützte Eigenschaften sind entsprechend gekennzeichnet. Alle anderen in dieser Tabelle aufgeführten nativen Eigenschaften sind konfigurierbar. Weitere Informationen zu benutzerdefinierten Eigenschaften finden Sie unter „Benutzerdefinierte ZFS-Eigenschaften“ auf Seite 192.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften

Eigenschaft	Typ	Standardwert	Beschreibung
`aclinherit`	Zeichenkette	`secure`	Legt fest, wie Einträge in Zugriffssteuerungslisten beim Erstellen von Dateien und Verzeichnissen vererbt werden. Werte: `discard`, `noallow`, `secure` und `passthrough`. Eine Beschreibung dieser Werte finden Sie unter „Eigenschaftsmodi von Zugriffssteuerungslisten“ auf Seite 239.
`aclmode`	Zeichenkette	`groupmask`	Legt fest, wie Einträge von Zugriffssteuerungslisten während einer `chmod`-Operation geändert werden. Werte: `discard`, `groupmask` und `passthrough`. Eine Beschreibung dieser Werte finden Sie unter„Eigenschaftsmodi von Zugriffssteuerungslisten“ auf Seite 239.
`atime`	Boolesch	`on`	Legt fest, ob beim Lesen von Dateien die Dateizugriffszeit aktualisiert wird. Das Deaktivieren dieser Eigenschaft vermeidet beim Lesen von Dateien Datenverkehr durch Schreiboperationen und kann somit zu erheblichen Leistungsverbesserungen führen, obwohl es für E-Mail-Programme und ähnliche Dienstprogramme unklar sein kann.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`available`	Zahl	entf.	Eine schreibgeschützte Eigenschaft, die den für ein Dataset und alle seine untergeordneten Objekte verfügbaren Speicherplatz angibt. Dabei wird angenommen, dass im Pool keine Aktivität vorliegt. Da Speicherplatz innerhalb eines Pools gemeinsam genutzt wird, kann verfügbarer Speicherplatz durch verschiedene Faktoren wie z. B. physische Speicherkapazität des Pools, Kontingente, Reservierungen oder andere im Pool befindliche Datasets beschränkt werden. Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`avail`) referenziert werden. Weitere Informationen zur Verwaltung von Speicherplatz finden Sie unter „ZFS-Speicherplatzverwaltung“ auf Seite 56.
`canmount`	Boolesch	`on`	Legt fest, ob ein bestimmtes Dateisystem mit dem Befehl `zfs mount` eingehängt werden kann. Diese Eigenschaft ist für jedes Dateisystem einstellbar und kann nicht vererbt werden. Durch Setzen dieser Eigenschaft auf `off` kann jedoch an untergeordnete Dateisysteme ein Einhängepunkt vererbt werden; die Dateisysteme selbst werden jedoch nicht eingehängt. Wenn die Option `noauto` gesetzt ist, kann ein Dataset nur explizit ein- oder ausgehängt werden. Das Dataset wird weder beim Erstellen oder Importieren automatisch noch mit dem Befehl `zfs mount- a` eingehängt oder mit dem Befehl `zfs unmount-a` ausgehängt. Weitere Informationen finden Sie unter „Die Eigenschaft `canmount`“ auf Seite 191.
`checksum`	Zeichenkette	`on`	Aktiviert/deaktiviert die Prüfsumme zur Validierung der Datenintegrität. Der Standardwert ist `on`. Dadurch wird automatisch ein geeigneter Algorithmus (gegenwärtig `fletcher2`) gesetzt. Werte: `on, off, fletcher2` , `fletcher4` und `sha256`. Der Wert `off` deaktiviert die Integritätsprüfung von Benutzerdaten. Der Wert `off` wird nicht empfohlen.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`compression`	Zeichenkette	`off`	Aktiviert oder deaktiviert die Komprimierung für dieses Dataset. Mögliche Werte sind `on` , `off`, and `lzjb`, `gzip` oder `gzip-`*N*. Derzeit hat das Setzen der Eigenschaft auf `lzjb`, `gzip` oder `gzip-` *N* dieselbe Wirkung wie die Einstellung auf `on`. Der Standardwert ist `off`. Durch Aktivieren der Komprimierung an einem Dateisystem mit bereits vorhandenen Daten werden nur neu hinzugekommene Daten komprimiert. Vorhandene Daten bleiben unkomprimiert. Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`compress`) referenziert werden.
`compressratio`	Zahl	entf.	Schreibgeschützte Eigenschaft, die das für das betreffende Dataset erreichte Komprimierungsverhältnis als Faktor ausdrückt. Die Komprimierung kann durch Ausführen von `zfs set compression=on` *Dataset* aktiviert werden. Dieser Wert wird aus der logischen Kapazität aller Dateien und der Kapazität der entsprechend referenzierten physischen Daten berechnet. Explizite Speicherplatzeinsparungen durch die Eigenschaft `compression` sind in diesem Wert enthalten.
`copies`	Zahl	`1`	Legt die Anzahl der Kopien von Benutzerdaten pro Dateisystem fest. Verfügbare Werte sind 1, 2 oder 3. Diese Kopien werden zusätzlich zu etwaigen Redundanzfunktionen auf Pool-Ebene angelegt. Der von zusätzlichen Kopien beanspruchte Speicherplatz wird auf die entsprechende Datei bzw. das Dataset angerechnet und zählt für Kontingente und Reservierungen. Darüber hinaus wird die Eigenschaft `used` entsprechend aktualisiert, wenn das Erstellen mehrerer Kopien aktiviert wurde. Sie sollten diese Eigenschaft beim Erstellen des Dateisystems setzen, da sich das Ändern dieser Eigenschaft bei einem bereits vorhandenen Dateisystem nur auf neu geschriebene Daten auswirkt.
`creation`	Zeichenkette	entf.	Schreibgeschützte Eigenschaft, die angibt, wann das betreffende Dataset erstellt wurde (Datum/Uhrzeit).
`devices`	Boolesch	`on`	Legt fest, ob Gerätedateien im Dateisystem geöffnet werden können.
`exec`	Boolesch	`on`	Legt fest, ob Programme innerhalb des betreffenden Dateisystems ausführbar sind. Wenn diese Eigenschaft auf `off` gesetzt ist, werden `mmap(2)`-Aufrufe mit `PROT_EXEC` nicht zugelassen.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`mounted`	Boolesch	entf.	Schreibgeschützte Eigenschaft, die angibt, ob gegenwärtig ein Dateisystem, Klon oder Snapshot eingehängt ist. Diese Eigenschaft gilt nicht für Volumes. Werte: „yes" oder „no".
`mountpoint`	Zeichenkette	entf.	Legt den Einhängepunkt für das betreffende Dateisystem fest. Wenn die Eigenschaft `mountpoint` für ein Dateisystem geändert wird, werden das Dateisystem selbst und alle seine untergeordneten Dateisysteme, die den Einhängepunkt geerbt haben, ausgehängt. Wenn der neue Wert `legacy` ist, bleiben sie ausgehängt. Andernfalls werden sie, wenn die Eigenschaft vorher auf `legacy` oder `none` gesetzt war bzw. die Dateisysteme vor dem Ändern der Eigenschaft eingehängt waren, am neuen Bestimmungsort automatisch eingehängt. Darüber hinaus wird der Netzwerkzugriff auf die betreffenden Dateisysteme gesperrt und am neuen Bestimmungsort freigegeben. Weitere Informationen zur Verwendung dieser Eigenschaft finden Sie unter „Verwalten von ZFS-Einhängepunkten" auf Seite 202.
`primarycache`	Zeichenkette	`off`	Kontrolliert, welche Daten im ARC im Cache gespeichert werden. Mögliche Werte sind `all`, `none` und `metadata`. `all`, both user data and metadata are cached. Ist diese Eigenschaft auf `none` gesetzt, werden weder Benutzerdaten noch Metadaten im Cache gespeichert. Ist diese Eigenschaft auf `metadata` gesetzt, werden nur Metadaten im Cache gespeichert. Der Standardwert ist `all`.
`origin`	Zeichenkette	entf.	Schreibgeschützte Eigenschaft für geklonte Dateisysteme bzw. Volumes, die angibt, aus welchem Snapshot der betreffende Klon erstellt wurde. Das ursprüngliche Dateisystem kann (auch mit den Optionen `-r` oder `-f`) nicht gelöscht werden, solange der betreffende Klon existiert. Bei ungeklonten Dateisystemen besitzt diese Eigenschaft den Wert none.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`quota`	Zahl (oder `none`)	`none`	Beschränkt den Speicherplatz, der von Datasets und untergeordneten Objekten belegt werden kann. Diese Eigenschaft erzwingt einen absoluten Grenzwert des Speicherplatzes, der belegt werden kann. Dazu zählt auch der von untergeordneten Datasets einschließl. Dateisystemen und Snapshots belegte Speicherplatz. Das Setzen eines Kontingentes für ein untergeordnetes Objekt eines Datasets, für den bereits ein Kontingent definiert wurde, überschreibt den vom übergeordneten Dataset geerbten Wert nicht, sondern setzt darüber hinaus einen zusätzlichen Grenzwert. Kontingente können nicht für Volumes eingestellt werden, da deren Eigenschaft `volsize` bereits ein Kontingent darstellt. Weitere Informationen zum Einstellen von Kontingenten finden Sie unter „Setzen von Kontingenten für ZFS-Dateisysteme“ auf Seite 209.
`readonly`	Boolesch	`off`	Legt fest, ob das betreffende Dataset geändert werden kann. Wenn diese Eigenschaft auf `on` gesetzt ist, können am Dataset keine Änderungen vorgenommen werden. Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`rdonly`) referenziert werden.
`recordsize`	Zahl	`128K`	Legt eine empfohlene Blockgröße für Dateien im Dateisystem fest. Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`recsize`) referenziert werden. Eine ausführliche Beschreibung finden Sie unter „Die Eigenschaft `recordsize`“ auf Seite 191.
`referenced`	Zahl	entf.	Eine schreibgeschützte Eigenschaft, die die Datenmenge festlegt, auf die ein Dataset zugreifen kann. Solche Daten können gemeinsam im Pool genutzt werden, dies braucht jedoch nicht der Fall zu sein. Bei der Erstellung eines Snapshots bzw. Klons wird anfänglich der gleiche Speicherplatzbetrag referenziert, der der Kapazität des Dateisystems bzw. Snapshots entspricht, aus dem er erstellt wurde, da der Inhalt identisch ist. Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`refer`) referenziert werden.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`refquota`	Zahl (oder „none“)	`none`	Legt fest, wie viel Speicherplatz ein Dataset belegen kann. Die Eigenschaft erzwingt einen absoluten Grenzwert des belegbaren Speicherplatzes. Dieser Grenzwert umfasst keinen durch untergeordnete Objekte wie z. B. Snapshots und Klone belegten Speicherplatz.
`refreservation`	Zahl (oder „none“)	`none`	Der garantierte Mindestspeicherplatz für ein Dataset ohne untergeordnete Objekte, wie etwa Snapshots oder Klone. Wenn der tatsächlich belegte Speicherplatz unter dem hier angegebenen Wert liegt, wird der Dataset behandelt, als belege er den mit `refreservation` angegebenen Speicherplatz. Reservierungen durch `refreservation` werden in die Berechnung der Speicherplatzkapazität für den diesem Dataset übergeordneten Dataset einbezogen und auf die Kontingente und Reservierung für den übergeordneten Dataset angerechnet. Wenn `refreservation` gesetzt ist, wird ein Snapshot nur zugelassen, wenn außerhalb dieser Reservierung genügend freier Speicherplatz im Pool vorhanden ist, um die Menge der aktuell *referenzierten* Byte im Dataset aufzunehmen. Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`refreserv`) referenziert werden.
`reservation`	Zahl (oder „none“)	`none`	Der Mindestspeicherplatz für ein Dataset und seine untergeordneten Objekte. Wenn der tatsächlich belegte Speicherplatz unter dem hier angegebenen Wert liegt, wird das Dataset behandelt, als belege es den in dieser Reservierung angegebenen Speicherplatz. Reservierungen werden in die Berechnung der Speicherplatzkapazität für das diesem Dataset übergeordnete Dataset einbezogen und auf die Kontingente und Reservierung für das übergeordnete Dataset angerechnet. Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`reserv`) referenziert werden. Weitere Informationen dazu finden Sie unter „Setzen von Reservierungen für ZFS-Dateisysteme“ auf Seite 213.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`secondarycache`	Zeichenkette	`off`	Kontrolliert, welche Daten im L2ARC im Cache gespeichert werden. Mögliche Werte sind `all`, `none` und `metadata`. `all`, both user data and metadata are cached. Ist diese Eigenschaft auf `none` gesetzt, werden weder Benutzerdaten noch Metadaten im Cache gespeichert. Ist diese Eigenschaft auf `metadata` gesetzt, werden nur Metadaten im Cache gespeichert. Der Standardwert ist `all`.
`setuid`	Boolesch	`on`	Legt fest, ob das `setuid`Bit im Dateisystem berücksichtigt wird.
`sharenfs`	Zeichenkette	`off`	Legt fest, ob das Dateisystem über NFS zugänglich ist und welche Optionen verwendet werden. Wenn diese Eigenschaft auf `on` gesetzt ist, wird der Befehl `zfs share` ohne Optionen aufgerufen. Andernfalls wird der Befehl `zfs share` mit den Optionen aufgerufen, die dem Inhalt dieser Eigenschaft entsprechen. Wenn die Eigenschaft auf `off` gesetzt ist, wird das Dateisystem über die älteren Befehle `share` und `unshare` und die Datei `dfstab` verwaltet. Weitere Informationen zur Nutzung von ZFS-Dateisystemen für den Netzwerkzugang finden Sie unter „Freigeben und Sperren von ZFS-Dateisystemen“ auf Seite 206.
`snapdir`	Zeichenkette	`hidden`	Legt fest, ob das Verzeichnis `.zfs` in der Root des Dateisystems verborgen oder sichtbar ist. Weitere Informationen zur Verwendung von Snapshots finden Sie unter „Überblick über ZFS-Snapshots“ auf Seite 217.
`type`	Zeichenkette	entf.	Schreibgeschützte Eigenschaft, die den Typ des betreffenden Datasets (`filesystem` (Dateisystem oder Klon), `volume` oder `snapshot`) angibt.
`used`	Zahl	entf.	Schreibgeschützte Eigenschaft, die den für ein Dataset und alle seine untergeordneten Objekte belegten Speicherplatz angibt. Eine ausführliche Beschreibung finden Sie unter „Die Eigenschaft `used`“ auf Seite 189.
`usedbychildren`	Zahl	off	Schreibgeschützte Eigenschaft, die den Speicherplatz angibt, der von untergeordneten Elementen dieses Datasets beansprucht wird, und der beim Löschen dieser untergeordneten Elemente frei werden würde. Die Abkürzung für die Eigenschaft ist `usedchild`.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`usedbydataset`	Zahl	off	Schreibgeschützte Eigenschaft, die den Speicherplatz angibt, der vom Dataset selbst beansprucht wird und der beim Löschen des Datasets und vorherigem Löschen aller Snapshots und Entfernen von `refreservation` frei werden würde. Die Abkürzung für diese Eigenschaft ist `usedds`.
`usedbyrefreservation`	Zahl	off	Schreibgeschützte Eigenschaft, die den von einem `refreservation`-Set auf diesem Dataset beanspruchten Speicherplatz angibt, der beim Entfernen von `refreservation` frei werden würde. Die Abkürzung für die Eigenschaft ist `usedds`.
`usedbysnapshots`	Zahl	off	Schreibgeschützte Eigenschaft, die den von Snapshots dieses Datasets beanspruchten Speicherplatz angibt. Insbesondere geht es dabei um den Speicherplatz, der beim Löschen aller Snapshots des Datasets frei werden würde. Beachten Sie, dass es sich dabei nicht einfach um die Summe der `used` Eigenschaften der Snapshots handelt, da Speicherplatz von mehreren Snapshots geteilt werden kann. Die Abkürzung für diese Eigenschaft ist `usedsnap`.
`volsize`	Zahl	entf.	Legt die logische Größe eines Volumes fest (gilt nur für Volumes). Eine ausführliche Beschreibung finden Sie unter „Die Eigenschaft `volsize`“ auf Seite 192.
`volblocksize`	Zahl	`8 KB`	Legt die Blockgröße eines Volumes fest (gilt nur für Volumes). Nach dem Schreiben von Daten auf das betreffende Volume kann die Blockgröße nicht mehr geändert werden, deswegen muss diese Eigenschaft bei der Erstellung des Volumes gesetzt werden. Die Standardblockgröße für Volumes ist 8 KB. Es sind alle Werte zur Potenz von 2 im Bereich von 512 Byte bis 128 KB zulässig. Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`volblock`) referenziert werden.

TABELLE 6–1 Beschreibungen nativer ZFS-Eigenschaften *(Fortsetzung)*

Eigenschaft	Typ	Standardwert	Beschreibung
`zoned`	Boolesch	entf.	Gibt an, ob dieses Dataset zu einer nicht-globalen Zone hinzugefügt wurde. Wenn diese Eigenschaft gesetzt ist, befindet sich der Einhängepunkt nicht in der globalen Zone, und ZFS kann ein solches Dateisystem auf Anforderung nicht einhängen. Bei der ersten Installation einer Zone wird diese Eigenschaft für alle hinzugefügten Dateisysteme gesetzt. Weitere Informationen zur Verwendung von ZFS mit installierten Zonen finden Sie in „Verwendung von ZFS in einem Solaris-System mit installierten Zonen“ auf Seite 282.
`xattr`	Boolesch	`on`	Legt fest, ob für das betreffende Dateisystem erweiterte Attribute aktiviert oder deaktiviert sind. Der Standardwert ist `on`.

Schreibgeschützte native ZFS-Eigenschaften

Schreibgeschützte native Eigenschaften können gelesen, aber nicht gesetzt werden. Schreibgeschützte native Eigenschaften werden nicht vererbt. Einige native Eigenschaften gelten nur für bestimmte Dataset-Typen. In solchen Fällen ist der entsprechende Dataset-Typ in der Beschreibung in Tabelle 6–1 aufgeführt.

Die schreibgeschützten nativen Eigenschaften sind hier aufgeführt und in Tabelle 6–1 beschrieben.

- `available`
- `creation`
- `mounted`
- `origin`
- `compressratio`
- `referenced`
- `type`
- `used`

  Ausführliche Informationen finden Sie unter „Die Eigenschaft `used`“ auf Seite 189.
- `usedbychildren`
- `usedbydataset`
- `usedbyrefreservation`
- `usedbysnapshots`

Weitere Informationen zur Verwaltung von Speicherplatz (einschließl. Informationen zu den Eigenschaften `used`, `referenced` und `available`) finden Sie unter „ZFS-Speicherplatzverwaltung“ auf Seite 56.

Die Eigenschaft `used`

Der vom betreffenden Dataset und allen seinen untergeordneten Datasets belegte Speicherplatz. Dieser Wert wird gegen die für das Dataset gesetzten Kontingente und Reservierungen geprüft. Der belegte Speicherplatz enthält nicht die Reservierungen für das Dataset selbst, schließt jedoch die Reservierungen für alle seine untergeordneten Datasets ein. Die Menge an Speicherplatz, die das Dataset aufgrund seines übergeordneten Datasets belegt sowie der beim rekursiven Löschen des Datasets freigegebene Speicherplatz ist der größere der Werte für den belegten Speicherplatz und die Reservierung.

Bei der Erstellung von Snapshots wird dieser Speicherplatz anfänglich vom Snapshot und dem Dateisystem (sowie eventuellen früheren Snapshots) gemeinsam genutzt. Wenn sich ein Dateisystem mit der Zeit ändert, gehört früher gemeinsam genutzter Speicherplatz dann ausschließlich dem Snapshot und wird in die Berechnung des vom Snapshot belegten Speicherplatzes einbezogen. Der von einem Snapshot belegte Speicherplatz gibt dessen einmalige Daten an. Darüber hinaus kann durch das Löschen von Snapshots der Speicherplatz, der Snapshots eindeutig zugewiesen ist (und von diesen verwendet wird), größer werden. Weitere Informationen zu Snapshots und Speicherplatzaspekten finden Sie in „Verhalten bei ungenügendem Speicherplatz“ auf Seite 57.

Der belegte, verfügbare und referenzierte Speicherplatz berücksichtigt keinen durch anstehende Änderungen belegten Speicherplatz. Anstehende Änderungen werden im Allgemeinen innerhalb weniger Sekunden abgeschlossen. Nach Abschluss eines Schreibvorgangs auf dem Datenträger durch `fsync(3c)` oder `O_SYNC` werden die Informationen zum belegten Speicherplatz nicht notwendigerweise sofort aktualisiert.

Die Informationen der Eigenschaften `usedbychildren`, `usedbydataset` , `usedbyrefreservation` und `usedbysnapshots` kann mit dem Befehl `zfs list - o space` angezeigt werden. Diese Eigenschaften teilen die `used` Eigenschaft in von untergeordneten Elementen beanspruchten Speicherplatz auf. Weitere Informationen finden Sie in Tabelle 6–1.

Konfigurierbare native ZFS-Eigenschaften

Konfigurierbare native Eigenschaften können gelesen und gesetzt werden. Konfigurierbare native Eigenschaften werden mithilfe des Befehls `zfs set`(siehe „Setzen von ZFS-Eigenschaften“ auf Seite 196) bzw. `zfs create` (siehe „Erstellen eines ZFS-Dateisystems“ auf Seite 176) gesetzt. Außer Kontingenten und Reservierungen werden konfigurierbare Eigenschaften vererbt. Weitere Informationen zu Kontingenten und Reservierungen finden Sie unter „Einstellen von ZFS-Kontingenten und -Reservierungen“ auf Seite 208.

Einige konfigurierbare native Eigenschaften gelten nur für bestimmte Dataset-Typen. In solchen Fällen ist der entsprechende Dataset-Typ in der Beschreibung in Tabelle 6–1 aufgeführt. Sofern nichts Anderes vermerkt ist, gilt eine Eigenschaft für alle Dataset-Typen: Dateisysteme, Volumes, Klone und Snapshots.

Die konfigurierbaren nativen Eigenschaften sind hier aufgeführt und in Tabelle 6–1 beschrieben.

- `aclinherit`

  Eine ausführliche Beschreibung finden Sie unter „Eigenschaftsmodi von Zugriffssteuerungslisten“ auf Seite 239.
- `aclmode`

  Eine ausführliche Beschreibung finden Sie unter „Eigenschaftsmodi von Zugriffssteuerungslisten“ auf Seite 239.
- `atime`
- `canmount`
- `checksum`
- `compression`
- `copies`
- `devices`
- `exec`
- `mountpoint`
- `primarycache`
- `quota`
- `readonly`
- `recordsize`

  Eine ausführliche Beschreibung finden Sie unter „Die Eigenschaft `recordsize`“ auf Seite 191.
- `refquota`
- `refreservation`
- `reservation`
- `secondarycache`
- `sharenfs`
- `setuid`
- `snapdir`
- `volsize`

  Eine ausführliche Beschreibung finden Sie unter „Die Eigenschaft `volsize`“ auf Seite 192.

- volblocksize
- zoned

Die Eigenschaft canmount

Wenn diese Eigenschaft auf „off" gesetzt ist, kann das Dateisystem nicht mithilfe der Befehle zfs mount bzw. zfs mount -a eingehängt werden. Diese Eigenschaft gleicht dem Setzen der Eigenschaft mountpoint auf none. Der Unterschied besteht darin, dass das Dataset trotzdem noch die normale Eigenschaft mountpoint besitzt, die vererbt werden kann. Sie können diese Eigenschaft beispielsweise auf „off" setzen und Eigenschaften setzen, die an untergeordnete Dateisysteme vererbt werden; das Dateisystem selbst wird jedoch nicht eingehängt und ist für Benutzer auch nicht zugänglich. In diesem Fall dient das übergeordnete Dateisystem, dessen Eigenschaft „canmount" auf „off" gesetzt ist, als *container*, sodass Sie für diesen Container Attribute einstellen können. Der Container selbst ist jedoch nicht zugänglich.

Im folgenden Beispiel wird das Dateisystem userpool erstellt und dessen Eigenschaft canmount auf „off" gesetzt. Einhängepunkte für untergeordnete Benutzerdateisysteme werden auf ein einziges Verzeichnis (/export/home) gesetzt. Am übergeordneten Dateisystem gesetzte Eigenschaften werden von untergeordneten Dateisystemen geerbt; das übergeordnete Dateisystem selbst wird jedoch nicht eingehängt.

```
# zpool create userpool mirror c0t5d0 c1t6d0
# zfs set canmount=off userpool
# zfs set mountpoint=/export/home userpool
# zfs set compression=on userpool
# zfs create userpool/user1
# zfs create userpool/user2
# zfs list -r userpool
NAME             USED  AVAIL  REFER  MOUNTPOINT
userpool         140K  8.24G  24.5K  /export/home
userpool/user1  24.5K  8.24G  24.5K  /export/home/user1
userpool/user2  24.5K  8.24G  24.5K  /export/home/user2
```

Durch Setzen der Eigenschaft canmount auf noauto kann das Dataset nur ausdrücklich, nicht aber automatisch eingehängt werden. Über diese Einstellung wird von der Solaris-Upgrade-Software festgelegt, dass nur Datasets aus der aktiven Boot-Umgebung (BU) beim Booten eingehängt werden.

Die Eigenschaft recordsize

Legt eine empfohlene Blockgröße für Dateien im Dateisystem fest.

Diese Eigenschaft dient zur Zusammenarbeit mit Datenbanken, die auf Dateien in festen Blockgrößen zugreifen. ZFS passt Blockgrößen automatisch an interne Algorithmen an, die für typische Zugriffsstrukturen optimiert wurden. Für Datenbanken, die sehr große Dateien erstellen, Dateien jedoch mit wahlfreiem Zugriff in kleineren Blöcken lesen, können diese

Algorithmen unter Umständen suboptimal sein. Wenn Sie die Eigenschaft `recordsize` auf einen Wert setzen, der der Datensatzgröße der betreffenden Datenbank entspricht bzw. größer als diese ist, kann dies erhebliche Leistungsverbesserungen zur Folge haben. Die Verwendung dieser Eigenschaft für allgemeine Dateisysteme wird nicht empfohlen und kann sich negativ auf die Leistung auswirken. Die angegebene Größe muss ein Zweierpotenzwert im Bereich von größer/gleich 512 Byte und kleiner/gleich 128 KB sein. Das Ändern der Eigenschaft `recordsize` eines Dateisystems wirkt sich nur auf Dateien aus, die nach der Änderung erstellt wurden. Bereits vorhandene Dateien bleiben unverändert.

Diese Eigenschaft kann auch mithilfe ihrer Kurzbezeichnung (`recsize`) referenziert werden.

Die Eigenschaft `volsize`

Die logische Kapazität eines Volumes. Standardmäßig wird beim Erstellen eines Volumes Speicherplatz der gleichen Kapazität reserviert. Alle Änderungen der Eigenschaft `volsize` werden entsprechend in der Reservierung geändert. Diese Überprüfungen dienen zum Verhindern unerwarteten Systemverhaltens. Je nachdem, wie ein Volume benutzt wird, kann es undefiniertes Systemverhalten oder Datenbeschädigung verursachen, wenn es weniger Speicherplatz enthält, als es eigentlich angefordert hat. Solche Effekte können auch auftreten, wenn die Kapazität eines Volumes geändert wird, während es in Benutzung ist. Dies gilt besonders dann, wenn die Volume-Kapazität verkleinert wird. Gehen Sie bei der Änderung einer Volume-Kapazität stets mit äußerster Sorgfalt vor.

Obwohl dies nicht empfohlen wird, können Sie ein Sparse-Volume erstellen, indem Sie für den Befehl `zfs create -V` das Flag `-s` angeben oder die Reservierung nach der Erstellung des Volumes entsprechend ändern. Bei *Sparse-Volumes* ist der Wert der Reservierung ungleich der Volume-Kapazität. Änderungen der Eigenschaft `volsize` wirken sich bei Sparse-Volumes nicht auf den Reservierungswert aus.

Weitere Informationen zur Verwendung von Volumes finden Sie unter „ZFS-Volumes“ auf Seite 279.

Benutzerdefinierte ZFS-Eigenschaften

Zusätzlich zu den nativen Standardeigenschaften unterstützt ZFS auch beliebige benutzerdefinierte Eigenschaften. Benutzerdefinierte Eigenschaften wirken sich nicht auf das ZFS-Verhalten aus, können jedoch zum Versehen von Datasets mit Informationen, die für Ihre lokalen Gegebenheiten wichtig sind, verwendet werden.

Namen benutzerdefinierter Eigenschaften müssen den folgenden Konventionen genügen:

- Sie müssen einen Doppelpunkt (': ') enthalten, damit sie von nativen Eigenschaften unterschieden werden können.
- Sie dürfen Kleinbuchstaben, Zahlen und die folgenden Interpunktionszeichen enthalten: ':', '+', '.', '_'.

- Der Name einer benutzerdefinierten Eigenschaft darf maximal 256 Zeichen lang sein.

Namen benutzerdefinierter Eigenschaften sollten generell in die folgenden beiden Komponenten aufgeteilt werden, obwohl dies für ZFS nicht obligatorisch ist:

module:property

Bei der programmatischen Verwendung von Eigenschaften sollten Sie für die Komponente *Modul* einen umgekehrten DNS-Domänennamen verwenden. Dies verringert die Wahrscheinlichkeit, dass zwei unabhängig voneinander entwickelte Pakete die gleiche Eigenschaft für unterschiedliche Zwecke nutzen. Eigenschaftsnamen, die mit „com.sun.“ beginnen, sind für Sun Microsystems reserviert.

Die Werte benutzerdefinierter Eigenschaften besitzen die folgenden Merkmale:

- Sie sind beliebige Zeichenketten, die immer vererbt und niemals validiert werden.
- Der Wert einer benutzerdefinierten Eigenschaft darf maximal 1024 Zeichen lang sein.

Beispiel:

```
# zfs set dept:users=finance userpool/user1
# zfs set dept:users=general userpool/user2
# zfs set dept:users=itops userpool/user3
```

Alle Befehle, die Eigenschaften verwenden (z. B. `zfs list`, `zfs get`, `zfs set` usw.) können native und benutzerdefinierte Eigenschaften nutzen.

Beispiel:

```
zfs get -r dept:users userpool
NAME            PROPERTY    VALUE           SOURCE
userpool        dept:users  all             local
userpool/user1  dept:users  finance         local
userpool/user2  dept:users  general         local
userpool/user3  dept:users  itops           local
```

Benutzerdefinierte Eigenschaften können mit dem Befehl `zfs inherit` gelöscht werden. Beispiel:

```
# zfs inherit -r dept:users userpool
```

Wenn die betreffende Eigenschaft nicht in einem übergeordneten Dataset definiert wurde, wird sie komplett entfernt.

Abfragen von ZFS-Dateisysteminformationen

Der Befehl `zfs list` bietet einen umfassenden Mechanismus zum Anzeigen und Abfragen von Dataset-Informationen. In diesem Abschnitt werden grundlegende und komplexere Abfragen erläutert.

Auflisten grundlegender ZFS-Informationen

Mit dem Befehl `zfs list` ohne Optionen können Sie sich grundlegende Dataset-Informationen anzeigen lassen. Dieser Befehl zeigt die Namen aller Datasets im System einschließl. der Eigenschaften `used`, `available`, `referenced` und `mountpoint` an. Weitere Informationen zu diesen Eigenschaften finden Sie unter „ZFS-Eigenschaften“ auf Seite 179.

Beispiel:

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
pool                   476K  16.5G    21K  /pool
pool/clone              18K  16.5G    18K  /pool/clone
pool/home              296K  16.5G    19K  /pool/home
pool/home/marks        277K  16.5G   277K  /pool/home/marks
pool/home/marks@snap      0      -   277K  -
pool/test               18K  16.5G    18K  /test
```

Mithilfe dieses Befehls können Sie auch Informationen zu bestimmten Datasets anzeigen. Geben Sie dazu in der Befehlszeile den Namen des gewünschten Datasets an. Darüber hinaus können Sie mit der Option `-r` rekursiv Informationen zu allen untergeordneten Datasets anzeigen. Beispiel:

```
# zfs list -r pool/home/marks
NAME                   USED  AVAIL  REFER  MOUNTPOINT
pool/home/marks        277K  16.5G   277K  /pool/home/marks
pool/home/marks@snap      0      -   277K  -
```

Geben Sie bei der Verwendung des Befehls `zfs list` für Datasets, Snapshots und Volumes absolute Pfadnamen an. Beispiel:

```
# zfs list /pool/home/marks
NAME              USED  AVAIL  REFER  MOUNTPOINT
pool/home/marks   277K  16.5G   277K  /pool/home/marks
```

Das folgende Beispiel zeigt, wie Informationen zum Dateisystem `tank/home/chua` und allen seinen untergeordneten Datasets angezeigt werden können.

```
# zfs list -r tank/home/chua
NAME                        USED  AVAIL  REFER  MOUNTPOINT
tank/home/chua                 26.0K  4.81G  10.0K  /tank/home/chua
```

```
tank/home/chua/projects       16K  4.81G   9.0K  /tank/home/chua/projects
tank/home/chua/projects/fs1    8K  4.81G     8K  /tank/home/chua/projects/fs1
tank/home/chua/projects/fs2    8K  4.81G     8K  /tank/home/chua/projects/fs2
```

Zusätzliche Informationen zum Befehl `zfs list` finden Sie in der Man Page `zfs(1M)`.

Erstellen komplexer ZFS-Abfragen

Die Ausgabe des Befehls `zfs list` kann mithilfe der Optionen `-o`, `-f` und `-H` speziell angepasst werden.

Mit der Option `-o` und einer kommagetrennten Liste gewünschter Eigenschaften können Sie die Ausgabe von Eigenschaftswerten anpassen. Geben Sie beliebige Dataset-Eigenschaften im gültigen Format an. Eine Liste aller unterstützten Dataset-Eigenschaften finden Sie unter „ZFS-Eigenschaften“ auf Seite 179. Zusätzlich zu den hier definierten Eigenschaften kann die Option `-o` auch das Literal `name` enthalten. In diesem Fall enthält die Befehlsausgabe auch den Namen des Datasets.

Im folgenden Beispiel wird mithilfe von `zfs list` der Dataset-Name zusammen mit den Eigenschaften `sharenfs` und `mountpoint` angezeigt.

```
# zfs list -o name,sharenfs,mountpoint
NAME                  SHARENFS         MOUNTPOINT
tank                  off              /tank
tank/home             on               /tank/home
tank/home/ahrens      on               /tank/home/ahrens
tank/home/bonwick     on               /tank/home/bonwick
tank/home/chua        on               /tank/home/chua
tank/home/eschrock    on               legacy
tank/home/moore       on               /tank/home/moore
tank/home/tabriz      ro               /tank/home/tabriz
```

Mit der Option `-t` können Sie festlegen, welche Dataset-Typen angezeigt werden sollen. Zulässige Typen sind in der folgenden Tabelle aufgeführt.

TABELLE 6–2 ZFS-Dataset-Typen

Typ	Beschreibung
`filesystem`	Dateisysteme und Klone
`volume`	Volumes
`snapshot`	Snapshots

Die Option `-t` liest eine kommagetrennte Liste der anzuzeigenden Dataset-Typen. Im folgenden Beispiel wird mithilfe der Optionen `-t` und `-o` für alle Dateisysteme gleichzeitig der Name und die Eigenschaft `used` angezeigt:

```
# zfs list -t filesystem -o name,used
NAME              USED
pool              476K
pool/clone         18K
pool/home         296K
pool/home/marks   277K
pool/test          18K
```

Mithilfe der Option `-H` kann bei der Ausgabe des Befehls `zfs list` die Titelzeile unterdrückt werden. Bei Verwendung der Option `-H` werden Leerzeichen als Tabulatorzeichen ausgegeben. Diese Option ist bei der Verwendung der Befehlsausgabe für programmatische Anwendungen (z. B. Skripten) nützlich. Das folgende Beispiel zeigt die Ausgabe des Befehls `zfs list` mit der Option `-H`.

```
# zfs list -H -o name
pool
pool/clone
pool/home
pool/home/marks
pool/home/marks@snap
pool/test
```

Verwalten von ZFS-Eigenschaften

Dataset-Eigenschaften werden mithilfe der Unterbefehle `set`, `inherit` und `get` des Befehls `zfs` verwaltet.

- „Setzen von ZFS-Eigenschaften“ auf Seite 196
- „Vererben von ZFS-Eigenschaften“ auf Seite 197
- „Abfragen von ZFS-Eigenschaften“ auf Seite 198

Setzen von ZFS-Eigenschaften

Sie können konfigurierbare Dataset-Eigenschaften mit dem Befehl `zfs set` setzen. Bei der Erstellung eines Datasets können Eigenschaften auch mit dem Befehl `zfs create` gesetzt werden. Eine Liste der konfigurierbaren Dataset-Eigenschaften finden Sie unter „Konfigurierbare native ZFS-Eigenschaften“ auf Seite 189. Der Befehl `zfs set` liest ein Eigenschaft-Wert-Paar im Format *Eigenschaft=Wert* sowie einen Dataset-Namen ein.

Im folgenden Beispiel wird die Eigenschaft `atime` von `tank/home` auf `off` gesetzt. Bei jedem Aufruf von `zfs set` kann nur eine Eigenschaft gesetzt bzw. geändert werden.

