

StoreBackup 3.5.2

<https://savannah.nongnu.org/projects/storebackup>

19. April 2022

1 Schnellstart

StoreBackup ist ein sehr speichereffizientes Platte-zu-Platte Backup Suite für GNU/Linux und andere unixoide Systeme. Weitere Details und Hilfen finden sich in späteren Kapiteln dieses Dokuments.

1. Lade die Quellen von <https://download.savannah.nongnu.org/releases/storebackup/> herunter
2. Entpacke sie (mit `tar -jxvf`) nach `/opt` (dies erzeugt den Ordner `/opt/storeBackup`).¹
3. Erzeuge symbolische Links: Gib in einem Terminal folgende zwei Kommandos ein (die zweite Linie endet mit: Leerstelle, Punkt):

```
1 # cd /usr/local/bin
2 # ln -s /opt/storeBackup/bin/* .
```

4. Starte Dein erstes Backup folgendermaßen (ersetze „Benutzer“ durch Deinen Benutzernamen):²

```
1 storeBackup.pl --sourceDir /home/Benutzer --backupDir /tmp/my_backup_destination
```

Das kann jetzt eine Zeitlang dauern. Öffne eine zweite Shell und sieh nach, was im Backup-Verzeichnis passiert. Am Schluss ist Dein Home-Verzeichnis nach `/tmp/my_backup_destination` gesichert worden.

Um weitere Details zu erfahren, musst Du weiterlesen; insbesondere Installation, Kapitel 2 und `storeBackup.pl`, Kapitel 6.2. Falls die Anweisungen oben für Dich zu schwierig waren: Im Folgenden wird alles von der Installation von storeBackup bis hin zur NFS Server Konfigurierung im Detail erläutert.

Siehe auch storeBackups Top Features auf der nächsten Seite

¹Du benötigst root-Rechte, um storeBackup nach `/opt/storeBackup` zu entpacken – und auch für die nächsten Schritte. StoreBackup kann jedoch auch an eine Stelle entpackt werden, wo Root-Rechte nicht notwendig sind. Wenn es ohne Root-Rechte gestartet wird, läuft es jedoch nur mit den dann aktuellen Rechten.

²Wenn Du storeBackup über den Paketmanager von Debian oder Ubuntu installiert hast, fehlt bei den Programmen das „.pl“. Anstelle von `storeBackup.pl` ist also `storeBackup` zu starten.

1.1 storeBackups Top Features

- Einfaches Zurücksichern – auch ohne storeBackup! Der wichtigste Aspekt eines Backup Programms ist einfache Rücksicherung aus einem transparenten (natürlichen) Speicherformat.
- Kopiert / komprimiert Dateien auf eine andere Platte und generiert Backups mit Zeitstempel bei gleichzeitiger Identifikation von Doubletten (z. B. umbenannte, kopierte Dateien / Verzeichnisse), die per Hardlink verbunden werden (jedes Backup ist vollständig).
- Erkennen von identischen Dateien (Doubletten) in unterschiedlichen, unabhängigen Backups (z. B. von unterschiedlichen Rechnern)
- Trennt große Image-Dateien (z. B. von TrueCrypt, mbox, XEN, KVM, VMware usw.) oder ganze Devices in kleine Teile und speichert Deltas zu existierenden Backups, was Platz und Zeit spart.
- Unterstützt Sparse Dateien.
- Ausgeklügelte Möglichkeiten zum Einbeziehen oder Ausschließen von Dateien und Ordnern
- Durchführung isolierter, inkrementeller Backups (z. B. bei Reise mit Laptop) und spätere Integration in die zentrale Sicherung
- Zeitversetzte Replikation von Backups auf zusätzliche Platten / Verzeichnisse; auch für komplexe Replikationsschemata
- Unterstützt die Überprüfung der Konsistenz der Backups mittels Prüfsummen (md5) um z. B. Festplattenfehler festzustellen.
- Ermöglicht die Überprüfung von (alten) Dateien im Quellverzeichnis gegen die Prüfsummen im Backup um z. B. Festplattenfehler festzustellen.
- storeBackup ist schnell – auch über langsame Leitungen mit hoher Latenz.
- storeBackup ist eine vollständige Suite von Backup-bezogenen Tools

1.2 Warum sollte man überhaupt ein Backup machen?

Die Antwort ist einfach – aus zwei Gründen:

1. Um den letzten (gesicherten) Zustand z. B. nach einem Hardware- oder Software Fehler wiederherzustellen.
2. Um alte Versionen einer Datei oder eines Ordners zurückzuholen, weil die Daten gelöscht oder zerstört wurden, ohne dass es bemerkt wurde (z. B. durch einen Software-Fehler). Oder man bemerkt erst später, dass die Datei(en) aus Versehen gelöscht wurde(n).

Das wichtigste Backup ist das, das Du nicht gemacht hast!

Neue Versionen werden unter <http://freshmeat.net/projects/storebackup> angekündigt. Um rechtzeitig informiert zu werden, sollte man den entsprechenden RSS-Feed abonnieren.

Bei Hinweisen, Kommentaren oder Fragen sende eine E-Mail an hjclaes@web.de

StoreBackup ist lizenziert unter den Bedingungen der GPL-v3 oder einer späteren Version.

Heinz-Josef Claes mit Hilfe von Mitwirkenden, 19. April 2022

Inhaltsverzeichnis

1	Schnellstart	1
1.1	storeBackups Top Features	2
1.2	Warum sollte man überhaupt ein Backup machen?	2
2	Installation	4
3	Erste Schritte	5
4	Was gibt's Neues?	6
4.1	Was gibt es Neues in storeBackup Version 3.5.2	6
4.2	Was gibt es Neues in storeBackup Version 3.5.1	7
4.3	Was gibt es Neues in storeBackup Version 3.5	7
4.4	Was gibt es Neues in storeBackup Version 3.4.3	8
4.5	Was gibt es Neues in storeBackup Version 3.4.2	8
4.6	Was gibt es Neues in storeBackup Version 3.4.1	8
4.7	Was gibt es Neues in storeBackup Version 3.4	9
4.8	Was gibt es Neues in storeBackup Version 3.3.1	9
4.9	Was gibt es Neues in storeBackup Version 3.3	9
5	Die Idee hinter storeBackup	10
5.1	Kurzbeschreibung	10
5.2	Noch ein Backup-Tool? – Die Wurzeln von storeBackup	11
5.3	Was wäre das ideale Backup-Tool?	11
5.4	Wie storeBackup arbeitet	12
5.4.1	Illustration	12
5.4.2	Reduzierung des Plattenplatzes	13
5.4.3	Performance	14
5.4.4	Beispiellauf	16
6	Komponenten / Programme	19
6.1	Unterstützte Plattformen und Tipps	19
6.2	storeBackup.pl	21
6.2.1	storeBackup.pl Optionen	23
6.3	storeBackupUpdateBackup.pl	34
6.4	storeBackupRecover.pl	37
6.5	storeBackupVersions.pl	38
6.6	storeBackupSearch.pl	39
6.7	storeBackupSetupIsolatedMode.pl	40
6.8	storeBackupMergeIsolatedBackup.pl	41
6.9	storeBackuppls.pl	41
6.10	storeBackupDel.pl	44
6.11	storeBackupMount.pl	45
6.12	storeBackupCheckBackup.pl	47
6.13	storeBackupCheckSource.pl	49
6.14	storeBackup_du.pl	50
6.15	storeBackupConvertBackup.pl	50
6.16	linkToDirs.pl	50
6.17	llt	52
6.18	multiTail.pl	52
7	Grundlegende Konzepte	54
7.1	Konfigurationsdatei und Kommandozeile	54
7.2	Auswahl von Verzeichnissen / Dateien für die Sicherung	56
7.3	Löschen von alten Backups	56
7.4	Definition von Regeln	58
7.4.1	Festlegen, ob eine Datei komprimiert werden soll	62

7.5	Sicherung von Image Files / raw Devices / Blocked Files	63
7.6	Verwendung der Option lateLinks	67
7.7	Isolated Mode / Offline Backups	69
7.7.1	So werden Isolated Backups verwendet:	70
7.7.2	Einrichten von Isolated Mode	71
7.7.3	Isolated Mode als Schutz vor Ransomware	74
7.8	Replikation von Backups	75
7.8.1	Einstieg mit storeBackups Replication Wizard	75
7.8.2	Warum das Kopieren von Backups kein Ersatz für die Replikation ist	77
7.8.3	Grundlegende Konzepte vor Verwendung der Replikation von storeBackup	79
7.8.4	Den Replication Wizard über ein Beispiel verstehen	82
7.8.5	Eine einfache Replikation ohne den Replication Wizard	83
7.8.6	Wie storeBackups Replikation funktioniert	86
7.8.7	Verwendung von Wildcards für die Replikation	88
7.9	Spezielle Dateien von storeBackup	90
7.10	Wie robust ist storeBackup?	91
7.10.1	Robustheit ohne lateLinks	91
7.10.2	Robustheit mit lateLinks, einfaches Beispiel	94
7.11	Konfiguration von NFS	103
7.12	Was ist ein Inode	104
7.13	Bedeutung der Statistikausgaben von storeBackup.pl	105
7.14	Monitoring	107
7.15	Limitierungen	107
8	Interna	108
8.1	Vorbemerkung	108
8.2	Vollständige Backups	108
8.3	Late Links Backups	110
8.4	Replikation	115
9	Verwendung von storeBackup (Beispiele)	126
9.1	Einige Informationen zu Beginn	126
9.2	Beispiel 1, sehr einfaches Backup	128
9.3	Beispiel 2, Backup von mehreren Verzeichnissen	128
9.4	Beispiel 3, ein kleines Backup täglich, ein großes einmal die Woche	128
9.5	Beispiel 4, Backup von unterschiedlichen Rechnern, Daten geteilt	129
9.6	Beispiel 5, unterschiedliche Aufbewahrungsfristen für verschiedene Verzeichnisse	130
9.7	Beispiel 6, Verwendung von lateLinks	130
10	FAQ, Frequently asked Questions	131
11	Mitwirkende	136
12	Change Log	136
13	License	156

2 Installation

Du solltest Die Idee hinter storeBackup (Kurzbeschreibung), siehe Kapitel 5.1, sowie Unterstützte Plattformen und Tipps, siehe Kapitel 6.1, lesen um zu sehen, ob storeBackup zu Deinen Anforderungen passt.

Die Installation ist simpel.

Lade das Archiv von <https://download.savannah.nongnu.org/releases/storebackup/> und gehe zu dem Verzeichnis, in das es entpackt werden soll.

Wenn Du nicht weißt, wohin Du es entpacken sollst, empfehle ich /opt. (Hierzu sind root-Berechtigungen

notwendig.) Wenn Du `/opt` gewählt hast, dann entspricht im folgenden Beispiel `path` dem Verzeichnis `/opt`.

```
$ cd path
$ tar jxvf pathToArchive/storeBackup-3.3.tar.bz2
```

Das Auspacken erzeugt das Verzeichnis `storeBackup` mit den Unterverzeichnissen `bin`, `lib`, `man`, und `doc`. Um zu vermeiden, bei jedem Start von `storeBackup.pl` (oder anderer Programme dort) den gesamten Pfad angeben zu müssen, gibt es zwei Wege:

Die erste Möglichkeit ist, die `$PATH` Variable zu setzen:

```
1 $ cd storeBackup/bin
2 $ export PATH=`pwd`: $PATH
```

(Die Hochkommata um `pwd` müssen „back quotes“, ASCII Code 96 sein; einige PDF Leser zeigen sie als normale Hochkommata an!)

`PATH` sollte in Deiner `.bashrc` (oder für die jeweils benutzte Shell) gesetzt werden.

Die zweite Möglichkeit ist, symbolische Links in einem Verzeichnis zu erzeugen, das bereits in `PATH` enthalten ist. Wenn z. B. `PATH` das Verzeichnis `/usr/local/bin` enthält (und Du Schreibberechtigung hast), kannst Du eingeben:

```
# cd /usr/local/bin
# ln -s path/storeBackup/bin/* .
```

Benutze hierfür keine harten Links, da `storeBackup` dann seine Bibliotheken nicht finden würde.

Wenn Du Zugriff auf die `man`-Pages über das „`man`“ Kommando haben willst, solltest Du `MANPATH` setzen:

```
1 $ cd storeBackup/man
2 $ export MANPATH=`pwd`: $MANPATH
```

Je nach Installationsort im Filesystem muss der Pfad nach dem `cd`-Kommando natürlich angepasst werden. Auch sollte `MANPATH` je nach Shell z.B. in der `.bashrc` gesetzt werden.

Bitte sieh auch in die Datei `README.1ST`, die sich im `doc` Verzeichnis befindet.

3 Erste Schritte

Nun zum ersten Backup.

Wir gehen davon aus, dass Du Dein Home-Verzeichnis nach `/tmp/my_backup_destination` sichern willst. (Wenn Dein Home-Verzeichnis dafür zuviel Platz belegt, nimm einfach einen kleineren Ordner in Deinem Home-Verzeichnis.)

Geh in Dein Home-Verzeichnis und gib ein:

```
1 $ mkdir /tmp/my_backup_destination
2 $ cd
3 $ storeBackup.pl --sourceDir . --backupDir /tmp/my_backup_destination
```

Alternativ kannst Du `storeBackup.pl` mit einer zusätzlichen Option starten:

```
1 $ storeBackup.pl --sourceDir . --backupDir /tmp/my_backup_destination --checkCompr
```

Diese Option `--checkCompr` bewirkt, dass `storeBackup.pl` bei allen Dateien, die größer als ein Kilobyte sind, den Inhalt dahingehend untersucht, ob sich Komprimieren lohnt. Ohne diese Option werden dazu statische Dateityp-Listen verwendet, die deutlich weniger präzise sind. (Im Kapitel 7.4.1, „Festlegen, ob eine Datei komprimiert werden soll“ erfährst Du später noch genauere Details.)

Falls `storeBackup.pl` nicht in Deinem Pfad ist, bekommst Du eine Fehlermeldung von der Shell und musst entweder `$PATH` setzen oder den vollen Pfad zu `storeBackup.pl` eingeben.³

Abhängig davon, wie viele Daten in Deinem Home-Verzeichnis sind, dauert es jetzt einige Zeit, weil `storeBackup.pl` Deine Dateien komprimiert. Hierfür werden alle Cores Deines Rechners verwendet. Aufgrund dieser Komprimierung ist das erste Backup sehr langsam.

³siehe Installation, Kapitel 2 falls Du nicht weißt, was das bedeuten soll.

Eine Verringerung der Belastung des Rechners durch storeBackup kannst Du (insbesondere beim ersten Lauf) durch Verwendung der Option `--compress 'nice bzip2'` und eventuell durch Aufrufen von `storeBackup.pl` mit `ionice -t -c 3 storeBackup.pl` erreichen.

Wenn das Backup fertig ist, erzeuge im gesicherten Dateibaum eine neue Datei, kopiere eine Datei, benenne eine Datei um und starte `storeBackup.pl` nochmals:

```
1 $ cd
2 $ storeBackup.pl --sourceDir . --backupDir /tmp/my_backup_destination
oder
1 $ storeBackup.pl --sourceDir . --backupDir /tmp/my_backup_destination --checkCompr
```

Diesmal wird es sehr viel schneller gehen.

Gehe zum Verzeichnis `/tmp/my_backup_destination`. Du findest dort ein Verzeichnis mit dem Namen `default`. Dieses heißt bei storeBackup eine „Serie“ (engl. „series“), weil es eine Serie von Backups für Deinen Rechner beinhaltet. Du kannst den Namen der Serie von „default“ einfach in den Namen Deines Computers ändern. Das erfolgt sehr einfach in der storeBackup Konfigurationsdatei, die später erklärt wird.

In dem `default`-Ordner findest Du zwei Unterverzeichnisse, deren Namen das Datum und die Uhrzeit der beiden Backups angeben. Gehe in diese Verzeichnisse (nimm dafür zwei Shells, für jedes Verzeichnis eine) und sieh sie Dir mit folgendem Kommando an:

```
$ ls -li
```

Option „i“ bewirkt das Anzeigen der Inode Nummer, die Du in der Spalte ganz links sehen kannst. Überprüfe, dass Dateien mit demselben Inhalt (insbesondere die, die Du kopiert, umbenannt und verschoben hast sowie die im umbenannten Ordner) auf denselben Inode verweisen – somit existieren diese Dateien (platzsparend) nur einmal im Dateisystem.

Hinweis: Du kannst die Optionen auch in der Kurzform verwenden:

```
1 $ storeBackup.pl --sourceDir . --backupDir /tmp/my_backup_destination --checkCompr
ist identisch mit:
1 $ storeBackup.pl -s . -b /tmp/my_backup_destination -C
```

Wenn Du storeBackup in einer Version vor 2.0 verwendet hattest und Backups folgendermaßen gemacht hast:

```
1 storeBackup.pl -s sourceDir -t targetDir    # !!! alte Syntax !!!
```

und jetzt mit Version 2.0 oder später fortfahren willst, verwende:

```
1 storeBackup.pl -s sourceDir --backupDir targetDir -S .
```

Die Parameter für `sourceDir` und `targetDir` sind dabei für beide Versionen gleich.

4 Was gibt's Neues?

Eine Liste aller Änderungen findet sich im Kapitel 12, „ChangeLog“. Dieses Kapitel listet ausschließlich eine Zusammenfassung neuer Features und Fehlerkorrekturen auf.

4.1 Was gibt es Neues in storeBackup Version 3.5.2

Diese Version korrigiert einige Fehler und passt sich an der Änderung des Verhaltens eines Caches (von Linux) an.

Option maxHardLinks Diese Option von `storeBackup.pl` konnte nur als Argument in der Kommandozeile konfiguriert werden. Dieses ist jetzt auch in der Konfigurationsdatei möglich.

Option checkDevices0 Beim Auslesen von Devices (z. B. `/dev/sdf`) mussten in `storeBackup.pl` interne Puffer angepasst werden um Performance Probleme zu vermeiden.

Option checkDevicesDir0 Hier wurde von `storeBackup.pl` bei mehrstufigen Angaben (z. B. `Dev/Stick`) nicht gesichert. Die Verwendung mehrstufiger (neuer oder im Backup bestehender) Directories wird jetzt unterstützt.

4.2 Was gibt es Neues in storeBackup Version 3.5.1

Diese Version korrigiert einige Fehler (siehe das ChangeLog für Details). Folgende Erweiterungen sind (neben kleineren) zusätzlich implementiert:

Option suppressInfo Diese Option von `storeBackup.pl` unterdrückt unerwünschte `INFO` Meldungen im Log.

Option maxHardLinks Begrenzt bei den Programmen `storeBackup.pl`, `storeBackupUpdateBackup.pl` und `linkToDirs.pl` die maximal verwendete Anzahl von Hardlinks. (Nähere Erläuterungen siehe gegen Ende des Kapitels 7.12, „Was ist ein inode“)

Option move Wenn Du Dein isoliertes Backup verschieben anstelle von kopieren willst, kannst Du die Option `--move` bei Aufruf von `storeBackupMergeIsolatedBackup.pl` verwenden. Isolated Backups können dazu verwendet werden zu verhindern, dass Ransomware (Malware, die Daten auf Massenspeichern verschlüsselt) die Backups ebenfalls verschlüsseln kann. Näheres dazu findet sich in Kapitel 7.7.3, „Isolated Mode als Schutz vor Ransomware“.

Optionen precommand und postcommand Den über diese Optionen gestarteten Skripten werden die Werte einiger `storeBackup` Optionen zur Verfügung gestellt. (Nähere Erläuterungen siehe in Kapitel 6.2.1, „storeBackup.pl Optionen“.)

Option lockFile Diese Option von `storeBackup.pl`, `storeBackupUpdateBackup.pl` und `storeBackupDel.pl` hat neue Default-Werte: Für den User `root` `/var/lock/storeBackup.lock` und für andere Benutzer `$HOME/storeBackup.lock`.

4.3 Was gibt es Neues in storeBackup Version 3.5

Diese Version korrigiert einige Fehler (siehe das ChangeLog für Details). Folgende Erweiterungen sind (neben kleineren) zusätzlich implementiert:

Berücksichtigen / Ausschließen von Dateien Es gibt verschiedene Möglichkeiten, Dateien in die Sicherung einzubeziehen oder auszuschließen. In Kapitel 7.2 „Auswahl von Verzeichnissen / Dateien für die Sicherung“ gibt es nun eine Übersicht und Erläuterungen, wie sie interagieren.

Option stayInFileSystem Durch Verwendung dieser Option wird erreicht, dass nur Dateisysteme, die durch die Optionen `sourceDir` und den Symlinks von `followLinks` gesichert werden.

Wildcards für das Verlinken von Serien und für Replikation Für die Konfiguration der Option `otherBackupSeries` können jetzt auch Wildcards verwendet werden. Mehr hierzu findet sich am Anfang des Kapitels 7.8.7 „Verwendung von Wildcards für die Replikation“.

Markierung von (nicht) fertigen Backups Vor dieser Version (3.5) wurden (noch) nicht fertige Backups durch die Existenz der Datei `.md5Checksums.notFinished` markiert. Diese Datei wurde nach der Erstellung des Backups und einem Dateisystem-Sync gelöscht.

Während ein Backup erstellt wird, wird es nie von einem anderen laufenden `storeBackup` Programm für externe Links verwendet. Das gilt auch für Backups, die nie beendet wurden.

Dieses Verhalten von `storeBackup.pl` wurde nun in die folgenden Schritte geändert:

1. Erzeuge das neue Backup-Verzeichnis und führe den Backup-Erstellungs-Teil von `storeBackup.pl` selbst durch.
2. Synchronisiere das Dateisystem.
3. Erzeuge die Markierungsdatei `.storeBackupLinks/backup.Finished`.
4. Führe die Löschung von alten Dateien (und, falls die Option `deleteNotFinishedDirs` gesetzt ist, das Löschen von unvollendeten Backups) durch.

Wenn Du auf diese Version umschwenkst und Deine Backups (usw.) laufen lässt, musst Du nichts besonderes beachten. *Aber wenn Du zu einer vorherigen Version zurück willst, nachdem Du Backups mit Version 3.5 (oder danach) erstellt hast, musst Du sicherstellen, dass alle nicht vollendeten Backups, die von Version 3.5 (oder später) erstellt wurden, gelöscht wurden!*

--force Option für das Aufsetzen des „Isolated Modes“ Beim Aufsetzen des Isolated Mode kannst Du nun die Verwendung des letzten existierenden Backups der Serie erzwingen – auch wenn dieses noch nicht mit `storeBackupUpdateBackup.pl` vervollständigt ist.

Log Datei Optionen für `storeBackupMount.pl` Die von `storeBackup.pl` bekannten Log-Optionen sind jetzt auch bei `storeBackupMount.pl` verwendbar. Siehe die Erläuterungen in Kapitel 6.11, `storeBackupMount.pl` für weitere Details.

Bezeichnung für Option in `multiTail.pl` geändert Änderung des Namens der Option `noOldFiles` in `noOfOldFiles` („number of old files“), damit sie bei allen Programmen identisch ist.

Raspberry PI In FAQ 8 gibt es einige Anmerkungen zum Raspberry Pi.

4.4 Was gibt es Neues in storeBackup Version 3.4.3

Diese Version korrigiert einige Fehler (siehe das ChangeLog für Details). Folgende Erweiterungen sind (neben kleineren) zusätzlich implementiert:

multitail Der Name von `multitail.pl` ist jetzt `multiTail.pl`, um Konflikte mit dem Open Source Programm `multitail` zu vermeiden.⁴ Du musst die Aufrufe von `multitail.pl` in `multiTail.pl` oder eventuell `multiTail` ändern.

multiTail.pl unterstützt jetzt farbige Ausgaben über die Optionen `--color` und `--grep`. Die neue Option `--print` zeigt die gesetzten Optionen.

Deduplikation Bei Verwendung von „blocked files“ wurden identische Blöcke manchmal nicht gefunden (korrigiert).

linkToDirs.pl Neue Option `--printDepth` und Zeitraum für `--progressReport` hinzugefügt.

Ubuntu (und eventuell andere Distributionen) setzen `$PWD` bei Start eines Programms mit `sudo` nicht. `storeBackup.pl` hängt jetzt nicht mehr von `$PWD` ab.

4.5 Was gibt es Neues in storeBackup Version 3.4.2

Diese Version korrigiert einige Fehler (siehe das ChangeLog für Details). Folgende Erweiterungen sind (neben kleineren) zusätzlich implementiert:

Sparse Dateien Wenn „blocked Files“ aus dem Backup wiederhergestellt werden, kannst Du mittels der Option `--createSparseFiles` (oder der Kurzform `-s`) eine Sparse Datei erzeugen lassen, falls im Backup Blöcke vollständig mit Nullen gefüllt waren. Siehe auch Kapitel 6.4, „`storeBackupRecover.pl`“. Weiterhin verfügt `linkToDirs.pl` nun über dieselben Optionen zur Unterstützung von Sparse Dateien, siehe Kapitel 6.16, „`linkToDirs.pl`“ für weiterführende Informationen.

Zeitbasierter Fortschrittsbericht Option `--progressReport (-P)` von `storeBackup.pl` erlaubt nun zusätzlich die Angabe eines Zeitraums, nach dem eine Log-Meldung ausgegeben wird.

4.6 Was gibt es Neues in storeBackup Version 3.4.1

Diese Version korrigiert einige Fehler (siehe das ChangeLog für Details). Folgende Erweiterungen sind (neben kleineren) zusätzlich implementiert:

saveRAM Fehler Die Funktion `MARK_DIR` und `MARK_DIR_REC` zum Definieren von Regeln funktionierten nicht mit der Option `saveRAM`.

Dokumentation Das neue Kapitel „Interna“ beschreibt, wie die `storeBackup`-Komponenten zusammenarbeiten. Dieses Wissen macht es einfacher in ungeplanten Situationen wie abgebrochenen Backups oder Replikationen sinnvoll zu reagieren. Sie Kapitel 8, „Interna“.

⁴Debian und Debian basierende Distributionen entfernen das `.pl` von den Namen der `storeBackup` Programme.

4.7 Was gibt es Neues in storeBackup Version 3.4

Diese Version korrigiert einige Fehler (siehe das ChangeLog für Details). Folgende Erweiterungen sind (neben kleineren) zusätzlich implementiert:

Sicherung auf nicht systemkompatiblen Dateisystemen Eine Linux Sicherung auf z. B. eine NTFS Platte oder über sshfs kann keine „special files“ speichern. Hierfür gibt es jetzt in `storeBackup.pl` die Optionen `archiveTypes` und `specialTypeArchiver`, die derartige Dateien einzeln in cpio- oder tar-Archive packen.

Spezialbehandlung von Verzeichnissen Außerhalb des Backuptools gesetzte Datei-Flags können z. B. dazu verwendet werden, bestimmte Verzeichnisse (optional auch die Unterverzeichnisse) vom Backup auszunehmen oder Dateien (nicht) zu komprimieren. Siehe hierzu Beispiel 5 in Kapitel 7.4, „Definition von Regeln“.

Einfaches Wiederaufsetzen von Isolated Mode Eine einmal für Isolated Backups generierte Konfigurationsdatei kann einfach für ein identisches Erzeugen desselben Isolated Mode verwendet werden.

Lesbarkeit Die in jedem Download enthaltene pdf-Version dieses Dokumentes verwendet jetzt zur besseren Lesbarkeit Microtype, siehe auch <http://mirrors.ibiblio.org/CTAN/macros/latex/contrib/microtype/microtype.pdf>.

4.8 Was gibt es Neues in storeBackup Version 3.3.1

Diese Version korrigiert hauptsächlich Fehler (siehe das ChangeLog für Details). Folgende Erweiterungen sind (neben kleineren) zusätzlich implementiert:

Mounten `storeBackupMount.pl` wurde neu geschrieben, daher mußt Du die Optionen anpassen. Du kannst nun auch andere Programme als `storeBackup.pl` aufrufen. Siehe die Erläuterungen in Kapitel 6.11, `storeBackupMount.pl` für weitere Details.

einfache Komprimierungs-Regel `storeBackup.pl` unterstützt jetzt mit der neuen Option `--checkCompr (-c)` eine (weitere) einfach zu nutzende Möglichkeit, die Komprimierung auf der Kommandozeile festzulegen. Bei Verwendung dieser Option wird der Inhalt aller Dateien größer als 1k dahingehend überprüft, ob eine Komprimierung sinnvoll erscheint. Siehe Kapitel `storeBackup.pl`, section 6.2 für Details.

4.9 Was gibt es Neues in storeBackup Version 3.3

Komprimierung: In älteren Versionen konntest Du Dateien, die *nicht* komprimiert werden sollten, durch eine Liste von Datei-Endungen (Suffixes) definieren. Du konntest ebenfalls eine Minimalgröße angeben unterhalb der Dateien ebenfalls nicht komprimiert wurden. Wenn Du Deine Konfiguration nicht änderst, bleibt alles beim Alten. Ab dieser Version kannst Du jedoch zusätzlich eine Liste von Dateien mit Endungen angeben, die *immer zu komprimieren* sind – und zwar ergänzend zur Liste der oben genannten Dateien, die *nicht zu komprimieren* sind, sowie eine Minimalgröße für die Kompressionsentscheidung. Bei Dateien, die nicht in die so beschriebenen Kategorien fallen, macht `storeBackup.pl` eine Abschätzung, ob die Größe der Datei durch Kompression verringert werden müßte. Siehe Kapitel 7.4.1, „Festlegen, ob eine Datei komprimiert werden soll“ für Detail-Informationen.

isolated mode: Wenn Du z. B. mit einem Laptop ohne Verbindung zu Deinem Backup auf einer Reise bist, kannst Du jetzt Delta-Backups (relativ zu Deinem „großen“ Backup) auf einem kleinen Speichermedium (z. B. einem Memory Stick) speichern und dieses Backup später in das zentrale integrieren. Siehe Kapitel 7.7, „Isolated Mode / Offline Backups“ für weiterführende Informationen.

Replikation von Backups: Ermöglicht Dir Replikationen von Deinem Backup zu anderen Standorten und / oder Speichermedien. Dies kannst Du dazu verwenden, kontinuierliche Kopien Deiner Backups zu erstellen. Siehe Kapitel 7.8, „Replikation von Backups“ für detaillierte Informationen.

linkToDirs.pl: Ermöglicht es, Backups mit anderen zu ver-hard-linken bzw. Dateien zu kopieren. Es funktioniert wie `cp -a`, außer dass alle identischen Dateien die in wählbaren Verzeichnissen existieren per hard link verknüpft werden. Das kann auch verwendet werden, um die Replikation von Backups in einigen speziellen Situationen zu unterstützen. Siehe Kapitel 6.16, `linkToDirs.pl` für detailliertere Informationen.

storeBackupCheckSource.pl: Dieses Tools soll Daten im zu sichernden Verzeichnis finden, bei denen im Laufe der Zeit Bits „umgekippt“ sind. Siehe Kapitel 6.13, `storeBackupCheckSource.pl` für detailliertere Informationen.

5 Die Idee hinter storeBackup

5.1 Kurzbeschreibung

StoreBackup ist ein Festplatte-zu-Festplatte Backup-Programm für GNU/Linux. Es läuft auch auf anderen unixoiden Systemen. Man kann mit einem gewöhnlichen Datei-Browser durch die Backups navigieren (lokal oder über NFS, Samba, SSH⁵ oder über fast jedes Netzwerk-Dateisystem). Dies ermöglicht dem Benutzer den einfachen und schnellen Zugriff auf seine gesicherten Dateien. Der Anwender muss die Dateien nur aus dem Backup-Verzeichnis kopieren (und möglicherweise entpacken). Es gibt auch ein Tool für den Administrator, das es ihm ermöglicht, auf einfache Art (Unter-) Verzeichnisse zurückzusichern. Jedes einzelne Backup eines Zeitpunktes kann gelöscht werden, ohne die anderen Backups zu beeinflussen. StoreBackup erkennt Dateien anhand ihres Inhalts und nicht nur an ihrem Namen oder ihrer Position im Dateisystem. Es erkennt, ob Dateien kopiert, umbenannt oder verschoben wurden. Wenn Dateien identisch sind, aber unterschiedliche Namen haben oder in anderen Verzeichnissen sind, verwendet storeBackup einen effizienten Weg (Hardlinks), um die Datei ohne erneute Kopie in das aktuelle Backup zu integrieren. Wenn ein Anwender seine Foto- oder Musiksammlung umorganisiert, werden die meisten Backup-Programme diese Daten nochmals über das Netzwerk transportieren und sie wieder im Backup speichern, was Zeit und Platz verschwendet. StoreBackup dagegen wird einfach Hardlinks zum identischen Inhalt im Backup verwenden und so eine Menge Zeit und Platz sparen.

StoreBackup kann große Image-Dateien (z. B. von virtuellen Maschinen) in kleine Teile zerlegen und benötigt dann nur den Platz für Änderungen in diesen Teilen. Das Wiederherstellen aus diesen Teilen zur vollständigen Image-Datei ist mit sehr einfachen Tools wie `cat` oder eventuell `bzcat` möglich (oder was auch immer für die Komprimierung verwendet wurde). Auf dieselbe Weise können auch Devices oder Partitionen (wie `/dev/sdb1`) zerlegt werden.

StoreBackup bietet sich für die allgemeine Verwendung für Umgebungen an, in denen kein Bandlaufwerk, aber eine weitere Festplatte oder ein weiterer Rechner verfügbar sind. In professionellen Umgebungen empfiehlt es sich durch den sehr schnellen und komfortablen Zugriff auf die Backups, wodurch Bandkosten und sowie administrative Aufwände eingespart werden können.

StoreBackup ist ein Kommandozeilentool. Es kann über cron automatisch gestartet werden. In der Regel ist ein graphisches Tool auf einem Server nicht gewünscht oder notwendig – und was noch wichtiger ist: Wenn Deine Rechner abgestürzt sind, hast Du wahrscheinlich keinen Zugriff auf eine grafische Benutzeroberfläche (mehr).

Speicher auf Festplatten, Memory Sticks oder ähnlichen Geräten bietet sich als Alternative oder Ergänzung zu Sicherungen auf Bändern an. StoreBackup verfügt über eine gute Performance, spart Speicherkapazität und erhöht die administrative Flexibilität:

- Verzeichnisse mit ihrer internen Baumstruktur werden an eine andere Stelle kopiert (z. B. `/home` → `/var/bkup/2003.12.13_02.04.26`). Die Rechte der Dateien bleiben dabei erhalten, was den Anwendern den direkten Zugriff auf das Backup ermöglicht. Der wichtigste Aspekt eines Backup-Tools ist einfaches und sicheres Wiederherstellen der Daten.
- Der Inhalt jeder zu sichernden Datei wird mit den bereits im Backup befindlichen Inhalten verglichen. Das bewirkt, dass Dateien mit demselben Inhalt nur einmal physisch im Backup existieren.
- Identische Dateien werden über Hardlinks verbunden und erscheinen im Backup an denselben Stellen wie im Original.

⁵siehe Details über Backups über SSH in FAQ4

- Dateien können vom Backup ausgeschlossen werden. Dieses kann über den Ausschluss ganzer Teilbäume oder über Regeln, die auf regulären Ausdrücken, Dateigröße, Gruppen, Anwendern oder anderen Kriterien beruhen, erfolgen.
- Dateien im Backup werden komprimiert, sofern sie nicht davon ausgeschlossen werden. Die Komprimierung kann auch ganz ausgeschaltet oder aufgrund einer Analyse von storeBackup dateispezifisch erfolgen.
- Image-Dateien oder Speichersysteme (mass storage devices), in denen sich von Zeit zu Zeit nur Teile ändern, können auf diese Unterschiede untersucht werden. Im Backup wird dann nur der Platz für die geänderten Blöcke benötigt (die zudem komprimiert werden können).
- Unabhängig voneinander erzeugte Backup-Serien (z. B. von unterschiedlichen Rechnern) können über Hardlinks auf dieselben Dateien verweisen. Auch vollständige oder Teil-Sicherungen können so durchgeführt werden – immer unter der Voraussetzung, dass Dateien mit identischem Inhalt im Backup nur einmal physisch existieren.
- Das Resultat einer Sicherung mit storeBackup ist immer ein „Vollständiges Backup“. Dieses kann später mit einfachen oder komplexen Regeln gelöscht werden.
- StoreBackup unterstützt viele weitere Einstellmöglichkeiten. Sie sind in diesem Dokument beschrieben.

5.2 Noch ein Backup-Tool? – Die Wurzeln von storeBackup

Es gibt wohl tausende von Backup-Programmen. Also warum noch eins? Der Grund entstand aus meiner Tätigkeit als Berater. Ich war die ganze Woche über unterwegs und hatte keine Möglichkeit, meine Daten zu Hause zu sichern. Alles, was ich hatte, war ein 250MB ZIP-Laufwerk, welches über eine parallele Schnittstelle mit meinem Laptop verbunden war. Auf dem ZIP-Laufwerk war nicht viel Platz und ich musste mit einer geringen Übertragungsgeschwindigkeit (etwa 200KB/s) und hoher Latenz auskommen. Im Gegensatz dazu wollte ich schnellen und einfachen Zugriff auf meine Daten – ich mochte die üblichen Varianten von voller, differentieller und inkrementeller Sicherung (z. B. mit tar oder dump) nicht: Einerseits ist es mühsam, Versionen einer Datei herauszusuchen und andererseits kann man Versionen des Backups nicht nach Bedarf löschen – dies muss vorher sorgfältig bei der Durchführung der Sicherungen geplant werden.

Es war mein Ziel, schnell Backups während der Arbeit zu machen und meine Dateien schnell und ohne Umstände zurückholen zu können.

Unter diesen Umständen entstand Ende 1999 die erste Version von storeBackup. Sie war allerdings in größeren Umgebungen nicht brauchbar. Die Performance war nicht gut genug, sie skalierte schlecht und konnte nicht mit ekeligen Dateinamen (z. B. mit „\n“) umgehen.

Basierend auf dieser Erfahrung mit der ersten Version schrieb ich dann eine neue, die etwas weniger als ein Jahr später unter der GPL veröffentlicht wurde. In der Zwischenzeit ist die Anzahl der Anwender gewachsen – vom Heimanwender, der seine E-Mail-Ordner von seinem Internetprovider sichert, über kleine und mittelständische Firmen bis hin zu Krankenhäusern und Universitäten mit allgemeinen Sicherungsanforderungen.

5.3 Was wäre das ideale Backup-Tool?

Der wichtigste Aspekt eines Backup-Systems ist, dass man seine Daten nicht nur zurückholen kann, sondern dass dies auch einfach ist.

Der folgende Absatz bezieht sich auf das Sichern von Dateien, nicht auf das von Datenbanken.

Das ideale Backup würde jeden Tag eine vollständige Kopie des gesamten Datenbestandes (inklusive der Zugriffsrechte) auf einem anderen Datenspeicher mit minimalem Aufwand für den Administrator und maximalem Komfort für den Anwender durchführen. Der Rechner und die Festplattensysteme hierzu sollten natürlich in einem entfernten, sicheren Gebäude stehen. Mit Hilfe eines Dateisystembrowsers könnte der Anwender im Backup suchen und seine Daten direkt zurückkopieren. Das Backup wäre so direkt und problemlos verwendbar, ohne dass spezielle Schulungen notwendig wären. Mit Backups umzugehen würde etwas *Normales* werden – da der Weg über die Administration in der Regel nicht notwendig wäre.

Der hier beschriebene Prozess hat einen „kleinen“ Nachteil: Er benötigt sehr viel Plattenspeicher und ist ziemlich langsam, da jedes Mal alle Dateien kopiert werden müssen.

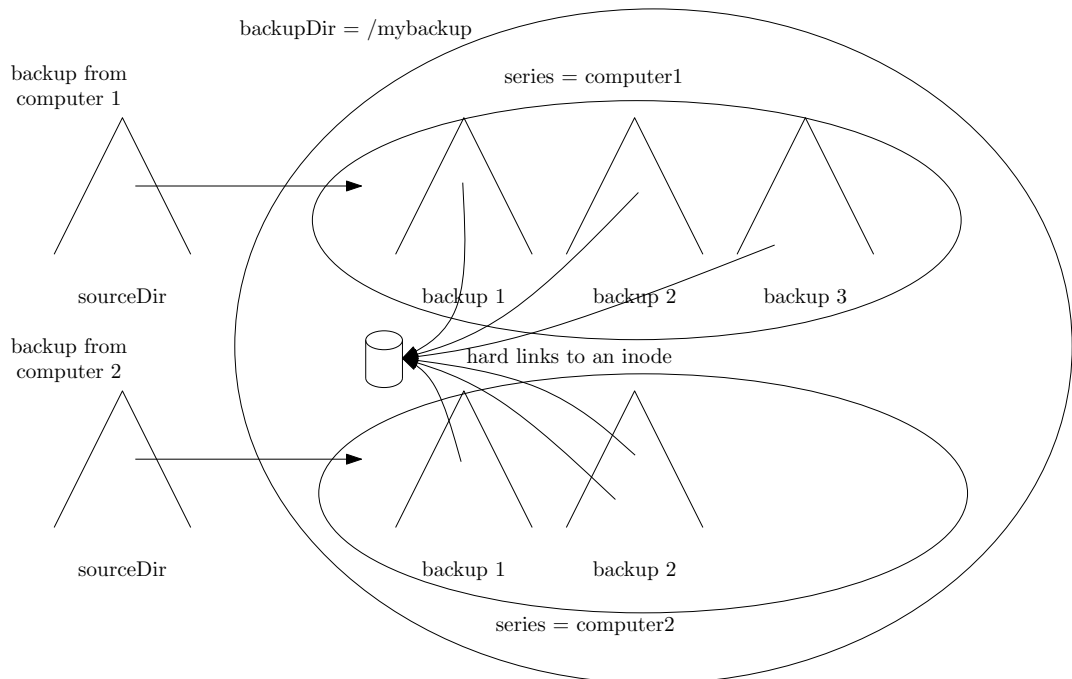
5.4 Wie storeBackup arbeitet

StoreBackup versucht das „ideale Backup“ zu realisieren und dabei zwei Probleme zu lösen: Plattenplatz und Performance.

Du kannst storeBackup auf einem Rechner laufen lassen und die Sicherungen auf ein lokales Laufwerk (oder ein temporär angeschlossenes Laufwerk) oder auf einen Netzwerk Mount (z. B. NFS, sshfs, cifs) schreiben. Dies ist die effizienteste Art der Nutzung. Aber Du kannst es auch so verwenden, dass es die Daten über das Netzwerk von anderen Rechnern holt, z. B. über cifs oder NFS.

5.4.1 Illustration

storeBackup ist ein Disk-to-Disk Backup Programm für GNU/Linux, welches auf Hardlinks basiert. Das folgende Bild verdeutlicht das:



Stell Dir vor, dass Du zwei Rechner über das Netzwerk mittels NFS⁶ auf einen File-Server sichern willst. (Um die grundlegenden Optionen von storeBackup zu verstehen, solltest Du die folgenden Erläuterungen auch lesen, wenn Du das Backup auf eine externe Platte (z. B. usb oder e-sata) schreiben willst.)

Die vorgegebenen Standard- („default-“) Werte aller storeBackup-Optionen sind so gewählt, dass nur zwei Optionen notwendig sind: `--sourceDir` und `--backupDir`. (Die Kurzform dieser Optionen sind `-s` und `-b`.) Alles andere arbeitet dann mit akzeptablen Standardwerten. Die Art, wie Quellverzeichnisse (`sourceDir`) angegeben werden können, ist sehr flexibel⁷.

Zuerst musst Du ein „`backupDir`“ festlegen, in das alle Deine Backups gesichert werden. Nehmen wir an, es ist `/my_backup_destination`. Um unterschiedliche Backups von verschiedenen Rechnern zu unterscheiden, erstellst Du jeweils ein Verzeichnis; in der Abbildung sind es `computer1` und `computer2`.

In der Terminologie von storeBackup sind `computer1` und `computer2` unterschiedliche Serien. Der Ausdruck „Serie“ („`series`“) wird verwendet, weil z. B. `computer1` einen Satz von sequentiellen Backups („eine Serie“), die zu Computer 1 gehört, beinhaltet. Derartige Serien werden in der Abbildung als backup 1, backup 2 und backup 3 dargestellt.

Diese Verzeichnisse kann man in storeBackups Konfigurationsdatei als „`series`“ wiederfinden. Wie dargestellt sind `/my_backup_destination/computer1` und `/my_backup_destination/computer2` die vollständigen Pfade zu diesen Verzeichnissen. In jedem „`series`“-Verzeichnis erzeugt storeBackup eine *Serie von Backups* des `sourceDir` des Computers, der mit der Serie assoziiert ist.

⁶siehe die Beschreibung in Kapitel 7.11 über die Konfiguration von NFS – Du musst Schreibrechte in dem Zielverzeichnis haben. Wenn Du die Sicherung als root durchführst, musst Du in dem gemounteten Verzeichnis ebenfalls root-Rechte haben.

⁷Falls Du mehr als einen Verzeichnisbaum gleichzeitig sichern willst, solltest Du dir später die Option „`followLinks`“ in Kapitel 6.2 ansehen

StoreBackup findet alle Dateien, die schon im Backup-Verzeichnis („my_backup_destination“) abgelegt sind anhand ihres Inhalts, sowohl innerhalb einer Backup-Serie oder auch in mehreren Serien von verschiedenen Computern).

StoreBackup wird alle diese Dateien mittels Hardlinks mit nur einem Inode (siehe Kapitel 7.12) verbinden. Namensänderungen oder das Ändern von Pfaden von Dateien stellen dabei kein Problem dar, da storeBackup anhand des Inhalts feststellt, ob Dateien identisch sind. Ändert sich in der Abbildung oben in backup 3 von computer 1 die Position der Datei (z. B. mit mv) oder wird im backup 2 von computer 2 die Datei kopiert, erkennt storeBackup das und linkt immer auf denselben Inode.

StoreBackup unterstützt mehrere Backup-Serien (z. B. tägliche oder wöchentliche Backups) von unterschiedlichen Quellen (z. B. unterschiedliche Server oder Workstations). Wie erwähnt ist der Standardname für eine Serie `default`. Wenn Du jedoch Backups von mehreren Computern planst, sollte jede Serie mit einem eigenen Namen versehen werden, der den zu sichernden Rechner beschreibt. Dies erfolgt mit der `--series` Option (Kurzform `-S`) auf der Kommandozeile oder einfach durch Setzen von „series“ in der Konfigurationsdatei (siehe auch `storeBackup.pl`, Kapitel 6.2).

Die nächsten Kapitel beschreiben weitere Details, wie storeBackup funktioniert.

5.4.2 Reduzierung des Plattenplatzes

Dateien als Ganzes speichern

Die erste Methode, den benötigten Plattenplatz zu sparen, besteht darin, die Daten zu komprimieren – falls das sinnvoll ist. StoreBackup erlaubt es, jeden Komprimierungsalgorithmus als externes Programm zu verwenden. Standard ist `bzip2`.

Wenn man sich Backups näher ansieht, fällt auf, dass sich von Sicherung zu Sicherung relativ wenige Dateien ändern – was der Grund für inkrementelle Backups ist. Man stellt auch fest, dass viele Dateien denselben Inhalt haben, weil die Nutzer (untereinander) Dateien kopieren oder eine Versionsverwaltung (wie `cvs`) verwendet wird. Zusätzlich werden Dateien oder ganze Verzeichnisbäume von den Anwendern umbenannt, was in inkrementellen Backups zu wiederholten (unnötigen) Sicherungen führt. Die Lösung hierfür ist, im Backup nach Dateien zu suchen, die schon denselben Inhalt haben (möglicherweise sind diese schon komprimiert) und dann auf diese zu verweisen. Mit dem Trick, hierfür Hardlinks auf bereits bestehende Dateien im Backup zu verwenden, ist jede Datei in jedem Backup präsent und steht doch nur einmal auf physisch der Platte. Kopien oder Umbenennungen benötigen nur den Platz eines Hardlinks im Backup, also fast nichts.

Oft muss nicht nur ein Rechner gesichert werden, sondern eine kleinere oder größere Anzahl. Diese haben oft einen nennenswerten Anteil an identischen Dateien, insbesondere in Verzeichnissen wie `/etc`, `/usr` oder `/home`. Offensichtlich sollte nur eine Kopie von identischen Dateien auf dem Backup Laufwerk liegen. Alle Verzeichnisse vom Backup-Server gleichzeitig zu mounten und zu sichern wäre die einfachste Lösung. Auf diese Art würden doppelte Dateien erkannt und mittels Hardlink verknüpft. Dieser Ansatz hat aber den Nachteil, dass alle Rechner während der gesamten Sicherungszeit für das Backup verfügbar sein müssten. Das wird in vielen Fällen nicht möglich sein, wenn z. B. ein Notebook gesichert werden soll. Aber gerade bei Notebooks wird man viele Dubletten finden, weil die Anwender lokale Kopien erzeugen. Aufgrund dieser Fälle, oder wenn Server unabhängig voneinander zu sichern sind, unterstützt storeBackup für die optimale Plattenplatznutzung das Verlinken von unabhängigen Backups. (Das heißt, völlig entkoppelt und möglicherweise von unterschiedlichen Rechnern.)

Dateien aufspalten: blocked files

Die Methodik von Komprimierung und Hardlinks von ganzen Dateien funktioniert sehr gut für „normale“ Office-, Konfigurations-, Programmcode- und alle anderen Typen von kleinen Dateien.

Sie schlägt bei großen Image-Dateien, bei denen sich nur Teile der Dateien ändern, mehr oder weniger fehl. So hat eine Datei mit z. B. 3 GB nur ein paar Megabyte Änderungen, aber die oben beschriebene Methodik würde immer die ganze Datei ins Backup kopieren oder komprimieren, was weder aus Sicht der Sicherungszeit noch der des benötigten Platzes effizient ist. Um dieses Problem zu lösen, kann storeBackup derartige Dateien gesondert verwalten.

Man kann in der Konfigurationsdatei einstellen, welche Dateien als „blocked files“ behandelt werden sollen. Für diese Dateien wird im Backup anstelle einer Datei ein Ordner angelegt (der Name des Ordners ist dabei identisch mit dem Namen der betreffenden Datei). Diese Datei wird nicht als Ganzes im Backup gespeichert; stattdessen wird sie in Form von kleinen, nummerierten Dateien (Blöcken) im dafür erzeugten

Ordner angelegt. Diese Blöcke können auch komprimiert werden.

Im nächsten Backup (nachdem in der Originaldatei etwas verändert wurde) überprüft storeBackup, in welchem dieser Blöcke eine Änderung erfolgte und kopiert / komprimiert *nur diese* Blöcke (erzeugt also nur diese Dateien). Die, die sich nicht geändert haben, werden per Hardlink verknüpft. Dieser auf md5-Summen basierende Vergleich erfolgt auch mit anderen „blocked files“, so dass z. B. bei einer VM, die für unterschiedliche Verwendungszwecke dupliziert wurde, die doppelten Blöcke von storeBackup erkannt werden. Dies gilt auch für mehrfach vorkommende Blöcke innerhalb einer Image-Datei (z. B. wenn unbenutzte Teile in diesen Dateien genullt sind und in größerem Umfang bei sogenannten sparse Dateien⁸). Das Resultat aus diesem Vorgehen ist ein dramatisch reduzierter Platzbedarf, verglichen mit dem Kopieren / Komprimieren der vollständigen Dateien, und es ist immer noch möglich, die Dateien auch ohne ein lauffähiges storeBackup zurückzusichern, was die Philosophie von storeBackup ist. (Zurücksichern ist die wichtigste Funktion eines Backups.) Dies könnte in z. B. 10 Jahren sehr nützlich sein – wer weiß, was dann ist (gibt’s storeBackup noch, unterschiedliche Perl Versionen, ...)

Löschen von Backups

StoreBackup bietet für das Löschen von Sicherungen vielfältige Optionen. Der große Vorteil ist hier, dass jede Sicherung ein vollständiges Backup ist und daher aufgrund nichtexistenter Abhängigkeiten der Backups untereinander beliebig gelöscht werden kann. Im Gegensatz zu „traditionellen“ Backups gibt es keine Notwendigkeit zu berücksichtigen, ob ein inkrementelles Backup vom vorherigen „Vollbackup“ abhängt.

Die Optionen ermöglichen das Löschen oder Erhalten von Backups an bestimmten Wochentagen oder des ersten oder letzten existierenden Backups einer Woche, eines Monats oder eines Jahres. Auch kann eine Minimalanzahl von Backups angegeben werden, die erhalten werden soll. Dies ist insbesondere dann sinnvoll, wenn die Backups nicht regelmäßig erstellt werden. So können die letzten Backups eines Laptops nach einem 4-wöchigen Urlaub selbst dann erhalten werden, wenn die Aufbewahrungsfrist nur drei Wochen ist. Sich widersprechende Regeln werden dabei von storeBackup „mit gesundem Menschenverstand“ aufgelöst.

5.4.3 Performance

Die oben beschriebene Methode Plattenplatz zu sparen geht davon aus, dass vor der Sicherung einer Datei im existierenden Backup nach identischen Dateien gesucht wird. Diese Suche erfolgt im vorhergehenden Backup sowie im neu erzeugten Backup selbst. Natürlich wäre es nicht sinnvoll, jede zu sichernde Datei direkt mit jeder im Backup zu vergleichen. Daher werden die md5-Summen der Dateien der vorherigen Sicherungen über eine Hash Tabelle mit der md5 Summe der zu sichernden Datei verglichen.

Das Berechnen einer md5-Summe geht schnell, aber bei großen Datenmengen ist das immer noch nicht schnell genug. Aus diesem Grund überprüft storeBackup initial, ob eine Datei seit dem letzten Backup unverändert geblieben ist (Pfad + Dateiname, ctime, mtime müssen gleich sein). Wenn das der Fall ist, wird die md5-Summe von der letzten Sicherung übernommen und der Hardlink gesetzt. Wenn dieser initiale Check einen Unterschied liefert, wird die md5-Summe der Datei berechnet und im Backup nachgesehen, ob schon eine Datei mit derselben Prüfsumme existiert. (Der Vergleich mit mehreren Backup-Serien verwendet einen erweiterten, aber ähnlich performanten Algorithmus.) Bei diesem Ansatz müssen nur wenige md5-Summen für ein (wiederholtes) Backup berechnet werden. Wenn Du storeBackup tunen willst, insbesondere beim Schreiben über NFS, sind zwei Möglichkeiten zu erwägen:

- Optimierung von NFS (siehe Kapitel 7.11)
- Verwendung der Option lateLinks von storeBackup und möglicherweise die Trennung des Löschens von alten Sicherungen vom Backup-Prozess.

Die Verwendung von storeBackup mit lateLinks ist vergleichbar mit einer asynchronen Client / Server Applikation – oder genauer formuliert wie die Verwendung mehrerer Batch-Läufe auf mehreren Rechnern:

- Überprüfung der zu sichernden Verzeichnisse, um festzustellen, was sich geändert hat und Sichern (oder Komprimieren) der betroffenen Daten auf den Backup-Server. Die dann noch

⁸Wikipedia/sparse-Datei: Eine Sparse-Datei (engl. sparse file; sparse für „dünnbesetzt“, „spärlich“ oder „zerstreut“) bezeichnet eine Datei, die in einem Dateisystem sehr kompakt gespeichert werden kann, wenn sie größtenteils aus Nullbytes besteht. In einer Sparse-Datei wechseln sich Bereiche, in denen sich explizit aufgezeichnete Daten befinden, mit Bereichen ab, die als Löcher bezeichnet werden und keinen Platz auf dem Speichermedium belegen.

fehlenden Ordner und Hardlinks werden in einer Datei protokolliert.

- Nimm diese Protokolldatei und erzeuge ein „normales“, vollständiges Backup.
- Lösche alte Backups gemäß der Löschregeln.

Die folgenden Performance Messungen zeigen nur die direkte Sicherungszeit (ohne den Aufruf von `storeBackupUpdateBackup.pl` (wenn nötig)⁹). Die Läufe wurden mit Beta Versionen von `storeBackup` 2.0 durchgeführt.

Zunächst einige Hintergrundinformationen zu den folgenden Werten: Das Backup lief auf einem Athlon X2, 2.3GHz, 4GB RAM. Das Netzwerk hatte 100 MBit/s, `storeBackup` wurde mit Standard-Parametern verwendet. Die Einheiten der Messwerte sind Stunden:Minuten:Sekunden oder Minuten:Sekunden. Die Größe von `sourceDir` war 12GB, die daraus resultierende Größe des Backups mit `storeBackup` waren 9.2 GB. Die Sicherungen wurden mit 4769 Verzeichnissen und 38499 Dateien durchgeführt. `StoreBackup.pl` linkte 5038 Dateien intern; diese waren demnach doppelt vorhanden. Die Datenquelle waren meine eigenen Dateien und der „Desktop“ von meinem Windows XP Laptop, also „reale“ Daten.

Die erste Tabelle zeigt die Zeit für das Kopieren der Daten mit Standardprogrammen auf den NFS Server. Der NFS Server war mit der Option `async`¹⁰ gemountet. Dieses ist nicht die Standardkonfiguration, sondern eine Performance-Optimierung.

Kommando	Dauer	Backup-Größe
<code>cp -a</code>	28:46	12 GB
<code>tar jcf</code>	01:58:20	9.4 GB
<code>tar cf</code>	21:06	12 GB

Die Ergebnisse sind wie erwartet: `tar` mit Kompression ist viel langsamer als die anderen, und `cp` ist langsamer als `tar`, da es viele Dateien anlegen muss. Eine Zahl überrascht: Die Größe des Backups mit `tar` ist 9.4 GB, die des Backups mit `storeBackup.pl` dagegen nur 9.2 GB. Der Grund dafür liegt in den 5038 intern verlinkten Dateien – die Duplikate werden bei `storeBackup` nur einmal gespeichert.

Was wir hier nicht sehen, ist der Effekt des Inhaltsvergleichs. Er bedeutet aber einen großen Unterschied bezüglich der Performance und insbesondere im benötigten Plattenplatz. Wenn sich der Zeitstempel einer Datei ändert, werden traditionelle Backupssysteme diese Datei(en) auch in einem inkrementellen Backup nochmals speichern – `storeBackup` wird nur einen Hardlink erstellen.