```
# zfs set atime=off tank/home
```

Darüber hinaus können bei der Erstellung eines Dateisystems beliebige Dateisystemeigenschaften gesetzt werden. Beispiel:

```
# zfs create -o atime=off tank/home
```

Nummerische Eigenschaften (Speicherkapazitätswerte) können durch Verwendung der folgenden verständlichen Suffixe (angegeben in Rangfolge) angegeben werden: `BKMGTPEZ`. Allen diesen Suffixen außer dem Suffix `B`, das für Byte steht, kann ein `b` (für „Byte") nachgestellt werden. In den folgenden vier Beispielen des Befehls `zfs set` werden entsprechende nummerische Ausdrücke angegeben, mit denen die Eigenschaft `quota` gesetzt wird. Damit werden Kontingente im Dateisystem `tank/home/marks` auf 50 GB gesetzt:

```
# zfs set quota=50G tank/home/marks
# zfs set quota=50g tank/home/marks
# zfs set quota=50GB tank/home/marks
# zfs set quota=50gb tank/home/marks
```

Bei Zeichenkettenwerten wird Groß- und Kleinschreibung unterschieden; diese Werte dürfen nur Kleinbuchstaben enthalten. Ausnahmen bilden die Werte der Eigenschaften `mountpoint` und `sharenfs`; die Werte dieser Eigenschaften dürfen sowohl Groß- als auch Kleinbuchstaben enthalten.

Weitere Informationen zum Befehl `zfs set` finden Sie in der Man Page `zfs(1M)`.

Vererben von ZFS-Eigenschaften

Alle konfigurierbaren Eigenschaften mit der Ausnahme von Kontingenten und Reservierungen erben ihren Wert von ihrem übergeordneten Dateisystem, es sei denn, diese Werte sind in den untergeordneten Dateisystemen explizit gesetzt. Wenn das entsprechende übergeordnete Dateisystem für eine vererbte Eigenschaft keinen Wert besitzt, wird der Standardwert für die betreffende Eigenschaft verwendet. Mit dem Befehl `zfs inherit` können Sie Eigenschaftswerte zurücksetzen, was zur Folge hat, dass der vom übergeordneten Dateisystem vererbte Wert verwendet wird.

Im folgenden Beispiel wird mithilfe des Befehls `zfs set` die Komprimierung für das Dateisystem `tank/home/bonwick` aktiviert. Danach wird mit `zfs inherit` der Wert der Eigenschaft `compression` zurückgesetzt, was zur Folge hat, dass der vom übergeordneten Dateisystem vererbte Wert (`off`) verwendet wird. Da weder bei `home` noch bei `tank` der Wert der Eigenschaft `compression` lokal gesetzt wurde, wird der Standardwert verwendet. Wenn bei beiden die Komprimierung aktiviert wäre, würde der Wert des direkten übergeordneten Dateisystems (in diesem Beispiel `home`) verwendet werden.

```
# zfs set compression=on tank/home/bonwick
# zfs get -r compression tank
NAME              PROPERTY      VALUE                  SOURCE
tank              compression   off                    default
tank/home         compression   off                    default
tank/home/bonwick compression   on                     local
```

```
# zfs inherit compression tank/home/bonwick
# zfs get -r compression tank
NAME              PROPERTY      VALUE                  SOURCE
tank              compression   off                    default
tank/home         compression   off                    default
tank/home/bonwick compression   off                    default
```

Der Unterbefehl `inherit` wird bei Angabe der Option `-r` rekursiv ausgeführt. Im folgenden Beispiel wird durch den Befehl der Wert für die Eigenschaft `compression` von `tank/home` und allen eventuell vorhandenen untergeordneten Dateisystemen geerbt.

```
# zfs inherit -r compression tank/home
```

Hinweis – Bitte beachten Sie, dass die Option `-r` die Eigenschaftswerte aller untergeordneten Datasets zurücksetzt.

Weitere Informationen zum Befehl `zfs` finden Sie in der Man Page `zfs(1M)`.

Abfragen von ZFS-Eigenschaften

Am Einfachsten können Eigenschaftswerte mit dem Befehl `zfs list` abgefragt werden. Weitere Informationen dazu finden Sie unter „Auflisten grundlegender ZFS-Informationen“ auf Seite 194. Für komplexere Abfragen und Skripten sollten Sie den Befehl `zfs get` verwenden, da dieser ausführlichere Informationen in einem anpassbaren Format anzeigt.

Sie können Dataset-Eigenschaften mit dem Befehl `zfs get` abrufen. Das folgende Beispiel zeigt, wie eine Eigenschaft eines Datasets abgerufen werden kann:

```
# zfs get checksum tank/ws
NAME              PROPERTY       VALUE                     SOURCE
tank/ws           checksum       on                        default
```

In der vierten Spalte `SOURCE` wird angezeigt, wo der betreffende Eigenschaftswert gesetzt wurde. In der folgenden Tabelle wird die Bedeutung der möglichen Ursprungswerte erläutert.

TABELLE 6–3 Mögliche `SOURCE`-Werte (`zfs get`)

SOURCE-Wert	Beschreibung
`default`	Diese Eigenschaft wurde für dieses Dataset bzw. seine übergeordneten Datasets nie explizit gesetzt. Es wird der Standardwert für diese Eigenschaft verwendet.

TABELLE 6–3 Mögliche SOURCE-Werte (`zfs get`) *(Fortsetzung)*

SOURCE-Wert	Beschreibung
`inherited from` *Dataset-Name*	Dieser Eigenschaftswert wurde vom übergeordneten Dataset *Dataset-Name* geerbt.
`local`	Dieser Eigenschaftswert wurde mithilfe von `zfs set` für dieses Dataset explizit gesetzt.
`temporary`	Dieser Eigenschaftswert wurde mithilfe von `zfs mount -o` gesetzt und gilt nur solange, wie das Dateisystem eingehängt ist. Weitere Informationen zu temporären Eigenschaften von Einhängepunkten finden Sie unter „Verwenden temporärer Einhängepunkte“ auf Seite 205.
- (keiner)	Diese Eigenschaft ist schreibgeschützt. Ihr Wert wird von ZFS bereitgestellt.

Sie können alle Dataset-Eigenschaften mit dem speziellen Schlüsselwort `all` abrufen. Im folgenden Beispiel werden alle Dataset-Eigenschaften mit dem Schlüsselwort `all` abgerufen:

```
# zfs get all pool
NAME  PROPERTY              VALUE                  SOURCE
pool  type                  filesystem             -
pool  creation              Thu Aug 27  9:33 2009  -
pool  used                  72K                    -
pool  available             66.9G                  -
pool  referenced            21K                    -
pool  compressratio         1.00x                  -
pool  mounted               yes                    -
pool  quota                 none                   default
pool  reservation           none                   default
pool  recordsize            128K                   default
pool  mountpoint            /pool                  default
pool  sharenfs              off                    default
pool  checksum              on                     default
pool  compression           on                     default
pool  atime                 on                     default
pool  devices               on                     default
pool  exec                  on                     default
pool  setuid                on                     default
pool  readonly              off                    default
pool  zoned                 off                    default
pool  snapdir               hidden                 default
pool  aclmode               groupmask              default
pool  aclinherit            restricted             default
pool  canmount              on                     default
pool  shareiscsi            off                    default
pool  xattr                 on                     default
```

```
pool  copies               1                    default
pool  version              4                    -
pool  utf8only             off                  -
pool  normalization        none                 -
pool  casesensitivity      sensitive            -
pool  vscan                off                  default
pool  nbmand               off                  default
pool  sharesmb             off                  default
pool  refquota             none                 default
pool  refreservation       none                 default
pool  primarycache         all                  default
pool  secondarycache       all                  default
pool  usedbysnapshots      0                    -
pool  usedbydataset        21K                  -
pool  usedbychildren       51K                  -
pool  usedbyrefreservation 0                    -
```

Hinweis – Die Eigenschaften `casesensitivity`, `nbmand`, `normalization`, `sharemgrutf8only` und `vscan` sind auf einen festen Wert gesetzt und werden unter Solaris 10 nicht unterstützt.

Mit der Option `-s` des Befehls `zfs get` können Sie die anzuzeigenden Eigenschaften nach Ursprungstyp angeben. Diese Option liest eine kommagetrennte Liste der gewünschten Ursprungstypen ein. Es werden nur Eigenschaften des gewünschten Ursprungstyps angezeigt. Zulässige Ursprungstypen sind `local`, `default`, `inherited`, `temporary` und `none`. Das folgende Beispiel zeigt alle Eigenschaften, die in `pool` lokal gesetzt wurden.

```
# zfs get -s local all pool
NAME             PROPERTY      VALUE                     SOURCE
pool             compression   on                        local
```

Alle der o. g. Optionen können zusammen mit der Option `-r` verwendet werden, um die angegebenen Eigenschaften aller untergeordneten Datasets rekursiv anzuzeigen. Im folgenden Beispiel werden alle temporären Eigenschaften aller Datasets in `tank` rekursiv angezeigt:

```
# zfs get -r -s temporary all tank
NAME             PROPERTY       VALUE                      SOURCE
tank/home          atime          off                      temporary
tank/home/bonwick  atime          off                      temporary
tank/home/marks    atime          off                      temporary
```

Mithilfe einer neuen Funktion können Sie mit `zfs get` Abfragen ohne Angabe eines Zieldateisystems durchführen, was bedeutet, dass alle Pools bzw. Dateisysteme abgefragt werden. Beispiel:

```
# zfs get -s local all
tank/home               atime          off                   local
```

```
tank/home/bonwick       atime          off                    local
tank/home/marks         quota          50G                    local
```

Weitere Informationen zum Befehl `zfs get` finden Sie in der Man Page `zfs(1M)`.

Abfragen von ZFS-Eigenschaften für Skripten

Der Befehl `zfs get` unterstützt die Optionen `-H` und `-o`, die speziell für die Verwendung dieses Befehl in Skripten vorgesehen sind. Mit der Option `-H` wird festgelegt, dass Titelzeilen unterdrückt und Leerzeichen als Tabulatorzeichen ausgegeben werden. Dadurch können Daten einfach analysiert werden. Sie können die Befehlsausgabe mit der Option `-o` speziell anpassen. Diese Option liest eine kommagetrennte Liste der auszugebenden Werte ein. Alle unter „ZFS-Eigenschaften“ auf Seite 179 definierten Eigenschaften sowie die Literale `name`, `value`, `property` und `source` können in der Liste für die Option `-o` angegeben werden.

Das folgende Beispiel zeigt, wie mithilfe der Optionen `-H` und `-o` des Befehls `zfs get` ein einzelner Wert abgerufen werden kann.

```
# zfs get -H -o value compression tank/home
on
```

Die Option `-p` gibt nummerische Werte exakt aus. 1 MB wird beispielsweise als 1000000 ausgegeben. Diese Option lässt sich wie folgt verwenden:

```
# zfs get -H -o value -p used tank/home
182983742
```

Mit der Option `-r` und allen der o. g. Optionen können Sie Werte für alle untergeordneten Datasets rekursiv abrufen. Im folgenden Beispiel wird mithilfe der Optionen `-r`, `-o` und `-H` für das Dateisystem `export/home` und alle seine untergeordneten Dateisysteme der Dataset-Name sowie der Wert der Eigenschaft `used` abgerufen (die Titelzeile der Befehlsausgabe wird unterdrückt):

```
# zfs get -H -o name,value -r used export/home
export/home     5.57G
export/home/marks       1.43G
export/home/maybee      2.15G
```

Einhängen und Freigeben von ZFS-Dateisystemen

In diesem Abschnitt wird die Verwaltung von Einhängepunkten und freigegebenen Dateisystemen in ZFS beschrieben.

- „Aushängen von ZFS-Dateisystemen“ auf Seite 206
- „Freigeben und Sperren von ZFS-Dateisystemen“ auf Seite 206

Verwalten von ZFS-Einhängepunkten

Standardmäßig werden alle ZFS-Dateisysteme beim Systemstart von ZFS mithilfe des SMF-Dienstes (Service Management Facility) `svc://system/filesystem/local` eingehängt. Dateisysteme werden unter */Pfad* eingehängt, wobei *Pfad* den Namen des Dateisystems bezeichnet.

Sie können den Standard-Einhängepunkt überschreiben, indem Sie die Eigenschaft `mountpoint` mithilfe des Befehls `zfs set` auf einen spezifischen Pfad setzen. ZFS erstellt diesen Einhängepunkt bei Bedarf automatisch und hängt dieses Dateisystem beim Aufrufen des Befehls `zfs mount -a` auch automatisch ein, ohne dass Sie dafür die Datei `/etc/vfstab` ändern müssen.

Die Eigenschaft `mountpoint` wird vererbt. Wenn die Eigenschaft `mountpoint` von `pool/home` beispielsweise auf `/export/stuff` gesetzt ist, erbt `pool/home/user` für seine Eigenschaft `mountpoint` den Wert `/export/stuff/user`.

Die Eigenschaft `mountpoint` kann auf `none` gesetzt werden, um ein Einhängen eines Dateisystems zu verhindern. Darüber hinaus ist die Eigenschaft `canmount` verfügbar, die angibt, ob ein Dateisystem eingehängt werden kann. Weitere Informationen zur Eigenschaft `canmount` finden Sie unter „Die Eigenschaft `canmount`“ auf Seite 191.

Bei Bedarf können Dateisysteme durch Setzen der Eigenschaft `mountpoint` auf `legacy` mithilfe des Befehls `zfs set` auch explizit durch ältere Legacy-Einhängesysteme verwaltet werden. Dies verhindert das automatische Einhängen und Verwalten des betreffenden Dateisystems durch ZFS. Stattdessen müssen Sie Legacy-Dienstprogramme wie die Befehle `mount` und `umount` und die Datei `/etc/vfstab` verwenden. Weitere Informationen zu Legacy-Einhängepunkten finden Sie unter „Legacy-Einhängepunkte“ auf Seite 203.

Beim Ändern von Verwaltungsstrategien für Einhängepunkte gibt es die folgenden beiden Verhalten:

- Verhalten bei automatischen Einhängepunkten
- Verhalten bei Legacy-Einhängepunkten

Automatische Einhängepunkte

- Bei Umschalten von `legacy` oder `none` hängt ZFS das betreffende Dateisystem automatisch ein.
- Wenn ZFS das Dateisystem gegenwärtig verwaltet, dieses aber ausgehängt ist und die Eigenschaft `mountpoint` geändert wird, bleibt das Dateisystem weiterhin ausgehängt.

Sie können den Standard-Einhängepunkt für das Root-Dataset bei dessen Erstellung auch mithilfe der Option `-m` von `zpool create` setzen. Weitere Informationen zum Erstellen von Pools finden Sie unter „Erstellen eines ZFS-Speicher-Pools“ auf Seite 66.

Datasets, deren Eigenschaft `mountpoint` nicht auf `legacy` gesetzt ist, werden von ZFS verwaltet. Im folgenden Beispiel wird ein Dataset erstellt, dessen Einhängepunkt automatisch von ZFS verwaltet wird.

```
# zfs create pool/filesystem
# zfs get mountpoint pool/filesystem
NAME              PROPERTY      VALUE                      SOURCE
pool/filesystem   mountpoint    /pool/filesystem           default
# zfs get mounted pool/filesystem
NAME              PROPERTY      VALUE                      SOURCE
pool/filesystem   mounted       yes                        -
```

Sie können die Eigenschaft `mountpoint` auch explizit setzen (siehe folgendes Beispiel):

```
# zfs set mountpoint=/mnt pool/filesystem
# zfs get mountpoint pool/filesystem
NAME              PROPERTY      VALUE                      SOURCE
pool/filesystem   mountpoint    /mnt                       local
# zfs get mounted pool/filesystem
NAME              PROPERTY      VALUE                      SOURCE
pool/filesystem   mounted       yes                        -
```

Beim Ändern der Eigenschaft `mountpoint` wird das betreffende Dateisystem automatisch aus dem alten Einhängepunkt ausgehängt und in den neuen Einhängepunkt eingehängt. Einhängepunktverzeichnisse werden je nach Bedarf erstellt. Wenn ZFS ein Dateisystem nicht aushängen kann, weil es noch aktiv ist, wird ein Fehler gemeldet, und das Aushängen muss manuell erzwungen werden.

Legacy-Einhängepunkte

Sie können ZFS-Dateisysteme mit Legacy-Dienstprogrammen verwalten, indem Sie die Eigenschaft `mountpoint` auf `legacy` setzen. Legacy-Dateisysteme müssen mithilfe der Befehle `mount` und `umount` sowie der Datei `/etc/vfstab` verwaltet werden. ZFS hängt Legacy-Dateisysteme beim Systemstart nicht automatisch ein, und die ZFS-Befehle `mount` und `umount` funktionieren mit Datasets dieses Typs nicht. Die folgenden Beispiele zeigen die Erstellung und Verwaltung eines ZFS-Datasets im Legacy-Modus:

```
# zfs set mountpoint=legacy tank/home/eschrock
# mount -F zfs tank/home/eschrock /mnt
```

Damit Legacy-Dateisysteme beim Systemstart automatisch eingehängt werden, müssen Sie zur Datei `/etc/vfstab` die entsprechenden Einträge hinzufügen. Das folgende Beispiel zeigt, wie der Eintrag in der Datei `/etc/vfstab` aussehen kann:

```
#device         device        mount           FS      fsck    mount   mount
#to mount       to fsck       point           type    pass    at boot options
#

tank/home/eschrock -        /mnt          zfs        -        yes        -
```

Bitte beachten Sie, dass die Einträge `device to fsck` und `fsck pass` auf `-` gesetzt sind, weil der Befehl `fsck` für ZFS-Dateisysteme nicht gilt. Weitere Informationen zu Datenintegrität und zum Nichtvorhandensein des Befehls `fsck` in ZFS finden Sie unter „Transaktionale Semantik“ auf Seite 41.

Einhängen von ZFS-Dateisystemen

ZFS hängt Dateisysteme beim Erstellen dieser Dateisysteme bzw. beim Systemstart automatisch ein. Der Befehl `zfs mount` muss nur verwendet werden, wenn Einhängeoptionen geändert oder Dateisystem explizit ein- oder ausgehängt werden.

Beim Aufrufen des Befehls `zfs mount` ohne Argumente werden alle von ZFS verwalteten und gegenwärtig eingehängten Dateisysteme angezeigt. Legacy-Einhängepunkte werden nicht angezeigt. Beispiel:

```
# zfs mount
tank                            /tank
tank/home                       /tank/home
tank/home/bonwick               /tank/home/bonwick
tank/ws                         /tank/ws
```

Mit der Option `-a` können Sie alle von ZFS verwalteten Dateisysteme einhängen. Legacy-Dateisysteme werden nicht eingehängt. Beispiel:

```
# zfs mount -a
```

Standardmäßig erlaubt ZFS das Einhängen in ein nicht leeres Verzeichnis nicht. Zum Erzwingen des Einhängens in ein nicht leeres Verzeichnis müssen Sie die Option `-O` verwenden. Beispiel:

```
# zfs mount tank/home/lalt
cannot mount '/export/home/lalt': directory is not empty
use legacy mountpoint to allow this behavior, or use the -O flag
# zfs mount -O tank/home/lalt
```

Legacy-Einhängepunkte müssen mit Legacy-Dienstprogrammen verwaltet werden. Wenn Sie versuchen, dafür ZFS-Befehle zu verwenden, wird ein Fehler ausgegeben. Beispiel:

```
# zfs mount pool/home/billm
cannot mount 'pool/home/billm': legacy mountpoint
```

```
use mount(1M) to mount this filesystem
# mount -F zfs tank/home/billm
```

Beim Einhängen eines Dateisystems werden verschiedene Einhängeoptionen verwendet, die auf den zum Dataset gehörenden Eigenschaftswerten beruhen. Zwischen Eigenschaften und Einhängeoptionen besteht der folgende Zusammenhang:

Eigenschaft	Einhängeoptionen
`devices`	`devices/nodevices`
`exec`	`exec/noexec`
`readonly`	`ro/rw`
`setuid`	`setuid/nosetuid`

Die Einhängeoption `nosuid` ist ein Alias-Name für `nodevices,nosetuid`.

Verwenden temporärer Einhängepunkte

Wenn o. g. Optionen explizit mithilfe der Option `-o` des Befehls `zfs mount` gesetzt werden, wird der zugehörige Eigenschaftswert temporär überschrieben. Diese Eigenschaftswerte werden vom Befehl `zfs get` mit dem Wert `temporary` gemeldet und beim Aushängen des betreffenden Dateisystems auf ihre ursprünglichen Werte zurückgesetzt. Beim Ändern eines Eigenschaftswerts während des Einhängens eines Datasets wird diese Änderung sofort übernommen und überschreibt eventuelle temporäre Einstellungen.

Im folgenden Beispiel wird am Dateisystem `tank/home/perrin` die Einhängeoption „schreibgeschützt" temporär gesetzt:

```
# zfs mount -o ro tank/home/perrin
```

In diesem Beispiel wird angenommen, dass das Dateisystem ausgehängt ist. Zum temporären Ändern einer Eigenschaft eines gegenwärtig eingehängten Dateisystems dient die spezielle Option `remount`. Im folgenden Beispiel wird die Eigenschaft `atime` für ein eingehängtes Dateisystem temporär auf `off` gesetzt.

```
# zfs mount -o remount,noatime tank/home/perrin
# zfs get atime tank/home/perrin
NAME             PROPERTY      VALUE                     SOURCE
tank/home/perrin atime         off                       temporary
```

Weitere Informationen zum Befehl `zfs mount` finden Sie in der Man Page `zfs(1M)`.

Aushängen von ZFS-Dateisystemen

Dateisysteme können mit dem Befehl `zfs unmount` ausgehängt werden. Der Befehl `unmount` akzeptiert als Argument entweder den Einhängepunkt oder den Dateisystemnamen.

Im folgenden Beispiel wird ein Dateisystem nach seinem Namen ausgehängt:

```
# zfs unmount tank/home/tabriz
```

Im folgenden Beispiel wird ein Dateisystem nach Einhängepunkt ausgehängt:

```
# zfs unmount /export/home/tabriz
```

Der Befehl `unmount` schlägt fehl, wenn das betreffende Dateisystem aktiv oder belegt ist. Ein Aushängen eines Dateisystems kann mit der Option `-f` erzwungen werden. Gehen Sie beim erzwungenen Aushängen eines Dateisystems äußerst sorgsam vor, wenn der Inhalt dieses Dateisystems noch verwendet wird, da unvorhergesehenes Anwendungsverhalten die Folge davon sein kann.

```
# zfs unmount tank/home/eschrock
cannot unmount '/export/home/eschrock': Device busy
# zfs unmount -f tank/home/eschrock
```

Zum Zweck der Abwärtskompatibilität kann der Legacy-Befehl `umount` auch zum Aushängen von ZFS-Dateisystemen verwendet werden. Beispiel:

```
# umount /export/home/bob
```

Weitere Informationen zum Befehl `zfs unmount` finden Sie in der Man Page `zfs(1M)`.

Freigeben und Sperren von ZFS-Dateisystemen

Wie bei Einhängepunkten kann ZFS Dateisysteme durch entsprechendes Setzen der Eigenschaft `sharenfs` automatisch für den Netzwerkzugriff freigeben. Mit dieser Methode müssen Sie die Datei `/etc/dfs/dfstab` nicht ändern, wenn ein neues Dateisystem hinzugefügt wurde. Die Eigenschaft `sharenfs` ist eine kommagetrennte Liste mit Optionen, die an den Befehl `share` übergeben wird. Der spezielle Wert `on` ist ein Aliasname für die Standardoptionen der Netzwerkfreigabe (`read/write`, d.h. Lese- und Schreibberechtigung für alle Benutzer). Der spezielle Wert `off` gibt an, dass das Dateisystem nicht von ZFS verwaltet wird und über herkömmliche Mittel wie z. B. die Datei `/etc/dfs/dfstab` für den Netzwerkzugriff freigegeben werden kann. Alle Dateisysteme, deren Eigenschaft `sharenfs` nicht auf `off` gesetzt ist, werden beim Systemstart freigegeben.

Einstellen der Freigabesemantik

Standardmäßig sind alle Dateisysteme für den Netzwerkzugriff gesperrt. Zum Freigeben eines neuen Dateisystems für den Netzwerkzugriff müssen Sie den Befehl `zfs set` mit der folgenden Syntax verwenden:

```
# zfs set sharenfs=on tank/home/eschrock
```

Die Eigenschaft wird vererbt, und alle Dateisysteme werden bei der Erstellung automatisch für den Netzwerkzugriff freigegeben, wenn deren Eigenschaft „inherited" nicht auf `off` gesetzt ist. Beispiel:

```
# zfs set sharenfs=on tank/home
# zfs create tank/home/bricker
# zfs create tank/home/tabriz
# zfs set sharenfs=ro tank/home/tabriz
```

Die Dateisysteme `tank/home/bricker` und `tank/home/tabriz` sind anfänglich mit Schreibzugriff freigegeben, da sie die Eigenschaft `sharenfs` von `tank/home` erben. Nach dem Setzen dieser Eigenschaft auf `ro` (schreibgeschützt) wird das Dateisystem `tank/home/tabriz` unabhängig davon, welchen Wert die Eigenschaft `sharenfs` für `tank/home` hat, schreibgeschützt freigegeben.

Sperren von ZFS-Dateisystemen für den Netzwerkzugriff

Obwohl die meisten Dateisysteme beim Systemstart, der Erstellung und beim Löschen automatisch für den Netzwerkzugriff freigegeben bzw. gesperrt werden, kann es manchmal vorkommen, dass Dateisysteme explizit für den Netzwerkzugriff gesperrt werden müssen. Dazu dient der Befehl `zfs unshare`. Beispiel:

```
# zfs unshare tank/home/tabriz
```

Dieser Befehl sperrt das Dateisystem `tank/home/tabriz` für den Netzwerkzugriff. Zum Sperren aller ZFS-Dateisysteme auf einem System benötigen Sie die Option `-a`.

```
# zfs unshare -a
```

Freigeben von ZFS-Dateisystemen für den Netzwerkzugriff

In den meisten Fällen reicht das automatische ZFS-Verhalten des Freigebens von Dateisystemen beim Systemstart und Erstellen eines Dateisystems für den Normalbetrieb aus. Falls Dateisysteme doch einmal für den Netzwerkzugriff gesperrt werden müssen, können Sie diese mit dem Befehl `zfs share` wieder freigeben. Beispiel:

```
# zfs share tank/home/tabriz
```

Sie können auch alle ZFS-Dateisysteme auf einem System mithilfe der Option `-a` freigeben.

```
# zfs share -a
```

Freigabeverhalten bei Legacy-Dateisystemen

Wenn die Eigenschaft `sharenfs` auf `off` gesetzt ist, versucht ZFS niemals, das betreffende Dateisystem für den Netzwerkzugriff freizugeben bzw. zu sperren. Diese Einstellung ermöglicht die Freigabeverwaltung mithilfe herkömmlicher Mittel wie z. B. der Datei `/etc/dfs/dfstab`.

Im Gegensatz zum traditionellen Befehl `mount` funktionieren die traditionellen Befehle `share` und `unshare` auch für ZFS-Dateisysteme. Deswegen können Sie ein Dateisystem manuell mit Optionen freigeben, die sich von den Einstellungen der Eigenschaft `sharenfs` unterscheiden. Davon wird jedoch abgeraten. Sie sollten NFS-Freigaben entweder vollständig von ZFS oder vollständig mithilfe der Datei `/etc/dfs/dfstab` verwalten lassen. Das administrative ZFS-Modell ist einfacher und nicht so aufwändig wie das traditionelle Modell. In einigen Fällen kann es jedoch vorkommen, dass Sie das Freigabeverhalten von Dateisystemen mithilfe des traditionellen Modells kontrollieren wollen.

Einstellen von ZFS-Kontingenten und -Reservierungen

Mit der Eigenschaft `quota` können Sie den Speicherplatz, den ein Dateisystem verwenden kann, beschränken. Darüber hinaus können Sie mit der Eigenschaft `reservation` für ein Dateisystem verfügbaren Speicherplatz garantieren. Beide Eigenschaften gelten für das Dataset, an dem sie gesetzt wurden, und für alle seine untergeordneten Datasets.

Wenn beispielsweise für das Dataset `tank/home` ein Kontingent festgelegt wurde, heißt das, dass der insgesamt von `tank/home` *und allen seinen untergeordneten Datasets* belegte Speicherplatz dieses Kontingent nicht überschreiten kann. Genauso wird beim Festlegen einer Reservierung für `tank/home` dieses Dataset *und allen seinen untergeordneten Datasets* dieser Speicherplatz garantiert. Der von einem Dataset und allen seinen untergeordneten Datasets belegte Speicherplatz wird von der Eigenschaft `used` verfolgt.

Die Eigenschaften `refquota` und `refreservation` stehen zur Verwaltung von Speicherplatz für Dateisysteme zur Verfügung, ohne dabei den Speicherplatz zu berücksichtigen, der von untergeordneten Elementen wie Snapshots und Klonen beansprucht wird.

In diesem Solaris-Release können Sie ein *user*- oder *group*-Kontingent für den Speicherplatz, der von zu einem bestimmten Benutzer oder einer bestimmten Gruppe gehörigen Dateien beansprucht wird. Die Kontingenteigenschaften des Benutzers oder der Gruppe können nicht auf einem Volume, einem Dateisystem vor Dateisystem-Version 4 oder einem Pool vor Pool-Version 15 eingerichtet werden.

Beachten Sie die folgenden Faktoren, um festzustellen, welche Kontingent- und Reservierungsfunktionen sich am besten für die Verwaltung Ihrer Dateisysteme anbieten:

- Die Eigenschaften `quota` und `reservation` eignen sich für die Verwaltung des durch Datasets belegten Speicherplatzes.
- Die Eigenschaften `refquota` und `refreservation` eignen sich für die Verwaltung des durch Datasets und Snapshots belegten Speicherplatzes.
- Die Einstellung von `refquota` oder `refreservation` auf einen Wert, der die Größe des Kontingents oder der Reservierung übersteigt, hat keine Wirkung. Wenn Sie die Eigenschaften `quota` oder `refquota` setzen, schlagen Vorgänge, bei denen einer dieser Werte überschritten wird, fehl. Es ist möglich, ein Kontingent (`quota`) zu überschreiten, das größer ist als `refquota`. Bei unreinen Snapshot-Blöcken sollte tatsächlich eher `quota` als `refquota` überschritten werden.
- Benutzer- und Gruppenkontingente vereinfachen die Verwaltung des Festplattenspeichers bei Vorhandensein vieler Benutzerkonten, zum Beispiel in einer universitären Umgebung.

Weitere Informationen zum Einrichten von Kontingenten und Reservierungen finden Sie unter „Setzen von Kontingenten für ZFS-Dateisysteme“ auf Seite 209und „Setzen von Reservierungen für ZFS-Dateisysteme“ auf Seite 213.

Setzen von Kontingenten für ZFS-Dateisysteme

ZFS-Kontingente können mit den Befehlen `zfs set` und `zfs get` festgelegt und angezeigt werden. Im folgenden Beispiel wird für das Dateisystem `tank/home/bonwick` ein Kontingent von 10 GB gesetzt.

```
# zfs set quota=10G tank/home/bonwick
# zfs get quota tank/home/bonwick
NAME              PROPERTY      VALUE                     SOURCE
tank/home/bonwick quota         10.0G                     local
```

ZFS-Kontingente wirken sich auch auf die Ausgabe der Befehle `zfs list` und `df` aus. Beispiel:

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank/home             16.5K  33.5G  8.50K  /export/home
tank/home/bonwick     15.0K  10.0G  8.50K  /export/home/bonwick
tank/home/bonwick/ws  6.50K  10.0G  8.50K  /export/home/bonwick/ws
# df -h /export/home/bonwick
Filesystem             size   used  avail capacity  Mounted on
tank/home/bonwick       10G     8K    10G     1%    /export/home/bonwick
```

Bitte beachten Sie, dass, obwohl für `tank/home` 33,5 GB Speicherplatz verfügbar sind, wegen des für `tank/home/bonwick` gesetzten Kontingents für `tank/home/bonwick` und `tank/home/bonwick/ws` nur 10 GB verfügbar sind.

Kontingente können nicht auf Werte gesetzt werden, die kleiner als der gegenwärtig vom betreffenden Dataset belegte Speicherplatz sind. Beispiel:

```
# zfs set quota=10K tank/home/bonwick
cannot set quota for 'tank/home/bonwick': size is less than current used or
reserved space
```

Es ist möglich, `refquota` für ein Dataset zu setzen, um den durch das Dataset belegbaren Speicherplatz einzuschränken. Dieser absolute Grenzwert berücksichtigt keinen Speicherplatz, der von Snapshots und Klonen belegt wird. Beispiel:

```
# zfs set refquota=10g students/studentA
# zfs list
NAME                 USED  AVAIL  REFER  MOUNTPOINT
profs                106K  33.2G    18K  /profs
students            57.7M  33.2G    19K  /students
students/studentA   57.5M  9.94G  57.5M  /students/studentA
# zfs snapshot students/studentA@today
# zfs list
NAME                       USED  AVAIL  REFER  MOUNTPOINT
profs                      106K  33.2G    18K  /profs
students                  57.7M  33.2G    19K  /students
students/studentA         57.5M  9.94G  57.5M  /students/studentA
students/studentA@today       0      -  57.5M  -
```

Sie können ein weiteres Kontingent für ein Dataset festlegen, um die Verwaltung des durch Snapshots belegten Speicherplatzes zu erleichtern. Beispiel:

```
# zfs set quota=20g students/studentA
# zfs list
NAME                       USED  AVAIL  REFER  MOUNTPOINT
profs                      106K  33.2G    18K  /profs
students                  57.7M  33.2G    19K  /students
students/studentA         57.5M  9.94G  57.5M  /students/studentA
students/studentA@today       0      -  57.5M  -
```

In diesem Szenario kann `studentA` den mit „refquota“ festgelegten absoluten Grenzwert (10 GB) erreichen und selbst bei vorhandenen Snapshots wiederherzustellende Dateien entfernen.

Im obigen Beispiel wird das kleinere der beiden Kontingente (10 GB im Gegensatz zu 20 GB) in der Ausgabe von `zfs list` angezeigt. Zum Anzeigen der Werte beider Kontingente verwenden Sie den Befehl `zfs get`. Beispiel:

```
# zfs get refquota,quota students/studentA
NAME               PROPERTY  VALUE              SOURCE
students/studentA  refquota  10G                local
students/studentA  quota     20G                local
```

Einrichten von Benutzer- oder Gruppenkontingenten auf einem ZFS-Dateisystem

Unter Verwendung der Befehle `zfs userquota` und `zfs groupquota` können Sie ein Benutzer- oder Gruppenkontingent wie folgt einrichten:

```
# zfs create students/compsci
# zfs set userquota@student1=10G students/compsci
# zfs create students/labstaff
# zfs set groupquota@staff=20GB students/labstaff
```

Zeigen Sie das aktuelle Benutzer- oder Gruppenkontingent wie folgt an:

```
# zfs get userquota@student1 students/compsci
NAME              PROPERTY            VALUE               SOURCE
students/compsci  userquota@student1  10G                 local
# zfs get groupquota@staff students/labstaff
NAME               PROPERTY          VALUE             SOURCE
students/labstaff  groupquota@staff  20G               local
```

Sie können die Speicherplatzbelegung durch allgemeine Benutzer und Gruppen durch Abfrage der folgenden Eigenschaften anzeigen:

```
# zfs userspace students/compsci
TYPE        NAME      USED  QUOTA
POSIX User  root      227M   none
POSIX User  student1  455M    10G
# zfs groupspace students/labstaff
TYPE         NAME   USED  QUOTA
POSIX Group  root   217M   none
POSIX Group  staff  217M    20G
```

Wenn Sie die Speicherplatzbelegung einzelner Benutzer oder Gruppen herausfinden möchten, fragen Sie die folgenden Eigenschaften ab:

```
# zfs get userused@student1 students/compsci
NAME              PROPERTY           VALUE              SOURCE
students/compsci  userused@student1  455M               local
# zfs get groupused@staff students/labstaff
NAME               PROPERTY         VALUE            SOURCE
students/labstaff  groupused@staff  217M             local
```

Die Eigenschaften der Benutzer- und Gruppenkontingente werden nicht über den Befehl `zfs get all` *dataset* angezeigt. Dieser führt die Eigenschaften aller Dateisysteme auf.

Sie können einen Benutzer oder eine Gruppe folgendermaßen entfernen:

```
# zfs set userquota@user1=none students/compsci
# zfs set groupquota@staff=none students/labstaff
```

ZFS-Benutzer- und Gruppenkontingente beinhalten folgende Funktionen:

- Bei Einrichtung auf einem Dateisystem wird ein auf einem übergeordneten Dateisystem eingerichtetes Benutzer- oder Gruppenkontingent nicht automatisch von einem untergeordneten Dateisystem übernommen.
- Allerdings wird das Benutzer- oder Gruppenkontingent angewendet, wenn ein Klon oder Snapshot aus einem Dateisystem mit einem Benutzer- oder Gruppenkontingent erstellt wird. Ebenso wird ein Benutzer- oder Gruppenkontingent im Dateisystem beinhaltet, wenn mit dem Befehl `zfs send` ein Stream erstellt wird, auch ohne Anwendung der Option `-R`.
- Benutzer ohne Privilegien haben nur Zugriff auf ihre eigene Speicherplatzbelegung. Der Root-Benutzer und ein Benutzer, dem das Privileg `userused` oder `groupused` erteilt wurde, hat Zugriff auf Informationen über die Speicherplatzverwaltung sämtlicher Benutzer und Gruppen.
- Die Eigenschaften `userquota` und `groupquota` können nicht auf ZFS-Volumes, auf einem Dateisystem vor Dateisystem-Version 4 oder auf einem Pool vor Pool-Version 15 eingerichtet werden.

Das Inkrafttreten von Benutzer- oder Gruppenkontingenten kann einige Sekunden dauern. Die Verzögerung kann bedeuten, dass Benutzer ihr Kontingent überschreiten, bevor das System dies registriert und weitere Schreibvorgänge mit der Fehlermeldung `EDQUOT` zurückweist.

Mit dem Legacy-Befehl `quota` können Sie Benutzerkontingente in einer NFS-Umgebung überprüfen, zum Beispiel beim Einhängen eines ZFS-Dateisystems. Ohne Auswahl von Optionen zeigt der Befehl `quota` nur Ergebnisse, wenn das Benutzerkontingent überschritten wird. Beispiel:

```
# zfs set userquota@student1=10m students/compsci
# zfs userspace students/compsci
TYPE        NAME      USED  QUOTA
POSIX User  root      227M   none
POSIX User  student1  455M    10M
# quota student1
Block limit reached on /students/compsci
```

Wenn Sie das Kontingent zurücksetzen und die Begrenzung des Kontingents nicht länger überschritten ist, müssen Sie zur Überprüfung des Benutzerkontingents den Befehl `quota -v` verwenden. Beispiel:

```
# zfs set userquota@student1=10GB students/compsci
# zfs userspace students/compsci
TYPE        NAME      USED  QUOTA
POSIX User  root      227M   none
POSIX User  student1  455M    10G
# quota student1
# quota -v student1
Disk quotas for student1 (uid 201):
Filesystem     usage  quota  limit    timeleft  files  quota  limit    timeleft
/students/compsci
              466029 10485760 10485760
```

Setzen von Reservierungen für ZFS-Dateisysteme

Unter einer ZFS-*Reservierung* versteht man eine Zuweisung von Speicherplatz aus dem Pool, die einem Dataset garantiert ist. Sie können keine Reservierungen machen, wenn der angeforderte Speicherplatz im Pool nicht zur Verfügung steht. Der Gesamtbetrag aller noch ausstehenden und nicht belegten Reservierungen darf die Gesamtsumme des nicht belegten Speicherplatzes im Pool nicht überschreiten. ZFS-Reservierungen können mit den Befehlen `zfs set` und `zfs get` festgelegt und angezeigt werden. Beispiel:

```
# zfs set reservation=5G tank/home/moore
# zfs get reservation tank/home/moore
NAME             PROPERTY      VALUE                      SOURCE
tank/home/moore  reservation   5.00G                      local
```

ZFS-Reservierungen wirken sich auch auf die Ausgabe des Befehls `zfs list` aus. Beispiel:

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank/home             5.00G  33.5G  8.50K  /export/home
tank/home/moore       15.0K  10.0G  8.50K  /export/home/moore
```

Bitte beachten Sie, dass `tank/home` 5 GB Speicherplatz belegt, obwohl der Gesamtbetrag des tatsächlich von `tank/home` und seinen untergeordneten Dateisystemen belegten Speicherplatzes viel weniger als 5 GB ist. Der belegte Speicherplatz berücksichtigt die Reservierung für `tank/home/moore`. Reservierungen werden in den belegten Speicherplatz des übergeordneten Datasets einbezogen und auf sein Kontingent bzw. seine Reservierung (bzw. beide) angerechnet.

```
# zfs set quota=5G pool/filesystem
# zfs set reservation=10G pool/filesystem/user1
cannot set reservation for 'pool/filesystem/user1': size is greater than
available space
```

Ein Dataset kann mehr Speicherplatz belegen, als für seine Reservierung vorgesehen ist, solange nicht reservierter Speicherplatz im Pool vorhanden ist und der aktuell vom Dataset belegte Speicherplatz unter seinem Kontingent liegt. Ein Dataset kann keinen Speicherplatz belegen, der für ein anderes Dataset reserviert wurde.

Reservierungen sind nicht kumulativ. Das bedeutet, dass durch einen zweiten Aufruf von `zfs set` zum Setzen einer Reservierung der Speicherplatz der zweiten Reservierung nicht zum Speicherplatz der vorhandenen Reservierung addiert wird. Die zweite Reservierung ersetzt stattdessen die erste Reservierung.

```
# zfs set reservation=10G tank/home/moore
# zfs set reservation=5G tank/home/moore
# zfs get reservation tank/home/moore
NAME             PROPERTY      VALUE                      SOURCE
tank/home/moore  reservation   5.00G                      local
```

Durch Setzen von `refreservation` können Sie einem Dataset Speicherplatz garantieren, in dem der von Snapshots und Klonen belegte Speicherplatz nicht berücksichtigt ist. Reservierungen durch `refreservation` werden in die Berechnung der Speicherplatzkapazität für das diesem Dataset übergeordnete Dataset einbezogen und auf die Kontingente und Reservierung für das übergeordnete Dataset angerechnet. Beispiel:

```
# zfs set refreservation=10g profs/prof1
# zfs list
NAME                     USED  AVAIL  REFER  MOUNTPOINT
profs                   10.0G  23.2G    19K  /profs
profs/prof1               10G  33.2G    18K  /profs/prof1
```

Sie können außerdem für dasselbe Dataset eine Reservierung festlegen, um Speicherplatz für das Dataset und für Snapshots zu garantieren. Beispiel:

```
# zfs set reservation=20g profs/prof1
# zfs list
NAME                     USED  AVAIL  REFER  MOUNTPOINT
profs                   20.0G  13.2G    19K  /profs
profs/prof1               10G  33.2G    18K  /profs/prof1
```

Normale Reservierungen werden in die Berechnung des belegten Speicherplatzes durch das übergeordnete Dataset einbezogen.

Im obigen Beispiel wird das kleinere der beiden Kontingente (10 GB im Gegensatz zu 20 GB) in der Ausgabe von `zfs list` angezeigt. Zum Anzeigen der Werte beider Kontingente verwenden Sie den Befehl `zfs get`. Beispiel:

```
# zfs get reservation,refreserv profs/prof1
NAME          PROPERTY        VALUE        SOURCE
```

```
profs/prof1  reservation     20G          local
profs/prof1  refreservation  10G          local
```

Wenn `refreservation` gesetzt ist, wird ein Snapshot nur zugelassen, wenn außerhalb dieser Reservierung genügend freier Speicherplatz im Pool vorhanden ist, um die Menge der aktuell *referenzierten* Byte im Dataset aufzunehmen.

◆ ◆ ◆ KAPITEL 7

Arbeiten mit ZFS-Snapshots und -Klonen

Dieses Kapitel enthält Informationen zum Erstellen und Verwalten von ZFS-Snapshots und -Klonen. Darüber hinaus finden Sie hier auch Informationen zum Speichern von Snapshots.

Dieses Kapitel enthält die folgenden Abschnitte:

Überblick über ZFS-Snapshots

Ein *Snapshot* ist eine schreibgeschützte Kopie eines Dateisystems bzw. Volumes. Snapshots können sehr schnell erstellt werden und belegen anfänglich keinen zusätzlichen Speicherplatz im Pool. Mit der Änderung von Daten innerhalb des aktiven Datasets belegt der Snapshot jedoch schrittweise mehr Speicherplatz, da er Verweise auf die älteren Daten speichert und so ein Löschen dieser Daten verhindert

ZFS-Snapshots besitzen die folgenden Leistungsmerkmale:

- Sie bleiben auch nach Systemneustarts wirksam.
- Die theoretische Maximalanzahl möglicher Snapshots beträgt 2^{64}.
- Sie verwenden keine zusätzlichen Datenträger zur Speicherung. Snapshots belegen Speicherplatz direkt im gleichen Speicher-Pool, indem auch das Dateisystem, von dem sie erstellt wurden, vorhanden ist.

- Rekursive Snapshots werden schnell als eine einzige Kernoperation erstellt. Snapshots werden entweder zusammen (d. h. alle auf einmal) oder gar nicht erstellt. Der Vorteil solcher simultaner Snapshots besteht darin, dass die Snapshot-Daten auch bei Dateisystemhierarchien zu einem einzigen konsistenten Zeitpunkt erstellt werden.

Auf Snapshots von Volumes kann nicht direkt zugegriffen werden, aber sie können geklont, als Sicherungskopie gesichert und wiederhergestellt werden. Informationen zum Erstellen von Sicherungskopien für ZFS-Snapshots finden Sie unter „Senden und Empfangen von ZFS-Daten“ auf Seite 225.

Erstellen und Löschen von ZFS-Snapshots

Snapshots werden mithilfe des Befehls `zfs snapshot` erstellt. Diesem wird als einziges Argument der Name des zu erstellenden Snapshots übergeben. Der Snapshot-Name ist wie folgt anzugeben:

filesystem@snapname
volume@snapname

Der Snapshot-Name muss den unter „Konventionen für das Benennen von ZFS-Komponenten“ auf Seite 45 aufgeführten Benennungskonventionen genügen.

Im folgenden Beispiel wird ein Snapshot des Dateisystems `tank/home/ahrens` mit dem Namen `friday` erstellt.

```
# zfs snapshot tank/home/ahrens@friday
```

Mithilfe der Option `-r` können Sie Snapshots für alle untergeordneten Dateisysteme erstellen. Beispiel:

```
# zfs snapshot -r tank/home@now
# zfs list -t snapshot
NAME                    USED  AVAIL  REFER  MOUNTPOINT
tank/home@now             0      -  29.5K  -
tank/home/ahrens@now      0      -  2.15M  -
tank/home/anne@now        0      -  1.89M  -
tank/home/bob@now         0      -  1.89M  -
tank/home/cindys@now      0      -  2.15M  -
```

Snapshots besitzen keine konfigurierbaren Eigenschaften, und es können für Snapshots auch keine Dataset-Eigenschaften gesetzt werden.

```
# zfs set compression=on tank/home/ahrens@tuesday
cannot set compression property for 'tank/home/ahrens@tuesday': snapshot
properties cannot be modified
```

Snapshots werden mithilfe des Befehls `zfs destroy` gelöscht. Beispiel:

```
# zfs destroy tank/home/ahrens@friday
```

Ein Dataset kann nicht gelöscht werden, wenn Snapshots dieses Datasets vorhanden sind. Beispiel:

```
# zfs destroy tank/home/ahrens
cannot destroy 'tank/home/ahrens': filesystem has children
use '-r' to destroy the following datasets:
tank/home/ahrens@tuesday
tank/home/ahrens@wednesday
tank/home/ahrens@thursday
```

Wenn von einem Snapshot Klone erstellt wurden, müssen diese zuerst gelöscht werden, bevor das Löschen des Snapshots möglich ist.

Weitere Informationen zum Unterbefehl `destroy` finden Sie unter „Löschen eines ZFS-Dateisystems“ auf Seite 177.

Umbenennen von ZFS-Snapshots

Sie können Snapshots umbenennen, sie müssen jedoch innerhalb des Pools und Datasets, in dem sie erstellt wurden, umbenannt werden. Beispiel:

```
# zfs rename tank/home/cindys@083006 tank/home/cindys@today
```

Die folgende Kurzsyntax führt die gleiche Umbenennungsoperation wie im obigen Beispiel gezeigt aus.

```
# zfs rename tank/home/cindys@083006 today
```

Die folgende Operation zum Umbenennen eines Snapshots wird nicht unterstützt, da sich Ziel-Pool und -Dateisystem von dem Pool und Dateisystem unterscheiden, in denen der betreffende Snapshot erstellt wurde.

```
# zfs rename tank/home/cindys@today pool/home/cindys@saturday
cannot rename to 'pool/home/cindys@today': snapshots must be part of same
dataset
```

Sie können Snapshots mithilfe des Befehls `zfs rename -r` rekursiv umbenennen. Beispiel:

```
# zfs list
NAME                         USED  AVAIL  REFER  MOUNTPOINT
users                        270K  16.5G    22K  /users
users/home                    76K  16.5G    22K  /users/home
users/home@yesterday           0      -    22K  -
users/home/markm              18K  16.5G    18K  /users/home/markm
users/home/markm@yesterday     0      -    18K  -
users/home/marks              18K  16.5G    18K  /users/home/marks
```

```
users/home/marks@yesterday      0      -    18K  -
users/home/neil               18K  16.5G    18K  /users/home/neil
users/home/neil@yesterday       0      -    18K  -
# zfs rename -r users/home@yesterday @2daysago
# zfs list -r users/home
NAME                        USED  AVAIL  REFER  MOUNTPOINT
users/home                   76K  16.5G    22K  /users/home
users/home@2daysago            0      -    22K  -
users/home/markm             18K  16.5G    18K  /users/home/markm
users/home/markm@2daysago      0      -    18K  -
users/home/marks             18K  16.5G    18K  /users/home/marks
users/home/marks@2daysago      0      -    18K  -
users/home/neil              18K  16.5G    18K  /users/home/neil
users/home/neil@2daysago       0      -    18K  -
```

Anzeigen von und Zugreifen auf ZFS-Snapshots

Sie können das Anzeigen von Snapshot-Listen in der `zfs list`-Ausgabeoutput durch Verwenden der Pool-Eigenschaft `listsnapshots` aktivieren oder deaktivieren. Diese Eigenschaft ist standardmäßig aktiviert.

Wenn Sie diese Eigenschaft deaktivieren, können Sie Snapshot-Informationen mit dem Befehl `zfs list -t snapshot` anzeigen. Oder aktivieren Sie die Pool-Eigenschaft `listsnapshots`. Beispiel:

```
# zpool get listsnapshots tank
NAME  PROPERTY       VALUE      SOURCE
tank  listsnapshots  on        default
# zpool set listsnapshots=off tank
# zpool get listsnapshots tank
NAME  PROPERTY       VALUE      SOURCE
tank  listsnapshots  off         local
```

Snapshots befinden sich im Verzeichnis `.zfs/snapshot` in der Root des betreffenden Dateisystems. Wenn das Dateisystem `tank/home/ahrens` beispielsweise in `/home/ahrens` eingehängt ist, befinden sich die Daten des Snapshots `tank/home/ahrens@thursday` im Verzeichnis `/home/ahrens/.zfs/snapshot/thursday` .

```
# ls /tank/home/ahrens/.zfs/snapshot
tuesday wednesday thursday
```

Snapshots können wie folgt angezeigt werden:

```
# zfs list -t snapshot
NAME                        USED  AVAIL  REFER  MOUNTPOINT
pool/home/anne@monday          0      -   780K  -
```

```
pool/home/bob@monday           0      -  1.01M  -
tank/home/ahrens@tuesday    8.50K      -   780K  -
tank/home/ahrens@wednesday  8.50K      -  1.01M  -
tank/home/ahrens@thursday      0      -  1.77M  -
tank/home/cindys@today      8.50K      -   524K  -
```

Snapshots, die für ein bestimmtes Dateisystem erstellt wurden, können wie folgt angezeigt werden:

```
# zfs list -r -t snapshot -o name,creation tank/home
NAME                          CREATION
tank/home/ahrens@tuesday    Mon Aug 31 11:03 2009
tank/home/ahrens@wednesday  Mon Aug 31 11:03 2009
tank/home/ahrens@thursday   Mon Aug 31 11:03 2009
tank/home/cindys@now        Mon Aug 31 11:04 2009
```

Speicherplatzverwaltung für Snapshots

Bei der Erstellung von Snapshots wird dieser Speicherplatz anfänglich vom Snapshot und dem Dateisystem (sowie eventuellen früheren Snapshots) gemeinsam genutzt. Wenn sich ein Dateisystem mit der Zeit ändert, gehört früher gemeinsam genutzter Speicherplatz dann ausschließlich dem Snapshot und wird in die Berechnung des vom Snapshot belegten Speicherplatzes (Eigenschaft `used`) einbezogen. Darüber hinaus kann durch das Löschen von Snapshots der Speicherplatz, der Snapshots eindeutig zugewiesen ist (und deswegen in der Eigeschaft *used* angegeben ist und somit belegt wird), größer werden.

Der in der Eigenschaft `referenced` eines Snapshots angegebene Speicherplatz ist bei der Erstellung dieses Snapshots gleich dem Speicherplatz, der vom betreffenden Dateisystem belegt wird.

Sie können zusätzliche Informationen darüber erhalten, wie die Werte der Eigenschaft "used" belegt werden. Neue schreibgeschützte Dateisystem-Eigenschaften beschreiben die Speicherplatzbelegung für Klone, Dateisysteme und Volumes. Beispiel:

```
$ zfs list -o space
NAME                  AVAIL   USED  USEDSNAP  USEDDS  USEDREFRESERV  USEDCHILD
rpool                 60.6G  6.37G         0     97K              0      6.37G
rpool/ROOT            60.6G  4.87G         0     21K              0      4.87G
rpool/ROOT/zfs1009BE  60.6G  4.87G         0    4.87G             0          0
rpool/dump            60.6G  1.50G         0    1.50G             0          0
rpool/swap            60.6G    16K         0     16K              0          0
```

Eine Beschreibung dieser Eigenschaften können Sie Tabelle 6–1 entnehmen.

Wiederherstellen eines früheren ZFS-Snapshots

Mit dem Befehl `zfs rollback` können Sie alle Änderungen rückgängig machen, die seit der Erstellung eines bestimmten Snapshots an einem Dateisystem vorgenommen wurden. Im Dateisystem wird der Zustand zum Zeitpunkt der Erstellung des betreffenden Snapshots wiederhergestellt. Standardmäßig stellt dieser Befehl stets den Zustand des zuletzt gemachten Snapshots wieder her.

Damit das Dateisystem im Zustand eines früheren Snapshots wiederhergestellt werden kann, müssen alle dazwischen liegenden Snapshots gelöscht werden. Frühere Snapshots können mithilfe der Option `-r` gelöscht werden.

Wenn Klone dazwischen liegender Snapshots vorhanden sind, müssen auch diese Klone mithilfe der Option `-R` gelöscht werden.

Hinweis – Das Dateisystem, dessen früherer Zustand wiederhergestellt werden soll, muss ausgehängt und dann wieder eingehängt werden, wenn es gegenwärtig eingehängt ist. Wenn das betreffende Dateisystem nicht ausgehängt werden kann, schlägt die Wiederherstellung des früheren Zustands fehl. Die Option `-f` erzwingt bei Bedarf das Aushängen des Dateisystems.

Im folgenden Beispiel wird das Dateisystem `tank/home/ahrens` auf den Snapshot mit dem Namen `dienstag` zurückgesetzt:

```
# zfs rollback tank/home/ahrens@tuesday
cannot rollback to 'tank/home/ahrens@tuesday': more recent snapshots exist
use '-r' to force deletion of the following snapshots:
tank/home/ahrens@wednesday
tank/home/ahrens@thursday
# zfs rollback -r tank/home/ahrens@tuesday
```

Im obigen Beispiel werden die Snapshots `wednesday` und `thursday` gelöscht, da Sie den Zustand des davor liegenden Snapshots `tuesday` wiederherstellen wollen.

```
# zfs list -r -t snapshot -o name,creation tank/home/ahrens
NAME                      CREATION
tank/home/ahrens@tuesday  Wed Aug 27 16:35 2008
```

Überblick über ZFS-Klone

Ein *Klon* ist ein schreibbares Volume bzw. Dateisystem, dessen anfänglicher Inhalt gleich dem des Datasets ist, von dem er erstellt wurde. Wie bei Snapshots werden auch Klone sehr schnell erstellt und belegen anfänglich keinen zusätzlichen Speicherplatz. Darüber hinaus können Sie Snapshots von Klonen erstellen.

- „Erstellen eines ZFS-Klons“ auf Seite 223
- „Löschen eines ZFS-Klons“ auf Seite 224
- „Ersetzen eines ZFS-Dateisystems durch einen ZFS-Klon“ auf Seite 224

Klone können nur von Snapshots erstellt werden. Beim Klonen eines Snapshots wird zwischen dem Klon und dem Snapshot eine implizite Abhängigkeit erstellt. Obwohl der betreffende Klon an anderer Stelle der Dataset-Hierarchie erstellt wird, kann der ursprüngliche Snapshot nicht gelöscht werden, solange von ihm ein Klon vorhanden ist. Die Eigenschaft `origin` enthält diese Abhängigkeit, und mit dem Befehl `zfs destroy` können solche Abhängigkeiten aufgelistet werden, falls sie vorhanden sind.

Klone erben keine Eigenschaften von dem Dataset, von dem sie erstellt wurden. Mit den Befehlen `zfs get` und `zfs set` können Sie die Eigenschaften eines geklonten Datasets anzeigen und ändern. Weitere Informationen zum Setzen von Eigenschaften von ZFS-Datasets finden Sie unter „Setzen von ZFS-Eigenschaften“ auf Seite 196.

Da ein Klon seinen gesamten Speicherplatz anfänglich mit dem ursprünglichen Snapshot gemeinsam nutzt, ist seine Eigenschaft `used` zu Beginn auf null gesetzt. Wenn mit der Zeit am Klon Änderungen vorgenommen werden, belegt er schrittweise mehr Speicherplatz. Die Eigenschaft `used` des ursprünglichen Snapshots berücksichtigt keinen vom Klon belegten Speicherplatz.

Erstellen eines ZFS-Klons

Klone werden mit dem Befehl `zfs clone` erstellt. Sie müssen den Snapshot, von dem der Klon erstellt werden soll, sowie den Namen des neuen Dateisystems bzw. Volumes angeben. Das neue Dateisystem bzw. Volume kann sich an beliebiger Stelle innerhalb der ZFS-Hierarchie befinden. Der Typ des neuen Datasets (z. B. Dateisystem oder Volume) ist gleich dem des Snapshots, von dem der Klon erstellt wurde. Sie können von einem Dateisystem in einem anderen Pool als dem des Snapshots des ursprünglichen Dateisystems keinen Klon erstellen.

Im folgenden Beispiel wird ein neuer Klon `tank/home/ahrens/bug123` mit dem gleichen anfänglichen Inhalt wie der des Snapshots `tank/ws/gate@gestern` erstellt.

```
# zfs snapshot tank/ws/gate@yesterday
# zfs clone tank/ws/gate@yesterday tank/home/ahrens/bug123
```

Im folgenden Beispiel wird für einen temporären Benutzer aus dem Snapshot `projects/newproject@today` ein geklonter Arbeitsbereich namens `projects/teamA/tempuser` erstellt. Danach werden die Eigenschaften des geklonten Arbeitsbereichs gesetzt.

```
# zfs snapshot projects/newproject@today
# zfs clone projects/newproject@today projects/teamA/tempuser
# zfs set sharenfs=on projects/teamA/tempuser
# zfs set quota=5G projects/teamA/tempuser
```

Löschen eines ZFS-Klons

ZFS-Klone werden mithilfe des Befehls `zfs destroy` gelöscht. Beispiel:

```
# zfs destroy tank/home/ahrens/bug123
```

Bevor ein übergeordneter Snapshot gelöscht werden kann, müssen zunächst seine Klone gelöscht werden.

Ersetzen eines ZFS-Dateisystems durch einen ZFS-Klon

Mit dem Befehl `zfs promote` können Sie ein aktives ZFS-Dateisystem durch einen Klon dieses Dateisystems ersetzen. Diese Funktion ermöglicht das Klonen und Ersetzen von Dateisystemen, sodass das *ursprüngliche* Dateisystem der Klon des betreffenden Dateisystems werden kann. Darüber hinaus ermöglicht diese Funktion das Löschen des Dateisystems, von dem der Klon ursprünglich erstellt wurde. Ohne diese „Klon-Promotion" können Dateisysteme, von denen Klone ursprünglich erstellt wurden, nicht gelöscht werden. Weitere Informationen zum Löschen von Klonen finden Sie unter „Löschen eines ZFS-Klons" auf Seite 224.

Im folgenden Beispiel wird das Dateisystem `tank/test/produktA` geklont; dann wird dieser Klon (`tank/test/produktAbeta`) zum Dateisystem `tank/test/produktA` gemacht.

```
# zfs create tank/test
# zfs create tank/test/productA
# zfs snapshot tank/test/productA@today
# zfs clone tank/test/productA@today tank/test/productAbeta
# zfs list -r tank/test
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank/test              314K  8.24G  25.5K  /tank/test
tank/test/productA     288K  8.24G   288K  /tank/test/productA
tank/test/productA@today      0      -   288K  -
```

```
tank/test/productAbeta      0  8.24G   288K  /tank/test/productAbeta
# zfs promote tank/test/productAbeta
# zfs list -r tank/test
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank/test              316K  8.24G  27.5K  /tank/test
tank/test/productA        0  8.24G   288K  /tank/test/productA
tank/test/productAbeta   288K  8.24G   288K  /tank/test/productAbeta
tank/test/productAbeta@today      0      -   288K  -
```

In der obigen Ausgabe des Befehls `zfs list` sehen Sie, dass die Speicherplatzangabe des ursprünglichen Dateisystems `produktA` durch die des Dateisystems `produktAbeta` ersetzt wurde.

Sie müssen den Ersetzungsvorgang durch Umbenennen der Dateisysteme abschließen. Beispiel:

```
# zfs rename tank/test/productA tank/test/productAlegacy
# zfs rename tank/test/productAbeta tank/test/productA
# zfs list -r tank/test
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank/test              316K  8.24G  27.5K  /tank/test
tank/test/productA     288K  8.24G   288K  /tank/test/productA
tank/test/productA@today      0      -   288K  -
tank/test/productAlegacy      0  8.24G   288K  /tank/test/productAlegacy
```

Optional können Sie auch das alte Dateisystem entfernen. Beispiel:

```
# zfs destroy tank/test/productAlegacy
```

Senden und Empfangen von ZFS-Daten

Der Befehl `zfs send` erstellt von Snapshots Datenstreaminstanzen, die auf die Standardausgabe geschrieben werden. Standardmäßig wird eine vollständige Datenstreaminstanz erzeugt. Sie können die Ausgabe in eine Datei oder ein anderes Dateisystem umleiten. Der Befehl `zfs receive` erstellt einen Snapshot, dessen Inhalt von einer Datenstreaminstanz an der Standardeingabe gelesen wird. Wenn eine vollständige Datenstreaminstanz gelesen wurde, wird ein neues Dateisystem erstellt. Mit diesen Befehlen können Sie ZFS-Snapshot-Daten senden und ZFS-Snapshot-Daten und Dateisysteme empfangen. Siehe hierzu die Beispiele im nachfolgenden Abschnitt.

- „Senden von ZFS-Snapshots“ auf Seite 226
- „Empfangen von ZFS-Snapshots“ auf Seite 227
- „Replikation von ZFS-Daten über Netzwerk“ auf Seite 232
- „Sichern von ZFS-Daten mit anderen Softwarepaketen zur Erstellung von Sicherungskopien“ auf Seite 232

Die folgenden Lösungen zum Sichern von ZFS-Daten stehen zur Verfügung:

- Sicherungsprodukte für Unternehmen - Wenn folgende Leistungsmerkmale erwünscht sind, sollte eine Unternehmenslösung für die Datensicherung in Betracht gezogen werden:
  - Wiederherstellung auf Dateibasis
  - Überprüfung der Datensicherungsmedien
  - Medienverwaltung
- Dateisystem-Snapshots und Wiederherstellen des früheren Zustands eines Dateisystems – Mit den Befehlen `zfs snapshot` und `zfs rollback` können Sie auf einfache Weise Kopien von Dateisystemen erstellen und aus diesen bei Bedarf Dateisysteme auf einen früheren Zustand zurücksetzen. Sie können die Lösung beispielsweise verwenden, wenn Sie ein Dateisystem bzw. Dateien aus einer früheren Dateisystemversion wiederherstellen möchten.

  Weitere Informationen zum Erstellen von Snapshots und Wiederherstellen einer früheren Version aus einem Snapshot finden Sie unter „Überblick über ZFS-Snapshots“ auf Seite 217.
- Sichern von Snapshots – Mit den Befehlen `zfs send` und `zfs receive` können Sie ZFS-Snapshots senden und empfangen. Sie können inkrementelle Änderungen zwischen Snapshots sichern, Dateien können jedoch nicht einzeln wiederhergestellt werden. Stattdessen muss der gesamte Dateisystem-Snapshot wiederhergestellt werden. Diese Befehle stellen keine vollständige Sicherungslösung für Ihre ZFS-Daten dar.
- Replikation über Netzwerk – Mit den Befehlen `zfs send` und `zfs receive` können Sie Dateisysteme von einem System auf ein anderes kopieren. Dieser Vorgang unterscheidet sich von herkömmlichen Praktiken bei Software zur Datenträgerverwaltung, die Datenspeichergeräte über WAN spiegeln. Dafür ist keine spezielle Konfiguration bzw. Hardware erforderlich. Der Vorteil der Replikation eines ZFS-Dateisystems besteht darin, dass das betreffende Dateisystem in einem Speicher-Pool eines anderen Systems neu erstellt werden kann und Sie für diesen neuen Pool verschiedene Replikationsmethoden wie z. B. RAID-Z angeben können, die Dateisystemdaten aber gleich bleiben.
- Archivierungsdienstprogramme – Speichern von ZFS-Daten mit Archivierungsdienstprogrammen wie `tar`, `cpio` und `pax` oder Datensicherungssoftware von Drittherstellern. Derzeit konvertieren `tar` und `cpio` die Zugriffssteuerungslisten im NFSv4-Stil korrekt, während `pax` Zugriffssteuerungslisten im NFSv4-Stil nicht korrekt konvertiert.

Senden von ZFS-Snapshots

Der Befehl `zfs send` dient zum Senden einer Snapshot-Kopie und Empfangen des Snapshots in einem anderen Pool auf demselben System oder in einem anderen Pool auf einem anderen System, das zur Aufbewahrung von Sicherungsdaten verwendet wird. Zum Senden des Snapshots an einen anderen Pool auf demselben System verwenden Sie beispielsweise die folgende Syntax:

```
# zfs send tank/data@snap1 | zfs recv spool/ds01
```

Wenn Sie den Snapshot-Datenstream an ein anderes System senden, setzen Sie für die Ausgabe von `zfs send` mit dem Befehl `ssh` eine Pipe. Beispiel:

```
host1# zfs send tank/dana@snap1 | ssh host2 zfs recv newtank/dana
```

Wenn ein vollständiger Datenstream gesendet wird, darf das Zieldateisystem nicht vorhanden sein.

Sie können inkrementelle Daten mit der Option `i` des Befehls `-zfs send` senden. Beispiel:

```
host1# zfs send -i tank/dana@snap1 tank/dana@snap2 | ssh host2 zfs recv newtank/dana
```

Bitte beachten Sie, dass das erste Argument (*snap1*) der frühere und das zweite Argument (*snap2*) der spätere Snapshot ist. In diesem Fall muss das Dateisystem `newtank/dana` vorhanden sein, damit die inkrementellen Daten erfolgreich gelesen werden können.

Die inkrementelle Quelle *snap1* kann als letzte Komponente des Snapshot-Namens angegeben werden. Dies bedeutet, dass Sie nach dem Zeichen `@` für *snap1* nur den Namen angeben müssen, von dem angenommen wird, dass er zum gleichen System wie *snap2* gehört. Beispiel:

```
host1# zfs send -i snap1 tank/dana@snap2 > ssh host2 zfs recv newtank/dana
```

Diese Syntax entspricht der im obigen Beispiel demonstrierten inkrementellen Syntax.

Die folgende Meldung wird angezeigt, wenn Sie versuchen, von einem anderen Dateisystem *Snapshot1* einen inkrementellen Datenstream zu erzeugen.

```
cannot send 'pool/fs@name': not an earlier snapshot from the same fs
```

Wenn mehrere Kopien gespeichert werden sollen, könnte eine Komprimierung der Datenstreaminstanz des ZFS-Snapshots mithilfe des Befehls `gzip` nützlich sein. Beispiel:

```
# zfs send pool/fs@snap | gzip > backupfile.gz
```

Empfangen von ZFS-Snapshots

Beim Empfangen von Datensystem-Snapshots sollten Sie folgende Schlüsselaspekte berücksichtigen:

- Es werden der Snapshot und das Dateisystem empfangen.
- Das Dateisystem und alle untergeordneten Dateisysteme werden ausgehängt.
- Während des Empfangs kann auf die betreffenden Dateisysteme nicht zugegriffen werden.
- Das ursprüngliche Dateisystem, das empfangen werden soll, darf bei der Übertragung nicht vorhanden sein.
- Falls ein Namenskonflikt mit einem anderen Dateisystem auftritt, können Sie das betreffende Dateisystem mit dem Befehl `zfs rename` umbenennen.

Beispiel:

```
# zfs send tank/gozer@0830 > /bkups/gozer.083006
# zfs receive tank/gozer2@today < /bkups/gozer.083006
# zfs rename tank/gozer tank/gozer.old
# zfs rename tank/gozer2 tank/gozer
```

Sie können `zfs recv` als Aliasnamen für den Befehl `zfs receive` verwenden.

Wenn Sie am Zieldateisystem eine Änderung vornehmen und danach eine weitere inkrementelle Sicherungskopie des Snapshots senden wollen, müssen Sie für das Zieldateisystem einen Rollback durchführen (den vorherigen Zustand wiederherstellen).

Wenn Sie das Dateisystem beispielsweise wie folgt ändern:

```
host2# rm newtank/dana/file.1
```

und eine inkrementelle Sicherungskopie von `tank/dana@snap3` senden, müssen Sie erst den vorherigen Zusstand des Zieldateisystems wiederherstellen, damit es den neuen inkrementellen Snapshot empfangen kann. Mithilfe der Option `-F` können Sie den Wiederherstellungsschritt überspringen. Beispiel:

```
host1# zfs send -i tank/dana@snap2 tank/dana@snap3 | ssh host2 zfs recv -F newtank/dana
```

Beim Empfang eines inkrementellen Snapshots muss das Zieldateisystem bereits vorhanden sein.

Wenn Sie am Dateisystem Änderungen vornehmen und den vorherigen Zustand des Zieldateisystems nicht wiederherstellen, sodass es den neuen inkrementellen Snapshot empfangen kann, oder Sie die Option `-F` nicht verwenden, wird die folgende Meldung angezeigt:

```
host1# zfs send -i tank/dana@snap4 tank/dana@snap5 | ssh host2 zfs recv newtank/dana
cannot receive: destination has been modified since most recent snapshot
```

Bevor das Ausführen der Option `-F` als erfolgreich gemeldet wird, werden die folgenden Überprüfungen durchgeführt:

- Wenn der letzte Snapshot nicht mit der inkrementellen Quelle identisch ist, wird weder die Wiederherstellung des früheren Zustands noch der Empfang der Sicherungskopie angeschlossen, und es wird eine Fehlermeldung angezeigt.
- Wenn Sie versehentlich den Namen eines anderen Dateisystems angeben, der mit dem inkrementellen Quellparameter des Befehls `zfs receive` nicht übereinstimmt, wird weder die Wiederherstellung des früheren Zustands noch der Empfang der Sicherungskopie angeschlossen, und es wird die folgende Fehlermeldung angezeigt.

  ```
  cannot send 'pool/fs@name': not an earlier snapshot from the same fs
  ```

Senden und Empfangen komplexer ZFS-Snapshot-Datenstreams

In diesem Abschnitt wird das Senden und Empfangen komplexerer Snapshot-Datenstreams mit den Befehlen `zfs send -I` und `-R` beschrieben.

Beachten Sie beim Senden und Empfangen von ZFS-Snapshot-Datenstreams die folgenden Faktoren:

- Verwenden Sie `zfs send` mit der Option `-I`, um alle inkrementellen Datenstreams aus einem Snapshot an einen kumulativen Snapshot zu senden. Oder verwenden Sie diese Option, um einen inkrementellen Datenstream aus dem ursprünglichen Snapshot zum Erstellen eines Klons zu senden. Der ursprüngliche Snapshot muss auf der Empfangsseite bereits vorhanden sein, damit der inkrementelle Stream angenommen werden kann.
- Mit `zfs send` und der Option `-R` senden Sie einen Replikations-Datenstream aller untergeordneten Dateisysteme. Nach dem Empfang werden alle Eigenschaften, Snapshots, abhängigen Dateisysteme und Klone beibehalten.
- Oder verwenden Sie beide Optionen, um einen inkrementellen Replikations-Datenstream zu senden.
  - Änderungen an Eigenschaften, Namensänderungen von Snapshots und Dateisystemen sowie Löschvorgänge werden beibehalten.
  - Wenn `zfs recv -F` beim Empfang des Replikations-Datenstreams nicht angegeben ist, werden Löschvorgänge von Datasets ignoriert. In diesem Fall behält die Syntax `zfs recv -F` auch die Bedeutung *Bei Bedarf Rollback* bei.
  - Wie in anderen Fällen (außer `zfs send -R`) mit `- i` oder `-I` werden bei Verwendung von `-I` alle Snapshots zwischen snapA und snapD gesendet. Bei Verwendung von `-i` wird nur snapD (für sämtliche untergeordneten Objekte) gesendet.
- Der Empfang dieser neuen Arten von Datenstreams per `zfs send` setzt voraus, dass auf dem empfangenden System eine Softwareversion ausgeführt wird, die in der Lage ist, diese Datenstreams zu senden. Die Version des Datenstreams wird erhöht.

  Es ist jedoch möglich, auf Streams aus älteren Pool-Versionen über neuere Softwareversionen zuzugreifen. So können Sie beispielsweise Datenstreams, die mit den neueren Optionen erstellt wurden, aus und in Pools der Version 3 senden. Zum Empfangen eines mit den neueren Optionen gesendeten Datenstreams muss jedoch aktuelle Software ausgeführt werden.

BEISPIEL 7–1 Beispiele - Senden und Empfangen komplexer ZFS-Snapshot-Datenstreams

Eine Gruppe inkrementeller Snapshots lässt sich mithilfe von `zfs send` und der Option `-I` zu einem Snapshot kombinieren. Beispiel:

```
# zfs send -I pool/fs@snapA pool/fs@snapD > /snaps/fs@all-I
```

BEISPIEL 7–1 Beispiele - Senden und Empfangen komplexer ZFS-Snapshot-Datenstreams *(Fortsetzung)*

Löschen der Snapshots B, C und D.

```
# zfs destroy pool/fs@snapB
# zfs destroy pool/fs@snapC
# zfs destroy pool/fs@snapD
```

Empfangen des kombinierten Snapshots.

```
# zfs receive -d -F pool/fs < /snaps/fs@all-I
# zfs list
NAME                      USED  AVAIL  REFER  MOUNTPOINT
pool                      428K  16.5G    20K  /pool
pool/fs                    71K  16.5G    21K  /pool/fs
pool/fs@snapA              16K      -  18.5K  -
pool/fs@snapB              17K      -    20K  -
pool/fs@snapC              17K      -  20.5K  -
pool/fs@snapD                0      -    21K  -
```

Außerdem können Sie mit `zfs send -I` einen Snapshot und einen Klon-Snapshot zu einem kombinierten Dataset verbinden. Beispiel:

```
# zfs create pool/fs
# zfs snapshot pool/fs@snap1
# zfs clone pool/fs@snap1 pool/clone
# zfs snapshot pool/clone@snapA
# zfs send -I pool/fs@snap1 pool/clone@snapA > /snaps/fsclonesnap-I
# zfs destroy pool/clone@snapA
# zfs destroy pool/clone
# zfs receive -F pool/clone < /snaps/fsclonesnap-I
```

Mit dem Befehl `zfs send -R` werden ein ZFS-Dateisystem und alle untergeordneten Dateisysteme bis hin zum genannten Snapshot repliziert. Nach dem Empfang werden alle Eigenschaften, Snapshots, abhängigen Dateisysteme und Klone beibehalten.

Im folgenden Beispiel werden Snapshots von Benutzerdateisystemen erstellt. Es wird ein Replikations-Datenstream aller Benutzer-Snapshots erstellt. Anschließend werden die ursprünglichen Dateisysteme und Snapshots gelöscht und wiederhergestellt.