Nun lassen wir `storeBackup.pl` auf denselben Daten laufen. Der NFS Server ist nach wie vor mit `async` gemountet. Es gab keine Änderungen in `sourceDir` zwischen dem zweiten und dritten Backup.

storeBackup	1.19, Standard		2.0, Standard		2.0, lateLinks		mount with <code>async</code>
1. Backup	49:51	100%	49:20	99%	31:14	63%	Dateisystem Cache geleert Dateisystem Cache gefüllt
2. Backup	02:45	100%	02:25	88%	00:42	25%	
3. Backup	01:51	100%	01:54	100%	00:26	23%	

Wir sehen Folgendes:

- Der erste Lauf von `storeBackup.pl` ist kürzer als `tar jcf` (`tar` mit Kompression). Es ist einfach zu verstehen, warum: `storeBackup.pl` verwendet beide Cores des Rechners, wohingegen die Kompression von `tar` nur einen Core verwendet. Wenn man sich die Zahlen aber näher betrachtet, stellt man fest, dass `storeBackup.pl` weniger als halb so lange benötigt (42%). Zusätzlich berechnet `storeBackup.pl` noch sämtliche md5-Summen und muss tausende von Dateien anlegen (siehe auch den Unterschied zwischen `tar` und `cp`). Der Zeitunterschied von über 50% für das Erstellen der Sicherung ergibt sich aus zwei Effekten: `storeBackup.pl` komprimiert nicht alle Dateien (abhängig von der Endung, z. B. werden `.bz2` Dateien nicht nochmals komprimiert) und es erkennt Dateien mit demselben Inhalt, für die nur ein Hardlink erzeugt wird (der Grund für die 9.2 anstatt der 9.4 GB).
- Das zweite Backup wurde mit neuem Mounten des Quellverzeichnisses erstellt, so dass der Lesecache nicht gefüllt war (und das Ergebnis verfälschen würde). Man sieht einige Verbesserungen zwischen der 1.19er und 2.0er Version, da bei der internen Parallelisierung von `storeBackup.pl` Optimierungen erfolgten.

Beim dritten Lauf sieht man keine Verbesserungen zwischen 1.19 und 2.0, weil das Lesen jetzt aus

⁹Dies ist nur notwendig, wenn `storeBackup.pl` mit der Option `lateLinks` aufgerufen wird. Die für den Lauf von `storeBackupUpdateBackup.pl` benötigte Zeit ist im nächsten Kapitel 5.4.4 aufgeführt.

¹⁰siehe NFS Konfiguration, Kapitel 7.11

dem Datei System Cache erfolgt, so dass der limitierende Faktor die Geschwindigkeit des NFS Servers ist – und die ist in beiden Fällen gleich.

- Mit der Option `lateLinks` kann man eine Verbesserung um den Faktor 4 sehen. Die hier benötigte Zeit hängt massiv von der Zeit für das Lesen des `sourceDir` (und der Zeit für das Lesen der Meta-Informationen aus dem vorherigen Backup) ab.

Nun das Ganze ohne „Tricks“ wie `async` Mounts über NFS:

Kommando	Dauer	Backup-Größe
<code>cp -a</code>	37:51	12 GB
<code>tar jcf</code>	02:02:01	9.4 GB
<code>tar cf</code>	25:05	12 GB

storeBackup	1.19, Standard		2.0, Standard		2.0, lateLinks		mount with sync
1. Backup	53:35	100%	49:20	100%	38:53	63%	Dateisystem Cache geleert Dateisystem Cache gefüllt
2. Backup	05:36	100%	05:24	96%	00:43	13%	
3. Backup	05:10	100%	04:54	95%	00:27	9%	

Wir sehen jetzt Folgendes:

- Alles ist jetzt wegen der – durch die synchrone Kommunikation bedingte – höheren Latenz mehr oder weniger langsamer. Wenn nur eine Datei geschrieben wird (wie bei `tar`, ist der Unterschied zu `async` geringer; wenn viele Dateien geschrieben werden, ist er größer.
- Wir sehen auch, dass bei Verwendung von `lateLinks` der Unterschied zwischen `sync` und `async` sehr gering ist. Der Grund dafür ist einfach zu verstehen: Es werden nur wenige Dateien über NFS geschrieben, so dass die Latenz nur eine geringe Bedeutung für die Gesamtdauer des Backups hat. Hier ergibt sich dann, dass Backups mit `lateLinks` und einem sehr schnellen Quellverzeichnis (Cache) jetzt *10 Mal* so schnell geworden sind.
- Weil die Latenz mit `lateLinks` nicht so wichtig für die Durchführung eines Backups ist, habe ich diesen Dateiserver über ein VPN¹¹ über das Internet gemountet. Dies bewirkt eine sehr hohe Latenz und eine Bandbreite von etwa 20KByte/s vom NFS Server und 50KByte/s zum NFS Server (beobachtet auf einem Netz-Monitoring Programm). Mit denselben Randbedingungen wie vorher (gemountet mit `async`, Quellverzeichnis im Cache, keine Änderungen) erhielt ich eine Beschleunigung des Backups (verglichen mit einem Backup ohne `lateLinks`) um den *Faktor 70*.
Wenn also Deine veränderten oder neuen Dateien (komprimiert) verglichen mit der verfügbaren Bandbreite nicht zu groß sind, kann `storeBackup` (mit `lateLinks` auch für Backups über VPNs oder Netzwerke mit hoher Latenz verwendet werden.¹² Natürlich sollte die Option `lateCompress` in so einem Fall nicht verwendet werden. Ein weitere Vorteil von `lateLinks` ist beim hier besprochenen Fall, dass die Parallelisierung aufgrund der Tatsache, dass das Lesen des Quellverzeichnisses kaum Interaktionen mit dem NFS Mount benötigt, besser funktioniert.

Fazit: Wenn man über NFS mountet, kann man `storeBackup` mit der Option `lateLinks` massiv beschleunigen. Zur Konfiguration siehe Kapitel 7.6.

Auch die Verwendung von „blocked files“ erhöht die Performance massiv, da in den Fällen, in denen sie sinnvollerweise verwendet werden, nur ein kleiner Prozentsatz der Dateien kopiert oder komprimiert werden muss. Siehe hierzu auch die Beschreibung über blocked files (siehe Kapitel 7.5) über den Einfluss dieser Option auf die Performance und den benötigten Plattenplatz.

5.4.4 Beispiellauf

Hier kann man eine statistische Ausgabe eines größeren Backup-Laufs sehen, der ein Backup vom Laptop auf einen NFS Server zeigt. Ich lasse diese Sicherung ein- oder zweimal in der Woche sowie einen kleineren täglich laufen. Und zwar so, wie in Beschreibung 3, Kapitel 9.4 aufgeführt.) Das Backup lief über mehr als 500.000 Einträge:

¹¹Die verwendete VPN Software war `openvpn`, die Verbindung wurde durch mehrere Firewalls getunnelt.

¹²Man kann zu große Dateien auch mit der Option `exceptRule` von `storeBackup.pl` vom Backup ausnehmen und später mit einer schnelleren Verbindung sichern.


```

1  STATISTIC 2008.09.08 23:40:17 3961 [sec] |      user|      system
2  STATISTIC 2008.09.08 23:40:17 3961 -----+-----+-----
3  STATISTIC 2008.09.08 23:40:17 3961 process|    386.30|    166.27
4  STATISTIC 2008.09.08 23:40:17 3961 childs |    209.02|    116.96
5  STATISTIC 2008.09.08 23:40:17 3961 -----+-----+-----
6  STATISTIC 2008.09.08 23:40:17 3961 sum      |    595.32|    283.23 => 878.55 (14m39s)
7  STATISTIC 2008.09.08 23:40:17 3961      directories = 43498
8  STATISTIC 2008.09.08 23:40:17 3961      files = 482516
9  STATISTIC 2008.09.08 23:40:17 3961      symbolic links = 12024
10 STATISTIC 2008.09.08 23:40:17 3961      late links = 462267
11 STATISTIC 2008.09.08 23:40:17 3961      named pipes = 3
12 STATISTIC 2008.09.08 23:40:17 3961      sockets = 48
13 STATISTIC 2008.09.08 23:40:17 3961      block devices = 0
14 STATISTIC 2008.09.08 23:40:17 3961      character devices = 0
15 STATISTIC 2008.09.08 23:40:17 3961      new internal linked files = 178
16 STATISTIC 2008.09.08 23:40:17 3961      old linked files = 462089
17 STATISTIC 2008.09.08 23:40:17 3961      unchanged files = 0
18 STATISTIC 2008.09.08 23:40:17 3961      copied files = 2896
19 STATISTIC 2008.09.08 23:40:17 3961      compressed files = 5204
20 STATISTIC 2008.09.08 23:40:17 3961      excluded files because rule = 78
21 STATISTIC 2008.09.08 23:40:17 3961      included files because rule = 0
22 STATISTIC 2008.09.08 23:40:17 3961      max size of copy queue = 22
23 STATISTIC 2008.09.08 23:40:17 3961      max size of compression queue = 361
24 STATISTIC 2008.09.08 23:40:17 3961      calculaed md5 sums = 50606
25 STATISTIC 2008.09.08 23:40:17 3961      forks total = 9176
26 STATISTIC 2008.09.08 23:40:17 3961      forks md5 = 3957
27 STATISTIC 2008.09.08 23:40:17 3961      forks copy = 12
28 STATISTIC 2008.09.08 23:40:17 3961      forks bzip2 = 5204
29 STATISTIC 2008.09.08 23:40:17 3961      sum of source = 10G (10965625851)
30 STATISTIC 2008.09.08 23:40:17 3961      sum of target all = 10.0G (10731903808)
31 STATISTIC 2008.09.08 23:40:17 3961      sum of target all = 97.87%
32 STATISTIC 2008.09.08 23:40:17 3961      sum of target new = 109M (114598007)
33 STATISTIC 2008.09.08 23:40:17 3961      sum of target new = 1.05%
34 STATISTIC 2008.09.08 23:40:17 3961      sum of md5ed files = 744M (779727492)
35 STATISTIC 2008.09.08 23:40:17 3961      sum of md5ed files = 7.11%
36 STATISTIC 2008.09.08 23:40:17 3961      sum internal linked (copy) = 32k (32472)
37 STATISTIC 2008.09.08 23:40:17 3961      sum internal linked (compr) = 6.2M (6543998)
38 STATISTIC 2008.09.08 23:40:17 3961      sum old linked (copy) = 3.3G (3515951642)
39 STATISTIC 2008.09.08 23:40:17 3961      sum old linked (compr) = 6.6G (7094777689)
40 STATISTIC 2008.09.08 23:40:17 3961      sum unchanged (copy) = 0.0 (0)
41 STATISTIC 2008.09.08 23:40:17 3961      sum unchanged (compr) = 0.0 (0)
42 STATISTIC 2008.09.08 23:40:17 3961      sum new (copy) = 11M (11090534)
43 STATISTIC 2008.09.08 23:40:17 3961      sum new (compr) = 99M (103507473)
44 STATISTIC 2008.09.08 23:40:17 3961      sum new (compr), orig size = 321M (336637589)
45 STATISTIC 2008.09.08 23:40:17 3961      sum new / orig = 32.96%
46 STATISTIC 2008.09.08 23:40:17 3961      size of md5Checksum file = 16M (16271962)
47 STATISTIC 2008.09.08 23:40:17 3961      size of temporary db files = 0.0 (0)
48 STATISTIC 2008.09.08 23:40:17 3961      precommand duration = 1s
49 STATISTIC 2008.09.08 23:40:17 3961      deleted old backups = 0
50 STATISTIC 2008.09.08 23:40:17 3961      deleted directories = 0
51 STATISTIC 2008.09.08 23:40:17 3961      deleted files = 0
52 STATISTIC 2008.09.08 23:40:17 3961      (only) removed links = 0
53 STATISTIC 2008.09.08 23:40:17 3961      freed space in old directories = 0.0 (0)
54 STATISTIC 2008.09.08 23:40:17 3961      add. used space in files = 125M (130869969)
55 STATISTIC 2008.09.08 23:40:17 3961      backup duration = 27m3s
56 STATISTIC 2008.09.08 23:40:17 3961      over all files/sec (real time) = 297.30
57 STATISTIC 2008.09.08 23:40:17 3961      over all files/sec (CPU time) = 549.22
58 STATISTIC 2008.09.08 23:40:17 3961      CPU usage = 54.13%

```

Das Backup lief 27 Minuten.

Sieh auf die Anzahl der berechneten md5-Summen: 50.606. Dieses ist die Anzahl der Dateien, die ein „gewöhnliches“ Backup (das den Inhalt nicht berücksichtigt) aufgrund eines geänderten Zeitstempels gespeichert hätte. (Ich hatte keine Daten verschoben; die Änderungen kamen hauptsächlich von OS Updates.) StoreBackup berechnet die md5 Summen und erkennt, dass sich rund 8.100 Dateien (copied + compressed files) geändert haben.

Hierdurch mussten nur 16% der Dateien, die normalerweise hätten gesichert werden müssen, gespeichert werden. Über die Zeit macht das einen großen Unterschied im benötigten Platz für die Sicherungen aus. Als weiterer Effekt kommt hinzu, dass die Dateien im Backup komprimiert werden.

Da das Backup mit Option `lateLinks` lief, musste später (über cron) `storeBackupUpdateBackup.pl` laufen

um die Links usw. zu setzen:

```
1  INFO      2008.09.09 02:17:52 13323 updating
   ↪ </disk1/store-backup/fschjc-gentoo-all/2008.09.08_23.13.14>
2  INFO      2008.09.09 02:17:52 13323 phase 1: mkdir, symlink and compressing files
3  STATISTIC 2008.09.09 02:18:18 13323 created 43498 directories
4  STATISTIC 2008.09.09 02:18:18 13323 created 12024 symbolic links
5  STATISTIC 2008.09.09 02:18:18 13323 compressed 0 files
6  STATISTIC 2008.09.09 02:18:18 13323 used 0.0 instead of 0.0 (0 <- 0)
7  INFO      2008.09.09 02:18:18 13323 phase 2: setting hard links
8  STATISTIC 2008.09.09 02:27:55 13323 linked 462267 files
9  INFO      2008.09.09 02:27:55 13323 phase 3: setting file permissions
10 STATISTIC 2008.09.09 02:31:05 13323 set permissions for 482442 files
11 INFO      2008.09.09 02:31:05 13323 phase 4: setting directory permissions
12 STATISTIC 2008.09.09 02:31:47 13323 set permissions for 43498 directories
```

Es dauerte 14 Minuten um das Backup für 500.000 Einträge zu vervollständigen.

6 Komponenten / Programme

storeBackup.pl	führt das Backup durch und generiert eine Konfigurationsdatei für sich selbst.
storeBackupUpdateBackup.pl	Wenn die Option <code>lateLinks</code> in storeBackup.pl verwendet wird, werden u.a. die notwendigen Hardlinks nicht direkt erzeugt. Damit läuft das Backup wesentlich schneller, insbesondere wenn über NFS gesichert wird. Dieses Programm überprüft alle Abhängigkeiten und generiert die Hardlinks. Als Resultat entsteht ein Backup mit denselben Eigenschaften wie ein Backup ohne <code>lateLinks</code> .
storeBackupRecover.pl	Spielt Dateien oder (Unter-)Verzeichnisbäume aus dem Backup zurück. Dabei werden alle Rechte restauriert und Hardlinks so gesetzt, wie sie vorher in den Quellverzeichnissen des Backups waren.
storeBackupVersions.pl	Ermittelt die Versionen von gesicherten Dateien.
storeBackupSearch.pl	Suchen nach verschiedenen Kriterien (Regeln) im Backup
st...upSetupIsolatedMode.pl	Kopiert Metadaten auf ein externes Medium, z. B. für Backups auf der Reise
st...upMergeIsolatedBackup.pl	Spielt Backups vom lokalen Medium zurück ins zentrale Master-Backup
storeBackupCheckBackup.pl	Überprüft die Integrität aller Dateien im Backup. Hierzu werden alle md5-Summen erneut berechnet und mit den gespeicherten verglichen.
storeBackupCheckSource.pl	Überprüft die Integrität aller unveränderten Dateien im Quellverzeichnis. Hierzu werden alle md5-Summen erneut berechnet und mit den gespeicherten verglichen.
storeBackuppls.pl	Listet die verschiedenen Backups (Verzeichnisse) auf und versieht sie mit zusätzlichen Informationen (Wochentag, Alter des Backups)
storeBackupDel.pl	Löscht alte Backups nach denselben Regeln wie storeBackup.pl. Hiermit können Backups asynchron gelöscht werden. Dieses Programm kann die Konfigurationsdatei von storeBackup.pl lesen.
storeBackupMount.pl	Dieses Programm kann verwendet werden, wenn ein Backup über NFS erstellt werden soll. Es pingt den Server an, mountet die Dateisysteme, ruft storeBackup.pl auf und unmountet die Dateisysteme. Es schreibt ein Log und hat eine detaillierte Fehlerbehandlung.
storeBackup_du.pl	Überprüft die Festplattenbelegung in einem oder mehreren Backup-Verzeichnissen.
storeBackupConvertBackup.pl	Konvertiert (sehr) alte Backups in neuere Formate. Dieses Programm sollte nur verwendet werden, wenn storeBackup.pl dazu auffordert.
linkToDirs.pl	kopiert / verlinkt Directories mit anderen
llt	Zeigt atime, ctime und mtime von Dateien an.
multiTail.pl	Ermöglicht es, mehrere Logdateien anzuzeigen. Hiermit können auch mehrere Logdateien zu einer zusammengeführt werden. Es ist robuster als 'tail -f'.

`st...up` ist eine Abkürzung für `storeBackup`.

Durch Aufruf der oben aufgeführten Programme mit der Option `-h` wird eine Kurzbeschreibung angezeigt.

6.1 Unterstützte Plattformen und Tipps

Von Anwendern weiß ich, dass die storeBackup-Tools unter GNU/Linux, FreeBSD, Solaris und AIX laufen. Sie sollten auf allen Unix-Plattformen lauffähig sein. Perl wird als Programmiersprache verwendet; daher ist eine lauffähige Perl-Implementierung für das Starten der Programme notwendig.

StoreBackup wird auf GNU/Linux entwickelt und getestet. Für alle Programme wird über die Option `-h` eine kurze Hilfe-Meldung generiert.

StoreBackup speichert seine Daten auf ein lokales Dateisystem – oder etwas, das wie ein lokales Dateisystem aussieht. Du kannst damit auf alle Dateisysteme (oder virtuelle Dateisysteme) sichern, die Hardlinks

und die Datentypen, die Du speichern willst, unterstützten (z. B. symbolische Links oder special files wie named pipes, falls Du die sichern willst). Die folgenden *Beispiele* zeigen einige der Möglichkeiten. (Wenn Du auf remote Dateisysteme¹³ speicherst, kannst Du die Sicherung durch Verwenden der Option `lateLinks` (siehe Kapitel 7.6) beschleunigen.

ext4 ist das aktuell (2012) schnellste Dateisystem für Linux. Es wird im Kernel gut unterstützt und wird für die vorhersehbare Zukunft verfügbar sein.

ext2 Du kannst dieses Dateisystem verwenden, aber es gibt einige Gründe, die dagegen sprechen: Dateisystem-Checks können „ewig“ dauern und es unterstützt keine Hashes für Dateinamen. Das bedeutet, dass der Zugriff auf die vielen kleinen Dateien, die von „blocked files“ generiert werden, langsam ist.

reiserfs ist das aktuell speicherplatzeffizienteste Dateisystem für Linux, da es „tail packing“ unterstützt. Der Platz in Dateisystemen wird in Blöcken verwaltet. Die Blockgröße ist typischerweise 4kB, daher werden pro Datei im Mittel etwa 2kB nicht genutzt. Wenn Du viele Dateien hast (z. B. bei Verwendung von „blocked files“ mit Kompression und daher undefinierter Dateilänge) bleibt ein größerer Anteil des verfügbaren Speicherplatzes ungenutzt. Mit „tail packing“ werden diese nicht gefüllten letzten Blöcke vom Dateisystem zusammengepackt. Reiserfs ist langsamer als ext4. Es wird im Kernel gut unterstützt und wird für die vorhersehbare Zukunft verfügbar sein.

vfat Dieses vorsintflutliche Dateisystem unterstützt keine Hardlinks oder die Unterscheidung von Dateinamen mit Groß- oder Kleinbuchstaben (versuche eine Datei mit Namen `DateiA` und eine mit `Dateia` im selben Verzeichnis anzulegen). Du kannst Deine Backups mit `storeBackup` nicht auf diesem Dateisystem speichern. Aber natürlich kannst Du Daten *von* einem derartigen Dateisystem mit `storeBackup` sichern.

ntfs Vorweg, die kannst Deine Backups auf einem ntfs Dateisystem sichern. Aber die Rechte und der Benutzer werden im Backup nicht verfügbar sein. Insbesondere, wenn Du ntfs auf einer externen Platte oder einem Memory Stick verwendest, könnte das egal sein. Lies die „Wichtige Anmerkung“ unter Punkt „CIFS“ weiter unten in dieser Liste!

NFS Das Network File System (Netzwerk-Datei-System) ermöglicht es Dir, Deine Backups transparent (also wie auf einer lokale Platte) über das Netzwerk zu speichern (siehe auch Konfiguration von NFS, Kapitel 7.11). Natürlich kannst Du die zu sichernden Daten auch mit `storeBackup` über NFS lesen, wenn Du sehr z. B. sehr alte Unix Maschinen sichern willst, für die es kein lauffähiges perl 5 gibt.

CIFS Es ist möglich, Deine Daten auf einem CIFS (Samba)-shares neu zu sichern. Abgesehen davon, dass es ein bisschen langsamer ist als NFS, unterstützt es keine Multiuser Mounts. Daher werden alle Deine Dateien im Backup *einem* User gehören. Falls Deine Umgebung eine Mehrbenutzerumgebung ist, in der jeder Benutzer direkten Zugriff auf nur seine Backupdaten haben soll, dann ist diese Art der Speicherung für Dich ungeeignet. Wenn aber jeder Benutzer alle Dateien im Backup sehen darf oder die Rücksicherung über einen Administrator läuft, dann ist es kein Problem einen Samba-Server (der oft die einzige Zugriffsmöglichkeit für kleine NAS Systeme ist) für die Speicherung von Backups zu verwenden. Natürlich kann man Daten auch über CIFS-shares lesen – es ist aber zu berücksichtigen, dass CIFS nur auf User-Basis gemounted werden kann. Es ist kein transparentes Netzwerk Dateisystem wie NFS.

Wichtige Anmerkung: Wenn Du Daten mit `storeBackupRecover.pl` aus dem Backup zurück sicherst, erhältst Du wieder die richtigen Rechte und Benutzer. `StoreBackupRecover.pl` schert sich in keiner Weise um die Rechte der Dateien im Backup. Die Meta-Informationen (inklusive Hard Links im gesicherten Verzeichnis) werden den Meta-Daten Dateien, die `storeBackup.pl` anlegt, entnommen. ABER, falls Du `storeBackup.pl` mit der Option `lateLinks` verwendest und es möglich ist, auf dem Backup Server `storeBackupUpdateBackup.pl` lokal laufen zu lassen, werden alle Rechte im Backup-Directory wie im Quellverzeichnis gesetzt.

sshfs Unter FAQ 4 gibt es eine kurze Beschreibung, wie sshfs zu konfigurieren ist. Lies die Kommentare zu CIFS im Punkt oben zur Beschreibung möglicher Einschränkungen.

¹³also nicht lokal auf dem Rechner gemountet, sondern über das Netz

6.2 storeBackup.pl

Dieses ist das Basisprogramm um Backups zu erzeugen. Neben vielen Optionen gibt es zwei Modi, die verwendet werden können:

1. Erzeugen eines Backups mit allem, was dazu notwendig ist: Kopieren, Komprimieren, Linken, Rechte setzen usw. Wenn Du Dich mit storeBackup noch nicht auskennst, solltest Du mit diesem Modus beginnen.
2. Erzeuge nur das absolut notwendige (Deltas) und überlasse den Rest storeBackupUpdateBackup.pl, was später gestartet wird. Dieses ist eine Art Client / Server Modus.

```
1 NAME
2 storeBackup.pl - fancy compressing managing checksumming hard-linking cp
3 -ua
4 DESCRIPTION
5 This program copies trees to another location. Every file copied is
6 potentially compressed (see --exceptSuffix). The backups after the first
7 backup will compare the files with an md5 checksum with the last stored
8 version. If they are equal, it will only make an hard link to it. It
9 will also check mtime, ctime and size to recognize identical files in
10 older backups very fast. It can also backup big image files fast and
11 efficiently on a per block basis (data deduplication).
12 You can overwrite options in the configuration file on the command line.
13 SYNOPSIS
14 storeBackup.pl --help
15 or
16 storeBackup.pl -g configFile
17 or
18 storeBackup.pl [-f configFile] [-s sourceDir]
19 [-b backupDirectory] [-S series] [--checkCompr] [--print]
20 [-T tmpdir] [-L lockFile] [--unlockBeforeDel]
21 [--exceptDirs dir1] [--contExceptDirsErr]
22 [--includeDirs dir1]
23 [--exceptRule rule] [--includeRule rule]
24 [--exceptTypes types]
25 [--specialTypeArchiver archiver [--archiveTypes]]
26 [--cpIsGnu] [--linkSymlinks]
27 [--precommand job] [--postcommand job]
28 [--followLinks depth] [--highLatency]
29 [--ignorePerms] [--lateLinks] [--lateCompress]]
30 [--checkBlocksSuffix suffix] [--checkBlocksMinSize size]
31 [--checkBlocksBS] [--checkBlocksCompr check|yes|no]
32 [--checkBlocksParallel] [--queueBlock]
33 [--checkBlocksRule0 rule [--checkBlocksBS0 size]
34 [--checkBlocksCompr0 key] [--checkBlocksRead0 filter]
35 [--checkBlocksParallel0]]
36 [--checkBlocksRule1 rule [--checkBlocksBS1 size]
37 [--checkBlocksCompr1 key] [--checkBlocksRead1 filter]
38 [--checkBlocksParallel1]]
39 [--checkBlocksRule2 rule [--checkBlocksBS2 size]
40 [--checkBlocksCompr2 kkey] [--checkBlocksRead2 filter]
41 [--checkBlocksParallel2]]
42 [--checkBlocksRule3 rule [--checkBlocksBS3 size]
43 [--checkBlocksCompr3 key] [--checkBlocksRead3 filter]
44 [--checkBlocksParallel3]]
45 [--checkBlocksRule4 rule [--checkBlocksBS4 size]
46 [--checkBlocksCompr4 key] [--checkBlocksRead4 filter]
47 [--checkBlocksParallel4]]
48 [--checkDevices0 list [--checkDevicesDir0]
49 [--checkDevicesBS0] [checkDevicesCompr0 key]
50 [--checkDevicesParallel0]]
51 [--checkDevices1 list [--checkDevicesDir1]
```

```

52 [--checkDevicesBS1] [checkDevicesCompr1 key]
53 [--checkDevicesParallel1]]
54 [--checkDevices2 list [--checkDevicesDir2]
55 [--checkDevicesBS2] [checkDevicesCompr2 key]
56 [--checkDevicesParallel2]]
57 [--checkDevices3 list [--checkDevicesDir3]
58 [--checkDevicesBS3] [checkDevicesCompr3 key]
59 [--checkDevicesParallel3]]
60 [--checkDevices4 list [--checkDevicesDir4]
61 [--checkDevicesBS4] [checkDevicesCompr4 key]
62 [--checkDevicesParallel1]]
63 [--saveRAM] [-c compress] [-u uncompress] [-p postfix]
64 [--noCompress number] [--queueCompress number]
65 [--noCopy number] [--queueCopy number]
66 [--withUserGroupStat] [--userGroupStatFile filename]
67 [--exceptSuffix suffixes] [--addExceptSuffix suffixes]
68 [--compressSuffix] [--minCompressSize size] [--comprRule]
69 [--doNotCompressMD5File] [--chmodMD5File] [-v]
70 [-d level] [--progressReport number[,timeframe]] [--printDepth]
71 [--ignoreReadError]
72 [--suppressWarning key] [--linkToRecent name]
73 [--doNotDelete] [--deleteNotFinishedDirs] [--maxHardLinks]
74 [--resetAtime] [--keepAll timePeriod] [--keepWeekday entry]
75 [--keepFirstOfYear] [--keepLastOfYear]
76 [--keepFirstOfMonth] [--keepLastOfMonth]
77 [--firstDayOfWeek day] [--keepFirstOfWeek]
78 [--keepLastOfWeek] [--keepDuplicate] [--keepMinNumber]
79 [--keepMaxNumber]
80 | [--keepRelative] ]
81 [-l logFile
82 [--plusLogStdout] [--suppressTime] [-m maxFilelen]
83 [--n noOfOldFiles] | [--saveLogs]]
84 [--compressWith compressprog]]
85 [--logInBackupDir] [--compressLogInBackupDir]
86 [--logInBackupDirFileName logFile]]
87 [otherBackupSeries ...]

```

Es müssen mindestens zwei Optionen gesetzt werden: --sourceDir und --backupDir. Es ist egal, ob sie auf der Kommandozeile, in der Konfigurationsdatei oder gemischt gesetzt werden.

Folgende Optionen können nur auf der Kommandozeile angegeben werden. Es gibt immer eine lange Version (wie --file) und manchmal auch eine kurze (-f).

--help Generiert einen langen Hilfetext mit einer Kurzbeschreibung aller Optionen.

--generate / -g Generiert eine Vorlage für eine Konfigurationsdatei. Nach dem Generieren kann diese mit einem Editor Deiner Wahl editiert werden. Ich empfehle die Verwendung einer Konfigurationsdatei, wenn mehr als simple Backups konfiguriert werden sollen.

--print Gibt die verwendeten Optionen aus (von der Kommandozeile und aus der Konfigurationsdatei) und terminiert danach. Im Fall von komplizierten Maskierungen (insbesondere auf der Kommandozeile) gibt diese Option die Möglichkeit zu sehen, was wirklich an das Programm übergeben wird.

--file / -f Gibt den Namen der Konfigurationsdatei an, die `storeBackup.pl` verwenden soll.

--checkCompr / -C Wenn Du nur ein einfaches Backup laufen lassen willst und Dich nicht auf die Voreinstellungen von „exceptSuffix“ (die eventuell nicht Deinen Bedürfnissen entsprechen) verlassen willst, dann setze diese Option. Sie überschreibt alle Einstellungen für „exceptSuffix“, „addExceptSuffix“, „minCompressSize“ und „comprRule“.

Wenn Du diese Option verwendest, wird `storeBackup.pl` für alle Dateien größer als 1k überprüfen, ob eine Kompression sinnvoll ist.

6.2.1 storeBackup.pl Optionen

Die folgenden Optionen können auf der Kommandozeile und in der Konfigurationsdatei verwendet werden (siehe Kapitel 7.1). Für die Kommandozeile existiert eine lange Version (wie `--sourceDir`), manchmal auch eine kurze (wie `-s`), sowie die Bezeichnung, die in der Konfigurationsdatei verwendet wird (wie `sourceDir`). Optionen, die ausschließlich auf der Kommandozeile aufgerufen werden können, sind im vorigen Kapitel 6.2 beschrieben.

`--sourceDir / -s / sourceDir` Pfad zum Verzeichnis, welches gesichert werden soll. Es kann nur *ein* Verzeichnis mit `storeBackup.pl` gesichert werden. Wenn Du mehr als ein Verzeichnis sichern willst, können die Optionen `--includeDirs`, `--exceptDirs` oder das komfortablere, empfohlene und unten beschriebene `--followLinks` verwendet werden.

`--backupDir / -b / backupDir` Das Backup-Verzeichnis, in dem *alle* Deine Sicherungen gespeichert werden. Falls Du nur eine Serie von Backups (z. B. von einem Computer) hast, wird dieses wahrscheinlich identisch mit dem Verzeichnis sein, in dem Deine Backups sind. In diesem Fall sollte die als nächstes beschriebene Option (`series`) auf `'.'` gesetzt werden. Beispiel:

```
backupDir = /backup
```

```
series = .
```

Dann sieht man die Backups direkt in `/backup`:

```
$ ls -l /backup
```

```
drwxr-xr-x 14 root root 528 Aug 24 21:33 2008.08.22_02.18.43
drwxr-xr-x 14 root root 528 Aug 24 21:33 2008.08.23_02.01.11
drwxr-xr-x 14 root root 528 Aug 24 21:33 2008.08.24_02.03.51
drwxr-xr-x 14 root root 528 Aug 24 21:33 2008.08.24_13.04.55
```

Falls sich in Deinem Backup-Verzeichnis unterschiedliche „Serien“ befinden, wirst Du unterschiedliche Unterverzeichnisse für diese unterschiedlichen Backup-Serien (die vielleicht von unterschiedlichen Computern stammen) erzeugen und `series` passend zu diesen Verzeichnisnamen einstellen. Lass uns annehmen, dass Du drei unterschiedliche Rechner sichern willst; „bob“, „joe“ und „bill“. In diesem Fall könnten drei unterschiedliche Verzeichnisse erzeugt werden:

```
$ ls -l /backup
```

```
drwxr-xr-x 2 root root 40 Aug 25 17:02 bill
drwxr-xr-x 2 root root 40 Aug 25 17:02 bob
drwxr-xr-x 2 root root 40 Aug 25 17:02 joe
```

Unterhalb dieser Verzeichnisse findest Du die individuellen Backups für „bill“, „bob“ und „joe“. Für „bill“ stellst Du z. B. ein:

```
backupDir = /backup
```

```
series = bill
```

Dann siehst Du Deine Backups in `/backup/bill`:

```
$ ls -l /backup
```

```
drwxr-xr-x 11 root root 432 Aug 24 21:33 2008.08.20_02.18.25
drwxr-xr-x 11 root root 432 Aug 24 21:33 2008.08.21_02.11.53
drwxr-xr-x 11 root root 432 Aug 24 21:33 2008.08.22_02.36.18
drwxr-xr-x 11 root root 432 Aug 24 21:33 2008.08.23_02.17.18
drwxr-xr-x 11 root root 432 Aug 24 21:33 2008.08.24_02.15.45
drwxr-xr-x 11 root root 432 Aug 24 21:33 2008.08.24_13.17.21
```

`--series / -S / series` Siehe auch Option `backupDir` oben.

Der Standardwert für `series` ist „default“. Um eine bestehende Serie umzubenennen, ist Folgendes zu tun:

- Lass `storeBackupUpdateBackup.pl` laufen, so dass noch nicht erzeugte Links (siehe auch Option `lateLinks` weiter unten) generiert werden.
- Benenne das Verzeichnis unterhalb von `backupDir` in den gewünschten Namen um.
- Konfiguriere diese Option (`series`) auf den Verzeichnisnamen, den Du im Schritt vorher gewählt hattest.

`--tmpdir / -T / tmpdir` Verzeichnis für temporäre Dateien, der Wert wird von der Umgebungsvariablen `$TMPDIR` übernommen. Falls diese nicht gesetzt ist, wird `/tmp` als Standardwert genommen.

- lockFile / -L / lockFile** `storeBackup.pl` verwendet eine Lock-Datei, um mehrfach laufende Instanzen zu verhindern. Der Standardname für diese Lock-Datei ist `/tmp/storeBackup.lock` für den User `root` und `$HOME/storeBackup.lock` für andere Benutzer.
Die verwendeten Lock-Dateien sind nicht für die Verwendung von `storeBackupUpdateBackup.pl` und `storeBackup.pl` oder anderen `storeBackup` Programmen in unterschiedlichen PID¹⁴ Namensräumen gedacht. Wenn Du Lock-Dateien über Rechnergrenzen hinweg nutzen willst, solltest Du Deine eigene Lösung bauen oder einen „Enterprise Job Scheduler“ verwenden.
- unlockBeforeDel / unlockBeforeDel** Lösche die Lock-Datei vor dem Löschen von alten Backups. Standard ist, die Lock-Datei nach dem Löschen von alten Sicherungen zu entfernen. Dieses „verkürzt“ die Backupzeit in gewissem Maße.
- exceptDirs / -e / exceptDirs** Du kannst eine Liste von Verzeichnissen angeben, die aus dem Backup ausgeschlossen werden. Es muss ein *relativer Pfad* angegeben werden, und zwar von dem Verzeichnis, das mit Option `sourceDir` angegeben wurde. Du kannst auch Wildcards verwenden. Ein Beispiel: Wenn alle User unterhalb von `sourceDir/home` angesiedelt sind und Du willst den Ordner `tmp` im User-Verzeichnis nicht sichern, kannst Du vorgeben:
`exceptDirs = home/*/tmp`
Um diese Wildcards zu interpretieren, verwendet `storeBackup.pl` eine Shell. Falls die resultierende Liste von Verzeichnissen zu lang wird (misst etwa 4K), funktioniert dieses Verfahren nicht mehr. In diesem Fall solltest Du auf die Option `exceptRule` (siehe weiter unten) ausweichen.
Um in der Konfigurationsdatei eine Liste von Verzeichnissen zu spezifizieren, schreib einfach:
`exceptDirs = home/*/tmp 'otherdir/tmp'`
Auf der Kommandozeile wird die Option einfach wiederholt:
`-e 'home/*/tmp' -e 'otherdir/tmp'`
Hier ist das Quoten von `home/*/tmp` wichtig, um die Interpretation des Ausdrucks bereits durch die gerade verwendete Shell zu vermeiden.
(In Kapitel 7.2 „Auswahl von Verzeichnissen / Dateien für die Sicherung“ findest Du eine Übersicht über die verschiedenen Möglichkeiten zum Auswählen und Ausschließen von Dateien oder Verzeichnissen.)
- contExceptDirsErr / contExceptDirsErr** Beim Setzen dieser Option wird `storeBackup.pl` auch bei Fehlen von einem oder mehreren mit `exceptDirs` spezifizierten Verzeichnissen die Sicherung fortführen. Die Voreinstellung ist, eine Fehlermeldung auszugeben und zu stoppen.
- includeDirs / -i / includeDirs** Wenn diese Option gesetzt ist, werden nur die Verzeichnisse gesichert, die hier angegeben sind. `StoreBackup.pl` wird nur Dateien sichern, die *nicht in exceptDirs* und *in includeDirs* sind.
(In Kapitel 7.2 „Auswahl von Verzeichnissen / Dateien für die Sicherung“ findest Du eine Übersicht über die verschiedenen Möglichkeiten zum Auswählen und Ausschließen von Dateien oder Verzeichnissen.)
- exceptRule / exceptRule** Wenn eine hier angegebene Regel zutrifft, wird die betroffene Datei aus dem Backup ausgeschlossen. Die Regel wird für reguläre („normale“) Dateien ausgeführt. Du kannst mehr über Regeln im Kapitel 7.4 erfahren.
(In Kapitel 7.2 „Auswahl von Verzeichnissen / Dateien für die Sicherung“ findest Du eine Übersicht über die verschiedenen Möglichkeiten zum Auswählen und Ausschließen von Dateien oder Verzeichnissen.)
- includeRule / includeRule** Wenn diese Option verwendet wird, werden Dateien, bei denen diese Regel zutrifft, gesichert. `StoreBackup.pl` sichert nur Dateien, die *nicht* ausgeschlossen werden und bei der `includeRule` passen. Du kannst mehr über Regeln im Kapitel 7.4 erfahren.
- writeExcludeLog / writeExcludeLog** Diese Option bewirkt, dass `storeBackup.pl` eine Datei mit den Namen der Dateien schreibt, die aufgrund von Regeln vom Backup ausgeschlossen werden. Diese Datei wird direkt im jeweiligen Backupverzeichnis mit dem Namen `.storeBackup.notSaved.bz2` gespeichert. Sie wird mit `bzip2` komprimiert.

¹⁴Prozess-ID

--exceptTypes / exceptTypes Die hier aufgeführten Dateitypen werden nicht gesichert. **StoreBackup.pl** erkennt:

- S – Socketdateien
- b – Blockdateien
- c – Zeichendateien
- f – gewöhnliche Dateien
- p – Pipes
- l – Symbolische Links

Sbc können nur gesichert werden, wenn auf dem Rechner **gnu-cp** im Pfad ist und Du die **gnucp** Option (siehe unten) aktiviert hast. Wenn Du angibst:

```
exceptTypes = Sbc
```

werden Dateien dieses Typs nicht im Backup gespeichert und es wird keine Warnung generiert. Diese Einstellung wird vor „exceptRule“ und „includeRule“ ausgewertet. Wenn einige Dateitypen generell ausgeschlossen werden sollen, benutze diese Option. (Sie ist schneller und auch einfacher zu verwenden.)

--archiveTypes archiveTypes Sichert den angegebenen Dateityp in einem Archiv statt direkt im Dateisystem. Dies ist nützlich, um „special files“ speichern zu können, obwohl das verwendete Dateisystem (z. B. NTFS) oder der verwendete Transportmechanismus (z. B. sshfs) dieses nicht unterstützt. Du kannst die folgenden Dateitypen auswählen:

- S – Socketdateien
- b – Blockdateien
- c – Zeichendateien
- p – Pipes

--specialTypeArchiver / specialTypeArchiver Mögliche Werte sind **cpio** and **tar**. Der Defaultwert ist **cpio**. Siehe Option **archiveTypes** oben.

Anmerkung: **tar** kann keine Sockets speichern; **cpio** ist nicht mehr Teil des aktuellen Posix-Standards.

--cpIsGnu / cpIsGnu Wenn diese Option gesetzt wird, können Dateien des Typs **Sbc** (siehe oben) gesichert werden. Für die Rücksicherung mit **storeBackupRecover.pl** wird **gnu-cp** ebenfalls benötigt. Wenn Du ein Linux System verwendest, ist Dein „cp“ Programm „gnu-cp“.

--linkSymlinks / linkSymlinks Wenn Du Deine Sicherung auf ein Dateisystem machst, das Hardlinks auf symbolische Links unterstützt, kannst Du diese Option aktivieren. GNU/Linux unterstützt dieses Feature. Die Voreinstellung ist, keine Hardlinks auf symbolische Links zu erlauben.

--precommand / precommand Du kannst *ein* Kommando oder Skript definieren, dass ausgeführt wird, bevor **storeBackup.pl** das Backup startet. Es starte erst, nachdem die Lock-Datei überprüft wurde. Wenn der Ausgabewert dieses Kommandos **!= 0** ist, wird **storeBackup.pl** sofort angehalten. Ausgaben dieses Kommandos auf **STDIN** werden als Warnung, die Ausgabe auf **STDERR** als Error in die **storeBackup.pl** Log Datei geschrieben.

Die Kommandozeilenparameter dieser Option werden genauso analysiert wie in der Konfigurationsdatei und sollten normalerweise gequotet werden. Das heißt, man kann bei dem Kommando Parameter verwenden, z. B.:

```
precommand = /backup/pre.sh param1 param2
```

ist dasselbe wie:

```
--precommand '/backup/pre.sh param1 param2'
```

Um während der Ausführung des Skriptes auf Parameter von **storeBackup** zurückgreifen zu können, werden die folgenden Shell-Variablen zur Verfügung gestellt:

- \$STBU_sourceDir,
- \$STBU_backupDir,
- \$STBU_series,
- \$STBU_tmpdir,
- \$STBU_lockFile,
- \$STBU_followLinks,
- \$STBU_lateLinks,
- \$STBU_lateCompress,
- \$STBU_verbose,

```

$STBU_doNotDelete,
$STBU_deleteNotFinishedDirs,
$STBU_logFile,
$STBU_logInBackupDir,
$STBU_logInBackupDirFileName.

```

--postcommand / postcommand Dieses Kommando wird nach Beendigung des Backups, aber vor dem Löschen alter Backups ausgeführt. `StoreBackup.pl` meldet, ob der Exit-Status !=0 ist. Der cli-Parameter für dieses Option wird wie eine Zeile in der Konfigurationsdatei interpretiert (siehe Option **precommand**). Dem über **postcommand** gestartetem Skript werden dieselben Shell-Variablen wie bei **precommand** zur Verfügung gestellt.

--followLinks / followLinks Wenn Du mehr als ein Verzeichnis sichern willst, solltest Du diese Option verwenden. Um z. B. `/boot`, `/etc` und `/home/tom` zu sichern, solltest Du (als root) so etwas wie das Folgende erzeugen:

```

1 # mkdir /backup
2 # cd /backup
3 # ln -s /boot boot
4 # ln -s /etc etc
5 # ln -s /home/tom home_tom
6 # ln -s . backup
7 # storeBackup.pl -g stbu.conf

```

Danach solltest Du in der Konfigurationsdatei (hier `stbu.conf` (neben anderem) einstellen:

```

1 # sourceDir = /backup
2 # followLinks = 1

```

Dies sagt `storeBackup.pl`, die erste Ebene (den ersten Level) von symbolischen Links unterhalb von `/backup` wie Verzeichnisse zu behandeln. Mit „`ln -s . backup`“ wird ein Unterverzeichnis im Backup erstellt, das exakt dem Verzeichnis `/backup` entspricht.

„`followLinks`“ konfiguriert `storeBackup.pl` so, dass *n* Level von symbolischen Links als Verzeichnisse interpretiert werden. Du kannst durch einfaches Setzen oder Löschen eines symbolischen Links beliebige Verzeichnisse Deines Dateisystems dem Backup in `/backup` hinzufügen und wieder entfernen.

--stayInFileSystem / stayInFileSystem Bei Verwendung dieser Option werden nur Dateien und Verzeichnisse gesichert, die über die Option **sourceDir** oder über die Hardlinks der Option **followLinks** spezifiziert wurden.

--highLatency / highLatency Verwende diese Option, wenn `storeBackup.pl` über eigene Leitung mit hoher Latenz sichert, wie z. B. bei einem VPN über das Internet. Dieses Option bedingt eine höhere Parallelisierung auf Kosten von erhöhtem CPU-Verbrauch. Es ist eine gute Idee, **--lateLinks** und je nach Anwendungsfall ggf. **--lateCompress** zu verwenden.

Verwende diese Option nicht, um reguläre Backups auf eine andere lokale Platte oder über NFS im lokalen Netzwerk durchzuführen.

--ignorePerms / ignorePerms Bei Setzen dieser Option werden Dateien im Backup nicht notwendigerweise dieselben Rechte und Eigentümer (owner) haben wie im Original. Dies beschleunigt das Backup. Eine Rücksicherung mit `storeBackupRecover.pl` wird die Rechte und Eigentümer der Dateien richtig zurücksichern. Es gibt verschiedene Möglichkeiten, die Performance zu verbessern, siehe Kapitel 5.4.3.

--lateLinks / lateLinks Diese Option reduziert die *direkte* Backup-Zeit zu Lasten eines zweiten Programms, das später laufen muss. Für ein lokales Backup auf eine andere Platte kann man eine Beschleunigung von 30–50% erwarten, bei einem Backup über NFS einen *Faktor* von 5 bis 10. Dieser Wert hängt sehr stark davon ab, wie viele neue Dateien zu sichern sind und wie schnell die Netzwerkanbindung ist. Bei einer Sicherung über ein VPN über das Internet habe ich eine Beschleunigung um einen Faktor 70 gemessen.

Wenn Du **lateLinks** verwenden willst, solltest Du Kapitel 7.6 lesen.

- `--lateCompress / lateCompress` Diese Option kann nur in Verbindung mit `lateLinks` verwendet werden. Sie bewirkt, dass die Kompression von Dateien $\geq \text{minCompressSize}$ erst erfolgt, wenn das Programm `storeBackupUpdateBackup.pl` läuft. Siehe auch Kapitel 7.6.
- `--checkBlocksSuffix` Die Konfiguration ist analog zu `exceptSuffix`; es wird eine Liste von zu überprüfenen Dateiendungen angegeben, z. B. `\.vdmk` für VMware Images. Das bedeutet lediglich, dass der letzte Teil des Dateinamens mit dem übereinstimmen muss, was hier definiert wird. Die nächsten hier beschriebenen Optionen werden nur ausgewertet, wenn `checkBlocksSuffix` gesetzt ist. Siehe Blocked Files (Kapitel 7.5) für mehr Informationen über diese Optionen mit „block“ im Namen.
- `--checkBlocksMinSize` Nur Dateien mit dieser Minimalgröße werden als „blocked files“ behandelt. Du kannst dieselben Kürzel verwenden wie in Definition von Regeln, siehe Kapitel 7.4 beschrieben, z. B. bedeutet 50M 50 Megabytes. Der Standardwert ist 100M.
- `--checkBlocksBS` Bestimmt die Blockgröße, in die die betroffenen Dateien aufgesplittet werden. Das Format dieser Option ist identisch zu `checkBlocksMinSize`. Der Standardwert ist 1M, der Minimalwert 10k.
- `--checkBlocksParallel` Gibt an, ob Dateien, die *nicht* hier (also in `checkBlockSuffix`) spezifiziert werden, parallel zu diesen abgearbeitet werden sollen. Das ist normalerweise nur dann sinnvoll, wenn die hier spezifizierten Dateien sehr klein oder auf einem eigenen Gerät (z. B. Festplatte) liegen. Die Voreinstellung ist `no`, das bedeutet keine Parallelisierung.
- `--checkBlocksCompr` Legt fest, ob die Blöcke komprimiert werden. Mögliche Werte sind `yes`, `no` oder `check`. Die Voreinstellung ist `no`. Diese Option betrifft nur Dateien, die mittels `checkBlocksSuffix` ausgewählt wurden. Wenn diese Option auf `check` gesetzt wird, wird jeder Block überprüft, ob eine Komprimierung (vermutlich) etwas bringt oder nicht; siehe Wie festlegen, ob eine Datei komprimiert werden soll (Kapitel 7.4.1).
- `--checkBlocksRulei` Die *i*te Regel zur Spezifizierung von Dateien, die als im Backup als „blocked files“ behandelt werden sollen. Es können 5 Regeln definiert werden, beginnend mit `checkBlocksRule0` bis `checkBlocksRule4`. Siehe Blocked Files (Kapitel 7.5) für mehr Informationen über die Optionen mit „block“ im Namen.
- `--checkBlocksBSi` Die zugehörige Blockgröße für die Blöcke im Backup. Der Standardwert ist 1M, der Minimalwert 10k.
- `--checkBlocksCompri` Legt fest, ob die Blöcke komprimiert werden. Mögliche Werte sind `yes`, `no` oder `check`. Der Standardwert ist `no`. Diese Option betrifft nur Dateien, die mittels `checkBlocksSuffix` ausgewählt wurden. Wenn diese Option auf `check` gesetzt wird, wird jeder Block überprüft, ob eine Komprimierung (vermutlich) etwas bringt oder nicht; siehe Wie festlegen, ob eine Datei komprimiert werden soll (Kapitel 7.4.1).
- `--checkBlocksReadi` Definiert den Filter für die zu spezifizierenden Dateien in `sourceDir`, z. B. `gunzip` oder `gzip -d`. Diese Option ist nützlich, wenn eine bereits komprimierte Imagedatei als „blocked file“ gesichert werden soll. (Das „blocked file“-Feature von `storeBackup` mit bereits komprimierten Dateien zu verwenden ist nicht sinnvoll.)
- `--checkBlocksParalleli` Gibt an, ob Dateien, die *nicht* hier (also in `checkBlockSuffix`) spezifiziert werden, parallel zu diesen abgearbeitet werden sollen. Dies ist normalerweise nur sinnvoll, wenn die hier spezifizierten Dateien sehr klein oder auf einem eigenen Gerät (z. B. Festplatte) liegen. Standard ist `no`, das bedeutet keine Parallelisierung.
- `--checkDevicesi` Liste der zu sichernden Devices (z. B. `/dev/sdd2 /dev/sde1`).
- `--checkDevicesDiri` Verzeichnis im Backup, in dem die zu sichernden Geräte (devices) gespeichert werden sollen (*relativer* Pfad). Das Gerät wird bei einer Rücksicherung mit `storeBackupRecover.pl` auch in diesem (relativen) Verzeichnis erstellt (bei Verwendung von Standardoptionen). In dem hier angegebenen Verzeichnis erzeugt `storeBackup.pl` ein oder mehrere Unterverzeichnisse, dessen Name aus den Parametern von `checkDevices` generiert wird (z. B. wird aus `/dev/sdc` dann `dev_sdc`).

- `--checkDevicesBSi` Definiert die Blockgröße, in welche die Devices durch `storeBackup.pl` aufgesplittet werden sollen. Das Format ist identisch zu dem von `checkBlocksMinSize`. Der Standardwert ist `1M`, der Minimalwert `10k`.
- `--checkDevicesCompri` Legt fest, ob die Blöcke komprimiert werden. Mögliche Werte sind `yes`, `no` oder `check`. Der Standardwert ist `no`.
Wenn diese Option auf `check` gesetzt wird, wird jeder Block überprüft, ob eine Komprimierung (vermutlich) etwas bringt oder nicht; siehe Wie festlegen, ob eine Datei komprimiert werden soll (Kapitel 7.4.1).
- `--checkDevicesParalleli` Gibt an, ob Devices, die *nicht* hier (also in `checkDevices`) spezifiziert werden, parallel zum Rest abgearbeitet werden sollen. Dies ist normalerweise nur sinnvoll, wenn die hier spezifizierten Geräte (Devices) auf einem eigenen Gerät (z. B. Festplatte) liegen.
Standard ist `no`, das bedeutet keine Parallelisierung.
Du solltest noch wissen, dass Dateien und Devices, die mit `checkBlocksRulei` oder `checkDevicesi` definiert wurden, *nie* parallel verarbeitet werden.
- `--saveRAM / saveRAM` Verwende diese Option, wenn `storeBackup.pl` auf einem System mit sehr wenig Hauptspeicher verwendet wird. Du wirst dann in `tmpDir` einige dbm-Dateien sehen. Durch Verwendung dieser Option wird `storeBackup.pl` ein bisschen langsamer, daher empfiehlt sich diese Option nur dann, wenn ohne sie Probleme auftreten. Auf aktuellen Computern sollte die Verwendung dieser Option nur notwendig sein, wenn viele Millionen Dateien zu sichern sind.
- `--compress / -c / compress` Hier wird das Kommando definiert, das `storeBackup.pl` für die Komprimierung verwendet. Standard ist `bzip2`.
Der Kommandozeilenparameter für diese Option wird wie eine Zeile in der Konfigurationsdatei geparkt und muss normalerweise auf der Kommandozeile gequotet werden. Das bedeutet, dass Parameter verwendet werden können, z. B.:
`compress = gzip -9`
was identisch ist mit:
`--compress 'gzip -9'`
- `--uncompress / -u / uncompress` Das Kommando, das `storeBackupRecover.pl` für das Entpacken von Dateien im Backup nutzt. Standard ist `„bzip2 -d“`. Dieser Wert *muss* zum Parameter der Option `compress` passen.
Der Kommandozeilenparameter für diese Option wird wie eine Zeile in der Konfigurationsdatei geparkt und muss normalerweise auf der Kommandozeile gequotet werden. Das bedeutet, dass Parameter verwendet werden können, z. B.:
`compress = gzip -d`
was identisch ist mit:
`--compress 'gzip -d'`
- `--postfix / -p / postfix` Die Endung, die `storeBackup.pl` für komprimierte Dateien verwendet. Diese sollte zur Option `compress` passend. Standard ist `.bz2`.
- `--noCompress / noCompress` Maximale Anzahl von parallelen Kompressions-Tasks. Bei GNU/Linux wird die Anzahl automatisch auf Anzahl der Cores + 1 gesetzt.
- `--queueCompress / queueCompress` Maximale Länge der Warteschlange für zu komprimierende Dateien. Der Standardwert ist 1000.
- `--noCopy / noCopy` Maximale Anzahl der parallelen Kopiervorgänge. Der Standardwert ist 1.
- `--queueCopy / queueCopy` Maximal Länge der Warteschlange für zu kopierende Dateien. Der Standardwert ist 1000.
- `--withUserGroupStat / withUserGroupStat` Schreibt Statistiken über den von Benutzern und Gruppen belegten Platz im `sourceDir` in die Logdatei.
- `--userGroupStatFile / userGroupStatFile` Schreibt Statistiken über den von Benutzern und Gruppen belegten Platz in die hier benannte Datei. Sie wird jedes Mal überschrieben.

`--exceptSuffix / exceptSuffix` Dateien mit den hier angegebenen Endungen werden nicht komprimiert. Diese Option kann auf der Kommandozeile wiederholt angegeben werden. Der Standardwert ist:

```
1 exceptSuffix = 'zip' 'bz2' 'gz' 'tgz' 'jpg' 'gif' 'tiff'
2 'tif' 'mpeg' 'mpg' 'mp3' 'ogg' 'gpg' 'png'
```

Du solltest einen Backslash (\) verwenden, um den Punkt zu maskieren, da ein Punkt allein irgendein Zeichen bedeutet (und nicht einen Punkt).

Um keine Datei zu komprimieren, kannst Du folgendes einstellen:

```
exceptSuffix = .*
```

`--addExceptSuffix / addExceptSuffix` Wenn Du zu der obigen Liste oben nur Dateierendungen hinzufügen willst, kannst Du diese Option verwenden. Auf der Kommandozeile kann diese Option mehrfach verwendet werden, um mehrere Suffixe hinzuzufügen. Siehe das Beispiel oben (bei `exceptSuffix`) zur Verwendung in der Konfigurationsdatei.

`--compressSuffix / compressSuffix` Liste der Suffixe der zu komprimierenden Dateien (Format wie bei `exceptSuffix`). Wenn Du diese Option verwendest, wird eine Regel gebildet aus den Optionen `exceptSuffix`, `addExceptSuffix` und `minCompressSize`. Weiterhin wird eine spezielle Regel-Funktion aufgerufen, die die durch die angegebenen Suffixe nicht abgedeckten Dateien daraufhin überprüft, ob eine Komprimierung lohnend sein dürfte. Einfache Beispiele sowie detaillierte Informationen hierzu werden in Wie festlegen, ob eine Datei komprimiert werden soll (Kapitel 7.4.1) erläutert.

`--minCompressSize / minCompressSize` Dateien, deren Größe unterhalb des hier angegebenen Wertes liegt, werden nicht komprimiert. Der Standardwert ist 1024.

Wenn dieser Wert bei zwei verlinkten Backups geändert wird (z.B. das erste Backup mit dem Standardwert und das zweite mit 512), so hat dies nur Auswirkungen auf Dateien mit neuem Inhalt. Dateien mit bereits im Backup existentem Inhalt werden zu den entsprechenden im Backup verlinkt. (In dem hier angesprochenen Beispiel für eine (neue) Datei mit 600 Byte im zweiten Backup würde diese nicht komprimiert werden, wenn eine derartige Datei mit demselben Inhalt bereits im Backup wäre.)

`--comprRule / comprRule` Diese Regel kann als Alternative zu den erwähnten Optionen `exceptSuffix`, `minCompressSize`, `addExceptSuffix` und `compressSuffix` definiert werden. Wenn sie gesetzt ist, werden die soeben genannten Optionen ignoriert (d.h. es wird *keine* Regel aus ihnen generiert). Siehe Definition von Regeln (Kapitel 7.4) für nähere Erläuterungen. Es könnte z.B. eine Regel definiert werden, die bewirkt, dass die Daten für bestimmte Benutzer zur einfacheren Rücksicherung (durch die Benutzer selbst) nicht komprimiert werden.

`--doNotCompressMD5File / doNotCompressMD5File` `storeBackup.pl` speichert Informationen über jede Datei im Top-Level-Directory des jeweiligen Backups in einer Datei namens `.md5CheckSums`. Sie wird normalerweise mit `bzip2` komprimiert. Das Setzen dieser Option unterbindet die Kompression. Verwende diese Option nur, wenn Dein Rechner sehr langsam ist oder nur einen Core hat. In diesem Fall wird `storeBackup` etwas schneller sein.

`--chmodMD5File` Jeder, der `storeBackupRecover.pl` verwenden können soll, muss die Datei `.md5CheckSums` lesen können (siehe auch Option oben). Die voreingestellten Rechte auf dieser Datei sind 0600, was bedeutet, dass nur der Erzeuger des Backups Zugriff darauf hat. Mit dieser Option kann der Zugriff auf andere ausgedehnt werden. Das bedeutet jedoch ein mögliches Sicherheitsproblem; in der Datei `.md5CheckSums` werden md5-Summen, Zeiten, UID, GUID, Modi und andere Informationen gespeichert.

Der direkte Zugriff auf die Dateien im Backup ist unabhängig von dieser Option.

`--verbose / -v / verbose` Erzeuge zusätzliche Meldungen.

`--debug / -d / debug` Erzeuge Debug-Meldungen:

0 – keine Debug-Meldungen (default)

1 – einige Debug-Meldungen

2 – viele Debug-Meldungen

Diese Option ist insbesondere in Kombination mit den Optionen `exceptRule` und `includeRule` nützlich.

- resetAtime / resetAtime** Stellt die Access Time (Zeit des letzten Zugriffs) im Backup auf die im Quellverzeichnis ein, ändert aber die ctime (Zeit der Dateierzeugung). Im Allgemeinen wirst Du diese Option nicht verwenden wollen.
- doNotDelete / doNotDelete** Führt alle Überprüfungen bezüglich des Löschens von Backups aus, löscht aber nichts. Diese Option ist nützlich bei der Verwendung von `storeBackupDel.pl`, das die Konfigurationsdatei von `storeBackup.pl` lesen kann. `StoreBackupDel.pl` kann alte Backups später asynchron löschen.
Um die Regeln darüber, welche Backups gelöscht werden sollen, zu verstehen, siehe die „keep*“ Optionen unten.
- deleteNotFinishedDirs / deleteNotFinishedDirs** Lösche Backups, die nicht beendet wurden und daher nicht vollständig sind. `StoreBackup.pl` und `storeBackupDel.pl` löschen nicht beendete Backups nur, wenn die Option `doNotDelete` auf `yes` gesetzt ist oder in der Kommandozeile verwendet wurde. Der Defaultwert für diese Option ist `no`.
- maxHardLinks / maxHardLinks** Gibt die maximale Anzahl von Hardlinks auf eine Datei vor, die von diesem Programm gesetzt wird. Der Default-Wert ist „0“, was bedeutet, dass die maximale Anzahl der des verwendeten Dateisystems entspricht. Die Verwendung dieser Option macht Sinn, wenn ein Backup (z.B. mit `cp -a`) auf ein anderes Dateisystem, was weniger Hardlinks als das Originale unterstützt, kopiert werden soll.
Die Option begrenzt nur die neu erzeugten Hardlinks – wenn die Anzahl schon über dem Limit ist, wird sie nicht reduziert! Um eine Reduktion der maximalen Anzahl von Hardlinks bei bestehenden Verzeichnissen zu erreichen, können diese unter Verwendung des Kommandos `linkToDirs.pl` (Kapitel 6.16) mit entsprechendem Setzen von Option `--maxHardLinks` kopiert werden.
- keepAll / keepAll** Behalte alle Backups einer Serie für den hier spezifizierten Zeitraum. Dieser Wert verhält sich wie ein Standardwert für *alle* Tage in der Option `keepWeekday` (siehe unten). Das Löschen alter Backups erfolgt erst dann, wenn das aktuelle Backup durchgelaufen ist oder mittels `storeBackupDel.pl`. Der Zeitraum muss im Format „dhms“ eingestellt werden, z. B. bedeutet „10d2h“ 10 Tage und 2 Stunden. Derartiges (plus 2 Stunden) ist sinnvoll, wenn Du 10 Tage festlegen willst; sonst kann eine Abweichung des Programmstarts von ein paar Minuten oder Sekunden als Resultat eine 9 tägige Aufbewahrungszeit ergeben. `StoreBackup` rechnet intern in Sekunden. Der Standardwert ist „30d“.
- keepWeekday / keepWeekday** Diese Option überschreibt die Einstellungen der Option `keepAll` für spezielle Wochentage. `Mon,Wed:40d5m Sat:60d10m` bedeutet:
- behalte Backups von Montag bis Mittwoch für 40 Tage + 5 Minuten
 - behalte Backups von Samstag für 60 Tage + 10 Minuten
 - Behalte Backups für die anderen Wochentage so lange wie in der Option `keepAll` angegeben.
- Du kannst auch das „Archiv Flag“ verwenden; es bedeutet, dass die betroffenen Backups nicht aufgrund der Option `keepMaxNumber` gelöscht werden. `Mon,Wed:a40d5m Sat:60d10m` bedeutet:
- behalte Backups von Montag und Mittwoch für 40 Tage + 5 Minuten + Archiv
Wenn Du mehr als in `keepMaxNumber` festgelegte Backups hast, dann werden die Backups von Montag oder Mittwoch, die in diese Kategorie fallen, nicht gelöscht.
 - behalte Backups von Samstagen für 60 Tage + 10 Minuten
Wenn Du mehr als `keepMaxNumber` Backups hast und Backups von Samstagen in diese Kategorie fallen, werden sie gelöscht.
 - behalte Backups der anderen Wochentage wie in der Option `keepAll` spezifiziert. Wenn Du mehr Backups hast als in `keepMaxNumber` spezifiziert, werden diese gelöscht, sofern sie in diese Kategorie fallen.
- Auf der Kommandozeile werden die Parameter zu dieser Option wie in der Konfigurationsdatei geparkt und müssen daher gequotet werden.
- keepFirstOfYear / keepFirstOfYear** Lösche das erste existierende Backup eines Jahres nicht. Das Format ist ein Zeitraum (siehe Option `keepAll` mit einem möglichen „Archiv Flag“.

`--keepLastOfYear / keepLastOfYear` Lösche das letzte existierende Backup eines Jahres nicht. Das Format ist ein Zeitraum (siehe Option `keepAll` mit einem möglichen „Archiv Flag“.

`--keepFirstOfMonth / keepFirstOfMonth` Lösche das erste existierende Backup eines Monats nicht. Das Format ist ein Zeitraum (siehe Option `keepAll` mit einem möglichen „Archiv Flag“.

`--keepLastOfMonth / keepLastOfMonth` Lösche das letzte existierende Backup eines Monats nicht. Das Format ist ein Zeitraum (siehe Option `keepAll` mit einem möglichen „Archiv Flag“.

`--firstDayOfWeek / firstDayOfWeek` Setzt den ersten Tag einer Woche für die Berechnungen, die auf `keepFirstOfWeek` und `keepLastOfWeek` basieren.
Der Standardwert ist „Sun“ (Sonntag).

`--keepFirstOfWeek / keepFirstOfWeek` Lösche das erste existierende Backup einer Woche nicht. Das Format ist ein Zeitraum (siehe Option `keepAll` mit einem möglichen „Archiv Flag“.

`--keepLastOfWeek / keepLastOfWeek` Lösche das letzte existierende Backup einer Woche nicht. Das Format ist ein Zeitraum (siehe Option `keepAll` mit einem möglichen „Archiv Flag“.

`--keepDuplicate / keepDuplicate` Behalte mehrfache Backups eines Tages für den hier spezifizierten Zeitraum. Wenn die Backups eines Tages älter als der hier angegebene Zeitraum sind, werden alle außer dem letzten Backup des betroffenen Tages gelöscht. Die Verwendung des „Archiv Flags“ ist nicht möglich. Das Format ist bei Option `keepAll` beschrieben.
Der Standardwert ist „7d“.

`--keepMinNumber / keepMinNumber` Behalte mindestens die hier angegebene Anzahl von Backups. Mehrfache Backups eines Tages zählen dabei als ein Backup.
Der Standardwert ist „10“.

`--keepMaxNumber / keepMaxNumber` Versuche, nur die hier angegebene Anzahl von Backups zu behalten. Wenn mehr Backups als hier angegeben existieren, wird die folgende Sequenz zum Löschen durchlaufen:

- Lösche alle Duplikate eines Tages, beginnend mit dem ältesten, mit Ausnahme des letzten Backups eines Tages.
- Wenn dies nicht reicht, lösche den notwendigen Rest von Backups, beginnend mit dem ältesten, aber *nie* ein Backup mit dem „Archiv Flag“ oder das letzte Backup. Siehe Option `keepWeekday` für Erläuterungen zum „Archiv Flag“.

`--keepRelative / -R / keepRelative` Diese Option ist ein alternatives Lösch-Schema. Wenn diese Option gesetzt ist, werden alle anderen `keep*` Optionen ignoriert. Auf der Kommandozeile werden die Parameter zu dieser Option wie eine Zeile in der Konfigurationsdatei geparkt und müssen daher i.d.R. gequotet werden.
Dieses Backup-Schema erlaubt es, das relative Alter von Backups, die Du behalten willst festzulegen – anstelle der Periode, die ein Backup erhalten werden soll.
Stell Dir vor, Du hättest immer die folgenden Backup verfügbar:

- 1 Backup von gestern
- 1 Backup von letzter Woche
- 1 Backup vom letzten Monat
- 1 Backup von vor drei Monaten

Beachte, dass das sehr wahrscheinlich *nicht* das ist, was Du willst, und zwar weil es ganz einfach bedeutet, dass Du tägliche Backups durchführen musst und jedes Backup für exakt 3 Monate behalten musst. Ansonsten könntest Du kein Backup vorhalten, das das geforderte *exakte* Alter hat. Was Du wirklich willst, ist daher so etwas wie:

- 1 Backup mit einem Alter zwischen 1 Stunde und 24 Stunden (1 Tag)
- 1 Backup mit einem Alter zwischen 1 Tag und 7 Tage
- 1 Backup mit einem Alter zwischen 14 und 31 Tagen

- 1 Backup mit einem Alter zwischen 80 und 100 Tagen

Das wäre eine sehr gewöhnliche Backup-Strategie, aber Du würdest Schwierigkeiten haben, dieses mit den anderen `keepFirstOf*` Optionen zu definieren, insbesondere, wenn Du die Backups nicht regelmäßig durchführst. Allerdings kannst Du dieses Verhalten sehr einfach mit `keepRelative` durchführen. Du musst nur folgendes angeben:

```
keepRelative = 1h 1d 7d 14d 31d 80d 100d
```

Das heißt, Du listet hier alle Intervalle auf, für die Du Backups haben willst. `storeBackup` wird die Backups in einer Art und Weise löschen, die dem gewünschten so nahe wie möglich kommt. (Wenn Du nicht genügend Backups machst, kann aber auch `storeBackup` nichts dagegen unternehmen.)

Beachte, dass dies bedeuten kann, dass `storeBackup` mehr Backups erhalten muss als Du denkst; z. B. könnte es zwei Backups in einer Periode behalten. In diesen Fällen „sieht `storeBackup` in die Zukunft“ und stellt fest, dass beide Backups *später* nötig sind, um Backups für alle Perioden vorzuhalten. Dies ist auch der Grund, warum im obigen Beispiel implizit die Periode 7-14 Tage spezifiziert wird, auch wenn Du in ihr kein Backup haben willst. Um Backups in der nächsten Periode (14-31 Tage) zu haben, benötigst Du immer auch ein Backup in der Periode 7-14 Tage. Aus diesem Grund erlaubt es die Syntax nicht, Perioden auszuschließen.

Schlussendlich solltest Du darauf achten, dass `storeBackup` alle Intervalle, für es kein passendes Backup finden kann, verschiebt: Wenn Dein erstes Backup zwischen 10 und 20 Tage sein soll, aber das nächste aktuelle 25 Tage alt ist, werden alle folgenden Perioden um 5 Tage verlängert. Wenn Du über einen längeren Zeitraum keine Backups gemacht hast, stellt dieses Verhalten sicher, dass dieser Zeitraum Dein Backup-Schema nicht durcheinanderbringt. Hier ein Beispiel, warum dieses sinnvoll ist: Wenn du Backups haben willst, die 1, 3, 7 und 10 Tage alt sind und Du gehst für 14 Tage in Urlaub, dann ist es sehr unwahrscheinlich, dass Du alle Backups gelöscht haben willst, wenn Du zurückkommst. Daher ignoriert `storeBackup` diese 14 Tage und behält die Backups entsprechend länger.

`--progressReport / -P / progressReport` Schreibe den Fortschritt nach der hier angegebenen Anzahl von Dateien in die Logdatei. Wenn Du eine Meldung spätestens nach einem bestimmten Zeitraum haben willst, kann Du diese durch ein Komma separiert anfügen, z. B.:

```
-P 1000,1m10s
```

auf der Kommandozeile, oder

```
progressReport = 1000,1m10s
```

in der Konfigurationsdatei.

Es dürfen keine Leerstellen im Parameter zu der Option enthalten sein. Die Syntax für den Zeitraum ist dieselbe wie bei den `keep*` Optionen.

`--printDepth / -D / printDepth` Schreibe die aktuell zu lesende Verzeichnisbaumtiefe während des Backups in die Logdatei.

`--ignoreReadError / ignoreReadError` Das Setzen dieser Option bewirkt das Ignorieren von Lesefehlern durch `storeBackup.pl`. Auf diese Art veranlassen nicht-lesbare Verzeichnisse `storeBackup.pl` nicht dazu, die Verarbeitung abubrechen. Diese Option wurde implementiert, um Shares von Windows Servern lesen zu können, die manchmal derartige Fehler liefern.

Normalerweise wird diese Option nicht benötigt.

`--suppressWarning / suppressWarning` Unterdrücke (ungewünschte) Warnungen, die ansonsten in die Logdatei oder nach stdout geschrieben würden. Dies ist eine Option für erfahrene Anwender. *Für normale Verwendung von `storeBackup` kann diese Option ignoriert werden.* In einigen Situationen kann es vorkommen, dass ein erfahrener Benutzer bestimmte Warnungen nicht mehr sehen möchte. Diese Option erlaubt es, derartige Warnungen abzuschalten. Sie ist nur für bestimmte nicht-kritische Warnungen verfügbar, z. B. für fehlende oder ausgeschlossene Verzeichnisse, für Dateien, die sich während des Backups ändern und für die Erzeugung der „default“ Serie.¹⁵

¹⁵Der Sinn dieser Option liegt darin, dass Anwender durch wiederholte, nicht sonderlich wichtige Warnungen dazu verleitet werden können, Warnungen generell nicht (mehr) zu beachten. Hier ein Beispiel, wie Du von dieser Option profitieren kannst: Angenommen, Du hast eine Liste von Verzeichnissen definiert, die vom Backup ausgenommen werden sollen, z. B. temporäre Verzeichnisse. Teilweise begrenzt Du auch die Anzahl der in das Backup einzubeziehenden Verzeichnisse. Wenn Du die einzubeziehenden Verzeichnisse auf eine Art und Weise begrenzt, dass sie nicht Teil des Backups sind, wird `storeBackup` für jedes dieser „fehlenden“ Verzeichnisse eine Warnung generieren. Trotzdem könntest Du beschließen, die auszuschließenden Verzeichnisse in der Konfigurationsdatei zu belassen, weil Du nicht riskieren willst, bei Veränderungen der Liste etwas zu vergessen. Aber Du möchtest die Warnungen auch nicht sehen, weil zu viele nicht-kritische Warnungen Dich davon abhalten, die wichtigen zu beachten. In einer derartigen Situation solltest du diese Option verwenden. Das bedeutet, dass Du bei

- Das Verwenden des `excDir`-Schlüsselworts unterdrückt Warnungen über nicht-existente exclude-Verzeichnisse.
- Das Verwenden des `filechange`-Schlüsselworts unterdrückt alle Warnungen darüber, dass festgestellt wurde, dass sich der Inhalt einer Datei während des Backups geändert hat.
- Das Verwenden des `crSeries`-Schlüsselworts unterdrückt die Warnung, falls `storeBackup.pl` ein Verzeichnis für die „default“ Serie anlegen muss.
- Das Verwenden des `hashCollision`-Schlüsselworts unterdrückt Warnungen, dass `storeBackup.pl` mögliche md5-Hash-Kollisionen gefunden hat.
- Das Verwenden des `fileNameWithLineFeed`-Schlüsselworts unterdrückt Warnungen über Dateinamen mit „line feed“ im Namen.
- Das Verwenden des `use_DB_File` Schlüsselwortes unterdrückt die entsprechende Warnung, falls in Deiner perl-Installation das CPAN Module `DB_File` nicht installiert ist. Du solltest es – wenn möglich – installieren; es erhöht die Performance und verringert den Speicherverbrauch.
- Das Verwenden des `use_MLDBM` Schlüsselwortes unterdrückt die Warnung, die generiert wird, wenn Du die Regel-Funktionen `MARK_DIR` oder `MARK_DIR_REC` in Kombination mit Option mit der Option `saveRAM` verwenden könntest *und* das perl CPAN Modul `MLDBM` auf Deinem Rechner nicht installiert ist.
- Das Verwenden des `use_IOCompressBzip2` Schlüsselwortes unterdrückt die Warnung, dass Du für eine bessere Performance das perl CPAN Modul `IO::Compress::Bzip2` installieren solltest. Ignoriere / unterdrücke diese Meldung, wenn Du `bzip2` nicht als Kompressions-Medhode verwendest.
- Das Verwenden des `noBackupForPeriod` Schlüsselwortes unterdrückt bei Verwendung von Option `keepRelative` die Warnung, dass für einen bestimmten Zeitraum keine Backups verfügbar sind.

`--linkToRecent` Durch Setzen dieser Option wird nach erfolgreichem Backup ein symbolischer Link mit dem hier definierten Namen auf dieses gerade durchgeführte Backup gesetzt. Wenn ein älterer symbolischer Link existiert, wird er gelöscht. Wenn der Name des symbolischen Links geändert wurde, wird er *nicht* automatisch gelöscht und muss manuell entfernt werden.

`--logFile / -l / logFile` Name der Logdatei. Standard ist stdout (BildschirmAusgabe).

`--plusLogStdout / plusLogStdout` Wenn die Option `logFile` gesetzt ist, kann hier konfiguriert werden, dass `storeBackup.pl` das Log zusätzlich auf stdout ausgibt.

`--suppressTime / suppressTime` Unterdrücke die Ausgabe der aktuellen Zeit in der Logdatei.

`--maxFilelen / -m / maxFilelen` Maximale Größe einer Logdatei. Nachdem diese Größe erreicht wird, wird die Logdatei rotiert (siehe Option `noOfOldFiles`) oder komprimiert (siehe Option `saveLogs`).

`--noOfOldFiles / -n / noOfOldFiles` Anzahl alter Logdateien, die rotiert werden sollen. Der Standardwert ist 5. Mit den Standardwerten sieht das so aus:

```

1 $ ls -l /tmp/storebackup.log*
2 -rw----- 1 hjc  root  328815 30. Aug 12:12 /tmp/storebackup.log
3 -rw----- 1 root root 1000087 27. Aug 21:18 /tmp/storebackup.log.1
4 -rw----- 1 root root 1000038 20. Aug 19:02 /tmp/storebackup.log.2
5 -rw----- 1 root root 1000094 11. Aug 18:51 /tmp/storebackup.log.3
6 -rw----- 1 root root 1000147 11. Aug 18:49 /tmp/storebackup.log.4
7 -rw----- 1 root root 1000030 11. Aug 18:49 /tmp/storebackup.log.5

```

Logdateien, die älter als *.5 sind, werden automatisch gelöscht.

einer Änderung der `includeDirs`-Liste nur *eine* Änderung (nämlich dort) anstelle von zwei (`includeDirs` und `exceptDirs`) durchgeführt werden muss. Allerdings gibt es auch Situationen, in denen die Unterdrückung von Warnungen über fehlende auszunehmende Verzeichnisse negative Konsequenzen haben kann: Angenommen, du hast ein temporäres Verzeichnis mit Namen `tests`, das ausgeschlossen werden soll. Wenn Du `tests` z.B. in `app1_tests` umbenennst (es aber nach wie vor nicht sichern willst), so wird es dann gesichert werden. Falls Du diese Art von Warnungen nicht unterdrückt hast, würde `storeBackup` Dich darüber informieren, dass `tests` nicht gefunden werden kann. Das würde Dich an die Änderung erinnern. Verwende diese Option daher mit Bedacht. Wenn Du Dir nicht sicher bist, ob Du sie verwenden solltest, solltest Du es besser nicht tun.

--saveLogs / saveLogs Speichere die Logs mit Datums- und Zeitstempel, statt sie nach dem Rotieren zu löschen. Durch Setzen dieser Option wird die Option **noOfOldFiles** deaktiviert.

--compressWith / compressWith Spezifiziert das Programm, mit dem die zu sichernden Logdateien gesichert werden sollen (z. B. **gzip -9**). Der Standardwert ist **bzip2**.
Die Parameter für diese Option werden auf der Kommandozeile so ausgewertet wie in der Konfigurationsdatei. Das bedeutet, dass sie auf der Kommandozeile gequotationet werden müssen.

--logInBackupDir / logInBackupDir Schreibt eine zusätzliche Logdatei in das Backup-Verzeichnis. Der Standardname ist **.storeBackup.log**, siehe auch Option **logInBackupDirFileName** unten. Diese Logdatei beinhaltet eventuell nicht alle Fehlermeldungen der anderen Logdateien, da das Backup-Verzeichnis ja erst existieren muss, bevor in dieses Log geschrieben werden kann.
Dieses Log ist nützlich, um historische Logdateien zu haben, während das „globale“ Log (von Option **logFile**) nützlich für das Monitoring ist.

--compressLogInBackupDir / compressLogInBackupDir Bewirkt die Komprimierung der Logdatei im Backup-Verzeichnis, falls spezifiziert.

--logInBackupDirFileName / logInBackupDirFileName Dateiname der Logdatei, die im Backup-Verzeichnis gespeichert werden soll. Der Standardwert ist **.storeBackup.log**.

...otherBackupSeries... / otherBackupSeries Auf der Kommandozeile ist dies keine Option, sondern ein List-Parameter. So muss auf der Kommandozeile geschrieben werden:

```
1 # storeBackup.pl <all_options> 0:server2 0-2:server3
```

In der Konfigurationsdatei entspräche das:

```
otherBackupSeries = 0:server2 0-2:server3
```

Hier kann eine Liste von anderen Backups angegeben werden, die für das Setzen von Hardlinks berücksichtigt werden sollen. Der Pfad zu den anderen Verzeichnissen muss ein relativer Pfad von **backupDir** sein!

Format (Beispiel):

```
1 otherSeries/2002.08.29_08.25.28 ## -> consider exactly this otherSeries
```

oder

```
1 0:otherSeries ## -> last (youngest) in <backupDir>/otherSeries
2 1:otherSeries ## -> first before last in <backupDir>/otherSeries
3 n:otherSeries ## -> n'th before last in <backupDir>/otherSeries
4 3-5:otherSeries ## -> 3rd, 4th and 5th in <backupDir>/otherSeries
5 all:otherSeries ## -> all in <backupDir>/otherSeries
```

Wenn Du hier nichts angibst, werden automatisch die neuesten Backups von allen anderen Serien in dem übergeordneten Verzeichnis genommen, das mit **backupDir** angegeben wurde.

Für die Konfiguration von **otherBackupSeries** können auch Wildcards verwendet werden. Mehr hierzu findet sich am Anfang des Kapitels 7.8.7 „Verwendung von Wildcards für die Replikation“.

6.3 storeBackupUpdateBackup.pl

Dieses Programm wird nur benötigt, wenn bei **storeBackup.pl** die Option **lateLinks** verwendet wurde. Siehe Kapitel 5.4.3 über Performance und warum Du diese Option vielleicht verwenden willst. Kapitel 7.6 zeigt Dir, wie Du sie verwenden kannst.

StoreBackupUpdateBackup.pl übernimmt die Aufgabe, Hardlinks, symbolische Links und Dateirechte zu setzen, Verzeichnisse zu erzeugen und Dateien zu komprimieren – Dinge, die **storeBackup.pl** bei Verwendung von **lateLinks** nicht macht. Bevor **storeBackupUpdateBackup.pl** das macht, überprüft es zuerst die Konsistenz des Backup-Bereichs, die aus der Verwendung von **lateLinks** resultiert; z. B. würde es erkennen, wenn ein Backup fehlt.

Um Inkonsistenzen zu beheben, kann **storeBackupUpdateBackup.pl** im interaktiven Modus betrieben werden (Option **--interactive**).

Verwende die Optionen `--genBackupBaseTreeConf` und `--genDeltaCacheConf`, um Konfigurationsdateien für die Replikation von Backups zu erstellen.

```
1 storeBackupUpdateBackup.pl - updates / finalizes backups created by
2 storeBackup.pl with option --lateLink, --lateCompress
3 SYNOPSIS
4 storeBackupUpdateBackup.pl -b backupDirectory [--autorepair]
5 [--print] [--verbose] [--debug] [--lockFile] [--noCompress]
6 [--progressReport number] [--checkOnly] [--copyBackupOnly]
7 [--dontCopyBackup] [-A archiveDurationDeltaCache]
8 [--dontDelInDeltaCache] [--createNewSeries]
9 [--noWarningDiffSeriesInBackupCopy] [--maxHardLinks]
10 [--logFile
11 [--suppressTime] [-m maxFilelen]
12 [[-n noOfOldFiles] | [--saveLogs]]
13 [--compressWith compressprog]]
14 storeBackupUpdateBackup.pl --interactive --backupDir topLevlDir
15 [--autorepair] [--print]
16 storeBackupUpdateBackup.pl --genBackupBaseTreeConf directory
17 storeBackupUpdateBackup.pl --genDeltaCacheConf directory
```

Die einzige Option, die angegeben werden muss, ist `backupDir`, die restlichen Optionen sind optional. Dieses Programm akzeptiert nur die Kommandozeile. Es kann nicht mit einer Konfigurationsdatei verwendet werden.

`--interactive / -i` Interaktiver Modus zum Reparieren / Löschen korrupter Backups, die mit `lateLinks` erzeugt wurden.

`--backupDir` Das Backup-Verzeichnis, in dem *alle* Deine Backups gespeichert sind. Wenn Du eine Serie von Backups (z. B. von einem Computer) hast, wird das normalerweise das Verzeichnis sein, in dem Deine Backups sind. Dieser Parameter ist identisch mit dem gleichnamigen Parameter von `storeBackup.pl` zu verstehen, siehe auch Kapitel 6.2.

`--autorepair / -a` Wenn `storeBackupUpdateBackup.pl` Inkonsistenzen findet, die das Gesamtbackup nicht beschädigen, werden bei Verwendung dieser Option die Referenzen automatisch ohne Nachfrage repariert. Es wird nur eine INFO-Meldung in die Logdatei ausgegeben und dokumentiert, was repariert wurde.

Wenn Du z. B. Dein *letztes* Backup mit `lateLinks` mit dem Kommando `rm` löscht (was Du nicht tun solltest bevor dieses Programm erfolgreich gelaufen ist!), dann ist die interne Struktur der Referenzen im Backup inkonsistent. `StoreBackupUpdateBackup.pl` (und auch `storeBackup.pl` werden feststellen, dass ein Backup, auf das referenziert wird, fehlt. Die Korrektur dieses Referenzierung führt aber nicht zu Datenverlust, deshalb kann die Struktur der Referenzen in diesem Beispiel ohne Benutzerinteraktion repariert werden. (Für mehr Informationen siehe Kapitel 7.9 über spezielle Dateien.)

`--print` Gibt die verwendeten Optionen aus und beendet das Programm danach. Im Falle von komplexem Quoting erleichtert dies ggf. die Fehlersuche.

`--verbose / -v` Erzeuge zusätzliche Meldungen.

`--debug / -d` Erzeuge detaillierte Informationen über Hardlinks, Kompression usw.

`--lockFile / -L / lockFile` `storeBackup.pl` verwendet eine Lock-Datei, um mehrfach laufende Instanzen zu verhindern. Der Standardname für diese Lock-Datei ist `/tmp/storeBackup.lock` für den User `root` und `$HOME/storeBackup.lock` für andere Benutzer.

Die verwendeten Lock-Dateien sind nicht für die Verwendung von `storeBackupUpdateBackup.pl` und `storeBackup.pl` oder anderen `storeBackup` Programmen in unterschiedlichen PID¹⁶ Namensräumen gedacht. Wenn Du Lock-Dateien über Rechnergrenzen hinweg nutzen willst, solltest Du Deine eigene Lösung bauen oder einen „Enterprise Job Scheduler“ verwenden.

¹⁶Prozess-ID

--noCompress Maximale Anzahl der parallel laufenden Kompressionsprogramme. Der Standardwert wird bei GNU/Linux automatisch gewählt und ist Anzahl der Cores plus 1.

--checkOnly / -c Bei Verwendung dieser Option wird keine Veränderung am Backup vorgenommen; nur die Konsistenz wird überprüft. Benutze diese Option in Verbindung mit **--debug**, um detaillierte Informationen zu bekommen.

--copyBackupOnly Falls Du die Replikation verwendest: kopiere nur zu / vom Delta Cache, kein Hardlinken, Komprimieren, Rechte setzen

--dontCopyBackup Führe keinerlei Replikationsaufträge aus.
WICHTIG: Wenn diese Option auf ein Master-Backup angewendet wird, unterbricht sie den Datenfluss für die Replikation!

--archiveDurationDeltaCache / -A Zeit, nach der bereits in die Backup-Kopie kopierte und verlinkte Backups gelöscht werden. Der Zeitraum muss im Format „dhms“ angegeben werden; z. B. bedeutet 10d4h 10 Tage und 4 Stunden, der Standardwert ist 99d. (Das Format ist identisch zur Option **keepAll** bei **storeBackup.pl**.)

--dontDelInDeltaCache Lösche kein Backup im Delta Cache

--createNewSeries Erzeuge neue Serie(n) im Delta Cache automatisch. Dieses ist insbesondere nützlich bei Verwendung von Wildcards, wenn neue Serien ins Backup für die Replikation „spontan“ erstellt werden sollen.

--noWarningDiffSeriesInBackupCopy Schreibe keine **WARNING** falls im Replikations-Verzeichnis andere Serien sind als im Delta Cache.

--maxHardLinks Gibt die maximale Anzahl von Hardlinks auf eine Datei vor, die von diesem Programm gesetzt wird. Der Default-Wert ist „0“, was bedeutet, dass die maximale Anzahl der des verwendeten Dateisystems entspricht. Die Verwendung dieser Option macht Sinn, wenn ein Backup (z.B. mit **cp -a**) auf ein anderes Dateisystem, was weniger Hardlinks als das Originale unterstützt, kopiert werden soll.
 Die Option begrenzt nur die neu erzeugten Hardlinks – wenn die Anzahl schon über dem Limit ist, wird sie nicht reduziert! Um eine Reduktion der maximalen Anzahl von Hardlinks bei bestehenden Verzeichnissen zu erreichen, können diese unter Verwendung des Kommandos **linkToDirs.pl** (Kapitel 6.16) mit entsprechendem Setzen von Option **--maxHardLinks** kopiert werden.

--progressReport Gib eine Fortschrittsanzeige abhängig vom hier eingestellten Wert *Anzahl* aus:
 nach jeweils *Anzahl* Dateien beim Komprimieren
 nach jeweils *Anzahl* * 1000 Dateien beim Linken
 nach jeweils *Anzahl* * 10000 beim Ändern der Rechte.

--logFile / -l Name der Logdatei. Standard ist die Ausgabe auf stdout.

--suppressTime Unterdrücke die Ausgabe der aktuellen Zeit in der Logdatei.

--maxFilelen / -m Maximale Größe einer Logdatei. Nachdem diese Größe erreicht wird, wird das Log file rotiert (siehe Option **noOfOldFiles**) oder komprimiert (siehe Option **saveLogs**).

--noOfOldFiles / -n Anzahl alter Logdateien, die rotiert werden sollen. Standardwert ist 5. Mit den Standardwerten sieht das so aus:

```

1 $ ls -l /tmp/storebackup.log*
2 -rw----- 1 hjc  root  328815 30. Aug 12:12 /tmp/storebackup.log
3 -rw----- 1 root root 1000087 27. Aug 21:18 /tmp/storebackup.log.1
4 -rw----- 1 root root 1000038 20. Aug 19:02 /tmp/storebackup.log.2
5 -rw----- 1 root root 1000094 11. Aug 18:51 /tmp/storebackup.log.3
6 -rw----- 1 root root 1000147 11. Aug 18:49 /tmp/storebackup.log.4
7 -rw----- 1 root root 1000030 11. Aug 18:49 /tmp/storebackup.log.5

```

Logdateien, die älter als *.5 sind, werden automatisch gelöscht.

- `--saveLogs` Speichere die Logs mit Datums- und Zeitstempel, statt sie nach dem Rotieren zu löschen. Durch Setzen dieser Option wird die Option `noOfOldFiles` deaktiviert.
- `--compressWith` Spezifiziert das Programm, mit dem die zu sichernden Logdateien gesichert werden sollen (z. B. `gzip -9`). Der Standardwert ist `bzip2`.
Parameter für diese Option werden auf der Kommandozeile so ausgewertet wie in der Konfigurationsdatei. Das bedeutet, dass sie auf der Kommandozeile gequotet werden müssen.
- `--genBackupBaseTreeConf` Falls Du Deine Backups replizieren willst, kannst Du diese Option verwenden, um eine der benötigten Konfigurationsdateien zu generieren. Siehe Replikation von Backups, Kapitel 7.8 für Details.
- `--genDeltaCacheConf` Falls Du Deine Backups replizieren willst, kannst Du diese Option verwenden, um eine der benötigten Konfigurationsdateien zu generieren. Siehe Replikation von Backups, Kapitel 7.8 für Details.

6.4 storeBackupRecover.pl

Spielt den Backup-Verzeichnisbaum oder Teile daraus aus dem Backup zurück.

```

1 storeBackupRecover.pl -r restore [-b root] -t targetDir [--flat]
2 [-o] [--tmpdir] [--noHardLinks] [-p number] [-v] [-n]
3 [--cpIsGnu] [--noGnuCp] [-s]
```

Der einfachste Weg, eine Datei oder eine kleine Anzahl von Dateien aus dem Backup zurückzuholen, ist, dies mit `cp` oder einem Dateimanager zu erledigen.

Dieses Tool dient zur Zurücksicherung (und wenn nötig zur Entkomprimierung). Es erstellt die Daten aus dem Backup heraus so, wie sie vorher im Original vorlagen: Rechte werden gesetzt (auch wenn `ignorePerms` in `storeBackup.pl` gesetzt war; diese Option betrifft nur die Rechte im Backup selbst) und auch vorher existierende Hardlinks werden wieder erzeugt.

Es sind mindestens zwei Optionen anzugeben: `restoreTree` und `targetDir`. `storeBackupRecover.pl` unterstützt nur die Kommandozeile.

- `--restoreTree / -r` Der Backup-Baum oder der Teil davon, der wiederhergestellt werden soll. Der einfachste Weg etwas zurückzusichern ist, in das Backup-Verzeichnis an die Stelle zu gehen, von wo zurückgesichert werden soll. Nehmen wir an, der Name des Verzeichnisses ist `mydir`. Dann gib ein:
`# storeBackupRecover.pl -r mydir -t /tmp/myRestorePlace`
wobei `/tmp/myRestorePlace` die Stelle ist, an die Du das Verzeichnis und seinen gesamten Inhalt zurückgespielt haben willst (siehe auch Option `targetDir`).

- `--backupRoot / -b` Normalerweise sollte keine Notwendigkeit bestehen, diese Option zu verwenden. Wenn Du ein Verzeichnis zurücksicherst, wird `storeBackupRecover.pl` nach der Datei `.md5Check.info` suchen, die im Wurzelverzeichnis von jedem Backup liegt. Wenn es mehr als eine solche Datei findet, wird eine ERROR-Meldung erzeugt. Das passiert, wenn man ein Backup von einem Backup macht und dieses dann zurücksichern will.

Wenn Du eine ERROR-Meldung „found info file a second time ...“ bekommst, dann musst Du das Wurzelverzeichnis des Backups, aus dem Du mit der Option `restoreTree` zurücksichern willst, angeben.

- `--targetDir / -t` Das Verzeichnis, in das die Dateien aus dem Backup zurückgesichert werden sollen. Falls Du die Option `flat` *nicht* verwendest, wird immer der gesamte Pfad im Backup an der hier spezifizierten Stelle erzeugt.

- `--flat` Die Verzeichnisstruktur wird nicht wiederhergestellt. Alle Dateien werden direkt im `targetDir` gespeichert. Dieses ist nur sinnvoll, wenn lediglich eine kleine Anzahl von Dateien zurückgeholt werden sollen.

- `--overwrite / -o` Überschreibe existierende Dateien. Das ist normalerweise keine gute Idee. Es ist i.d.R. besser, in ein separates Verzeichnis zurückzusichern und die Dateien später zu verschieben.

- `--tmpdir / -T /tmpdir` Verzeichnis für temporäre Dateien, der Wert wird von der Umgebungsvariablen `$TMPDIR` übernommen. Falls diese nicht gesetzt ist, wird `/tmp` als Standardwert genommen.

--noHardLinks Erzeuge bei der Rücksicherung keine Hardlink-Strukturen, sondern kopiere die Dateien einzeln.

--noRestoreParallel / -p Maximale Anzahl von parallel zu startenden Prozessen, um Dateien zu entkomprimieren. Der Standardwert ist 12.
Reduziere diese Anzahl, wenn Du „blocked files“ restaurierst und Dein Rechner nicht genügend Hauptspeicher hat.

--verbose / -v Generiere zusätzliche Meldungen.

--noRestored / -n Schreibe nach der Restaurierung die Anzahl von zurückgesicherten Verzeichnissen, Hardlinks, symbolischen Links, Dateien, ...

--noGnuCp Wenn `storeBackup.pl` so konfiguriert wurde, dass `gnucp` (Option `cpIsGnu`) verwendet wurde, damit spezielle Dateien wie zeichenorientierte Devices gesichert werden können, liest das Programm `storeBackupRecover.pl` diese Information aus dem Backup. Wenn jedoch der Rechner, auf dem Du zurücksicherst, `gnucp` nicht beherrscht, kannst Du `storeBackupRecover.pl` mit Hilfe dieser Option so konfigurieren, `cp` nicht zu verwenden.
Falls Du Deine Backups ohne `gnucp` gemacht hast, wird `storeBackupRecover.pl` diese Funktionalität nicht verwenden. Es gibt dazu keinen Grund, weil keine speziellen Dateien gesichert werden konnten. Wenn Du nur GNU/Linux Systeme verwendest, vergisst Du diese Option am Besten.

--createSparseFiles / -s Erzeugt Sparse Dateien aus als „blocked Files“ gesicherten Dateien, wenn komplette Blöcke im Backup mit Nullen gefüllt sind.

6.5 storeBackupVersions.pl

`storeBackupVersions.pl` identifiziert unterschiedliche Versionen einer Datei, die mit `storeBackup.pl` gesichert wurden. Verwende dieses Programm, wenn Du sehen willst, wie viele unterschiedliche Versionen einer bestimmten Datei existieren und wo sich eine Datei mit einem bestimmten Inhalt in einer Backup-Serie befindet.

Wenn Du eine etwas „ausgefuchstere“ Suche benötigst, die auf Dateinamen, Größe, Datum und anderem basiert, schau Dir `storeBackupSearch`, siehe Kapitel 6.6 an.

```
1 storeBackupVersions.pl -f file [-b root] [-v]
2 [-l [-a | [-s] [-u] [-g] [-M] [-c] [-m]]]
```

Dieses Programm akzeptiert nur Optionen auf der Kommandozeile. Die einzige Option, die gesetzt werden muss, ist `--file`.

--file / -f Name (und Pfad) der Datei im Backup. Schreibe den Namen genau so, wie er im Backup geschrieben ist.

--backupRoot / -b Normalerweise sollte keine Notwendigkeit bestehen, diese Option zu verwenden. Wenn Du ein Verzeichnis zurücksicherst, wird `storeBackupRecover.pl` nach der Datei `.md5Check.info` suchen, die im Wurzelverzeichnis von jedem Backup liegt. Wenn es mehr als eine solche Datei findet, wird eine ERROR-Meldung erzeugt. Das passiert, wenn man ein Backup von einem Backup macht und dieses dann zurücksichern will.

Wenn Du eine ERROR Meldung „found info file a second time ...“ bekommst, dann musst Du das Wurzelverzeichnis des Backups, aus dem Du mit der Option `restoreTree` zurücksichern willst, angeben.

--verbose / -v Generiere zusätzliche Meldungen.

--locateSame / -l Lokalisiere Dateien mit demselben Inhalt im Backup.

--showAll / -A Dasselbe wie `-s -u -g -M -c -m`

--size / -s Zeige die Größe (angenehm lesbar) der Datei im Original (vor Sicherung) an.

--uid / -u Zeige die UID vor der Sicherung an.

--gid / -g Zeige die GID vor der Sicherung an.

`--mode / -M` Zeige die Berechtigung vor der Sicherung an.

`--ctime / -c` Zeige die creation time (Erzeugungszeit) vor der Sicherung an.

`--mtime / -m` Zeige die modification time (Veränderungszeit) vor der Sicherung an.

`--atime / -a` Zeige die access time (Zugriffszeit) vor der Sicherung an.

6.6 storeBackupSearch.pl

`StoreBackupSearch.pl` erlaubt die Suche in speziellen Backups, in einer Backup Serie oder in allen Backups unter `backupDir`. Du kannst aus einer frei wählbaren Kombination von Dateiname (Pfad), Größe, Berechtigungen, Besitzer (UID, GID), Erstellungszeit, Modifikationszeit und Dateityp aus dem ursprünglichen Quellverzeichnis eine Regel für die Suche definieren.

Siehe Kapitel 7.4 dazu, wie Regeln definiert werden. Dies wird Dir weiterhelfen, wenn Du zumindest rudimentäre Kenntnisse über Scripting oder Programmierung hast.

```
1 storeBackupSearch.pl -g configFile
2
3 storeBackupSearch.pl [-f configFile] [-b backupDirectory]
4 [-s rule] [--absPath] [-w file] [--parJobs number]
5 [-d level] [--once] [--print] [-T tmpdir] [backupRoot . . .]
```

Dieses Programm erlaubt das Setzen von Optionen auf der Kommandozeile und in einer Konfigurationsdatei. Die Optionen `backupDir` und `searchRule` müssen gesetzt werden.

Zuerst werden hier die Optionen beschrieben, die nur auf der Kommandozeile möglich sind. Es gibt immer eine lange Option (wie `--file`) und teilweise auch eine Kurzbezeichnung (`-f`).

`--generate / -g` Erzeuge eine Vorlage für eine Konfigurationsdatei. Danach kannst Du diese Vorlage mit einem Editor Deiner Wahl editieren. Es ist einfacher, die Regeln in eine Konfigurationsdatei zu schreiben, weil die Shell auf der Kommandozeile Quotes entfernt.

`--print` Gibt die gewählten Optionen (von der Konfigurationsdatei und von der Kommandozeile) aus und stoppt das Programm. Im Falle von kompliziertem Quoting (insbesondere auf der Kommandozeile) gibt Dir das die Möglichkeit zu sehen, was wirklich vom Programm ausgewertet wird.

`--file / -f` Name der zu verwendenden Konfigurationsdatei.

Die folgenden Optionen können auf der Kommandozeile und in der Konfigurationsdatei verwendet werden (siehe Kapitel 7.1). Es gibt immer eine lange Version für die Kommandozeile (wie `--searchRule`) und teilweise eine Kurzform für die Kommandozeile (wie `-s`) sowie die Bezeichnung in der Konfigurationsdatei (wie `searchRule`).

`--backupDir / backupDir` Das Backupverzeichnis, in dem gesucht werden soll. Diese Option kann auf das gesamte Backup-Repository, auf eine Serie oder ein einzelnes Backupverzeichnis gesetzt werden.

`--searchRule / -s / searchRule` Die Regel zum Suchen, siehe Kapitel 7.4.

`--absPath / -a / absPath` Die Ausgabe der Dateien erfolgt mit absoluten Pfadnamen.

`--writeToFile / -w / writeToFile` Die Ausgabe der Suche wird in die hier angegebene Datei geschrieben; Standardwert ist stdout.

`--parJobs / -p / parJobs` Maximale Anzahl von parallelen Suchoperationen. Der Standardwert wird bei GNU/Linux automatisch als Anzahl der Cores + 1 gewählt.

`--debug / -d / debug` Debug Level; mögliche Werte sind 0, 1 und 2; Standard ist 0.

`--once / -o / once` Zeige jede gefundene Datei nur einmal an (Kriterium ist der Dateinhalt bzw. die md5-Summe jeder Datei).

`--tmpdir / -T / tmpdir` Verzeichnis für temporäre Dateien, der Wert wird von der Umgebungsvariablen `$TMPDIR` übernommen. Falls diese nicht gesetzt ist, wird `/tmp` als Standardwert genommen.

...backupRoot... / backupRoot Auf der Kommandozeile ist dieses keine Option, sondern ein List-Parameter, z. B.:

```
1 # storeBackupSearch.pl <all_options> 2008.08.27_16.59.01 2008.08.30_10.13.38
```

In der Konfigurationsdatei entspricht dies:

```
backupRoot = 2008.08.27_16.59.01 2008.08.30_10.13.38
```

Für die Suche kann hier ein *relativer Pfad* zu den zu durchsuchenden Verzeichnissen angegeben werden. Dieser kann eine bestimmte Sicherung selbst (wie in diesem Beispiel) oder eine ganze Backup-Serie sein, in der Verzeichnisse mit Sicherungen (und so weiter) gespeichert wurden. Auf diese Art kann in genau spezifizierten Verzeichnissen gesucht werden.

Es werden die Leserechte benötigt, um die `.md5Checksum.*`-Dateien im Backup zu lesen.

6.7 storeBackupSetupIsolatedMode.pl

storeBackupSetupIsolatedMode.pl ist Teil der Isolated Mode / Offline Backup Funktionalität¹⁷ von storeBackup. Es kopiert die Metadaten des letzten Backups einer Serie von Backups in ein anderes Dateisystem (z. B. auf einen Memory Stick oder eine kleine Festplatte). Es generiert optional eine angepasste Version der storeBackup.pl Konfigurationsdatei.

Dies kann dazu verwendet werden, inkrementelle Backups mit storeBackup.pl auf ein Medium mit geringer Kapazität zu schreiben, z. B. auf einen Memory Stick während einer Reise – ohne Zugriff auf das zentrale Backup zu haben. Später können die lokalen Backups in das zentrale Master-Backup mit Hilfe des Kommandos storeBackupMergeIsolatedBackup.pl¹⁸ integriert werden. Es kopiert die Daten vom Stick in das Master-Backup. Ein Lauf von storeBackupUpdateBackup.pl¹⁹ to komplettiert das inkrementelle Backup zu einem Vollbackup.

In Isolated Mode / Offline Backups, siehe Kapitel 7.7 findest du eine allgemeine Beschreibung darüber, wie der Isolated Mode verwendet wird.

Basierend auf einer Konfigurationsdatei:

```
1 storeBackupSetupIsolatedMode.pl -f configFile -t targetDir
2 [-S series] [-g newConfigFile]
3 [-e explicitBackup] [-v] [-F]
```

ohne Konfigurationsdatei:

```
1 storeBackupSetupIsolatedMode.pl -b backupDir -t targetDir
2 [-S series] [-e explicitBackup] [-v] [-F]
```

--configFile / -f Originäre Konfigurationsdatei von storeBackup.pl, die für die normalen Backups zum Master-Backup verwendet wird. In dieser Konfigurationsdatei werden die Zeichen # und ; als Kommentar-Zeichen verwendet. Die Anpassung der Konfiguration funktioniert nur dann richtig, wenn vor nicht verwendeten Schlüsselworten das ; verwendet wird (so wie in der generierten Originalversion von storeBackup.pl)!

storeBackupSetupIsolatedMode.pl erzeugt einen zusätzlichen Eintrag in der neu generierten Konfigurationsdatei (siehe Option --generate) mit Namen mergeBackupDir, der auf das ursprüngliche backupDir-Verzeichnis verweist. Dieser Eintrag wird von storeBackup.pl ignoriert. Er wird von storeBackupMergeIsolatedBackup.pl dazu verwendet, die inkrementellen Backups vom lokalen Medium in das Master-Backup zu kopieren.

Wenn Du nach einiger Zeit nochmals die Meta-Daten für ein identisches „Isolated Backup“ auf einen sauberen (alte Backups löschen!) Memory-Stick kopieren willst, kannst Du dieses Programm mit der in der Vergangenheit durch dieses Programm generierten Konfigurationsdatei aufrufen.

--targetDir / -t Das neue Top-Level Backup-Verzeichnis auf dem lokalen Medium (z. B. Memory Stick), in das storeBackup.pl sichern soll. Dieses Verzeichnis muss bereits existieren.

--backupDir / -b Falls Du keine Konfigurationsdatei von storeBackup.pl verwenden willst, kannst Du den Pfad zu Deinem Master-Backup hiermit festlegen.

¹⁷isolierter Modus, Modus ohne Verbindung zum Master Backup

¹⁸siehe storeBackupMergeIsolatedBackup.pl

¹⁹siehe storeBackupUpdateBackup.pl

--series / -S Wenn mehr als eine Serie in Deinem Master-Backup (**backupDir**) gespeichert ist, musst Du über diese Option festlegen, welche auf dem lokalen Medium verwendet werden soll.

--generate / -g Legt den Namen für die zu erzeugende Konfigurationsdatei fest, falls die Option **--configFile** benutzt wird. Wenn kein Name angegeben wird, wird er nach dem Muster **isolate--** plus dem Namen, der bei **--configFile** angegeben wurde, erstellt.

--explicitBackup / -e Legt das exakte Backup fest, das kopiert werden soll. Der Defaultwert ist das letzte Backup der gewählten Serie.

--verbose / -v Gibt zusätzliche Meldungen aus.

--force / -F Erzwingt die Verwendung des letzten Backups (mit Option **lateLinks** von **storeBackup.pl**) der Serie, auch wenn dieses letzte Backup der Serie noch nicht mit **storeBackupUpdateBackup.pl** zu einem Full-Backup vervollständigt wurde.

6.8 storeBackupMergeIsolatedBackup.pl

storeBackupMergeIsolatedBackup.pl ist Teil der Isolated Mode / Offline Backup Funktionalität von **storeBackup**. Es kopiert die inkrementellen Backups, die im Isolated Mode durchgeführt wurden, in das Master-Backup (siehe Isolated Mode / Offline Backups, Kapitel 7.7 und **storeBackupSetupIsolatedMode**, Kapitel 6.7).

Basierend auf einer Konfigurationsdatei:

```
1 storeBackupMergeIsolatedBackup.pl -f isolateConfigFile [-v] [--force]
2 [-T tmpdir]
```

Keine Konfigurationsdatei:

```
1 storeBackupSetupIsolatedMode.pl -t targetDir -b backupDir
2 [-S series] [-v] [--force] [-T tmpdir]
```

--configFile / -f Für den Isolated Mode verwendete Konfigurationsdatei, die den zusätzlichen Schlüsselbegriff **mergeBackupDir**, der auf das Master-Backup verweist, enthält.

--force Erzwingt das Kopieren der Dateien, ohne die Backups aufzulisten oder Sicherheitsabfragen zu stellen.

--targetDir / -t Gibt das Backup-Verzeichnis auf dem lokalen Medium (z.B. Stick) an, auf dem die Backups gemacht wurden.

--series / -S Falls auf dem Stick mehr als eine Backup-Serie gespeichert ist, muss hier angegeben werden, welche vom lokalen Medium ins Master-Backup kopiert werden soll.

--verbose / -v Gibt zusätzliche Meldungen aus.

--tmpdir / -T /tmpdir Verzeichnis für temporäre Dateien. Der Wert wird von der Umgebungsvariablen **\$TMPDIR** übernommen. Falls diese nicht gesetzt ist, wird **/tmp** als Standardwert genommen.