```
# zfs snapshot -r users@today
# zfs list
NAME                 USED  AVAIL  REFER  MOUNTPOINT
users                187K  33.2G    22K  /users
users@today             0      -    22K  -
users/user1          18K  33.2G    18K  /users/user1
users/user1@today       0      -    18K  -
```

BEISPIEL 7–1 Beispiele - Senden und Empfangen komplexer ZFS-Snapshot-Datenstreams *(Fortsetzung)*

```
users/user2          18K  33.2G    18K  /users/user2
users/user2@today      0      -    18K  -
users/user3          18K  33.2G    18K  /users/user3
users/user3@today      0      -    18K  -
# zfs send -R users@today > /snaps/users-R
# zfs destroy -r users
# zfs receive -F -d users < /snaps/users-R
# zfs list
NAME                USED  AVAIL  REFER  MOUNTPOINT
users               196K  33.2G    22K  /users
users@today            0      -    22K  -
users/user1          18K  33.2G    18K  /users/user1
users/user1@today      0      -    18K  -
users/user2          18K  33.2G    18K  /users/user2
users/user2@today      0      -    18K  -
users/user3          18K  33.2G    18K  /users/user3
users/user3@today      0      -    18K  -
```

Mit dem Befehl `zfs send -R` können Sie das `users` Dataset und seine untergeordneten Objekte replizieren und den replizierten Datenstream an einen anderen Pool, `users2`, senden.

```
# zfs create users2 mirror c0t1d0 c1t1d0
# zfs receive -F -d users2 < /snaps/users-R
# zfs list
NAME                 USED  AVAIL  REFER  MOUNTPOINT
users                224K  33.2G    22K  /users
users@today             0      -    22K  -
users/user1           33K  33.2G    18K  /users/user1
users/user1@today     15K      -    18K  -
users/user2           18K  33.2G    18K  /users/user2
users/user2@today        0      -    18K  -
users/user3           18K  33.2G    18K  /users/user3
users/user3@today        0      -    18K  -
users2               188K  16.5G    22K  /users2
users2@today            0      -    22K  -
users2/user1          18K  16.5G    18K  /users2/user1
users2/user1@today       0      -    18K  -
users2/user2          18K  16.5G    18K  /users2/user2
users2/user2@today       0      -    18K  -
users2/user3          18K  16.5G    18K  /users2/user3
users2/user3@today       0      -    18K  -
```

Replikation von ZFS-Daten über Netzwerk

Mit den Befehlen `zfs send` und `zfs recv` können Sie eine Datenstreaminstanz über Netzwerk von einem System auf ein anderes kopieren. Beispiel:

```
# zfs send tank/cindy@today | ssh newsys zfs recv sandbox/restfs@today
```

Dieser Befehl sendet den Snapshot `tank/cindy@today` und empfängt ihn im Dateisystem `sandbox/restfs`. Darüber hinaus wird auf dem System `newsys` der Snapshot `restfs@today` erstellt. In diesem Beispiel wird auf dem entfernten System `ssh` verwendet.

Sichern von ZFS-Daten mit anderen Softwarepaketen zur Erstellung von Sicherungskopien

Neben den Befehlen `zfs send` und `zfs receive` können Sie zum Sichern von ZFS-Dateien auch Archivierungsdienstprogramme wie z. B. `tar` und `cpio` verwenden. Alle diese Dienstprogramme sichern ZFS-Dateiattribute und -Zugriffssteuerungslisten und können diese auch wiederherstellen. Überprüfen Sie die entsprechenden Optionen der Befehle `tar` und `cpio`.

Aktuelle Informationen zu Problemen im Zusammenhang mit ZFS und Sicherungssoftware von Drittherstellern finden Sie in den Solaris 10-Versionshinweisen oder den ZFS FAQs:

`http://opensolaris.org/os/community/zfs/faq/#backupsoftware`

◆ ◆ ◆ KAPITEL 8

Schützen von ZFS-Dateien mit Zugriffssteuerungslisten

Dieses Kapitel enthält Informationen zum Arbeiten mit Zugriffssteuerungslisten. Solche Listen schützen ZFS-Dateien, weil sie im Vergleich zu den UNIX-Standardzugriffsrechten feiner abgestimmte Zugriffsrechte definieren und anwenden.

Dieses Kapitel enthält die folgenden Abschnitte:

Neues Solaris-Modell für Zugriffssteuerungslisten

Frühere Versionen des Betriebssystems Solaris unterstützten eine auf der POSIX-Spezifikation beruhende Implementierung von Zugriffssteuerungslisten. POSIX-basierte Zugriffssteuerungslisten dienen zum Schutz von UFS-Dateien und werden von NFS-Versionen vor NFSv4 verwendet.

Seit der Einführung von NFSv4 unterstützt ein neues Modell für Zugriffssteuerungslisten vollständig die Interoperabilität, die NFSv4 für die Kommunikation zwischen UNIX-Clients und anderen Clients bietet. Die neue, in der NFSv4-Spezifikation definierte Implementierung von Zugriffssteuerungslisten bietet eine reichhaltigere Semantik, die auf NT-basierten Zugriffssteuerungslisten beruht.

Die Hauptunterschiede des neuen Zugriffssteuerungslistenmodells bestehen in Folgendem:

- Basiert auf der NFSv4-Spezifikation und ähnelt NT-Zugriffssteuerungslistenmodellen,
- enthält einen feiner abstimmbaren Satz an Zugriffsrechten, Weitere Informationen finden Sie in Tabelle 8–2.

- wird mit den Befehlen `chmod` und `ls` anstatt `setfacl` und `getfacl` eingestellt und angezeigt,
- enthält eine reichhaltigere Vererbungssemantik, um festlegen zu können, wie Zugriffsrechte von über- und untergeordneten Verzeichnissen geregelt werden, usw. Weitere Informationen dazu finden Sie in „Vererbung von Zugriffssteuerungslisten“ auf Seite 238.

Beide Zugriffssteuerungslistenmodelle bieten eine feiner abstimmbare Kontrolle von Zugriffsrechten als die Standardzugriffsrechte. Wie bei POSIX-basierten Zugriffssteuerungslisten bestehen auch die neuen Zugriffssteuerungslisten aus mehreren Zugriffssteuerungseinträgen.

POSIX-basierte Zugriffssteuerungslisten definieren mithilfe eines einzigen Eintrags, welche Zugriffsrechte zulässig und unzulässig sind. Das neue Zugriffssteuerungslistenmodell besitzt zwei Arten von Zugriffssteuerungseinträgen, die sich auf die Überprüfung von Zugriffsrechten auswirken: `ALLOW` (Erlauben) und `DENY` (Verweigern). Demzufolge können Sie nicht aus einem einzigen Zugriffssteuerungseintrag schließen, ob die in diesem Zugriffssteuerungseintrag nicht definierten Zugriffsrechte zulässig sind oder nicht.

Die Konvertierung zwischen NFSv4-basierten und POSIX-basierten Zugriffssteuerungslisten geschieht wie folgt:

- Bei der Verwendung von Dienstprogrammen, die Zugriffsrechte überprüfen (z. B. die Befehle `cp`, `mv`, `tar`, `cpio` oder `rcp`) werden die POSIX-basierten Zugriffssteuerungslisten in die entsprechenden NFSv4-basierten Zugriffssteuerungslisten konvertiert, damit UFS-Dateien mit Zugriffssteuerungslisten in ein ZFS-Dateisystem transferiert werden können.
- Einige NFSv4-basierte Zugriffssteuerungslisten werden in POSIX-basierte Zugriffssteuerungslisten konvertiert. Wenn eine NFSv4–basierte Zugriffssteuerungsliste nicht in eine POSIX-basierte Zugriffssteuerungsliste konvertiert werden kann, wird in etwas die folgende Meldung angezeigt:

  ```
  # cp -p filea /var/tmp
  cp: failed to set acl entries on /var/tmp/filea
  ```

- Wenn Sie auf einem System, auf dem die neueste Solaris-Version installiert ist, mit `tar` oder `cpio` ein UFS-Archiv mit der Option zur Beibehaltung der Zugriffssteuerungsliste (`tar -p` bzw. `cpio -P`) erstellen, gehen beim Extrahieren des Archivs auf einem System, auf dem eine frühere Solaris-Version installiert ist, die Zugriffssteuerungslisten verloren.

  Alle Dateien werden mit den ordnungsgemäßen Dateimodi extrahiert, aber die Zugriffssteuerungslisten werden ignoriert.
- Der Befehl `ufsrestore` ermöglicht es, Daten in einem ZFS-Dateisystem wiederherzustellen. Wenn die Originaldaten POSIX-basierte Zugriffssteuerungslisten enthalten, werden diese in NFSv4-basierte Zugriffssteuerungslisten konvertiert.
- Wenn Sie versuchen, eine NFSv4-basierte Zugriffssteuerungsliste an einer UFS-Datei zu setzen, wird eine Meldung wie die folgende angezeigt:

```
chmod: ERROR: ACL type's are different
```

- Wenn Sie versuchen, eine POSIX-basierte Zugriffssteuerungsliste an einer ZFS-Datei zu setzen, wird eine Meldung wie die folgende angezeigt:

```
# getfacl filea
File system doesn't support aclent_t style ACL's.
See acl(5) for more information on Solaris ACL support.
```

Informationen zu Einschränkungen mit Zugriffssteuerungslisten und Backup-Software finden Sie unter „Sichern von ZFS-Daten mit anderen Softwarepaketen zur Erstellung von Sicherungskopien“ auf Seite 232.

Syntaxbeschreibungen zum Setzen von Zugriffssteuerungslisten

Es gibt zwei grundlegende Zugriffssteuerungslistenformate:

Syntax zum Setzen gewöhnlicher Zugriffssteuerungslisten

`chmod [Optionen] A[Index]{+|=}owner@ |group@ |everyone@:` *Zugriffsrechte*/...[:*Vererbungsflags*]: `deny | allow` *Datei*

`chmod [Optionen] A-owner@, group@, everyone@:` *Zugriffsrechte*/...[:*Vererbungsflags*]: `deny | allow` *Datei ...*

`chmod [Optionen] A[Index]-` *Datei*

Syntax zum Setzen komplexerer Zugriffssteuerungslisten

`chmod [Optionen] A[Index]{+|=}user|group:name:`*Zugriffsrechte* /...[:*Vererbungsflags*]`:deny | allow` *Datei*

`chmod [Optionen] A-user|group:name:`*Zugriffsrechte* /...[:*Vererbungsflags*]`:deny | allow` *Datei ...*

`chmod [Optionen] A[Index]-` *Datei*

owner@, group@, everyone@
: Der *Zugriffssteuerungslisten-Eintragstyp* für die gewöhnliche Zugriffssteuerungslistensyntax. Eine Beschreibung der *Zugriffssteuerungslisten-Eintragstypen* finden Sie in Tabelle 8–1.

user oder group:*Zugriffssteuerungslisten-Eintragstyp-ID=Benutzername oder Gruppenname*
: Der *Zugriffssteuerungslisten-Eintragstyp* für die explizite Zugriffssteuerungslistensyntax. Der *Zugriffssteuerungslisten-Eintragstyp* „user” bzw. ”group” muss auch die *Zugriffssteuerungslisteneintrags-ID*, den *Benutzernamen* bzw. *Gruppennamen* enthalten. Eine Beschreibung der *Zugriffssteuerungslisten-Eintragstypen* finden Sie in Tabelle 8–1.

Zugriffsrechte/.../
Die Zugriffsrechte, die gewährt bzw. verweigert werden. Eine Beschreibung von Zugriffsrechten für Zugriffssteuerungslisten finden Sie in Tabelle 8–2.

Vererbungsflags
Eine optionale Liste von Vererbungsflags von Zugriffssteuerungslisten. Eine Beschreibung der Vererbungsflags von Zugriffssteuerungslisten finden Sie in Tabelle 8–3.

`deny | allow`
Legt fest, ob Zugriffsrechte gewährt oder verweigert werden.

Im folgenden Beispiel spielt die *Zugriffssteuerungslisteneintrags-ID* keine Rolle.

```
group@:write_data/append_data/execute:deny
```

Das folgende Beispiel enthält eine *Zugriffssteuerungslisteneintrags-ID*, da ein spezifischer Benutzer (*Zugriffssteuerungslisten-Eintragstyp*) in der Zugriffssteuerungsliste enthalten ist.

```
0:user:gozer:list_directory/read_data/execute:allow
```

Ein Zugriffssteuerungslisteneintrag sieht ungefähr wie folgt aus:

```
2:group@:write_data/append_data/execute:deny
```

Die **2** bzw. die *Index-ID* in diesem Beispiel identifiziert den Zugriffssteuerungslisteneintrag in der größeren Zugriffssteuerungsliste, in der für Eigentümer, spezifische UIDs, Gruppen und allgemeine Zugriffsrechte mehrere Einträge vorhanden sein können. Sie können die *Index-ID* mit dem Befehl `chmod` angeben und somit festlegen, welchen Teil der Zugriffssteuerungsliste Sie ändern möchten. Sie können beispielsweise Index-ID 3 als `A3` im Befehl `chmod` angeben (siehe folgendes Beispiel):

```
chmod A3=user:venkman:read_acl:allow filename
```

Zugriffssteuerungslisten-Eintragstypen, die Eigentümer, Gruppen und andere Parameter in Zugriffssteuerungslisten darstellen, sind in der folgenden Tabelle beschrieben.

TABELLE 8–1 Zugriffssteuerungslisten- Eintragstypen

Zugriffssteuerungslisten-Eintragstyp	Beschreibung
`owner@`	Das Zugriffsrecht, das dem Eigentümer des Objekts gewährt wird.
`group@`	Das Zugriffsrecht, das der Eigentümergruppe des Objekts gewährt wird.
`everyone@`	Der Zugriff, der allen anderen Benutzern oder Gruppen gewährt wird, denen keine anderen Zugriffssteuerungslisteneinträge entsprechen.

TABELLE 8–1 Zugriffssteuerungslisten- Eintragstypen *(Fortsetzung)*

Zugriffssteuerungslisten-Eintragstyp	Beschreibung
`user`	Das Zugriffsrecht, das einem zusätzlichen (und durch den Benutzernamen anzugebenden) Benutzer des Objekts gewährt wird. Muss die *Zugriffssteuerungslisteneintrags-ID* enthalten, die wiederum den betreffenden *Benutzernamen* oder die *Benutzer-ID* enthält. Wenn es sich bei diesem Wert um keine gültige nummerische Benutzer-ID oder keinen *Benutzername* handelt, ist der Zugriffssteuerungslisten-Eintragstyp ungültig.
`group`	Das Zugriffsrecht, das einer zusätzlichen (und durch den Gruppennamen anzugebenden) Benutzergruppe des Objekts gewährt wird. Muss die *Zugriffssteuerungslisteneintrags-ID* enthalten, die wiederum den betreffenden *Gruppennamen* oder die *Gruppen-ID* enthält. Wenn es sich bei diesem Wert um keine gültige nummerische Gruppen-ID oder keinen *Gruppenname* handelt, ist der Zugriffssteuerungslisten-Eintragstyp ungültig.

In der folgenden Tabelle sind die Zugriffsrechte für Zugriffssteuerungslisten beschrieben.

TABELLE 8–2 Zugriffssteuerungslisten-Zugriffsrechte

Zugriffsrecht	Kurzform	Beschreibung
add_file	w	Berechtigung zum Hinzufügen neuer Dateien zu einem Verzeichnis.
add_subdirectory	p	Berechtigung zum Erstellen von Unterverzeichnissen in einem Verzeichnis.
append_data	p	Platzhalter. Gegenwärtig nicht implementiert.
delete	d	Berechtigung zum Löschen einer Datei.
delete_child	D	Berechtigung zum Löschen einer Datei bzw. eines Verzeichnisses in einem Verzeichnis.
execute	x	Berechtigung zum Ausführen einer Datei bzw. Durchsuchen eines Verzeichnisses.
list_directory	r	Berechtigung zum Auflisten des Inhalts eines Verzeichnisses.
read_acl	c	Berechtigung zum Lesen der Zugriffssteuerungsliste (`ls`).
read_attributes	a	Berechtigung zum Lesen grundlegender (nicht zu Zugriffssteuerungslisten gehörender) Attribute einer Datei. Grundlegende Attribute sind Attribute auf der stat-Ebene. Durch Setzen dieses Zugriffsmaskenbits kann ein Objekt `ls`(1) und `stat`(2) ausführen.
read_data	r	Berechtigung zum Lesen des Inhalts einer Datei.

TABELLE 8–2 Zugriffssteuerungslisten-Zugriffsrechte *(Fortsetzung)*

Zugriffsrecht	Kurzform	Beschreibung
read_xattr	R	Berechtigung zum Lesen der erweiterten Attribute einer Datei bzw. Durchsuchen des Verzeichnisses erweiterter Dateiattribute.
synchronize	s	Platzhalter. Gegenwärtig nicht implementiert.
write_xattr	W	Berechtigung zum Erstellen erweiterter Attribute bzw. Schreiben in das Verzeichnis erweiterter Attribute. Durch Gewähren dieses Zugriffsrechtes kann ein Benutzer für eine Datei ein Verzeichnis erweiterter Attribute erstellen. Die Zugriffsrechte der Attributdatei legen das Zugriffsrecht des Benutzers auf das Attribut fest.
write_data	w	Berechtigung zum Ändern oder Ersetzens des Inhalts einer Datei.
write_attributes	A	Berechtigung zum Setzen der Zeitmarken einer Datei bzw. eines Verzeichnisses auf einen beliebigen Wert.
write_acl	C	Berechtigung zum Schreiben bzw. Ändern der Zugriffsteuerungliste mithilfe des Befehls `chmod`.
write_owner	o	Berechtigung zum Ändern des Eigentümers bzw. der Gruppe einer Datei bzw. Berechtigung zum Ausführen der Befehle `chown` oder `chgrp` für die betreffende Datei. Berechtigung zum Übernehmen der Eigentümerschaft einer Datei bzw. zum Ändern der ihrer Gruppe, zu der der betreffende Benutzer gehört. Wenn Sie die Benutzer- bzw. Gruppeneigentümerschaft auf einen beliebigen Benutzer bzw. eine beliebige Gruppe setzen wollen, ist das Zugriffsrecht `PRIV_FILE_CHOWN` erforderlich.

Vererbung von Zugriffssteuerungslisten

Der Zweck der Vererbung von Zugriffssteuerungslisten besteht darin, dass neu erstellte Dateien bzw. Verzeichnisse die vorgesehenen Zugriffssteuerungslisten erben, ohne dass die vorhandenen Zugriffsrecht-Bits des übergeordneten Verzeichnisses ignoriert werden.

Standardmäßig werden Zugriffssteuerungslisten nicht weitergegeben. Wenn Sie für ein Verzeichnis eine komplexe Zugriffssteuerungsliste setzen, wird diese nicht an untergeordnete Verzeichnisse vererbt. Sie müssen die Vererbung einer Zugriffssteuerungsliste für Dateien bzw. Verzeichnisse explizit angeben.

In der folgenden Tabelle sind die optionalen Vererbungsflags aufgeführt.

TABELLE 8–3 Vererbungsflags von Zugriffssteuerungslisten

Vererbungsflag	Kurzform	Beschreibung
`file_inherit`	`f`	Die Zugriffssteuerungsliste wird nur vom übergeordneten Verzeichnis an die Dateien des betreffenden Verzeichnisses vererbt.
`dir_inherit`	`d`	Die Zugriffssteuerungsliste wird nur vom übergeordneten Verzeichnis an die untergeordneten Verzeichnisse des betreffenden Verzeichnisses vererbt.
`inherit_only`	`i`	Die Zugriffssteuerungsliste wird vom übergeordneten Verzeichnis vererbt, gilt jedoch nur für neu erstellte Dateien bzw. Unterverzeichnisse, aber nicht für das betreffende Verzeichnis selbst. Für dieses Flag ist das Flag `file_inherit` bzw. `dir_inherit` (oder beide) erforderlich. Diese legen fest, was vererbt wird.
`no_propagate`	`n`	Die Zugriffssteuerungsliste wird vom übergeordneten Verzeichnis an die erste Hierarchieebene des betreffenden Verzeichnisses und nicht an untergeordnete Ebenen vererbt. Für dieses Flag ist das Flag `file_inherit` bzw. `dir_inherit` (oder beide) erforderlich. Diese legen fest, was vererbt wird.
-	entf.	Zugriff nicht gewährt.

Darüber hinaus können Sie mithilfe der Dateisystemeigenschaft `aclinherit` für das Dateisystem mehr oder weniger strenge Vererbungsrichtlinien für Zugriffssteuerungslisten festlegen. Weitere Informationen finden Sie im folgenden Abschnitt.

Eigenschaftsmodi von Zugriffssteuerungslisten

Das ZFS-Dateisystem enthält zwei Eigenschaftsmodi für Zugriffssteuerungslisten:

- `aclinherit` – Diese Eigenschaft legt das Verhalten der Vererbung von Zugriffssteuerungslisten fest und kann die folgenden Werte annehmen:
  - `discard` – Für neu erstellte Objekte werden beim Erstellen von Dateien und Verzeichnissen keine Zugriffssteuerungslisteneinträge vererbt. Die Zugriffssteuerungsliste der Datei bzw. des Verzeichnisses entspricht dem Zugriffsrechtsmodus dieser Datei bzw. dieses Verzeichnisses.
  - `noallow` – Bei neuen Objekten werden nur vererbbare Zugriffssteuerungslisteneinträge mit Zugriffstyp `deny` vererbt.
  - `restricted` – Für neu erstellte Objekte werden beim Vererben von Zugriffssteuerungslisteneinträgen die Zugriffsrechte `write_owner` und `write_acl` entfernt.

- `passthrough` – Ist der Eigenschaftswert auf `passthrough` gesetzt, werden die Dateien mit einem durch die vererbbaren Zugriffssteuerungseinträge festgelegten Modus erstellt. Wenn keine den Modus betreffenden vererbbaren Zugriffssteuerungseinträge vorhanden sind, wird ein mit der Forderung der Anwendung vereinbarter Modus gesetzt.
- `passthrough-x` – Hat die gleiche Semantik wie `passthrough`, mit der Ausnahme, dass bei Aktivierung von `passthrough-x` Dateien mit der Ausführungsberechtigung (`x`) erstellt werden, jedoch nur, wenn die Ausführungsberechtigung im Dateierstellungsmodus und in einer vererbbaren Zugriffssteuerungsliste festgelegt wurde, die sich auf den Modus auswirkt.

Der Standardmodus für `aclinherit` ist `restricted`.

- `aclmode` – Diese Eigenschaft ändert das Zugriffssteuerungslistenverhalten, wenn der Datei- bzw. Verzeichnismodus vom Befehl `chmod` geändert bzw. eine Datei neu erstellt wird, und kann die folgenden Werte annehmen:
  - `discard` – Alle Zugriffssteuerungslisteneinträge außer denen, die den Modus der Datei bzw. des Verzeichnisses definieren, werden entfernt.
  - `groupmask` – Zugriffssteuerungslisten-Zugriffsrechte für Benutzer bzw. Gruppen werden eingeschränkt, sodass sie nicht größer werden als durch die Gruppenzugriffsrecht-Bits festgelegt, es sei denn, es handelt sich um einen Benutzereintrag mit der gleichen Benutzer-ID wie der Eigentümer der Datei bzw. des Verzeichnisses. Dann werden die Zugriffssteuerungslisten-Zugriffsrechte eingeschränkt, sodass sie nicht größer werden als durch die Zugriffsrecht-Bits des Eigentümers gesetzt.
  - `passthrough` – Während einer `chmod`-Operation werden Zugriffssteuerungseinträge außer `owner@`, `group@` oder `everyone@` nicht geändert. Zugriffssteuerungseinträge mit `owner@`, `group@` oder `everyone@` werden deaktiviert, um den Dateimodus wie durch die `chmod`-Operation gefordert zu setzen.

  Der Standardmodus für `aclmode` ist `groupmask`.

Setzen von Zugriffssteuerungslisten an ZFS-Dateien

In ZFS bestehen Zugriffssteuerungslisten aus einer Folge von Zugriffssteuerungslisteneinträgen. ZFS bietet ein *reines* Zugriffssteuerungslistenmodell, in dem alle Dateien eine Zugriffssteuerungsliste besitzen. Normalerweise sind solche Zugriffssteuerungslisten *gewöhnlich*, da sie nur die herkömmlichen UNIX-Einträge `owner/group/other` repräsentieren.

ZFS-Dateien besitzen zwar trotzdem Zugriffsrecht-Bits und einen Zugriffsmodus files, diese stellen jedoch einen „Speicher" des Inhalts der entsprechenden Zugriffssteuerungsliste dar. Wenn Sie also die Zugriffsrechte einer Datei ändern, wird die entsprechende

Zugriffssteuerungsliste automatisch aktualisiert. Darüber hinaus kann nach dem Entfernen einer komplexen Zugriffssteuerungsliste, mit der ein Benutzer Zugriff auf eine Datei bzw. ein Verzeichnis hatte, wegen der Zugriffsrecht-Bits dieser Datei bzw. dieses Verzeichnisses, die einen Zugriff für alle Benutzer festlegten, dieser Benutzer unter Umständen noch immer Zugriff auf diese Datei bzw. dieses Verzeichnis haben. Alle Entscheidungen in Sachen Zugriffskontrolle werden von den Zugriffsrechten der Zugriffssteuerungsliste dieser Datei bzw. dieses Verzeichnisses festgelegt.

Die grundlegenden Regeln zum Zugriff auf ZFS-Dateien mithilfe von Zugriffssteuerungslisten sind wie folgt:

- ZFS verarbeitet Zugriffssteuerungslisteneinträge in der Reihenfolge, wie sie in der Zugriffssteuerungsliste aufgeführt sind (d. h. von oben nach unten).
- Es werden nur Zugriffssteuerungslisteneinträge berücksichtigt, deren Benutzer-ID mit der ID des Zugriff anfordernden Benutzer übereinstimmt.
- Wenn ein Zugriffsrecht einmal gewährt wurde, kann es durch nachfolgende Zugriffssteuerungslisteneinträge in der gleichen Zugriffssteuerungsliste nicht mehr rückgängig gemacht werden.
- Eigentümern von Dateien wird stets bedingungslose das Zugriffsrecht `write_acl` auch dann gewährt, wenn es in der Zugriffssteuerungsliste explizit verweigert wird. Nicht angegebene Zugriffsrechte werden verweigert.

  Falls Zugriffsrechte verweigert werden bzw. Angaben zum betreffenden Zugriffsrecht fehlen, legen die Zugriffsrechte des betreffenden untergeordneten Systems fest, welche Rechte dem Eigentümer einer Datei bzw. dem Superuser gewährt werden. Dadurch wird verhindert, dass der Zugriff für Dateieigentümernicht gesperrt wird und Superuser Dateien aus Gründen der Wiederherstellung ändern können.

Wenn Sie für ein Verzeichnis eine komplexe Zugriffssteuerungsliste setzen, wird diese nicht automatisch an untergeordnete Verzeichnisse dieses Verzeichnisses vererbt. Wenn Sie für ein Verzeichnis eine komplexe Zugriffssteuerungsliste setzen und diese an die untergeordnete Verzeichnisse dieses Verzeichnisses vererbt werden soll, müssen Sie die Zugriffssteuerungslisten-Vererbungsflags verwenden. Weitere Informationen finden Sie in Tabelle 8–3 und unter „Festlegen der Vererbung von Zugriffssteuerungslisten an ZFS-Dateien im ausführlichen Format“ auf Seite 250.

Erstellen einer neuen Datei und (je nach dem Wert von `umask`) einer gewöhnlichen Standard-Zugriffssteuerungsliste ähnlich wie im folgenden Beispiel:

```
$ ls -v file.1
-r--r--r--   1 root     root      206663 Aug 31 11:53 file.1
     0:owner@:write_data/append_data/execute:deny
     1:owner@:read_data/write_xattr/write_attributes/write_acl/write_owner
         :allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
```

```
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Bitte beachten Sie, dass in diesem Beispiel jede Benutzerkategorie (`owner@`, `group@`, `everyone@`) zwei Zugriffssteuerungslisteneinträge besitzt. Ein Eintrag gilt für die Verweigerung (`deny`), der andere zum Gewähren (`allow`) von Zugriffsrechten.

Im Folgenden finden Sie eine Beschreibung dieser Datei-Zugriffssteuerungsliste:

`0:owner@` Dem Eigentümer werden Schreib- und Ausführungsrechte für die Datei verweigert (`write_data/append_data/execute:deny`).

`1:owner@` Der Eigentümer kann den Dateiinhalt lesen und ändern (`read_data/write_data/append_data`). Der Eigentümer kann darüber hinaus auch Dateiattribute wie z. B. Zeitmarken, erweiterte Attribute und Zugriffssteuerungslisten ändern (`write_xattr/write_attributes /write_acl`). Zusätzlich dazu kann der Eigentümer die Datei-Eigentümerschaft ändern (`write_owner:allow`).

`2:group@` Der Gruppe wird das Ausführen der Datei verweigert (`write_data/append_data/execute:deny`).

`3:group@` Der Gruppe wird die Berechtigung zum Lesen der Datei gewährt (`read_data:allow`).

`4:everyone@` Allen übrigen Benutzern und Gruppen wird die Berechtigung zum Ausführen und Ändern des Inhalts der Datei bzw. Ändern von Dateiattributen verweigert (`write_data/append_data/write_xattr/execute/ write_attributes/write_acl/write_owner:deny`).

`5:everyone@` Allen übrigen Benutzern und Gruppen wird die Berechtigung zum Lesen der Datei bzw. der Dateiattribute gewährt (`read_data/read_xattr/read_attributes/read_acl/ synchronize:allow`). Das Zugriffsrecht `synchronize` ist zurzeit nicht implementiert.

Erstellen eines neuen Verzeichnisses und (je nach dem Wert von `umask`) einer standardmäßigen Verzeichnis-Zugriffssteuerungsliste ähnlich wie im folgenden Beispiel:

```
$ ls -dv dir.1
drwxr-xr-x   2 root     root           2 Aug 31 11:54 dir.1
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data:deny
```

```
3:group@:list_directory/read_data/execute:allow
4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
    /write_attributes/write_acl/write_owner:deny
5:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
    /read_acl/synchronize:allow
```

Im Folgenden finden Sie eine Beschreibung dieser Verzeichnis-Zugriffssteuerungsliste:

`0:owner@` Die Verweigerungsliste für den Eigentümer ist leer (`::deny`).

`1:owner@` Der Eigentümer kann den Inhalt des Verzeichnisses lesen und ändern (`list_directory/read_data/add_file/write_data/add_subdirectory/append_data`), durchsuchen (`execute`) und die Dateiattribute wie z. B. Zeitmarken, erweiterte Attribute und Zugriffssteuerungslisten ändern (`write_xattr/write_attributes/write_acl`). Zusätzlich dazu kann der Eigentümer die Verzeichnis-Eigentümerschaft ändern (`write_owner:allow`).

`2:group@` Die Gruppe kann den Verzeichnisinhalt weder erweitern noch ändern (`add_file/write_data/add_subdirectory/append_data :deny`).

`3:group@` Die Gruppe kann den Verzeichnisinhalt auflisten und lesen. Darüber hinaus kann die Gruppe kann den Verzeichnisinhalt auch durchsuchen (`list_directory/read_data/execute:allow`).

`4:everyone@` Allen übrigen Benutzern und Gruppen wird die Berechtigung zum Erweitern bzw. Änderns des Verzeichnisinhalts verweigert (`add_file/write_data/add_subdirectory/append_data:deny`). Darüber hinaus wird auch die Berechtigung zum Ändern von Verzeichnisattributen verweigert. (`write_xattr /write_attributes/write_acl/write_owner:deny`).

`5:everyone@` Allen übrigen Benutzern und Gruppen wird die Berechtigung zum Lesen und Durchsuchen des Verzeichnisinhalts und der Verzeichnisattribute gewährt (`list_directory/read_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow`). Das Zugriffsrecht `synchronize` ist zurzeit nicht implementiert.

Setzen und Anzeigen von Zugriffssteuerungslisten an ZFS-Dateien im ausführlichen Format

Mit dem Befehl `chmod` können Sie Zugriffssteuerungslisten von ZFS-Dateien ändern. Die folgende `chmod`-Syntax zum Ändern von Zugriffssteuerungslisten nutzt zur Erkennung des

Zugriffssteuerungslistenformats die *Zugriffssteuerungslisten-Spezifikation.* Eine Beschreibung der *Zugriffssteuerungslisten-Spezifikation* finden Sie unter „Syntaxbeschreibungen zum Setzen von Zugriffssteuerungslisten“ auf Seite 235.

- Hinzufügen von Zugriffssteuerungslisten-Eintragstypen
  - Hinzufügen eines Zugriffssteuerungslisten-Eintrags für einen Benutzer

    ```
    % chmod A+acl-specification filename
    ```

  - Hinzufügen eines Zugriffssteuerungslisten-Eintrags nach *Index-ID*

    ```
    % chmod Aindex-ID+acl-specification filename
    ```

    Mit dieser Syntax wird der neue Zugriffssteuerungslisten-Eintrag an der entspechenden *Index-ID* eingefügt.
- Ersetzen eines Zugriffssteuerungslisten-Eintrags

  ```
  % chmod A=acl-specification filename
  ```

  ```
  % chmod Aindex-ID=acl-specification filename
  ```
- Entfernen von Zugriffssteuerungslisten-Einträgen
  - Entfernen eines Zugriffssteuerungslisten-Eintrags nach *Index-ID*

    ```
    % chmod Aindex-ID- filename
    ```

  - Entfernen eines Zugriffssteuerungslisten-Eintrags nach Benutzername

    ```
    % chmod A-acl-specification filename
    ```

  - Entfernen aller komplexen Zugriffssteuerungslisten-Einträge aus einer Datei

    ```
    % chmod A- filename
    ```

Ausführliche Informationen zu Zugriffssteuerungslisten werden mithilfe des Befehls `ls - v` angezeigt. Beispiel:

```
# ls -v file.1
-rw-r--r--   1 root     root      206663 Aug 31 11:53 file.1
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Informationen zur Verwendung des Zugriffssteuerungslisten-Kompaktformats finden Sie unter „Setzen und Anzeigen von Zugriffssteuerungslisten an ZFS-Dateien im Kompaktformat“ auf Seite 258.

BEISPIEL 8–1 Ändern gewöhnlicher Zugriffssteuerungslisten an ZFS-Dateien

Dieser Abschnitt enthält Beispiele zum Setzen und Anzeigen gewöhnlicher Zugriffssteuerungslisten.

Im folgenden Beispiel besitzt `Datei.1` eine gewöhnliche Zugriffssteuerungsliste:

```
# ls -v file.1
-rw-r--r--   1 root     root      206663 Aug 31 11:53 file.1
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Im folgenden Beispiel wird die Berechtigung `write_data` für `group@` gewährt.

```
# chmod A2=group@:append_data/execute:deny file.1
# chmod A3=group@:read_data/write_data:allow file.1
# ls -v file.1
-rw-rw-r--   1 root     root      206663 Aug 31 11:53 file.1
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:append_data/execute:deny
     3:group@:read_data/write_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Im folgenden Beispiel werden die Berechtigungen an `Datei.1` auf 644 zurückgesetzt.

```
# chmod 644 file.1
# ls -v file.1
-rw-r--r--   1 root     root      206663 Aug 31 11:53 file.1
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
```

BEISPIEL 8–1 Ändern gewöhnlicher Zugriffssteuerungslisten an ZFS-Dateien *(Fortsetzung)*

```
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

BEISPIEL 8–2 Setzen komplexer Zugriffssteuerungslisten an ZFS-Dateien

Dieser Abschnitt enthält Beispiele zum Setzen und Anzeigen komplexer Zugriffssteuerungslisten.

Im folgenden Beispiel werden die Berechtigungen `read_data/execute` für den Benutzer `gozer` und das·Verzeichnis `Test.Verz` gesetzt.

```
# chmod A+user:gozer:read_data/execute:allow test.dir
# ls -dv test.dir
drwxr-xr-x+  2 root     root           2 Aug 31 12:02 test.dir
     0:user:gozer:list_directory/read_data/execute:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Im folgenden Beispiel werden die Berechtigungen `read_data/execute` für den Benutzer `gozer` entfernt.

```
# chmod A0- test.dir
# ls -dv test.dir
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data:deny
     3:group@:list_directory/read_data/execute:allow
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
```

BEISPIEL 8–2 Setzen komplexer Zugriffssteuerungslisten an ZFS-Dateien *(Fortsetzung)*

```
        /read_acl/synchronize:allow
```

BEISPIEL 8–3 Interaktion von Zugriffssteuerungslisten mit Berechtigungen an ZFS-Dateien

Diese Beispiele demonstrieren die Interaktion zwischen dem Setzen von Zugriffssteuerungslisten und dem anschließenden Ändern der Berechtigungs-Bits einer Datei bzw. eines Verzeichnisses.

Im folgenden Beispiel besitzt `Datei.2` eine gewöhnliche Zugriffssteuerungsliste:

```
# ls -v file.2
-rw-r--r--   1 root     root        2836 Aug 31 12:06 file.2
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Im folgenden Beispiel werden gewährte Zugriffssteuerungslisten-Berechtigungen vom Benutzer `everyone@` entfernt.

```
# chmod A5- file.2
# ls -v file.2
-rw-r-----+  1 root     root        2836 Aug 31 12:06 file.2
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
```

In dieser Ausgabe werden die Dateiberechtigungs-Bits von 655 auf 650 zurückgesetzt Leseberechtigungen für `everyone@` werden faktisch von den Dateiberechtigungs-Bits entfernt, wenn die gewährten Zugriffssteuerungslisten-Zugriffsrechte für `everyone@` entfernt werden.