6.9 storeBackups.pl

storeBackups.pl zeigt Informationen über das Alter und die Löschregeln einer Backup-Serie an.

```
1 storeBackups.pl -f configFile [--print] [storeBackup-dir]
2 storeBackups.pl [-v] [--print] storeBackup-dir
```

Es gibt zwei mögliche Arten, **storeBackups** aufzurufen (mit Beispielen):

```
1 # /opt/test/storeBackup/bin/storeBackups.pl .
2 1 Fri Jul 04 2008 2008.07.04_20.17.13 -61
3 2 Sat Jul 05 2008 2008.07.05_21.19.09 -60
4 3 Sun Jul 06 2008 2008.07.06_17.38.22 -59
5 4 Mon Jul 07 2008 2008.07.07_17.31.43 -58
6 5 Fri Jul 11 2008 2008.07.11_19.20.14 -54
7 6 Sat Jul 12 2008 2008.07.12_18.17.21 -53
```

```

8 7 Sun Jul 13 2008 2008.07.13_17.07.53 -52
9 8 Mon Jul 14 2008 2008.07.14_06.28.29 -51
10 9 Tue Jul 15 2008 2008.07.15_07.44.41 -50
11 10 Wed Jul 16 2008 2008.07.16_17.56.35 -49
12 11 Thu Jul 17 2008 2008.07.17_10.13.47 -48
13 12 Fri Jul 18 2008 2008.07.18_14.13.26 -47
14 13 Sat Jul 19 2008 2008.07.19_16.03.40 -46
15 14 Fri Jul 25 2008 2008.07.25_09.29.39 -40
16 15 Mon Jul 28 2008 2008.07.28_19.01.04 -37
17 16 Wed Jul 30 2008 2008.07.30_17.25.43 -35
18 17 Thu Jul 31 2008 2008.07.31_16.45.56 -34
19 18 Fri Aug 01 2008 2008.08.01_16.43.56 -33
20 19 Mon Aug 04 2008 2008.08.04_17.26.42 -30
21 20 Thu Aug 07 2008 2008.08.07_16.16.21 -27
22 21 Fri Aug 08 2008 2008.08.08_20.59.46 -26
23 22 Sat Aug 09 2008 2008.08.09_20.48.31 -25
24 23 Sun Aug 10 2008 2008.08.10_14.29.18 -24
25 24 Mon Aug 11 2008 2008.08.11_19.51.32 -23
26 25 Tue Aug 12 2008 2008.08.12_14.13.02 -22
27 26 Wed Aug 13 2008 2008.08.13_20.41.43 -21
28 27 Thu Aug 14 2008 2008.08.14_16.44.02 -20
29 28 Fri Aug 15 2008 2008.08.15_19.47.29 -19
30 29 Mon Aug 18 2008 2008.08.18_18.29.06 -16
31 30 Tue Aug 19 2008 2008.08.19_17.58.42 -15
32 31 Wed Aug 20 2008 2008.08.20_18.53.46 -14
33 32 Thu Aug 21 2008 2008.08.21_19.56.03 -13
34 33 Fri Aug 22 2008 2008.08.22_23.32.10 -12
35 34 Sun Aug 24 2008 2008.08.24_12.57.36 -10
36 35 Tue Aug 26 2008 2008.08.26_10.34.06 -8 not finished
37 36 Tue Aug 26 2008 2008.08.26_10.59.46 -8
38 37 Tue Aug 26 2008 2008.08.26_13.07.08 -8

```

Wie man sieht, wurde das 35. noch vorhandene Backup nicht beendet. Mit der Option `verbose` ergibt sich:

```

1 # /opt/test/storeBackup/bin/storeBackups.pl -v .
2 . . .
3 37 Tue Aug 26 2008 2008.08.26_13.07.08 -8
4 version -> 1.3
5 date -> 2008.08.26 13.07.08
6 sourceDir -> '/backup'
7 followLinks -> 1
8 compress -> 'bzip2'
9 uncompress -> 'bzip2' '-d'
10 postfix -> '.bz2'
11 exceptSuffix -> '.bz2' '.gif' '.gpg' '.gz' '.jpg' '.mp3' '.mpeg' '.mpg' '.ogg' '.png' '.tgz'
   ↳ '.tif' '.tiff' '.zip'
12 exceptDirs -> '/backup/home_hjc/nosave'
13 includeDirs ->
14 exceptRule -> '$file =~ /acronis.*tib/' 'or' '$file =~ m#/te?mp/#' 'or' '$file =~
   ↳ m#/thumbnails/#'
15 includeRule ->
16 exceptTypes ->
17 preservePerms -> yes
18 lateLinks -> yes
19 lateCompress -> no
20 cpIsGnu -> yes

```

Hier wird nur die Ausgabe für das letzte Backup gezeigt. Man sieht, mit welchen Optionen `storeBackup.pl` aufgerufen wurde, um das Backup zu generieren.

```

storeBackups.pl -f configFile [--print] [storeBackup-dir] oder
storeBackups.pl --file configFile [--print] [storeBackup-dir]

```

Hier wird die Konfigurationsdatei von `storeBackup.pl` gelesen und der Lösch-Status für jedes Backup ausgegeben:

```

1 # storeBackups.pl -f /backup/stbu-gentoo.conf .
2 . . .
3 WARNING backup <./2008.08.26_10.34.06> not finished
4 analysis of old Backups in <.>:
5 Fri 2008.07.04_20.17.13 (61): will be deleted
6 Sat 2008.07.05_21.19.09 (60): keepWeekDays(60d)
7 Sun 2008.07.06_17.38.22 (59): keepWeekDays(60d)
8 Mon 2008.07.07_17.31.43 (58): keepWeekDays(60d)
9 Fri 2008.07.11_19.20.14 (54): keepWeekDays(60d)
10 Sat 2008.07.12_18.17.21 (53): keepMinNumber30, keepWeekDays(60d)
11 Sun 2008.07.13_17.07.53 (52): keepMinNumber29, keepWeekDays(60d)
12 Mon 2008.07.14_06.28.29 (51): keepMinNumber28, keepWeekDays(60d)
13 Tue 2008.07.15_07.44.41 (50): keepMinNumber27, keepWeekDays(60d)
14 Wed 2008.07.16_17.56.35 (49): keepMinNumber26, keepWeekDays(60d)
15 Thu 2008.07.17_10.13.47 (48): keepMinNumber25, keepWeekDays(60d)
16 Fri 2008.07.18_14.13.26 (47): keepMinNumber24, keepWeekDays(60d)
17 Sat 2008.07.19_16.03.40 (46): keepMinNumber23, keepWeekDays(60d)
18 Fri 2008.07.25_09.29.39 (40): keepMinNumber22, keepWeekDays(60d)
19 Mon 2008.07.28_19.01.04 (37): keepMinNumber21, keepWeekDays(60d)
20 Wed 2008.07.30_17.25.43 (35): keepMinNumber20, keepWeekDays(60d)
21 Thu 2008.07.31_16.45.56 (34): keepMinNumber19, keepWeekDays(60d)
22 Fri 2008.08.01_16.43.56 (33): keepMinNumber18, keepWeekDays(60d)
23 Mon 2008.08.04_17.26.42 (30): keepMinNumber17, keepWeekDays(60d)
24 Thu 2008.08.07_16.16.21 (27): keepMinNumber16, keepWeekDays(60d)
25 Fri 2008.08.08_20.59.46 (26): keepMinNumber15, keepWeekDays(60d)
26 Sat 2008.08.09_20.48.31 (25): keepMinNumber14, keepWeekDays(60d)
27 Sun 2008.08.10_14.29.18 (24): keepMinNumber13, keepWeekDays(60d)
28 Mon 2008.08.11_19.51.32 (23): keepMinNumber12, keepWeekDays(60d)
29 Tue 2008.08.12_14.13.02 (22): keepMinNumber11, keepWeekDays(60d)
30 Wed 2008.08.13_20.41.43 (21): keepMinNumber10, keepWeekDays(60d)
31 Thu 2008.08.14_16.44.02 (20): keepMinNumber9, keepWeekDays(60d)
32 Fri 2008.08.15_19.47.29 (19): keepMinNumber8, keepWeekDays(60d)
33 Mon 2008.08.18_18.29.06 (16): keepMinNumber7, keepWeekDays(60d)
34 Tue 2008.08.19_17.58.42 (15): keepMinNumber6, keepWeekDays(60d)
35 Wed 2008.08.20_18.53.46 (14): keepMinNumber5, keepWeekDays(60d)
36 Thu 2008.08.21_19.56.03 (13): keepMinNumber4, keepWeekDays(60d)
37 Fri 2008.08.22_23.32.10 (12): keepMinNumber3, keepWeekDays(60d)
38 Sun 2008.08.24_12.57.36 (10): keepMinNumber2, keepWeekDays(60d)
39 Tue 2008.08.26_10.59.46 (8): will be deleted
40 Tue 2008.08.26_13.07.08 (8): keepMinNumber1, keepWeekDays(60d)

```

... und mit Option `-print` sieht man die Parameter, die bei `storeBackup.pl` für das Löschen verwendet wurden:

```

1 # storeBackups.pl -f /backup/stbu-gentoo.conf . --print
2 combined configuration and command line options
3 options with parameters:
4 file </backup/stbu-gentoo.conf>
5 firstDayOfWeek <Sun>
6 keepAll <60d>
7 keepDuplicate <7d>
8 keepFirstOfMonth <undef>
9 keepFirstOfWeek <undef>
10 keepFirstOfYear <undef>
11 keepLastOfMonth <undef>
12 keepLastOfWeek <undef>
13 keepLastOfYear <undef>
14 keepMaxNumber <0>
15 keepMinNumber <30>
16 keepRelative <undef>
17 keepWeekday <undef>
18 options without parameters:

```

```
19 list parameters:
20 <.>
```

6.10 storeBackupDel.pl

`storeBackupDel.pl` löscht alte Backups nach den in der Konfigurationsdatei definierten Regeln. Diese Regeln (keep*) werden als Optionen von `storeBackup.pl` in Kapitel 6.2 beschrieben. Wenn Du diese Option verwenden willst, solltest Du `storeBackup.pl` über eine Konfigurationsdatei konfigurieren, die dann von `storeBackupDel.pl` gelesen wird. Falls Du alte Sicherungen asynchron löschen willst, empfiehlt es sich ebenfalls, Beispiel 6, Kapitel 9.7 zu lesen.

```
1 $prog [-f configFile] [--print]
2 [-b backupDirectory] [-S series] [--doNotDelete]
3 [--deleteNotFinishedDirs] [-L lockFile]
4 [--keepAll timePeriod] [--keepWeekday entry] [--keepFirstOfYear]
5 [--keepLastOfYear] [--keepFirstOfMonth] [--keepLastOfMonth]
6 [--keepFirstOfWeek] [--keepLastOfWeek]
7 [--keepDuplicate] [--keepMinNumber] [--keepMaxNumber]
8 [-l logFile]
9 [--plusLogStdout] [--suppressTime] [-m maxFilelen]
10 [[-n noOfOldFiles] | [--saveLogs]
11 [--compressWith compressprog]]
```

Es müssen mindestens zwei Optionen gesetzt werden: `--backupDir` und `--series` (wobei letzteres den Default Wert `default` hat). Es spielt keine Rolle, ob sie in der Konfigurationsdatei, auf der Kommandozeile oder gemischt gesetzt werden. Die restlichen Optionen sind wie bei `storeBackup.pl`.

Zunächst einmal die Optionen, die nur auf der Kommandozeile gesetzt werden können. Es gibt immer eine lange Option (wie `--configFile` und teilweise auch ein Kürzel (`-f`)).

`--print` Gibt die gewählten Optionen (von der Konfigurationsdatei und von der Kommandozeile) aus und stoppt das Programm. Im Falle von kompliziertem Quoting (insbesondere auf der Kommandozeile) gibt Dir das die Möglichkeit zu sehen, was wirklich vom Programm ausgewertet wird.

`--configFile / -f` Name der zu verwendenden Konfigurationsdatei.

Die Optionen in der Konfigurationsdatei können einfach überschrieben werden (insbesondere zum Ändern von `backupDir` und um das Setzen von `doNotDelete` rückgängig zu machen. Siehe auch Kapitel 7.1.

Die folgenden Optionen sind identisch zu denen in `storeBackup.pl`:

- `backupDir`
- `series`
- `lockFile`
- `doNotDelete`
- `deleteNotFinishedDirs`
- `keepAll`
- `keepWeekday`
- `keepFirstOfYear`
- `keepLastOfYear`
- `keepFirstOfMonth`
- `keepLastOfMonth`
- `firstDayOfWeek`

- keepFirstOfWeek
- keepLastOfWeek
- keepDuplicate
- keepMinNumber
- keepMaxNumber
- keepRelative
- logFile
- plusLogStdout
- suppressTime
- maxFilelen
- noOfOldFiles
- saveLogs
- compressWith

6.11 storeBackupMount.pl

`storeBackupMount.pl` stellt Dir ein „Skript“ zur Verfügung, um die für das Backup benötigten Verzeichnisse zu mounten, `storeBackup.pl` zu starten und die Verzeichnisse wieder zu entmounten. Vor dem Mount-Versuch kann es mittels `ping` überprüfen, ob der betroffene Server erreichbar ist. Wenn die Verzeichnisse bereits (oder immer) gemountet sind, besteht für die Verwendung von `storeBackupMount.pl` kein Grund. Es kann auch zum Starten anderer Programme wie `storeBackupUpdateBackup.pl` oder `storeBackupDel.pl` verwendet werden.

Es gibt unterschiedliche Möglichkeiten, `storeBackupMount.pl` zu benutzen:

```

1 storeBackupMount.pl --help
2 oder
3 storeBackupMount.pl -g configFile
4 oder
5 storeBackupMount.pl -f configFile
6 oder
7 storeBackupMount.pl [-s servers] [-d] [-l logFile
8 [--suppressTime] [-m maxFilelen]
9 [[-n noOfOldFiles] | [--saveLogs]]
10 [--compressWith compressprog]]
11 [--storeBackup storeBackup-Params]
12 [--storeBackupUpdateBackup storeBackupUpdateBackup-Params]
13 [--storeBackupCheckBackup storeBackupCheckBackup-Params]
14 [--storeBackupCheckSource storeBackupCheckSource-Params]
15 [--storeBackupDel storeBackupDel-Params]
16 [--printAndStop] [-k killTime] [-T tmpdir] [mountPoints...]
```

Du kannst eine Konfigurationsdatei erzeugen (Option `-g`), diese verwenden (Option `-f`) oder Deine Einstellungen über die Kommandozeile vornehmen.

Um die Verzeichnisse mounten zu können, muss ein Eintrag in wie der Folgende in `/etc/fstab` existieren:

```

1 /dev/sda5          /add          reiserfs      noatime       0 1
2 lotte:/disk1 /backup nfs    rsize=8192,wsiz=8192,user,exec,async,noatime 1 1
```

Der erste Mount-Punkt `/add` betrifft ein lokales Device. In diesem Beispiel ist das das Dateisystem, welches gesichert werden soll. Der zweite (`/backup`) liegt auf dem NFS Server unter `/disk1`. Der Rest sind NFS Parameter – siehe Kapitel 7.11 über die Konfiguration von NFS.

Wenn Einträge wie diese in `/etc/fstab` sind, kann man das Dateisystem mit:

```
mount /add
```

`mount /backup`

mounten, und das ist exakt das, was `storeBackupMount.pl` tut.

Es gibt einen klaren Unterschied bei der Verwendung der Kommandozeile oder der Konfigurationsdatei: Da mehrere Programme gestartet werden können, muss auch die Reihenfolge des Startens festgelegt werden. Dieses erfolgt:

- auf der Kommandozeile durch Verwendung von Optionen wie `--storeBackup` oder `--storeBackupDel` in der gewünschten Startreihenfolge
- in der Konfigurationsdatei durch Auflisten der zu startenden Programme als Parameter beim Schlüsselwort `orderOfExecution`.

`storeBackupMount.pl` beendet die Ausführung, sobald eines der Programme nicht erfolgreich lief.

Du kannst folgende Optionen verwenden:

`--help` Ausgabe aller Kommandozeilen-Optionen.

`--printAndStop` Gibt die Optionen aus und beendet das Programm direkt.

`--generate / -g` Dateiname für die zu erzeugende Konfigurationsdatei festlegen. Diese Option ist nur für die Kommandozeile verfügbar.

`--file / -f` Dateiname der zu verwendenden Konfigurationsdatei festlegen. Diese Option ist nur für die Kommandozeile verfügbar.

`--servers / -s / servers` Liste der Server (Name oder IP Adresse), deren Erreichbarkeit mittels `ping` überprüft wird. Diese Option kann auf der Kommandozeile wiederholt verwendet werden.

`--debug / -d` Erzeuge zusätzliche Meldungen.

`--logFile / -l` Logdatei für diesen Prozess, der Standardwert ist `stdout`.

`--suppressTime / suppressTime` Unterdrücke die Ausgabe der aktuellen Zeit in der Logdatei.

`--maxFilelen / -m / maxFilelen` Maximale Größe einer Logdatei. Nachdem diese Größe erreicht wird, wird die Logdatei rotiert (siehe Option `noOfOldFiles`) oder komprimiert (siehe Option `saveLogs`).

`--noOfOldFiles / -n / noOfOldFiles` Anzahl alter Logdateien, die rotiert werden sollen. Der Standardwert ist 5. Mit den Standardwerten sieht das so aus:

```
1 $ ls -l /tmp/storebackup.log*
2 -rw----- 1 hjc  root  328815 30. Aug 12:12 /tmp/storebackup.log
3 -rw----- 1 root  root  1000087 27. Aug 21:18 /tmp/storebackup.log.1
4 -rw----- 1 root  root  1000038 20. Aug 19:02 /tmp/storebackup.log.2
5 -rw----- 1 root  root  1000094 11. Aug 18:51 /tmp/storebackup.log.3
6 -rw----- 1 root  root  1000147 11. Aug 18:49 /tmp/storebackup.log.4
7 -rw----- 1 root  root  1000030 11. Aug 18:49 /tmp/storebackup.log.5
```

Logdateien, die älter als *.5 sind, werden automatisch gelöscht.

`--saveLogs / saveLogs` Speichere die Logs mit Datums- und Zeitstempel, statt sie nach dem Rotieren zu löschen. Durch Setzen dieser Option wird die Option `noOfOldFiles` deaktiviert.

`--compressWith / compressWith` Spezifiziert das Programm, mit dem die zu sichernden Logdateien gesichert werden sollen (z. B. `gzip -9`). Der Standardwert ist `bzip2`.

Die Parameter für diese Option werden auf der Kommandozeile so ausgewertet wie in der Konfigurationsdatei. Das bedeutet, dass sie auf der Kommandozeile gequotet werden müssen.

`--storeBackup / storeBackup` Lege hiermit fest, dass `storeBackup.pl` gestartet wird und spezifiziere die Optionen / Parameter dafür. Um diesen *einen* Parameter auf mehrere für `storeBackup.pl` abzubilden, werden die Anführungszeichen der Kommandozeile entfernt und die Parameter zu dieser Option werden nochmals ausgewertet.

Beispiel: Wenn Du `'-f /backup/stbu.conf'` auf der Kommandozeile setzt, dann wird `storeBackup.pl`

mit den beiden Parametern `-f` und `/backup/stbu.conf` aufgerufen.

Wenn Du die Konfigurationsdatei verwendest, setz einfach:

```
storeBackup = -f /backup/stbu.conf
```

um dasselbe Resultat zu erreichen. Wenn Dir das alles nicht wirklich klar ist, solltest Du Kapitel 7.1 lesen.

`--storeBackupUpdateBackup / storeBackupUpdateBackup` Bestimme, dass `storeBackupUpdateBackup.pl` aufgerufen wird und setze seine Parameter. Siehe Option `--storeBackup` für Details.

`--storeBackupCheckBackup / storeBackupCheckBackup` Bestimme, dass `storeBackupCheckBackup.pl` aufgerufen wird und setze seine Parameter. Siehe Option `--storeBackup` für Details.

`--storeBackupCheckSource / storeBackupCheckSource` Bestimme, dass `storeBackupCheckSource.pl` aufgerufen wird und setze seine Parameter. Siehe Option `--storeBackup` für Details.

`--storeBackupDel / storeBackupDel` Bestimme, dass `storeBackupDel.pl` aufgerufen wird und setze seine Parameter. Siehe Option `--storeBackup` für Details.

`--killTime / -k / killTime` Zeit, bis die gestarteten Programme abgeschossen werden, d.h. jedes individuelle Programm kann maximal `killTime` laufen. Das Format dieser Option ist `dhms`, dabei bedeutet z.B. `10d4h` 10 Tage und 4 Stunden. Siehe auch die Erläuterungen zu den `keep*` Optionen von `storeBackup.pl` für weitere Beispiele. Der Defaultwert ist 365 Tage.

`--tmpdir / -T /tmpdir` Verzeichnis für temporäre Dateien, der Wert wird von der Umgebungsvariablen `$TMPDIR` übernommen. Falls diese nicht gesetzt ist, wird `/tmp` als Standardwert genommen.

`...mountPoints...` / `mountPoints` Auf der Kommandozeile ist dies keine Option, sondern ein List-Parameter. So muss auf der Kommandozeile geschrieben werden:

```
1 # storeBackupMount.pl <all_options> /backupDisk /otherDisk
```

In der Konfigurationsdatei entspräche das:

```
mountPoints = /backupDisk /otherDisk
```

Hier gibst Du die Liste der Mount-Punkte an, die für die Durchführung des Backups benötigt werden. Dieses muss eine Liste von Pfaden sein, die in `/etc/fstab` definiert wurden. Falls Du `ro` oder `rw` zum Anfang des Mount-Punkte hinzufügst, kannst Du damit diese Option in der `/etc/fstab` überschreiben. Beispiel:

```
ro,/fileSystemToRead
```

`mountet /fileSystemToRead read only` (nur lesen), auch wenn in dem entsprechenden Eintrag in der `/etc/fstab` `read/write` (lesen und schreiben) angegeben ist. Diese Möglichkeit kann aber nur von root verwendet werden!

6.12 storeBackupCheckBackup.pl

Wenn Du Daten von einer Platte auf eine andere kopierst, können unbemerkt Bit-Fehler (z.B. im Hauptspeicher, in der CPU, in Verbindungsleitungen) passieren, die dazu führen, dass fehlerhafte Daten in Deinem Backup gespeichert werden. Dieses kann insbesondere über lange Zeiträume auf der Backup-Platte passieren. Da `storeBackup` neben den eigentlichen Daten auch Prüfsummen speichert, ist es möglich, die gespeicherten Daten mit diesen Prüfsummen, die aus den Daten im zu sichernden Verzeichnis berechnet wurden, zu vergleichen, um so die Wahrscheinlichkeit zu erhöhen, dass die Daten nicht korrupt sind.²⁰ Aber keine Panik, so etwas passiert nicht ständig.

`storeBackupCheckBackup.pl` überprüft die Konsistenz eines oder mehrerer Backups, indem es die während des Backups erzeugten md5-Summen in `.md5CheckSums` (siehe Kapitel 7.9) mit den neu errechneten aus den Backupdateien vergleicht. Es überprüft ebenfalls alle Dateien im Backup und in `.md5CheckSums` auf Existenz oder nicht-Existenz.

²⁰Dies ist keine 100%tige Sicherheit. Es ist immer noch möglich, dass sowohl die Prüfsummen als auch die Daten z.B. schon beim Lesen durch einen Hauptspeicherfehler falsch sind. In diesem Fall wird `storeBackupCheckSource.pl`, Kapitel 6.13 zwar die Sicherheit erhöhen. Aber generell ist der einzige Schutz gegen derartige Vorfälle vollständig redundante Hardware, die *sehr* teuer ist.

```

1 storeBackupCheckBackup.pl -c backupDir [-p number] [-i]
2 [-w filePrefix] [--lastOfEachSeries] [--wrongFileTables]
3 [--includeRenamedBackups] [-T tmpdir]
4 [--logFile
5 [--plusLogStdout] [--suppressTime] [-m maxFilelen]
6 [--noOfOldFiles] | [--saveLogs]]

```

--print Gibt die Konfigurationsparameter aus und beendet das Programm.

--checkDir / -c Das Backupverzeichnis, in dem gesucht werden soll. Diese Option kann auf das gesamte Backup-Repository, auf eine Serie oder ein einzelnes Backupverzeichnis gesetzt werden.

--wrongFileTables / -w Schreibe Dateinamen, bei denen Fehler gefunden wurden, aufgelistet in einfache, zeilenorientierte Dateien, um diese später (nicht automatisiert) weiterzuverarbeiten. Der Parameter zu dieser Option ist ein Präfix für die zu erzeugenden Dateien. Wenn der Präfix z. B. als /tmp/bugsB- angegeben wird, werden folgende Dateien generiert:

```

/tmp/bugsB-files.missing.txt
/tmp/bugsB-md5sums.missing.txt
/tmp/bugsB-md5sums.wrong.txt

```

--lastOfEachSeries Überprüfe nur das letzte Backup einer jeden gefundenen Serie.

--verbose / -v Erzeuge zusätzliche Meldungen.

--parJobs / -p Maximale Anzahl von parallelen Suchoperationen. Der Standardwert wird bei GNU/Linux automatisch als Anzahl der Cores + 1 gewählt.

--includeRenamedBackups / -i Mit dieser Option können auch umbenannte Backups mit der Option --checkDir berücksichtigt werden. Umbenannte Optionen müssen hier allerdings zwingend dem Schema backupDirName-something entsprechen, z. B. 2012.01.30_15.21.03-renamed. Siehe auch Kapitel 7.3.

--tmpdir / -T /tmpdir Verzeichnis für temporäre Dateien, der Wert wird von der Umgebungsvariablen \$TMPDIR übernommen. Falls diese nicht gesetzt ist, wird /tmp als Standardwert genommen.

--logFile / -l Name der Logdatei. Standard ist die Ausgabe auf stdout.

--plusLogStdout / plusLogStdout Wenn die Option logFile gesetzt ist, kann hier konfiguriert werden, dass storeBackup.pl das Log zusätzlich auf stdout ausgibt.

--suppressTime Unterdrücke die Ausgabe der aktuellen Zeit in der Logdatei.

--maxFilelen / -m Maximale Größe einer Logdatei. Nachdem diese Größe erreicht wird, wird das Log file rotiert (siehe Option noOfOldFiles) oder komprimiert (siehe Option saveLogs).

--noOfOldFiles / -n Anzahl alter Logdateien, die rotiert werden sollen. Standardwert ist 5. Mit den Standardwerten sieht das so aus:

```

1 $ ls -l /tmp/storebackup.log*
2 -rw----- 1 hjc root 328815 30. Aug 12:12 /tmp/storebackup.log
3 -rw----- 1 root root 1000087 27. Aug 21:18 /tmp/storebackup.log.1
4 -rw----- 1 root root 1000038 20. Aug 19:02 /tmp/storebackup.log.2
5 -rw----- 1 root root 1000094 11. Aug 18:51 /tmp/storebackup.log.3
6 -rw----- 1 root root 1000147 11. Aug 18:49 /tmp/storebackup.log.4
7 -rw----- 1 root root 1000030 11. Aug 18:49 /tmp/storebackup.log.5

```

Logdateien, die älter als *.5 sind, werden automatisch gelöscht.

--saveLogs Speichere die Logs mit Datums- und Zeitstempel, statt sie nach dem Rotieren zu löschen. Durch Setzen dieser Option wird die Option noOfOldFiles deaktiviert.

--compressWith Spezifiziert das Programm, mit dem die zu sichernden Logdateien gesichert werden sollen (z. B. gzip -9). Der Standardwert ist bzip2.

Parameter für diese Option werden auf der Kommandozeile so ausgewertet wie in der Konfigurationsdatei. Das bedeutet, dass sie auf der Kommandozeile gequotet werden müssen.

6.13 storeBackupCheckSource.pl

Wenn Du Daten von einer Platte auf eine andere kopierst, können unbemerkt Bit-Fehler (z. B. im Hauptspeicher, in der CPU, in Verbindungsleitungen) passieren, die dazu führen, dass fehlerhafte Daten in Deinem Backup gespeichert werden. Dieses kann insbesondere über lange Zeiträume auf der Backup-Platte passieren. Da storeBackup neben den eigentlichen Daten auch Prüfsummen speichert, ist es möglich, die gespeicherten Daten mit diesen Prüfsummen, die aus den Daten im zu sichernden Verzeichnis berechnet wurden, zu vergleichen, um so die Wahrscheinlichkeit zu erhöhen, dass die Daten nicht korrupt sind. Das ist der Grund, warum es storeBackupCheckBackup.pl gibt.

Ein solcher Bit-Fehler kann aber auch in Deinem Quellverzeichnis (also dem, das gesichert wird) nach einem längeren Zeitraum passieren. Mit diesem Programm kannst Du Dateien, die sich seit dem Backup im Quellverzeichnis nicht geändert haben, mit Hilfe der Prüfsummen aus dem Backup überprüfen.

```
1 storeBackupCheckSource.pl -s sourceDir -b singleBackupDir [-v]
2 [-w filePrefix]
3 [--logFile
4 [--plusLogStdout] [--suppressTime] [-m maxFilelen]
5 [[-n noOfOldFiles] | [--saveLogs]]
```

--sourceDir / -s Hier muss das Verzeichnis angegeben werden, das bei Erzeugung des Backups als Parameter für die Option **sourceDir** bei **storeBackup.pl** angegeben wurde.

--singleBackupDir / -b Das Backup-Verzeichnis, mit dem verglichen werden soll. Dieses muss *ein direktes* Backup-Directory sein.

--wrongFileTables / -w Bei Verwendung dieser Option schreibt **storeBackupCheckSource.pl** zwei Dateien, deren Präfix der Definition dieser Option entspricht. Wenn Du sie z. B. auf **/tmp/faults-** setzt, dann werden die Dateien **/tmp/faults-md5sums.missing.txt** und **/tmp/faults-md5sums.wrong.txt** geschrieben. Sie enthalten zeilenweise die Namen der Dateien, bei denen die md5-Summe fehlt oder falsch (unterschiedlich²¹ zum **sourceDir**) im Backup steht. Diese Dateien werden zusätzlich zu den error-Meldungen geschrieben und können dazu verwendet werden, weitere Untersuchungen vorzunehmen oder das Backup zu reparieren.

In den geschriebenen Dateien wird ein \ (Backslash) mit den Zeichen \5C und ein \n (Zeilenvorschub) mit den Zeichen \0A ausgegeben.

--verbose / -v Verbose. Gibt zusätzliche Informationen aus:

- immer ausgegeben: **ERROR** Meldung, wenn die MD5 Summe unterschiedlich ist
- immer ausgegeben: **WARNING** wenn Rechte, UID oder GID unterschiedlich sind
- nur ausgegeben, wenn **-v** verwendet: **MISSING** wenn die Datei nicht im **sourceDir** ist
- nur ausgegeben, wenn **-v** verwendet: **INFO** wenn die Datei identisch ist

--logFile / -l Name der Logdatei. Standard ist die Ausgabe auf stdout.

--plusLogStdout / plusLogStdout Wenn die Option **logFile** gesetzt ist, kann hier konfiguriert werden, dass **storeBackup.pl** das Log zusätzlich auf stdout ausgibt.

--suppressTime Unterdrücke die Ausgabe der aktuellen Zeit in der Logdatei.

--maxFilelen / -m Maximale Größe einer Logdatei. Nachdem diese Größe erreicht wird, wird das Log file rotiert (siehe Option **noOfOldFiles**) oder komprimiert (siehe Option **saveLogs**).

--noOfOldFiles / -n Anzahl alter Logdateien, die rotiert werden sollen. Standardwert ist 5. Mit den Standardwerten sieht das so aus:

```
1 $ ls -l /tmp/storebackup.log*
2 -rw----- 1 hjc  root  328815 30. Aug 12:12 /tmp/storebackup.log
3 -rw----- 1 root  root 1000087 27. Aug 21:18 /tmp/storebackup.log.1
```

²¹Dies kann bedeuten, dass die in den Metadaten des Backups gespeicherte md5-Summe falsch ist oder dass auf einer Platte Bits „umgekippt“ sind. (Natürlich kann es auch bedeuten, dass es andere Gründe für diesen Fehler gibt, z. B. Fehler im RAM deines Rechners oder ein Fehler von storeBackup.)

```

4  -rw----- 1 root root 1000038 20. Aug 19:02 /tmp/storebackup.log.2
5  -rw----- 1 root root 1000094 11. Aug 18:51 /tmp/storebackup.log.3
6  -rw----- 1 root root 1000147 11. Aug 18:49 /tmp/storebackup.log.4
7  -rw----- 1 root root 1000030 11. Aug 18:49 /tmp/storebackup.log.5

```

Logdateien, die älter als *.5 sind, werden automatisch gelöscht.

--saveLogs Speichere die Logs mit Datums- und Zeitstempel, statt sie nach dem Rotieren zu löschen. Durch Setzen dieser Option wird die Option **noOfOldFiles** deaktiviert.

--compressWith Spezifiziert das Programm, mit dem die zu sichernden Logdateien gesichert werden sollen (z. B. **gzip -9**). Der Standardwert ist **bzip2**.

Parameter für diese Option werden auf der Kommandozeile so ausgewertet wie in der Konfigurationsdatei. Das bedeutet, dass sie auf der Kommandozeile gequotet werden müssen.

6.14 storeBackup_du.pl

storeBackup_du.pl untersucht die Festplattenbelegung in einem oder mehreren Backup-Verzeichnissen. **sumLocal** gibt dabei die in Backups lokal liegende Datenmenge und **sumShared** die Datenmenge, die mit anderen Backups über Hardlinks geteilt wird.

```
1 storeBackup_du.pl [-v] [-l] backupdirs ...
```

--verbose / -v Zeigt die akkumulierten Werte für mehrere Versionen von gesicherten Dateien.

--links / -l Gibt aus, wie viele Hardlinks die Dateien haben und wie viel Plattenplatz dadurch jeweils gespart wird.

...backupdirs... die zu untersuchenden Backupverzeichnisse

6.15 storeBackupConvertBackup.pl

Du solltest dieses Programm nur aufrufen, wenn **storeBackup.pl** darauf hinweist.

Konvertiert alte Backups, die mit älteren Versionen von **storeBackup.pl** erzeugt wurden, auf die neueste Version.

Die aktuelle Version des Backup-Formats ist 1.3

Die Version kann mit folgendem Kommando angezeigt werden:

```
1 head -1 < ...<storeBackupDir>/date_time/.md5CheckSums.info
```

Rufe **storeBackupConvertBackup.pl** mit den Backupverzeichnissen auf, die zu konvertieren sind:

```
1 storeBackupConvertBackup.pl storeBackup-dir
```

6.16 linkToDirs.pl

Erzeugt eine deduplizierte Kopie von Dateien in definierten Verzeichnisse an eine andere Stelle. Verwendet so viele Hardlinks wie möglich, um Speicherplatz zu sparen.

linkToDirs.pl ist ein allgemein verwendbares Tool. Es ist aber besonders hilfreich, wenn Du mit **storeBackup** erzeugte Backups auf ein anderes Dateisystem kopieren willst.

Anmerkung: Während viele Kopiertools genau zwei primäre Parameter haben (Ziel und Quelle), hat **linkToDirs.pl** drei:

- Quelle(n)
- Ziel
- und Ort(e) zum Referenzieren

Die Stelle zum Referenzieren ist die Lokation, an der nach vorhandenem Inhalt gesucht wird, mit dem verhardlinkt werden kann (siehe Option **--linkWith**).

Die Verwendung der Option **--linkWith** ist nicht zwingend. Wenn Du sie verwendest, kannst Du optional mehrere Verzeichnisse zum Referenzieren über Hardlinks angeben (die Option **--linkWith** kann wiederholt angewendet werden).

Dateien mit demselben Inhalt wie dem in den Referenzverzeichnissen (diese müssen auf demselben Dateisystem wie das Zielverzeichnis liegen) werden verhardlinkt. Hardlinks in den kopierten Dateien bleiben erhalten oder werden wieder neu erzeugt: `linkToDirs.pl` setzt bei identischen Dateien immer Hardlinks, außer in den Dateien innerhalb der Referenzverzeichnisse, die mittels `--linkWith` angegeben wurden. Diese werden nicht verändert. Falls sich dort also zwei identische Dateien befinden, die nicht verhardlinkt sind, bleiben sie so (ohne Hardlink) bestehen. `linkToDirs.pl` unterstützt das Hardlinken von symbolischen Links.

(Wenn keine identischen Dateien existieren, werden natürlich nur Kopien angelegt.)

Hardlinks folgen bei Linux diesen Regeln:

- Hardlinks können keine Directories verlinken.
- Hardlinks können keine Dateisystemgrenzen überschreiten.

Wenn es nicht möglich ist, einen Hardlink auf die referenzierte Datei zu erzeugen (z. B. wegen der genannten Limitationen von Hardlinks), kopiert `linkToDirs.pl` die betreffende Datei auf dem Ziel-Dateisystem neu und verlinkt danach gegen diese. Auf diese Weise kann `linkToDirs.pl` verwendet werden, einen deduplizierten Status aus dem Quellverzeichnis auf einem Zielverzeichnis zu erhalten (auch wenn das Ziel-Dateisystem weniger Hardlinks pro Datei unterstützt als das Quell-Dateisystem). Hardlinks können aber nicht über Dateisystemgrenzen angelegt werden.

`linkToDirs.pl` ist ein Tool zur allgemeinen Verwendung. Nichtsdestotrotz hat es einen besonderen Nutzen für `storeBackup`. Wie Du weißt, eliminiert `storeBackup` Platzverschwendung im Backup durch Deduplikation über Hardlinks. Aber Hardlinks können nicht über Dateisystem Grenzen angelegt werden.

Wenn Du ein existierendes, mit `storeBackup` erzeugtes Backup auf eine neue Platte (ein neues Dateisystem) kopieren willst, ermöglicht es `linkToDirs.pl` Dir, alle Vorteile des Platzsparens von `storeBackup` zu erhalten.

```
1 linkToDirs.pl [--linkWith copyBackupDir] [--linkWith ...]
2 --targetDir targetForSourceDir
3 [--progressReport number[,timeframe]]
4 [--printDepth] [--dontLinkSymlinks]
5 [--ignoreErrors] [--saveRAM] [-T tmpdir]
6 [--createSparseFiles [--blockSize]]
7 [--maxHardLinks]
8 sourceDir ...
```

`--help / -h` Gibt eine Hilfe aus

`--linkWith / -w` Das Referenzverzeichnis. Die Dateien hier werden beim Hardlinken berücksichtigt. Diese Option kann wiederholt verwendet werden. (Wie zu erwarten werden die Verzeichnisse rekursiv durchlaufen.)

`--targetDir / -t` Das Zielverzeichnis. Die bei `sourceDir` angegebenen Verzeichnisse werden hierher kopiert.

`--dontLinkSymlinks` Symbolische Links werden nicht per Hardlinks verbunden. Standard ist, identische symbolische Links per Hardlink zu verbinden.

`--progressReport / -P / progressReport` Schreibe den Fortschritt nach der hier angegebenen Anzahl von Dateien in die Logdatei. Wenn Du eine Meldung spätestens nach einem bestimmten Zeitraum haben willst, kann Du diese durch ein Komma separiert anfügen, z. B.:

`-P 1000,1m10s` auf der Kommandozeile, oder

`progressReport = 1000,1m10s` in der Konfigurationsdatei.

Es dürfen keine Leerstellen im Parameter zu der Option enthalten sein. Die Syntax für den Zeitraum ist dieselbe wie bei den `keep*` Optionen.

`sourceDir` Das Quellverzeichnis. Dateien (oder existierende `storeBackup`-Backups) aus diesem Verzeichnis werden nach `targetDir` kopiert. `sourceDir` kann wiederholt mit unterschiedlichen Verzeichnissen angegeben werden. Es können auch Wildcards verwendet werden. Die Kopierfunktionalität geht rekursiv in alle Unterverzeichnisse der angegebenen `sourceDir`.

`--ignoreErrors` Das Programm stoppt nicht, wenn Fehler beim Kopieren / Linken auftreten.

- saveRAM / saveRAM** Verwende diese Option, wenn `storeBackup.pl` auf einem System mit sehr wenig Hauptspeicher verwendet wird. Du wirst dann in `tmpDir` einige dbm-Dateien sehen. Durch Verwendung dieser Option wird `storeBackup.pl` ein bisschen langsamer, daher empfiehlt sich diese Option nur dann, wenn ohne sie Probleme auftreten. Auf aktuellen Computern sollte die Verwendung dieser Option nur notwendig sein, wenn viele Millionen Dateien zu kopieren sind.
- tmpdir / -T /tmpdir** Verzeichnis für temporäre Dateien, der Wert wird von der Umgebungsvariablen `$TMPDIR` übernommen. Falls diese nicht gesetzt ist, wird `/tmp` als Standardwert genommen.
- createSparseFiles / -s** Unterschiede in der Anzahl der belegten Blöcke, der Blockgröße und der Dateigröße werden dazu verwendet, um mögliche Sparse Dateien zu erkennen. Wenn diese Option gesetzt ist, wird im Fall einer möglichen Sparse Datei mit dem externen Program `cp` kopiert. Auf Linux und vielen anderen Systemen ist `gnucp` installiert, welches Sparse Dateien unterstützt. Diese Option könnte auf anderen Betriebssystemen je nach verwendetem `cp` nicht funktionieren.
- blockSize** Die Blockgröße, die verwendet wird, um eine Sparse Datei zu identifizieren. Der Defaultwert ist 512.
- maxHardLinks** Gibt die maximale Anzahl von Hardlinks auf eine Datei vor, die von diesem Programm gesetzt wird. Der Default-Wert ist „0“, was bedeutet, dass die maximale Anzahl der des verwendeten Dateisystems entspricht. Die Verwendung dieser Option macht Sinn, wenn ein Verzeichnis (z.B. mit `cp -a`) auf ein anderes Dateisystem, was weniger Hardlinks als das Originale unterstützt, kopiert werden soll.
Die Option begrenzt nur die neu erzeugten Hardlinks – wenn die Anzahl schon über dem Limit ist, wird sie nicht reduziert!

6.17 llt

Anzeige von Änderungs-, Zugriffs- und Modifikationszeiten einer Datei

- ```
1 llt [-r] [-i] [-a|-m|-c] [files] [dirs]
```
- help / -h** Gibt eine Hilfmeldung aus
  - reverse / -r** Sortiere n umgekehrter Reihenfolge
  - insensitive / -i** Sortiere, ohne Groß- und Kleinschrift zu beachten
  - access / -a** Sortiere nach Zugriffszeit
  - modification / -m** Sortiere nach Modifikationszeit
  - change / -c** Sortiere nach Änderungszeit
  - unixTime / -u** Zeige Unix-Zeit (unsigned integer)

## 6.18 multiTail.pl

`multiTail.pl` liest eine oder mehrere Logdateien. Diese können auf den Bildschirm ausgegeben werden oder in eine andere Logdatei gespeichert werden. Auf diese Art können mehrere Logdateien „gemischt“ werden.

Es ist sehr robust – es stört sich nicht daran, wenn eine Datei gelöscht, verschoben oder neu erzeugt wird. Man kann es auch mit Dateien, die noch nicht existieren, starten.

- ```
1 multiTail.pl [-a] [-d delay] [-p begin|end]
2 [--print] [-t] [-o outFile [-m max] [-P]
3 [[-n noFiles] | [-s [-c compressprog]] ]
4 ]
5 [-C color=pattern [-C color=pattern ...]]
6 [-g expression] files...
```

Alle Optionen sind optional, Du kannst das Programm einfach mit dem Namen eines oder mehrerer Logdateien als Parameter starten.

--addName / -a Füge den Dateinamen am Anfang einer jeden Zeile, die eingelesen wird, hinzu.

--delay / -d Intervall in Sekunden, in dem jede Datei auf neue Daten überprüft werden soll. Dieser Wert kann kleiner als 1 sein, z. B. 0.2. Der Standardwert ist 5 (Sekunden).

--position / -p Lies die Dateien bei Start des Programm von Anfang an oder vom Ende. Erlaubte Parameter sind **begin** oder **end**. Standard ist **begin**.

--print Gibt die verwendeten Optionen aus (von der Kommandozeile und aus der Konfigurationsdatei) und terminiert danach. Im Fall von komplizierten Maskierungen (insbesondere auf der Kommandozeile) gibt diese Option die Möglichkeit zu sehen, was wirklich an das Programm übergeben wird.

--withTime -t Füge einen Zeitstempel zur Ausgabe hinzu.

--out / -o Schreibe die Ausgabe in die hier angegebene Datei, Standard ist stdout.

--maxFilelen / -m / maxFilelen Maximale Größe einer Logdatei. Nachdem diese Größe erreicht wird, wird die Logdatei rotiert (siehe Option **noOfOldFiles**) oder komprimiert (siehe Option **saveLogs**).

--withPID / -P Schreibe die PID (Process ID) von **multiTail.pl**; Standard ist, dies nicht zu tun.

--maxlines / -l Maximale Zeilenanzahl, die auf einmal von einer Logdatei gelesen wird. Standard ist 100. Wenn Du z. B. **--delay 3** konfigurierst, dann werden alle 3 Sekunden maximal 100 Zeilen gelesen. Der Grund für diese Einschränkung ist die Begrenzung der Last durch **multiTail.pl**, falls extrem viel in die Logdatei geschrieben wird.

--noOfOldFiles / -n / noOfOldFiles Anzahl alter Logdateien, die rotiert werden sollen. Standardwert ist 5. Mit den Standardwerten sieht das so aus:

```

1 $ ls -l /tmp/storebackup.log*
2 -rw----- 1 hjc  root  328815 30. Aug 12:12 /tmp/storebackup.log
3 -rw----- 1 root root 1000087 27. Aug 21:18 /tmp/storebackup.log.1
4 -rw----- 1 root root 1000038 20. Aug 19:02 /tmp/storebackup.log.2
5 -rw----- 1 root root 1000094 11. Aug 18:51 /tmp/storebackup.log.3
6 -rw----- 1 root root 1000147 11. Aug 18:49 /tmp/storebackup.log.4
7 -rw----- 1 root root 1000030 11. Aug 18:49 /tmp/storebackup.log.5

```

Log Files alter als *.5 werden automatisch gelöscht.

--saveLogs / saveLogs Speichere die Logs mit Datums- und Zeitstempel, statt sie nach dem Rotieren zu löschen. Durch Setzen dieser Option wird die Option **noOfOldFiles** deaktiviert.

--compressWith / compressWith Spezifiziert das Programm, mit dem die zu sichernden Logdateien gesichert werden sollen (z. B. **gzip -9**). Der Standardwert ist **bzip2**. Parameter für diese Option werden auf der Kommandozeile so ausgewertet wie in der Konfigurationsdatei. Das bedeutet, dass sie auf der Kommandozeile gequotet werden müssen.

--color / -C Filter, um Zeilen in einer bestimmten Farbe auszugeben, z. B. bedeutet **red=ERROR**, dass die *gesamte Zeile* mit **ERROR** in rot ausgegeben wird. Du kannst für den Teil rechts von „=" Pattern Matching verwenden. Die unterstützten Farben sind **red**, **green**, **yellow**, **blue**, **magenta** und **cyan**.

--grep / -g Das hier angegebene Pattern wird verwendet, um hier matchende Zeilen aus den Log-Files zu filtern. Für die Ausgabe wird keine Farbe gesetzt. Wenn diese Option nicht angegeben wird, werden alle Zeilen ausgegeben.

BEISPIEL:

```

1 multiTail.pl -p end -a -d 0.5 -C red=ERROR -C yellow=WARNING -g 'BEGIN|END|ERROR|WARNING' *.log

```

Hier werden nur Zeilen, die **BEGIN**, **END**, **ERROR** oder **WARNING** enthalten, ausgegeben. Zeilen mit **WARNING** werden in gelb, Zeilen mit **ERROR** in rot auf dem Bildschirm dargestellt.

7 Grundlegende Konzepte

7.1 Konfigurationsdatei und Kommandozeile

In allen Programmen wird ein Modul verwendet, das die Kombination von Optionen auf der Kommandozeile und in der Konfigurationsdatei verwaltet. Weil dieses Modul in allen Programmen verwendet wird, trifft diese Beschreibung grundsätzlich für alle zu. Es gibt jedoch Programme, die beide Schnittstellen unterstützen (wie `storeBackup.pl`) und andere, die nur die Kommandozeile unterstützen (wie `storeBackupMount.pl`). Generell unterstützen komplexere Programme beide Schnittstellen und einfachere nur die Kommandozeile.

Konfigurationsdatei

Die Struktur der Konfigurationsdatei ist:

```
1 keyword = Liste der Parameter
```

Es macht keinen Unterschied, es so zu schreiben:

```
1 keyword=Liste      der      Parameter
```

Wenn Du zu viele Parameter für eine optisch noch schöne Zeilenlänge in der Konfigurationsdatei hast, kannst Du in der nächsten Zeile weiterschreiben. Hierzu musst Du eine oder mehrere Leerstellen oder Tabulatoren an den Beginn der (Folge-) Zeile setzen:

```
1 keyword = Liste der Parameter die aber wirklich wirklich
2 wirklich zu lang für eine einzige Zeile sind
```

Du kannst Kommentare schreiben, indem Du ein Doppelkreuz an den Beginn der betroffenen Zeile stellst:

```
1 # Dies ist ein Kommentar
```

Du kannst einen Kommentar auch erzeugen, indem Du einen Semikolon (;) an den Beginn einer Zeile setzt. StoreBackup nutzt dieses zur besseren Lesbarkeit, um auskommentierte Schlüsselworte in Konfigurationsdateien zu platzieren. Teilweise wird diese Methode auch verwendet, um auskommentierte Schlüsselworte zu erkennen. Du solltest also die Vorgehensweise nicht verändern.

Du kannst, wie in einer Shell, Environment-Variablen verwenden, hier ist es `$VAR`:

```
1 keyword = $VAR $VARvar
```

Falls `$VAR` auf `XXX` gesetzt war, ergibt das folgendes Ergebnis:

```
1 keyword = XXX XXXvar
```

Du kannst Quotes benutzen:

```
1 keyword = 1 2 "1 2" '1 2' $VAR "$VAR 1" '$VAR' '$VAR 1'
```

Dieses wird intern folgendermaßen expandiert (die Klammern gibt es nicht, sie sollen nur die Gruppierung der Parameter zeigen):

```
1 keyword = <1> <2> <1 2> <1 2> <XXX> <XXX 1> <$VAR> <$VAR 1>
```

Weiterhin kannst Du „special characters“ mit einem Backslash (\) maskieren. Es handelt sich um folgende Zeichen:

```
1 $ " ' \
```

Es hängt vom Schlüsselwort ab, wie viele „Worte“ Du ihnen zuweisen kannst. Es gibt auch hinterlegte Regeln, die es nur erlauben, bestimmte Schlüsselworte zu verwenden, wenn andere gesetzt sind. Und schlussendlich sind nicht alle Buchstaben (oder Worte) für alle Schlüsselworte erlaubt.

Kommandozeile

Auf der Kommandozeile gibt es immer eine lange Option und manchmal ein Kürzel. Die lange Option beginnt mit `--`, während die kurze mit `-` beginnt. Beispiel:

```
1 # storeBackup.pl --file backup.config
2 # storeBackup.pl -f backup.conf
```

ist äquivalent. Dieses ist gleichzeitig ein Beispiel für eine Option, die genau *einen* Parameter (den Dateinamen) haben kann.

Es gibt andere Optionen, die mehr als einen Parameter haben können:

```
1 # storeBackup.pl .... -e /proc -e /tmp -e /var/tmp
```

In diesem Beispiel wird die Option `-e` (dasselbe wie `--exceptDirs`) verwendet, um mehrere Verzeichnisse anzugeben, die nicht in die Sicherung sollen. Diese Option wird einfach wiederholt. Es spielt keine Rolle, ob die lange oder die kurze oder eine Mischung daraus verwendet wird.

Sehen wir uns nun eine Option an, die für die Definition des Programms verwendet wird, mit dem Dateien aus dem Backup entkomprimiert werden. Nehmen wir `gzip -d`, ein Programm mit einem Parameter:

```
1 # storeBackup.pl .... --uncompress "gzip -d"
```

Wie in der Beschreibung von `storeBackup.pl` in Kapitel 6.2 vermerkt, wird der Parameter dieser Option so ausgewertet, als wenn er in der Konfigurationsdatei stehen würde (die Quotes werden von der Shell entfernt):

```
1 uncompress = gzip -d
```

Auf diese Art sieht `storeBackup.pl` zwei Parameter (`gzip` und `-d`) für die Option `--uncompress`. Der Erste wird dann als das Programm angenommen, die folgenden als Parameter für dieses Programm.

Manchmal können auch List-Parameter (Parameter ohne eine Option) verwendet werden. Im Programm `storeBackup.pl` heißen sie z. B. `otherBackupSeries`.

Zusammenspiel von Konfigurationsdatei und Kommandozeile

In einigen Programmen (wie `storeBackup.pl`) kannst du sowohl Kommandozeilenparameter als auch eine Konfigurationsdatei verwenden. In der Regel ist es übersichtlicher und einfacher, die Konfigurationsdatei zu verwenden.

In einigen Situationen ist es aber sehr praktisch, eine Option in der Konfigurationsdatei auf der Kommandozeile zu überschreiben. Du kannst das ganz einfach dadurch erreichen, dass die Option auf der Kommandozeile *auch* gesetzt wird. In den Programmen, die mit `storeBackup` geliefert werden, überschreiben die Kommandozeilenparameter die Einstellungen in der Konfigurationsdatei.

Es gibt eine spezielle Situation, in der dieses eigentlich unmöglich ist. Stell Dir vor, in der Konfigurationsdatei von `storeBackup.pl` steht:

```
1 doNotDelete = yes
```

Dieses bedeutet, dass `storeBackup.pl` alteBackups nicht löscht. Dieses erfolgt später in einem gesonderten Ablauf mit `storeBackupDel.pl`. Aber das einzige, was `storeBackupDel.pl` zu bieten hat, ist diese Option als Flag zu verwenden: `--doNotDelete`, was lediglich *yes* in der Konfigurationsdatei bedeutet. Es gibt keine gesonderte Option, um *no* zu sagen. Anstelle für derartige Fälle dutzende von Optionen einzuführen, verwenden die Programme die folgende Syntax:

```
1 # storeBackupDel.pl -f backup.config --unset doNotDelete
```

Diese setzt die in der Konfigurationsdatei gesetzte Option `doNotDelete` zurück („unset“).

Du kannst auch schreiben:

```
1 # storeBackupDel.pl -f backup.config --unset --doNotDelete
```

was der Name dieser Option auf der Kommandozeile ist.

Für List-Parameter gibt es eine Zuordnung der List-Parameter ohne Option zu einer speziellen Option in der Konfigurationsdatei – für `storeBackup.pl` ist das `otherBackupSeries`; für `storeBackupSearch.pl` ist es `backupRoot`. Dieses ist bei den einzelnen Programmen dokumentiert.

7.2 Auswahl von Verzeichnissen / Dateien für die Sicherung

StoreBackup erlaubt es, unterschiedliche Methoden für die Auswahl von Verzeichnissen oder individuellen Dateien zur Sicherung – oder dem Ausschluss aus der Sicherung – zu verwenden. Die unten gelisteten Möglichkeiten von `storeBackup.pl` können kombiniert werden.

Auswahl von Verzeichnissen:

- **includeDirs** Ermöglicht die Selektion der angegebenen Verzeichnisse für das Backup. Du *musst* den relativen Pfad unter `sourceDir` angeben. Falls Du diese Option verwendest, werden nur diese Verzeichnisse gesichert – sonst nichts. Du kannst Unterverzeichnisse mit der Option `exceptDirs` von der Sicherung ausnehmen.
- **exceptDirs** Ermöglicht den Ausschluss der gewählten Verzeichnisse aus dem Backup. Du *musst* den relativen Pfad unter `sourceDir` angeben. Falls Du diese Option verwendest, werden die ausgewählten Directories nicht gesichert. Unterverzeichnisse, die mit dieser Option ausgewählt werden, können nicht mit `includeDirs` (doch noch) in das Backup einbezogen werden.
- **followLinks** Ermöglicht die sehr flexible Abbildung von Verzeichnissen in das Backup. Um diese Option zu verwenden, erzeuge ein spezielles Verzeichnis für das Backup, in das die zu sichernden Verzeichnisse über symbolische Links abgebildet werden. Wahrscheinlich wirst Du diese Option mit `exceptDirs` kombinieren wollen. Siehe auch die Beschreibung der Option `followLinks` in Kapitel 6.2. Im Beispiel 2 wird die Verwendung von `followLinks` ebenfalls erläutert.
- **stayInFileSystem** Bei Verwendung dieser Option werden nur Dateien und Verzeichnisse gesichert, die über die Option `sourceDir` oder über die Hardlinks (die als Verzeichnisse interpretiert werden) der Option `followLinks` spezifiziert werden.

Bei der Auswahl von Verzeichnissen für das Einbeziehen oder den Ausschluss aus der Sicherung können Wildcards verwendet werden. StoreBackup ruft für ihre Auflösung eine Shell auf und schreibt das Resultat in die Log-Datei(en). Falls Dir *Wildcard* nichts sagt und Du wissen willst, was das ist, kannst Du eine Internet-Suchmaschine verwenden, um nach „wildcard shell“ zu suchen.

Auswahl von Dateien:

- **includeRule** Wenn Du diese Regel angibst, werden *nur* Dateien, die dieser Auswahl genügen, gesichert. Diese Regel wird *vor* `exceptRule` (soweit definiert) ausgewertet.
- **exceptRule** Wenn Du diese Regel angibst, werden Dateien, die der Auswahl genügen, von der Sicherung ausgenommen. Da diese Regel *nach* Option `includeRule` ausgewertet wird, wird eine Datei, die *sowohl* in `includeRule` *als auch* in `exceptRule` ausgewählt wird, *nicht* gesichert.
- **exceptTypes** Hier angegebene Dateitypen werden nicht im Backup gesichert. Diese Option ist nützlich, wenn Du beispielsweise einige Arten von „special files“ nicht sichern willst.

Da sich `includeRule` und `exceptRule` nur auf *Dateien* beziehen, wird die Sicherung der Verzeichnisstruktur von ihnen nicht beeinflusst. Insbesondere ist die Selektion von Verzeichnissen selbst nicht möglich.

In einer Regel kannst Du die Regel-Funktionen `MARC_DIR` und `MARC_DIR_REC` verwenden, um *alle Dateien* eines Verzeichnisses (ggf. inklusive seiner Unterverzeichnisse) auszuwählen. Siehe BEISPIEL 5 in Kapitel 7.4, „Definition von Regeln“ über die Verwendung dieser Regel-Funktionen.

Um eine bessere Kontrolle und Nachvollziehbarkeit über die aufgrund von *Regeln* ausgeschlossenen Dateien zu erhalten, ist es möglich, mit der Option `writeExcludeLog` die betroffenen Dateien (mit relativem Pfad) in die Datei `.storeBackup.notSaved.bz2` ins oberste Verzeichnis des gerade erstellten Backups zu schreiben.

7.3 Löschen von alten Backups

Der eher konventionelle Ansatz

StoreBackup gibt Dir viele Möglichkeiten zum Löschen (oder Behalten) der alten Backups. Falls Du ein Backup hast, was nie gelöscht werden sollte, ist die einfachste Art, das zu erreichen, es durch Anhängen eines Minus-Zeichens – gefolgt von Text – umzubenennen. Zum Beispiel:

```
$ mv 2003.07.28_06.12.41 2003.07.28_06.12.41-archive
```

Umbenannte Backup müssen genau diesem Muster entsprechen: `yyyy.mm.dd_dd.mm.ss-(.+)`

Wichtig: Falls Du die Option `lateLinks` von `storeBackup.pl` verwendest, darfst Du erst umbenennen, nachdem `storeBackupUpdateBackup.pl` erfolgreich gelaufen ist!

Umbenannte Backups werden von neuen Backups nicht mehr referenziert (via Hardlinks), also in der Regel erst umbenennen, wenn das Backup nicht mehr das Neueste ist!

Archivieren durch ein einfaches Umbenennen ist möglich, weil `storeBackup.pl` und `storeBackupDel.pl` nur Verzeichnisse (Backups) löschen, die *exakt* dem Muster `YYYY.MM.DD_hh.mm.ss` entsprechen.

Die einfachste Art, ein Backup zu löschen, ist mittels `rm -rf`. *Mach das nicht, wenn Du die Option `lateLinks` verwendest!* Wenn Du zu alte Sicherungen abhängig von Regeln löschen willst, hast Du mehrere Möglichkeiten. Du kannst die Aufbewahrungszeit auf Basis von bestimmten Wochentagen (mit einem zugrunde liegendem Standardwert für alle Tage) bestimmen; du kannst die Aufbewahrungszeit mit den Optionen `keepFirstOfYear`, `keepLastOfYear`, `keepFirstOfMonth` und `keepLastOfMonth` festlegen. Oder Du kannst mit `keepFirstOfWeek` oder `keepLastOfWeek` definieren, wie lange Sicherungen aufbewahrt bleiben sollen. In all diesen Fällen musst Du den Zeitraum für die Aufbewahrung angeben. Wie das geht, ist bei den Beschreibungen der jeweiligen Optionen von `storeBackup.pl` erklärt.

Nun stell Dir vor, dass Du Deine Sicherungen unregelmäßig durchführst, vielleicht von einem Laptop auf einen Server; oder Du machst Deine Sicherungen, wenn Du meinst, einen wichtigen Arbeitsschritt fertiggestellt zu haben. In diesen Fällen ist es nützlich zu sagen: „Behalte das letzte Backup eines Tages länger“ (mit `keepDuplicate`). Wenn Du für einen Monat in Urlaub warst, und `keepAll` auf `30d` gesetzt war, dann wirst Du wahrscheinlich nicht wollen, dass `storeBackup.pl` alle Deine alten Backups löscht, wenn Du es das erste Mal startest, nachdem Du zurück bist. Vermeiden kannst Du dies mittels der Option `keepMinNumber`. Andererseits, wenn Du nur begrenzten Platz für Deine Sicherungen hast, möchtest Du vielleicht die Anzahl der Sicherungen generell begrenzen – dies geht mit `keepMaxNumber`.

Mit `keepDuplicate` gibst Du einen Zeitraum an, in dem `storeBackup` doppelte Backups eines Tages aufbewahrt. Nach diesem Zeitraum wird nur das letzte Backup eines Tages erhalten.

Mit `keepMinNumber` gibst Du die minimale Anzahl von Backups an, die erhalten wird. Die Logik hierfür ist wie folgt:

- Lösche keine Backups, die den anderen `keep*` Optionen behalten werden sollen.
- Wenn das nicht reicht, lösche beginnend mit den neuesten keine anderen Backups. Dubletten eines Tages werden hierbei nicht berücksichtigt.

Mit `keepMaxNumber` bestimmst Du die maximale Anzahl von Backups. Falls notwendig, wird `storeBackup` demzufolge das älteste Backup löschen. Um besondere Backups vom Löschen auszunehmen, kann ein „Archiv Flag“ bei den `keep*` Optionen verwendet werden. Backups, auf die dieses zutrifft, werden durch `keepMaxNumber` niemals gelöscht. So ist es möglich, dass mehr Backups übrigbleiben als mit `keepMaxNumber` vorgesehen – aber das „Archiv“ Flag ist nützlich, um besondere Backups wie „letztes Backup eines Monats“ oder „letztes Backup eines Jahres“ vor dem Löschen zu schützen.