Im folgenden Beispiel wird die vorhandene Zugriffssteuerungsliste mit den Zugriffsrechten `read_data/write_data` für `everyone@` ersetzt.

BEISPIEL 8–3 Interaktion von Zugriffssteuerungslisten mit Berechtigungen an ZFS-Dateien *(Fortsetzung)*

```
# chmod A=everyone@:read_data/write_data:allow file.3
# ls -v file.3
-rw-rw-rw-+  1 root     root        2455 Aug 31 12:08 file.3
     0:everyone@:read_data/write_data:allow
```

In dieser Ausgabe ersetzt die `chmod`-Syntax faktisch die Zugriffssteuerungsliste mit den Berechtigungen `read_data/write_data:allow` durch Lese- und Schreibberechtigungen für Eigentümer, Gruppen und `everyone@`. In diesem Modell gibt `everyone@` den Zugriff für beliebige Benutzer bzw. Gruppen an. Da keine Zugriffssteuerungslisteneinträge für `owner@` bzw. `group@` existieren, die die Zugriffsrechte für Eigentümer und Gruppen überschreiben könnten, werden die Berechtigungs-Bits auf 666 gesetzt.

Im folgenden Beispiel wird die vorhandene Zugriffssteuerungsliste mit Leseberechtigungen für den Benutzer `gozer` ersetzt.

```
# chmod A=user:gozer:read_data:allow file.3
# ls -v file.3
----------+  1 root     root        2455 Aug 31 12:08 file.3
     0:user:gozer:read_data:allow
```

In diesem Beispiel werden die Dateizugriffsrechte mit 000 berechnet, da für `owner@`, `group@` bzw. `everyone@` keine Zugriffssteuerungslisteneinträge vorhanden sind, die die herkömmlichen Berechtigungskomponenten einer Datei repräsentieren. Der Dateieigentümer kann dieses Problem durch Zurücksetzen der Zugriffsrechte (und der Zugriffssteuerungsliste) wie folgt beheben:

```
# chmod 655 file.3
# ls -v file.3
-rw-r-xr-x+  1 root     root        2455 Aug 31 12:08 file.3
     0:user:gozer::deny
     1:user:gozer:read_data:allow
     2:owner@:execute:deny
     3:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     4:group@:write_data/append_data:deny
     5:group@:read_data/execute:allow
     6:everyone@:write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:deny
     7:everyone@:read_data/read_xattr/execute/read_attributes/read_acl
         /synchronize:allow
```

BEISPIEL 8–4 Wiederherstellen gewöhnlicher Zugriffssteuerungslisten an ZFS-Dateien

Mit dem Befehl `chmod` können Sie alle komplexen Zugriffssteuerungslisten einer Datei bzw. eines Verzeichnisses entfernen.

Im folgenden Beispiel besitzt `test5.dir` zwei komplexe Zugriffssteuerungslisten.

```
# ls -dv test5.dir
drwxr-xr-x+  2 root     root           2 Aug 31 12:11 test5.dir
     0:user:lp:read_data:file_inherit:deny
     1:user:gozer:read_data:file_inherit:deny
     2:owner@::deny
     3:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     4:group@:add_file/write_data/add_subdirectory/append_data:deny
     5:group@:list_directory/read_data/execute:allow
     6:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     7:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Im folgenden Beispiel werden die komplexen Zugriffsteuerungslisten für die Benutzer `gozer` und `lp` entfernt. Die verbleibende Zugriffsteuerungsliste enthält de sechs Standardwerte für `owner@`, `group@` und `everyone@`.

```
# chmod A- test5.dir
# ls -dv test5.dir
drwxr-xr-x   2 root     root           2 Aug 31 12:11 test5.dir
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data:deny
     3:group@:list_directory/read_data/execute:allow
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Festlegen der Vererbung von Zugriffssteuerungslisten an ZFS-Dateien im ausführlichen Format

Sie können festlegen, ob Zugriffsteuerungslisten von Dateien und Verzeichnissen vererbt werden sollen oder nicht. Standardmäßig werden Zugriffssteuerungslisten nicht weitergegeben. Wenn Sie für ein Verzeichnis eine komplexe Zugriffssteuerungsliste setzen, wird diese nicht an untergeordnete Verzeichnisse vererbt. Sie müssen die Vererbung einer Zugriffssteuerungsliste für Dateien oder Verzeichnisse explizit angeben.

Darüber hinaus werden zwei Eigenschaften von Zugriffssteuerungslisten bereitgestellt, die in Dateisystemen global gesetzt werden können: `aclinherit` und `aclmode`. Standardmäßig ist `aclinherit` auf `restricted` und `aclmode` auf `groupmask` gesetzt .

Weitere Informationen dazu finden Sie in „Vererbung von Zugriffssteuerungslisten“ auf Seite 238.

BEISPIEL 8–5 Gewähren der Standardvererbung von Zugriffssteuerungslisten

Standardmäßig werden Zugriffssteuerungslisten nicht durch eine Verzeichnisstruktur weitergegeben.

Im folgenden Beispiel wird eine komplexe Zugriffssteuerungsliste mit den Zugriffsrechten `read_data/write_data/execute` für den Benutzer `gozer` und das·Verzeichnis `test.dir` gesetzt.

```
# chmod A+user:gozer:read_data/write_data/execute:allow test.dir
# ls -dv test.dir
drwxr-xr-x+  2 root     root           2 Aug 31 13:02 test.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Beim Erstellen eines Unterverzeichnisses in `test.dir` wird die Zugriffssteuerungsliste für den Benutzer `gozer` nicht weitergegeben. Der Benutzer `gozer` hätte nur dann Zugriff auf das Unterverzeichnis `sub.dir`, wenn ihm die Berechtigungen für `sub.dir` Zugriff als Dateieigentümer, Gruppenmitglied oder `everyone@` gewähren würden.

BEISPIEL 8–5 Gewähren der Standardvererbung von Zugriffssteuerungslisten *(Fortsetzung)*

```
# mkdir test.dir/sub.dir
# ls -dv test.dir/sub.dir
drwxr-xr-x   2 root     root           2 Aug 31 13:26 test.dir/sub.dir
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data:deny
     3:group@:list_directory/read_data/execute:allow
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

BEISPIEL 8–6 Gewähren der Vererbung von Zugriffssteuerungslisten an Dateien und Verzeichnissen

In dieser Beispielfolge sind die Zugriffssteuerungslisten für Dateien und Verzeichnisse aufgeführt, die beim Setzen des Flags `file_inherit` angewendet werden.

Im folgenden Beispiel werden die Zugriffsrechte `read_data/write_data` für den Benutzer `gozer` und das·Verzeichnis `test.dir` hinzugefügt, sodass dieser Benutzer Leseberechtigung für neu erstellte Dateien besitzt.

```
# chmod A+user:gozer:read_data/write_data:file_inherit:allow test2.dir
# ls -dv test2.dir
drwxr-xr-x+  2 root     root           2 Aug 31 13:26 test2.dir
     0:user:gozer:read_data/write_data:file_inherit:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Im folgenden Beispiel werden die Zugriffsrechte des Benutzers `gozer` auf die neu erstellte Datei `test2.dir/file.2` angewendet. Durch die gewährte Zugriffssteuerungslisten-Vererbung `read_data:file_inherit:allow` hat der Benutzer `gozer` Leseberechtigung für den Inhalt neu erstellter Dateien.

BEISPIEL 8–6 Gewähren der Vererbung von Zugriffssteuerungslisten an Dateien und Verzeichnissen *(Fortsetzung)*

```
# touch test2.dir/file.2
# ls -v test2.dir/file.2
-rw-r--r--+  1 root     root           0 Aug 31 13:27 test2.dir/file.2
     0:user:gozer:write_data:deny
     1:user:gozer:read_data/write_data:allow
     2:owner@:execute:deny
     3:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     4:group@:write_data/append_data/execute:deny
     5:group@:read_data:allow
     6:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     7:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Da die Eigenschaft `aclmode` für diese Datei auf den Standardwert `groupmask` gesetzt ist, hat der Benutzer `gozer` für die Datei `file.2` nicht das Zugriffsrecht `write_data`, da es die Gruppenberechtigung der Datei nicht zulässt.

Bitte beachten Sie, dass das Zugriffsrecht `inherit_only`, das beim Setzen der Flags `file_inherit` bzw. `dir_inherit` angewendet wird, zum Weitergeben der Zugriffssteuerungsliste durch die Verzeichnisstruktur dient. Somit werden dem Benutzer `gozer` nur Zugriffsrechte für `everyone@` gewährt bzw. verweigert, wenn er nicht Eigentümer der betreffenden Datei ist bzw. nicht zur Eigentümergruppe gehört. Beispiel:

```
# mkdir test2.dir/subdir.2
# ls -dv test2.dir/subdir.2
drwxr-xr-x+  2 root     root           2 Aug 31 13:28 test2.dir/subdir.2
     0:user:gozer:list_directory/read_data/add_file/write_data:file_inherit
         /inherit_only:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

In der folgenden Beispielfolge sind die Zugriffssteuerungslisten für Dateien und Verzeichnisse aufgeführt, die beim Setzen des Flags `file_inherit` bzw. `dir_inherit` angewendet werden.

BEISPIEL 8–6 Gewähren der Vererbung von Zugriffssteuerungslisten an Dateien und Verzeichnissen *(Fortsetzung)*

Im folgenden Beispiel werden dem Benutzer `gozer` Lese-, Schreib- und Ausführungsberechtigungen gewährt, die für neu erstellte Dateien und Verzeichnisse vererbt wurden.

```
# chmod A+user:gozer:read_data/write_data/execute:file_inherit/dir_inherit:allow
test3.dir
# ls -dv test3.dir
drwxr-xr-x+  2 root     root           2 Aug 31 13:29 test3.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute
         :file_inherit/dir_inherit:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow

# touch test3.dir/file.3
# ls -v test3.dir/file.3
-rw-r--r--+  1 root     root           0 Jun 20 14:42 test3.dir/file.3
     0:user:gozer:write_data/execute:deny
     1:user:gozer:read_data/write_data/execute:allow
     2:owner@:execute:deny
     3:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     4:group@:write_data/append_data/execute:deny
     5:group@:read_data:allow
     6:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     7:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow

# mkdir test3.dir/subdir.1
# ls -dv test3.dir/subdir.1
drwxr-xr-x+  2 root     root           2 Aug 31 13:32 test3.dir/subdir.1
     0:user:gozer:list_directory/read_data/add_file/write_data/execute
         :file_inherit/dir_inherit/inherit_only:allow
     1:user:gozer:add_file/write_data:deny
     2:user:gozer:list_directory/read_data/add_file/write_data/execute:allow
     3:owner@::deny
```

BEISPIEL 8–6 Gewähren der Vererbung von Zugriffssteuerungslisten an Dateien und Verzeichnissen *(Fortsetzung)*

```
     4:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     5:group@:add_file/write_data/add_subdirectory/append_data:deny
     6:group@:list_directory/read_data/execute:allow
     7:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     8:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Da die Zugriffsrecht-Bits des übergeordneten Verzeichnisses für `group@` und `everyone@` Schreib- und Ausführungsberechtigungen verweigern, wird in diesem Beispiel dem Benutzer `gozer` ebenfalls die Schreib- und Ausführungsberechtigung verweigert. Der Standardwert der Eigenschaft `aclmode` ist `restricted`, was bedeutet, dass die Zugriffsrechte `write_data` und `execute` nicht vererbt werden.

Im folgenden Beispiel werden dem Benutzer `gozer` Lese-, Schreib- und Ausführungsberechtigungen gewährt, die für neu erstellte Dateien und Verzeichnisse vererbt wurden, aber nicht an die untergeordnete Dateien des Verzeichnisses weitergegeben werden.

```
# chmod A+user:gozer:read_data/write_data/execute:file_inherit/no_propagate:allow
test4.dir
# ls -dv test4.dir
drwxr-xr-x+  2 root     root           2 Aug 31 13:34 test4.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute
         :file_inherit/no_propagate:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Wie das folgende Beispiel zeigt, werden beim Erstellen eines neuen Unterverzeichnisses die Zugriffsrechte `read_data/write_data/execute` für den Benutzer `gozer` nicht an das neue Verzeichnis `sub4.dir` weitergegeben.

```
mkdir test4.dir/sub4.dir
# ls -dv test4.dir/sub4.dir
drwxr-xr-x   2 root     root           2 Aug 31 13:35 test4.dir/sub4.dir
```

BEISPIEL 8–6 Gewähren der Vererbung von Zugriffssteuerungslisten an Dateien und Verzeichnissen *(Fortsetzung)*

```
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/write_xattr/execute/write_attributes/write_acl
         /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data:deny
     3:group@:list_directory/read_data/execute:allow
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
         /write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Wie das folgende Beispiel zeigt, werden die Zugriffsrechte `read_data/write_data/execute` für den Benutzer `gozer` an die neu erstellte Datei weitergegeben.

```
# touch test4.dir/file.4
# ls -v test4.dir/file.4
-rw-r--r--+  1 root     root           0 Aug 31 13:35 test4.dir/file.4
     0:user:gozer:write_data/execute:deny
     1:user:gozer:read_data/write_data/execute:allow
     2:owner@:execute:deny
     3:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     4:group@:write_data/append_data/execute:deny
     5:group@:read_data:allow
     6:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
     7:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

BEISPIEL 8–7 Vererbung von Zugriffssteuerungslisten mit dem Zugriffssteuerungslistenmodus "Pass Through"

Wenn die Eigenschaft `aclmode` des Dateisystems `tank/cindy` auf `passthrough` gesetzt ist, erbt der Benutzer `gozer` die auf `test4.dir` angewendete Zugriffssteuerungsliste für die neu erstellte Datei `file.4` wie folgt:

```
# zfs set aclmode=passthrough tank/cindys
# touch test4.dir/file.4
# ls -v test4.dir/file.4
-rw-r--r--+  1 root     root           0 Aug 31 13:39 test4.dir/file.4
     0:user:gozer:write_data/execute:deny
     1:user:gozer:read_data/write_data/execute:allow
     2:owner@:execute:deny
     3:owner@:read_data/write_data/append_data/write_xattr/write_attributes
```

BEISPIEL 8–7 Vererbung von Zugriffssteuerungslisten mit dem Zugriffssteuerungslistenmodus "Pass Through" *(Fortsetzung)*

```
        /write_acl/write_owner:allow
     4:group@:write_data/append_data/execute:deny
     5:group@:read_data:allow
     6:everyone@:write_data/append_data/write_xattr/execute/write_attributes
        /write_acl/write_owner:deny
     7:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
        :allow
```

Diese Ausgabe zeigt, dass die am übergeordneten Verzeichnis `test4.dir` gesetzte Zugriffssteuerungsliste `read_data/write_data/execute:allow:file_inherit/dir_inherit` an den Benutzer `gozer` weitergegeben wird.

BEISPIEL 8–8 Vererbung von Zugriffssteuerungslisten mit dem Zugriffssteuerungslistenmodus „Discard”

Wenn die Eigenschaft `aclmode` eines Dateisystems auf `discard` gesetzt ist, können Zugriffssteuerungslisten potenziell ignoriert werden, wenn sich die Zugriffsrecht-Bits eines Verzeichnisses ändern. Beispiel:

```
# zfs set aclmode=discard tank/cindys
# chmod A+user:gozer:read_data/write_data/execute:dir_inherit:allow test5.dir
# ls -dv test5.dir
drwxr-xr-x+  2 root     root           2 Aug 31 13:40 test5.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute
        :dir_inherit:allow
     1:owner@::deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
        /append_data/write_xattr/execute/write_attributes/write_acl
        /write_owner:allow
     3:group@:add_file/write_data/add_subdirectory/append_data:deny
     4:group@:list_directory/read_data/execute:allow
     5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
        /write_attributes/write_acl/write_owner:deny
     6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow
```

Wenn Sie später die Zugriffsrechte an einem Verzeichnis einschränken, gelten die komplexen Zugriffssteuerungslisten nicht mehr. Beispiel:

```
# chmod 744 test5.dir
# ls -dv test5.dir
drwxr--r--   2 root     root           2 Aug 31 13:40 test5.dir
     0:owner@::deny
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
```

BEISPIEL 8–8 Vererbung von Zugriffssteuerungslisten mit dem Zugriffssteuerungslistenmodus „Discard" *(Fortsetzung)*

```
        /append_data/write_xattr/execute/write_attributes/write_acl
        /write_owner:allow
     2:group@:add_file/write_data/add_subdirectory/append_data/execute:deny
     3:group@:list_directory/read_data:allow
     4:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
        /execute/write_attributes/write_acl/write_owner:deny
     5:everyone@:list_directory/read_data/read_xattr/read_attributes/read_acl
        /synchronize:allow
```

BEISPIEL 8–9 Vererbung von Zugriffssteuerungslisten mit dem Zugriffssteuerungslistenmodus „Noallow"

Im folgenden Beispiel sind zwei komplexe Zugriffsteuerungslisten mit Dateivererbung gesetzt. Eine Zugriffssteuerungsliste gewährt die Berechtigung `read_data`, und die andere Zugriffssteuerungsliste verweigert die Berechtigung `read_data`. Dieses Beispiel zeigt auch, wie Sie mit dem gleichen Aufruf des Befehls `chmod` zwei Zugriffssteuerungslisten angeben können.

```
# zfs set aclinherit=noallow tank/cindys
# chmod A+user:gozer:read_data:file_inherit:deny,user:lp:read_data:file_inherit:allow
test6.dir
# ls -dv test6.dir
drwxr-xr-x+  2 root     root           2 Aug 31 13:43 test6.dir
     0:user:gozer:read_data:file_inherit:deny
     1:user:lp:read_data:file_inherit:allow
     2:owner@::deny
     3:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
        /append_data/write_xattr/execute/write_attributes/write_acl
        /write_owner:allow
     4:group@:add_file/write_data/add_subdirectory/append_data:deny
     5:group@:list_directory/read_data/execute:allow
     6:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr
        /write_attributes/write_acl/write_owner:deny
     7:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow
```

Wie das folgende Beispiel zeigt, gilt beim Erstellen einer neuen Datei die Zugriffssteuerungsliste, die das Zugriffsrecht `read_data` gewährt, nicht mehr.

```
# touch test6.dir/file.6
# ls -v test6.dir/file.6
-rw-r--r--   1 root     root           0 Aug 31 13:44 test6.dir/file.6
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
        /write_acl/write_owner:allow
```

BEISPIEL 8–9 Vererbung von Zugriffssteuerungslisten mit dem Zugriffssteuerungslistenmodus „Noallow" *(Fortsetzung)*

```
2:group@:write_data/append_data/execute:deny
3:group@:read_data:allow
4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
    /write_acl/write_owner:deny
5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
    :allow
```

Setzen und Anzeigen von Zugriffssteuerungslisten an ZFS-Dateien im Kompaktformat

Sie können Zugriffsrechte an ZFS-Dateien in einem Kompaktformat setzen und anzeigen, das für die einzelnen Berechtigungen 14 eindeutige Buchstaben verwendet. Die Buchstaben zur kompakten Darstellung der Zugriffsrechte sind in Tabelle 8–2 und Tabelle 8–3 aufgeführt.

Kompaktausgaben von Zugriffssteuerungslisten für Dateien und Verzeichnisse werden mithilfe des Befehls `ls - V` angezeigt. Beispiel:

```
# ls -V file.1
-rw-r--r--   1 root     root      206663 Aug 31 13:54 file.1
           owner@:--x-----------:------:deny
           owner@:rw-p---A-W-Co-:------:allow
           group@:-wxp----------:------:deny
           group@:r-------------:------:allow
        everyone@:-wxp---A-W-Co-:------:deny
        everyone@:r-----a-R-c--s:------:allow
```

Die Bedeutung dieser Kompaktausgabe ist wie folgt:

`owner@` Dem Eigentümer wird das Ausführen der Datei verweigert (`x= execute`).

`owner@` Der Eigentümer kann den Dateiinhalt lesen und ändern (`rw=read_data/write_data`), (`p= append_data`). Der Eigentümer kann darüber hinaus auch Dateiattribute wie z. B. Zeitmarken, erweiterte Attribute und Zugriffssteuerungslisten ändern (`A=write_xattr` , `W=write_attributes`, `C= write_acl`). Zusätzlich dazu kann der Eigentümer die Datei-Eigentümerschaft ändern (`o=write_owner`).

`group@` Der Gruppe werden Änderungs- und Ausführungsberechtigung für die Datei verweigert (`write_data`, `p= append_data` und `x=execute`).

`group@` Der Gruppe wird die Berechtigung zum Lesen der Datei gewährt (`r= read_data`).

`everyone@` Allen übrigen Benutzern bzw. Gruppen wird die Berechtigung zum Ausführen und Ändern der Datei sowie Ändern von Dateiattributen verweigert (`w=write_data`, `x= execute`, `p=append_data`, `A=write_xattr`, `W=write_attributes` , `C=write_acl` und `o= write_owner`).

`everyone@` Allen übrigen Benutzern und Gruppen wird die Berechtigung zum Lesen der Datei bzw. der Dateiattribute gewährt (`r=read_data`, `a=append_data`, `R=read_xattr` , `c=read_acl` und `s= synchronize`). Das Zugriffsrecht `synchronize` ist zurzeit nicht implementiert.

Das Kompaktformat von Zugriffssteuerungslisten hat gegenüber dem ausführlichen Format folgende Vorteile:

- Zugriffsrechte können für den Befehl `chmod` als positionale Argumente angegeben werden.
- Der Bindestrich (-), der für die Verweigerung von Zugriffsrechten steht, kann weggelassen werden, und es müssen nur die erforderlichen Buchstaben angegeben werden.
- Zugriffsrechte und Vererbungsflags werden in der gleichen Weise gesetzt.

Informationen zur Verwendung des ausführlichen Formats von Zugriffssteuerungslisten finden Sie unter „Setzen und Anzeigen von Zugriffssteuerungslisten an ZFS-Dateien im ausführlichen Format“ auf Seite 243.

BEISPIEL 8–10 Setzen und Anzeigen von Zugriffssteuerungslisten im Kompaktformat

Im folgenden Beispiel besitzt `file.1` eine gewöhnliche Zugriffssteuerungsliste:

```
# ls -V file.1
-rw-r--r--   1 root     root      206663 Aug 31 13:54 file.1
            owner@:--x-----------:------:deny
            owner@:rw-p---A-W-Co-:------:allow
            group@:-wxp----------:------:deny
            group@:r-------------:------:allow
         everyone@:-wxp---A-W-Co-:------:deny
         everyone@:r-----a-R-c--s:------:allow
```

Im diesem Beispiel werden die Berechtigungen `read_data/execute` für den Benutzer `gozer` an der Datei file.1 hinzugefügt.

```
# chmod A+user:gozer:rx:allow file.1
# ls -V file.1
-rw-r--r--+  1 root     root      206663 Aug 31 13:54 file.1
        user:gozer:r-x-----------:------:allow
            owner@:--x-----------:------:deny
            owner@:rw-p---A-W-Co-:------:allow
            group@:-wxp----------:------:deny
            group@:r-------------:------:allow
```

BEISPIEL 8–10 Setzen und Anzeigen von Zugriffssteuerungslisten im Kompaktformat *(Fortsetzung)*

```
         everyone@:-wxp---A-W-Co-:------:deny
         everyone@:r-----a-R-c--s:------:allow
```

Eine andere Methode zum Hinzufügen der gleichen Zugriffsrechte für den Benutzer `gozer` besteht im Einfügen einer neuen Zugriffssteuerungsliste an einer neuen Position (z. B. Position 4). Somit werden die vorhandenen Zugriffssteuerungslisten an den Positionen 4-6 nach unten verschoben. Beispiel:

```
# chmod A4+user:gozer:rx:allow file.1
# ls -V file.1
-rw-r--r--+  1 root     root      206663 Aug 31 14:05 file.1
           owner@:--xp----------:------:deny
           owner@:rw-----A-W-Co-:------:allow
           group@:-wxp----------:------:deny
           group@:r-------------:------:allow
       user:gozer:r-x-----------:------:allow
         everyone@:-wxp---A-W-Co-:------:deny
         everyone@:r-----a-R-c--s:------:allow
```

Im folgenden Beispiel werden dem Benutzer `gozer` unter Verwendung des Zugriffssteuerungslisten-Kompaktformats Lese-, Schreib- und Ausführungsberechtigungen gewährt, die für neu erstellte Dateien und Verzeichnisse vererbt wurden.

```
# chmod A+user:gozer:rwx:fd:allow dir.2
# ls -dV dir.2
drwxr-xr-x+  2 root     root           2 Aug 28 13:21 dir.2
       user:gozer:rwx-----------:fd----:allow
           owner@:--------------:------:deny
           owner@:rwxp---A-W-Co-:------:allow
           group@:-w-p----------:------:deny
           group@:r-x-----------:------:allow
         everyone@:-w-p---A-W-Co-:------:deny
         everyone@:r-x---a-R-c--s:------:allow
```

Sie können Zugriffsrechte und Vererbungsflags auch aus der Ausgabe des Befehls `ls - V` in das Kompaktformat des Befehls `chmod` kopieren. Wenn Sie beispielsweise die Zugriffsrechte und Vererbungsflags von `dir.2` für den Benutzer `gozer` auf den Benutzer `cindys` an `dir.2` übertragen möchten, müssen Sie die entsprechenden Zugriffsrechte und Vererbungsflags (`rwx-----------:f-----:allow`) nur in die Befehlszeile des Befehls `chmod` kopieren. Beispiel:

```
# chmod A+user:cindys:rwx-----------:fd----:allow dir.2
# ls -dV dir.2
drwxr-xr-x+  2 root     root           2 Aug 28 14:12 dir.2
      user:cindys:rwx-----------:fd----:allow
```

BEISPIEL 8–10 Setzen und Anzeigen von Zugriffssteuerungslisten im Kompaktformat *(Fortsetzung)*

```
        user:gozer:rwx-----------:fd----:allow
            owner@:--------------:------:deny
            owner@:rwxp---A-W-Co-:------:allow
            group@:-w-p----------:------:deny
            group@:r-x----------:------:allow
         everyone@:-w-p---A-W-Co-:------:deny
         everyone@:r-x---a-R-c--s:------:allow
```

BEISPIEL 8–11 Vererbung von Zugriffssteuerungslisten mit dem Zugriffssteuerungslistenvererbungsmodus "Pass Through"

Wenn die Eigenschaft `aclinherit` des Dateisystems auf `passthrough` gesetzt ist, werden alle vererbbaren Zugriffssteuerungslisten ohne jegliche Änderung der Einträge bei der Vererbung weitergegeben. Ist diese Eigenschaft auf `passthrough` gesetzt, werden Dateien mit einem Berechtigungsmodus erstellt, der von der vererbbaren Zugriffssteuerungsliste bestimmt wird. Wenn keine den Berechtigungsmodus betreffenden vererbbaren Zugriffssteuerungseinträge vorhanden sind, wird ein mit der Forderung der Anwendung vereinbarter Berechtigungsmodus gesetzt.

In den folgenden Beispielen wird die Vererbung von Berechtigungs-Bits durch das Setzen des Modus `aclinherit` auf `passthrough` in kompakter Zugriffssteuerungslistensyntax veranschaulicht.

In diesem Beispiel wird eine Zugriffssteuerungsliste für `test1.dir` gesetzt, um die die Vererbung zu erzwingen. Durch die Syntax wird für neu erstellte Dateien ein Zugriffssteuerungseintrag `owner@`, `group@` und `everyone@` erstellt. Neu erstellte Verzeichnisse erben die Zugriffssteuerungseinträge `@owner`, `group@` und `everyone@`. Darüber hinaus erben Verzeichnisse sechs weitere Zugriffssteuerungseinträge, die die Einträge an neu erstellte Verzeichnisse und Dateien weitergeben.

```
# zfs set aclinherit=passthrough tank/cindys
# pwd
/tank/cindys
# mkdir test1.dir

# chmod A=owner@:rwxpcCosRrWaAdD:fd:allow,group@:rwxp:fd:allow,everyone@::fd:allow
test1.dir
# ls -Vd test1.dir
drwxrwx---+  2 root     root           2 Aug 31 14:11 test1.dir
            owner@:rwxpdDaARWcCos:fd----:allow
            group@:rwxp----------:fd----:allow
         everyone@:--------------:fd----:allow
```

BEISPIEL 8–11 Vererbung von Zugriffssteuerungslisten mit dem Zugriffssteuerungslistenvererbungsmodus "Pass Through" *(Fortsetzung)*

In diesem Beispiel erbt eine neu erstellte Datei die Zugriffssteuerungsliste, die für die Weitergabe an neu erstellte Dateien angegeben wurde.

```
# cd test1.dir
# touch file.1
# ls -V file.1
-rwxrwx---+  1 root     root           0 Aug 31 14:14 file.1
            owner@:rwxpdDaARWcCos:------:allow
            group@:rwxp----------:------:allow
         everyone@:--------------:------:allow
```

In diesem Beispiel erbt ein neu erstelltes Verzeichnis sowohl die Zugriffssteuerungseinträge, die den Zugriff auf dieses Verzeichnis regeln, als auch diejenigen für die künftige Weitergabe an untergeordnete Objekte des neu erstellten Verzeichnisses.

```
# mkdir subdir.1
# ls -dV subdir.1
drwxrwx---+  2 root     root           2 Aug 31 14:15 subdir.1
            owner@:rwxpdDaARWcCos:fdi---:allow
            owner@:rwxpdDaARWcCos:------:allow
            group@:rwxp----------:fdi---:allow
            group@:rwxp----------:------:allow
         everyone@:--------------:fdi---:allow
         everyone@:--------------:------:allow
```

Die Einträge `-di--` und `f-i---` gelten für die Weitergabe der Vererbung und werden bei der Zugriffssteuerung nicht berücksichtigt. In diesem Beispiel wird eine Datei mit einer gewöhnlichen Zugriffssteuerungsliste in einem anderen Verzeichnis erstellt, wo keine vererbten Zugriffssteuerungseinträge vorhanden sind.

```
# cd /tank/cindys
# mkdir test2.dir
# cd test2.dir
# touch file.2
# ls -V file.2
-rw-r--r--   1 root     root           0 Aug 31 14:16 file.2
            owner@:--x-----------:------:deny
            owner@:rw-p---A-W-Co-:------:allow
            group@:-wxp----------:------:deny
            group@:r-------------:------:allow
         everyone@:-wxp---A-W-Co-:------:deny
         everyone@:r-----a-R-c--s:------:allow
```

BEISPIEL 8–12 Vererbung von Zugriffssteuerungslisten mit dem Zugriffssteuerungslistenvererbungsmodus "Pass Through-X"

Ist `aclinherit=passthrough-x` aktiviert, so werden Dateien mit der Ausführungsberechtigung (`x`) für `owner@`, `group@` oder `everyone@` erstellt, allerdings nur, wenn die Ausführungsberechtigung im Dateierstellungsmodus und in einem vererbbaren Zugriffssteuerungseintrag, der den Modus betrifft, eingestellt ist.

Das folgende Beispiel zeigt, wie die Ausführungsberechtigung durch Einstellen des Modus `aclinherit` auf `passthrough-x` vererbt wird.

```
# zfs set aclinherit=passthrough-x tank/cindys
```

Die folgende Zugriffssteuerungsliste ist auf `/tank/cindys/test1.dir` gesetzt, um ausführbare Zugriffssteuerungslisten-Vererbung für Dateien für `owner@` bereitzustellen,

```
# chmod A=owner@:rwxpcCosRrWaAdD:fd:allow,group@:rwxp:fd:allow,everyone@::fd:allow test1.dir
# ls -Vd test1.dir
drwxrwx---+  2 root     root           2 Aug 31 15:05 test1.dir
            owner@:rwxpdDaARWcCos:fd----:allow
            group@:rwxp----------:fd----:allow
         everyone@:--------------:fd----:allow
```

Es wird eine Datei (`file1`) mit den angeforderten Zugriffsrechten `0666` erstellt. Die resultierenden Zugriffsrechte sind `0660`. Die Ausführungsberechtigung wurde nicht vererbt, weil der Erstellungsmodus dies nicht fordert.

```
# touch test1.dir/file1
# ls -V test1.dir/file1
-rw-rw----+  1 root     root           0 Aug 31 15:06 test1.dir/file1
            owner@:rw-pdDaARWcCos:------:allow
            group@:rw-p----------:------:allow
         everyone@:--------------:------:allow
```

Als Nächstes wird das Executable `t` unter Verwendung des Compilers `cc` im Verzeichnis `testdir` erstellt.

```
# cc -o t t.c
# ls -V t
-rwxrwx---+  1 root     root        7396 Dec  3 15:19 t
                owner@:rwxpdDaARWcCos:------:allow
                group@:rwxp----------:------:allow
             everyone@:--------------:------:allow
```

Die resultierenden Zugriffsrechte sind `0770`, weil `cc` die Zugriffsrechte `0777` gefordert hat, weswegen die Ausführungsberechtigung von den Einträgen `owner@`, `group@` und `everyone@`

BEISPIEL 8–12 Vererbung von Zugriffssteuerungslisten mit dem Zugriffssteuerungslistenvererbungsmodus "Pass Through-X" *(Fortsetzung)*

geerbt wurde.

◆ ◆ ◆ KAPITEL 9

9 Delegierte ZFS-Administration

In diesem Kapitel wird erläutert, wie mit der delegierten Administration Benutzern ohne ausreichende Zugriffsrechte ermöglicht werden kann, ZFS-Administrationsaufgaben zu erledigen.

Delegierte ZFS-Administration im Überblick

Dieses Leistungsmerkmal dient zum Verteilen fein abgestimmter Zugriffsrechte an bestimmte Benutzer, Gruppen oder an „everyone“. Es werden zwei Arten der delegierten Zugriffsrechte unterstützt:

- Einzelne Zugriffsrechte wie create, destroy, mount, snapshot usw. können ausdrücklich angegeben werden.
- Auch können Gruppen von Zugriffsrechten, so genannte *Zugriffsrechtsätze*, definiert werden. Ein Zugriffsrechtsatz kann zu einem späteren Zeitpunkt aktualisiert werden und die Änderungen wirken sich automatisch auf alle Benutzer des Satzes aus. Zugriffsrechtsätze beginnen mit dem Zeichen @ und sind auf eine Länge von 64 Zeichen begrenzt. Für die auf das Zeichen @ folgenden übrigen Zeichen im Namen gelten dieselben Einschränkungen wie für normale ZFS-Dateisystemnamen.

Die delegierte ZFS-Administration bietet ähnliche Möglichkeiten wie das RBAC-Sicherheitsmodell. Das Modell der delegierten ZFS-Administration stellt für die Verwaltung von ZFS-Speicher-Pools und -Dateisystemen die folgenden Vorteile dar:

- Bei einer Migration des ZFS-Speicher-Pools werden seine Zugriffsberechtigungen mit ihm weitergegeben.

- Es gilt die dynamische Vererbung, sodass gesteuert werden kann, wie die Zugriffsrechte durch die Dateisysteme weitergegeben werden.
- Möglich ist eine Konfiguration, bei der nur der Ersteller eines Dateisystems dieses wieder löschen kann.
- Zugriffsrechte können gezielt an bestimmte Dateisysteme verteilt werden. Neu erstellte Dateisysteme können Zugriffsrechte automatisch übernehmen.
- Dieses Modell ermöglicht eine einfache NFS-Administration. So könnte beispielsweise ein Benutzer mit ausdrücklichen Zugriffsrechten über NFS einen Snapshot im entsprechenden `.zfs/snapshot` -Verzeichnis erstellen.

Die delegierte Administration bietet sich für die Verteilung von ZFS-Aufgaben an. Informationen zum Einsatz von RBAC für allgemeine Solaris-Administrationsaufgaben finden Sie in Teil III, „Roles, Rights Profiles, and Privileges“ in *System Administration Guide: Security Services*.

Deaktivieren von delegierten ZFS-Zugriffsrechten

Die delegierte Administration lässt sich durch Setzen der Eigenschaft `delegation` für den Pool aktivieren bzw. deaktivieren. Beispiel:

```
# zpool get delegation users
NAME  PROPERTY    VALUE       SOURCE
users  delegation  on          default
# zpool set delegation=off users
# zpool get delegation users
NAME  PROPERTY    VALUE       SOURCE
users  delegation  off         local
```

Standardmäßig ist die Eigenschaft `delegation` aktiviert.

Delegieren von ZFS-Zugriffsrechten

Mit dem Befehl `zfs allow` können Sie Nicht-Root-Benutzern Zugriffsrechte für ZFS-Datasets gewähren. Dafür stehen folgende Möglichkeiten zur Verfügung:

- Einzelne Zugriffsrechte können einem Benutzer, einer Gruppe oder global gewährt werden.
- Gruppen von Einzelzugriffsrechten können in Form von *Zugriffsrechtsätzen* ebenfalls an Benutzer, Gruppen oder global vergeben werden.
- Die Gewährung von Zugriffsrechten ist sowohl lokal für das jeweilige Dataset als auch allgemein an alle untergeordneten Objekte des aktuellen Datasets möglich.

In der folgenden Tabelle finden Sie eine Beschreibung der delegierbaren Vorgänge und aller abhängigen Zugriffsrechte, die zum Durchführen des delegierten Vorgangs erforderlich sind.

Zugriffsrecht (Unterbefehl)	Beschreibung	Abhängigkeiten
`allow`	Die Fähigkeit, eigene Zugriffsrechte an andere Benutzer weiterzugeben.	Das Zugriffsrecht, das gewährt werden soll, muss ebenfalls vorhanden sein.
`clone`	Die Fähigkeit, beliebige Snapshots des Datasets zu klonen.	Die Fähigkeiten `create` und `mount` müssen im ursprünglichen Dateisystem vorhanden sein.
`create`	Die Fähigkeit, untergeordnete Datasets zu erstellen.	Die Fähigkeit `mount` muss ebenfalls vorhanden sein.
`destroy`	Die Fähigkeit, ein Dataset zu löschen.	Die Fähigkeit `mount` muss ebenfalls vorhanden sein.
`Einhängen`	Die Fähigkeit, ein Dataset ein- und auszuhängen und Geräteverknüpfungen für Volumes zu erstellen oder zu löschen.	
`promote`	Die Fähigkeit, einen Klon zu einem Dataset zu machen.	Die Fähigkeiten `mount` und `promote` müssen im ursprünglichen Dateisystem vorhanden sein.
`receive`	Die Fähigkeit, mit dem Befehl `zfs receive` untergeordnete Dateisysteme zu erstellen.	Die Fähigkeiten `mount` und `create` müssen ebenfalls vorhanden sein.
`rename`	Die Fähigkeit, ein Dataset umzubenennen.	Die Fähigkeiten `create` und `mount` müssen im neuen übergeordneten Objekt vorhanden sein.
`rollback`	Die Fähigkeit, frühere Zustände aus einem Snapshot wiederherzustellen.	Die Fähigkeit `mount` muss ebenfalls vorhanden sein.
`send`	Die Fähigkeit, einen Snapshot-Datenstream zu senden.	
`share`	Die Fähigkeit, ein Dataset freizugeben und zu sperren.	
`snapshot`	Die Fähigkeit, Snapshots von Datasets herzustellen.	

Sie können die folgenden Zugriffsrechte erteilen, wobei diese auf Zugriff, Lesezugriff oder Bearbeitungszugriff beschränkt sein können:

- `groupquota`
- `groupused`
- `userprop`
- `userquota`
- `userused`

Darüber hinaus können Nicht-Root-Benutzern folgende ZFS-Eigenschaften delegiert werden.

- aclinherit
- aclmode
- atime
- canmount
- casesensitivity
- checksum
- compression
- copies
- devices
- exec
- mountpoint
- nbmand
- normalization
- primarycache
- quota
- readonly
- recordsize
- refreservation
- reservation
- secondarycache
- setuid
- shareiscsi
- sharenfs
- sharesmb
- snapdir
- utf8only
- version
- volblocksize
- volsize
- vscan
- xattr
- zoned

Einige dieser Eigenschaften können nur bei der Erstellung des Datasets gesetzt werden. Eine Beschreibung dieser Eigenschaften finden Sie unter „ZFS-Eigenschaften“ auf Seite 179.

Beschreibung der Syntax zum Delegieren von Zugriffsrechten (`zfs allow`)

Die Syntax für `zfs allow` lautet:

```
# zfs allow -[ldugecs] everyone|user|group[,,...] perm|@setname,...] filesystem| volume
```

Die folgende `zfs allow`-Syntax (fett gedruckt) gibt an, wem die Zugriffsrechte übertragen werden:

`zfs allow [-uge]|user|group|everyone` `[,...]` *filesystem* | *volume*

Mehrere Entitäten können durch Komma getrennt als Liste angegeben werden. Wenn keine `-uge`-Option angegeben ist, werden die Argumente der Reihenfolge nach als das Schlüsselwort `everyone`, dann als Benutzername und schließlich als Gruppenname interpretiert. Um einen Benutzer oder eine Gruppe mit dem Namen „everyone "anzugeben, verwenden Sie die Option `-u` bzw. `-g`. Mit der Option `-g` geben Sie eine Gruppe mit demselben Namen eines Benutzers an. Die Option `-c` gewährt „Create-time"-Zugriffsrechte.

Die folgende `zfs allow`-Syntax (fett gedruckt) veranschaulicht, wie Zugriffsrechte und Zugriffsrechtsätze angegeben werden:

`zfs allow [-s] ... perm|@setname [,...]` *filesystem* | *volume*

Mehrere Zugriffsrechte können durch Komma getrennt als Liste angegeben werden. Die Namen der Zugriffsrechte sind mit den ZFS-Unterbefehlen und -Eigenschaften identisch. Weitere Informationen finden Sie im vorherigen Abschnitt.

Mehrere Zugriffsrechte lassen sich in *Zugriffsrechtsätzen* zusammenfassen. Sie werden durch die Option `-s` gekennzeichnet. Zugriffsrechtsätze können von anderen `zfs allow`-Befehlen für das angegebene Dateisystem und dessen untergeordnete Objekte verwendet werden. Zugriffsrechtsätze werden dynamisch ausgewertet, sodass Änderungen an einem Satz unverzüglich aktualisiert werden. Für Zugriffsrechtsätze gelten dieselben Benennungsregeln wie für ZFS-Dateisysteme, wobei der Name allerdings mit dem Zeichen `@` beginnen muss und nicht mehr als 64 Zeichen lang sein darf.

Die folgende `zfs allow`-Syntax (fett gedruckt) gibt an, wie die Zugriffsrechte übertragen werden:

`zfs allow [-ld]` `... ...` *filesystem* | *volume*

Die Option `-l` gibt an, dass das Zugriffsrecht für das angegebene Dataset, nicht aber für dessen untergeordnete Objekte gewährt wird, es sei denn, es wurde auch die Option `-d` angegeben. Die Option `-d` bedeutet, dass das Zugriffsrecht nicht für dieses Dataset, sondern für dessen untergeordnete Objekte gewährt wird, es sei denn, es wurde auch die Option `-l` angegeben. Wenn keine der Optionen `-ld` angegeben wird, gelten die Zugriffsrechte für das Dateisystem bzw. Volume und alle untergeordneten Objekte.

Löschen von delegierten ZFS-Zugriffsrechten (`zfs unallow`)

Mit dem Befehl `zfs unallow` können Sie zuvor erteilte Zugriffsrechte wieder löschen.

Gehen wir beispielsweise davon aus, dass Sie die Zugriffsrechte `create`, `destroy`, `mount` und `snapshot` wie folgt delegiert haben:

```
# zfs allow cindys create,destroy,mount,snapshot tank/cindys
# zfs allow tank/cindys
-------------------------------------------------------------
Local+Descendent permissions on (tank/cindys)
        user cindys create,destroy,mount,snapshot
-------------------------------------------------------------
```

Zum Entziehen dieser Zugriffsrechte müssen Sie die folgende Syntax verwenden:

```
# zfs unallow cindys tank/cindys
# zfs allow tank/cindys
```

Arbeiten mit der delegierten ZFS-Administration

In diesem Abschnitt finden Sie Beispiele zum Anzeigen und Delegieren von ZFS-Zugriffsrechten.

Anzeigen von delegierten ZFS-Zugriffsrechten (Beispiele)

Mit dem folgenden Befehl werden die Zugriffsrechte angezeigt:

```
# zfs allow dataset
```

Dieser Befehl gibt die für das jeweilige Dataset gesetzten oder gewährten Zugriffsrechte aus. Die Ausgabe enthält folgende Komponenten:

- Zugriffsrechtsätze
- Spezifische Zugriffsrechte oder Create-time-Zugriffsrechte
- Lokales Dataset
- Lokale und untergeordnete Datasets
- Nur untergeordnete Datasets

BEISPIEL 9–1 Anzeigen einfacher Zugriffsrechte für die delegierte Administration

Die folgende Ausgabe in diesem Beispiel bedeutet, dass der Benutzer `cindys` berechtigt ist, im Dateisystem `tank/cindys` Objekte zu erstellen, zu löschen, einzuhängen und Snapshots zu erstellen.

```
# zfs allow tank/cindys
       -------------------------------------------------------------
       Local+Descendent permissions on (tank/cindys)
```

BEISPIEL 9–1 Anzeigen einfacher Zugriffsrechte für die delegierte Administration *(Fortsetzung)*

```
                user cindys create,destroy,mount,snapshot
```

BEISPIEL 9–2 Anzeigen komplexer Zugriffsrechte für die delegierte Administration

Die Ausgabe in diesem Beispiel zeigt die folgenden Zugriffsrechte für die Dateisysteme `pool/fred` und `pool` auf.

Für das Dateisystem `pool/fred`:

- Es sind zwei Zugriffsrechtsätze definiert:
  - `@eng` (create, destroy, snapshot, mount, clone, promote, rename)
  - `@simple` (create, mount)
- Die Create-time-Zugriffsrechte werden für den Zugriffsrechtsatz `@eng` und die Eigenschaft `mountpoint` gesetzt. „Create-time“ bedeutet, dass der Zugriffsrechtsatz `@eng` und die Eigenschaft `mountpoint` nach der Erstellung eines Datasets gewährt werden.
- Dem Benutzer `tom` wird der Zugriffsrechtsatz `@eng` und dem Benutzer `joe` werden die Zugriffsrechte `create`, `destroy` und `mount` für lokale Dateisysteme gewährt.
- Benutzer `fred` erhält den Zugriffsrechtsatz `@basic` sowie die Zugriffsrechte `share` und `rename` für das lokale und die untergeordneten Dateisysteme.
- Dem Benutzer `barney` und der Gruppe `staff` wird der Zugriffsrechtsatz `@basic` ausschließlich für untergeordnete Dateisysteme übertragen.

Für das Dateisystem `pool`:

- Der Zugriffsrechtsatz `@simple` (create, destroy, mount) ist definiert.
- Die Gruppe `staff` erhält den Zugriffsrechtsatz `@simple` für das lokale Dateisystem.

Sehen Sie hier die Ausgabe für dieses Beispiel:

```
$ zfs allow pool/fred
-------------------------------------------------------------------------------
Permission sets on (pool/fred)
        @eng create,destroy,snapshot,mount,clone,promote,rename
        @simple create,mount
Create time permissions on (pool/fred)
        @eng,mountpoint
Local permissions on (pool/fred)
        user tom @eng
        user joe create,destroy,mount
Local+Descendent permissions on (pool/fred)
        user fred @basic,share,rename
Descendent permissions on (pool/fred)
        user barney @basic
```

BEISPIEL 9–2 Anzeigen komplexer Zugriffsrechte für die delegierte Administration *(Fortsetzung)*

```
        group staff @basic
-------------------------------------------------------------------------------
Permission sets on (pool)
        @simple create,destroy,mount
Local permissions on (pool)
        group staff @simple
-------------------------------------------------------------------------------
```

Delegieren von ZFS-Zugriffsrechten (Beispiele)

BEISPIEL 9–3 Delegieren von Zugriffsrechten an einzelne Benutzer

Wenn Sie einem einzelnen Benutzer die Zugriffsrechte `create` und `mount` übertragen möchten, müssen Sie sich vergewissern, dass dieser über die erforderlichen Zugriffsrechte für den zugrunde liegenden Einhängepunkt verfügt.

Um beispielsweise dem Benutzer `marks` die Zugriffsrechte `create` und `mount` für `tank` zu erteilen, setzen Sie zuerst diese Zugriffsrechte:

```
# chmod A+user:marks:add_subdirectory:fd:allow /tank
```

Anschließend gewähren Sie mit dem Befehl `zfs allow` die Zugriffsrechte `create` , `destroy` und `mount`. Beispiel:

```
# zfs allow marks create,destroy,mount tank
```

Jetzt kann der Benutzer `marks` unter `tank` eigene Dateisysteme anlegen. Beispiel:

```
# su marks
marks$ zfs create tank/marks
marks$ ^D
# su lp
$ zfs create tank/lp
cannot create 'tank/lp': permission denied
```

BEISPIEL 9–4 Delegieren der Zugriffsrechte create und destroy an Gruppen

Das folgende Beispiel veranschaulicht, wie ein Dateisystem eingerichtet wird, damit jedes Mitglied der Gruppe `staff` Dateisysteme unter `tank` erstellen und einhängen sowie eigene Dateisysteme löschen kann. Die Gruppenmitglieder von `staff` können jedoch nicht die Dateisysteme anderer Gruppenmitglieder löschen.

BEISPIEL 9–4 Delegieren der Zugriffsrechte create und destroy an Gruppen *(Fortsetzung)*

```
# zfs allow staff create,mount tank
# zfs allow -c create,destroy tank
# zfs allow tank
-------------------------------------------------------------
Create time permissions on (tank)
        create,destroy
Local+Descendent permissions on (tank)
        group staff create,mount
-------------------------------------------------------------
# su cindys
cindys% zfs create tank/cindys
cindys% exit
# su marks
marks% zfs create tank/marks/data
marks% exit
cindys% zfs destroy tank/marks/data
cannot destroy 'tank/mark': permission denied
```

BEISPIEL 9–5 Delegieren von Zugriffsrechten auf der richtigen Ebene der Dateisystemhierarchie

Achten Sie darauf, den Benutzern die Zugriffsrechte auf der richtigen Ebene der Dateisystemhierarchie zuweisen. So werden beispielsweise dem Benutzer `marks` die Zugriffsrechte `create`, `destroy` und `mount` für das lokale und für die untergeordneten Dateisysteme gewährt. Benutzer `marks` erhält lokale Zugriffsrechte zum Erstellen von Snapshots des Dateisystems `tank`, nicht aber des eigenen Dateisystems. Ihm wurde das Zugriffsrecht `snapshot` also nicht auf der richtigen Ebene der Dateisystemhierarchie übertragen.

```
# zfs allow -l marks snapshot tank
# zfs allow tank
-------------------------------------------------------------
Local permissions on (tank)
        user marks snapshot
Local+Descendent permissions on (tank)
        user marks create,destroy,mount
-------------------------------------------------------------
# su marks
marks$ zfs snapshot tank/@snap1
marks$ zfs snapshot tank/marks@snap1
cannot create snapshot 'mark/marks@snap1': permission denied
```

Um Benutzer `marks` Zugriffsrechte für die untergeordnete Ebene zu gewähren, verwenden Sie den Befehl `zfs allow` mit der Option `-d`. Beispiel:

BEISPIEL 9–5 Delegieren von Zugriffsrechten auf der richtigen Ebene der Dateisystemhierarchie *(Fortsetzung)*

```
# zfs unallow -l marks snapshot tank
# zfs allow -d marks snapshot tank
# zfs allow tank
-------------------------------------------------------------
Descendent permissions on (tank)
        user marks snapshot
Local+Descendent permissions on (tank)
        user marks create,destroy,mount
-------------------------------------------------------------
# su marks
$ zfs snapshot tank@snap2
cannot create snapshot 'tank@snap2': permission denied
$ zfs snapshot tank/marks@snappy
```

Jetzt kann der Benutzer `marks` nur Snapshots unterhalb der Ebene `tank` erstellen.

BEISPIEL 9–6 Definieren und Anwenden komplexer delegierter Zugriffsrechte

Benutzern oder Gruppen können spezifische Zugriffsrechte gewährt werden. So werden beispielsweise mit dem folgenden `zfs allow`-Befehl spezifische Zugriffsrechte an die Gruppe `staff` übertragen. Darüber hinaus gelten die Zugriffsrechte `destroy` und `snapshot`, sobald`tank`-Dateisysteme erstellt werden.

```
# zfs allow staff create,mount tank
# zfs allow -c destroy,snapshot tank
# zfs allow tank
-------------------------------------------------------------
Create time permissions on (tank)
        destroy,snapshot
Local+Descendent permissions on (tank)
        group staff create,mount
-------------------------------------------------------------
```

Da Benutzer `marks` ein Mitglied der Gruppe `staff` ist, kann er in `tank` Dateisysteme erstellen. Zusätzlich kann Benutzer `marks` Snapshots von `tank/marks2` erstellen, da er über die spezifischen Zugriffsrechte verfügt. Beispiel:

```
# su marks
$ zfs create tank/marks2
$ zfs allow tank/marks2
-------------------------------------------------------------
Local permissions on (tank/marks2)
        user marks destroy,snapshot
-------------------------------------------------------------
```

BEISPIEL 9–6 Definieren und Anwenden komplexer delegierter Zugriffsrechte *(Fortsetzung)*

```
Create time permissions on (tank)
        destroy,snapshot
Local+Descendent permissions on (tank)
        group staff create
        everyone mount
-------------------------------------------------------------
```

Er kann jedoch keine Snapshots in `tank/marks` erstellen, da ihm die Zugriffsrechte hierfür fehlen. Beispiel:

```
$ zfs snapshot tank/marks2@snap1
$ zfs snapshot tank/marks@snappp
cannot create snapshot 'tank/marks@snappp': permission denied
```

Wenn Sie in Ihrem home-Verzeichnis über die Berechtigung `create` verfügen, können Sie eigene Snapshot-Verzeichnisse erstellen. Dies kann sich bei Dateisystemen als hilfreich erweisen, die per NFS eingehängt werden. Beispiel:

```
$ cd /tank/marks2
$ ls
$ cd .zfs
$ ls
snapshot
$ cd snapshot
$ ls -l
total 3
drwxr-xr-x   2 marks    staff          2 Dec 15 13:53 snap1
$ pwd
/tank/marks2/.zfs/snapshot
$ mkdir snap2
$ zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank                   264K  33.2G  33.5K  /tank
tank/marks            24.5K  33.2G  24.5K  /tank/marks
tank/marks2             46K  33.2G  24.5K  /tank/marks2
tank/marks2@snap1     21.5K      -  24.5K  -
tank/marks2@snap2         0      -  24.5K  -
$ ls
snap1  snap2
$ rmdir snap2
$ ls
snap1
```

BEISPIEL 9–7 Definieren und Anwenden von delegierten Zugriffsrechtsätzen für ZFS

Das folgende Beispiel zeigt, wie der Zugriffsrechtsatz `@myset` erstellt wird und dieser und das Zugriffsrecht rename an die Gruppe `staff` für das Dateisystem `tank` übertragen werden. Der Benutzer `cindys`, Mitglied der Gruppe `staff`, verfügt über die Berechtigung zum Erstellen von Dateisystemen in `tank`. Benutzer `lp` verfügt jedoch nicht über die Berechtigung zum Erstellen von Dateisystemen in `tank`.

```
# zfs allow -s @myset create,destroy,mount,snapshot,promote,clone,readonly tank
# zfs allow tank
-------------------------------------------------------------
Permission sets on (tank)
        @myset clone,create,destroy,mount,promote,readonly,snapshot
-------------------------------------------------------------
# zfs allow staff @myset,rename tank
# zfs allow tank
-------------------------------------------------------------
Permission sets on (tank)
        @myset clone,create,destroy,mount,promote,readonly,snapshot
Local+Descendent permissions on (tank)
        group staff @myset,rename
# chmod A+group:staff:add_subdirectory:fd:allow tank
# su cindys
cindys% zfs create tank/data
Cindys% zfs allow tank
-------------------------------------------------------------
Permission sets on (tank)
        @myset clone,create,destroy,mount,promote,readonly,snapshot
Local+Descendent permissions on (tank)
        group staff @myset,rename
-------------------------------------------------------------
cindys% ls -l /tank
total 15
drwxr-xr-x   2 cindys   staff          2 Aug  8 14:10 data
cindys% exit
# su lp
$ zfs create tank/lp
cannot create 'tank/lp': permission denied
```

Löschen von ZFS-Zugriffsrechten (Beispiele)

Mit dem Befehl `zfs unallow` lassen sich übertragene Zugriffsrechte wieder löschen. Beispielsweise verfügt Benutzer `cindys` über die Zugriffsrechte zum Erstellen, Löschen, Einhängen und Anfertigen von Snapshots im Dateisystem `tank/cindys`.

```
# zfs allow cindys create,destroy,mount,snapshot tank/cindys
       # zfs allow tank/cindys
       -------------------------------------------------------------
       Local+Descendent permissions on (tank/cindys)
               user cindys create,destroy,mount,snapshot
       -------------------------------------------------------------
```

Mit der folgenden `zfs unallow`-Syntax entziehen Sie das Zugriffsrecht von Benutzer `cindys` zum Erstellen von Snapshots für das Dateisystem `tank/cindys`:

```
# zfs unallow cindys snapshot tank/cindys
# zfs allow tank/cindys
-------------------------------------------------------------
Local+Descendent permissions on (tank/cindys)
        user cindys create,destroy,mount
-------------------------------------------------------------
cindys% zfs create tank/cindys/data
cindys% zfs snapshot tank/cindys@today
cannot create snapshot 'tank/cindys@today': permission denied
```

Betrachten wir als weiteres Beispiel die nachfolgenden Zugriffsrechte des Benutzers `marks` in `tank/marks`:

```
# zfs allow tank/marks
-------------------------------------------------------------
Local+Descendent permissions on (tank/marks)
        user marks create,destroy,mount
-------------------------------------------------------------
```

In diesem Beispiel werden mit der folgenden `zfs unallow`-Syntax alle Zugriffsrechte für Benutzer `marks` von `tank/marks` gelöscht:

```
# zfs unallow marks tank/marks
```

Die folgende `zfs unallow`-Syntax löscht einen Zugriffsrechtsatz für das Dateisystem `tank`.

```
# zfs allow tank
-------------------------------------------------------------
Permission sets on (tank)
        @myset clone,create,destroy,mount,promote,readonly,snapshot
Create time permissions on (tank)
        create,destroy,mount
Local+Descendent permissions on (tank)
        group staff create,mount
-------------------------------------------------------------
# zfs unallow -s @myset tank
$ zfs allow tank
-------------------------------------------------------------
```

```
Create time permissions on (tank)
        create,destroy,mount
Local+Descendent permissions on (tank)
        group staff create,mount
-------------------------------------------------------------
```

◆ ◆ ◆ KAPITEL 10

Fortgeschrittene ZFS-Themen

Dieses Kapitel enthält Informationen zu ZFS-Volumes, zur Verwendung von ZFS auf Solaris-Systemen mit installierten Zonen, zu Speicher-Pools mit alternativem Root-Verzeichnis sowie zu ZFS-Zugriffsrechtsprofilen.

Dieses Kapitel enthält die folgenden Abschnitte:

ZFS-Volumes

Ein ZFS-Volume ist ein Dataset, das eine Blockeinheit darstellt. ZFS-Volumes sind im Verzeichnis `/dev/zvol/{dsk,rdsk}/path` als Geräte aufgeführt.

Im folgenden Beispiel wird das ZFS-Volume `tank/vol` mit einer Kapazität von 5 GB erstellt:

```
# zfs create -V 5gb tank/vol
```

Beim Erstellen von Volumes wird automatisch eine Reservierung der anfänglichen Volume-Kapazität angelegt. Die Reservierungsgröße wird stets an die Kapazität des betreffenden Volumes angepasst, sodass kein unvorhergesehenes Verhalten auftritt. Wenn die Kapazität des Volumes beispielsweise kleiner wird, können Daten beschädigt werden. Deswegen müssen Sie beim Ändern der Kapazität eines Volumes äußerst sorgfältig vorgehen.

Wenn Sie einen Snapshot eines Volumes erstellen, dessen Kapazität sich ändern kann, können darüber hinaus Dateisysteminkonsistenzen entstehen, wenn Sie versuchen, den betreffenden Snapshot mittels Rollback rückgängig zu machen oder zu klonen.

Informationen zu Dateisystemeigenschaften, die auf Volumes angewendet werden können, finden Sie in Tabelle 6–1.

Auf Solaris-Systemen mit installierten Zonen können Sie keine ZFS-Volumes in einer nicht-globalen Zone erstellen bzw. klonen. Alle Versuche zum Erstellen bzw. Klonen eines Volumes innerhalb einer nicht-globalen Zone schlagen fehl. Informationen zur Verwendung von ZFS-Volumes in einer globalen Zone finden Sie unter „Hinzufügen von ZFS-Volumes zu einer nicht-globalen Zone“ auf Seite 285.

Verwendung von ZFS-Volumes als Swap- bzw. Dump-Gerät

Während der Installation eines ZFS-Root-Dateisystems oder einer Migration von einem UFS-Root-Dateisystem wird auf einem ZFS-Volume im ZFS-Root-Pool ein Swap-Gerät erstellt. Beispiel:

```
# swap -l
swapfile                  dev    swaplo   blocks     free
/dev/zvol/dsk/rpool/swap 253,3        16  8257520  8257520
```

Während der Installation eines ZFS-Root-Dateisystems oder einer Migration von einem UFS-Root-Dateisystem wird auf einem ZFS-Volume im ZFS-Root-Pool ein Dump-Gerät erstellt. Nach der Einrichtung ist keine Verwaltung des Dump-Geräts erforderlich. Beispiel:

```
# dumpadm
      Dump content: kernel pages
       Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash/t2000
  Savecore enabled: yes
```

Aufgrund von CR 6724860 müssen Sie bei Verwendung eines ZFS-Dump-Volumes `savecore` manuell ausführen, um einen Speicherabzug bei Systemabsturz zu speichern.

Wenn Sie den Swap-Bereich oder das Dump-Gerät nach der Installation oder dem Upgrade des Systems ändern müssen, benutzen Sie hierzu die Befehle `swap` und `dumpadm` wie in vorherigen Solaris-Versionen. Zum Einrichten eines zusätzlichen Swap-Bereiches müssen Sie ein ZFS-Volume einer bestimmten Kapazität erstellen und für dieses Gerät dann die Swap-Funktion aktivieren. Beispiel:

```
# zfs create -V 2G rpool/swap2
# swap -a /dev/zvol/dsk/rpool/swap2
# swap -l
swapfile                   dev  swaplo blocks   free
/dev/zvol/dsk/rpool/swap  256,1      16 2097136 2097136
/dev/zvol/dsk/rpool/swap2 256,5      16 4194288 4194288
```

Swaps dürfen nicht in eine Datei eines ZFS-Dateisystems durchgeführt werden. ZFS unterstützt keine Konfigurationen für Swap-Dateien.

Informationen zum Anpassen der Größe von Swap- und Dump-Volumes finden Sie unter „Anpassen der Größe von ZFS-Swap- und Dump-Geräten“ auf Seite 159.

Verwendung von ZFS-Volumes als Solaris-iSCSI-Zielgerät

Solaris-iSCSI-Zielgeräte und -Initiatoren werden in Solaris unterstützt.

Darüber hinaus können Sie durch Setzen der Eigenschaft `shareiscsi` des Volumes ZFS-Volumes auf einfache Weise als iSCSI-Zielgerät konfigurieren. Beispiel:

```
# zfs create -V 2g tank/volumes/v2
# zfs set shareiscsi=on tank/volumes/v2
# iscsitadm list target
Target: tank/volumes/v2
    iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
    Connections: 0
```

Nach dem Erstellen des iSCSI-Zielgeräts muss der iSCSI-Initiator definiert werden. Weitere Informationen zu Solaris-iSCSI-Zielgeräten und -Initiatoren finden Sie in Kapitel 14, „Configuring Solaris iSCSI Targets and Initiators (Tasks)“ in *System Administration Guide: Devices and File Systems*.

Hinweis – Solaris-iSCSI-Zielgeräte können auch mit dem Befehl `iscsitadm` erstellt und verwaltet werden. Wenn die Eigenschaft `shareiscsi` eines ZFS-Volumes gesetzt ist, darf der Befehl `iscsitadm` nicht zum Erstellen des gleichen Zielgeräts verwendet werden, da ansonsten für das gleiche Zielgerät duplizierte Zielgerätinformationen erstellt werden.

Als iSCSI-Zielgerät konfigurierte ZFS-Volumes werden genauso wie andere ZFS-Datasets verwaltet. Operationen zum Umbenennen, Exportieren und Importieren funktionieren bei iSCSI-Zielgeräten jedoch etwas anders.

- Wenn Sie ein ZFS-Volume umbenennen, bleibt der Name des iSCSI-Zielgeräts gleich. Beispiel:

  ```
  # zfs rename tank/volumes/v2 tank/volumes/v1
  # iscsitadm list target
  Target: tank/volumes/v1
      iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
      Connections: 0
  ```

- Durch das Exportieren eines Pools, das ein für den Netzwerkzugriff freigegebenes ZFS-Volume enthält, wird das betreffende Zielgerät entfernt. Durch das Importieren eines Pools, das ein freigegebenes ZFS-Volume enthält, wird das betreffende Zielgerät für den Netzwerkzugriff freigegeben. Beispiel:

```
# zpool export tank
# iscsitadm list target
# zpool import tank
# iscsitadm list target
Target: tank/volumes/v1
    iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
    Connections: 0
```

Alle Konfigurationsinformationen zu einem iSCSI-Zielgerät werden im Dataset gespeichert. Wie bei einem über NFS für den Netzwerkzugriff freigegebenem Dateisystem, wird ein in ein anderes System importiertes iSCSI-Zielgerät entsprechend für den Netzwerkzugriff freigegeben.

Verwendung von ZFS in einem Solaris-System mit installierten Zonen

In den folgenden Abschnitten wird die Verwendung von ZFS auf Systemen mit Solaris-Zonen beschrieben.

Informationen zum Konfigurieren eines Systems mit einem ZFS-Root-Dateisystem, das mithilfe von Solaris Live Upgrade migriert oder gepatcht werden soll, finden Sie unter „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 10/08)“ auf Seite 141oder „Verwenden von Solaris Live Upgrade zum Migrieren oder Aktualisieren eines Systems mit Zonen (Solaris 10 5/09 und Solaris 10 10/09)“ auf Seite 147.

Für die Zusammenarbeit von ZFS-Datasets mit Zonen sollten Sie folgende Aspekte berücksichtigen:

- Sie können ein ZFS-Dateisystem bzw. einen ZFS-Klon ohne Delegierung der Verwaltungssteuerung zu einer nicht-globalen Zone hinzufügen.
- ZFS-Volumes können als Gerät zu nicht-globalen Zonen hinzugefügt werden.
- Gegenwärtig können ZFS-Snapshots nicht in Zonen erstellt werden.

In den folgenden Abschnitten wird mit dem Begriff „ZFS-Dataset” ein Dateisystem bzw. Klon bezeichnet.

Durch Hinzufügen eines Datasets kann sich einen nicht-globale Zone Speicherplatz mit der globalen Zone teilen, obwohl der Zonen-Administrator in der zugrunde liegenden Dateisystemhierarchie keine Eigenschaften einstellen bzw. keine neuen Dateisysteme erstellen kann. Dies ist auch beim Hinzufügen anderer Dateisystemtypen zu einer Zone der Fall und sollte dann verwendet werden, wenn das Hauptziel in der gemeinsamen Nutzung von Speicherplatz besteht.

In ZFS können Datasets auch an eine nicht-globale Zone delegiert werden. Dadurch erlangt der Zonen-Administrator vollständige Kontrolle über diese Datasets und alle seine untergeordneten Objekte. Der Zonen-Administrator kann Dateisysteme bzw. Klone in diesem Dataset erstellen oder zerstören sowie Dataset-Eigenschaften ändern. Der Zonen-Administrator hat jedoch keine Kontrolle über Datasets, die nicht zur betreffenden Zone hinzugefügt wurden, und darf die für die oberste Hierarchieebene des exportierten Datasets gesetzten Kontingente nicht überschreiten.

Bei der Verwendung von ZFS auf Systemen mit installierten Solaris-Zonen sollten Sie die folgenden Interaktionen berücksichtigen:

- Die Eigenschaft `mountpoint` eines zu einer nicht-globalen Zone hinzugefügten ZFS-Dateisystems muss auf „legacy" gesetzt sein.
- Sehen Sie aufgrund von CR 6449301 davon ab, einer nicht-globalen Zone bei der Konfiguration ein ZFS-Dataset hinzuzufügen. Sie können das·ZFS-Dataset hinzufügen, nachdem die Zone installiert wurde.
- Befinden sich der `zonepath` der Quelle und der `zonepath` des Ziels in einem ZFS und im gleichen Pool, so verwendet der Befehl `zoneadm clone` nun automatisch die ZFS-Klonfunktion, um die Zone zu klonen. Der Befehl `zoneadm clone` erstellt einen ZFS-Snapshot des `zonepath`-Quellverzeichnisses und richtet das `zonepath`-Zielverzeichnis ein. Das Klonen einer Zone mit dem Befehl `zfs clone` ist nicht möglich. Weitere Informationen finden Sie in Teil II, „Zonen" in *Systemverwaltungshandbuch: Solaris Container – Ressourcenverwaltung und Solaris Zones*.
- Wenn Sie ein ZFS-Dateisystem an eine nicht-globale Zone delegieren, müssen Sie dieses Dateisystem vor der Verwendung von Solaris Live Upgrade aus der nicht-globalen Zone entfernen. Andernfalls schlägt das Live Upgrade mit einem Fehler wegen des Schreibschutzes des Dateisystems fehl.

Hinzufügen von ZFS-Dateisystemen zu einer nicht-globalen Zone

Wenn das Hauptziel ausschließlich in der gemeinsamen Nutzung von Speicherplatz mit der globalen Zone besteht, können Sie ein ZFS-Dateisystem zu einer nicht-globalen Zone als generisches Dateisystem hinzufügen. Die Eigenschaft `mountpoint` eines zu einer nicht-globalen Zone hinzugefügten ZFS-Dateisystems muss auf „legacy" gesetzt sein.

Sie können ein ZFS-Dateisystem mit dem Unterbefehl `add fs` des Befehls `zonecfg` zu einer nicht-globalen Zone hinzufügen. Beispiel:

Im folgenden Beispiel wird ein ZFS-Dateisystem durch den globalen Administrator in der globalen Zone zu einer nicht-globalen Zone hinzugefügt.

```
# zonecfg -z zion
zonecfg:zion> add fs
zonecfg:zion:fs> set type=zfs
zonecfg:zion:fs> set special=tank/zone/zion
zonecfg:zion:fs> set dir=/export/shared
zonecfg:zion:fs> end
```

Diese Syntax fügt das ZFS-Dateisystem `tank/zone/zion` zur bereits konfigurierten und unter `/export/shared` eingehängten Zone `zion` hinzu. Die Eigenschaft `mountpoint` des Dateisystems muss auf `legacy` gesetzt sein, und das Dateisystem darf nicht schon irgendwo anders eingehängt sein. Der Zonenadministrator kann Dateien im Dateisystem erstellen und löschen. Das Dateisystem kann nicht an einer anderen Stelle eingehängt werden, und der Zonenadministrator kann keine Dateisystemeigenschaften wie z. B. „atime", „readonly", „compression" usw. ändern. Der Administrator der globalen Zone ist für das Einstellen und Steuern von Dateisystemeigenschaften verantwortlich.

Weitere Informationen zum Befehl `zonecfg` sowie zur Konfiguration von Ressourcentypen mithilfe von `zonecfg` finden Sie unter Teil II, „Zonen" in *Systemverwaltungshandbuch: Solaris Container – Ressourcenverwaltung und Solaris Zones*.

Delegieren von Datasets in eine nicht-globale Zone

Wenn das Hauptziel in der Delegierung der Speicherplatzverwaltung an eine Zone besteht, können Datasets mit dem Unterbefehl `add dataset` des Befehls `zonecfg` zu einer nicht-globalen Zone hinzugefügt werden.

Im folgenden Beispiel wird ein ZFS-Dateisystem durch den globalen Administrator in der globalen Zone an eine nicht-globale Zone delegiert.