Verwendung von `keepRelative` als Lösch-Strategie

Diese Option aktiviert eine alternative Lösch-Strategie, die es Dir erlaubt, das relative Alter von Backups anstelle einer Aufbewahrungsperiode zu definieren.

Stell Dir vor, Du hättest immer die folgenden Backup verfügbar:

- 1 Backup von gestern
- 1 Backup von letzter Woche
- 1 Backup vom letzten Monat
- 1 Backup von vor drei Monaten

Beachte, dass das sehr wahrscheinlich *nicht* das ist, was Du willst, und zwar weil es ganz einfach bedeutet, dass Du tägliche Backups durchführen musst und jedes Backup für exakt 3 Monate behalten musst. Ansonsten könntest Du kein Backup vorhalten, dass das geforderte *exakte* Alter hat.

Was Du wirklich willst, ist daher so etwas wie:

- 1 Backup mit einem Alter zwischen 1 Stunde und 24 Stunden (1 Tag)
- 1 Backup mit einem Alter zwischen 1 Tag und 7 Tage

- 1 Backup mit einem Alter zwischen 14 und 31 Tagen
- 1 Backup mit einem Alter zwischen 80 und 100 Tagen

Das wäre eine sehr gewöhnliche Backup-Strategie, aber Du würdest Schwierigkeiten haben, dieses mit den anderen `keepFirstOf*` Optionen zu definieren, insbesondere, wenn Du die Backups nicht regelmäßig durchführst. Allerdings kannst Du dieses Verhalten sehr einfach mit `keepRelative` durchführen. Du musst nur Folgendes angeben:

```
keepRelative = 1h 1d 7d 14d 31d 80d 100d
```

Das heißt, Du listet hier alle Intervalle auf, für die Du Backups haben willst. `storeBackup` wird die Backups in einer Art und Weise löschen, die dem gewünschten so nahe wie möglich kommt. (Wenn Du nicht genügend Backups machst, kann aber auch `storeBackup` nichts dagegen unternehmen.)

Beachte, dass dieses bedeuten kann, dass `storeBackup` mehr Backups erhalten muss als Du denkst; z. B. könnte es zwei Backups in einer Periode behalten. In diesen Fällen „sieht `storeBackup` in die Zukunft“ und stellt fest, dass beide Backups *später* nötig sind, um Backups für alle Perioden vorzuhalten. Dies ist auch der Grund, warum im obigen Beispiel implizit die Periode 7-14 Tage spezifiziert wird, auch wenn Du in ihr kein Backup haben willst. Um Backups in der nächsten Periode (14-31 Tage) zu haben, benötigst Du immer auch ein Backup in der Periode 7-14 Tage. Aus diesem Grund erlaubt es die Syntax nicht, Perioden auszuschließen.

Schlussendlich solltest Du darauf achten, dass `storeBackup` alle Intervalle, für es kein passendes Backup finden kann, verschiebt: Wenn Dein erstes Backup zwischen 10 und 20 Tage sein soll, aber das nächste aktuelle 25 Tage alt ist, werden alle folgenden Perioden um 5 Tage verlängert. Wenn Du über einen längeren Zeitraum keine Backups gemacht hast, stellt dieses Verhalten sicher, dass dieser Zeitraum Dein Backup-Schema nicht durcheinanderbringt. Hier ein Beispiel, warum dieses sinnvoll ist: Wenn du Backups haben willst, die 1, 3, 7 und 10 Tage alt sind und Du gehst für 14 Tage in Urlaub, dann ist es sehr unwahrscheinlich, dass Du alle Backups gelöscht haben willst, wenn Du zurückkommst. Daher ignoriert `storeBackup` diese 14 Tage und behält die Backups entsprechend länger.

7.4 Definition von Regeln

Regeln können z. B. in `storeBackup.pl`, siehe Kapitel 6.2 (Optionen `exceptRule`, `includeRule`, `comprRule`) sowie in `storeBackupSearch.pl`, siehe section 6.6 (Option `searchRule`) definiert werden. Beide unterstützen die Definition von Regeln auf der Kommandozeile und in der Konfigurationsdatei.

Dieser Teil der Beschreibung zeigt, wie in `storeBackup` Regeln verwendet werden können. Wenn Du Dich nicht mit Pattern Matching in perl auskennst, solltest Du die Beispiele sehr vorsichtig und immer nur etwas verändern. Du kannst jedoch leicht Fehlermeldungen bekommen, die Du nicht verstehst.

Alle Beispiele sind so erklärt, wie sie in Konfigurationsdateien verwendet werden können. Meistens wird ein Schlüsselwort von `storeBackup.pl` benutzt (`exceptRule`²²), aber die Syntax für die Regeln ist identisch für die anderen Schlüsselwörter (z. B. `includeRule` oder `searchRule`). Später sehen wir dann, wie die Regeln auf der Kommandozeile verwendet werden können.

Alle Zeiten oder Zeiträume, die im folgenden erwähnt werden, betreffen die Zeit des Backups selbst, *nicht* die der Dateien im Backup selbst!

Allgemein gesagt: Regeln sind ein paar Zeilen Perl mit einigen Spezialitäten. Starten wir mit einem einfachen und typischen Beispiel:

BEISPIEL 1:

```
1 exceptRule = '$size > 1610612736'
```

(Beachte die Anführungszeichen und erzeuge eine Konfigurationsdatei mittels `storeBackup.pl` oder mit `storeBackupSearch.pl` und lies die Kommentare am Anfang darüber, wie das Quoting funktioniert und wie Environment-Variablen interpretiert werden.)

Diese Regel trifft für alle Dateien mit mehr 1.5GB ($1.5 * 1024^3$) Bytes zu. `$size` repräsentiert die Größe jeder individuellen Datei. In diesem Beispiel werden alle Dateien mit mehr als 1.5GB Größe nicht gesichert. Das ist nicht gut lesbar, daher gibt es folgende Möglichkeit:

```
1 exceptRule = '$size > &::SIZE("1.5G")'
```

²²In Kapitel 7.2 „Auswahl von Verzeichnissen / Dateien für die Sicherung“ findest Du eine Übersicht über die verschiedenen Möglichkeiten zum Auswählen und Ausschließen von Dateien oder Verzeichnissen.

(Beachte alle Anführungszeichen.) Dieses hat denselben Effekt wie die Definition vorher. `&::SIZE` ist eine Funktion, die den zahlenmäßigen Wert des Strings „1.5G“ errechnet. Du kannst Kennungen von 'k' bis 'P' mit der folgenden Bedeutung verwenden:

1k	1 kilobyte = 1024 Byte
1M	1 Megabyte = 1024 ² Byte
1G	1 Gigabyte = 1024 ³ Byte
1T	1 Terabyte = 1024 ⁴ Byte
1P	1 Petabyte = 1024 ⁵ Byte

z. B. ist `&::SIZE("0.4T")` richtig, aber nicht `&::SIZE("1G1M")`.

BEISPIEL 2:

```
1 exceptRule = '$file =~ m#bak$#'
```

(Beachte alle Anführungszeichen.) Diese Regel gilt für alle Dateien mit der Endung '.bak', was bedeutet, dass diese nicht gesichert werden. `$file` repräsentiert jeden individuellen Dateinamen mit dem *relativen Pfad* unterhalb der Option `sourceDir` von `storeBackup.pl`. Falls Du das komische Ding hinter `$file` nicht verstehst – das nennt sich regulärer Ausdruck. Siehe `man perlretut` (perl regular expression tutorial) für eine ausführliche Dokumentation. Aber Du solltest auch so in der Lage sein, es für einfache Anforderungen anzupassen:

```
1 exceptRule = '$file =~ m#bak$#' or '$file =~ m#mpg$#'
```

(Beachte alle Anführungszeichen und *alle* Leerstellen.) Diese Regel gilt für alle Dateien, die mit '.bak' oder mit '.mpg' enden und verhindert daher ihre Sicherung.

```
1 exceptRule = '$file =~ m#bak$#' or '$file =~ m#mpg$#'
2 or '$file =~ m#avi$#'
```

Es sollte jetzt keine Überraschung sein, dass hiermit Dateien mit '.bak', '.mpg' bzw. '.avi' nicht gesichert werden.

Nun schreiben wir eine Regel, die verhindert, dass Dateien gesichert werden, die mit '.bak', '.mpg' oder '.avi' enden sowie alle Dateien, die größer als 500 Megabyte sind:

```
1 exceptRule = '$file =~ m#bak$#' or '$file =~ m#mpg$#'
2 or '$file =~ m#avi$#' or '$size > &::SIZE("0.5G")'
```

Wenn Du `debug = 2` setzt, kannst Du für jede einzelne Datei sehen, *wie* die Regeln ausgewertet werden. Wenn Du `debug = 1` setzt, kannst Du für die einzelnen Dateien (nur) das Ergebnis sehen. Mit `debug = 0` (default) erfolgt keine Ausgabe.

Du kannst die folgenden vorgelegten Variablen verwenden:

<code>\$file</code>	Dateiname mit relativem Pfad vom originalen <code>sourceDir</code>
<code>\$size</code>	Größe der Datei in Byte
<code>\$mode</code>	Rechte der Datei (Integer, verwende 0...mit oktalen Zahlen, z. B. <code>\$mode = 0644</code>)
<code>\$ctime</code>	Erzeugungszeit in Sekunden seit Epoch (1. Jan. 1970), siehe unten
<code>\$mtime</code>	Modifikationszeit in Sekunden seit Epoch, siehe unten
<code>\$uid</code>	User ID (String, falls im OS bekannt), z. B. <code>\$uid eq 'bob'</code>
<code>\$uidn</code>	User ID (numerischer Wert), z. B. <code>\$uidn = 1001</code>
<code>\$gid</code>	Gruppen ID (String, falls im OS bekannt), siehe <code>\$uid</code>
<code>\$gidn</code>	Gruppen ID (numerischer Wert), see <code>\$uidn</code>
<code>\$type</code>	Typ der Datei, ein Buchstabe von <code>SbcFp1</code> sein, s. Option <code>exceptTypes</code> in <code>storeBackup.pl</code>
<code>\$ENV{XXX}</code>	verwendet den Inhalt der Environment Variable <code>XXX</code>

Falls Du `$ctime` oder `$mtime` verwenden willst, ist es nicht das reine Vergnügen, die Anzahl der Sekunden seit Epoch (1.1.1970) zu berechnen. Aus diesem Grund unterstützt `storeBackup` eine spezielle Funktion `&::DATE` um Dein Leben angenehmer zu machen:

BEISPIEL 3:

```
1 searchRule = '$mtime > &::DATE("14d")' and '$mtime < &::DATE("3d12h")'
```

Mit dieser Suchregel (in `storeBackupSearch.pl` findest Du alle Dateien, die neuer als exakt 14 Tage und älter als 3 Tage und 12 Stunden sind. Die Syntax, die von `&::DATE` verstanden wird, ist:

1.

d	day (Tag)
h	hour (Stunde)
m	minute (Minute)
s	second (Sekunde)

Daher bedeutet „3d2h50s“ 3 Tage, 2 Stunden und 50 Sekunden. Mit Hilfe der Funktion oben kann „jetzt“ minus diesem Zeitraum angegeben werden.

2.

YYYY.MM.DD	year.month.day (Jahr.Monat.Tag)
YYYY.MM.DD_hh.mm.ss	dasselbe Format wie backupDir
2008.04.30	spezifiziert den 30. April 2008, 0:00,
2008.04.30_14.03.05	spezifiziert den 30. April 2008, um 14 Uhr 3 Min. und 5 Sek.

Mit der Funktion `&::DATE` kann auch ein fester Zeitpunkt angegeben werden.

Du hast bereits einige Möglichkeiten gesehen, wie „Variablen“ gruppiert werden können. Es geht mit **and** und **or**. Du kannst verwenden:

```
1 and, or, not, (, )
```

Alles ist wie in perl. (Um ehrlich zu sein: es wird auch durch den Perl-Interpreter ausgewertet.) Aber Du solltest diese Ausdrücke (also **und**, **oder**, **(**, etc.) mit einem oder mehreren Leerzeichen (white space) umgeben, ansonsten wird `debug = 2` nicht richtig funktionieren!

BEISPIEL 4:

```
1 searchRule = ( '$mtime < &::DATE("14d")' and '$mtime > &::DATE("3d12h")' )
2 and not '$file =~ m#bak$'
```

Finde alle Dateien, die älter als 3 Tage und 12 Stunden und jünger als 14 Tage sind, aber nicht solche, die auf `.bak` enden.

Beachte, dass **and**, **not**, **(** und **)** mit mindestens einem Leerzeichen umrahmt werden.

BEISPIEL 5:

StoreBackup ermöglicht Dir, sein Verhalten mit Hilfe von Regeln, die durch das Setzen von „Marker-Dateien“, die im Quellverzeichnis gesetzt werden können, zu kontrollieren. Diese können Auswirkungen auf ein spezielles Verzeichnis oder auch rekursiv auf deren Unterverzeichnisse haben.

In der Konfigurationsdatei von `storeBackup.pl` definierst Du:

```
1 exceptRule= '&::MARK_DIR($file)'
```

Dann werden alle Dateien in Verzeichnissen mit der Default-„Marker“-Datei `.storeBackupMark` nicht gesichert, weil die Funktion `MARK_DIR` in diesen Fällen 1 zurückliefert.

Du kannst auch eine andere Marker-Datei verwenden, z. B. `.dontBackup`:

```
1 exceptRule= '&::MARK_DIR($file, ".dontBackup")'
```

In diesem Fall musst Du – natürlich – die Datei `.dontBackup` in den Verzeichnissen, die Du nicht sichern willst erzeugen:

```
1 $ touch .dontBackup
```

Wenn Du willst, dass alle Dateien rekursiv unter Verzeichnissen mit `.dontBackup` vom Backup ausgeschlossen werden, dann definiere die Regel folgendermaßen:

```
1 exceptRule= '&::MARK_DIR_REC($file, ".dontBackup")'
```

Da es sich um Regeln handelt, kannst Du die Bedeutung auch umdrehen:

```
1 exceptRule= not '&::MARK_DIR($file, ".backupThisDir")'
```

Als Namen für die Marker Datei verwendest Du hier `.backupThisDir`. Alle Backups *ohne* diesen Marker werden jetzt *nicht* gesichert.

Du kannst beliebige Kombinationen mit anderen Elementen der Regeln erstellen. Wenn Du z. B. in Verzeichnissen mit dem Marker `.saveRootOnly` oder deren Unterverzeichnissen nur Dateien vom User `root` sichern willst, definiere:

```

1 exceptRule= not ( '&::MARK_DIR_REC($file, ".saveRootOnly")'
2 and '$uid eq "root"' )

```

Jetzt stell Dir vor, Du willst rekursiv Dateien in Verzeichnissen mit dem Marker `.dontBackup` *nicht* sichern *und* rekursiv in Verzeichnissen mit Marker `.saveRootOnly` nur Dateien von root sichern. Das kannst Du folgendermaßen erreichen:

```

1 exceptRule = '&::MARK_DIR_REC($file, ".dontBackup")'
2 or not
3 ( '&::MARK_DIR_REC($file, ".saveRootOnly")'
4 and '$uid eq "root"' )

```

Du kannst jede Kombination mit unterschiedlichen Marker-Dateien in unterschiedlichen oder denselben Regeln verwenden.

Natürlich kannst Du `MARK_DIR` und `MARK_DIR_REC` auch in anderen Regeln verwenden, insbesondere bei Option `comprRule`.

Verwendung von Regeln auf der Kommandozeile

Sehen wir uns das an:

```

1 exceptRule = '$size > &::SIZE("1.5G")'

```

Wenn wir das Kommando so benutzen:

```

1 --exceptRule '$size > &::SIZE("1.5G")'          ### FALSCH ###

```

bekommen wir hier hässliche Fehlermeldungen, weil die Shell die einfachen Anführungszeichen (') entfernt und storeBackup das Resultat genauso interpretiert wie in der Konfigurationsdatei (siehe Erläuterung in jeder generierten Konfigurationsdatei ganz oben). Hier wird sich storeBackup darüber beklagen, dass die Environment-Variable `$size` nicht gesetzt ist. (Das `$`-Zeichen ist nicht mehr maskiert, weil die Shell die einfachen Anführungszeichen entfernt hat.) Daher müssen wir das `$`-Zeichen maskieren; außerdem auch die doppelten Anführungszeichen ("), weil storeBackup sie als Gruppierungszeichen interpretieren würde, also nicht direkt in den Perl-Interpreter weiterleiten würde. Die richtige Schreibweise für die Option wäre also:

```

1 --exceptRule '$size > &::SIZE(1.5G)'          ### RICHTIG ###

```

Beispiel 4 müssten wir so angeben:

```

1 --searchRule '( $mtime < &::DATE(14d) and $mtime > &::DATE(3d12h) ) and not $file =~ m#bak$#'

```

Im Falle von Problemen solltest Du die Perl-Fehlermeldung lesen, die zeigt, was wirklich beim Perl-Interpreter ankommt. Abgesehen davon wird die Option `--print` jeden Parameter nach dem Parsen (Auswerten) durch die Shell und storeBackup anzeigen. Du kannst `--print` auch in Verbindung mit Konfigurationsdateien verwenden.

Die folgenden vordefinierten Funktionen können in den Regeln (rules) verwendet werden:

`&::DATE(zeitraum)` Berechnet die relative Zeit von „jetzt“ und gibt Dir die Möglichkeit, diese mit `$ctime` oder `$mtime` zu vergleichen. Setze `zeitraum` z.B. auf `"3d2h"`. Für Details siehe Beispiel 3 in diesem Kapitel.

`&::SIZE(groesse)` Berechnet die Größe `groesse`, die in lesbarer Form (z.B. 1.5G) angegeben wird als einfache Zahl (in Bytes). Für Details siehe Beispiel 1 in diesem Kapitel.

`&::MARK_DIR($file [, dateiMarkierung])` Überprüft, ob `dateiMarkierung` im Verzeichnis von `$file` existiert. Der Defaultwert für `dateiMarkierung` ist `.storeBackupMark`. Für Details siehe Beispiel 5 in diesem Kapitel.

`&::MARK_DIR_REC($file [, dateiMarkierung])` Überprüft, ob `dateiMarkierung` im Verzeichnis von `$file` oder in einem darüber liegenden Verzeichnis existiert. Der Defaultwert für `dateiMarkierung` ist `.storeBackupMarkRec`. Für Details siehe Beispiel 5 in diesem Kapitel.

7.4.1 Festlegen, ob eine Datei komprimiert werden soll

Das Verhalten von storeBackup bezüglich der Kompression von Dateien kann mit `comprRule` `exceptSuffix`, `addExceptSuffix`, `compressSuffix` und `minCompressSize` gesteuert werden. Dabei wird die Option `comprRule` aus den anderen generiert, so dass sie nicht konfiguriert werden muss. Wenn Du `comprRule` aber selbst setzt, werden die Parameter der anderen Optionen ignoriert. (Du musst `comprRule` direkt verwenden, wenn Du sehr ausgefallene Dinge konfigurieren willst.)

Verwendung der Vorgaben:

Wenn Du die Standardwerte verwendest, verhält sich storeBackup Version 3.3 genau so wie die vorherigen Versionen. Die Standardwerte sind:

```
1 exceptSuffix = zip bz2 gz tgz jpg gif tiff tif mpeg mpg mp3 ogg
2   gpg png
3 addExceptSuffix =
4 compressSuffix =
5 minCompressSize = 1024
```

Du kannst den Wert von `minCompressSize` oder die Endungen bei `exceptSuffix` ändern oder Endungen zu `addExceptSuffixes` hinzufügen. So lange wie `compressSuffix` *nicht* gesetzt ist, wird storeBackup intern eine Regel der folgenden Art generieren (hier mit Standardwerten):

Komprimiere keine Dateien, die kleiner als 1k Byte sind oder eine der Endungen wie in exceptSuffix oder addExceptSuffix definiert haben.

Wenn Du `exceptSuffix` oder `minCompressSize` nicht definierst, werden die default Werte genommen!²³

Gar keine Komprimierung:

Wenn die Regel `comprRule` 0 zurückliefert, wird die untersuchte Datei nicht komprimiert. Daher kann man einfach Folgendes konfigurieren:

```
1 comprRule = 0
```

Jede Datei komprimieren:

Wenn die Regel `comprRule` 1 zurückliefert, wird die untersuchte Datei komprimiert. Daher kann man einfach Folgendes konfigurieren:

```
1 comprRule = 1
```

Aber das ist nicht wirklich sinnvoll. Warum eine Datei komprimieren, die nicht weiter komprimiert werden kann und daher nach der Komprimierung größer ist als vorher?

Verwendung einer Positiv- und einer Negativliste

In den meisten Fällen kennst Du Dateitypen (über ihre Endung), die Du *nicht* komprimieren willst (wie `.jpg`) und andere, bei denen es sinnvoll ist, sie zu komprimieren (wie `.doc`, `.bmp`, `.txt`, ...).

Wenn Du eine oder mehrere Endungen bei `compressSuffix`, z. B. `.pdf` definierst:

```
1 compressSuffix = pdf
```

dann wird sich storeBackup folgendermaßen verhalten:

Dateien, die kleiner sind als der Wert von minCompressSize werden nicht komprimiert. Dateien mit Endungen wie in exceptSuffix oder addExceptSuffix werden nicht komprimiert. Dateien mit Endungen wie in comprSuffix definiert werden komprimiert. Für den Rest der Dateien wird auf Basis von COMPRESSION_CHECK entschieden, ob die Dateie komprimiert wird.

Lass storeBackup entscheiden:

Es gibt eine spezielle Regel-Funktion, die abschätzt, ob sich das Komprimieren einer Datei lohnt. Diese Regel-Funktion liefert 1, wenn sie meint, dass die Datei komprimiert werden sollte und 0 wenn nicht. Nun definieren wir diese einfache Regel:

```
1 comprRule = '&::COMPRESSION_CHECK($file)'
```

²³Diese ist kompatibel zum Verhalten vor der Einführung von `compression Suffix` und `COMPRESSION_CHECK`.

Die Regel oben funktioniert recht zuverlässig, aber oft ist es nicht nötig diese Auswertung zu machen. Daher kannst Du einfach die anderen Optionen (siehe oben) setzen und musst Dich nicht um `comprRule` kümmern (siehe die automatisch generierte Regel wie oben beschrieben).

Empfehlung:

Folgendes passt in den meisten Fällen: Definiere einige Endungen für zu komprimierende (`comprSuffix`) bzw. nicht zu komprimierende Endungen (ergänze bei Bedarf die Liste über `addExceptSuffix`) in Abhängigkeit von Deinen Anforderungen. Das war's.

Anmerkung: Wenn Du das „blocked file“-Feature von storeBackup nutzt, kannst Du diese Funktionalität dort auch nutzen, z. B. durch Setzen der Option `checkBlocksCompr` mit `check`. Siehe Blocked Files (Kapitel 7.5) für weiterführende Informationen.

Die individuelle Lösung

Wenn Du sehr spezielle Anforderungen hast, z. B. alles so wie oben beschrieben zu konfigurieren, *aber* die Dateien für bestimmte User oder Gruppen *nicht* zu komprimieren, musst (und kannst) Du individuelle Regeln definieren. Beachte dabei die Hinweise von Kapitel 7.4 und verwende sehr kleine Backups mit Debug-Level 3 zum Testen.

7.5 Sicherung von Image Files / raw Devices / Blocked Files

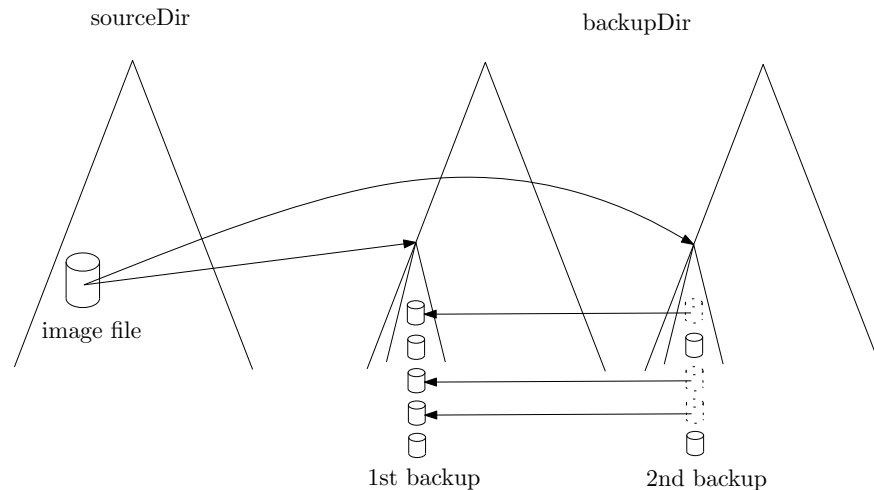
Die Möglichkeiten von Blocked Files

Große Image-Dateien, die sich nur in Teilen geändert haben, komplett zu sichern ist ineffizient: Platz und Zeit werden verschwendet. Hier einige Beispiele:

- Einige E-Mail Programme verwenden das traditionelle mbox (mailbox) Format, um E-Mails zu speichern. Das ist praktisch, weil es sich dabei um ein sehr gut unterstütztes Format handelt. Aber Du erhältst eine große Datei von eventuell mehreren Gigabyte mit allen Deinen E-Mails darin. Das zu sichern heißt alles zu sichern – trotz der Tatsache, dass sich nur ein sehr kleiner Teil darin geändert hat.
Dieselbe Kategorie sind die `.pst` Dateien von Outlook. Wenn Du solche Dateien sichern musst (und sie groß sind), solltest Du darüber nachdenken, „Blocked Files“ zu verwenden.
- Wenn Du eine Image-Datei mit einem verschlüsselten Dateisystem darin verwendest, wie das z. B. TrueCrypt macht, solltest Du die verschlüsselte Datei sichern, nicht die Dateien darin. Wenn Du die Dateien darin sicherst, brauchst Du einen weiteren verschlüsselten Container, was bedeutet, dass das Backup-Programm alle Passworte benötigt, um automatisch die Sicherungen durchzuführen – was eine perfekte Sicherheitslücke ist.
Aus diesem Grund ist es sinnvoll, das binäre Daten-Image selbst so zu sichern, wie es ist. Wenn Du eine einfache Kopie machst, wird diese jedes Mal die Größe der Image-Datei haben (Du kannst diese Daten nicht komprimieren). Dieses ist eine ideale Situation, storeBackups „Blocked Files“-Möglichkeiten (ohne Kompression) zu nutzen; womit Du viele historisierte Versionen des Images haben kannst, ohne zu viel Platz zu benötigen und ohne ein Sicherheitsloch (storeBackup braucht kein Passwort zu wissen und weiß nichts über den Inhalt, den es sichert).
- Images für Hypervisor wie Xen, DVM oder VMware sind ein anderes Beispiel für den sehr erfolgreichen Einsatz von „Blocked Files“.
- Du solltest „Blocked Files“ nicht für Dateien verwenden, die als Ganzes komprimiert sind, wie jpegs oder komprimierte oder verschlüsselte Dateien (`.gz`, `.bze`, `gpg`, etc). Änderungen an einem Teil derartiger Dateien führen meist zu einer Änderung der gesamten Datei.
- Das Feature von Blocked Files ist auch nicht sinnvoll für Datenbank-Dumps, weil storeBackup (bis jetzt) mit festen Blockgrößen arbeitet. Wenn man nur ein Byte an den Beginn einer Datei hinzufügt, sind alle Blöcke unterschiedlich.

So funktioniert's

Wenn Du eine Datei für das blockweise Sichern spezifizierst (Beschreibung siehe unten), dann geht `storeBackup.pl` wie folgt vor:



1. Erzeugt ein Verzeichnis mit derselben Kombination von Pfad und Dateinamen wie die originale Image Datei im Quellverzeichnis.
2. Teilt die Quelldatei in Blöcke und überprüft, ob die Blöcke irgendwo anders im Backup vorkommen (siehe Option `otherBackupSeries` von `storeBackup.pl`). Wenn ein Block schon existiert, wird ein Hardlink gesetzt; wenn er nicht existiert, wird er kopiert oder komprimiert gespeichert.
3. Die md5-Summe aller dieser Dateien wird in einer speziellen Datei namens `.md5BlockCheckSums.bz2` in diesem Verzeichnis gespeichert.
4. `storeBackup.pl` berechnet ebenfalls die md5-Summe der gesamten Datei und speichert sie in der Datei `.md5CheckSum`.

Weil auf schon vorhandene Dateien über Hardlinks referenziert wird, ist jedes Backup vollständig (full backup).

Wenn Du die Option `lateLinks`, siehe Kapitel 7.6 verwendest, werden die Links später erzeugt. Wenn Du die Option `lateCompress` verwendest, wird die Kompression ebenfalls erst später durchgeführt.

Speichern großer Image-Dateien

Es gibt zwei Arten zu konfigurieren, welche Dateien `storeBackup.pl` als „Blocked Files“ behandeln soll:

1. Der einfachste Weg ist, die folgenden Optionen zu verwenden:

checkBlocksSuffix Die Konfiguration ist ähnlich zu `exceptSuffix`; eine Liste von Endungen wird auf Übereinstimmung überprüft, z. B. `\.vmdk` für VMware Images. Sie bedeuten einfach nur, dass der letzte Teil des Dateinamens mit dem übereinstimmen muss, was Du hier vorgegeben hast.

Die hier als nächstes beschriebenen Optionen werden nur ausgewertet, wenn `checkBlocksSuffix` gesetzt ist.

checkBlocksMinSize Nur Dateien mit der hier festgelegten Minimalgröße werden als Blocked Files behandelt. Du kannst dieselben Kürzel verwenden wie in Definition von Regeln, siehe Kapitel 7.4, das bedeutet z. B. `50M` 50 Megabytes. Der Standardwert ist `100M`.

checkBlocksBS Legt die Blockgröße fest, in die die betroffenen „Blocked Files“ von `storeBackup.pl` zerteilt werden. Das Format ist mit `checkBlocksMinSize` identisch. Der Standardwert ist `1M`; der Minimalwert ist `10k`.

checkBlocksCompr Definiert, ob die Blöcke komprimiert werden sollen. Mögliche Werte sind `yes`, `no` oder `check`. Auf der Kommandozeile ist `--checkBlocksCompr` zu setzen.

Dieses Flag betrifft nur die Dateien, die mit `checkBlocksSuffix` ausgewählt wurden.

Beispiel:

Du willst alle Deine VMware-Images und auch einige Outlook `.pst`-Dateien sichern. Das Blocked File-Feature von `storeBackup` soll für Dateien mit minimal 50 Megabyte für Dateien, die auf `.vmdk` oder `.pst` enden verwendet werden. Die gewählte Blockgröße ist `500k` und die resultierenden Blöcke im Backup werden komprimiert:


```

1 checkBlocksSuffix = 'vmdk' 'pst'
2 checkBlocksMinSize = 50M
3 checkBlocksBS = 500k
4 checkBlocksCompr = yes

```

- Die flexiblere Art und Weise, mit Blocked Files zu arbeiten, ist, Regeln wie in Definition von Regeln, siehe Kapitel 7.4 beschrieben zu verwenden. Die folgenden Optionen sind fünf Mal verfügbar, es gibt `checkBlocksRule0`, `checkBlocksRule1`, `checkBlocksRule2`, `checkBlocksRule3` und `checkBlocksRule4`:

`checkBlocksRulei` Die *i*te Regel zur Festlegung, welche Dateien als Blocked Files im Backup gespeichert werden sollen.

`checkBlocksBSi` Die korrespondierende Blockgröße für die Blöcke im Backup. Der Standardwert ist 1 Megabyte; der minimal zulässige Werte 10k.

`checkBlocksCompri` Die Blöcke werden komprimiert, wenn auf `yes` gesetzt. Wenn auf `no` gesetzt, werden sie nicht komprimiert. Wenn auf `check` gesetzt, schätzt `storeBackup` selbst ab, ob sich eine Kompression lohnen würde. Dieses kann zu einem Mix von komprimierten und kopierten Blöcken führen.

`checkBlocksReadi` Definiert einen Filter für das Lesen der hier als „Blocked Files“ spezifizierten Dateien, z. B. `gunzip` oder `gzip -d`. Diese Option kann nützlich sein, wenn eine bereits komprimierte Datei vorliegt. (Die Verwendung des „Blocked File“-Features von `storeBackup` mit bereits komprimierten Dateien ist nicht sinnvoll.)

Beispiel:

Angenommen, Du hast ein TrueCrypt-Image auf Deiner Platte und willst jedes Mal ein Backup davon anlegen, wenn Du `storeBackup.pl` startest. Du wählst den unauffälligen Namen `myPics.iso`, Blockgröße ist 1M, keine Kompression. Dementsprechend legst Du Regel 0 fest:

```

1 checkBlocksRule0= '$file =~ m#/myPicsiso$#'
2 #checkBlocksBS0=
3 #checkBlocksCompr0=
4 checkBlocksRule1= '$size > &::SIZE("50M")' and
5 ( '$file =~ m#pst$#' or '$file =~ m#windows_D/Outlook/#' )
6 checkBlocksBS1=200k
7 checkBlocksCompr1=check

```

Hier hast Du auch Regel 1 definiert, die für alle Dateien größer als 50 Megabyte, die auf `.pst` enden oder im relativen Pfad unterhalb von `windows_D/Outlook` liegen. (Ich verwende diese Konfiguration, um Daten meines dual Boot Laptops zu sichern.) Wenn Du mit dem Erstellen von Regeln nicht vertraut bist, solltest Du Kapitel 7.4 lesen..

Du kannst `checkBlocksSuffix` und `checkBlocksRulei` gleichzeitig in der Konfigurationsdatei verwenden. `StoreBackup` überprüft erst `checkBlocksRulei` (in aufsteigender Reihenfolge), danach `checkBlocksSuffix`.

Sichern von ganzen Devices

Das Sichern eines ganzen Devices (wie `/dev/sdc` oder `/dev/sdc1`) erfolgt mit `storeBackup` auf dieselbe Art und Weise wie das Sichern einer Image-Datei. Du wählst die Devices mit `checkDevicesi`, die Blockgröße im Backup mit `checkDevicesBSi` und schaltest die Komprimierung mit `checkDevicesCompri` an, aus, oder auf `check`. Zusätzlich musst Du den relativen Pfad mit `checkDevicesDiri` im Backup festlegen. Dort wird der Inhalt des Devices gespeichert.

Die Blöcke, die sich im Backup aus den Image-Dateien oder Devices ergeben, werden mit Hardlinks verbunden, sofern `storeBackup` Übereinstimmungen findet.

Die Optionen im Detail:

`checkDevicesi` Liste der zu sichernden Devices (z. B. `/dev/sdd2` `/dev/sde1`).

`--checkDevicesDiri` Verzeichnis, in dem die Inhalte der Devices innerhalb des Backups gespeichert werden (relativer Pfad). Die Image-Datei wird in dieses (relative) Verzeichnis zurückgesichert, wenn Du `storeBackupRecover.pl` benutzt (bei Verwendung von Standard-Parametern). In diesem Verzeichnis erzeugt `storeBackup` ein Unterverzeichnis mit einem von den Parametern der Option `checkDevices` abgeleiteten Name, z. B. wird `/dev/sdc` zu `/dev_sdc`.

checkDevicesBSi Legt die Blockgröße fest, in die die Devices von **storeBackup.pl** aufgeteilt werden. Das Format ist identisch mit **checkBlocksMinSize**. Der Standardwert ist 1M; der minimal erlaubte Werte ist 10k.

checkDevicesCompri Legt fest, ob die Blöcke komprimiert werden sollen. Mögliche Werte sind **yes**, **no** oder **check**; der Standardwert ist **no**.

Diese Option betrifft nur Devices, die mit **checkDevicesi** ausgewählt wurden. Wenn Du diese Option auf **check** setzt, wird jeder Block individuell auf Komprimierung überprüft.

Wahl der Blockgröße

Es gibt keine feste Regel für die „beste“ Blockgröße. Ich habe einige Messungen bezüglich der Blockgröße und des benötigten Platzes gemacht. Das zweite Backup wurde dabei mit **lateLinks** gemacht (siehe Kapitel 7.6), so dass ich mit **df** sehen konnte, wie viel Platz wirklich benötigt wurde. Das verwendete Dateisystem war reiserfs mit tail packing. Wenn Du ein Dateisystem ohne tail packing (wie ext2, ext3 oder ext4) verwendest, ist der Overhead größer und kleine Blockgrößen werden uninteressanter (genauso wie bei Verwendung von Komprimierung). Das Resultat hängt auch von der Applikation ab, die in die originale Image-Datei schreibt.

Alle Beispiele wurden aus Performance-Gründen ohne Kompression durchgeführt. Sie erfolgten mit echten Daten. In meinen realen Backups verwende ich natürlich die Komprimierung. Das zweite Backup zeigt den Platz, der für die Änderungen benötigt wird. Die Prozentzeile darunter zeigt die Relation zwischen dem ersten und zweiten Backup. Die Summenzeile zeigt die Summe des Platzbedarfs von erstem und zweitem Backup. Die nächste Zeile (1x) zeigt das Verhältnis des letzten Werte in der Zeile (große Blockgröße, 5M) zur aktuellen Spalte für ein Folgebackup. Die letzte Zeile zeigt dasselbe für 10 Folgebackups (es wurde angenommen, dass der Platzbedarf bei jedem zusätzlich benötigten Backup gleich ist.) Daher enthält diese Zeile die interessantesten Werte.

Das erste Beispiel zeigt das Resultat bei Sicherung einer großen Outlook .pst-Datei von 1.2GB mit den Änderungen, die ich zwischen zwei Tagen hatte:

Blockgröße	50k	100k	200k	1M	5M
1. Backup [kB]	1219253	1172263	1172863	1173801	1173724
2. Backup [kB]	7692	13445	22720	73826	240885
	0.63%	1.15%	1.94%	6.29%	20.52%
Summe [kB]	1226945	1185708	1195583	1247627	1414609
1x	86.73%	83.82%	84.52%	88.20%	100.00%
10x	36.18%	36.47%	39.08%	53.37%	100.00%

Das zweite Beispiel wurde mit einer kleineren Outlook Datei von 117 Megabyte durchgeführt. Diese dient als „Posteingang“. Die Zahlen zeigen ein anderes Verhalten als beim ersten Beispiel.

Blockgröße	50k	100k	200k	1M	5M
1. Backup [kB]	122487	118221	118891	119184	119181
2. Backup [kB]	33400	51240	74424	107632	119181
	27.27%	43.34%	62.60%	90.31%	100.00%
Summe [kB]	155887	169461	193315	226816	238362
1x	65.40%	71.09%	81.10%	95.16%	100.00%
10x	34.82%	48.10%	65.84%	91.19%	100.00%

Das dritte Beispiel zeigt das Resultat bei Speicherung einer VMware Image Datei von 2.1 GB. Zwischen dem ersten und dem zweiten Backup wurde die VM gebootet, ein Programm zum Updaten meines Navigationssystems auf einen neuen Stand gebracht und das Navigations Gerät selbst für einen Update angeschlossen.

Blockgröße	50k	100k	200k	1M	5M
1. Backup [kB]	2162595	2106781	2112547	2117178	2117094
2. Backup [kB]	53656	80609	131701	438241	1112652
	2.48%	3.83%	6.23%	20.70%	52.56%
Summe [kB]	2216251	2187390	2244248	2555419	3229746
1x	68.62%	67.73%	69.49%	79.12%	100.00%
10x	20.38%	21.99%	25.90%	49.08%	100.00%

Bei allen Tabellen sieht man, dass ab einem gewissen Punkt eine Verkleinerung der Blockgröße den benötigten Platz nicht mehr reduziert. Ein optimaler Wert scheint zwischen 50k und 200k zu liegen (sofern tail packing verwendet wird).

Es gibt einen weiteren wichtigen Aspekt bezüglich der Blockgröße: Bei kleineren Blockgrößen wird die Performance schlechter. Um eine akzeptable Performance zu erreichen, sind folgende Optimierungen implementiert:

- Wenn Du die Blöcke mit `storeBackup.pl` nicht komprimierst (überhaupt nicht oder erst später wegen Verwendung von `lateCompress`), wird keine Parallelisierung verwendet.
- Wenn die Blöcke von `storeBackup.pl` komprimiert werden und eine Blockgröße von 1 Megabyte oder mehr verwendet wird, parallelisiert `storeBackup.pl`.
- Falls die Kompression innerhalb von `storeBackup.pl` mit `bzip2` erfolgt und eine Blockgröße kleiner 1 Megabyte konfiguriert ist, versucht `storeBackup.pl` das Perl-Modul `IO::Compress::Bzip2` zu verwenden. Wenn es auf dem System installiert ist, wird es verwendet.

Am Besten machst Du eigene Tests, um ein Gefühl für eine sinnvolle Blockgröße für Deinen Anwendungsfall zu bekommen.

7.6 Verwendung der Option `lateLinks`

Du kannst `storeBackup` als ein Programm (`storeBackup.pl`) verwenden, das alles macht, oder Du kannst die unterschiedlichen Aktivitäten aufteilen. Es gibt primär einen Vorteil bei einer Aufteilung: Die Zeit für das Backup selbst ist aus Sicht des zu sichernden Rechners kürzer.

Die Verwendung von `lateLinks` ist sinnvoll, wenn Du Deine Backups auf einen NFS-Server speicherst und meinst, dass es eine gute Idee ist, das Ganze zu beschleunigen (siehe Kapitel 5.4.3, Performance). Die Konfiguration von `lateLinks` ist ein wenig komplizierter als die Verwendung von `storeBackup.pl` als alleiniges Programm, da Du mehrere Programme kombinieren musst.

`StoreBackup.pl` als alleiniges Programm erledigt folgende Aufgaben:

1. Die Link-Konsistenz von allen Backups (von allen Backup Serien) wird überprüft. Was das bedeutet, wird später erläutert.
2. Laden von Metadaten von einem oder mehreren älteren Backups. Dieser Schritt ist wie eine Initialisierung, in der Dateinamen, md5-Summen, Datum, Zeiten und einige andere Informationen von den alten Backups geholt werden.
3. Überprüfen von allen zu sichernden Dateien, ob eine andere Datei mit demselben Inhalt bereits in diese alten Backups existiert, von denen Dateinamen, md5-Summen usw. geladen wurden.
4. Die veränderten Daten werden ins Backup transferiert: durch kopieren, komprimieren oder durch Hardlinks. Normalerweise wird auch die Verzeichnisstruktur generiert.
5. Für jede Datei werden Besitzer und Rechte so gesetzt, wie sie im Quellverzeichnis sind.
6. Abhängig von den Regeln in den `keep*` Optionen von `storeBackup.pl` werden alte Backups gelöscht.

Wenn Du `storeBackup.pl` mit der Option `lateLinks` startest, werden der Datentransfer (siehe Schritt 4) und sämtliche Aktionen auf dem entfernten Dateisystem auf das absolut notwendige Minimum reduziert:

- Die veränderten oder neuen Dateien (inklusive Spezialdateien) werden kopiert. Veränderte Dateien, die komprimiert werden sollen, werden abhängig von Option `lateCompress` nur kopiert. Es hängt von Deiner Situation ab, ob die Verwendung von `lateCompress` sinnvoll ist oder nicht.
- Es werden keine Hardlinks im neuen Backup erzeugt.
- Verzeichnisse werden nur erzeugt, wenn sie für das Kopieren / Komprimieren von Dateien notwendig sind.

- Im neuen Backup wird folgende Datei erzeugt: `.storeBackupLinks/linkFile.bz2`. Sie beinhaltet alle Informationen darüber, was hätte getan werden müssen, um ein vollständiges Backup mit allen Hardlinks, Verzeichnissen und Komprimierungen zu erzeugen. Die richtigen Rechte (die auch nicht gesetzt werden), sind in der Datei `.md5Checksums` im obersten Backupverzeichnis gespeichert. Diese Datei wird in einem „vollen“ Backuplauf von `storeBackup.pl` ebenfalls generiert. Sie wird für die Rücksicherung von Daten ausgewertet (`storeBackupRecover.pl`).

Du kannst `storeBackup.pl` unabhängig von Option `lateLinks` immer so konfigurieren, dass Schritt 6, das Löschen von alten Backups, nicht erfolgt. Insbesondere wenn das Backup auf einen NFS-Server erfolgt, dauert das ein wenig und verlängert das Backup. Verwende `storeBackupDel.pl` (das die Konfigurationsdatei von `storeBackup.pl` lesen kann), um das Löschen von alten Backups vom direkten Backup-Prozess zu trennen.

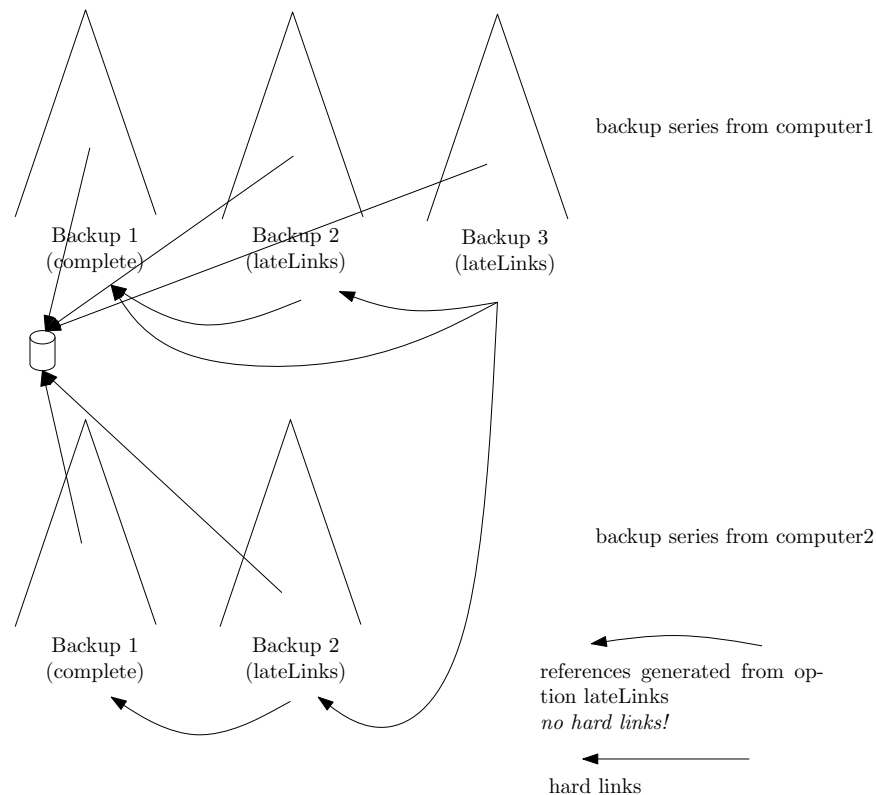
Es ist wichtig zu verstehen, dass die Verwendung der Option `lateLinks` ein unvollständiges Backup erzeugt. Derartige Backups enthalten alle Daten, die gesichert werden sollten; daher ist die Basis vorhanden. Aber der Ablauf von `storeBackup.pl` ist nicht vollständig:

- Verzeichniseinträge fehlen
- Dateien sind nicht komprimiert (falls `lateCompress` verwendet wurde)
- Es wurden keine Hardlinks gesetzt.
- Die Rechte sind weder für Dateien noch für Verzeichnisse richtig gesetzt.
- Beachte, dass alle Informationen, die notwendig sind, um ein vollständiges Backup zu vollenden, vorhanden sind! Im Falle der Hardlinks bedeutet das, dass in der Datei `linkFile.bz2` die Informationen darüber enthalten sind, welche Hardlinks zu Dateien in alten Backups zu setzen sind. In diesen älteren Backups kann aber eventuell auch nur ein Eintrag in `linkFile.bz2` existieren, und so weiter. Also pass auf: Wenn Du eines der Backups löschst, auf das referenziert wird, zerstörst Du *alle folgenden Backups*, die darauf direkt oder indirekt verweisen. **Lösche Backups nur mit `storeBackup.pl` oder `storeBackupDel.pl` – verwende niemals `rm` oder ähnliches!** Diese beiden Programme achten auf die gerade beschriebenen Abhängigkeiten. Falls Du Backups mit `rm` löschst, stell sicher, dass `storeBackupUpdateBackup.pl` vorher alle Backups erfolgreich beendet hat. Wenn alle Hardlinks gesetzt sind, existiert absolut kein Unterschied zwischen den unterschiedlich erzeugten Backups.

Mit der Option `lateLinks` werden temporär inkrementelle Backups erzeugt, aus denen später mittels `storeBackupUpdateBackup.pl` unter Zuhilfenahme alter Backups volle Backups generiert werden.

Die folgende Abbildung zeigt zwei über Kreuz verlinkte Backup-Serien von unterschiedlichen Rechnern.²⁴ Du siehst, dass die aus Option `lateLinks` resultierenden Abhängigkeiten komplex sein können. Die Hardlinks selbst sind niemals ein Problem, weil es kein Original oder eine „richtige“ Referenz gibt – jeder Hardlink ist ein Originalzeiger auf die Datei oder präziser auf den Inode (siehe Kapitel 7.12). Eine Datei wird erst mit dem letzten Hardlink gelöscht. Jedoch sind die Referenzen von `lateLinks` nur Dateinamen in einer Datei und haben nichts mit dem Dateisystem zu tun.

²⁴siehe Kapitel 6.2, Option `otherBackupDir` über derartige Konfigurationen



Verwende `storeBackupUpdateBackup.pl`, siehe Kapitel 6.3, um ein nicht beendetes Backup zu beenden (all dieses Linken und Komprimieren und so weiter). Es wird Deine Backups analysieren (unter `backupDir`) und die richtige Reihenfolge für die Komplettierung finden. *Nach dem erfolgreichen, gesonderten Lauf von `storeBackupUpdateBackup.pl` befinden sich Deine Backups in demselben Zustand, als hättest Du die Backups ohne Option `lateLinks` durchgeführt.* Neben anderem wird die Datei `.storeBackupLinks/linkFile.bz2` gelöscht und alles hart verlinkt (und komprimiert) sowie die Rechte analog zum Quellverzeichnis gesetzt (außer Du hast Option `ignorePerms` in `storeBackup.pl` gesetzt).

Wenn Du Option `lateLinks` verwendest, solltest Du `storeBackupUpdateBackup.pl` regelmäßig, z. B. jede Nacht laufen lassen und überprüfen, ob ERROR-Meldungen erzeugt wurden.

Zusammenfassung:

- Mit `lateLinks` werden nicht vollendete Backups erzeugt. Mit `storeBackupUpdateBackup.p` können diese später vervollständigt werden.
- Du kannst kein vollständiges Backup erstellen (also ein Backup ohne `lateLinks`), wenn dieses neue Backup auf ein vorheriges unvollständiges Backup verweisen soll. Das ist einfach deshalb so, weil Hardlinks auf nicht vorhandene Dateien nicht gesetzt werden können.
- Bei Verwendung von `storeBackup.pl` oder `storeBackupDel.pl` können keine Backups gelöscht werden, auf die noch unvollendete Backups verweisen.

7.7 Isolated Mode / Offline Backups

Der typische Anwendungsfall für die hier beschriebene Verwendung von `storeBackup` ist eine Reise mit dem Laptop mit einem kleinen Backup-Medium (z. B. ein Memory Stick).

`StoreBackup isolatedMode` liefert die ideale Lösung für dieses Szenario.

Vorteile:

- Du kannst inkrementelle Backups laufen lassen, ohne Zugriff auf das Master-Backup zu haben.
- Du kannst diese inkrementellen Backups auf kleinen, portablen Medien wie einem Memory Stick machen.²⁵

²⁵Wir beziehen uns hier auf die Verwendung eines Memory Sticks für die Reise, aber es kann natürlich jede Art von Speicher verwendet werden, auch große Festplatten.

- Du hast die Historie aller Änderungen während der Reise.
- Du kannst die inkrementellen Backups von der Reise einfach in Dein Master-Backup integrieren, nachdem Du zurückgekommen bist.

Einschränkungen:

- Es ist empfehlenswert, Konfigurationsdateien zu verwenden, wenn storeBackup im Isolated Mode verwendet werden soll.²⁶
- Bei Verwendung dieser Methodik hast Du natürlich *keinen* Zugriff auf die Daten in Deinem Master-Backup .
- Wenn die auf ältere Dateien aus Deinem-Master-Backup zugreifen willst, benötigst Du den Zugriff darauf (oder jemand muss Dir die Dateien per E-Mail sondern oder ähnliches).

Angenommen, Du hast eine große Backup-Platte (oder ein RAID oder was auch immer) bei der Arbeit oder zu Hause, auf dem Dein Master-Backup liegt. Nun bist Du für sagen wir eine Woche unterwegs. Du willst in der Zeit Backups durchführen, ohne (mit ausreichender Performance) Zugriff auf Dein Master-Backup zu haben (dieses wird im Folgenden als Offline Backups bezeichnet). Du möchtest auch das bequemste Speichermedium während der Reise verwenden. Und am Wichtigsten: Du weißt, dass Du Deine Backups von unterwegs in das Master-Backup integrieren möchtest.²⁷

Beim `isolatedMode` von storeBackup wird der Memory Stick initialisiert, indem die Metadaten des letzten Backups auf ihn geschrieben werden. Weiterhin wird die Konfigurationsdatei für `storeBackup.pl` für die Reise angepasst.²⁸ Dann trittst Du die Reise an und erstellst inkrementelle Backups auf Deinem Stick. Diese Backups benötigen gewöhnlich nur sehr wenig Speicher, weil sie nur die (komprimierten) Deltas (Unterschiede) beinhalten. Wenn Du zurückkommst und Zugriff auf das Master-Backup hast, startest Du `storeBackupMergeIsolatedBackup.pl`, welches Deine inkrementellen Backups an die richtige Stelle ins Master-Backup kopiert. Mittels `storeBackupUpdateBackup.pl` werden dann automatisch alle Maßnahmen ergriffen, um die inkrementellen Backups in platzsparende volle Backups umzuwandeln.²⁹ Natürlich unterstützen Dich die Tools von storeBackup dabei, um diese Schritte einfach und reproduzierbar zu machen.

7.7.1 So werden Isolated Backups verwendet:

- Bereite Deinen lokalen Speicher (z. B. den Memory Stick) vor:
Dieser Schritt ist nicht wirklich spezifisch bezogen auf storeBackup. Er ist vergleichbar mit den grundlegenden Schritten, um irgendein Linux-Speichermedium vorzubereiten: Erzeugen eines Linux Dateisystems (z. B. `ext2`, `reiserfs`), welches in der Lage ist, Hardlinks zu speichern. Du kannst das mit graphischen Tools wie `gparted` oder auf der Kommandozeile durchführen (mit `fdisk -l` zur Identifikation des Sticks und `mkfs.ext2` oder `mkfs.reiserfs` oder anderes um das Dateisystem zu erzeugen. Du kannst auch einen Stick mit NTFS verwenden.
Monte den Stick und erzeuge ein Verzeichnis als oberstes Backup Verzeichnis für `storeBackup.pl`.
- Verwende `storeBackupSetupIsolatedMode.pl` um die Metadaten automatisiert aus Deinem Master-Backup auf das für die Reise zu verwendende Medium zu kopieren. Es müssen nur ein paar Megabyte kopiert werden – `storeBackup.pl` mit Option `lateLinks` benötigt nur die Metadaten und keine „echten“ Daten, um inkrementelle Backups erzeugen zu können.
ANMERKUNG: Dieses Skript ändert auch Deine Konfigurationsdatei für `storeBackup.pl` – die Änderungen hängen davon ab, welche Einstellungen Du gewählt hattest.
Aus Sicherheitsgründen nimmt `storeBackupSetupIsolatedMode.pl` Metadaten ausschließlich aus fertigen Vollbackups. Wenn das letzte Backup der gewählten Serie kein Vollbackup ist, wirst Du aufgefordert, `storeBackupUpdateBackup.pl` laufen zu lassen.

²⁶Es handelt sich nicht wirklich um eine Einschränkung; aber falls Du noch keine Konfigurationsdateien verwendest, wäre jetzt der richtige Zeitpunkt, damit zu beginnen, um alle Vorteile aus diesem Modus zu ziehen. Die Verwendung ist mit Konfigurationsdateien einfacher und angenehmer.

²⁷Eine Integration könnte auch mit `linkToDirs.pl` erfolgen, aber `isolatedMode` benötigt *nicht* den Speicher für das gesamte Backup. Auch ist es für diese Verwendung einfacher und angenehmer in der Handhabung.

²⁸Siehe Kapitel 7.7.1. Abhängig vom Aufruf von `storeBackupSetupIsolatedMode.pl` wird eine neue Konfigurationsdatei aus Deiner existierenden generiert.

²⁹Das bedeutet, dass Kompression, Hardlinken, Setzen von Datum/Zeit, Rechner und Eigentümer durchgeführt werden.

Abhängig vom Aufruf von `storeBackupSetupIsolatedMode.pl` wird ggf. auch eine angepasste Version Deiner Konfigurationsdatei erzeugt.

- Verwende `storeBackup.pl`, um wie üblich Deine Backups zu erzeugen. Falls Deine Konfigurationsdatei durch `storeBackupSetupIsolatedMode.pl` angepasst wurde, kannst Du die neue erzeugte verwenden. Andernfalls musst Du die Optionen selbst anpassen (z. B. `backupDir`, `lateLinks`).
- Nach den Sicherungsläufen mit `storeBackup.pl`, mit denen die inkrementellen Backups erzeugt wurden, verwendest Du `storeBackupMergeIsolatedBackup.pl`, um die (inkrementellen) Backups in Dein Master-Backup einzufügen, sobald Du Zugriff darauf hast. Alle diese Schritte sind einfacher, wenn Du Konfigurationsdateien verwendest.
Wie Du siehst, ist es eine gute Idee, Konfigurationsdateien im Zusammenhang mit dem Isolated Mode zu verwenden.
- Nach dem Lauf von `storeBackupMergeIsolatedBackup.pl` sind Deine lokalen Backups in das Master-Backup-Verzeichnis kopiert worden. Du solltest jetzt `storeBackupUpdateBackup.pl` laufen lassen (oder warten, bis z. B. cron das erledigt). Wenn alles geklappt hat, kannst Du die betroffenen Dateien (Verzeichnisse) auf Deinem lokalen Medium (z. B. dem Memory Stick) löschen.

Du kannst den Isolated Mode natürlich zusammen mit der Replikation von Backups verwenden, siehe Replikation von Backups, siehe Kapitel 7.8.

Du findest detaillierte Informationen in `storeBackupSetupIsolatedMode.pl`, siehe Kapitel 6.7. Siehe auch die Beschreibung von `storeBackupMergeIsolatedBackup.pl`, Kapitel 6.8.

WARNUNG: Eine Sicherung Deines `lateLinks`-Backups auf ein Dateisystem im `fat` oder `vfat` Format (z. B. auf dem Stick) wird nicht unterstützt! Diese Art von Dateisystem ist nicht in der Lage, zwischen Dateinamen mit Groß- oder Kleinbuchstaben zu unterscheiden. Das bedeutet, dass der Dateiname `datei.txt` identisch mit `Datei.txt` ist – und das ohne eine Warnung oder Fehlermeldung. Wenn Du zwei Dateien oder Verzeichnisse mit demselben Namen (also nur unterschieden durch Groß- und Kleinbuchstaben) in einem Verzeichnis hast, werden definitiv nicht alle Dateien in Deinem Backup sein. Du kannst aber NTFS verwenden, wenn Du willst!

7.7.2 Einrichten von Isolated Mode

Um die nötigen Schritte zu erklären, gehen wir unter Verwendung einer Konfigurationsdatei Schritt für Schritt durch ein simples Beispiel.³⁰

Du kannst z. B. (erst mal) diese Demo durchspielen und dann die Pfade an Deinen Gegebenheiten anpassen.

Lauf eines Backups gegen das Master-Backup Als Erstes tun wir das, was Du schon durchgeführt haben solltest: Ein (neues) Backup im Master-Backup erzeugen.

Für diese Demonstration erzeuge ich in meinem Home-Verzeichnis das Verzeichnis `isol-test`, in diesem ein Master-Backup-Verzeichnis (`backup`) und ein Source-Verzeichnis (`source`). Dort kopiere eine Datei hinein und generiere eine Konfigurationsdatei:

```
1 # cd
2 # mkdir isol-test
3 # cd isol-test
4 # mkdir backup source
5 # ls
6 # cp -v /bin/ls source
7 # ls -l source
8 # storeBackup.pl -g stbu.conf
```

Verwende als nächstes einen Editor Deiner Wahl und ändere die folgenden Einträge in `stbu.conf`:

```
1 sourceDir=source
2 backupDir=backup
```

³⁰Im Folgenden wird davon ausgegangen, dass alle Programme von `storeBackup` im `$PFAD` liegen, so dass sie ohne Pfad aufrufbar sind.

Nun lassen wir ein Vollbackup laufen, damit wir die Metadaten auf das externe Medium kopieren können:

```
1 # storeBackup.pl -f stbu.conf
2 WARNING 2012.06.09 09:07:57 5647 created directory <backup/default>
3 ... <snip, deleted output of storeBackup.pl>
```

Die Warnung teilt uns mit, dass `storeBackup.pl` ein Unterverzeichnis für die Serie `default` angelegt hat.

Einrichten von Isolated Mode Du kannst das Kommando `ls` oder einen Dateibrowser verwenden um zu sehen, was im Backup-Verzeichnis `backup` erzeugt wurde.

Verbinde nun Dein externes Medium, z. B. jenen Memory Stick, mit dem Rechner: Er muss nun mit einem Linux Dateisystem oder mit NTFS (*nicht* FAT) bespielt werden. Stell sicher, dass er immer unter denselben Pfad gemountet wird. Dies kann auf unterschiedliche Art und Weise erfolgen, eventuell abhängig von der Unterstützung durch die Distribution und / oder der GUI, die Du verwendest. Wenn Du keine Ahnung hast wie Du das machen könntest, hast, suche mit Hilfe einer Internet-Suchmaschine wie Google oder einer anderen nach "`blkid fstab`".

In den folgenden Einstellungen gehe ich davon aus, dass Dein externes Medium nach `/media/stick` gemountet wurde. Pass den Pfad `/media/stick` im Weiteren an Deine Gegebenheiten an.

Jetzt können wir den Stick initialisieren, nachdem wir ein Backup-Verzeichnis auf ihm erzeugt haben:

```
1 # mkdir /media/stick/stbu
2 # storeBackupSetupIsolatedMode.pl -f stbu.conf -t /media/stick/stbu
3 INFO 2012.06.09 09:27:29 5888 ./isolate-stbu.conf: changed <backupDir> to '/media/stick/stbu'
4 INFO 2012.06.09 09:27:29 5888 ./isolate-stbu.conf: created <mergeBackupDir> as 'backup'
5 INFO 2012.06.09 09:27:29 5888 ./isolate-stbu.conf: setting <otherBackupSeries> to 0:default
6 INFO 2012.06.09 09:27:29 5888 ./isolate-stbu.conf: changed <lateLinks> to 'yes'
7 INFO 2012.06.09 09:27:29 5888 you may want to adjust <./isolate-stbu.conf> to your needs
```

Das Programm `storeBackupSetupIsolatedMode.pl` gibt aus, dass es eine neue Konfigurationsdatei namens `isolate-stbu.conf` mit einigen Anpassungen erzeugt hat: `backupDir` wurde auf das Verzeichnis auf dem Stick gesetzt und `lateLinks` (zur Option `lateLinks` von `storeBackup.pl`, siehe Kapitel 7.6) wurde eingeschaltet. Der Eintrag `mergeBackupDir`, der von `storeBackupMergeIsolatedBackup.pl` später dazu verwendet wird, Dein Isolated Backup in das zentrale Backup (im Verzeichnis `backup`) zu integrieren, wird ebenfalls erzeugt. Letztendlich wird `otherBackupSeries` auf nur diese eine Backup-Serie eingestellt. Die Erzeugung von Referenzen zu anderen Backup-Serien (die in diesem einfachen Beispiel nicht existieren) ist so nicht möglich, wenn Du Backups auf Deinen Stick erzeugst.³¹

Schau Dir die erzeugte Konfigurationsdatei an. Das Korrigieren der Optionen funktioniert nur, wenn unbenutzte Optionen durch ein Semikolon (;) und *nicht* mit einem Doppelkreuz (#) kommentiert sind.

Wenn Du die Funktionalität des „isolated mode“ *später nochmals verwenden* willst, kannst Du einen sauberen Memorystick nehmen, auf dem das Backupverzeichnis unter demselben Pfad erreichbar ist wie bei der ersten Konfiguration. Dann nimmst Du einfach die erzeugte Konfigurationsdatei, um die Metadaten auf den Stick zu kopieren:

```
1 # storeBackupSetupIsolatedMode.pl -f isolate-stbu.conf -v
```

Die Option `-v` erzeugt einige Ausgaben, damit Du siehst, was passiert.

Lauf von Backups auf dem lokalen Speichermedium Nun kopieren wir eine neue Datei in das `source` Verzeichnis und lassen ein Backup laufen:

```
1 # cp /bin/pwd source
2 # storeBackup.pl -f isolate-stbu.conf
```

Das war's. Nun sehen wir nach, was passiert ist:

³¹Das trifft auf dieses einfache Beispiel zu. Aber wenn Du mehrere Backup-Serien mit `storeBackupSetupIsolatedMode.pl` auf Dein lokales Speichermedium kopierst, kannst Du diese Option so korrigieren, dass Cross-Links zwischen ihnen ermöglicht werden – das geht aber nur, wenn beide Backup-Serien mit demselben Namen der Serien (Pfade) im Master-Backup verwendet werden.


```

1 # ls -lh /media/stick/stbu/default/*
2 /media/stick/stbu/default/2012.06.09_09.07.57:
3 total 0
4
5 /media/stick/stbu/default/2012.06.09_09.56.36:
6 total 16K
7 -rw----- 1 root root 13K Jun  9 09:56 pwd.bz2

```

Wie Du siehst, gibt es keine gesicherte Datei in dem ersten Backup-Verzeichnis (2012.06.09_09.07.57), weil nur Metadaten von `storeBackupSetupIsolatedMode.pl` kopiert wurden. Im zweiten Backup siehst Du die neue Datei `pwd`, aber nicht die Datei `ls`, da sie nicht verändert wurde. Sie wird nach der Integration per Hardlink in das Master-Backup eingeblendet werden. Wenn du einige Interna verstehen willst, solltest Du in die Kommandodatei für `storeBackupUpdateBackup.pl` sehen, um zu sehen, was zu verlinken ist:

```

1 # bzcatt /media/stick/stbu/default/*/.storeBackupLinks/linkFile.bz2
2 # link md5sum
3 #     existingFile
4 #     newLink
5 # compress md5sum
6 #     fileToCompress
7 # dir dirName
8 # symlink file
9 #     target
10 # linkSymlink link
11 #     existingFile
12 #     newLink
13 link 92385e9b8864032488e253ebde0534c3
14 ../2012.06.09_09.07.57/./ls.bz2
15 ls.bz2

```

Du kannst so viele zusätzliche Backups starten wie Du willst, es muss aber natürlich genügend Platz auf dem lokalen Speichermedium sein. Verwende `df -h /media/stick` (passe den Pfad auf Deine Gegebenheiten an), um zu sehen, wie viel Platz noch frei ist. Du kannst auch `du` verwenden, um zu sehen, wie viel Platz bisher verwendet wurde:

```

1 # du -sh /media/stick/stbu/default/*
2 24K      /media/stick/stbu/default/2012.06.09_09.07.57
3 44K      /media/stick/stbu/default/2012.06.09_09.56.36

```

Einfügen Deiner Isolated Backups in das Master-Backup Das bedeutet einfach nur, dass inkrementelle Backups vom lokalen Medium in das Master-Backup kopiert werden. Diese Aufgabe wird von `storeBackupMergeIsolatedBackup.pl` erledigt:

```

1 # storeBackupMergeIsolatedBackup.pl -f isolate-stbu.conf
2 in directory </media/stick/stbu/default>, copy
3 <2012.06.09_09.56.36>
4 to
5 <backup/default>
6 ?
7 yes / no -> yes
8 INFO    2012.06.09 10:15:11  6557 copying data . . .
9 INFO    2012.06.09 10:15:11  6557 finished copying data
10 INFO    2012.06.09 10:15:11  6557 please run
11 INFO    2012.06.09 10:15:11  6557      storeBackupUpdateBackup.pl -b "backup"

```

Das Programm verwendet den von `storeBackupSetupIsolatedMode.pl` eingefügten Parameter der Option `mergeBackupDir`, um den Pfad zum Master-Backup zu erhalten. Aus Sicherheitsgründen fragt es Dich, ob Du die in der Liste angegebenen Backups (hier nur eins) in das Master-Backup kopieren willst. Nachdem Du mit `yes` geantwortet hast, werden die Dateien kopiert.

Um ein „normales“ Vollbackup zu erhalten, starte `storeBackupUpdateBackup.pl -b backup`.

Wenn Du den Isolated Mode in genau derselben Art und Weise ein zweites Mal (nach Einspielen der Backups in das Master-Backup) verwendest, kannst Du von `storeBackupSetupIsolatedMode.pl` die Option `--backupDir` verwenden (weil Du schon eine funktionierende Konfigurationsdatei hast) oder einfach eine neue Konfigurationsdatei mit einem anderen Namen (siehe Option `--generate`) generieren und die alte verwenden, die Du eventuell an Deine Bedürfnisse angepasst hast.

7.7.3 Isolated Mode als Schutz vor Ransomware

Als Ransomware wird eine Schadsoftware bezeichnet, die Daten mittels public key (asymmetrisch) – üblicherweise in erpresserischer Absicht – verschlüsselt und die Originaldaten löscht. Der private key für die Entschlüsselung verbleibt dabei beim Erpresser. Gegen eine Geldsumme ist dieser dann erhältlich. Ob er wirklich ausgehändigt wird, ist allerdings fraglich.

Wird `storeBackup` in der einfachsten Konfiguration verwendet, z.B. durch Sicherung auf eine direkt erreichbare Festplatte (lokal eingebaut, per USB angeschlossen, über NFS beschreibbar, ...) kann eine Ransomware auch das Backup verschlüsseln – es also für eine Rücksicherung unbrauchbar machen.

Mit `storeBackup` gibt es (mindestens) zwei Möglichkeiten, eine Verschlüsselung des Backups zu verhindern.³² In beiden Fällen müssen die zu sichernden Daten `sourceDir` und das Backup `backupDir` auf unterschiedlichen Rechnern gehostet werden. Das Erlangen von `root`-Rechten auf dem Source-Host bedingt so keinerlei direkte Rechte auf dem Backup-Host:

1. Die einfachste Methode ist, die Backups vom Backup-Server „ziehen“ zu lassen. In diesem Fall liest der Backup-Server die Daten vom zu sichernden Rechner und dieser zu sichernde Rechner benötigt keinerlei Zugriffsrechte (und sollte sie auch nicht haben) auf dem Backup-Server. Der Nachteil dieser Methode ist die schlechtere Performance verglichen mit einem vom zu sichernden Rechner auf den Backup-Server unter Verwendung von `lateLinks`.
2. Es ist eine sehr gute Idee, jetzt das Kapitel 7.7.2, „Einrichten von Isolated Mode“ zu lesen und am Besten gleich auszuprobieren.

Unter Verwendung von Isolated Mode kann der beschriebene Nachteil des schreibenden Zugriffs auf das Backup-Verzeichnis umgangen werden, obwohl das Backup wie unter Punkt 1 beschrieben auf dem zu sichernden Rechner (ggf. mit `lateLinks`) läuft. Im Folgenden wird der zu sichernde Rechner als Backup-Client und der Backup-Host als Backup-Server bezeichnet. Das Vorgehen für die Sicherung ist:

(a) Vorbereitung:

- i. Erzeuge ein Verzeichnis auf dem Backup-Server, auf dem der Backup-Client zur lesenden Zugriff hat (z.B. NFS „read only“) (`mergeBackupDir`). Über diesen Weg kann vom Client aus später auf Backups für Rücksicherungen zugegriffen werden.
- ii. Kopiere mindestens ein fertiges Backup des zu sichernden Backup-Clients in diesen Bereich. (Hierzu kannst Du beispielsweise temporär NFS von „read only“ auf „read write“ ändern oder schon vorhandene Backups in den entsprechenden Bereich schieben oder kopieren.)
- iii. Gib ein Verzeichnis außerhalb des oben genannten „read only“ Bereichs (`mergeBackupDir`) auf dem Backup-Server als „read write“ (`backupDir`) frei. Diese Bereich (`backupDir`) kann auf demselben Filesystem auf dem Backup-Server liegen wie `mergeBackupDir`.³³

(b) Durchführung der Backups:

- i. Starte ein „Isolated Backup“ (mit `lateLinks`) auf dem Backup-Client. Dieses Backup schreibt in das mittels `backupDir` ausgewählte Verzeichnis. (Siehe Punkt 2(a)iii in dieser Aufzählung.) Ein derartiges Backup kann mehrmals hintereinander ausgeführt werden, was aber im Sinne des Schutzes vor Ransomware nicht sinnvoll ist.
- ii. Starte auf dem Backup-Server `storeBackupMergeIsolatedBackup.pl` und verwende die Option `--move`, falls das neu erstellte, inkrementelle „Isolated Mode“ Backup (mit `lateLinks`) auf demselben Dateisystem liegt wie die aus Sicht des Clients „read only“ Variante.

³²Die Backup-Platte nur für den Vorgang des Backups anzuschließen hilft nicht oder nur sehr eingeschränkt, da die Ransomware schon während der Durchführung des Backups aktiv werden kann.

³³Der Vorteil, wenn es auf demselben Dateisystem liegt, ist eine bessere Performance, da der gesicherte Bereich mittels `--move` in den „read only“-Bereich verschoben werden kann, was viel schneller als eine Kopie ist. Der Vorteil, es in einen getrennten Dateisystembereich zu legen ist die konsequentere Trennung – Nachteil ist, dass die Daten kopiert in den „read only“-Bereich kopiert werden müssen. Zur Konfiguration von `storeBackupMergeIsolatedBackup.pl` siehe Punkt 2(b)ii.

- iii. Starte `storeBackupUpdateBackup.pl` auf dem Backup-Server, um die fehlenden Hardlinks zu erzeugen und (falls konfiguriert) Replikationen anzustoßen.

7.8 Replikation von Backups

ANMERKUNG: Falls Du diese Replikation in Deinem Rechenzentrum verwenden willst und Fragen dazu hast, was zusätzlich zu dem hier beschriebenen Vorgehen möglich ist, dann schreibe bitte eine E-Mail. Es ist mit einigen Skripten möglich, das hier beschriebene Verhalten und die Möglichkeiten für die Replikation (z. B. zu anderen Standorten) auszubauen. Das Resultat derartiger Diskussionen könnte eine bessere Dokumentation oder neue Erweiterungen sein.

Wenn Du `storeBackup` verwendest, dann erzeugst Du neue Backups durch das Setzen von Hardlinks auf ältere Backups in derselben Serie oder vielleicht auch auf andere Serien. Du solltest Deine Sicherungen auf eine andere Platte (oder vielleicht auch auf einen anderen Rechner) speichern, so dass ein Fehler einer zu sichernden Platte (*source disk*) nicht auch Dein Backup zerstört. Diese Art von Fehlern betrifft Hardware (die Platte selbst), Dateisystem oder z. B. das Abbrennen eines Hauses. Ein RAID hilft nicht gegen die verschiedenen Möglichkeiten, seine Daten zu verlieren. (Um mehr Informationen zu diesem Thema zu bekommen, suche mit einer Suchmaschine nach *warum ein RAID kein backup ist.*)

7.8.1 Einstieg mit `storeBackups Replication Wizard`

Dieses Beispiel ist für diejenigen gedacht, die zumindest etwas Erfahrung mit `storeBackup` und schon einige Backups erstellt haben. Falls Du keinerlei Erfahrung mit `storeBackup` hast, lerne zunächst, wie man ein Master-Backup erstellt, bevor Du die Replikation desselben planst. Für erfahrende Benutzer erklären spätere Kapitel dieses Dokumentes alle Details, die in dieser Kurzanleitung nicht erwähnt werden.

`StoreBackups Replication Wizard` (`storeBackupReplicationWizard.pl`) erstellt Dir eine Konfiguration, mit der Du direkt anfangen kannst, `storeBackups` Möglichkeiten der Replikation für das typische aller Szenarien zu nutzen. `StoreBackups Replication Wizard` ist ein interaktives Programm. Es führt viele Konsistenzprüfungen durch und fragt wenn nötig nach.

Der `Replication Wizard` erzeugt drei Konfigurationsdateien. Du kannst die Replikation selbstverständlich auch ohne `storeBackups Replication Wizard` aufsetzen und Du bekommst in den folgenden Kapiteln alle Informationen, die dazu nötig sind. Wie auch immer – in diesem Abschnitt wird erläutert, wie man die Replikation so schnell und einfach wie möglich verwenden kann (das unter der Annahme, dass Du ein typisches Szenario verwendest).

Falls Du Deine Backups im Moment *nicht* mit der Option `lateLinks` laufen lässt und die Replikation verwenden willst, *musst Du für Sicherungen mit `storeBackup.pl` die Option `lateLinks` einschalten*. Siehe Kapitel 7.8.5 für weitere Informationen. In diesem Beispiel siehst Du auch, dass die Option `lateLinks` durch den Wizard richtig gesetzt wird.

Jetzt und im Folgenden musst Du `lateLinks=yes` in der Konfigurationsdatei setzen und diese zwei Kommandos ausführen (die in ein kleines Skript gepackt werden sollten):

```
1 # storeBackup.pl -f stbu.config
2 # storeBackupUpdateBackup.pl -b <dirOfMasterBackup>
```

In dem Beispiel weiter unter ist `<dirOfMasterBackup>` `/masterBackup`; verwende aber die zu Deinem Zweck passenden Verzeichnisse. Verwende ebenso den aktuellen Namen Deiner Konfigurationsdatei anstelle von `stbu.config`.

Das Beispiel geht davon aus, dass folgende vier unterschiedliche Verzeichnisse verwendet werden:

1. `/home`, das Du sichern willst
2. `/masterBackup`, wo Dein Master-Backup steht
3. `/extDisk/backupCopy`, der Ort, zu dem Du Dein Master Backup replizieren willst (dies wird Deine Backup-Kopie).
4. `/deltaCache` ist die Stelle, an der die zu replizierenden Deltas zwischengespeichert werden, bis sie zur Backup-Kopie (die sich in `/extDisk/backupCopy` befindet) repliziert werden.

Du benötigst Schreibrechte in allen diese Verzeichnissen.

Weiterhin gehen wir davon aus, dass die zu replizierende Backup-Serie `homeBackup` heißt. Für weitergehende Informationen über Backup Serien, siehe Kapitel 3, Erste Schritte. In diesem Beispiel wird ebenfalls angenommen, dass Du schon Backups in Deinem Master-Backup hast. Wir werden die Backups der Serie `homeBackup` daraus zum Replikationsziel kopieren und den Replikationsprozess aufsetzen. Falls Du noch keine Backups erzeugt hast, siehe in anderen Beispielen wie in Kapitel 7.8.4 nach (oder erzeuge ein Backup und mach hier weiter).

Fangen wir an:

1. Kopiere die bestehenden Backups, um eine Basis für die Replikation zu schaffen. Das kann einige Zeit dauern:

```
# linkToDirs.pl /masterBackup/homeBackup -t /extDisk/backupCopy
```

2. Als nächstes sehen wir kurz auf die Hilfe des Replication Wizards:

```
1 storeBackupReplicationWizard.pl -h
```

3. Nun starte den Replication Wizard, wobei Du ihm die Verzeichnisse des Master-Backups, des `deltaCaches` und den Ort, an den Du die replizierten Daten haben willst, angeben musst. Keines dieser drei Verzeichnisse darf ein Unterverzeichnis eines anderen sein. Siehe Kapitel 7.8.3, Grundlegende Konzepte vor Verwendung der Replikation von `storeBackup` für weitere Informationen.

```
1 storeBackupReplicationWizard.pl -S homeBackup -m /masterBackup -c /extDisk/backupCopy/ -d  
↪ /deltaCache
```

(oder

```
1 storeBackupReplicationWizard.pl --series homeBackup --masterBackupDir /masterBackup  
↪ --backupCopyDir /extDisk/backupCopy/ --deltaCacheDir /deltaCache
```

)

4. An dieser Stelle kannst Du – wenn Du willst – den Inhalt der drei replikationsbezogenen Konfigurationsdateien ansehen. (Du findest sie mit der Endung `.config` in `/masterBackup`, `/deltaCache` und `/extDisk/backupCopy`.) Das geht z. B. so:

```
1 cat /masterBackup/storeBackupBaseTree.conf  
2 cat /deltaCache/deltaCache.conf
```

5. Jetzt kannst du Dein erstes Backup mit Replikation laufen lassen:

```
# storeBackup.pl -s /home -b /masterBackup -S homeBackup --lateLinks 0:homeBackup
```

Dieses Kommando erzeugt ein Backup in `/masterBackup`. Wenn Du dort nachschaust, siehst Du die relativ zum letzten Backup geänderten Dateien (das Delta) sowie die Kommandodatei³⁴, die die Informationen darüber enthält, was zur Vervollständigung des Backups noch getan werden muss. Der letzte Parameter (`0:homeBackup`) stellt sicher, dass nur Hardlinks zum vorherigen Backup derselben Serie gemacht werden. Da wir nur diese eine Serie replizieren wollen, ist es nicht möglich, Querverweise zu anderen Serien zu machen! (Diese Konfiguration ist nur notwendig, wenn sich mehrere Backup-Serien in Deinem Master-Backup befinden.)

6. Als nächstes kopierst Du die Deltas zu dem Platz (`/deltaCache`), an dem sie aufbewahrt werden, bis Du das externe Laufwerk anschließt und die Deltas dorthin replizierst. In diesem Schritt komplettierst Du gleichzeitig das gerade erstellte Backup im Master-Backup-Verzeichnis. Das folgende Kommando liest dazu die Konfigurationsdatei `/masterBackup/storeBackupBaseTree.conf`:

```
# storeBackupUpdateBackup.pl -b /masterBackup
```

³⁴Das ist die Datei `.storeBackupLinks/linkFile.bz2` im Basisverzeichnis des Backups, das Du gerade erzeugt hast.

7. Nun kannst Du die Replikation durch Fertigstellung des kopierten Backups beenden:

```
# storeBackupUpdateBackup.pl -b /extDisk/backupCopy
```

Sieh Dir auch die Backup-Kopie `/extDisk/backupCopy` an – es ist jetzt eine vollständige Kopie. Sieh auch in `/deltaCache`. Das Delta wurde nach `/deltaCache/processedBackups` verschoben.³⁵

Nachdem Du Deine Umgebung (mit dem Wizard in den Schritten oben) aufgesetzt hast, solltest Du in Zukunft folgendermaßen vorgehen:

- Lass Deine Backups mit `storeBackup.pl` so laufen wie Du willst – aber verwende die Option `lateLinks` (und begrenze das Hardlinken (die Referenzen) auf die Serien, die Du replizieren willst).³⁶
- starte

```
# storeBackupUpdateBackup.pl -b /masterBackup
```

um Deine Backups fertigzustellen und die Deltas in den Delta-Cache zu kopieren. Mach das am besten direkt nach dem Lauf von `storeBackup.pl`. Wenn Du einen Server hast, ist es am einfachsten, diese Kommando nachts mittels cron zu starten. Du brauchst die Option `--autorepair` wahrscheinlich nicht.
- Verbinde Deine externe Platte zu einem Dir genehmen Zeitpunkt (z. B. einmal in der Woche), mounte diese externe Platte nach `/extDisk` so das der Pfad `/extDisk/backupCopy` ist. Starte:

```
# storeBackupUpdateBackup.pl -b /extDisk/backupCopy
```

Wenn das Kommando durchgelaufen ist, umounte die Platte („Einbindung lösen“) und trenne Sie von Stromnetz und Computer. Du brauchst die Option `--autorepair` wahrscheinlich nicht.

Fragen, die in diesem Kapitel „Replikation von Backups“ nicht adressiert wurden, werden weiter unter behandelt.

7.8.2 Warum das Kopieren von Backups kein Ersatz für die Replikation ist

Wir werden die Diskussion durch ein Beispiel etwas weniger abstrakt machen. Eine typische Backup-Strategie ist, dass Du Deine Backups jeden Tag in das Master-Backup laufen lässt (eventuell automatisch mit cron) und diese Daten periodisch außerhalb des Gebäudes (oder zumindest auf ein anderes physisches Gerät) überträgst. Lass uns zur Veranschaulichung annehmen, dass Du eine separate Festplatte außer Haus lagerst und sie wöchentlich aktualisierst.³⁷ Einmal in der Woche fügst Du die neuen Backups im Master-Backup zu den Kopien auf der externen (außerhäusigen) Platte hinzu, die Du dazu temporär holst und mit Deinem Rechner verbindest.

Das neue Backup einfach mit „`cp -a`“ zu kopieren ist eine schlechte Idee, weil die neu kopierten Verzeichnisse (Backups) nicht mit den auf der externen Platte befindlichen verlinkt würden.³⁸ Du kannst `linkToDirs.pl` verwenden, um die neuen Backups im Master-Backup mit den vorhandenen in der Backup-Kopie zu verlinken (und zu kopieren). Das Tool `linkToDirs.pl` ist praktisch für ad hoc Replikationen, aber nicht das beste für geplante und automatisierte.

Eine andere übliche Methode, die neuen Backups zu kopieren, ist, ein Synchronisationstool wie `rsync` zu verwenden. Dabei ergeben sich zwei Probleme: Erstens dauert es *sehr* lange, wenn Du viele Backups in Deinem Master-Backup hast und zweitens synchronisiert Du jeden Fehler im Master-Backup mit, und das ist bestimmt nicht das, was Du willst. Stell Dir vor, die Platte für den Master-Backup bekommt irgendwann einen Blockfehler in einer Datei aus einem Backup von vor einem Monat. Wenn Du nun mit z. B. `rsync` synchronisierst, wirst Du die defekte Datei kopieren. Im schlimmsten aller Fälle könntest Du Dir das ganze Backup zerstören (ohne zusätzliche Sicherheit zu bekommen). Wenn du die Replikation von `storeBackup` verwendest, haben alte Dateien nichts mit der Replikation zu tun.

³⁵Die Deltas werden aus Gründen der Datenverfügbarkeit nicht direkt gelöscht. Du kannst die Aufbewahrungszeit mit den Optionen `--archiveDurationDeltaCache` und `--dontDelInDeltaCache` des Programms `tt storeBackupUpdateBackup.pl` beeinflussen.

³⁶Diese Einschränkung könnte in Zukunft entfallen.

³⁷Wenn Du eine kontinuierliche Replikation an einem anderen Standort aufbauen willst – das geht mit `storeBackup` auch. Aber für diese Erläuterungen nehmen wir eine einzelne externe Platte an, da dies eine übliche und angemessene Strategie ist. Die verwendeten Mechanismen sind in beiden Fällen dieselben. Du kannst auch mehr als eine externe Platte verwenden. Angenommen, Du hast zwei externe Kopien und replizierst die Backups alternierend jede Woche auf sie. Du kannst hierfür dieselben Mechanismen wie beschrieben verwenden. Der einzige Unterschied ist, beide Platten in den Konfigurationsdateien bekannt zu machen und jede Woche eine andere anzuschließen.

³⁸Das bedeutet, dass Du *viel* mehr Platz für Deine Backups benötigst als im Master-Backup.

ABER HALT: Was, wenn die *neu kopierten* Daten defekt sind, weil einige Sektoren auf der Platte reproduzierbar kaputt sind oder defekter RAM (oder etwas anderes) dazu führt, dass die Daten schon im Master-Backup defekt ankommen? Wirst Du je merken, dass (Teile von) Deinen Daten im Backup defekt sind? Das Backup Programm `storeBackup.pl` wird Dir dasselbe mitteilen wie `rsync` – nämlich nichts, weil das nicht unter ihrer Kontrolle ist. Aus diesem Grund solltest Du `storeBackupCheckBackup.pl`, das die Prüfsummen für alle Dateien nachrechnet, regelmäßig verwenden. Durch das Kontrollieren mit diesem Programm kannst Du Fehler in alten Backups erkennen, die Du manuell korrigieren kannst. Und Du kannst frühzeitig feststellen, ob neue Backups defekt sind. Daher empfehlen wir, `storeBackupCheckBackup.pl` bei neuen Backups wöchentlich und bei älteren alle paar Monate laufen zu lassen (was sehr lange dauern kann).

Wenn Du Fehler auf einer Platte entdeckst, solltest Du Dir das Problem genauer ansehen und nicht zögern, die Platte auszutauschen.

Die der Replikation von `storeBackup` zugrunde liegende Idee ist, die oben beschriebenen Punkte anzugehen. Eine Replikation bedeutet, dass wir denselben Zustand in zwei Lokationen haben wollen (z. B. im Master-Backup und in der Backup-Kopie). Das ist das, was wir in dem Beispiel oben mit dem Kommando `cp -a` getan haben. Nehmen wir an, das war das Kommando vom Montag. Nach einem Tag haben wir im Master-Backup eine Änderung (das neue Backup vom Dienstag). Für die Replikation benötigen wir nur die *Unterschiede* zwischen dem Backup vom Montag und dem vom Dienstag. Wenn wir einen cleveren Algorithmus hätten, der diese Unterschiede feststellt, könnten wir sie zur externen Platte transportieren und dort so ein neues Vollbackup (mit allen Links, Berechtigungen usw.) erstellen. Als Ergebnis enthielte das Master-Backup auf der externen Platte genau dieselben Daten wie das Master-Backup.

Wenn wir die externe Platte einmal pro Woche anschließen³⁹ benötigen wir einen Platz, um die Unterschiede zu den vorherigen Sicherungen zu speichern. Wir bekommen diese Deltas von Montag auf Dienstag, Dienstag auf Mittwoch usw. Wir erzeugen dann die Vollbackups auf der externen Platte mit der Backup-Kopie, z. B.:

- Backup von Dienstag → erzeugt vom Vollbackup von Montag plus (Deltas vom Montag auf Dienstag)
- Backup von Mittwoch → erzeugt vom Vollbackup von Dienstag plus (Deltas vom Dienstag auf Mittwoch)
- Backup von Donnerstag → erzeugt vom Vollbackup von Mittwoch plus (Deltas vom Mittwoch auf Donnerstag)
- ...

Das bedeutet, dass wir die Deltas zwischen zwei aufeinander folgenden Backups im Master-Backup benötigen. Im Prinzip gibt es zwei Wege, diese zu ermitteln:

1. Berechnung der Unterschiede, also so etwas wie eine „inverse Deduplikation“ (`storeBackup` sucht nach Dateien (oder Teilen davon) mit identischem Inhalt und verhardlinkt diese).
2. Identifikation der Unterschiede direkt da, wo das Backup erzeugt wird. An dieser Stelle werden die Unterschiede ermittelt und sind bekannt.

Der zweite Weg ist der für Replikationen typischerweise verwendete, z. B. bei Replikationen von Datenbanken oder LDAP. `StoreBackup` verwendet ebenfalls diesen Weg um Replikationen zu erzeugen.

`StoreBackup` generiert Deltas zu (einem oder mehreren) existierenden Backups mit der Option `lateLinks` und speichert sie in einem „Delta Cache“.⁴⁰ (siehe Kapitel 7.6 für weiterführende Informationen dazu, wie `lateLinks` zu konfigurieren ist).

Die Replikations-Funktionalität von `storeBackup` bietet die folgenden Features:

- Die Replikation von Backups baut auf `storeBackups` existierender Möglichkeit auf, die Deltas zu vorherigen Backups plus die notwendigen Information für ein Vollbackup zu generieren.

³⁹oder wir replizieren zu einer anderen (Online) Lokation ohne direktes Routing zwischen dem Master-Backup und der Backup-Kopie

⁴⁰Der Delta Cache wird nur verwendet, wenn die Replikationsfunktionalität verwendet wird.

- Du kannst konfigurieren, welche Serien repliziert werden sollen. Im Master-Backup muss die Anzahl der durch Replikation zu erzeugenden Kopien nicht bekannt sein. Dies ermöglicht die Entkopplung von Quelle (Master-Backup) und Ziel (Backup-Kopie).
- Die Replikation von Backups kann vollständig automatisiert werden. In Kombination mit einigen Konfigurationsdateien, die für die Replikation benötigt werden, kannst Du einfach cron (oder ähnliches) verwenden, um `storeBackupUpdateBackup.pl`, siehe Kapitel 6.3 laufen zu lassen. Du kannst auch `storeBackupDel.pl`, siehe Kapitel 6.10 verwenden, um (sehr) alte Backups in der Replikation automatisch zu löschen.
- Speichermedien mit Replikationen müssen nicht ständig angeschlossen sein. Du kannst entscheiden, ob eine oder mehrere der Replikationen permanent oder nicht permanent (so dass manueller Eingriff notwendig ist) erreichbar sind. Dies kann vom Ort abhängen, an den Du replizierst (extern über WAN oder über LAN) sowie von Deiner Backup-Strategie.
- Aus Sicherheitsgründen ist es möglich, die Replikation so aufzusetzen, dass das Master-Backup nicht direkt von den Kopien erreichbar ist (und umgekehrt).
- Die Replikation erfolgt asynchron. Das bedeutet, Du kannst Deine disk1 für Backup-Kopie Nummer 1 an geraden Wochenenden und Backup-Kopie Nummer 2 auf disk2 an ungeraden Wochenenden anschließen. Aber das Resultat sind dieselben Backups auf beiden Platten; *unabhängig von Deinen Löschroutinen für jedes Backup, inklusive Master-Backup*.
- Das Löschen von alten Backups auf den Backup-Kopien und dem Master-Backup kann mit demselben Tool erfolgen (`storeBackupDel.pl`, siehe Kapitel 6.10). Insbesondere können auch unterschiedliche Löschroutinen verfolgt werden, z. B. könntest Du sehr lang zurückreichende Backups auf Deinem (langsameren) Replikationsmedien vorhalten.
- Replikationen verhalten sich wie normale Sicherungen von `storeBackup`. Du kannst natürlich auch eine Überprüfung des Backups mit `storeBackupCheckBackup.pl` (siehe Kapitel 6.12) auf den Replikationen laufen lassen.
- Die Implementierung ist robust. Wenn irgendetwas mit der Replikation schief läuft, (also Daten auf dem Weg zum Replikations-Backup aus welchen Gründen auch immer verloren gehen), kannst Du `linkToDirs.pl` verwenden, um einen neuen identischen Status (Backup Versionen) zu erzeugen und mit der Replikation fortzufahren. Du kannst das *ohne* alles (vom Master-Backup oder einer anderen Replikation) kopieren zu müssen. (Aber Du benötigst hierfür eine direkte IP Verbindung zwischen den betroffenen Backups / Replikationen während dieser Zeit oder Du benötigst eine Zwischenkopie). Ein weiterer Vorteil dieses Vorgehens ist, dass Du möglicherweise existierende Fehler in älteren Backups nicht von einem Backup zum anderen transportierst, wie es möglicherweise bei Verwendung von Standard-Synchronisationstools passieren würde.

Kurz gesagt, Du machst ein „normales“ Backup (ohne Replikation) mit `storeBackup.pl`. Du hast typischerweise eine Stelle, an der dieses Backup gespeichert wird (siehe Option `backupDir`). Dieses Backup wird als Master-Backup bezeichnet. Wenn die Platte (oder genauer das Dateisystem für dieses „Master-Backup“) ausfällt, verlierst Du Dein Backup und damit die *Historie Deiner Daten*. Es ist eine Art des Datenverlustes, die mit `storeBackups` Replikationsfeature vermieden werden kann.

7.8.3 Grundlegende Konzepte vor Verwendung der Replikation von `storeBackup`

Im vorherigen Kapitel wurden die Haupteigenschaften von `storeBackups` Replikationsfunktionalität aufgeführt. In diesem Kapitel zeigen wir ein einfaches und typisches Beispiel der Konfigurationen, die nötig sind, um Deine Backupdaten auf eine andere Platte (oder in eine andere Lokation) zu replizieren. (In Kapitel 7.8.6, sehen wir noch ein fortgeschrittenes Beispiel.)

Aber *zuerst* solltest Du einige wichtige Konventionen und Konzepte der Replikationsfunktion von `storeBackup` kennen. Bei dieser Replikationsfunktionalität gibt es vier wichtige Speicherorte, deren Konzept Du verstanden haben musst. Diese vier Speicherorte sind normale Verzeichnisbäume.⁴¹

⁴¹In der Praxis können die vier konzeptuellen Speicherorte mehr als vier physische Stellen werden, weil die Replikation nicht auf eine einzige Kopie beschränkt ist.

Von diesen vier konzeptbedingten Stellen ist eine die mit den zu sichernden Dateien. Die anderen drei haben mit der Replikation zu tun:

1. „Master-Backup“⁴²
2. „backup copy“⁴³
3. „deltaCache“.

Keines dieser drei Verzeichnisse darf ein Unterverzeichnis eines anderen sein. Sie sind separate Verzeichnisbäume.⁴⁴

Das was wir hier „Master-Backup“ nennen, ist etwas, mit dem Du schon vertraut bist, falls Du irgendeine Art von Backup gemacht hast.⁴⁵

Der nächste wichtige Speicherort für Replikationen ist die Backup-Kopie. Es ist offensichtlich – das ist der Ort, an den repliziert wird.⁴⁶

Der letzte der wichtigen Speicherorte für die Replikation ist der Cache für die Deltas (und der Metadaten), der von `storeBackup` dazu verwendet wird, seine Replikationsfunktionalität möglichst effizient bereitzustellen. Wir bezeichnen diesen Ort als „deltaCache“. Der `deltaCache` existiert, da es so möglich ist, das Master-Backup (inklusive Hardlinks) unabhängig von den Backup-Kopien zu vervollständigen.

Ein weiterer wichtiger Aspekt für das Verstehen der Replikation ist, dass jedes dieser drei Backup-bezogenen Verzeichnisse seine eigene Konfigurationsdatei im Root-Verzeichnis⁴⁷ des jeweiligen Verzeichnisbaumes haben muss. Der Grund hierfür ist, dass auf diese Art mit den fixen Lokationen für die Konfigurationsdateien die Bearbeitung ohne zusätzliche Optionen (oder Problemen aus der daraus resultierenden Komplexität) mittels `storeBackupUpdateBackup.pl` verwaltet werden kann.

Bei einer Replikation mit `storeBackup` ist der Datenfluss immer: `masterBackup` → `deltaCache` → Backup-Kopie(n).

1. „Master-Backup“ enthält seine eigene, einmalige `storeBackupBaseTree.conf`
2. jedes „backup copy“ Directory tree enthält sein eigenes, einmaliges `storeBackupBaseTree.conf`
3. „deltaCache“ enthält `deltaCache.conf`

Das „Master-Backup“ Directory enthält eine Datei namens `storeBackupBaseTree.conf`. Diese Konfigurationsdatei definiert, welche Backup-Serien zu `deltaCache` kopiert werden.

Jedes „backup copy“ Directory enthält eine Datei namens `storeBackupBaseTree.conf`, welches eine individuelle Konfiguration für die jeweilige Kopie ist. Sie definiert, welche Backup-Serien zu der jeweiligen Backup-Kopie kopiert werden müssen.

Das „deltaCache“-Verzeichnis enthält `deltaCache.conf` im Wurzelverzeichnis des Baumes. Der Sinn dieser Konfigurationsdatei ist, eine zentrale Stelle zu haben, die genau festlegt, welche Backup-Serie in welche Backup-Kopie übertragen werden soll (physische Pfade werden hierzu nicht benötigt). Diese Informationen werden von `storeBackupUpdateBackup.pl` verwendet, um zu entscheiden, ob das Delta eines Backups als verarbeitet registriert und später gelöscht werden kann. `storeBackupUpdateBackup.pl` muss wissen, wohin ein Backup kopiert werden soll und welche bereits kopiert wurden.

Diese Konfigurationsdateien enthalten einige Optionen (z. B. `backupTreeName`) für die Du eine eindeutige Bezeichnung wählen musst. Dieser Parameter ist lediglich der Name einer Referenz für eine andere Lokation.

⁴²Mehrere Master-Backups, die in ein und denselben Delta Cache replizieren, werden nicht unterstützt. Auch wenn es ungetestet und nicht supported ist, könnte es (mit unterschiedlichen Bezeichnungen für die Serien) funktionieren.

⁴³In realen Anwendungen können diese eine unbegrenzte Anzahl von Verzeichnissen sein.

⁴⁴Das bedeutet, dass die Verzeichnisse nicht verschachtelt sein dürfen. Keines der drei für die Replikation benötigten Verzeichnisse darf ein Unterverzeichnis eines anderen sein. Der Delta-Cache kann unterhalb des Quellverzeichnisses (`sourceDir`) liegen, falls er vom Backup über `storeBackup.pl` ausgeschlossen wird.

⁴⁵Bei `storeBackup` bedeutet ein „Master-Backup“ eine Backup-Serie (oder mehrere Backup-Serien). Die Bezeichnung „Serie“ kommt daher, weil das Verzeichnis einer Serie eine Serie von Backups (z. B. eines pro Tag) von Deinem Rechner beinhaltet. Siehe auch Kapitel 5.4.1

⁴⁶Die Replikation kann dazu verwendet werden, mehrere Kopien des Master-Backups an unterschiedliche Lokationen durchzuführen.

⁴⁷oberstes Verzeichnis

Es ist kein Pfad im Dateisystem oder ein aktueller Verzeichnisname. Es ist eine eindeutige Bezeichnung, die Du festlegen musst. Dieses wird weiter unten noch näher beleuchtet.

Es gibt keinen Informationsaustausch zwischen zwei unterschiedlichen Backup-Kopien. Sinnvoll ist das für den Heimanwender deshalb, weil die externe(n) Platte(n) für die Replikation nicht immer angeschlossen sein müssen, und für den professionellen Administrator, weil er aus Sicherheitsgründen kein Routing zwischen ihnen haben mag.

Jedoch willst Du vermutlich zum Verständnis des Gesamtkonzeptes der Replikation von `storeBackup` verstehen, wozu die Konfiguration der Replikation diese eindeutigen Backup-Bezeichnungen (die keine Verzeichnisnamen sind) benötigt. Warum wird nicht einfach der Verzeichnisname verwendet? Die Gründe, warum `storeBackup` eindeutige Bezeichner und keine Verzeichnisnamen verwendet, können an zwei Beispielen erläutert werden.

Stell Dir als erstes vor, jemand will zwei Backup-Kopien (Replikationen) auf zwei externen Platten erstellen, eine aktualisiert an geraden und eine an ungeraden Wochenenden. Angenommen, die Platten werden an demselben Mount-Pfad eingebunden. Der sinnvollste Weg, den Wechsel dieser beiden unterschiedlichen Kopien zu verwalten, ist der über diese eindeutigen Bezeichnungen. Stell Dir für dieses Beispiel vor, die eindeutigen Bezeichnungen wären `KopieA` und `KopieB`. Dies erlaubt `storeBackup` zu entscheiden, ob zu beiden repliziert wurde (kopiert und Hardlinks gesetzt), so dass es in das Unterverzeichnis `processedBackup` verschoben werden kann – auch wenn der Prozess unterbrochen wurde o.ä. Eine andere Implementierung würde auf diese Vorteile verzichten.

Ein anderes Beispiel ist ein Administrator, der zwei Replikationen durchführen will: Eine bleibt im Rechenzentrum und eine kommt in ein entferntes. Er setzt den Server im selben Rechenzentrum auf, der die Daten vom Delta-Cache über Mount-Punkte holt. Im entfernten Rechenzentrum kopiert er die Konfiguration und setzt den Server genauso auf. Die Verwendung von eindeutigen Bezeichnern (die entkoppelt vom Pfad sind) machen das Leben für den Administrator leichter.

Die Konfigurationsdatei von `deltaCache` weiß nichts über das Verzeichnis, in das die Backup-Kopie repliziert wird. Stattdessen steht in der Konfigurationsdatei nur die eindeutige Bezeichnung, was flexibler ist. Wenn Du das Verzeichnis der Backup-Kopie änderst, muss die Konfigurationsdatei vom `deltaCache` nicht geändert werden. Und, so wie in den oben beschriebenen Beispielen dargestellt, können zwei unterschiedliche Bezeichner auf denselben Pfad verweisen.

Du wirst also wahrscheinlich mindestens vier unterschiedliche Konfigurationsdateien für Deine `storeBackup`-Replikation verwenden. Diese sind die drei oben beschriebenen Dateien sowie Deine normale `storeBackup` Konfigurationsdatei.⁴⁸

Die Verwendung von Replikationen wird von zwei Optionen von `storeBackup.pl` beeinflusst: `--lateLinks` und `--otherBackupSeries`.

Wenn Du Deine Backups bisher nicht mit der Option `lateLinks` laufen lässt und die Replikation verwenden willst, *musst Du diese einschalten*. Hierdurch gibt es keinen Nachteil. Sie trennt den Prozess eines Vollbackups lediglich in zwei Schritte, ohne ansonsten irgendetwas am Ergebnis zu verändern.

Du solltest auch auf die Option `--otherBackupSeries` in der `storeBackup` Konfigurationsdatei achten bzw. auf ihre Anwendung auf der Kommandozeile (z. B. `0:homeBackup` im Beispiel weiter oben).

Wenn Du nur eine Serie Deiner Backups replizierst, ist es nicht möglich, diese mit anderen Serien zu verlinken! (Diese Restriktion trifft natürlich nur zu, wenn du mehrere Backup-Serien (von z. B. unterschiedlichen Rechnern) im Master-Backup hast. Eine replizierte Serie kann nicht auf eine andere Serie verweisen, die **nicht** in dieselbe Backup-Kopie repliziert wird. (Umgekehrt kann aber eine Serie, die nicht repliziert wird, auf beliebige Serien, die repliziert werden, verweisen.)

Diese Restriktion könnte in Zukunft entfallen. (Das bedeutet, dass die nicht referenzierbaren Dateien automatisch zu den Deltas (für `deltaCache`) hinzugefügt würden, wenn `storeBackupUpdateBackup.pl` im Master-Backup läuft.

Kurz gesagt, stelle für die ersten Replikationen sicher, dass nur Hardlinks auf die **eigene Serie** erzeugt werden. Alles, was im Master-Backup verlinkt wird, muss auch in jeder Backup-Kopie vorhanden sein. Wenn Du alle Serien replizierst, brauchst Du an der Hardlink-Konfiguration nichts zu ändern.

Es ist alles sehr einfach, aber nur wenn Du verstehst, was passiert. (Und natürlich ist die Konfiguration

⁴⁸Du kannst hier auch Optionen auf der Kommandozeile verwenden, aber das ist komplizierter.

etwas komplizierter, wenn Du unterschiedliche Serien (überlappend) auf unterschiedliche Backup-Kopien spielst.)

Wenn `storeBackupUpdateBackup.pl` auf den Backup-Kopien läuft, wird `autorepair` automatisch eingeschaltet (generiert aber nur `INFO` und keine `ERROR` Meldungen).

7.8.4 Den Replication Wizard über ein Beispiel verstehen

Falls Du das Kapitel „Einstieg mit storeBackups Replication Wizard“ gelesen hast, musst Du dieses Kapitel nicht unbedingt im Detail lesen. Dieses Beispiel ist nützlich, wenn Deine Anforderungen atypisch sind (das heißt das genannte Kapitel war nicht anwendbar) und Du Dich daher näher mit dem Replication Wizard beschäftigen solltest. Dieses Kapitel wird Dir Details erläutern, so dass Du komplexere Konfigurationen erstellen kannst.

Der Replication Wizard erzeugt drei Konfigurationsdateien. Dieses einfache Beispiel wird Dir helfen, diese Dateien sowie die Arbeitsweise des Replication Wizards zu verstehen.

Das folgende kommandozeilenbasierte Beispiel zeigt den Replication Wizard und demonstriert ein vollständig durchgeführtes Backup und eine Replikation unter Verwendung einer temporären Installation. Um das Beispiel einfach zu halten, verwenden wir die `default`-Serie (siehe Kapitel 3, Erste Schritte für weitergehende Informationen über Backup Serien).

Die Dateien, die Du sicherst, befinden sich in `/tmp/repliTest/localDisk/sourceFiles`. Das Backup kommt nach `/tmp/repliTest/externalDisk_1/masterBup`. Die eigentlichen Replikationen kommen in das Verzeichnis `/tmp/repliTest/externalDisk_2/copyBup`. Eigentlich sollte das replizierte Backup auf einer anderen physischen Platte, also z. B. auf einer USB-Harddisk oder einem anderen Server liegen. In dieser Hinsicht unterscheidet sich dieses Beispiel von dem, was Du bei Deinem realen Szenario tun solltest. (Weiter unterscheidet es sich in: 1) Wir sichern nur ein paar Dateien, 2) wir verwenden keine Konfigurationsdatei für `storeBackup.pl` und 3) wir kopieren keine existierende Backups als Initialisierung zum Delta Cache.)

Zuerst erzeugen wir temporär einige Dateien, die wir dann sichern können. Die Inhalte sind nicht wichtig – dies ist nur ein Beispiel:

```
1 mkdir -p /tmp/repliTest/localDisk/sourceFiles /tmp/repliTest/localDisk/deltaCache
  ↪ /tmp/repliTest/externalDisk_1/masterBup /tmp/repliTest/externalDisk_2/copyBup
2 cd /tmp/repliTest/
3 cp /bin/ls /tmp/repliTest/localDisk/sourceFiles
4 touch /tmp/repliTest/localDisk/sourceFiles/test.txt
5 ls -la /tmp/repliTest/localDisk/sourceFiles
```

Wir gehen davon aus, dass `storeBackup/bin` in Deinem Pfad ist. Wenn nicht, erzeuge symbolische Links wie in Kapitel 1 Erste Schritte beschrieben. Wenn nötig, starte in einem Terminal diese zwei Kommandos (die zweite Zeile endet mit: Leerzeichen Punkt):

```
1 cd /usr/local/bin
2 ln -s /opt/storeBackup/bin/* .
3 cd -
```

Als nächstes führe ein initiales Backup mit der Option `lateLinks` aus. Das erzeugt etwas, was wir replizieren können.

```
1 storeBackup.pl -s /tmp/repliTest/localDisk/sourceFiles/ -b
  ↪ /tmp/repliTest/externalDisk_1/masterBup/ --lateLinks
```

Während des Backups solltest Du eine Warnung erhalten:

```
1 WARNING 2012.07.21 16:12:11 12580 created directory <backup//default>
```

Sieh dir als nächstes die Hilfe des Replication Wizards an:

```
1 storeBackupReplicationWizard.pl -h
```

Nun starte den Replication Wizard und übergib ihm den Pfad zum Master-Backup, zum Delta Cache und den Ort, an den das Master-Backup repliziert werden soll. Keines dieser drei Verzeichnisse darf ein Unterverzeichnis eines anderen sein. Und selbstverständlich sollte sich das Verzeichnis für die Backup-Kopie auf einem eigenen, physisch getrennten Laufwerk (extern oder einem anderen Server zugeordnet) befinden.

(Der Delta Cache kann auf derselben Platte wie das Quellverzeichnis (also das, was gesichert wird) abgelegt werden.) Siehe Kapitel 7.8.3 Grundlegende Konzepte vor Verwendung der Replikation von storeBackup für weitere Informationen.

```
1 storeBackupReplicationWizard.pl -m /tmp/repliTest/externalDisk_1/masterBup/ -c
  ↪ /tmp/repliTest/externalDisk_2/copyBup/ -d /tmp/repliTest/localDisk/deltaCache
```

Da wir die Serie nicht explizit angegeben haben, wird nachgefragt:

```
1 found series <default>
2 replicate it?
```

Antworte mit **yes** auf den Prompt und der Wizard läuft bis zum Schluss durch. (Mit der Option **--series** wäre die Nachfrage entfallen.)

Jetzt kannst du den Inhalt der drei replikationsbezogenen Konfigurationsdateien inspizieren. (Sie sind in `/tmp/repliTest/externalDisk_1/masterBup/`, `/tmp/repliTest/localDisk/deltaCache` sowie im Verzeichnis für die externe Kopie `/tmp/repliTest/externalDisk_2/copyBup` mit der Endung `.conf`.) Zum Beispiel:

```
1 cat /tmp/repliTest/externalDisk_1/masterBup/storeBackupBaseTree.conf
2 cat /tmp/repliTest/localDisk/deltaCache/deltaCache.conf
```

Du kannst Dir auch die aktuell vorhandenen Dateien ansehen, bevor das Backup repliziert ist:

```
1 find /tmp/repliTest/ -print | sort
```

Der letzte Schritt, um das Backup zu beenden, ist, `storeBackupUpdateBackup.pl` genauso aufzurufen, wie ansonsten auch bei Verwendung der Option `lateLinks`. Mit Hilfe der Konfigurationsdateien, die der Replication Wizard erzeugt hatte, replizieren (kopieren) die folgenden Schritte Dein Master-Backup in das Replikationsverzeichnis, das Du angegeben hattest. Später kannst Du die beiden Kommandos in ein Skript packen bzw. als cron Job aufrufen.⁴⁹

```
1 storeBackupUpdateBackup.pl -b /tmp/repliTest/externalDisk_1/masterBup/
2 storeBackupUpdateBackup.pl -b /tmp/repliTest/externalDisk_2/copyBup/
```

Du kannst Dir die Liste der Dateien jetzt wieder anzeigen lassen und siehst, dass die Dateien repliziert wurden.

(Beachte auch `deltaCache/processedBackups`)

```
1 find /tmp/repliTest/ -print | sort
```

Wie bereits bemerkt, ist der Replication Wizard ein interaktives Programm. Wenn du es ein wenig ausprobieren willst, verwende es mit unterschiedlichen Optionen oder mit fehlender Umgebung, z. B. mit einem fehlenden Master-Backup in einem ähnlichen Beispiel wie oben dargestellt.

7.8.5 Eine einfache Replikation ohne den Replication Wizard

Nachdem Du mit dem Replication Wizard vertraut bist, wird dieses Kapitel Dein Verständnis vertiefen, so dass Du komplexere Konfigurationen realisieren kannst. Falls Deine Replikationsanforderungen einfach und eher typisch sind, ist es nicht notwendig, dieses Kapitel zu lesen.

Wenn Du Deine Backups momentan nicht mit der Option `lateLinks` laufen lässt und die Replikation verwenden willst, *musst du die Option `lateLinks` einschalten*. Das bedeutet praktisch, dass Du vorher etwas in der Art laufen ließt:

```
1 # storeBackup.pl -f stbu.config
```

Jetzt (und in Zukunft) musst Du `lateLinks=yes` in der Konfigurationsdatei setzen und die folgenden zwei Kommandos laufen lassen (die Du in ein ausführbares Skript schreiben kannst):

```
1 # storeBackup.pl -f stbu.config
2 # storeBackupUpdateBackup.pl -b <dirOfMasterBackup>
```

⁴⁹Beim zweiten Kommando solltest Du vorher überprüfen, ob die externe Platte angeschlossen ist.

Hier hat `storeBackup.pl` alle oben beschriebenen Deltas erzeugt, die in Deinem Master-Backup gespeichert wurden. Wenn Du Dir die Daten näher ansiehst, stellst Du fest, dass keine Hardlinks auf bereits existierende Dateien im vorherigen Backup erstellt wurden. Nachdem `storeBackupUpdateBackup.pl` gelaufen ist, wurden alle „fehlenden“ Schritte durchgeführt (wie Linken, Rechte setzen). Es ist nichts anderes als das, was `storeBackup.pl` sonst als alles-in-einem Applikation in einem Schritt tut.

Das Resultat ist exakt dasselbe (ein Vollbackup), wie wenn Du `storeBackup.pl` ohne die Option `lateLinks` laufen lassen würdest. Der Batch oben ist nur ein einfaches Beispiel – `storeBackupUpdateBackup.pl` kannst Du auch später auf Deinem Server laufen lassen. Siehe Kapitel 7.6 für weitere Informationen.

Falls Du die Replikation von `storeBackup` nutzen willst, musst Du sie natürlich erst konfigurieren. Der Replication Wizard kann Dich dabei unterstützen. In diesem Kapitel machen wir das aber „zu Fuß“.