```
# zonecfg -z zion
zonecfg:zion> add dataset
zonecfg:zion:dataset> set name=tank/zone/zion
zonecfg:zion:dataset> end
```

Im Gegensatz zum Hinzufügen von Dateisystemen wird durch diese Syntax das ZFS-Dateisystem `tank/zone/zion` in der bereits konfigurierten Zone `zion` sichtbar. Der Zonenadministrator kann Dateisystemeigenschaften einstellen und untergeordnete Objekte erstellen. Darüber hinaus kann der Zonenadministrator Snapshots sowie Klone erstellen und die Dateisystemhierarchie auch anderweitig steuern.

Wenn Sie Solaris Live Upgrade zur Aktualisierung der ZFS-BU mit nicht-globalen Zonen verwenden, sollten Sie vorher jegliche delegierte Datasets entfernen. Andernfalls schlägt das Live Upgrade mit einem Fehler wegen des Schreibschutzes des Dateisystems fehl. Beispiel:

```
zonecfg:zion>
zonecfg:zone1> remove dataset name=tank/zone/zion
zonecfg:zone1> exit
```

Weitere Informationen zu zulässigen Aktionen innerhalb von Zonen finden Sie unter „Verwalten von ZFS-Eigenschaften innerhalb einer Zone“ auf Seite 286.

Hinzufügen von ZFS-Volumes zu einer nicht-globalen Zone

ZFS-Volumes können nicht mit dem Unterbefehl `add dataset` des Befehls `zonecfg` zu einer nicht-globalen Zone hinzugefügt werden. Wenn das Hinzufügen eines ZFS-Volumes erkannt wird, kann für die Zone kein Boot-Vorgang durchgeführt werden. Mithilfe des Unterbefehls `add device` des Befehls `zonecfg` können Zonen jedoch Volumes hinzugefügt werden.

Im folgenden Beispiel wird ein ZFS-Volume durch den globalen Administrator in der globalen Zone zu einer nicht-globalen Zone hinzugefügt:

```
# zonecfg -z zion
zion: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:zion> create
zonecfg:zion> add device
zonecfg:zion:device> set match=/dev/zvol/dsk/tank/vol
zonecfg:zion:device> end
```

Diese Syntax exportiert das ZFS-Volume `tank/vol` in die Zone. Bitte beachten Sie, dass das Hinzufügen eines Volumes im Raw-Modus auch dann Sicherheitsrisiken in sich birgt, wenn das Volume keinem physischen Datenspeichergerät entspricht. Der Zonenadministrator könnte dadurch insbesondere fehlerhafte Dateisysteme erstellen, die beim Einhängen einen Systemabsturz verursachen. Weitere Informationen zum Hinzufügen von Geräten zu Zonen und den damit verbundenen Sicherheitsrisiken finden Sie unter „Informationen zur Eigenschaft `zoned`“ auf Seite 287.

Weitere Informationen zum Hinzufügen von Geräten zu Zonen finden Sie unter Teil II, „Zonen“ in *Systemverwaltungshandbuch: Solaris Container – Ressourcenverwaltung und Solaris Zones*.

Verwenden von ZFS-Speicher-Pools innerhalb einer Zone

ZFS-Speicher-Pools können in Zonen werder erstellt noch geändert werden. Das Modell der delegierten Administration zentralisiert die Verwaltung physischer Datenspeichergeräte in der globalen Zone und die Verwaltung von virtuellem Speicherplatz in den nicht-globalen Zonen. Während Datasets auf Pool-Ebene zu einer Zone hinzugefügt werden können, sind Befehle, die die physischen Eigenschaften eines Pools ändern (z. B. Erstellen, Hinzufügen oder Entfernen von Datenspeichergeräten) innerhalb einer Zone nicht zulässig. Auch wenn physische Datenspeichergeräte mit dem Unterbefehl `add device` des Befehls `zonecfg` zu einer Zone hinzugefügt oder Dateien verwendet werden, erlaubt der Befehl `zpool` kein Erstellen neuer Pools innerhalb der Zone.

Verwalten von ZFS-Eigenschaften innerhalb einer Zone

Nach dem Hinzufügen eines Datasets zu einer Zone kann der Zonenadministrator bestimmte Dataset-Eigenschaften einstellen. Wenn ein Dataset zu einer Zone hinzugefügt wird, sind alle seine übergeordneten Datasets als schreibgeschützte Datasets sichtbar. Das Dataset selbst sowie alle seine untergeordneten Datasets besitzen diesen Schreibschutz jedoch nicht. Betrachten wir zum Beispiel die folgende Konfiguration:

```
global# zfs list -Ho name
tank
tank/home
tank/data
tank/data/matrix
tank/data/zion
tank/data/zion/home
```

Beim Hinzufügen von `tank/data/zion` zu einer Zone besäße jedes Dataset folgende Eigenschaften.

Dataset	Sichtbar	Beschreibbar	Unveränderbare Eigenschaften
`tank`	Ja	Nein	-
`tank/home`	Nein	-	-
`tank/data`	Ja	Nein	-
`tank/data/matrix`	Nein	-	-

Dataset	Sichtbar	Beschreibbar	Unveränderbare Eigenschaften
`tank/data/zion`	Ja	Ja	`sharenfs`, `zoned`, `quota`, `reservation`
`tank/data/zion/home`	Ja	Ja	`sharenfs`, `zoned`

Bitte beachten Sie, dass übergeordnete Datasets von `tank/zone/zion` schreibgeschützt sichtbar, alle untergeordneten Datasets beschreibbar und nicht zur übergeordneten Hierarchie gehörende Datasets nicht sichtbar sind. Der Zonenadministrator kann die Eigenschaft `sharenfs` nicht ändern, da nicht-globale Zonen nicht als NFS-Server fungieren können. Darüber hinaus kann der Zonenadministrator auch nicht die Eigenschaft `zoned` ändern, da dies ein Sicherheitsrisiko (siehe nächster Abschnitt) darstellen würde.

Alle anderen konfigurierbaren Eigenschaften (außer der Eigenschaft `quota`) und das Dataset selbst können geändert werden. Durch dieses Verhalten kann der globale Zonenadministrator den Speicherplatzverbrauch aller Datasets in den nicht-globalen Zonen überwachen.

Darüber hinaus können nach dem Hinzufügen eines Datasets zu einer nicht-globalen Zone die Eigenschaften `sharenfs` und `mountpoint` vom globalen Zonenadministrator nicht geändert werden.

Informationen zur Eigenschaft `zoned`

Beim Hinzufügen eines Datasets zu einer nicht-globalen Zone muss dieses Dataset entsprechend gekennzeichnet werden, sodass bestimmte Eigenschaften nicht im Kontext der globalen Zone interpretiert werden. Nach dem Hinzufügen eines Datasets zu einer nicht-globalen Zone unter der Kontrolle des Zonenadministrators ist dessen Inhalt nicht mehr vertrauenswürdig. Wie bei anderen Dateisystemen auch können in der nicht-globalen Zone setuid-Binärdateien, symbolische Verknüpfungen oder anderweitig fragwürdige Inhalte auftreten, die sich negativ auf die Sicherheit der globalen Zone auswirken. Darüber hinaus kann die Eigenschaft `mountpoint` nicht im Kontext der globalen Zone interpretiert werden, da der Zonenadministrator andernfalls den Namensplatz der globalen Zone ändern kann. Zur Lösung des letzteren Problems nutzt ZFS die Eigenschaft `zoned`, die anzeigt, dass ein Dataset zu einem bestimmten Zeitpunkt an eine nicht-globale Zone delegiert wurde.

Die Eigenschaft `zoned` ist ein boolescher Wert, der beim Booten einer Zone, die ein ZFS-Dataset enthält, automatisch auf „true” gesetzt wird. Diese Eigenschaft muss vom Zonenadministrator nicht manuell gesetzt werden. Wenn die Eigenschaft `zoned` gesetzt ist, kann das·Dataset in der globalen Zone nicht eingehängt bzw. für den Netzwerkzugriff freigegeben werden und wird bei Ausführung des Befehls `zfs share -a` bzw. `zfs mount -a` ignoriert. Im folgenden Beispiel wurde `tank/zone/zion` zu einer Zone hinzugefügt und `tank/zone/global` nicht:

```
# zfs list -o name,zoned,mountpoint -r tank/zone
NAME                  ZONED  MOUNTPOINT
tank/zone/global        off  /tank/zone/global
tank/zone/zion           on  /tank/zone/zion
# zfs mount
tank/zone/global           /tank/zone/global
tank/zone/zion             /export/zone/zion/root/tank/zone/zion
```

Bitte beachten Sie den Unterschied zwischen der Eigenschaft `mountpoint` und dem Verzeichnis, in das das Dataset `tank/zone/zion` gegenwärtig eingehängt ist. Die Eigenschaft `mountpoint` zeigt an, wo das Dataset auf dem Datenträger gespeichert ist, und nicht, wo es gegenwärtig eingehängt ist.

Beim Entfernen eines Datasets aus einer Zone bzw. Löschen einer Zone wird die Eigenschaft `zoned` **nicht** automatisch auf „false" zurückgesetzt. Dieses Verhalten resultiert aus den diesen Aufgaben innewohnenden Sicherheitsrisiken. Da ein nicht vertrauenswürdiger Benutzer vollständigen Zugriff auf das Dataset und seine untergeordneten Datasets hatte, kann die Eigenschaft `mountpoint` auf ungültige Werte gesetzt bzw. es können in den Dateisystemen setuid-Binärdateien vorhanden sein.

Zum Vermeiden versehentlicher Sicherheitsrisiken muss die Eigenschaft `zoned` vom Administrator der globalen Zone manuell zurückgesetzt werden, wenn das betreffende Dataset wiederverwendet werden soll. Bevor Sie die Eigenschaft `zoned` auf `off` setzen, müssen Sie sich vergewissern, dass die Eigenschaft `mountpoint` des Datasets und aller seiner untergeordneten Datasets auf zulässige Werte gesetzt ist und keine setuid-Binärdateien vorhanden sind, oder die Eigenschaft `setuid` deaktivieren.

Nachdem Sie sich überzeugt haben, dass keine Sicherheitslücken existieren, können Sie die Eigenschaft `zoned` mithilfe der Befehle `zfs set` bzw. `zfs inherit` deaktivieren. Wenn die Eigenschaft `zoned` deaktiviert wird, wenn das betreffende Dataset innerhalb einer Zone verwendet wird, kann das System unvorhersehbar reagieren. Deswegen sollten Sie den Wert dieser Eigenschaft nur dann ändern, wenn Sie sich sicher sind, dass das Dataset nicht mehr von einer nicht-globalen Zone verwendet wird.

Verwenden von ZFS-Speicher-Pools mit alternativem Root-Verzeichnis

Beim Erstellen eines Pools wird es inhärent mit dem Host-System verknüpft. Das Host-System „kennt" der Pool, sodass es „weiß", wenn es nicht vefügbar ist. Obwohl dies für den Normalbetrieb nützlich ist, können diese Informationen hinderlich sein, wenn das System von einem alternativen Datenträger gebootet oder ein Pool auf Wechsel-Datenträgern erstellt wird. Zur Lösung dieses Problems bietet ZFS *Speicher-Pools mit alternativem Root-Verzeichnis.* Speicher-Pools mit alternativem Root-Verzeichnis gelten nur temporär für einen Systemneustart, und alle Einhängepunkte werden relativ zum neuen Root-Verzeichnis des betreffenden Pools geändert.

Erstellen von ZFS-Speicher-Pools mit alternativem Root-Verzeichnis

Speicher-Pools mit alternativem Root-Verzeichnis werden am Häufigsten für Wechsel-Datenträger verwendet. In einem solchen Fall wird ein einziges Dateisystem benötigt, und es soll in jede beliebige Stelle auf dem Zielsystem eingehängt werden können. Wenn mithilfe der Option `-R` ein Speicher-Pool mit alternativem Root-Verzeichnis erstellt wird, wird der Einhängepunkt des Root-Dateisystems automatisch auf `/` gesetzt, was dem alternativen Root-Verzeichnis selbst entspricht.

Im folgenden Beispiel wird ein Pool namens `morpheus` mit `/mnt` als alternativem Root-Verzeichnis erstellt:

```
# zpool create -R /mnt morpheus c0t0d0
# zfs list morpheus
NAME                    USED  AVAIL  REFER  MOUNTPOINT
morpheus               32.5K  33.5G     8K  /mnt/
```

Bitte beachten Sie das einzige Dateisystem `morpheus`, dessen Einhängepunkt das alternative Root-Verzeichnis `/mnt` ist. Der auf Datenträger gespeicherte Einhängepunkt ist `/` und der vollständige Pfad zu `/mnt` wird nur im Kontext des alternativen Root-Verzeichnisses interpretiert. Dieses Dateisystem kann dann auf einem anderen System unter einem beliebigen Speicher-Pool mit alternativem Root-Verzeichnis exportiert und importiert werden.

Importieren von Speicher-Pools mit alternativem Root-Verzeichnis

Pools können auch mithilfe eines alternativen Root-Verzeichnisses importiert werden. Dies ist in Situationen bei der Datenwiederherstellung nützlich, wo Einhängepunkte nicht im Kontext des aktuellen Root-Verzeichnisses interpretiert werden sollen, sondern relativ zu einem temporären Verzeichnis, in dem Reparaturen ausgeführt werden können. Diese Funktion kann auch beim Einhängen von Wechseldatenträgern (siehe oben) verwendet werden.

Im folgenden Beispiel wird ein Pool namens `morpheus` mit `/mnt` als alternativem Root-Verzeichnis importiert. Es wird vorausgesetzt, dass `morpheus` vorher exportiert wurde.

```
# zpool import -R /mnt morpheus
# zpool list morpheus
NAME                    SIZE    USED   AVAIL    CAP  HEALTH     ALTROOT
morpheus               33.8G   68.0K   33.7G     0%  ONLINE     /mnt
# zfs list morpheus
NAME                    USED  AVAIL  REFER  MOUNTPOINT
morpheus               32.5K  33.5G     8K  /mnt/morpheus
```

ZFS-Zugriffsrechtsprofile

Wenn Sie ZFS-Verwaltungsaufgaben ohne das Superuser-Benutzerkonto (Root) durchführen wollen, können Sie zum Ausführen von ZFS-Administrationsaufgaben mithilfe der folgenden Profile eine Rolle annehmen:

- ZFS-Speicherplatzverwaltung – Erstellen, Löschen und Ändern von Datenspeichergeräten in einem ZFS-Speicher-Pool
- ZFS-Dateisystemverwaltung – Erstellen, Löschen und Ändern von ZFS-Dateisystemen

Weitere Informationen zum Erstellen bzw. Annehmen von Rollen finden Sie im *System Administration Guide: Security Services* .

Zusätzlich zur Arbeit mit RBAC-Rollen für die Administration von ZFS-Dateisystemen empfiehlt sich der Einsatz der delegierten ZFS-Administration für verteilte ZFS-Verwaltungsaufgaben. Weitere Informationen finden Sie in Kapitel 9, „Delegierte ZFS-Administration“.

◆ ◆ ◆ KAPITEL 11

Problembehebung und Datenwiederherstellung in ZFS

Dieses Kapitel enthält Informationen zum Erkennen und Beseitigen von ZFS-Fehlerzuständen. Darüber hinaus werden hier auch Maßnahmen zum Vermeiden von Fehlfunktionen beschrieben.

Dieses Kapitel enthält die folgenden Abschnitte:

ZFS-Fehlerzustände

ZFS ist ein Dateisystem mit Datenträgerverwaltungsfunktionen und kann deswegen viele verschiedene Fehlerzustände annehmen. In diesem Kapitel werden die verschiedenen Fehlerzustände kurz umrissen. Dann wird erläutert, wie sie auf einem laufenden System erkannt werden können. Dieses Kapitel schließt mit einer Diskussion zum Beheben von Problemen ab. In ZFS treten drei verschiedene Fehlergrundtypen auf:

Bitte beachten Sie, dass in einem einzigen Pool alle drei Fehlertypen auftreten können. Deswegen umfasst eine vollständige Problembehebungsroutine das Finden und Beheben eines Fehlers, Weitergehen zum nächsten Fehler usw.

Fehlende Datenspeichergeräte in einem ZFS-Speicher-Pool

Wenn ein Datenspeichergerät vollständig aus dem System entfernt wird, erkennt ZFS, dass es nicht geöffnet werden kann und versetzt es in den Fehlerstatus UNAVAIL. Je nach der Datenreplikationsebene des betreffenden Pools kann es sein, dass dadurch der gesamte Pool nicht mehr zur Verfügung steht. Bei der Entfernung eines Datenträgers in Konfigurationen mit Datenspiegelung oder RAID-Z bleibt der Pool weiterhin verfügbar. Wenn alle Komponenten einer Datenspiegelungskonfiguration, mehrere Datenspeichergeräte in einer RAID-Z-Konfiguration oder ein Datenträger der obersten Hierarchieebene entfernt werden, geht der Pool in den Status `FAULTED`. Daten werden erst nach dem Wiedereingliedern der fehlenden Datenspeichergeräte verfügbar.

Beschädigte Datenspeichergeräte in einem ZFS-Speicher-Pool

Der Begriff „beschädigt" deckt eine breite Vielfalt möglicher Fehler ab. Dazu werden beispielsweise die folgenden Fehler gezählt:

- vorübergehende E/A-Fehler aufgrund eines fehlerhaften Datenträgers bzw. Controllers
- Datenbeschädigung auf Datenträgern aufgrund kosmischer Strahlung
- Treiberfehler, wegen denen Daten falsch transportiert werden
- vesehentliches Überschreiben von Bereichen auf einem physischen Datenträger durch einen anderen Benutzer

In einigen Fällen treten solche Fehler nur vorübergehend auf, so z. B. bei sporadischen E/A-Fehlern aufgrund eines Controller-Problems. In anderen Fällen ist der Schaden bleibend, wie z. B. bei der Datenbeschädigung auf einem Datenträger. Auch wenn der Schaden bleibend ist, heißt das nicht, dass der Fehler nicht noch einmal auftreten kann. Wenn ein Administrator beispielsweise versehentlich Bereiche eines Datenträgers überschreibt, ist kein Hardwareausfall aufgetreten, und das Datenspeichergerät muss nicht ausgetauscht werden. Herauszufinden, was genau mit einem Datenspeichergerät passiert ist, stellt keine leichte Aufgabe dar. Dieses Problem wird später eingehender behandelt.

Beschädigte ZFS-Daten

Datenbeschädigung tritt auf, wenn sich Gerätefehler (die auf fehlende bzw. beschädigte Datenspeichergeräte hinweisen) auf virtuelle Geräte der obersten Hierarchieebene auswirken. So können beispielsweise in einer Hälfte einer Datenspiegelungskonfiguration Tausende Gerätefehler auftreten, ohne dass dies zur Datenbeschädigung führt. Wenn in der anderen Hälfte der Datenspiegelungskonfiguration an genau der gleichen Stelle ein Fehler auftritt, verursacht das eine Datenbeschädigung.

Eine Datenbeschädigung ist stets bleibend und muss bei der Reparatur besonders berücksichtigt werden. Auch wenn die betreffenden Datenspeichergeräte repariert oder ausgetauscht werden, sind die ursprünglichen Daten für immer verloren. In den meisten Fällen müssen Daten in solchen Situationen aus Sicherungskopien wiederhergestellt werden Datenfehler werden beim Auftreten protokolliert und können durch regelmäßige Datenträgerbereinigung (siehe folgender Abschnitt) in Grenzen gehalten werden. Beim Entfernen eines beschädigten Datenblocks erkennt der nächste Durchlauf der Datenträgerbereinigung, dass die Datenbeschädigung nicht mehr auf dem System vorhanden ist und entfernt die Protokollierung des Fehlers vom System.

Überprüfen der ZFS-Datenintegrität

Für ZFS existiert kein Dienstprogramm wie `fsck`. Dieses Dienstprogramm diente traditionell zur Validierung und Reparatur von Daten.

Datenreparatur

Bei herkömmlichen Dateisystemen ist die Art und Weise des Schreibens von Daten inhärent anfällig für unerwartete Ausfälle, die zu Dateninkonsistenzen führen. Da herkömmliche Dateisysteme nicht transaktional sind, können unreferenzierte Datenblöcke, ungültige Verknüpfungszähler oder andere inkonsistente Datenstrukturen auftreten. Mit der Einführung des sog. Journaling wurden zwar einige dieser Probleme behoben, es können jedoch neue Probleme auftreten, wenn Transaktionen nicht rückgängig gemacht werden können. Mit ZFS treten solche Probleme nicht mehr auf. Inkonsistente Daten können nur bei Hardware-Ausfällen (die Verwendung redundanter Pools vermeidet dies) oder bei Fehlern in der ZFS-Software auftreten.

Angesichts der Tatsache, dass das Dienstprogramm `fsck` in einzelnen Dateisystemen auftretende bekannte Fehler beseitigen soll, ist die Entwicklung eines solchen Dienstprogramms für ein Dateisystem ohne bekannte Fehlerquellen unmöglich. Zukünftige Erfahrungen zeigen vielleicht, dass bestimmte Datenbeschädigungen entsprechend häufig auftreten und so einfach sind, dass dafür ein Reparaturdienstprogramm entwickelt werden kann; diese Probleme können jedoch durch den Einsatz redundanter Pools vermieden werden.

Wenn ein Pool nicht redundant ist, besteht immer die Möglichkeit, dass Sie nach einer Datenbeschädigung nicht mehr auf Ihre Daten zugreifen können.

Datenvalidierung

Neben der Datenreparatur stellt das Dienstprogramm `fsck` sicher, dass Daten auf einem Datenträger problemfrei sind. Diese Validierung wird herkömmlicherweise durch Aushängen des betreffenden Dateisystems und Ausführen des Dienstprogramms `fsck`für dieses

durchgeführt. Während dieses Vorgangs muss das System möglicherweise in den Einzelbenutzermodus gebracht werden. Daraus resultieren Ausfallzeiten, die zur Größe des zu überprüfenden Dateisystems proportional sind. Statt eines Dienstprogramms zum expliziten Ausführen der entsprechenden Überprüfungen besitzt ZFS einen Mechanismus zur regelmäßigen Überprüfung aller Daten. Diese als *Bereinigung* (engl. scrubbing) bezeichnete Funktionalität wird häufig in Arbeitsspeichern und anderen Systemen als Methode des Erkennens und Vermeidens von Problemen eingesetzt, dieHardware-Ausfälle bzw. Softwarefehlfunktionen verursachen.

Kontrollieren der ZFS-Datenbereinigung

Wenn ZFS einen Fehler erkennt (durch die Bereinigung oder beim routinemäßigen Zugriff auf Dateien), wird dieser intern protokolliert, sodass Sie einen schnellen Überblick über alle bekannten Fehler im Pool erhalten.

Explizite ZFS-Datenbereinigung

Die einfachste Methode zum Überprüfen der Datenintegrität besteht im Durchführen einer expliziten Bereinigung aller im Pool enthaltenen Daten. Dieser Vorgang durchläuft alle Daten im Pool einmal und stellt sicher, dass alle Datenblöcke gelesen werden können. Die Bereinigung erfolgt so schnell, wie es das entsprechende Datenspeichergerät zulässt, obwohl die Priorität von E/A-Operationen unter der von normalen Prozessen liegt. E/A-Operationen können sich negativ auf die Systemleistung auswirken, obwohl das Dateisystem während der Bereinigung weiterhin verwendbar sein und kurze Antwortzeiten aufweisen sollte. Explizite Bereinigungen können mit dem Befehl `zpool scrub` ausgeführt werden. Beispiel:

```
# zpool scrub tank
```

Der Status des aktuellen Bereinigungsvorgangs kann in der Ausgabe des Befehls `zpool status` angezeigt werden. Beispiel:

```
# zpool status -v tank
  pool: tank
 state: ONLINE
 scrub: scrub completed after 0h7m with 0 errors on Tue Sep  1 09:20:52 2009
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0

errors: No known data errors
```

Bitte beachten Sie, dass pro Pool gleichzeitig nur ein aktiver Bereinigungsvorgang ausgeführt werden kann.

Mithilfe der Option `-s` können Sie einen laufenden Bereinigungsvorgang anhalten. Beispiel:

```
# zpool scrub -s tank
```

In den meisten Fällen sollte ein Bereinigungsvorgang vollständig abgeschlossen werden, da die Datenintegrität nur so bestätigt werden kann. Falls sich ein Bereinigungsvorgang negativ auf die Systemleistung auswirkt, können Sie ihn jederzeit abbrechen.

Durch regelmäßige Bereinigungen wird auch ein kontinuierlicher E/A-Datenverkehr mit allen Datenträgern des Systems gewährleistet. Ein Nebeneffekt der Bereinigung besteht darin, dass die Stromverwaltung im Leerlauf befindliche Datenträger nicht in den Energiesparmodus schalten kann. Wenn das System im Allgemeinen kontinuierlich E/A-Opeartionen ausführt oder Stromverbrauch kein Problem ist, kann dieser Aspekt unberücksichtigt bleiben.

Weitere Informationen zur Interpretation der Ausgabe des Befehls `zpool status` finden Sie unter „Abfragen des Status von ZFS-Speicher-Pools“ auf Seite 95.

Datenbereinigung und Resilvering in ZFS

Nach dem Ersetzen eines Datenspeichergeräts wird ein sog. „Resilvering” (Wiederaufspielen von Daten) durchgeführt, um Daten von unbeschädigten Kopien auf den neuen Datenträger zu kopieren. Dieser Vorgang ist eine Form der Datenträgerbereinigung. Aus diesem Grund kann in einem Pool gleichzeitig nur ein solcher Vorgang stattfinden. Bei einem laufenden Bereinigungsvorgang unterbricht das Resilvering die aktuell laufende Bereinigung und setzt sie nach seinem Abschluss fort.

Weitere Informationen zum Resilvering finden Sie unter „Anzeigen des Resilvering-Status“ auf Seite 311.

Erkennen von Problemen in ZFS

In den folgenden Abschnitten wird beschrieben, wie Sie Probleme in ZFS-Speicher-Pools und -Dateisystemen erkennen können.

Die folgenden Leistungsmerkmale dienen zur Problemerkennung in ZFS-Konfigurationen:

- Ausführliche Informationen zu ZFS-Speicher-Pools mit dem Befehl `zpool status`
- Pool- und Gerätefehler werden mit ZFS/FMA-Diagnosemeldungen gemeldet
- Frühere ZFS-Befehle, die Informationen zum Pool-Status geändert haben, können mit dem Befehl `zpool history` angezeigt werden.

Die meisten ZFS-Probleme können mit dem Befehl `zpool status` erkannt werden. Dieser Befehl analysiert die verschiedenen Fehlfunktionen im System, erkennt die wichtigsten Probleme und zeigt Empfehlungen zu Abhilfemaßnahmen sowie für weitere Informationen Verweise auf entsprechende Artikel in der Sun Knowledge Base an. Bitte beachten Sie, dass der Befehl nur ein einziges Problem im Pool erkennen kann, obwohl mehrere Probleme vorhanden sein können. Bei Datenbeschädigungsfehlern wird beispielsweise stets vorausgesetzt, dass ein Datenspeichergerät ausgefallen ist. Durch Ersetzen des fehlerhaften Datenspeichergeräts ist das Problem der Datenbeschädigung noch nicht behoben.

ZFS enthält auch Diagnoseprogramm, das Pool- und Datenträgerausfälle diagnostiziert und meldet. Darüber hinaus werden auch mit solchen Pool- bzw. Datenträgerausfällen im Zusammhang stehende Prüfsummen-, E/A-, Geräte- und Poolfehler gemeldet. Von `fmd` gemeldete ZFS-Fehler werden auf der Konsole angezeigt und in der Systemprotokolldatei festgehalten. In den meisten Fällen leitet Sie die `fmd`-Meldung für weitere Abhilfeanweisungen an den Befehl `zpool status` weiter.

Der grundlegende Problembehebungsvorgang läuft wie folgt ab:

- Suchen Sie, falls möglich, mit dem Befehl `zpool history` die früheren ZFS-Befehle, die zur Fehlersituation geführt haben. Beispiel:

  ```
  # zpool history tank
  History for 'tank':
  2009-09-01.09:26:15 zpool create tank mirror c0t1d0 c0t2d0 c0t3d0
  2009-09-01.09:26:34 zfs create tank/erick
  2009-09-01.09:26:41 zfs set checksum=off tank/erick
  ```

  Bitte beachten Sie, dass in der obigen Befehlsausgabe die Prüfsummen für das Dateisystem `tank/erick` deaktiviert sind. Diese Konfiguration wird nicht empfohlen.
- Suchen Sie die Fehler in den `fmd`-Meldungen, die in der Systemkonsole bzw. den Dateien `/var/adm/messages` angezeigt werden.
- Weitere Reparaturanweisungen finden Sie im Befehl `zpool status -x`
- Führen Sie Reparaturen durch, z. B.:
  - Ersetzen und Inbetriebnehmen des fehlerhaften bzw. fehlenden Datenspeichergeräts
  - Wiederherstellen der fehlerhaften Konfiguration bzw. der beschädigten Daten aus einer Sicherungskopie
  - Überprüfen der Wiederherstellungsmaßnahme mit dem Befehl `zpool status - x`
  - Anlegen einer Sicherungskopie der wiederhergetsellten Konfiguration (falls möglich)

In diesem Kapitel wird beschrieben, wie Sie die Ausgabe des Befehls `zpool status` interpretieren können, damit Sie den Fehlertyp erkennen und einem der folgenden Abschnitte Reparaturanweisungen entnehmen können. Obwohl die meisten Aufgaben vom Befehl automatisch ausgeführt werden, müssen Sie genau wissen, welche Probleme erkannt wurden, damit Sie den richtigen Fehlertyp finden können.

Ermitteln, ob in einem ZFS-Speicher-Pool Probleme vorhanden sind

Mit dem Befehl `zpool status -x` können Sie am einfachsten herausfinden, ob in einem System Probleme bestehen. Dieser Befehl zeigt nur Pools an, in denen Probleme auftreten. Wenn in einem System alle Pools ordnungsgemäß funktionieren, zeigt der Befehl nur eine Meldung wie die folgende an:

```
# zpool status -x
all pools are healthy
```

Ohne das Flag `-x` zeigt der Befehl die gesamten Statusinformationen aller Pools (oder eines in der Befehlszeile angegebenen Pools) an, auch wenn diese ordnungsgemäß funktionieren.

Weitere Informationen zu Befehlszeilenoptionen des Befehls `zpool status` finden Sie unter „Abfragen des Status von ZFS-Speicher-Pools“ auf Seite 95.

Überprüfen der Ausgabe des Befehls `zpool status`

Die gesamte Ausgabe des Befehls `zpool status` sieht ungefähr wie folgt aus:

```
# zpool status tank
  pool: tank
 state: DEGRADED
status: One or more devices has been taken offline by the administrator.
        Sufficient replicas exist for the pool to continue functioning in a
        degraded state.
action: Online the device using 'zpool online' or replace the device with
        'zpool replace'.
 scrub: none requested
 config:

        NAME         STATE     READ WRITE CKSUM
        tank         DEGRADED     0     0     0
          mirror     DEGRADED     0     0     0
            c1t0d0   ONLINE       0     0     0
            c1t1d0   OFFLINE      0     0     0

errors: No known data errors
```

Diese Ausgabe ist in verschiedene Abschnitte unterteilt:

Gesamtinformationen zum Pool-Status

Dieser Abschnitt der Ausgabe des Befehls `zpool status` enthält die folgenden Felder (einige dieser Felder werden nur angezeigt, wenn im Pool Probleme auftreten):

`pool` Der Name des Pools.

`state` Der gegenwärtige Funktionsstatus des Pools. Diese Informationen geben nur die Fähigkeit des Pools zur Aufrechterhaltung der aktuellen Replikationsmethode an. Pools mit dem Status `ONLINE` können trotzdem fehlerhafte Datenspeichergeräte enthalten oder Datenbeschädigungen aufweisen.

`status` Eine Beschreibung des im Pool auftretenden Fehlers. Dieses Feld wird nicht angezeigt, wenn keine Probleme gefunden wurden.

`action` Eine empfohlene Aktion zur Fehlerbehebung. Diese Feld enthält eine Abkürzung, die auf eine der folgenden Abschnitte verweist. Dieses Feld wird nicht angezeigt, wenn keine Probleme gefunden wurden.

`see` Ein Verweis auf einen Artikel in der Sun Knowledge Base, der ausführliche Reparaturinformationen enthält. Online-Artikel werden öfter als dieses Handbuch aktualisiert und enthalten stets die aktuellsten Reparaturanweisungen. Dieses Feld wird nicht angezeigt, wenn keine Probleme gefunden wurden.

`scrub` Zeigt den aktuellen Status eines Bereinigungsvorgangs an (Datum und Uhrzeit der letzten Bereinigung, Informationen zu einer laufenden Bereinigung, Informationen zur Anforderung von Bereinigungen).

`errors` Zeigt bekannte bzw. unbekannte Datenfehler an.

Pool-Konfigurationsinformationen

Das Feld `config` in der Ausgabe des Befehls `zpool status` beschreibt die Konfigurationsstruktur der Datenspeichergeräte, die den Pool bilden, sowie deren Status und die von diesen Geräten herrührenden Fehler. Der Status kann die folgenden Werte annehmen: `ONLINE`, `FAULTED`, `DEGRADED`, `UNAVAILABLE` oder `OFFLINE`. Wenn der Zustand eines Pools nicht `ONLINE` ist, wurde die Fehlertoleranz des Pools eingeschränkt.

Im zweiten Abschnitt der Konfigurationsinformationen wird die Fehlerstatistik angezeigt. Diese Fehler werden in drei Kategorien eingeteilt:

- `READ` – E/A-Fehler während einer Leseoperation.
- `WRITE` – E/A-Fehler während einer Schreiboperation.
- `CKSUM` – Prüfsummenfehler. Das Datenspeichergerät gab im Verlauf einer Leseoperation beschädigte Daten zurück.

Mithilfe dieser Kategorien kann ermittelt werden, ob es sich um bleibende Schäden handelt. Eine geringe Anzahl auftretender E/A-Fehler kann die Folge zeitweiliger Ausfälle sein, während eine größere Anzahl an E/A-Fehlern auf ein bleibendes Problem mit dem entsprechenden Gerät hinweisen kann. Diese Fehler entsprechen nicht notwendigerweise Datenbeschädigungen. Wenn das Gerät zu einer redundanten Konfiguration gehört, kann es sein, dass die Datenträger nicht behebbare Fehler aufweisen, währenddessen auf der RAID-Z-

bzw. Datenspiegelungsebene keine Fehler angezeigt werden. In solchen Fällen bedeutet dies, dass ZFS unbeschädigte Daten erfolgreich angerufen und die beschädigten Daten durch diese Daten aus vorhandenen Replikationen ersetzt hat.

Weitere Informationen zur Interpretation dieser Fehler zum Ermitteln von Geräteproblemen finden Sie unter „Ermitteln des Gerätefehlertyps“ auf Seite 303.

Zusätzliche hilfreiche Informationen werden in der letzten Spalte der Ausgabe des Befehls `zpool status` angezeigt. Diese Information ergänzen die im Feld `state` enthaltenen Informationen und helfen bei der Ermittlung von Fehlerzuständen. Bei Datenspeichergeräten mit dem Status `FAULTED` zeigt dieses Feld an, ob auf das betreffende Gerät zugegriffen werden kann oder die Daten auf dem Gerät beschädigt sind. Wenn auf das Datenspeichergerät mithilfe von Resilvering Daten neu aufgespielt werden, zeigt dieses Feld den Verlauf dieses Vorgangs an.

Weitere Informationen zur Überwachung des Resilvering-Vorgangs finden Sie in „Anzeigen des Resilvering-Status“ auf Seite 311.

Status eines Bereinigungsvorgangs

Im dritten Abschnitt der Ausgabe des Befehls `zpool status` wird der aktuelle Status von Bereinigungsvorgängen angezeigt, die explizit ausgeführt werden. Diese Informationen weisen nicht auf Fehler hin, die im System aufgetreten sind, können aber zum Ermitteln der Genauigkeit des Meldens von Datenbeschädigungsfehlern herangezogen werden. Wenn die letzte Bereinigung erst vor kurzem ausgeführt wurde, sind Datenbeschädigungen höchstwahrscheinlich bereits bekannt.

Weitere Informationen zur Datenbereinigung und zur Interpretation dieser Informationen finden Sie unter „Überprüfen der ZFS-Datenintegrität“ auf Seite 293.

Datenbeschädigungsfehler

Der Befehl `zpool status` zeigt auch an, ob im Zusammenhang mit einem Pool bekannte Fehler aufgetreten sind. Diese Fehler können während der Bereinigung eines Datenträgers oder im Normalbetrieb gefunden worden sein. ZFS unterhält ein ständiges Protokoll aller mit dem Pool im Zusammenhang stehenden Datenfehler. Nach jedem Abschluss einer vollständigen Systembereinigung wird das Protokoll entsprechend aktualisiert.

Datenbeschädigungsfehler sind stets schwerwiegend. Deren Vorhandensein weist darauf hin, dass bei mindestens einem Anwendungsprogramm aufgrund beschädigter Daten im Pool ein E/A-Fehler aufgetreten ist. Gerätefehler innerhalb eines redundanten Pools verursachen keine Datenbeschädigung und werden in diesem Protokoll nicht festgehalten. Standardmäßig wird nur die Anzahl der gefundenen Fehler angezeigt. Eine vollständige Liste mit Fehlern und deren Informationen kann mit dem Befehl `zpool status -v` angezeigt werden. Beispiel:

```
# zpool status -v
  pool: tank
 state: UNAVAIL
status: One or more devices are faulted in response to IO failures.
action: Make sure the affected devices are connected, then run 'zpool clear'.
   see: http://www.sun.com/msg/ZFS-8000-HC
 scrub: scrub completed after 0h0m with 0 errors on Tue Sep  1 09:51:01 2009
config:

        NAME        STATE     READ WRITE CKSUM
        tank        UNAVAIL      0     0     0  insufficient replicas
          c1t0d0    ONLINE       0     0     0
          c1t1d0    UNAVAIL      4     1     0  cannot open

errors: Permanent errors have been detected in the following files:

/tank/data/aaa
/tank/data/bbb
/tank/data/ccc
```

Eine ähnliche Meldung wird auch von `fmd` auf der Systemkonsole angezeigt und in der Datei `/var/adm/messages` protokolliert. Diese Meldungen können auch mit dem Befehl `fmdump` verfolgt werden.

Weitere Informationen zur Interpretation von Datenbeschädigungsfehlern finden Sie unter „Ermitteln der Art der Datenbeschädigung“ auf Seite 314.

Systemprotokoll mit ZFS-Fehlermeldungen

Neben dem ständigen Protokollieren von Fehlern innerhalb eines Pools zeigt ZFS beim Auftreten bestimmter Ereignisse auch Systemprotokollmeldungen an. In den folgenden Situationen werden Ereignisse zur Benachrichtigung des Administrators ausgelöst:

- **Statusübergang eines Datenspeichergeräts** – Wenn ein Datenspeichergerät in den Status `FAULTED` übergeht, protokolliert ZFS eine Meldung, die darauf hinweist, dass die Fehlertoleranz des Pools beeinträchtigt werden könnte. Eine ähnliche Meldung wird gesendet, wenn das Gerät später wieder in Betrieb genommen und die ordnungsgemäße Pool-Funktion wiederhergestellt wird.
- **Datenbeschädigung** – Beim Erkennen von Datenbeschädigungen protokolliert ZFS eine Meldung, die beschreibt, wann und wo die Datenbeschädigung erkannt wurde. Diese Meldung wird nur beim allerersten Auftreten des Ereignisses protokolliert. Ein nachfolgendes Auftreten dieser Ereignisses löst keine Meldung mehr aus.

- **Pool- und Geräteausfälle** – Beim Auftreten von Pool- bzw. Geräteausfällen meldet der Fehlerverwaltungsdämon diese Fehler mithilfe von Systemprotokollmeldungen und mit dem Befehl `fmdump`.

Wenn ZFS einen Gerätefehler erkennt und diesen automatisch behebt, wird keine Benachrichtigung gesendet. Solche Fehler stellen keine Einschränkung der Pool-Redundanz bzw. Datenintegrität dar und sind darüber hinaus normalerweise die Folge eines Treiberproblems mit eigenen entsprechenden Fehlermeldungen.

Reparieren einer beschädigten ZFS-Konfiguration

ZFS unterhält im Root-Dateisystem einen Cache aktiver Pools und ihrer Konfiguration. Wenn diese Datei beschädigt wird bzw. nicht mehr mit den auf Datenträger gespeicherten Daten übereinstimmt, kann auf einen Pool nicht mehr zugegriffen werden. ZFS versucht, eine solche Situation zu vermeiden, obwohl wegen der Eigenschaften des zugrundeliegenden Dateisystems und Speicherplatzes Daten immer beschädigt werden können. Solche Situationen führen normalerweise zu einem Verschwinden des Pools aus dem System, wenn er verfügbar sein sollte und können sich auch als teilweise Konfigurationen, denen eine unbekannte Anzahl an virtuellen Geräten der obersten Hierarchie fehlt, manifestieren. In allen Fällen kann die Konfiguration durch Exportieren des Pools (falls er zugänglich ist) und anschließendes Importieren wiederhergestellt werden.

Weitere Informationen zum Importieren und Exportieren von Pools finden Sie unter „Migrieren von ZFS-Speicher-Pools“ auf Seite 105.

Reparieren eines ausgefallenen Geräts

Wenn auf ein Gerät nicht zugegriffen werden kann, wird es in der Ausgabe des Befehls `zpool status` mit dem Status `UNAVAILABLE` angezeigt. Dieser Status bedeutet, dass ZFS beim ersten Zugriff auf den Pool nicht auf das betreffende Datenspeichergerät zugreifen konnte oder es seitdem nicht mehr verfügbar ist. Wenn durch dieses Datenspeichergerät ein virtuelles Gerät der obersten Hierarchieebene nicht mehr verfügbar ist, können von diesem Pool keine Daten abgerufen werden. Andernfalls kann die Fehlertoleranz des Pools beeinträchtigt werden. In jedem Fall muss das Gerät zum Wiederherstellen des Normalbetriebs wieder funktionsfähig in das System integriert werden.

Nach einem Geräteausfall wird von `fmd` in etwa die folgende Meldung angezeigt:

```
SUNW-MSG-ID: ZFS-8000-FD, TYPE: Fault, VER: 1, SEVERITY: Major
EVENT-TIME: Tue Sep  1 09:36:46 MDT 2009
PLATFORM: SUNW,Sun-Fire-T200, CSN: -, HOSTNAME: neo
SOURCE: zfs-diagnosis, REV: 1.0
```

```
EVENT-ID: a1fb66d0-cc51-cd14-a835-961c15696fed
DESC: The number of I/O errors associated with a ZFS device exceeded
acceptable levels.  Refer to http://sun.com/msg/ZFS-8000-FD for more information.
AUTO-RESPONSE: The device has been offlined and marked as faulted.  An attempt
will be made to activate a hot spare if available.
IMPACT: Fault tolerance of the pool may be compromised.
REC-ACTION: Run 'zpool status -x' and replace the bad device.
```

Der nächste Schritt besteht darin, mithilfe des Befehls `zpool status -x` ausführliche Informationen zum Geräteproblem und eine mögliche Lösung anzuzeigen. Beispiel:

Diese Befehlsausgabe zeigt, dass das fehlende Gerät `c1t1d0` nicht funktioniert. Ersetzen Sie das Gerät, wenn das Laufwerk fehlerhaft ist.

Nehmen Sie das ersetzte Gerät dann mit dem Befehl `zpool online` in Betrieb. Beispiel:

```
# zpool online tank c1t1d0
```

Bestätigen Sie, dass der Pool mit dem ersetzten Datenspeichergerät ordnungsgemäß funktioniert.