Wenn wir das einfache Beispiel der Replikation auf ein externes Laufwerk von oben aufgreifen, geht das wie folgt beschrieben. Wir gehen wieder davon aus, dass vier unterschiedliche Verzeichnisse betroffen sind:

1. `/home` was Du sichern willst
2. `/masterBackup` der Ort Deines Master-Backups
3. `/extDisk/backupCopy` der Ort, an den das Master-Backup repliziert werden soll (die Backup-Kopie)
4. `/deltaCache` der Aufbewahrungsort für die Deltas, bis sie zur Backup-Kopie (`/extDisk/backupCopy`) transportiert werden.

Des Weiteren gehen wir davon aus, dass du die Backup-Serie `homeBackup` kopieren willst. Du benötigst Schreibrechte in allen diesen Verzeichnissen (in `/home` reichen Leserechte).

1. Kopiere die schon vorhandenen Backups als Basis für die Replikation:
`# linkToDirs.pl /masterBackup/homeBackup -t /extDisk/backupCopy`
2. Nun musst Du eine Konfigurationsdatei in Deinem Master-Backup erstellen, damit `storeBackup` weiß, dass Du replizieren willst:
`# storeBackupUpdateBackup.pl --genBackupBaseTreeConf /masterBackup`
3. Editiere die generierte Konfigurationsdatei `/masterBackup/storeBackupBaseTree.conf`, so dass sie den folgenden Inhalt hat:⁵⁰

```
1 backupTreeName=myMasterBackup
2 backupType=master
3 seriesToDistribute=homeBackup
4 deltaCache=/deltaCache
```

In der Master-Backup-Konfigurationsdatei wird die Bezeichnung für `backupTreeName` nur für Error-Meldungen sowie Warnungen usw. benötigt. Sie existiert hauptsächlich, um künftige Erweiterungen abzudecken, da hierdurch sämtliche Backups über eine eindeutige Bezeichnung verfügen.

Du kannst den Parameter für die eindeutigen Bezeichner `backupTreeName` wie immer Du willst ändern. (`myMasterBackup` war im Beispiel gewählt). Aber Du musst `backupType` auf `master` setzen!

4. Jetzt kannst Du Dein erstes Backup starten, das repliziert wird:
`# storeBackup.pl -s /home -b /masterBackup -S homeBackup --lateLinks 0:homeBackup`
Dieses Kommando erzeugt ein Master-Backup in `/masterBackup`. Wenn Du nachsiehst, findest Du dort die Delta-Dateien sowie die Kommandodatei⁵¹, in der steht, was noch notwendig ist, um das Backup zu komplettieren. Der letzte Parameter (`0:homeBackup`) bewirkt, dass nur Hardlinks auf ältere Versionen derselben Backup-Serie generiert werden. Da wir nur diese eine Serie replizieren, können keine Querverweise auf andere Serien verwendet werden! (Diese Einstellung ist nur notwendig, wenn mehrere Backup-Serien in Deinem Master-Backup befinden.)
5. Der nächste Schritt ist, die Deltas nach `deltaCache` zu kopieren, wo sie für die Replikation auf die externe Platte vorgehalten werden sollen. In diesem Schritt findet ebenfalls die Vervollständigung zum Master-Backup statt. Das folgende Kommando liest die hierzu soeben angelegte Konfigurationsdatei `/masterBackup/storeBackupBaseTree.conf`:
`# storeBackupUpdateBackup.pl -b /masterBackup`

⁵⁰Die „Regeln“ für die Konfigurationsdatei sind identisch mit denen für alle anderen Konfigurationsdateien.

⁵¹Das ist `.storeBackupLinks/linkFile.bz2` im Wurzelverzeichnis des gerade erstellten Backups.

6. Jetzt musst Du die Konfigurationsdatei für den Delta-Cache generieren. Das soeben gestartete Kommando hat die Deltas von Deinem Master-Backup zum Delta-Cache kopiert. (Du solltest Dir das Verzeichnis `deltaCache` ansehen.)

```
# storeBackupUpdateBackup.pl --genDeltaCacheConf /deltaCache
```

7. Editiere die generierte Konfigurationsdatei `/deltaCache/deltaCache.conf`, so dass sie den folgenden Inhalt hat:⁵²

```
1 backupCopy0=myBackupCopy homeBackup
2 ;backupCopy1=
3 ;backupCopy2=
```

Ändere die anderen Schlüsselwörter (`backupCopy1` bis `backupCopy9`) nicht, da wir nur eine Replikation konfigurieren. (Der Delta Cache ist die zentrale Verteilstation für alle definierten Replikationen.) `myBackupCopy` ist einfach der Name (kein Pfad) für die Backup-Kopie (auf der externen Platte), zu der repliziert werden soll. Du kannst jeden Namen wählen den Du willst, aber er muss exakt derselbe wie in der Konfiguration in `/extDisk/backupCopy` sein. Nach „`myBackupCopy`“ kommt die Liste der Serien, die repliziert werden sollen. Hier ist es nur die eine Serie `homeBackup`.

8. Als nächstes musst Du der Backup-Kopie noch mitteilen, welche Daten aus dem Delta Cache zu ihr repliziert werden soll. Generiere hierzu eine Konfigurationsdatei:

```
# storeBackupUpdateBackup.pl --genBackupBaseTreeConf /extDisk/backupCopy
```

9. Editiere die Konfigurationsdatei `/extDisk/backupCopy/storeBackupBaseTree.conf`, so dass sie den folgende Inhalt hat:⁵³

```
1 backupTreeName=myBackupCopy
2 backupType=copy
3 seriesToDistribute=homeBackup
4 deltaCache=/deltaCache
```

Der Name des Backups muss derselbe sein, wie der im `deltaCache` (`/deltaCache/deltaCache.conf`) spezifizierte. Um `storeBackupUpdateBackup.pl` darüber zu informieren, dass es Deltas *vom* Delta Cache kopieren soll, muss die Option `backupType` auf `copy` gesetzt werden.⁵⁴

10. Nun kannst Du die Replikation durch Komplettierung des Backups in der Backup-Kopie beenden:

```
# storeBackupUpdateBackup.pl -b /extDisk/backupCopy
```

Schau in `/ext/backupCopy` nach. Dort befindet sich nun ein vollständiges Backup. Sieh Dir auch das Verzeichnis `/deltaCache` an. Das Backup wurde nach `/deltaCache/processedBackups` verschoben.⁵⁵

Nachdem Du die Umgebung aufgesetzt hast, machst Du einfach Folgendes:

- Starte Dein Backup mit `storeBackup.pl` wie immer Du willst – aber verwende die Option `lateLinks` (und begrenze das Hardlinken auf die Serien, die Du replizieren willst.)⁵⁶
- starte

```
# storeBackupUpdateBackup.pl -b /masterBackup
```

um Deine Backup-Kopie zu vervollständigen und die Deltas zum Delta-Cache zu kopieren. Am besten machst Du das direkt nach dem Lauf von `storeBackup.pl`. Wenn Du einen eigenen Server verwendest, ist es am einfachsten, dieses Kommando nachts per cron auf dem Server zu starten.
- Verbinde Deine externe Festplatte wann immer Du willst (z. B. einmal die Woche), mounte sie an `/extDisk`, so dass der Pfad zu Deiner Backup-Kopie `/extDisk/backupCopy` ist. Starte

```
# storeBackupUpdateBackup.pl -b /extDisk/backupCopy
```

Wenn das Kommando ausgeführt ist, ummounte das externe Laufwerk und trenne es von Deinem Rechner.

⁵²Die „Regeln“ für die Konfigurationsdatei sind so wie für alle Konfigurationsdateien.

⁵³Die „Regeln“ für die Konfigurationsdatei sind so wie für alle Konfigurationsdateien.

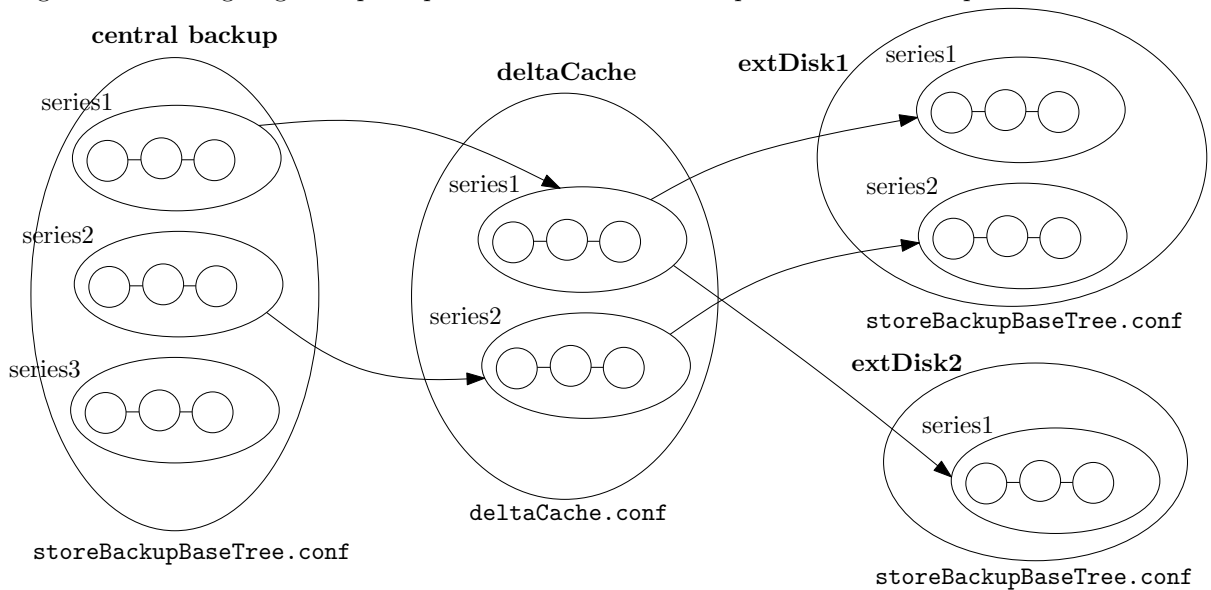
⁵⁴im Gegensatz zum Wert `master`, der `storeBackupUpdateBackup.pl` anweist, Deltas *zum* Delta Cache zu kopieren.

⁵⁵Diese Deltas werden aus Sicherheitsgründen nicht sofort gelöscht. Du kannst den Zeitraum für die Aufbewahrung dieser Daten mit den Optionen `--archiveDurationCopyStation` und `--dontDelInCopyStation` von `tt storeBackupUpdateBackup.pl` einstellen.

⁵⁶Dieses Einschränkung könnte in Zukunft entfallen.

7.8.6 Wie storeBackups Replikation funktioniert

Wenn Du eine komplexe Konfiguration erstellen willst, wird dieses Kapitel für Dich interessant sein. Die folgende Abbildung zeigt den prinzipiellen Datenfluss beim Replizieren von Backups:



Wichtig: Bevor Du mit der Replikation von Backups beginnen kannst, musst Du eine (gewöhnliche) Version mindestens eines Backups vom Master-Backup an die Stellen kopieren, an denen Deine Backup-Kopien liegen sollen. Die Replikation basiert im Wesentlichen auf inkrementellen Backups (Option `lateLinks` von `storeBackup.pl` und muss immer gegen die Backup-Version gelinkt werden, gegen die die inkrementelle Version generiert wurde! Du kannst für das Kopieren der (alten) Backups `cp -a` oder (besser) `linkToDirs.pl` verwenden.

Angenommen, Dein Master-Backup befindet sich in `/masterBackup` und Du willst die Serien `series1`, `series2` und `series3` nach `/extDisk1/stbu` kopieren (welches das oberste Verzeichnis Deiner Backup-Kopie werden soll), dann startest Du:⁵⁷

```
1 # linkToDirs.pl /masterBackup/series1 /masterBackup/series2 /masterBackup/series3 --targetDir
   ↪ /extDisk1/stbu
```

Im dargestellten Beispiel oben gibt es ein Master-Backup, in dem sämtliche Backups gespeichert werden. In diesem Master-Backup gibt es drei unterschiedliche Serien mit Namen `series1`, `series2` und `series3`. In jeder dieser drei Serien gibt es drei inkrementelle Backups (als Kreise dargestellt), die mit der Option `lateLinks` erstellt wurden und die repliziert werden sollen.

Die Konfigurationsdatei im obersten Verzeichnis des Master-Backups (`storeBackupBaseTree.conf`) ist so konfiguriert, dass zwei (`series1`, `series2`) der drei Serien auf die externen Platten konfiguriert werden sollen. Die Konfigurationsdateien können z. B. so aussehen (ohne Kommentare):

```
1 backupTreeName=myMasterBackup
2 backupType=master
3 seriesToDistribute=series1 series2
4 deltaCache=/deltaCache
```

Du kannst diesen Typ von Konfigurationsdatei folgendermaßen erzeugen:

```
storeBackupUpdateBackup.pl --genBackupBaseTreeConf directory
```

wobei `directory` das Wurzelverzeichnis Deines Master-Backups ist. Nach der Generierung passt Du die Konfigurationsdatei gemäß Deinen Anforderungen an.

Die Beispielkonfiguration oben teilt `storeBackup` (genauer `storeBackupUpdateBackup.pl`) mit, dass Du Dein Master-Backup als `myMasterBackup` (wähle hier irgendeinen eindeutigen Namen aus) bezeichnet hast und dass dieses Dein Master-Backup ist. Die Konfigurationsdatei liegt auch fest, dass `series1` und `series2` kopiert werden sollen und dass die zentrale Verteilstation, Dein „deltaCache“ im Verzeichnis `/deltaCache` ist. Nachdem die inkrementellen Backups mittels `storeBackupUpdateBackup.pl` wie oben beschrieben kopiert

⁵⁷`linkToDir.pl` ist Bestandteil von `storeBackup`

wurden, generiert derselbe Lauf des Programms die fehlenden Hardlinks (usw.) in dem Master-Backup, so dass alle Sicherungen Vollbackups werden.

In diesem Beispiel nehme ich an, dass der Server (oder allgemeiner der Rechner), der gesichert werden soll, zwei interne Platten hat. Auf / befinden sich das zu sichernde Dateisystem sowie die Anwenderdaten und auf der zweiten Platte, die unter /**backup** gemountet wird, befindet sich das Master-Backup (und ggf. Daten, die Du nicht sichern willst). Den „Delta Cache“ auf die erste Platte zu platzieren ist sinnvoll: Wenn die erste (Betriebssystem-Platte) ausfällt, kannst Du später Deine Backup-Platte verwenden. Wenn die Backup-Platte ausfällt, hast du noch die Deltas für die externe Platte auf der ersten Platte. Wenn beide gleichzeitig ausfallen, hast Du noch die externe Platte und verlierst einige Tage (falls Du keine tägliche Synchronisation machst).

Im Delta-Cache musst Du noch eine Konfigurationsdatei mit Namen **deltaCache.conf** in dessen Wurzelverzeichnis erstellen (siehe unten) und konfigurieren. Der Inhalt wäre wie folgt:

```
1 backupCopy0= 'extDisk1' series1 series2
2 backupCopy1= 'extDisk2' series1
3 ;backupCopy2=
4 ;backupCopy3=
5 ;backupCopy4=
6 ;backupCopy5=
7 ;backupCopy6=
8 ;backupCopy7=
9 ;backupCopy8=
10 ;backupCopy9=
```

Du musst zwei Ziele für das Kopieren angeben, weil wir in diesem Beispiel zwei Kopien erstellen wollen. Die erste Zeile (**backupCopy0**) in der Konfigurationsdatei oben definiert, dass **series1** und **series2** in die Backup-Kopie mit dem eindeutigen Bezeichner **extDisk1** kopiert werden sollen. Die zweite Zeile legt fest, dass **series1** in die Backup-Kopie mit dem eindeutigen Bezeichner **extDisk2** kopiert werden soll.

Du kannst die soeben genannte Konfigurationsdatei folgendermaßen erzeugen:

```
storeBackupUpdateBackup.pl --genCopyStationConf directory
```

wobei *directory* das Wurzelverzeichnis des Delta-Caches ist. Editiere die Konfigurationsdatei nach Erstellung gemäß Deinen Anforderungen.

Schlussendlich muss eine Konfigurationsdatei für das Verzeichnis erstellt werden, in das Du Deine Daten replizieren willst. Weil Du (in diesem Beispiel) zu zwei unterschiedlichen Speichersystemen replizieren willst (**extDisk1** und **extDisk2**) musst Du zwei unterschiedliche Konfigurationsdateien in den jeweiligen Wurzelverzeichnissen erstellen:

```
storeBackupUpdateBackup.pl --genBackupBaseTreeConf directory
```

Hierbei ist *directory* das Wurzelverzeichnis Deiner Backup-Kopie. Die generierte Konfigurationsdatei mit dem Namen **storeBackupBaseTree.conf** wird dann in jenem Wurzelverzeichnis gespeichert. Editiere die Konfigurationsdatei nach der Generierung passend zu Deinen Gegebenheiten.

Eventuell möchtest Du zusätzliche Optionen mit **storeBackupUpdateBackup.pl** verwenden, z. B.:

```
storeBackupUpdateBackup.pl --progressReport 200 --archiveDurationCopyStation 32d -b directory
```

Das bedeutet, dass über Fortschritte bei der Bearbeitung des Backups berichtet wird und dass alte Backups aus dem Delta Cache nach etwa einem Monat (32 Tage) gelöscht werden; *aber nur, wenn sie an die Backup-Kopien geliefert und erfolgreich mit Hardlinks versehen wurden.*

Letztendlich musst Du den letzten Schritt noch wiederholen und eine Konfigurationsdatei für **extDisk2** konfigurieren:

```
1 backupTreeName=extDisk2
2 backupType=copy
3 seriesToDistribute= series1
4 deltaCache=/deltaCache
```

Das war's schon. Nun wird jeder Start von `storeBackupUpdateBackup.pl` auf dem Master-Backup die erforderlichen Deltas (also Backups, die mit `lateLinks` erzeugt wurden) in den Delta Cache kopieren und die Backups im Master-Backup zu Vollbackups komplettieren. Wenn `storeBackupUpdateBackup.pl` auf den Backup-Kopien läuft, werden die Deltas aus dem Delta-Cache geholt und die Backup-Kopien zu vollständigen Backups verlinkt. Weiterhin werden diese Aufrufe die Delta-Backups im Delta-Cache löschen – die Details können dabei mit Hilfe der Option `--archiveDurationCopyStation` festgelegt werden.

7.8.7 Verwendung von Wildcards für die Replikation

Falls Du Wildcards für die Konfiguration der Replikation verwenden willst, solltest Du deren zugrundeliegenden Prinzipien verstanden haben, bevor Du mit diesem Kapitel fortfährst.

Konfiguration der Hardlink Option `otherBackupSeries` mittels Wildcards

Um die Ergebnisse der Verwendung von Wildcards einfach und transparent darzustellen, schreibt `storeBackup.pl` die Ergebnisse der Wildcard-Expansion in die Log-Dateien.

Stell dir vor, Du willst alle Serien, die im zweiten Schritt repliziert werden sollen per Hard Link verknüpfen. Um dieses einfach konfigurieren zu können, sollen alle Serien *aufßer* die, deren Name mit `.norepl` endet, repliziert werden. Ich nehme dabei an, dass der Name jeder Serie dem Namen des Servers entspricht (hier: `server1` und `server2`). Die Namen der Serien sind daher einfach `server1` und `server2`, die Namen der Serien, die *nicht* repliziert werden sollen sind `server1.norepl` und `server2.norepl`. Du willst, dass alle Serien miteinander verlinkt werden, aber natürlich nicht, dass eine zu replizierende Serie in eine nicht zu replizierende Serie verlinkt wird. Das würde in der Replikation zu einer Fehlermeldung führen. Du kannst für den beschriebenen Anwendungsfall die folgende Konfiguration in den Konfigurationsdateien verwenden:

```
für server1 → otherBackupSeries = 0:* -:*.norepl
für server2 → otherBackupSeries = 0:* -:*.norepl
für server1.norepl → otherBackupSeries = 0:*
für server2.norepl → otherBackupSeries = 0:*
```

Alternativ kannst du die folgende Syntax verwenden:

```
für server1 → otherBackupSeries = +0:* -:*.norepl
für server2 → otherBackupSeries = +0:* -:*.norepl
für server1.norepl → otherBackupSeries = +0:*
für server2.norepl → otherBackupSeries = +0:*
```

Das „+“ Zeichen ist optional. Du kannst Plus und Minus Zeichen auch ohne Wildcards verwenden, aber insbesondere das Minus Zeichen macht dann nicht viel Sinn.⁵⁸

Starten von `storeBackup.pl` (hier wird beispielhaft die Kommandozeile verwendet) ergibt die folgenden Log-Meldungen:⁵⁹

```
1 $ storeBackup.pl -s s -b b -S server1 '0:*' -- '-:*.norepl'
2 ....
3 INFO      2014.02.22 10:02:20 6822 consider series <*>:
4 INFO      2014.02.22 10:02:20 6822     consider series <server1>
5 INFO      2014.02.22 10:02:20 6822     consider series <server1.norepl>
6 INFO      2014.02.22 10:02:20 6822     consider series <server2>
7 INFO      2014.02.22 10:02:20 6822     consider series <server2.norepl>
8 INFO      2014.02.22 10:02:20 6822 avoid series <*.norepl>:
9 INFO      2014.02.22 10:02:20 6822     avoid series <server1.norepl>
10 INFO     2014.02.22 10:02:20 6822     avoid series <server2.norepl>
11 INFO     2014.02.22 10:02:20 6822 resulting series to hard link
12 INFO     2014.02.22 10:02:20 6822     series <server1>
13 INFO     2014.02.22 10:02:20 6822     series <server2>
14 .....
```

Der einzige Unterschied zwischen der Konfiguration (siehe z. B. `server1` und `server1.norepl`) der beiden unterschiedlichen Fälle ist „-:*.norepl“. Wenn Du die Konfigurationsdateien generieren willst, musst Du zwischen den Serien, die repliziert werden sollen und denen, die nicht repliziert werden sollen, unterscheiden.

⁵⁸ Wenn Du das Minus Zeichen auf der Kommandozeile verwendest, vergiss nicht es mit `--` zu maskieren, damit `storeBackup` erkennt, dass es keine Option ist!

⁵⁹ Die Series-Verzeichnisse müssen vor dem Starten des Kommandos bereits existieren, damit die Wildcards expandiert werden können!

Aber wenn die Konfigurationsdateien erzeugt werden, muss Du wissen, wozu sind da sind – demzufolge sollte das kein Problem sein. Der Vorteil der Verwendung von Wildcards ist die Möglichkeit, Serien zu gruppieren ohne die Namen der Serien jedesmal bei einer Ergänzung wissen (und *komplettieren*) zu müssen. Natürlich kannst Du durch sinnvolle Namensgebung auch mehr als zwei unterschiedliche Gruppen von Serien bilden. Dies ist nur ein einfaches Beispiel, um das Prinzip zu erläutern.

Konfigurierung der Replikationsoptionen `seriesToDistribute` und `backupCopy*` mittels Wildcards

Wenn Du die Replikation von Backup Serien dynamisch angepasst konfigurieren willst (und eventuell andere Serien davon ausnehmen willst), kannst Du für die Konfiguration der Replikationen Wildcards verwenden. Im Beispiel werden die folgenden Verzeichnisse in `/tmp` zur Erläuterung der Verwendung von Wildcards verwendet:

`/tmp/a/b` Master Backup Directory

`/tmp/a/d` Delta Cache Directory

`/tmp/a/c` Replikations “copy” Directory

Zuerst musst Du einige Backup-Serien-Verzeichnisse erzeugen und die Konfigurationsdateien für die Replikation erzeugen:

```
1 $ cd /tmp/a
2 $ mkdir server1 server1.norepl server2 server2.norepl
3 $ storeBackupReplicationWizard.pl -m b -c c -d d -S server1
```

Im nächsten Schritt so konfigurieren, dass das aufgelistete Resultat erreicht wird:

```
1 $ grep -vP '\A\s*\Z|\A[#;]' b/storeBackupBaseTree.conf
2 backupTreeName='Master Backup'
3 backupType=master
4 seriesToDistribute= +* -*.norepl
5 deltaCache=/tmp/a/d

1 $ grep -vP '\A\s*\Z|\A[#;]' d/deltaCache.conf
2 backupCopy0='Backup Copy' +*

1 $ grep -vP '\A\s*\Z|\A[#;]' c/storeBackupBaseTree.conf
2 backupTreeName='Backup Copy'
3 backupType=copy
4 seriesToDistribute= +*
5 deltaCache=/tmp/a/d
```

Wichtig: Erinnere Dich daran, dass Du Backups mit Replikation mit `storeBackup.pl` mit der Option `lateLinks` durchführen *musst*!

Wie Du feststellen kannst, ist die Syntax sehr ähnlich zu der bei `otherBackupSeries`. Anstelle von „+*“ kannst Du auch „*“ schreiben.

Die Konfiguration oben bewirkt eventuell die folgende Fehlermeldung wenn das Replikationsprogramm für das Replikationsverzeichnis aufgerufen wird:

```
1 $ storeBackupUpdateBackup.pl -b c
2 ....
3 ERROR      2014.02.22 11:59:48 10007 c/storeBackupBaseTree.conf series <server1> missing in
   ↳ <Backup Copy>, defined in /tmp/a/d/deltaCache.conf
4 ERROR      2014.02.22 11:59:48 10007 use option --createNewSeries if you want missing series to
   ↳ be created automatically
5 ....
```

Die einzige Möglichkeit für `storeBackup`, den `deltaCache` mit dem Replikationskopie-Verzeichnis zu vergleichen, ist, die Wildcards zu expandieren. Aber wenn die Replikation für eine neue Serie das erste Mal läuft, existiert diese Serie natürlich noch nicht im Replikationsverzeichnis. Du kannst es manuell mit `mkdir` anlegen (was Du wahrscheinlich nicht willst), oder das tun, was `storeBackupUpdateBackup.pl` Dir vorschlägt – die neue Serie automatisch erzeugen:

```
1 $ storeBackupUpdateBackup.pl -b c --createNewSeries
```

7.9 Spezielle Dateien von storeBackup

Ändere niemals die unten beschriebenen Dateien! Sie sind absolut wichtig, damit storeBackup richtig funktioniert!

In einem Backup werden immer die folgenden Einträge generiert. Lösche sie nicht. Stelle auch sicher, dass Du derartige Einträge nicht in der obersten Verzeichnisebene Deines Quellverzeichnis hast:

.md5CheckSums.info Diese Datei beinhaltet Metadaten über das Backup. Beispiel (einige Zeile sind zur besseren Lesbarkeit abgeschnitten):

```
1 version=1.3
2 date=2008.09.06 10.23.33
3 sourceDir='/home/hjc'
4 followLinks=0
5 compress='bzip2'
6 uncompress='bzip2' '-d'
7 postfix='.bz2'
8 exceptSuffix='bz2' 'gif' 'jpg' 'mp3' 'mpeg' 'mpg' 'ogg'
9 exceptDirs='/home/hjc/Mail' '/home/hjc/Maildir' '/home/hjc/nosave' '/home/hjc/tmp'
10 includeDirs=
11 exceptRule='$size > &::SIZE("100M")'
12 includeRule=
13 exceptTypes=
14 preservePerms=yes
15 lateLinks=yes
16 lateCompress=yes
17 cpIsGnu=yes
```

.md5CheckSums[.bz2] Diese Datei enthält alle Informationen über die Dateien, Verzeichnisse, ... im Backup. Hier einige Zeilen als Beispiel:

```
1 # contents/md5 compr dev-inode inodeBackup ctime mtime atime size uid
2 gid mode filename
3 dir 0 2097-386 0 1169342164 1094800914 1200948038 0 1049 100 493 c++
4 063e5feb114a82059e7f44c5fb0e548c c 2097-1834 1372638 1169343033 1078512595 1125554314
   ↪ 489786 1049 1001 384 mbox
5 symlink 0 2097-31105 0 1169350675 1169350675 1169350675 0 1049 0 0 .Xresources
```

Die Rechte werden als dezimale Werte (nicht oktal) gespeichert.

.storeBackupLinks Ein Verzeichnis, das leer ist, wenn alle Links gesetzt sind.

Diese Dateien können direkt im Backup Verzeichnis stehen:

.md5CheckSums.notFinished Bis Version 3.4.3: Die Existenz dieser Datei zeigt, dass dieses Backup nicht richtig beendet wurde (d.h. es wurde z. B. mit Strg-c abgebrochen).

.storeBackupLinks/backup.Finished Backup erzeugt mit Version 3.5 oder höher: Die Existenz dieser Datei zeigt an, dass dieses Backup korrekt beendet wurde. Wenn es fehlt, wurde das Backup nicht richtig beendet (d.h. es wurde z. B. mit Strg-c abgebrochen).

.storeBackup.log[.bz2] Die Logdatei von storeBackup.pl. Dies ist der Standardname, den Du mit Optionen von storeBackup.pl ändern kannst (Optionen logInBackupDir und compressLogInBackupDir).

.storeBackup.notSaved.bz2 Falls Du Dateien über Regeln ausschließt, kann Du (mit writeExcludeLog) eine Liste mit den betroffenen Dateien generieren lassen.

Die folgenden Dateien existieren nur, wenn Du Option lateLinks, siehe Kapitel 7.6 verwendest. Mit einem erfolgreichen Lauf von storeBackupUpdateBackup.pl, siehe Kapitel 6.3 werden diese Dateien wieder gelöscht.

.storeBackupLinks/linkFile.bz2 Enthält neben der Datei linkFile.bz2 Informationen darüber, was von storeBackupUpdateBackup.pl zu ergänzen ist.

.storeBackupLinks/linkTo Enthält relative Pfade zu den Backups, auf die linkFile.bz2 verweist, z. B.:

```

1  ../2008.09.05_16.07.23
2  ../../lotte/2008.09.06_02.00.04

```

Hier siehst Du den relativen Pfad zu einem vorherigen Backup und einen Link zum Backup einer anderen Serie.

`.storeBackupLinks/linkFrom<number>` Jede dieser Dateien enthält relative Pfade *von* Backups zu dem aktuellen, Beispiel:

```

1  ../2008.09.06_10.23.33

```

7.10 Wie robust ist storeBackup?

In der folgenden Beschreibung wird zwischen der Verwendung der Option `--lateLinks` und deren nicht-Verwendung unterschieden. Die Option hat Einfluss auf das Verhalten von `storeBackup` und ist gleichzeitig Basis vieler Features. Bei der Verwendung von `lateLinks` werden die Hard Links im Backup erst später (zeitversetzt) gesetzt, so dass temporär eine Art inkrementelle Backups (hier für die Replikation auch als „Delta Backup“ bezeichnet) existieren, die neben den neu dazukommenden Dateien unter anderem auch Informationen beinhalten, welche Hard Links für ein vollständiges Backup noch zu setzen sind.

In diesem Kapitel geht es darum zu zeigen, inwieweit sich Fehler bei der Entstellung eines Backups⁶⁰ „auswachsen“ oder inwieweit diese sich fortpflanzen.

7.10.1 Robustheit ohne lateLinks

Gehen wir von der folgenden Situation für das dargestellte Source Directory aus. Das `find`-Kommando liefert nur die Dateien (`-type f`, Zeile 1 im folgendem Listing), die explizit nicht aufgeführten Directories sind natürlich auch zu erstellen, wenn man das Beispiel nachvollziehen möchte.

```

1  $ find test -type f -print | sort
2  test/source1
3  test/source1/1
4  test/source1/2
5  test/source1/sub1/1.1
6  test/source1/sub1/ls

```

Die Dateien haben einen ihrem Namen entsprechenden Inhalt. Die Struktur für dieses Beispiel kann mit folgenden Kommandos z. B. in `/tmp` erstellt werden:

```

1  $ mkdir -p /tmp/test/backup
2  $ mkdir /tmp/test/source1
3  $ echo 1 > /tmp/test/source1/1
4  $ echo 2 > /tmp/test/source1/2
5  $ mkdir /tmp/test/source1/sub1
6  $ echo 1 > /tmp/test/source1/sub1/1.1
7  $ cp /bin/ls /tmp/test/source1/sub1/ls
8  $ find test -ls | sort
9  44959719      4 -rw-rw-r--  1 hjc      zusammen      2 Apr  6 09:39 test/source1/1
10 44959720      4 -rw-rw-r--  1 hjc      zusammen      2 Apr  6 09:39 test/source1/2
11 44959726      4 -rw-rw-r--  1 hjc      zusammen      2 Apr  6 09:41 test/source1/sub1/1.1
12 44959727    144 -rwxr-xr-x  1 hjc      zusammen    146392 Apr  6 09:41 test/source1/sub1/ls
13 45031462      4 drwxrwxr-x  4 hjc      zusammen     4096 Apr  6 09:38 test
14 45031463      4 drwxrwxr-x  2 hjc      zusammen     4096 Apr  6 09:32 test/backup
15 45031464      4 drwxrwxr-x  3 hjc      zusammen     4096 Apr  6 09:40 test/source1
16 45031466      4 drwxrwxr-x  2 hjc      zusammen     4096 Apr  6 09:41 test/source1/sub1

```

Die Dateien werden später im Backup als Kopien vorgehalten; Die Datei `ls` (Zeile 12) wird komprimiert gespeichert. Dann entpacken wir das aktuelle `storeBackup` tar File und setzen `$PATH` auf das zugehörige `bin` Verzeichnis (nicht dargestellt). Alternativ kann der Aufruf der `storeBackup` Programme jeweils mit vollem Pfad erfolgen.

Anschließend löschen wir die Dateien `1`, `1.1` und `ls.bz2` aus dem Backup und sehen uns das Ergebnis an:

⁶⁰Keine Sorge: `storeBackup` verliert normalerweise keine Daten und bietet auch Mechanismen, einen derartigen Verlust zu erkennen.

```

1 $ storeBackup.pl -s test/source1/ -b test/backup/ -S series1
2 ...
3 $ rm test/backup/series1/2022.04.04_23.14.56/1 test/backup/series1/2022.04.04_23.14.56/sub1/1.1
  ↳ test/backup/series1/2022.04.04_23.14.56/sub1/ls.bz2
4 $ find test -print | sort
5 test/backup/series1/2022.04.04_23.05.46/2
6 test/backup/series1/2022.04.04_23.05.46/.md5CheckSums.bz2
7 test/backup/series1/2022.04.04_23.05.46/.md5CheckSums.fail
8 test/backup/series1/2022.04.04_23.05.46/.md5CheckSums.info
9 test/backup/series1/2022.04.04_23.05.46/.storeBackupLinks/backup.Finished
10 test/backup/series1/2022.04.04_23.14.56/2
11 test/backup/series1/2022.04.04_23.14.56/.md5CheckSums.bz2
12 test/backup/series1/2022.04.04_23.14.56/.md5CheckSums.info
13 test/backup/series1/2022.04.04_23.14.56/.storeBackupLinks/backup.Finished
14 test/source1/1
15 test/source1/2
16 test/source1/sub1/1.1
17 test/source1/sub1/ls
18 hjc@umpf:~/lotte/stbu/s

```

Danach starten wir storeBackup.pl und erhalten als Ergebnis ein ganz normales Backup mit Standardwerten von storeBackup.pl:

```

1 $ storeBackup.pl -s test/source1/ -b test/backup/ -S series1
2 ...
3 INFO      2022.04.09 02:53:25 532109 scanning directory </tmp/test/backup> for existing backups
4 INFO      2022.04.09 02:53:25 532109 scanning directory </tmp/test/backup/series1> for existing
  ↳ backups
5 STATISTIC 2022.04.09 02:53:25 532109 found 1 backup series, 2 backups, 0 renamed backups
6 INFO      2022.04.09 02:53:25 532109 consistency check finished successfully
7 INFO      2022.04.09 02:53:25 532109 found no references to backups from lateLinks that need
  ↳ storeBackupUpdateBackup run
8 INFO      2022.04.09 02:53:25 532109 creating lock file </home/hjc/storeBackup.lock>
9 INFO      2022.04.09 02:53:25 532109 analysis of old Backups in </tmp/test/backup/series1>:
10 INFO      2022.04.09 02:53:25 532109      (2) Mon 2022.04.04_23.05.46 (4d3h): keepDuplicate(7d)
11 INFO      2022.04.09 02:53:25 532109      (1) Mon 2022.04.04_23.14.56 (4d3h): keepMinNumber2,
  ↳ keepWeekDays(30d)
12 INFO      2022.04.09 02:53:25 532109      (0) Sat 2022.04.09_02.53.25 (0d0h): keepMinNumber1,
  ↳ keepWeekDays(30d)
13 INFO      2022.04.09 02:53:25 532109 start reading
  ↳ /tmp/test/backup/series1/2022.04.04_23.14.56/.md5CheckSums.bz2
14 INFO      2022.04.09 02:53:25 532109 finished reading
  ↳ /tmp/test/backup/series1/2022.04.04_23.14.56/.md5CheckSums.bz2 (5 entries)
15 INFO      2022.04.09 02:53:25 532109 4 entries in dbm files
16 INFO      2022.04.09 02:53:25 532109 0 entries in dbm block files
17 INFO      2022.04.09 02:53:26 532109 setting atime, mtime of directories ...
18 INFO      2022.04.09 02:53:26 532109 syncing ...
19 STATISTIC 2022.04.09 02:53:26 532109 [sec] |      user|      system
20 STATISTIC 2022.04.09 02:53:26 532109 -----+-----+-----
21 STATISTIC 2022.04.09 02:53:26 532109 process|      0.16|      0.01
22 STATISTIC 2022.04.09 02:53:26 532109 childs |      0.18|      0.02
23 STATISTIC 2022.04.09 02:53:26 532109 -----+-----+-----
24 STATISTIC 2022.04.09 02:53:26 532109 sum    |      0.34|      0.03 => 0.37 (0s)
25 STATISTIC 2022.04.09 02:53:26 532109                                directories = 1
26 STATISTIC 2022.04.09 02:53:26 532109                                files = 4
27 STATISTIC 2022.04.09 02:53:26 532109                                symbolic links = 0
28 STATISTIC 2022.04.09 02:53:26 532109                                late links = 0
29 STATISTIC 2022.04.09 02:53:26 532109                                named pipes = 0
30 STATISTIC 2022.04.09 02:53:26 532109                                sockets = 0
31 STATISTIC 2022.04.09 02:53:26 532109                                block devices = 0
32 STATISTIC 2022.04.09 02:53:26 532109                                character devices = 0
33 STATISTIC 2022.04.09 02:53:26 532109                                new internal linked files = 1
34 STATISTIC 2022.04.09 02:53:26 532109                                old linked files = 1
35 STATISTIC 2022.04.09 02:53:26 532109                                unchanged files = 2

```

```

36 STATISTIC 2022.04.09 02:53:26 532109 copied files = 1
37 STATISTIC 2022.04.09 02:53:26 532109 compressed files = 1
38 STATISTIC 2022.04.09 02:53:26 532109 blocked files = 0
39 STATISTIC 2022.04.09 02:53:26 532109 excluded files because rule = 0 (0.0 )
40 STATISTIC 2022.04.09 02:53:26 532109 included files because rule = 0 (0.0 )
41 STATISTIC 2022.04.09 02:53:26 532109 max size of copy queue = 1
42 STATISTIC 2022.04.09 02:53:26 532109 max size of compression queue = 1
43 STATISTIC 2022.04.09 02:53:26 532109 calculated md5 sums = 7
44 STATISTIC 2022.04.09 02:53:26 532109 forks total = 3
45 STATISTIC 2022.04.09 02:53:26 532109 forks md5 = 2
46 STATISTIC 2022.04.09 02:53:26 532109 forks copy = 0
47 STATISTIC 2022.04.09 02:53:26 532109 forks bzip2 = 1
48 STATISTIC 2022.04.09 02:53:26 532109 sum of source = 143k (146398)
49 STATISTIC 2022.04.09 02:53:26 532109 sum of target all = 63k (64076)
50 STATISTIC 2022.04.09 02:53:26 532109 sum of target all = 43.77%
51 STATISTIC 2022.04.09 02:53:26 532109 sum of target new = 63k (64072)
52 STATISTIC 2022.04.09 02:53:26 532109 sum of target new = 43.77%
53 STATISTIC 2022.04.09 02:53:26 532109 sum of md5ed files = 143k (146398)
54 STATISTIC 2022.04.09 02:53:26 532109 sum of md5ed files = 100.00%
55 STATISTIC 2022.04.09 02:53:26 532109 sum internal linked (copy) = 2.0 (2)
56 STATISTIC 2022.04.09 02:53:26 532109 sum internal linked (compr) = 0.0 (0)
57 STATISTIC 2022.04.09 02:53:26 532109 sum old linked (copy) = 2.0 (2)
58 STATISTIC 2022.04.09 02:53:26 532109 sum old linked (compr) = 0.0 (0)
59 STATISTIC 2022.04.09 02:53:26 532109 sum unchanged (copy) = 4.0 (4)
60 STATISTIC 2022.04.09 02:53:26 532109 sum unchanged (compr) = 0.0 (0)
61 STATISTIC 2022.04.09 02:53:26 532109 sum new (copy) = 2.0 (2)
62 STATISTIC 2022.04.09 02:53:26 532109 sum new (compr) = 63k (64070)
63 STATISTIC 2022.04.09 02:53:26 532109 sum new (compr), orig size = 143k (146392)
64 STATISTIC 2022.04.09 02:53:26 532109 sum new / orig = 43.77%
65 STATISTIC 2022.04.09 02:53:26 532109 size of md5Checksum file = 337 (337)
66 STATISTIC 2022.04.09 02:53:26 532109 size of temporary db files = 0.0 (0)
67 STATISTIC 2022.04.09 02:53:26 532109 deleted old backups = 0
68 STATISTIC 2022.04.09 02:53:26 532109 deleted directories = 0
69 STATISTIC 2022.04.09 02:53:26 532109 deleted files = 0
70 STATISTIC 2022.04.09 02:53:26 532109 (only) removed links = 0
71 STATISTIC 2022.04.09 02:53:26 532109 freed space in old directories = 0.0 (0)
72 STATISTIC 2022.04.09 02:53:26 532109 add. used space in files = 63k (64409)
73 STATISTIC 2022.04.09 02:53:26 532109 backup duration = 1s
74 STATISTIC 2022.04.09 02:53:26 532109 over all files/sec (real time) = 4.00
75 STATISTIC 2022.04.09 02:53:26 532109 over all files/sec (CPU time) = 10.81
76 STATISTIC 2022.04.09 02:53:26 532109 CPU usage = 37.00%
77 INFO 2022.04.09 02:53:26 532109 removing lock file </home/hjc/storeBackup.lock>
78 END 2022.04.09 02:53:26 532109 backing up directory </tmp/test/source1> to
↳ </tmp/test/backup/series1/2022.04.09_02.53.25
79 $ find test -print | sort
80 test/backup/series1/2022.04.04_23.05.46/2
81 test/backup/series1/2022.04.04_23.05.46/.md5CheckSums.bz2
82 test/backup/series1/2022.04.04_23.05.46/.md5CheckSums.fail
83 test/backup/series1/2022.04.04_23.05.46/.md5CheckSums.info
84 test/backup/series1/2022.04.04_23.05.46/.storeBackupLinks/backup.Finished
85 test/backup/series1/2022.04.04_23.14.56/2
86 test/backup/series1/2022.04.04_23.14.56/.md5CheckSums.bz2
87 test/backup/series1/2022.04.04_23.14.56/.md5CheckSums.info
88 test/backup/series1/2022.04.04_23.14.56/.storeBackupLinks/backup.Finished
89 test/backup/series1/2022.04.09_02.53.25/1
90 test/backup/series1/2022.04.09_02.53.25/2
91 test/backup/series1/2022.04.09_02.53.25/.md5CheckSums.bz2
92 test/backup/series1/2022.04.09_02.53.25/.md5CheckSums.info
93 test/backup/series1/2022.04.09_02.53.25/.storeBackupLinks/backup.Finished
94 test/backup/series1/2022.04.09_02.53.25/sub1/1.1
95 test/backup/series1/2022.04.09_02.53.25/sub1/1s.bz2

```

Im Backup fehlende Dateien werden also beim nächsten Backup (hier immer ohne lateLinks) wieder im

Backup ergänzt. Dieses erfolgt aber nur im letzten Backup der Serie, weil das Programm feststellt, dass das Setzen eines Hard Links nicht klappt und versucht die Datei zu kopieren, was erfolgreich ist.⁶¹ Länger zurückliegende Sicherungen werden nicht verändert. Wird also eine Datei in einem Backup von vor z. B. drei Monaten gelöscht oder anderweitig verändert, so wird dieses weder bemerkt noch repariert. Dieses betrifft auch die Replikation, siehe Replikation von Backups, Kapitel 7.8. Derartige Änderungen können mittels `storeBackupCheckBackup.pl` überprüft und festgestellt werden. Hier geht es darum, wie sich das Speicherverfahren insbesondere beim im nächsten Kapitel erläuterten Verfahren *mit lateLinks* verhält. Bisher (Version 3.5.2) ist das hier beschriebne Verhalten nur für non-block-Dateien implementiert.

7.10.2 Robustheit mit lateLinks, einfaches Beispiel

Um eine Vergleichbarkeit zwischen mit und ohne `lateLinks` zu ermöglichen wird die Sicherung mit demselben Dateischema durchgeführt wie im vorigen Kapitel *ohne lateLinks*. Der dem hier beschriebenen Verhalten zugrunde liegende Algorithmus ist seit Version 3.5.2 in `storeBackup` implementiert.

```
1 $ find test -print | sort
2 test/source1
3 test/source1/1
4 test/source1/2
5 test/source1/sub1/1.1
6 test/source1/sub1/ls
```

Um das Beispiel kurz zu halten, wird die Option `lateLinks` gleich bei der ersten Sicherung verwendet. Das führt dazu, dass doppelt vorkommende Dateien nur einmal kopiert werden und noch kein Hard Link angelegt wird.

```
1 $ storeBackup.pl -s test/source1 -b test/backup -S series1 --lateLinks
2 ...
3 INFO      2022.04.05 08:53:24 132532 scanning directory </tmp/test/backup> for existing backups
4 INFO      2022.04.05 08:53:24 132532 scanning directory </tmp/test/backup/series1> for existing
   ↳ backups
5 STATISTIC 2022.04.05 08:53:24 132532 found 1 backup series, 0 backups, 0 renamed backups
6 INFO      2022.04.05 08:53:24 132532 consistency check finished successfully
7 INFO      2022.04.05 08:53:24 132532 found no references to backups from lateLinks that need
   ↳ storeBackupUpdateBackup run
8 INFO      2022.04.05 08:53:24 132532 creating lock file </home/hjc/storeBackup.lock>
9 INFO      2022.04.05 08:53:24 132532 analysis of old Backups in </tmp/test/backup/series1>:
10 INFO     2022.04.05 08:53:24 132532      (0) Tue 2022.04.05_08.53.24 (0d0h): keepMinNumber1,
   ↳ keepWeekDays(30d)
11 INFO     2022.04.05 08:53:24 132532 0 entries in dbm files
12 INFO     2022.04.05 08:53:24 132532 0 entries in dbm block files
13 INFO     2022.04.05 08:53:24 132532 lateLinks is switched on
14 STATISTIC 2022.04.05 08:53:25 132532 [sec] |      user|      system
15 STATISTIC 2022.04.05 08:53:25 132532 -----+-----+-----
16 STATISTIC 2022.04.05 08:53:25 132532 process|      0.17|      0.02
17 STATISTIC 2022.04.05 08:53:25 132532 childs |      0.19|      0.01
18 STATISTIC 2022.04.05 08:53:25 132532 -----+-----+-----
19 STATISTIC 2022.04.05 08:53:25 132532 sum    |      0.36|      0.03 => 0.39 (0s)
20 STATISTIC 2022.04.05 08:53:25 132532                                directories = 1
21 STATISTIC 2022.04.05 08:53:25 132532                                files = 4
22 STATISTIC 2022.04.05 08:53:25 132532                                symbolic links = 0
23 STATISTIC 2022.04.05 08:53:25 132532                                late links = 1
24 STATISTIC 2022.04.05 08:53:25 132532                                named pipes = 0
25 STATISTIC 2022.04.05 08:53:25 132532                                sockets = 0
26 STATISTIC 2022.04.05 08:53:25 132532                                block devices = 0
27 STATISTIC 2022.04.05 08:53:25 132532                                character devices = 0
28 STATISTIC 2022.04.05 08:53:25 132532                                new internal linked files = 1
29 STATISTIC 2022.04.05 08:53:25 132532                                old linked files = 0
30 STATISTIC 2022.04.05 08:53:25 132532                                unchanged files = 0
31 STATISTIC 2022.04.05 08:53:25 132532                                copied files = 2
```

⁶¹Derselbe Algorithmus sorgt auch dafür, dass bei Erreichen der für das jeweilige Filesystem möglichen maximalen Anzahl von Hard Links mit einer neuen Datei weiter verlinkt wird.

```

32 STATISTIC 2022.04.05 08:53:25 132532 compressed files = 1
33 STATISTIC 2022.04.05 08:53:25 132532 blocked files = 0
34 STATISTIC 2022.04.05 08:53:25 132532 excluded files because rule = 0 (0.0 )
35 STATISTIC 2022.04.05 08:53:25 132532 included files because rule = 0 (0.0 )
36 STATISTIC 2022.04.05 08:53:25 132532 max size of copy queue = 1
37 STATISTIC 2022.04.05 08:53:25 132532 max size of compression queue = 1
38 STATISTIC 2022.04.05 08:53:25 132532 calculated md5 sums = 8
39 STATISTIC 2022.04.05 08:53:25 132532 forks total = 3
40 STATISTIC 2022.04.05 08:53:25 132532 forks md5 = 2
41 STATISTIC 2022.04.05 08:53:25 132532 forks copy = 0
42 STATISTIC 2022.04.05 08:53:25 132532 forks bzip2 = 1
43 STATISTIC 2022.04.05 08:53:25 132532 sum of source = 143k (146398)
44 STATISTIC 2022.04.05 08:53:25 132532 sum of target all = 63k (64076)
45 STATISTIC 2022.04.05 08:53:25 132532 sum of target all = 43.77%
46 STATISTIC 2022.04.05 08:53:25 132532 sum of target new = 63k (64074)
47 STATISTIC 2022.04.05 08:53:25 132532 sum of target new = 43.77%
48 STATISTIC 2022.04.05 08:53:25 132532 sum of md5ed files = 143k (146398)
49 STATISTIC 2022.04.05 08:53:25 132532 sum of md5ed files = 100.00%
50 STATISTIC 2022.04.05 08:53:25 132532 sum internal linked (copy) = 2.0 (2)
51 STATISTIC 2022.04.05 08:53:25 132532 sum internal linked (compr) = 0.0 (0)
52 STATISTIC 2022.04.05 08:53:25 132532 sum old linked (copy) = 0.0 (0)
53 STATISTIC 2022.04.05 08:53:25 132532 sum old linked (compr) = 0.0 (0)
54 STATISTIC 2022.04.05 08:53:25 132532 sum unchanged (copy) = 0.0 (0)
55 STATISTIC 2022.04.05 08:53:25 132532 sum unchanged (compr) = 0.0 (0)
56 STATISTIC 2022.04.05 08:53:25 132532 sum new (copy) = 4.0 (4)
57 STATISTIC 2022.04.05 08:53:25 132532 sum new (compr) = 63k (64070)
58 STATISTIC 2022.04.05 08:53:25 132532 sum new (compr), orig size = 143k (146392)
59 STATISTIC 2022.04.05 08:53:25 132532 sum new / orig = 43.77%
60 STATISTIC 2022.04.05 08:53:25 132532 size of md5Checksum file = 345 (345)
61 STATISTIC 2022.04.05 08:53:25 132532 size of temporary db files = 0.0 (0)
62 STATISTIC 2022.04.05 08:53:25 132532 deleted old backups = 0
63 STATISTIC 2022.04.05 08:53:25 132532 deleted directories = 0
64 STATISTIC 2022.04.05 08:53:25 132532 deleted files = 0
65 STATISTIC 2022.04.05 08:53:25 132532 (only) removed links = 0
66 STATISTIC 2022.04.05 08:53:25 132532 freed space in old directories = 0.0 (0)
67 STATISTIC 2022.04.05 08:53:25 132532 add. used space in files = 63k (64419)
68 STATISTIC 2022.04.05 08:53:25 132532 backup duration = 1s
69 STATISTIC 2022.04.05 08:53:25 132532 over all files/sec (real time) = 4.00
70 STATISTIC 2022.04.05 08:53:25 132532 over all files/sec (CPU time) = 10.26
71 STATISTIC 2022.04.05 08:53:25 132532 CPU usage = 39.00%
72 INFO 2022.04.05 08:53:25 132532 removing lock file </home/hjc/storeBackup.lock>
73 WARNING 2022.04.05 08:53:25 132532 -- 1 WARNING OCCURRED DURING THE BACKUP! --
74 END 2022.04.05 08:53:25 132532 backing up directory </tmp/test/source1> to
↳ </tmp/test/backup/series1/2022.04.05_08.53.24>
75 $ find test -print | sort
76 test
77 test/backup
78 test/backup/series1
79 test/backup/series1/2022.04.05_09.08.26
80 test/backup/series1/2022.04.05_09.08.26/1
81 test/backup/series1/2022.04.05_09.08.26/2
82 test/backup/series1/2022.04.05_09.08.26/.md5Checksums.bz2
83 test/backup/series1/2022.04.05_09.08.26/.md5Checksums.info
84 test/backup/series1/2022.04.05_09.08.26/.storeBackupLinks
85 test/backup/series1/2022.04.05_09.08.26/.storeBackupLinks/backup.Finished
86 test/backup/series1/2022.04.05_09.08.26/.storeBackupLinks/linkFile.bz2
87 test/backup/series1/2022.04.05_09.08.26/sub1
88 test/backup/series1/2022.04.05_09.08.26/sub1/ls.bz2
89 test/source1
90 test/source1/1
91 test/source1/2
92 test/source1/sub1

```



```

93 test/source1/sub1/1.1
94 test/source1/sub1/ls

```

Danach löschen wir die Dateien 1, 1.1 und ls.bz2 und starten storeBackup wie gehabt mit lateLinks. Auch dieses Mal läuft storeBackup.pl ohne Fehlermeldung, weil zum Erstellen eines Backups mit Option lateLinks nur die Metadaten des benötigten Backups bzw. nur die der alten Backups benötigt werden. (siehe Isolated Mode / Offline Backups, Kapitel 7.7 und storeBackupSetupIsolatedMode, Kapitel 6.7)

```

1  $ rm test/backup/series1/2022.04.05_09.08.26/1
   ↳ test/backup/series1/2022.04.05_09.08.26/sub1/ls.bz2 test/source1/sub1/1.1
2  $ storeBackup.pl -s test/source1 -b test/backup -S series1 --lateLinks
3  ...
4  INFO      2022.04.05 09:42:40 134411 scanning directory </tmp/test/backup> for existing backups
5  INFO      2022.04.05 09:42:40 134411 scanning directory </tmp/test/backup/series1> for existing
   ↳ backups
6  STATISTIC 2022.04.05 09:42:40 134411 found 1 backup series, 1 backups, 0 renamed backups
7  INFO      2022.04.05 09:42:40 134411 consistency check finished successfully
8  INFO      2022.04.05 09:42:40 134411 found no references to backups from lateLinks that need
   ↳ storeBackupUpdateBackup run
9  INFO      2022.04.05 09:42:40 134411 creating lock file </home/hjc/storeBackup.lock>
10 INFO      2022.04.05 09:42:40 134411 analysis of old Backups in </tmp/test/backup/series1>:
11 INFO      2022.04.05 09:42:40 134411 (1) Tue 2022.04.05_09.08.26 (0d0h): affected by
   ↳ unresolved links, keepDuplicate(7d)
12 INFO      2022.04.05 09:42:40 134411 (0) Tue 2022.04.05_09.42.40 (0d0h): affected by
   ↳ unresolved links, keepMinNumber1, keepWeekDays(30d)
13 INFO      2022.04.05 09:42:40 134411 start reading
   ↳ /tmp/test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.bz2
14 INFO      2022.04.05 09:42:40 134411 finished reading
   ↳ /tmp/test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.bz2 (5 entries)
15 INFO      2022.04.05 09:42:40 134411 4 entries in dbm files
16 INFO      2022.04.05 09:42:40 134411 0 entries in dbm block files
17 INFO      2022.04.05 09:42:40 134411 lateLinks is switched on
18 INFO      2022.04.05 09:42:40 134411 syncing ...
19 STATISTIC 2022.04.05 09:42:40 134411 [sec] |      user|      system
20 STATISTIC 2022.04.05 09:42:40 134411 -----+-----+-----
21 STATISTIC 2022.04.05 09:42:40 134411 process|      0.16|      0.02
22 STATISTIC 2022.04.05 09:42:40 134411 childs |      0.15|      0.01
23 STATISTIC 2022.04.05 09:42:40 134411 -----+-----+-----
24 STATISTIC 2022.04.05 09:42:40 134411 sum    |      0.31|      0.03 => 0.34 (0s)
25 STATISTIC 2022.04.05 09:42:40 134411          directories = 1
26 STATISTIC 2022.04.05 09:42:40 134411          files = 4
27 STATISTIC 2022.04.05 09:42:40 134411          symbolic links = 0
28 STATISTIC 2022.04.05 09:42:40 134411          late links = 4
29 STATISTIC 2022.04.05 09:42:40 134411          named pipes = 0
30 STATISTIC 2022.04.05 09:42:40 134411          sockets = 0
31 STATISTIC 2022.04.05 09:42:40 134411          block devices = 0
32 STATISTIC 2022.04.05 09:42:40 134411          character devices = 0
33 STATISTIC 2022.04.05 09:42:40 134411          new internal linked files = 0
34 STATISTIC 2022.04.05 09:42:40 134411          old linked files = 4
35 STATISTIC 2022.04.05 09:42:40 134411          unchanged files = 0
36 STATISTIC 2022.04.05 09:42:40 134411          copied files = 0
37 STATISTIC 2022.04.05 09:42:40 134411          compressed files = 0
38 STATISTIC 2022.04.05 09:42:40 134411          blocked files = 0
39 STATISTIC 2022.04.05 09:42:40 134411          excluded files because rule = 0 (0.0 )
40 STATISTIC 2022.04.05 09:42:40 134411          included files because rule = 0 (0.0 )
41 STATISTIC 2022.04.05 09:42:40 134411          max size of copy queue = 0
42 STATISTIC 2022.04.05 09:42:40 134411 max size of compression queue = 0
43 STATISTIC 2022.04.05 09:42:40 134411          calculated md5 sums = 0
44 STATISTIC 2022.04.05 09:42:40 134411          forks total = 0
45 STATISTIC 2022.04.05 09:42:40 134411          forks md5 = 0
46 STATISTIC 2022.04.05 09:42:40 134411          forks copy = 0
47 STATISTIC 2022.04.05 09:42:40 134411          forks bzip2 = 0
48 STATISTIC 2022.04.05 09:42:40 134411          sum of source = 143k (146398)