```
# zpool status -x tank
pool 'tank' is healthy
```

Wiedereingliedern eines Datenspeichergeräts in das System

Die Art und Weise, wie ein fehlendes Datenspeichergerät wieder in ein System eingegliedert wird, hängt vom jeweiligen Gerät ab. Wenn auf das Gerät über das Netzwerk zugegriffen wird, muss die Netzwerkverbindung wiederhergestellt werden. Wenn das Datenspeichergerät ein USB-Stick oder ein anderer Wechsel-Datenträger ist, muss es wieder an das System angeschlossen werden. Wenn es sich bei dem betreffenden Gerät um eine lokale Festplatte handelt, kann es sein, dass es aufgrund eines Controller-Ausfalls für das System unsichtbar ist. In diesem Fall muss der Controller ausgetauscht werden, sodass die betreffenden Datenträger wieder verfügbar sind. Es existieren noch andere Fehlersituationen; diese hängen von der Art der Hardware und ihrer Konfiguration ab. Wenn ein Laufwerk ausfällt und dieses für das System nicht mehr sichtbar ist (ein äußerst umwahrscheinliches Ereignis), muss das Datenspeichergerät als beschädigt eingestuft werden. Folgen Sie der unter „Ersetzen oder Reparieren eines beschädigten Geräts“ auf Seite 303 beschriebenen Vorgehensweise.

Benachrichtigen von ZFS zur Geräteverfügbarkeit

Nach dem Wiedereingliedern eines Datenspeichergeräts in das System erkennt ZFS unter Umständen seine Verfügbarkeit automatisch; dies muss aber nicht so ein. Wenn sich der Pool vorher im Zustand „FAULTED" befand oder das System im Rahmen der Wiedereingliederung neu gestartet wurde, sucht ZFS beim Zugreifen auf den Pool automatisch alle Datenspeichergeräte ab. Wenn sich der Pool im Zustand „DEGRADED" befand und das Gerät während des Systembetriebs ausgetauscht wurde, müssen Sie ZFS benachrichtigen, dass das Gerät jetzt verfügbar ist und darauf mithilfe des Befehls `zpool online` zugegriffen werden kann. Beispiel:

```
# zpool online tank c0t1d0
```

Weitere Informationen zum Inbetriebnehmen von Geräten finden Sie unter „Inbetriebnehmen eines Gerätes" auf Seite 85.

Ersetzen oder Reparieren eines beschädigten Geräts

In diesem Abschnitt wird beschrieben, wie die verschiedenen Fehlertypen eines Datenspeichergerätes ermittelt, vorübergehende Fehler gelöscht und Geräte ausgetauscht werden können.

Ermitteln des Gerätefehlertyps

Der Begriff *beschädigtes Gerät* ist nicht klar umrissen und kann verschiedene mögliche Situationen beschreiben:

- **Bitfäule** – Mit der Zeit können äußere Einflüsse wie Magnetfelder und kosmische Strahlung dazu führen, dass auf Datenträgern gespeicherte Bits unvorhergesehene Werte annehmen. Solche Ereignisse sind relativ selten, treten aber häufig genug auf, um potenzielle Datenbeschädigung in größeren und lange laufenden Systemen zu verursachen. Solche Fehler sind normalerweise vorübergehend.
- **Fehlgeleitete Lese- bzw. Schreiboperationen** – Firmware- bzw. Hardwarefehler können dazu führen, dass Lese- und Schreiboperationen ganzer Datenblöcke den falschen Bereich auf dem Datenträger referenzieren. Diese Fehler sind normalerweise vorübergehend, obwohl eine große Anzahl solcher Fehler auf ein fehlerhaftes Laufwerk hinweisen kann.
- **Administratorfehler** – Administratoren können versehentlich Datenträgerbereiche mit ungültigen Daten überschreiben (z. B. das Kopieren von `/dev/zero` auf bestimmte Datenträgerbereiche) und so Daten auf dem Datenträger dauerhaft beschädigen. Solche Fehler sind stets vorübergehend.

- **Zeitweilige Ausfälle**– Datenträger können zeitweilig ausfallen, weswegen E/A-Operationen fehlschlagen. Diese Situation tritt normalerweise bei Datenspeichergeräten auf, auf die über das Netzwerk zugegriffen wird, obwohl auch bei lokalen Datenträgern solche Ausfälle auftreten können. Solche Fehler sind können vorübergehend sein.
- **Fehlerhafte bzw. unzuverlässig arbeitende Hardware** – Unter diese Kategorie fallen alle durch fehlerhafte Hardware verursachten Probleme. Dies können dauerhafte E/A-Fehler, falscher Datentransport und daraus folgende regellose Datenbeschädigung sowie eine Reihe anderer Fehler sein. Solche Fehler sind normalerweise dauerhaft.
- **Außer Betrieb genommene Datenspeichergeräte** – Wenn ein Gerät außer Betrieb genommen wurde (Status „OFFLINE"), wird angenommen, dass es der Administrator in diesen Zustand versetzt hat, weil es als fehlerhaft vermutet wird. Der Administrator, der das Gerät in diesen Zustand versetzt hat, kann überprüfen, ob diese Annahme richtig ist.

Die genaue Ermittlung von Fehlerursachen kann sich schwierig gestalten. Der erste Schritt besteht darin, die Fehlerzähler in der Ausgabe des Befehls `zpool status` wie folgt zu überprüfen:

```
# zpool status -v pool
```

Die Fehler werden in E/A- und Prüfsummenfehler unterteilt. Daraus lässt sich unter Umständen auf den Fehlertyp schließen. Im Normalbetrieb treten während einer langen Systemlaufzeit nur einige wenige Fehler auf. Wenn Sie eine hohe Fehleranzhal sehen, deutet dies wahrscheinlich auf einen bevorstehenden bzw. vollständigen Geräteausfall hin. Administratorfehler können jedoch ebenfalls zu hohen Fehleranzahlen führen. Eine andere Informationsquelle ist das Systemprotokoll. Wenn das Protokoll eine große Anzahl an Meldungen von SCSI- bzw. Fibre Channel-Treibern enthält, weist dies möglicherweise auf schwerwiegende Hardwareprobleme hin. Wenn keine Systemmeldungen protokolliert werden, ist der Schaden wahrscheinlich vorübergehend.

Das Ziel besteht in der Beantwortung der folgenden Frage:

Ist es wahrscheinlich, dass an diesem Gerät wieder ein Fehler auftritt?

Nur einmal auftretende Fehler werden als *vorübergehend* eingestuft und ziehen keine potenziellen Ausfälle nach sich. Fehler, die dauerhaft bzw. ernstlich genug sind, um potenzielle Hardwareausfälle auszulösen, werden als „schwerwiegend" eingestuft." Die Ermittlung von Fehlertypen sprengt den Rahmen der gegenwärtig mit ZFS verfügbaren automatisierten Softwarelösungen und muss manuell vom Administrator durchgeführt werden. Nach der Ermittlung des Fehlertyps können entsprechende Abhilfemaßnahmen ergriffen werden. Diese bestehen entweder im Löschen vorübergehender Fehler oder im Austauschen des betreffenden Datenspeichergeräts aufgrund schwerwiegender Fehler. Die entsprechenden Reparaturvorgänge werden in den nächsten Abschnitten erläutert.

Auch wenn Gerätefehler als vorübergehend eingestuft wurden, können sie trotzdem nicht mehr rückgängig zu machende Datenfehler im Pool verursacht haben. Diese Fehler erfordern auch dann spezielle Reparaturmaßnahmen, wenn das zugrunde liegende Datenspeichergerät ordnungsgemäß funktioniert oder anderweitig repariert wurde. Weitere Informationen zum Beseitigen von Datenfehlern finden Sie unter „Reparieren beschädigter Daten“ auf Seite 313.

Löschen vorübergehender Fehler

Wenn Gerätefehler als vorübergehend eingestuft wurden und die zukünftige ordnungsgemäße Funktion des betreffenden Datenspeichergeräts nicht beeinträchtigen, können die entsprechenden vorübergehenden Fehler gelöscht werden. Damit wird festgelegt, dass kein schwerwiegender Fehler aufgetreten ist. Mit dem Befehl `zpool clear` können Sie Fehlerzähler für Datenspeichergeräte mit RAID-Z- bzw. Datenspiegelungskonfigurationen zurücksetzen. Beispiel:

```
# zpool clear tank c1t1d0
```

Diese Syntax löscht alle Fehler und setzt alle Fehlerzähler zurück, die mit dem betreffenden Datenspeichergerät in Zusammenhang stehen.

Zum Löschen aller mit den virtuellen Geräten im Pool in Verbindung stehenden Fehler und Zurücksetzen aller Fehlerzähler dient die folgende Syntax:

```
# zpool clear tank
```

Weitere Informationen zum Löschen von Pool-Fehlern finden Sie unter „Löschen von Gerätefehlern im Speicher-Pool“ auf Seite 86.

Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool

Wenn ein Datenspeichergerät dauerhaft beschädigt ist bzw. ein solcher Schaden bevorsteht, muss es ausgetauscht werden. Ob das betreffende Gerät ersetzt werden kann, hängt von der Konfiguration ab.

Ermitteln, ob ein Gerät ausgetauscht werden kann

Damit ein Gerät ausgetauscht werden kann, muss sich der Pool im Status `ONLINE` befinden. Das betreffende Gerät muss zu einer redundanten Konfiguration gehören, oder es muss ordnungsgemäß funktionieren (d. h. sich im Status `ONLINE` befinden). Wenn der Datenträger zu einer redundanten Konfiguration gehört, müssen ausreichende Replikationen vorhanden sein, aus denen unbeschädigte Daten wiederhergestellt werden können. Wenn zwei Datenträger in einer vierfachen Datenspiegelungskonfiguration fehlerhaft sind, können beide Datenträger ausgetauscht werden, da gültige Datenreplikationen vorhanden sind. Wenn jedoch zwei Datenträger in einer vierfachen RAID-Z-Konfiguration fehlerhaft sind, kann keiner der beiden Datenträger ausgetauscht werden, da nicht genügend gültige Datenreplikationen verfügbar sind, von denen Daten wiederhergestellt werden können. Wenn das Gerät beschädigt, aber noch in Betrieb ist, kann es ausgetauscht werden, solange sich der Pool nicht im Status `FAULTED` befindet. Wenn nicht genügend Replikationen mit gültigen Daten verfügbar sind, werden jedoch auch die beschädigten Daten auf das neue Datenspeichergerät kopiert.

In der folgenden Konfiguration kann der Datenträger `c1t1d0` ausgetauscht werden, und die gesamten Daten im Pool werden von der ordnungsgemäßen Replikation `c1t0d0` kopiert.

```
    mirror            DEGRADED
    c1t0d0             ONLINE
    c1t1d0             FAULTED
```

Der Datenträger `c1t0d0` kann ebenfalls ausgetauscht werden, obwohl keine Datenselbstheilung erfolgen kann, da keine ordnungsgemäße Datenreplikation vorhanden ist.

In der folgenden Konfiguration kann keiner der fehlerhaften Datenträger ausgetauscht werden. Die Datenträger mit dem Status `ONLINE` können ebenfalls nicht ausgetauscht werden, da der Pool selbst fehlerhaft ist (Status „FAULTED”).

```
    raidz              FAULTED
    c1t0d0             ONLINE
    c2t0d0             FAULTED
    c3t0d0             FAULTED
    c3t0d0             ONLINE
```

In der folgenden Konfiguration können alle Datenträger der obersten Hierarchieebene ausgetauscht werden, obwohl auch auf den alten Datenträgern befindliche ungültige Daten auf den neuen Datenträger kopiert werden.

```
c1t0d0         ONLINE
c1t1d0         ONLINE
```

Wenn alle Datenträger fehlerhaft sind, kann nichts ausgetauscht werden, da dann der Pool selbst fehlerhaft ist.

Datenspeichergeräte, die nicht ausgetauscht werden können

Wenn ein Pool durch den Ausfall eines Datenspeichergeräts fehlerhaft wird oder das betreffende Gerät in einer nicht redundanten Konfiguration zuviele Datenfehler enthält, kann es nicht sicher ausgetauscht werden. Ohne ausreichende Redundanz sind keine entsprechenden gültigen Daten vorhanden, die die Fehler auf dem beschädigten Gerät beseitigen könnten. In einem solchen Fall kann der Pool nur gelöscht und die Konfiguration muss neu erstellt werden. Dadurch werden Daten im Verlauf dieses Vorgangs wiederhergestellt.

Weitere Informationen zum Wiederherstellen eines gesamten Pools finden Sie unter „Reparieren von Schäden am gesamten ZFS-Speicher-Pool“ auf Seite 316.

Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool

Wenn Sie ermittelt haben, dass ein Datenspeichergerät ausgetauscht werden kann, können Sie es mithilfe des Befehls `zpool replace` ersetzen. Mit dem folgenden Befehl ersetzen Sie ein beschädigtes Datenspeichergerät durch ein anderes Gerät:

```
# zpool replace tank c1t1d0 c2t0d0
```

Dieser Befehl beginnt die Migration von Daten vom beschädigten (oder, bei redundanten Konfigurationen, anderen) Datenspeichergerät auf das neue Gerät. Nach Abschluss des Befehls wird das beschädigte Datenspeichergerät von der Konfiguration abgetrennt und kann dann aus dem System entfernt werden. Wenn Sie das Datenspeichergerät bereits entfernt und an der gleichen Stelle durch ein neues Gerät ersetzt haben, sollten Sie das Einzelgeräteformat des Befehls verwenden. Beispiel:

```
# zpool replace tank c1t1d0
```

Dieser Befehl formatiert das neue Datenspeichergerät entsprechend und beginnt dann mithilfe von Resilvering mit dem Aufspielen von Daten aus der Konfiguration.

Weitere Informationen zum Befehl `zpool replace` finden Sie unter „Austauschen von Geräten in einem Speicher-Pool“ auf Seite 86.

BEISPIEL 11–1 Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool

Das folgende Beispiel zeigt, wie ein Gerät (`c1t3d0`) in dem Speicher-Pool mit Datenspiegelung `tank` auf einem Sun Fire x4500-System ersetzt wird. Wenn Sie die Festplatte `c1t3d0` durch eine neue Festplatte an derselben Position (`c1t3d0`) ersetzen, müssen Sie die Festplatte zunächst dekonfigurieren. Führen Sie dazu die folgenden Grundschritte durch:

- Nehmen Sie zuerst die zu ersetzende Festplatte außer Betrieb. Eine in Gebrauch befindliche Festplatte kann nicht dekonfiguriert werden.

BEISPIEL 11–1 Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool *(Fortsetzung)*

- Ermitteln Sie die betroffene Festplatte (`c1t3d0`) und entfernen Sie sie aus der Konfiguration. Der Pool wird durch die außer Betrieb genommene Festplatte in der Datenspiegelungskonfiguration in einen eingeschränkten Zustand versetzt, bleibt aber weiterhin verfügbar.
- Ersetzen Sie die Festplatte physisch (`c1t3d0`). Vergewissern Sie sich vor dem Ausbauen des fehlerhaften Laufwerks, dass die blaue Ausbaubereitschaft-LED leuchtet.
- Konfigurieren Sie die Festplatte (`c1t3d0`) erneut.
- Setzen Sie die Festplatte (`c1t3d0`) wieder in Betrieb.
- Führen Sie den Befehl `zpool replace` aus, um die Festplatte (`c1t3d0`) zu ersetzen.

Hinweis – Wenn Sie zuvor die Eigenschaft `autoreplace` des Pools auf „On“ gesetzt haben, wird jedes neue Gerät, das sich an der physischen Stelle eines zuvor zum Pool gehörenden Geräts befindet, automatisch formatiert und ohne den Befehl `zpool replace` ersetzt. Dieses Leistungsmerkmal wird möglicherweise nicht auf jeder Art von Hardware unterstützt.

- Wenn eine ausgefallene Festplatte automatisch durch eine Hot-Spare-Festplatte ersetzt wurde, müssen Sie die Hot-Spare-Festplatte möglicherweise nach dem Ersetzen der ausgefallenen Festplatte abtrennen. Wenn `c2t4d0` beispielsweise noch immer eine aktive Spare-Festplatte ist, nachdem die ausgefallene Festplatte ersetzt wurde, trennen Sie sie ab.

```
# zpool detach tank c2t4d0
```

```
# zpool offline tank c1t3d0
# cfgadm | grep c1t3d0
sata1/3::dsk/c1t3d0            disk         connected    configured   ok
# cfgadm -c unconfigure sata1/3
Unconfigure the device at: /devices/pci@0,0/pci1022,7458@2/pci11ab,11ab@1:3
This operation will suspend activity on the SATA device
Continue (yes/no)? yes
# cfgadm | grep sata1/3
sata1/3                        disk         connected    unconfigured ok
<Replace the physical disk c1t3d0>
# cfgadm -c configure sata1/3
# cfgadm | grep sata3/7
sata3/7::dsk/c5t7d0            disk         connected    configured   ok
# zpool online tank c1t3d0
# zpool replace tank c1t3d0
# zpool status
  pool: tank
 state: ONLINE
```

BEISPIEL 11–1 Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool *(Fortsetzung)*

```
 scrub: resilver completed after 0h0m with 0 errors on Tue Apr 22 14:44:46 2008
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0

errors: No known data errors
```

Beachten Sie, dass in der obigen `zpool`-Ausgabe sowohl die neue als auch die alte Festplatte unter *replacing* (wird ersetzt) angezeigt werden kann. Beispiel:

```
replacing     DEGRADED     0     0    0
  c1t3d0s0/o  FAULTED      0     0    0
  c1t3d0      ONLINE       0     0    0
```

Dieser Text bedeutet, dass der Austauschvorgang läuft und an der neuen Festplatte das Resilvering durchgeführt wird.

Wenn Sie eine Festplatte (`c1t3d0`) durch eine andere Festplatte (`c4t3d0`) ersetzen, müssen Sie den Befehl `zpool replace` erst nach dem physischen Austausch der Festplatte ausführen. Beispiel:

```
# zpool replace tank c1t3d0 c4t3d0
# zpool status
  pool: tank
 state: DEGRADED
 scrub: resilver completed after 0h0m with 0 errors on Tue Apr 22 14:54:50 2008
config:

        NAME           STATE     READ WRITE CKSUM
        tank           DEGRADED     0     0     0
          mirror       ONLINE       0     0     0
            c0t1d0     ONLINE       0     0     0
            c1t1d0     ONLINE       0     0     0
          mirror       ONLINE       0     0     0
            c0t2d0     ONLINE       0     0     0
```

BEISPIEL 11–1 Austauschen eines Datenspeichergeräts in einem ZFS-Speicher-Pool *(Fortsetzung)*

```
            c1t2d0      ONLINE       0     0     0
          mirror        DEGRADED     0     0     0
            c0t3d0      ONLINE       0     0     0
            replacing   DEGRADED     0     0     0
              c1t3d0    OFFLINE      0     0     0
              c4t3d0    ONLINE       0     0     0

errors: No known data errors
```

Mitunter muss der Befehl `zpool status` mehrmals ausgeführt werden, bevor der Austauschvorgang abgeschlossen ist.

```
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Tue Apr 22 14:54:50 2008
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror    ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c4t3d0  ONLINE       0     0     0
```

BEISPIEL 11–2 Ersetzen eines fehlgeschlagenen Protokolliergeräts

Das folgende Beispiel zeigt die Wiederherstellung des Normalzustands nach Fehlschlag eines Protokolliergeräts `c0t5d0` im Speicher-Pool, `pool`. Führen Sie dazu die folgenden Grundschritte durch:

- Überprüfen Sie die Ausgabe des Befehls `zpool status -x` und die FMA-Diagnosemeldung, beschrieben unter:

  `http://www.sun.com/msg/ZFS-8000-K4`
- Ersetzen Sie das fehlgeschlagene Protokolliergerät physisch.
- Nehmen Sie das Protokolliergerät in Betrieb.
- Löschen Sie den Fehlerzustand des Pools.

BEISPIEL 11–2 Ersetzen eines fehlgeschlagenen Protokolliergeräts *(Fortsetzung)*

```
# zpool status -x
  pool: pool
 state: FAULTED
status: One or more of the intent logs could not be read.
        Waiting for adminstrator intervention to fix the faulted pool.
action: Either restore the affected device(s) and run 'zpool online',
        or ignore the intent log records by running 'zpool clear'.
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        pool        FAULTED      0     0     0 bad intent log
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
        logs        FAULTED      0     0     0 bad intent log
          c0t5d0    UNAVAIL      0     0     0 cannot open
<Physically replace the failed log device>
# zpool online pool c0t5d0
# zpool clear pool
```

Anzeigen des Resilvering-Status

Je nach Kapazität des Datenträgers und Datenmenge im Pool kann das Austauschen eines Datenspeichergeräts eine geraume Zeit in Anspruch nehmen. Der Vorgang des Übertragens von Daten von einem Datenspeichergerät auf ein anderes Gerät nennt man *Resilvering*. Er kann mithilfe des Befehls `zpool status` überwacht werden.

Herkömmliche Dateisysteme kopieren Daten auf Datenblockebene. Da in ZFS die künstliche Schicht des Volume Managers beseitigt wurde, kann das Resilvering hier viel leistungsfähiger und kontrollierter durchgeführt werden. Die beiden Hauptvorteile dieser Funktion bestehen in Folgendem:

- ZFS kopiert beim Resilvering nur die Mindestmenge erforderlicher Daten. Im Falle eines kurzen Ausfalls (im Gegensatz zu einem vollständigen Austausch von Datenspeichergeräten) können Daten innerhalb weniger Minuten bzw. Sekunden auf den gesamten Datenträger aufgespielt werden, und es ist kein Kopieren des gesamten Datenträgers erforderlich. Der gesamte Vorgang wird auch nicht durch Protokollierung „verunreinigter" Datenbereiche, die manche Softwarepakete zur Datenträgerverwaltung unterstützen, verkompliziert. Nach dem Austauschen eines Datenträgers ist der Zeitraum, den das Wiederaufspielen von Daten benötigt, proportional zur Datenmenge auf dem betreffenden Datenträger. Das Austauschen eines 500 GB-Datenträgers braucht nur Sekunden zu dauern, wenn im Pool nur einige wenige GB mit Daten belegt sind.

- Das Resilvering kann unterbrochen werden und ist sicher. Wenn am System ein Stromausfall auftritt oder es neu gestartet wird, fährt der Resilvering-Vorgang an genau der Stelle fort, wo er unterbrochen wurde, ohne dass dazu manuelle Eingriffe erforderlich sind.

Der Status des Resilvering-Vorgangs kann mit dem Befehl `zpool status` angezeigt werden. Beispiel:

```
# zpool status tank
  pool: tank
 state: ONLINE
status: One or more devices is currently being resilvered.  The pool will
        continue to function, possibly in a degraded state.
action: Wait for the resilver to complete.
 scrub: resilver in progress for 0h2m, 16.43% done, 0h13m to go
config:
        NAME                  STATE     READ WRITE CKSUM
        tank                  DEGRADED     0     0     0
          mirror              DEGRADED     0     0     0
            replacing         DEGRADED     0     0     0
              c1t0d0          ONLINE       0     0     0
              c2t0d0          ONLINE       0     0     0
            c1t1d0            ONLINE       0     0     0
```

In diesem Beispiel wird der Datenträger `c1t0d0` durch das Gerät `c2t0d0` ersetzt. Dieses Ereignis wird in der Statusausgabe durch das Wort *replacing* in der Spalte für virtuelle Gerätenamen in der Konfiguration angegeben. Bei diesem Gerät handelt es sich um keinen Datenträger im eigentlichen Sinne, und Sie können mit diesem virtuellen Gerätetyp auch keinen Pool erstellen. Der Zweck dieses Gerätenamens besteht lediglich im Anzeigen des Resilvering-Vorgangs, sodass genau zu sehen ist, welcher Datenträger ausgetauscht wird.

Bitte beachten Sie, dass ein Pool, in dem ein Resilvering stattfindet, in den Status `ONLINE` oder `DEGRADED` gebracht wird, da es während des Resilvering-Vorgangs nicht das erforderliche Niveau an Datenredundanz gewährleisten kann. Das Resilvering findet so schnell wie möglich statt, obwohl die damit verbundenen E/A-Prozesse zur Minimierung der Beeinträchtigung der Systemleistung stets eine niedrigere Priorität als E/A-Benutzeranforderungen haben. Nach dem Abschluss des Resilvering wird das System in die neue und vollständige Konfiguration gebracht. Beispiel:

```
# zpool status tank
  pool: tank
 state: ONLINE
scrub: resilver completed after 0h0m with 0 errors on Tue Sep  1 10:55:54 2009
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
```

```
          mirror    ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0

errors: No known data errors
```

Der Pool befindet sich jetzt wieder im Status `ONLINE`, und der ursprüngliche beschädigte Datenträger (`c1t0d0`) wuerde aus der Konfiguration entfernt.

Reparieren beschädigter Daten

In den folgenden Abschnitten wird beschrieben, wie Sie den Typ der Datenbeschädigung ermitteln und (falls möglich) Daten reparieren können.

- „Ermitteln der Art der Datenbeschädigung“ auf Seite 314
- „Reparatur beschädigter Dateien bzw. Verzeichnisse“ auf Seite 315
- „Reparieren von Schäden am gesamten ZFS-Speicher-Pool“ auf Seite 316

Zur Minimierung möglicher Datenbeschädigungen nutzt ZFS Prüfsummenberechnung, Redundanz und Datenselbstheilung. Wenn ein Pool keine redundante Konfiguration besitzt, können jedoch trotzdem Datenbeschädigungen auftreten, wenn diese auftraten, als sich der Pool im Status „DEGRADED” befand oder eine unvorhergesehene Ereignisabfolge mehrere Datenkopien beschädigt hat. Unabhängig von der Ursache ist das Ergebnis gleich: Daten werden beschädigt und sind deswegen nicht mehr verfügbar. Die erforderlichen Abhilfemaßnahmen hängen von der Art der beschädigten Daten und deren relativem Wert ab. Es können zwei grundlegende Arten von Daten beschädigt werden:

- Pool-Metadaten – Damit Pools geöffnet werden können und auf Datasets zugegriffen werden kann, muss ZFS eine Reihe spezieller Daten verarbeiten. Wenn diese Daten beschädigt sind, stehen der gesamte Pool bzw. ein Teil der Datasets nicht mehr zur Verfügung.
- Objektdaten – In diesem Fall sind Daten innerhalb einer bestimmten Datei bzw. eines Verzeichnisses beschädigt. Dieses Problem kann dazu führen, dass auf einen Teil dieser Datei bzw. dieses Verzeichnisses nicht mehr zugegriffen werden kann oder das betreffende Objekt in seiner Gesamtheit nicht mehr verfügbar ist.

Daten werden während des Normalbetriebs und der Datenbereinigung („Scrubbing”) auf Integrität überprüft. Weitere Informationen zum Überprüfen der Integrität von Pool-Daten finden Sie unter „Überprüfen der ZFS-Datenintegrität“ auf Seite 293.

Ermitteln der Art der Datenbeschädigung

Der Befehl `zpool status` zeigt standardmäßig nur an, dass eine Datenbeschädigung aufgetreten ist, es ist jedoch nicht ersichtlich, wo sie sich ereignet hat. Beispiel:

```
# zpool status
   pool: monkey
state: ONLINE
status: One or more devices has experienced an error resulting in data
         corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
         entire pool from backup.
    see: http://www.sun.com/msg/ZFS-8000-8A
scrub: none requested
config:

         NAME        STATE     READ WRITE CKSUM
         monkey      ONLINE       0     0     0
           c1t1d0s6  ONLINE       0     0     0
           c1t1d0s7  ONLINE       0     0     0

errors: 8 data errors, use '-v' for a list
```

In jeder Zeile wird nur angezeigt, dass zu einem bestimmten Zeitpunkt ein Fehler aufgetreten ist. Diese Fehler müssen nicht mehr notwendigerweise im System vorhanden sein. Unter Normalbedingungen trifft diese Situation zu. Bestimmte zeitweilige Ausfälle können zu Datenbeschädigungen führen, die nach dem Ende des Ausfalls automatisch behoben werden. Ein vollständiger Bereinigungsvorgang im Pool untersucht jeden aktiven Datenblock im Pool, und das Fehlerprotokoll wird nach Abschluss der Bereinigung geleert. Wenn Sie sehen, dass die betreffenden Fehler im Pool nicht mehr auftreten und Sie nicht auf den Abschluss des Bereinigungsvorgangs warten möchten, können Sie alle Fehler im Pool mithilfe des Befehls `zpool online` löschen.

Wenn die Datenbeschädigung in Metadaten für den gesamten Pool aufgetreten ist, sieht die Befehlsausgabe etwas anders aus. Beispiel:

```
# zpool status -v morpheus
  pool: morpheus
    id: 1422736890544688191
 state: FAULTED
status: The pool metadata is corrupted.
action: The pool cannot be imported due to damaged devices or data.
   see: http://www.sun.com/msg/ZFS-8000-72
config:
```

```
        morpheus    FAULTED   corrupted data
          c1t10d0   ONLINE
```

Im Falle eines beschädigten Pools wird dieser in den Status FAULTED gebracht, da er nicht das erforderliche Niveau an Datenredundanz gewährleisten kann.

Reparatur beschädigter Dateien bzw. Verzeichnisse

Bei beschädigten Dateien bzw. Verzeichnissen kann es je nach Art der Datenbeschädigung sein, dass das System noch immer funktioniert. Schäden sind praktisch nicht wieder rückgängig zu machen, wenn im System nirgendwo ordnungsgemäße Kopien der Daten vorhanden sind. Wenn die betreffenden Daten wertvoll sind, können Sie diese nur aus Sicherungskopien wiederherstellen. Auch in diesem Fall kann es sein, dass Sie nach dieser Datenbeschädigung den Normalbetrieb wiederherstellen können, ohne den gesamten Pool wiederherstellen zu müssen.

Wenn sich der Schaden innerhalb eines Dateidatenblocks befindet, kann die betreffende Datei sicher gelöscht werden, wodurch der Fehler aus dem System entfernt wird. Mit dem Befehl `zpool status -v` können Sie sich eine Liste von Dateinamen mit dauerhaften Fehlern ausgeben lassen. Beispiel:

```
# zpool status -v
   pool: monkey
state: ONLINE
status: One or more devices has experienced an error resulting in data
         corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
         entire pool from backup.
    see: http://www.sun.com/msg/ZFS-8000-8A
scrub: none requested
config:

         NAME        STATE     READ WRITE CKSUM
         monkey      ONLINE       0     0     0
           c1t1d0s6  ONLINE       0     0     0
           c1t1d0s7  ONLINE       0     0     0

errors: Permanent errors have been detected in the following files:

/monkey/a.txt
/monkey/bananas/b.txt
/monkey/sub/dir/d.txt
/monkey/ghost/e.txt
/monkey/ghost/boo/f.txt
```

Die Bedeutung dieser obigen Ausgabe ist wie folgt:

- Wenn der vollständige Pfad zur Datei gefunden wurde und der Dataset eingehängt ist, wird der vollständige Pfad zur Datei angezeigt. Beispiel:

  ```
  /monkey/a.txt
  ```

- Wenn der vollständige Pfad zur Datei gefunden wurde, das Dataset jedoch nicht eingehängt ist, wird der Dataset-Name ohne vorangestellten Schrägstrich (/), gefolgt vom Pfad zur Datei innerhalb des Datasets angezeigt. Beispiel:

  ```
  monkey/ghost/e.txt
  ```

- Wenn die Objektnummer zu einem Dateipfad nicht erfolgreich konvertiert werden kann, weil ein Fehler auftrat oder dem Objekt kein realer Pfad zugewiesen ist (z. B. bei `dnode_t`), dann wird der Dataset-Name, gefolgt von der Objektnummer angezeigt. Beispiel:

  ```
  monkey/dnode:<0x0>
  ```

- Wenn ein Objekt im Metaobjekt-Set (MOS) beschädigt ist, wird ein spezielles Tag `<metadata>`, gefolgt von der Objektnummer angezeigt.

Wenn Metadaten einer Datei bzw. eines Verzeichnisses beschädigt sind, kann die betreffende Datei nur an einen anderen Ort verschoben werden. Sie können Dateien bzw. Verzeichnisse sicher an eine unkritische Stelle kopieren, sodass das ursprüngliche Objekt am Ausgangsort wiederhergestellt werden kann.

Reparieren von Schäden am gesamten ZFS-Speicher-Pool

Wenn Pool-Metadaten beschädigt wurden und deswegen auf den Pool nicht mehr zugegriffen werden kann, müssen Sie den Pool und alle seine Daten aus einer Sicherungskopie wiederherstellen. Das eingesetzte Verfahren hängt weitgehend von der Pool-Konfiguration und der Strategie beim Erstellen von Sicherungskopien ab. Zuerst sollten Sie die von `zpool status` angezeigte Konfiguration speichern, sodass Sie diese nach dem Löschen des Pools wiederherstellen können. Löschen Sie den Pool dann mit dem Befehl `zpool destroy -f`. Legen Sie sich an sicherer Stelle eine Datei an, in der die Struktur der Datasets sowie die einzelnen lokal gesetzten Eigenschaften beschrieben sind, da diese Informationen nicht mehr zugänglich sind, wenn auf den Pool nicht mehr zugegriffen werden kann. Mithilfe der Pool-Konfiguration und der Dataset-Struktur können Sie nach dem Löschen des Pools die vollständige Konfiguration wiederherstellen. Danach können Sie mithilfe eines beliebigen Wiederherstellungsverfahrens den Pool wieder mit Daten „auffüllen“.

Reparieren eines Systems, das nicht hochgefahren werden kann

ZFS soll trotz möglicher Fehler robust und stabil sein. Dennoch können Softwarefehler oder bestimmte unerwartete Fehlfunktionen beim Zugriff auf einen Pool zum Systemabsturz führen. Jeder Pool muss im Rahmen des Systemstarts geöffnet werden, was bedeutet, dass Fehler beim Zugreifen auf einen Pool in eine Systemabsturz-Neustart-Schleife münden können. Als Ausweg aus einer solchen Situation muss ZFS mitgeteilt werden, Pools beim Systemstart zu ignorieren.

ZFS unterhält in `/etc/zfs/zpool.cache` einen internen Cache aktiver Pools und ihrer Konfiguration. Der Speicherort und der Inhalt dieser Datei sind nicht öffentlich zugänglich und Änderungen unterworfen. Wenn ein System nicht mehr hochgefahren werden kann, sollten Sie es mit der Meilenstein-Option `none` (`-m milestone=none`) booten. Nach dem Hochfahren des Systems sollten Sie das Root-Dateisystem ohne Schreibschutz einhängen und dann `/etc/zfs/zpool.cache` umbenennen oder verschieben. Dadurch „vergisst" ZFS, dass im System Pools vorhanden sind, sodass es auf den den Fehler verursachenden schadhaften Pool nicht mehr zugreifen kann. Danach können Sie durch Absetzen des Befehls `svcadm milestone all` den normalen Systemzustand wiederherstellen. Beim Booten von einem alternativen Root-Verzeichnis zu Reparaturzwecken können Sie ein ähnliches Verfahren anwenden.

Wenn das System läuft, können Sie versuchen, den Pool mithilfe des Befehls `zpool import` zu importieren. Dadurch tritt jedoch wahrscheinlich der gleiche Fehler wie beim Systemstart auf, da dieser Befehl zum Zugriff auf Pools das gleiche Verfahren verwendet. Wenn das System mehrere Pools besitzt, gehen Sie wie folgt vor:

- Benennen Sie die Datei `zpool.cache` um oder verschieben Sie diese in ein anderes Verzeichnis (siehe oben).
- Finden Sie heraus, bei welchem Pool Probleme auftreten, indem Sie sich mithilfe des Befehls `fmdump -eV` die Pools, für die kritische Fehler gemeldet wurden, anzeigen lassen.
- Importieren Sie die Pools nacheinander. Lassen Sie dabei die Pools aus, bei denen gemäß der Ausgabe des Befehls `fmdump` Probleme aufgetreten sind.

Index

B

C

D

E

L

M

N

T

U

V

W

X

Z