```



```

49 STATISTIC 2022.04.05 09:42:40 134411 sum of target all = 143k (146398)
50 STATISTIC 2022.04.05 09:42:40 134411 sum of target all = 100.00%
51 STATISTIC 2022.04.05 09:42:40 134411 sum of target new = 0.0 (0)
52 STATISTIC 2022.04.05 09:42:40 134411 sum of target new = 0.00%
53 STATISTIC 2022.04.05 09:42:40 134411 sum of md5ed files = 0.0 (0)
54 STATISTIC 2022.04.05 09:42:40 134411 sum of md5ed files = 0.00%
55 STATISTIC 2022.04.05 09:42:40 134411 sum internal linked (copy) = 0.0 (0)
56 STATISTIC 2022.04.05 09:42:40 134411 sum internal linked (compr) = 0.0 (0)
57 STATISTIC 2022.04.05 09:42:40 134411 sum old linked (copy) = 6.0 (6)
58 STATISTIC 2022.04.05 09:42:40 134411 sum old linked (compr) = 143k (146392)
59 STATISTIC 2022.04.05 09:42:40 134411 sum unchanged (copy) = 0.0 (0)
60 STATISTIC 2022.04.05 09:42:40 134411 sum unchanged (compr) = 0.0 (0)
61 STATISTIC 2022.04.05 09:42:40 134411 sum new (copy) = 0.0 (0)
62 STATISTIC 2022.04.05 09:42:40 134411 sum new (compr) = 0.0 (0)
63 STATISTIC 2022.04.05 09:42:40 134411 sum new (compr), orig size = 0.0 (0)
64 STATISTIC 2022.04.05 09:42:40 134411 sum new / orig = 0.00%
65 STATISTIC 2022.04.05 09:42:40 134411 size of md5Checksum file = 345 (345)
66 STATISTIC 2022.04.05 09:42:40 134411 size of temporary db files = 0.0 (0)
67 STATISTIC 2022.04.05 09:42:40 134411 deleted old backups = 0
68 STATISTIC 2022.04.05 09:42:40 134411 deleted directories = 0
69 STATISTIC 2022.04.05 09:42:40 134411 deleted files = 0
70 STATISTIC 2022.04.05 09:42:40 134411 (only) removed links = 0
71 STATISTIC 2022.04.05 09:42:40 134411 freed space in old directories = 0.0 (0)
72 STATISTIC 2022.04.05 09:42:40 134411 add. used space in files = 345 (345)
73 STATISTIC 2022.04.05 09:42:40 134411 backup duration = 1s
74 STATISTIC 2022.04.05 09:42:40 134411 over all files/sec (real time) = 4.00
75 STATISTIC 2022.04.05 09:42:40 134411 over all files/sec (CPU time) = 11.76
76 STATISTIC 2022.04.05 09:42:40 134411 CPU usage = 34.00%
77 INFO 2022.04.05 09:42:40 134411 removing lock file </home/hjc/storeBackup.lock>
78 END 2022.04.05 09:42:40 134411 backing up directory </tmp/test/source1> to
↪ </tmp/test/backup/series1/2022.04.05_09.42.40>

```

Beim Lauf von storeBackupUpdateBackup.pl, welches die fehlenden Hard Links erzeugt bzw. erzeugen will kommen dann die Fehlermeldungen, weil die Dateien, auf die gelinkt werden soll, nicht (mehr) vorhanden sind:

```

1 $ storeBackupUpdateBackup.pl -b test/backup
2 BEGIN 2022.04.05 09:51:17 134713 checking references and backup copying in <test/backup>
3 VERSION 2022.04.05 09:51:17 134713 storeBackupUpdateBackup.pl, 3.5.2 pre
4 INFO 2022.04.05 09:51:17 134713 creating lock file </home/hjc/storeBackup.lock>
5 INFO 2022.04.05 09:51:17 134713 scanning directory <test/backup> for existing backups
6 INFO 2022.04.05 09:51:17 134713 scanning directory <test/backup/series1> for existing
↪ backups
7 STATISTIC 2022.04.05 09:51:17 134713 found 1 backup series, 2 backups, 0 renamed backups
8 INFO 2022.04.05 09:51:17 134713 consistency check finished successfully
9 INFO 2022.04.05 09:51:17 134713 listing references:
10 INFO 2022.04.05 09:51:17 134713 /tmp/test/backup/series1/2022.04.05_09.42.40
11 INFO 2022.04.05 09:51:17 134713 -> /tmp/test/backup/series1/2022.04.05_09.08.26
12 INFO 2022.04.05 09:51:17 134713 (1/1) updating
↪ </tmp/test/backup/series1/2022.04.05_09.08.26>
13 INFO 2022.04.05 09:51:17 134713 phase 1: mkdir, symlink and compressing files
14 STATISTIC 2022.04.05 09:51:17 134713 created 1 directories
15 STATISTIC 2022.04.05 09:51:17 134713 created 0 symbolic links
16 STATISTIC 2022.04.05 09:51:17 134713 compressed 0 files
17 STATISTIC 2022.04.05 09:51:17 134713 used 0.0 instead of 0.0 (0 <- 0 ; 0.0%)
18 INFO 2022.04.05 09:51:17 134713 phase 2: setting hard links
19 ERROR 2022.04.05 09:51:17 134713 cannot open
↪ </tmp/test/backup/series1/2022.04.05_09.08.26/./1> for reading and copying to
↪ </tmp/test/backup/series1/2022.04.05_09.08.26/sub1/1.1>
20 ERROR 2022.04.05 09:51:17 134713 could not link/copy
↪ /tmp/test/backup/series1/2022.04.05_09.08.26/./1
↪ /tmp/test/backup/series1/2022.04.05_09.08.26/sub1/1.1
21 STATISTIC 2022.04.05 09:51:17 134713 linked 0 files

```

```

22 INFO      2022.04.05 09:51:17 134713 phase 3: setting file permissions
23 ERROR      2022.04.05 09:51:17 134713 cannot access
    ↳ </tmp/test/backup/series1/2022.04.05_09.08.26/1>
24 ERROR      2022.04.05 09:51:17 134713 cannot access
    ↳ </tmp/test/backup/series1/2022.04.05_09.08.26/sub1/1.1>
25 ERROR      2022.04.05 09:51:17 134713 cannot access
    ↳ </tmp/test/backup/series1/2022.04.05_09.08.26/sub1/ls.bz2>
26 STATISTIC 2022.04.05 09:51:17 134713 set permissions for 1 files
27 INFO      2022.04.05 09:51:17 134713 phase 4: setting directory permissions
28 STATISTIC 2022.04.05 09:51:17 134713 set permissions for 1 directories
29 INFO      2022.04.05 09:51:17 134713 scanning directory <test/backup> for existing backups
30 INFO      2022.04.05 09:51:17 134713 scanning directory <test/backup/series1> for existing
    ↳ backups
31 STATISTIC 2022.04.05 09:51:17 134713 found 1 backup series, 2 backups, 0 renamed backups
32 INFO      2022.04.05 09:51:17 134713 consistency check finished successfully
33 INFO      2022.04.05 09:51:17 134713 listing references:
34 INFO      2022.04.05 09:51:17 134713      /tmp/test/backup/series1/2022.04.05_09.42.40
35 INFO      2022.04.05 09:51:17 134713      -> /tmp/test/backup/series1/2022.04.05_09.08.26
36 INFO      2022.04.05 09:51:17 134713 (2/1) updating
    ↳ </tmp/test/backup/series1/2022.04.05_09.42.40>
37 INFO      2022.04.05 09:51:17 134713 phase 1: mkdir, symlink and compressing files
38 STATISTIC 2022.04.05 09:51:17 134713 created 1 directories
39 STATISTIC 2022.04.05 09:51:17 134713 created 0 symbolic links
40 STATISTIC 2022.04.05 09:51:17 134713 compressed 0 files
41 STATISTIC 2022.04.05 09:51:17 134713 used 0.0 instead of 0.0 (0 <- 0 ; 0.0%)
42 INFO      2022.04.05 09:51:17 134713 phase 2: setting hard links
43 ERROR      2022.04.05 09:51:17 134713 cannot open
    ↳ </tmp/test/backup/series1/2022.04.05_09.42.40/../../2022.04.05_09.08.26/sub1/1.1> for reading
    ↳ and copying to </tmp/test/backup/series1/2022.04.05_09.42.40/1>
44 ERROR      2022.04.05 09:51:17 134713 could not link/copy
    ↳ /tmp/test/backup/series1/2022.04.05_09.42.40/../../2022.04.05_09.08.26/sub1/1.1
    ↳ /tmp/test/backup/series1/2022.04.05_09.42.40/1
45 ERROR      2022.04.05 09:51:17 134713 cannot open
    ↳ </tmp/test/backup/series1/2022.04.05_09.42.40/../../2022.04.05_09.08.26/sub1/ls.bz2> for
    ↳ reading and copying to </tmp/test/backup/series1/2022.04.05_09.42.40/sub1/ls.bz2>
46 ERROR      2022.04.05 09:51:17 134713 could not link/copy
    ↳ /tmp/test/backup/series1/2022.04.05_09.42.40/../../2022.04.05_09.08.26/sub1/ls.bz2
    ↳ /tmp/test/backup/series1/2022.04.05_09.42.40/sub1/ls.bz2
47 ERROR      2022.04.05 09:51:17 134713 cannot open
    ↳ </tmp/test/backup/series1/2022.04.05_09.42.40/../../2022.04.05_09.08.26/sub1/1.1> for reading
    ↳ and copying to </tmp/test/backup/series1/2022.04.05_09.42.40/sub1/1.1>
48 ERROR      2022.04.05 09:51:17 134713 could not link/copy
    ↳ /tmp/test/backup/series1/2022.04.05_09.42.40/../../2022.04.05_09.08.26/sub1/1.1
    ↳ /tmp/test/backup/series1/2022.04.05_09.42.40/sub1/1.1
49 STATISTIC 2022.04.05 09:51:17 134713 linked 1 files
50 INFO      2022.04.05 09:51:17 134713 phase 3: setting file permissions
51 ERROR      2022.04.05 09:51:17 134713 cannot access
    ↳ </tmp/test/backup/series1/2022.04.05_09.42.40/1>
52 ERROR      2022.04.05 09:51:17 134713 cannot access
    ↳ </tmp/test/backup/series1/2022.04.05_09.42.40/sub1/ls.bz2>
53 ERROR      2022.04.05 09:51:17 134713 cannot access
    ↳ </tmp/test/backup/series1/2022.04.05_09.42.40/sub1/1.1>
54 STATISTIC 2022.04.05 09:51:17 134713 set permissions for 1 files
55 INFO      2022.04.05 09:51:17 134713 phase 4: setting directory permissions
56 STATISTIC 2022.04.05 09:51:17 134713 set permissions for 1 directories
57 INFO      2022.04.05 09:51:17 134713 scanning directory <test/backup> for existing backups
58 INFO      2022.04.05 09:51:17 134713 scanning directory <test/backup/series1> for existing
    ↳ backups
59 STATISTIC 2022.04.05 09:51:17 134713 found 1 backup series, 2 backups, 0 renamed backups
60 INFO      2022.04.05 09:51:17 134713 consistency check finished successfully
61 INFO      2022.04.05 09:51:17 134713 found no references to backups from lateLinks that need
    ↳ storeBackupUpdateBackup run

```

```

62 STATISTIC 2022.04.05 09:51:17 134713 duration = 1s
63 STATISTIC 2022.04.05 09:51:17 134713 [sec] | user| system
64 STATISTIC 2022.04.05 09:51:17 134713 -----+-----+-----
65 STATISTIC 2022.04.05 09:51:17 134713 process| 0.13| 0.04
66 STATISTIC 2022.04.05 09:51:17 134713 childs | 0.09| 0.00
67 STATISTIC 2022.04.05 09:51:17 134713 -----+-----+-----
68 STATISTIC 2022.04.05 09:51:17 134713 sum | 0.22| 0.04 => 0.26
69 INFO 2022.04.05 09:51:17 134713 syncing ...
70 ERROR 2022.04.05 09:51:17 134713 -- 14 ERRORS OCCURRED DURING THE BACKUP! --
71 END 2022.04.05 09:51:17 134713 checking references and copying in <test/backup>

```

Beim Blick auf die Dateien im Listing unten fallen die Dateien `.md5CheckSums.fail` auf, die von `storeBackup-UpdateBackup.pl` in das jeweils zu bearbeitende Backup-Verzeichnis geschrieben wird (Zeile 8 und Zeile 16):

```

1 $ find test -print | sort
2 test
3 test/backup
4 test/backup/series1
5 test/backup/series1/2022.04.05_09.08.26
6 test/backup/series1/2022.04.05_09.08.26/2
7 test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.bz2
8 test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.fail
9 test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.info
10 test/backup/series1/2022.04.05_09.08.26/.storeBackupLinks
11 test/backup/series1/2022.04.05_09.08.26/.storeBackupLinks/backup.Finished
12 test/backup/series1/2022.04.05_09.08.26/sub1
13 test/backup/series1/2022.04.05_09.42.40
14 test/backup/series1/2022.04.05_09.42.40/2
15 test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.bz2
16 test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.fail
17 test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.info
18 test/backup/series1/2022.04.05_09.42.40/.storeBackupLinks
19 test/backup/series1/2022.04.05_09.42.40/.storeBackupLinks/backup.Finished
20 test/backup/series1/2022.04.05_09.42.40/sub1 test/source1
21 test/source1/1 test/source1/2 test/source1/sub1 test/source1/sub1/1.1
22 test/source1/sub1/ls

```

Wie im folgenden Listing gezeigt enthalten sie die Prüfsummen der fehlenden Dateien sowie deren Namen, die dann von `storeBackup.pl` zur Identifikation und zum Ersatz der gelöschten Dateien verwendet werden:

```

1 $ cat test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.fail
2 b026324c6904b2a9cb4b88d6d61c81d1 1
3 b026324c6904b2a9cb4b88d6d61c81d1 sub1/1.1
4 6d2b4ff5fd937cd034aa2a2cf203e20f sub1/ls
5 $ cat test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.fail
6 b026324c6904b2a9cb4b88d6d61c81d1 1
7 6d2b4ff5fd937cd034aa2a2cf203e20f sub1/ls
8 b026324c6904b2a9cb4b88d6d61c81d1 sub1/1.1

```

Anschließend wird `storeBackup.pl` zum Antransport der fehlenden Dateien verwendet. Gelöschte Dateien, die nicht mehr „antransportiert“ werden (also im source Verzeichnis gelöscht wurden) können durch dieses Verfahren nicht ergänzt werden.

Ein anschließender Lauf von `storeBackup.pl` ergänzt die fehlenden Daten mit Unterstützung der Informationen in den `*.fail` Dateien:

```

1 $ storeBackup.pl -s test/source1 -b test/backup -S series1 --lateLinks
2 ...
3 INFO 2022.04.05 10:04:11 135148 scanning directory </tmp/test/backup> for existing backups
4 INFO 2022.04.05 10:04:11 135148 scanning directory </tmp/test/backup/series1> for existing
  ↳ backups
5 STATISTIC 2022.04.05 10:04:11 135148 found 1 backup series, 2 backups, 0 renamed backups
6 INFO 2022.04.05 10:04:11 135148 consistency check finished successfully
7 INFO 2022.04.05 10:04:11 135148 found no references to backups from lateLinks that need
  ↳ storeBackupUpdateBackup run
8 INFO 2022.04.05 10:04:11 135148 creating lock file </home/hjc/storeBackup.lock>

```

```

9  INFO      2022.04.05 10:04:11 135148 analysis of old Backups in </tmp/test/backup/series1>:
10 INFO      2022.04.05 10:04:11 135148      (2) Tue 2022.04.05_09.08.26 (0d0h): keepDuplicate(7d)
11 INFO      2022.04.05 10:04:11 135148      (1) Tue 2022.04.05_09.42.40 (0d0h): keepDuplicate(7d)
12 INFO      2022.04.05 10:04:11 135148      (0) Tue 2022.04.05_10.04.11 (0d0h): keepMinNumber1,
    ↪ keepWeekDays(30d)
13 INFO      2022.04.05 10:04:11 135148 start reading
    ↪ /tmp/test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.bz2
14 INFO      2022.04.05 10:04:11 135148 finished reading
    ↪ /tmp/test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.bz2 (5 entries)
15 INFO      2022.04.05 10:04:11 135148 4 entries in dbm files
16 INFO      2022.04.05 10:04:11 135148 0 entries in dbm block files
17 INFO      2022.04.05 10:04:11 135148 lateLinks is switched on
18 INFO      2022.04.05 10:04:11 135148 correcting </tmp/test/source1/1>
19 INFO      2022.04.05 10:04:11 135148 correcting </tmp/test/source1/sub1/ls>
20 INFO      2022.04.05 10:04:12 135148 correcting </tmp/test/source1/sub1/1.1>
21 INFO      2022.04.05 10:04:12 135148 syncing ...
22 STATISTIC 2022.04.05 10:04:12 135148 [sec] |      user|      system
23 STATISTIC 2022.04.05 10:04:12 135148 -----+-----+-----
24 STATISTIC 2022.04.05 10:04:12 135148 process|      0.16|      0.03
25 STATISTIC 2022.04.05 10:04:12 135148 childs |      0.16|      0.04
26 STATISTIC 2022.04.05 10:04:12 135148 -----+-----+-----
27 STATISTIC 2022.04.05 10:04:12 135148 sum    |      0.32|      0.07 => 0.39 (0s)
28 STATISTIC 2022.04.05 10:04:12 135148      directories = 1
29 STATISTIC 2022.04.05 10:04:12 135148      files = 4
30 STATISTIC 2022.04.05 10:04:12 135148      symbolic links = 0
31 STATISTIC 2022.04.05 10:04:12 135148      late links = 1
32 STATISTIC 2022.04.05 10:04:12 135148      named pipes = 0
33 STATISTIC 2022.04.05 10:04:12 135148      sockets = 0
34 STATISTIC 2022.04.05 10:04:12 135148      block devices = 0
35 STATISTIC 2022.04.05 10:04:12 135148      character devices = 0
36 STATISTIC 2022.04.05 10:04:12 135148      new internal linked files = 1
37 STATISTIC 2022.04.05 10:04:12 135148      old linked files = 1
38 STATISTIC 2022.04.05 10:04:12 135148      unchanged files = 1
39 STATISTIC 2022.04.05 10:04:12 135148      copied files = 1
40 STATISTIC 2022.04.05 10:04:12 135148      compressed files = 1
41 STATISTIC 2022.04.05 10:04:12 135148      blocked files = 0
42 STATISTIC 2022.04.05 10:04:12 135148      excluded files because rule = 0 (0.0 )
43 STATISTIC 2022.04.05 10:04:12 135148      included files because rule = 0 (0.0 )
44 STATISTIC 2022.04.05 10:04:12 135148      max size of copy queue = 1
45 STATISTIC 2022.04.05 10:04:12 135148 max size of compression queue = 1
46 STATISTIC 2022.04.05 10:04:12 135148      calculated md5 sums = 3
47 STATISTIC 2022.04.05 10:04:12 135148      forks total = 1
48 STATISTIC 2022.04.05 10:04:12 135148      forks md5 = 0
49 STATISTIC 2022.04.05 10:04:12 135148      forks copy = 0
50 STATISTIC 2022.04.05 10:04:12 135148      forks bzip2 = 1
51 STATISTIC 2022.04.05 10:04:12 135148      sum of source = 143k (146398)
52 STATISTIC 2022.04.05 10:04:12 135148      sum of target all = 63k (64076)
53 STATISTIC 2022.04.05 10:04:12 135148      sum of target all = 43.77%
54 STATISTIC 2022.04.05 10:04:12 135148      sum of target new = 63k (64072)
55 STATISTIC 2022.04.05 10:04:12 135148      sum of target new = 43.77%
56 STATISTIC 2022.04.05 10:04:12 135148      sum of md5ed files = 0.0 (0)
57 STATISTIC 2022.04.05 10:04:12 135148      sum of md5ed files = 0.00%
58 STATISTIC 2022.04.05 10:04:12 135148      sum internal linked (copy) = 2.0 (2)
59 STATISTIC 2022.04.05 10:04:12 135148      sum internal linked (compr) = 0.0 (0)
60 STATISTIC 2022.04.05 10:04:12 135148      sum old linked (copy) = 2.0 (2)
61 STATISTIC 2022.04.05 10:04:12 135148      sum old linked (compr) = 0.0 (0)
62 STATISTIC 2022.04.05 10:04:12 135148      sum unchanged (copy) = 2.0 (2)
63 STATISTIC 2022.04.05 10:04:12 135148      sum unchanged (compr) = 0.0 (0)
64 STATISTIC 2022.04.05 10:04:12 135148      sum new (copy) = 2.0 (2)
65 STATISTIC 2022.04.05 10:04:12 135148      sum new (compr) = 63k (64070)
66 STATISTIC 2022.04.05 10:04:12 135148      sum new (compr), orig size = 143k (146392)
67 STATISTIC 2022.04.05 10:04:12 135148      sum new / orig = 43.77%

```

```

68 STATISTIC 2022.04.05 10:04:12 135148      size of md5CheckSum file = 345  (345)
69 STATISTIC 2022.04.05 10:04:12 135148      size of temporary db files = 0.0  (0)
70 STATISTIC 2022.04.05 10:04:12 135148      deleted old backups = 0
71 STATISTIC 2022.04.05 10:04:12 135148      deleted directories = 0
72 STATISTIC 2022.04.05 10:04:12 135148      deleted files = 0
73 STATISTIC 2022.04.05 10:04:12 135148      (only) removed links = 0
74 STATISTIC 2022.04.05 10:04:12 135148 freed space in old directories = 0.0  (0)
75 STATISTIC 2022.04.05 10:04:12 135148      add. used space in files = 63k (64417)
76 STATISTIC 2022.04.05 10:04:12 135148      backup duration = 1s
77 STATISTIC 2022.04.05 10:04:12 135148 over all files/sec (real time) = 4.00
78 STATISTIC 2022.04.05 10:04:12 135148 over all files/sec (CPU time) = 10.26
79 STATISTIC 2022.04.05 10:04:12 135148      CPU usage = 39.00%
80 INFO      2022.04.05 10:04:12 135148 removing lock file </home/hjc/storeBackup.lock>
81 END      2022.04.05 10:04:12 135148 backing up directory </tmp/test/source1> to
    ↪ </tmp/test/backup/series1/2022.04.05_10.04.11>

```

Das folgende Listing zeigt, dass das letzte Backup wieder mit den gelöschten Dateien gefüllt ist (Zeile 22–32):

```

1  $ find test -print | sort
2  test
3  test/backup
4  test/backup/series1
5  test/backup/series1/2022.04.05_09.08.26
6  test/backup/series1/2022.04.05_09.08.26/2
7  test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.bz2
8  test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.fail
9  test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.info
10 test/backup/series1/2022.04.05_09.08.26/.storeBackupLinks
11 test/backup/series1/2022.04.05_09.08.26/.storeBackupLinks/backup.Finished
12 test/backup/series1/2022.04.05_09.08.26/sub1
13 test/backup/series1/2022.04.05_09.42.40
14 test/backup/series1/2022.04.05_09.42.40/2
15 test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.bz2
16 test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.fail
17 test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.info
18 test/backup/series1/2022.04.05_09.42.40/.storeBackupLinks
19 test/backup/series1/2022.04.05_09.42.40/.storeBackupLinks/backup.Finished
20 test/backup/series1/2022.04.05_09.42.40/.storeBackupLinks/linkFrom0
21 test/backup/series1/2022.04.05_09.42.40/sub1
22 test/backup/series1/2022.04.05_10.04.11
23 test/backup/series1/2022.04.05_10.04.11/1
24 test/backup/series1/2022.04.05_10.04.11/.md5CheckSums.bz2
25 test/backup/series1/2022.04.05_10.04.11/.md5CheckSums.info
26 test/backup/series1/2022.04.05_10.04.11/.storeBackupLinks
27 test/backup/series1/2022.04.05_10.04.11/.storeBackupLinks/backup.Finished
28 test/backup/series1/2022.04.05_10.04.11/.storeBackupLinks/linkFile.bz2
29 test/backup/series1/2022.04.05_10.04.11/.storeBackupLinks/linkTo
30 test/backup/series1/2022.04.05_10.04.11/sub1
31 test/backup/series1/2022.04.05_10.04.11/sub1/1.1
32 test/backup/series1/2022.04.05_10.04.11/sub1/ls.bz2
33 test/source1
34 test/source1/1
35 test/source1/2
36 test/source1/sub1
37 test/source1/sub1/1.1
38 test/source1/sub1/ls

```

Zum Schluss setzt der Lauf von storeBackupUpdateBackup.pl die noch die Hard Links, wie am Ergebnis zu sehen ist:

```

1  $ storeBackupUpdateBackup.pl -b test/backup/
2  BEGIN      2022.04.05 10:13:13 135454 checking references and backup copying in <test/backup>
3  VERSION    2022.04.05 10:13:13 135454 storeBackupUpdateBackup.pl, 3.5.2 pre

```

```

4 INFO      2022.04.05 10:13:13 135454 creating lock file </home/hjc/storeBackup.lock>
5 INFO      2022.04.05 10:13:13 135454 scanning directory <test/backup> for existing backups
6 INFO      2022.04.05 10:13:13 135454 scanning directory <test/backup/series1> for existing
  ↳ backups
7 STATISTIC 2022.04.05 10:13:13 135454 found 1 backup series, 3 backups, 0 renamed backups
8 INFO      2022.04.05 10:13:13 135454 consistency check finished successfully
9 INFO      2022.04.05 10:13:13 135454 listing references:
10 INFO     2022.04.05 10:13:13 135454 /tmp/test/backup/series1/2022.04.05_10.04.11
11 INFO     2022.04.05 10:13:13 135454 -> /tmp/test/backup/series1/2022.04.05_09.42.40
12 INFO     2022.04.05 10:13:13 135454 (1/1) updating
  ↳ </tmp/test/backup/series1/2022.04.05_10.04.11>
13 INFO     2022.04.05 10:13:13 135454 phase 1: mkdir, symlink and compressing files
14 STATISTIC 2022.04.05 10:13:13 135454 created 1 directories
15 STATISTIC 2022.04.05 10:13:13 135454 created 0 symbolic links
16 STATISTIC 2022.04.05 10:13:13 135454 compressed 0 files
17 STATISTIC 2022.04.05 10:13:13 135454 used 0.0 instead of 0.0 (0 <- 0 ; 0.0%)
18 INFO     2022.04.05 10:13:13 135454 phase 2: setting hard links
19 -0-/tmp/test/backup/series1/2022.04.05_10.04.11
20 STATISTIC 2022.04.05 10:13:13 135454 linked 1 files
21 INFO     2022.04.05 10:13:13 135454 phase 3: setting file permissions
22 STATISTIC 2022.04.05 10:13:13 135454 set permissions for 4 files
23 INFO     2022.04.05 10:13:13 135454 phase 4: setting directory permissions
24 STATISTIC 2022.04.05 10:13:13 135454 set permissions for 1 directories
25 -4-
26 INFO     2022.04.05 10:13:14 135454 scanning directory <test/backup> for existing backups
27 INFO     2022.04.05 10:13:14 135454 scanning directory <test/backup/series1> for existing
  ↳ backups
28 STATISTIC 2022.04.05 10:13:14 135454 found 1 backup series, 3 backups, 0 renamed backups
29 INFO     2022.04.05 10:13:14 135454 consistency check finished successfully
30 INFO     2022.04.05 10:13:14 135454 found no references to backups from lateLinks that need
  ↳ storeBackupUpdateBackup run
31 STATISTIC 2022.04.05 10:13:14 135454 duration = 1s
32 STATISTIC 2022.04.05 10:13:14 135454 [sec] | user| system
33 STATISTIC 2022.04.05 10:13:14 135454 -----+-----+-----
34 STATISTIC 2022.04.05 10:13:14 135454 process| 0.13| 0.02
35 STATISTIC 2022.04.05 10:13:14 135454 childs | 0.08| 0.01
36 STATISTIC 2022.04.05 10:13:14 135454 -----+-----+-----
37 STATISTIC 2022.04.05 10:13:14 135454 sum | 0.21| 0.03 => 0.24
38 $ find test -print | sort
39 test
40 test/backup
41 test/backup/series1
42 test/backup/series1/2022.04.05_09.08.26
43 test/backup/series1/2022.04.05_09.08.26/2
44 test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.bz2
45 test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.fail
46 test/backup/series1/2022.04.05_09.08.26/.md5CheckSums.info
47 test/backup/series1/2022.04.05_09.08.26/.storeBackupLinks
48 test/backup/series1/2022.04.05_09.08.26/.storeBackupLinks/backup.Finished
49 test/backup/series1/2022.04.05_09.08.26/sub1
50 test/backup/series1/2022.04.05_09.42.40
51 test/backup/series1/2022.04.05_09.42.40/2
52 test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.bz2
53 test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.fail
54 test/backup/series1/2022.04.05_09.42.40/.md5CheckSums.info
55 test/backup/series1/2022.04.05_09.42.40/.storeBackupLinks
56 test/backup/series1/2022.04.05_09.42.40/.storeBackupLinks/backup.Finished
57 test/backup/series1/2022.04.05_09.42.40/sub1
58 test/backup/series1/2022.04.05_10.04.11
59 test/backup/series1/2022.04.05_10.04.11/1
60 test/backup/series1/2022.04.05_10.04.11/2
61 test/backup/series1/2022.04.05_10.04.11/.md5CheckSums.bz2

```



```

62 test/backup/series1/2022.04.05_10.04.11/.md5CheckSums.info
63 test/backup/series1/2022.04.05_10.04.11/.storeBackupLinks
64 test/backup/series1/2022.04.05_10.04.11/.storeBackupLinks/backup.Finished
65 test/backup/series1/2022.04.05_10.04.11/sub1
66 test/backup/series1/2022.04.05_10.04.11/sub1/1.1
67 test/backup/series1/2022.04.05_10.04.11/sub1/ls.bz2
68 test/source1
69 test/source1/1
70 test/source1/2
71 test/source1/sub1
72 test/source1/sub1/1.1
73 test/source1/sub1/ls
74 INFO      2022.04.05 10:13:14 135454 syncing ...
75 END       2022.04.05 10:13:14 135454 checking references and copying in <test/backup>

```

Zur Überprüfung des Beispiels lassen wir noch storeBackupCheckBackup.pl laufen. Wie zu sehen ist sind im letzten Backup (siehe Zeile 21) keine Fehler (mehr):

```

1  $ storeBackupCheckBackup.pl -c test/backup/
2  BEGIN      2022.04.05 10:17:25 135613 checking backups in </tmp/test/backup>
3  VERSION    2022.04.05 10:17:25 135613 storeBackupCheckBackup.pl, 3.5.2 pre
4  INFO       2022.04.05 10:17:25 135613 scanning directory <test/backup> for existing backups
5  INFO       2022.04.05 10:17:25 135613 scanning directory <test/backup/series1> for existing
    ↳ backups
6  STATISTIC  2022.04.05 10:17:25 135613 found 1 backup series, 3 backups, 0 renamed backups
7  INFO       2022.04.05 10:17:25 135613 consistency check finished successfully
8  INFO       2022.04.05 10:17:25 135613 found no references to backups from lateLinks that need
    ↳ storeBackupUpdateBackup run
9  INFO       2022.04.05 10:17:25 135613 backup directories to check
10 INFO       2022.04.05 10:17:25 135613 /tmp/test/backup/series1/2022.04.05_09.08.26
11 INFO       2022.04.05 10:17:25 135613 /tmp/test/backup/series1/2022.04.05_09.42.40
12 INFO       2022.04.05 10:17:25 135613 /tmp/test/backup/series1/2022.04.05_10.04.11
13 INFO       2022.04.05 10:17:25 135613 -- checking </tmp/test/backup/series1/2022.04.05_09.08.26>
    ↳ ...
14 ERROR      2022.04.05 10:17:25 135613 file </tmp/test/backup/series1/2022.04.05_09.08.26/1> is
    ↳ missing
15 ERROR      2022.04.05 10:17:25 135613 file
    ↳ </tmp/test/backup/series1/2022.04.05_09.08.26/sub1/1.1> is missing
16 ERROR      2022.04.05 10:17:25 135613 file
    ↳ </tmp/test/backup/series1/2022.04.05_09.08.26/sub1/ls.bz2> is missing
17 INFO       2022.04.05 10:17:25 135613 -- checking </tmp/test/backup/series1/2022.04.05_09.42.40>
    ↳ ...
18 ERROR      2022.04.05 10:17:25 135613 file </tmp/test/backup/series1/2022.04.05_09.42.40/1> is
    ↳ missing
19 ERROR      2022.04.05 10:17:25 135613 file
    ↳ </tmp/test/backup/series1/2022.04.05_09.42.40/sub1/ls.bz2> is missing
20 ERROR      2022.04.05 10:17:25 135613 file
    ↳ </tmp/test/backup/series1/2022.04.05_09.42.40/sub1/1.1> is missing
21 INFO       2022.04.05 10:17:25 135613 -- checking </tmp/test/backup/series1/2022.04.05_10.04.11>
    ↳ ...
22 INFO       2022.04.05 10:17:26 135613 -- no WARNINGS OCCURRED DURING THE CHECK! --
23 ERROR      2022.04.05 10:17:26 135613 -- 6 ERRORS OCCURRED DURING THE CHECK! --

```

Im Backup fehlende Dateien werden also beim nächsten Backup (hier mit lateLinks) *in der Kombination* von storeBackup.pl und storeBackupUpdateBackup.pl wieder im Backup ergänzt. Das Verhalten ähnelt unter dieser Voraussetzung dem dem *ohne* lateLinks. Ist diese Kombination nicht gegeben, findet auch kein „Nachkopieren“ statt.

Bisher (Version 3.5.2) ist das hier beschriebne Verhalten nur für non-block-Dateien implementiert.

7.11 Konfiguration von NFS

Nehmen wir an, dass Dein Server, auf den Du Deine Backups über NFS schreiben willst, „nfserver“ heißt und der Pfad zum Backup /storeBackup ist. Dann kannst Du den folgenden Eintrag in /etc/exports auf

`nfsserver` verwenden (Beispiel für GNU/Linux, kann auf anderen unixoiden Betriebssystemen anders aussehen):

```
1 /storeBackup 192.168.1.0/24(async,rw,no_root_squash)
```

192.168.1.0/24 bedeutet, dass der Zugriff von jeder IP-Adresse, die mit 192.168.1 anfängt, erlaubt ist.

Dann startest Du:

```
1 # exportfs -a
```

um die Einträge dem NFS bekannt zu machen. Mit

```
1 # exportfs -v
```

kannst Du überprüfen, wie das NFS konfiguriert ist.

Du musst wahrscheinlich die IP-Adresse und eventuell die Netzmaske auf Deine Situation anpassen. Die Verwendung von `no_root_squash` ist wichtig für root-User auf dem Client. So bekommt root vom Client auf dem gemounteten Dateisystem auch root-Rechte. Die Verwendung von `async` bringt für die NFS-gemounteten Dateisysteme eine deutlich bessere Performance (siehe auch `man mount` für weitere Erläuterungen). Wenn Du `async` verwendest, ist `storeBackup` nicht mehr in der Lage festzustellen, ob das Dateisystem vollläuft.

In `/etc/fstab` auf dem NFS Client (wo `storeBackup.pl` läuft), solltest Du konfigurieren:

```
1 nfsserver:/storeBackup /backup nfs user,exec,async,noatime 1 1
```

Dieses mountet das Dateisystem `/storeBackup` auf `nfsserver` als `/backup` auf dem Client. Dies passiert beim Booten oder wenn Du

```
1 # mount /backup
```

auf dem NFS Client eingibst.

Es gibt viele andere NFS Optionen. Diese kurze Beschreibung versucht nur, einige nützliche Hinweise zu geben, nicht, NFS zu erklären.

Lese- oder Schreibzugriff?

Du wirst sicherlich Schreibzugriff auf das Backup für alle Anwender von `storeBackup.pl` gewähren, jedoch nur Lesezugriff für die Benutzer. Es gibt mindestens zwei Wege, das zu erreichen:

1. Monte das betreffende NFS Verzeichnis für das Backup (z. B. `/backup`) nur lesbar (read only). Schreibe in die Konfigurationsdatei von `storeBackup.pl`:

```
1 precommand = mount /backup -o remount,rw
2 postcommand = mount /backup -o remount,ro
```

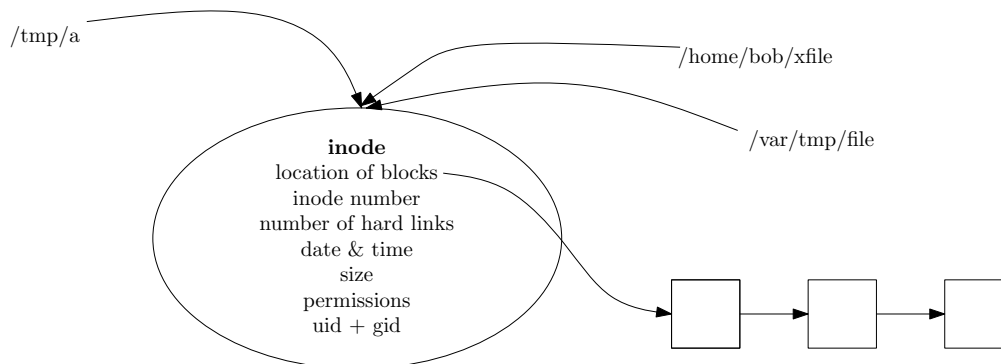
Dies gibt `storeBackup.pl` Schreibrechte (rw = read write) während des Backups. Natürlich kannst Du auch ein Skript schreiben, dass dieses vor und nach `storeBackup.pl` ausführt und damit dasselbe bewirkt.

Der Nachteil dieser Methode ist, dass die Anwender während des Backups auch Schreibzugriff auf die Backups haben.

2. Erzeuge zwei Einhängpunkte für den NFS-Server: einen rw und einen ro. Beschränke den Zugriff auf den rw-Mount auf root. Du kannst die Mount-Punkte für `storeBackup.pl` auch nur für die Zeit des Backup mounten. Hierfür kannst Du `storeBackupMount.pl`, siehe Kapitel 6.11 verwenden.

7.12 Was ist ein Inode

In Unix-ähnlichen Umgebungen (wie Linux), ist das zentrale Element für einen Dateisystem-Eintrag der Inode (index node):



Dieser Inode beinhaltet verschiedene Metadaten-Informationen und die Lokation der Datei auf der Platte (Sektoren). (Der Teil des Bildes zeigt absolut nicht die wirkliche Situation.)

In der Abbildung oben sieht man drei Hardlinks („Dateinamen“) für diesen Inode, daher ist die Anzahl der Hardlinks für ihn 3. Dieses wird mit `ls -li` sichtbar:

```
$ ls -li /tmp/a /home/bob/xfile /var/tmp/file
114693 -rw-r--r-- 3 hjc hjc 5 Sep 6 13:55 /tmp/a
114693 -rw-r--r-- 3 hjc hjc 5 Sep 6 13:55 /home/bob/xfile
114693 -rw-r--r-- 3 hjc hjc 5 Sep 6 13:55 /var/tmp/file
```

Die erste Zahl ist die Inode Nummer.

Natürlich müssen sich alle Hardlinks auf einen Inode in demselben Dateisystem befinden. Wenn die unterschiedlichen Verzeichnisse in dem Beispiel nicht im selben Dateisystem wären, könntest Du dieses Beispiel nicht exakt so nachbilden. Um von der Shell einen Hardlink zu erzeugen, verwende das Kommando `ln`.

Alle Einträge (Links) für einen Inode haben dieselben Rechte, GID und UID. Siehe Kapitel 7.15, Limitierungen, um zu verstehen, was das für storeBackup bedeutet.

Nebenbei: Du kannst keinen Hardlink auf ein Verzeichnis erzeugen, weil dies eine Endlosschleife erzeugen könnte. Nichtsdestotrotz nutzen Verzeichnisse ebenfalls das Feature von mehreren Links (Namen) auf einen Inode: Sieh auf den Verzeichnis-„Namen“, „.“ und „..“.

Die maximal mögliche Anzahl der Hardlinks auf einen Inode hängt vom verwendeten Dateisystem (z.B. `ext3`, `ext4`, `btrfs`) ab. StoreBackup verwendet generell ein einfaches Verfahren um mit dieser Begrenzung umzugehen: Wenn es nicht mehr möglich ist, einen weiteren Hardlink auf einen Inode zu erzeugen, wird die Datei neu kopiert und weitere Hardlinks werden dann auf diese Datei erzeugt. Hierdurch spielt die Anzahl der unterstützten Hardlinks auf dem jeweiligen Dateisystem keine Rolle.

Dieses Verhalten trifft aber *nicht* auf die Standard-Kommandos eines Linux oder Unix-Systems zu, die Du vielleicht zum Kopieren von Backups verwenden willst. Aus diesem Grund kann die maximal verwendete Anzahl von Hardlinks mit Hilfe der Option `maxHardLinks` bei den Programmen `storeBackup.pl`, `storeBackupUpdateBackup.pl` und `linkToDirs.pl` begrenzt werden.

7.13 Bedeutung der Statistikausgaben von storeBackup.pl

Nach dem Erzeugen eines neuen Backups und möglicherweise dem Löschen von alten zeigt `storeBackup.pl` einige statistische Werte an:

directories Anzahl der Verzeichnisse, die `storeBackup.pl` im Quellverzeichnis gelesen hat.

files Anzahl der Dateien (genauer Links), die `storeBackup.pl` im Quellverzeichnis gefunden hat. Dieser Wert beinhaltet alle Typen von Dateien, die `storeBackup.pl` zu lesen konfiguriert ist.

symbolic links Anzahl der symbolischen Links, die im Quellverzeichnis gefunden wurden.

name pipes Anzahl der named pipes, die im Quellverzeichnis gefunden wurden.

new internal linked files Anzahl der Dateien mit bereits ermitteltem Inhalt, die im aktuellen Backup (aber nicht in vorherigen) gefunden wurden.

old linked files Anzahl der Dateien, die in den referenzierten Backups existieren und bei denen Name, Größe, ctime und mtime gleich sind

unchanged files Anzahl der Dateien, die mit demselben Inhalt in den alten Backups gefunden wurden

copied files Anzahl Dateien mit neuem Inhalt, die ins Backup Verzeichnis kopiert wurden.

compressed files Anzahl Dateien mit neuem Inhalt, die ins Backup-Verzeichnis komprimiert wurden.

excluded files because pattern Anzahl Dateien, die aufgrund von `exceptPattern` ausgeschlossen wurden

included files because pattern Anzahl Dateien, die aufgrund von `includePattern` gesichert wurden

max size of copy queue Maximale Größe der Kopier-Warteschlange während des Backups

max size of compress queue Maximale Größe der Komprimier-Warteschlange während des Backups

calculated md5-Sums Anzahl der Dateien, für die eine md5-Summe berechnet wurde

forks total Summe aller Forks (Anzahl md5-Forks, Komprimierungs-Forks, Kopier-Forks, Forks für named pipes)

forks md5 Anzahl Forks für das Programm md5sum.

forks copy Anzahl Forks für das Programm cp

forks <compress> Anzahl Forks für das Programm `compress`

sum of source Anzahl Bytes im Quellverzeichnis

sum of target all Anzahl Bytes im Backup-Verzeichnis

sum of target new Anzahl Bytes von neu kopierten oder komprimierten Dateien im Backup-Verzeichnis

sum of md5ed files Anzahl Bytes aller Dateien, für die eine md5-Summe berechnet wurde

sum internal linked (copy) Anzahl Bytes für alle Dateien, die im Backup intern verlinkt wurden (siehe: new internal linked files), d.h. Links auf in dieses Backup kopierte Dateien.

sum internal linked (compr) Anzahl Bytes für alle Dateien, die im Backup intern verlinkt wurden (siehe: new internal linked files), d.h. Links auf in dieses Backup komprimierte Dateien.

sum old linked (copy) Anzahl Bytes aller Dateien, die zu älteren Dateien verlinkt wurden (siehe: old linked files). Hierbei werden die Dateien gezählt, die gegen ins Backup kopierte Dateien gelinkt wurden.

sum old linked (compr) Anzahl Bytes aller Dateien, die zu älteren Dateien verlinkt wurden (siehe: old linked files). Hierbei werden die Dateien gezählt, die gegen ins Backup komprimierte Dateien gelinkt wurden.

sum unchanged (copy) Anzahl Bytes aller Dateien, die in vorherigen Backups mit demselben Namen, Größe und Zeitstempel existieren und die nicht komprimiert wurden. Diese werden direkt mit denen in älteren Backups verlinkt.

sum unchanged (compr) Anzahl Bytes aller Dateien, die in vorherigen Backups mit demselben Namen, Größe und Zeitstempel existieren und die komprimiert wurden. Diese werden direkt mit denen in älteren Backups verlinkt.

sum new (copy) Anzahl Bytes aller Dateien, die ins Backup kopiert wurden.

sum new (compr) Anzahl Bytes aller Dateien, die ins Backup komprimiert wurden.

sum new (compr), orig size Größe der Dateien im Quellverzeichnis

sum new / orig Verhältnis der Größe der neuen Dateien im Backup zum Quellverzeichnis

size of md5Checksum file Größe der Datei `<backupDir>/md5Checksums[bz2]`

size of temporary db files Größe der Indexdateien, die während des Backups in `tmpdir` bestanden

deleted old backups Anzahl der gelöschten alten Backups

deleted directories Anzahl der Verzeichnisse, die in alten Backups gelöscht wurden

deleted files Anzahl der Dateien, die in alten Backups wirklich gelöscht wurden (entfernen des letzten Links)

(only) removed links Anzahl der Links, die in alten Backup gelöscht wurden (Dateien existieren noch)

freed space in old directories Freigewordener Platz durch Löschen von alten Backups (ohne Meta-Informationen)

add. used space in files Zusätzlich benötigter Platz für dieses Backup: Differenz zwischen neu benötigten Platz und frei gewordenem Platz von gelöschten alten Backups.

backup duration Dauer des Backups: Zeit für `precommand`, Backup, `postcommand` und Löschen alter Backups.

over all files/sec (real time) Anzahl Dateien geteilt durch reale Dauer

over all files/sec (CPU time) Anzahl Dateien geteilt durch (User- und Systemzeit) = Dateien pro CPU Sekunden

CPU usage Mittlere CPU Auslastung für die Dauer des Backups

PROGRESS 2009.05.09 10:16:43 22774 5000 files processed (324M, 152M) (340234099, 159903981)

Fortschrittsanzeige:

`storeBackup.pl` las bisher 5000 Dateien. Die erste Zahl (324M oder 340234099 Bytes) ist die Gesamtgröße aller bisher gelesenen Dateien im Quellverzeichnis. Die zweite Zahl (152M oder 159903981 Bytes) ist der Platz, der bisher im Backup belegt wurde. Dieser Wert hängt von den bisher kopierten Daten sowie den Effekten der Kompression und der Hardlinks ab.

7.14 Monitoring

Wenn Du Deine Backups monitoren willst, kannst Du einfach mittels `grep` nach `^ERROR` und `/` oder `^WARNING` suchen. Der Start eines Programms (welches Log Dateien schreibt) beginnt typischerweise mit dem Schreiben einer `BEGIN`- und endet mit einer `END`-Meldung, sofern keine Fehler auftraten, die das Programm direkt beendeten.

Wenn Du Deine Systeme mit Nagios oder Icinga überwachst, kannst Du ein Plugin von <http://exchange.nagios.org/directory/Plugins/Backup-and-Recovery/storeBackup> verwenden.

7.15 Limitierungen

- `storeBackup` kann normale Dateien, Verzeichnisse, symbolische Links und named pipes sichern. Andere Dateitypen können nur gesichert werden, wenn `cpIsGnu` gesetzt wird (und wenn `gnu cp` auf dem System installiert ist).
- Die Rechte im Backup-Verzeichnisbaum sind gleich denen im Quellverzeichnis. Unter besonderen, seltenen Bedingungen ist es möglich, dass ein Anwender eine oder mehrere seiner Dateien nicht lesen kann, weil sie mit denen anderer User verlinkt sind. Mit dem Rücksicherungstool wird alles beim Zurücksichern mit den richtigen Rechten versehen (`storeBackupRecover.pl`).
- `storeBackup` verwendet Hardlinks, um Plattenplatz zu sparen. GNU/Linux unterstützt mit dem `ext2`-Dateisystem bis zu 32000, `reiserfs` bis zu 64535 Hardlinks auf einem 32 Bit Betriebssystem. Wenn `storeBackup` mehr Hardlinks benötigt, speichert es eine neue Kopie (ggf. komprimiert) der Datei. Bei Verwendung von `ext2` (oder `ext3`, `ext4`) könntest Du in die Limitierung der statischen maximalen Anzahl von Inodes laufen (`btrfs` und `reiserfs` verwalten die Anzahl der Inodes dagegen dynamisch). (Es wird ein Inode für jede unterschiedliche Datei im Backup benötigt, *nicht* für jeden Hardlink.)

- Eine Änderung des Komprimierungsprogramms wird bisher nicht unterstützt. Innerhalb eines Backup-Verzeichnisses (`backupDir`) sollte überall dasselbe Komprimierungsprogramm verwendet werden.

8 Interna

8.1 Vorbemerkung

Manchmal erreichen mich Fragen wie „Kann ich Serien verschieben?“ oder „Was passiert, wenn das Löschen eines Backups abgebrochen wird?“ und ähnliche. Die meisten Antworten auf diese Fragen sind sehr einfach zu geben, wenn man die Interna von `storeBackup` versteht. Daher denke ich, es ist an der Zeit zu erklären, wie `storeBackup` arbeitet und auf welchen Ideen es basiert.

Wenn Du nicht mit grundlegenden Unix / Linux Dateisystem-Konzepten sowie einfachen Konsolen-Kommandos vertraut bist, dann könntest Du Probleme damit haben, dieses Kapitel zu verstehen. In diesem Fall wäre es wahrscheinlich besser, der Vorgehensweise nicht direkt zu folgen, sondern sie erst in einer Testumgebung nachzuvollziehen.

8.2 Vollständige Backups

Dieses Kapitel beschreibt die Abhängigkeiten (oder auch nicht vorhandenen Abhängigkeiten), wenn alle Backups vollständig erzeugt wurden. Das bedeutet, dass keinerlei „late-links“ zu erzeugen und das alle Replikationen abgeschlossen sind. Um ein Backup, mit „lateLinks“ oder Replikationen zu vervollständigen musst Du `storeBackupUpdateBackup.pl` verwenden. Wenn Du während der Erstellung abgebrochene Backups hast, sind diese einfach defekt und daher auf ihre Weise auch „fertig“.⁶²

Ein einfaches, erzeugtes Backup (ohne Replikation, ohne lateLinks) ist nichts als ein Verzeichnis mit Dateien darin. Es befindet sich in `backupDir/series/timestamp`. Die Dateien in dem Backup *können* per Hardlink mit anderen Dateien im selben Verzeichnis oder mit Dateien irgendwo anders (normalerweise andere Backups) verbunden sein. Das Verwenden von Hardlinks hat *nichts* mit storeBackups Funktionalität zu tun. (Natürlich versucht `storeBackup` Hardlinks zu verwenden um Platz zu sparen (Deduplikation), aber mehr bedeuten sie nicht.) Machen wir folgendes Gedankenexperiment: Du hast zwei Dateien im Backup, die durch Hardlinks verbunden sind, und löscht eine davon. Dann kopierst Du die verbliebene so, dass sie mit derselben Kombination aus Pfad / Dateiname den Platz der anderen einnimmt. Das Resultat sind zwei Dateien mit demselben Inhalt anstelle von einer Datei (Inode) mit zwei Hardlinks. Diese (nutzlose) Änderung im Backup hat für `storeBackup` keine Bedeutung; es ist nicht einmal in der Lage, die Änderung zu erkennen.

Es ist folglich sehr einfach: `storeBackup` weiß nichts davon und kümmert sich auch nicht darum, ob (identische) Dateien mit Hardlinks verbunden sind oder nicht. Das ist der Grund dafür, warum Du (vollständige) Backups (ohne lateLinks Referenzen auf sie) löschen kannst, wie Du willst (das Löschen betrifft nur die Anzahl der Hardlinks in anderen Backups) oder warum Du Deine Backups auf andere Dateisysteme mit weniger oder mehr unterstützten Hardlinks pro Datei mit `linkToDirs.pl` kopieren kannst. Die Hardlinks werden ausschließlich durch das Dateisystem verwaltet, nicht durch `storeBackup`.

Aber `storeBackup` hat andere Informationen über das Backup. In jedem Backup gibt es eine Datei namens `.md5Checksums.bz2` (eventuell nicht komprimiert). Sie hat folgende Struktur:

```

1 # contents/md5 compr dev-inode inodeBackup ctime mtime atime size uid gid mode filename
2 dir 0 18-139259 0 1376114212 1376114212 1376114500 0 1049 1049 509 sub1
3 c5f89e40c144b6fb8b61f2ef72e4b556 c 18-141517 148481 1376114209 1376114209 1376114500 31400 1049 1049 493
4 ↪ pwd
5 c5f89e40c144b6fb8b61f2ef72e4b556 c 18-141521 148481 1376114212 1376114212 1376114500 31400 1049 1049 493
6 ↪ sub1/pwd
7 b5607b4dc7d896c0fab5c4a308239161 c 18-141513 147606 1376114202 1376114202 1376114500 110088 1049 1049 493
8 ↪ ls
9 b5607b4dc7d896c0fab5c4a308239161 c 18-141516 147606 1376114205 1376114205 1376114500 110088 1049 1049 493
10 ↪ sub1/ls
```

⁶²Bis einschließlich Version 3.4.3 kann ein abgebrochenes Backup über die Flag-Datei `.md5Checksums.notfinished` identifiziert werden. Seit Version 3.5 wird ein abgebrochenes Backup über das Fehlen von `.storeBackupLinks/backup.Finished` erkannt.

Dies ist die Ausgabe eines kleinen Beispiels. In der ersten Spalte siehst Du das Schlüsselwort `,dir'` mit der Bedeutung, dass die in der Zeile beschriebenen Elemente Verzeichnisse sind. Darunter siehst Du die md5-Summen der Dateien `pwd` und `ls` in verschiedenen Unterverzeichnissen. Du stellst auch fest, dass einige der md5 Summen identisch sind – es handelt sich also um Kopien derselben Dateien in unterschiedlichen Verzeichnissen. Es gibt noch eine weitere Datei, `.md5CheckSums.info`.⁶³

```

1 version=1.3
2 date=2013.08.10 08.04.29
3 sourceDir='/tmp/a/s'
4 followLinks=0
5 compress='bzip2'
6 uncompress='bzip2' '-d'
7 postfix='.bz2'
8 comprRule='$size > 1024 and' 'not $file =~
  ↳ /\.zip\Z|\.bz2\Z|\.gz\Z|\.tgz\Z|\.jpg\Z|\.gif\Z|\.tiff\Z|\.mpe?g\Z|\.mp[34]\Z|\.mpe?[34]\Z|\.ogg\Z|\.gpg\Z/i'
9 exceptDirs=
10 includeDirs=
11 exceptRule=
12 includeRule=
13 writeExcludeLog=no
14 exceptTypes=
15 archiveTypes=
16 specialTypeArchiver=cpio
17 checkBlocksRule0=
18 checkBlocksBS0=
19 checkBlocksCompr0=
20 checkBlocksRead0=
21 checkBlocksRule1=
22 checkBlocksBS1=
23 checkBlocksCompr1=
24 checkBlocksRead1=
25 checkBlocksRule2=
26 checkBlocksBS2=
27 checkBlocksCompr2=
28 checkBlocksRead2=
29 checkBlocksRule3=
30 checkBlocksBS3=
31 checkBlocksCompr3=
32 checkBlocksRead3=
33 checkBlocksRule4=
34 checkBlocksBS4=
35 checkBlocksCompr4=
36 checkBlocksRead4=
37 preservePerms=yes
38 lateLinks=no
39 lateCompress=no
40 cplIsGnu=no
41 logInBackupDir=no
42 compressLogInBackupDir=no
43 logInBackupDirFileName='.storeBackup.log'
44 linkToRecent=

```

Sie übermittelt den `storeBackup`-Skripten Informationen, die benötigt werden, um weiterführende Aktionen mit den Backups durchzuführen, z. B. wie zurückgesichert werden soll: In diesem Beispiel haben die komprimierten Dateien im Backup die Endung `.bz2` (Schlüsselwort `postfix`) und sollen mit dem externen Kommando `bzip2 -d` (Schlüsselwort `uncompress`) entpackt werden.⁶⁴

Weiterhin gibt es noch ein leeres Verzeichnis mit der Bezeichnung `.storeBackupLinks`, das Informationen darüber enthält, ob es mit `lateLinks` erzeugt wurde und noch nicht mit `storeBackupUpdateBackup.pl` vervollständigt wurde. Aber das wird später erklärt.

Der Eintrag für „Kompression“ (`c` oder `u`) kann zusätzlich der Wert `b` annehmen. Das bedeutet, dass die entsprechende Datei ein „blocked file“ ist. Anstelle der Datei wird im Backup ein gleichnamiges Verzeichnis angelegt. Neben den aufgespaltenen Teilen der Datei existiert dort eine Datei namens `.md5BlockCheckSums.bz2`, in der die Informationen über Teilstücke des „blocked files“ stehen, z. B.:

⁶³Die Zeile 8 ist leicht gekürzt, da anderweitig schlecht darstellbar.

⁶⁴Die Information, ob eine Datei komprimiert wurde, ist in `.md5CheckSums.bz2` gespeichert: In der zweiten Spalte steht `c` für „komprimiert“, und `u` für „unkomprimiert“.

```

1 66fa1a8f82c35ca87a08aed9701c5d20 c add_VirtualBox/HardDisks/debian.vdi/0000000001.bz2
2 c522c1db31cc1f90b5d21992fd30e2ab c add_VirtualBox/HardDisks/debian.vdi/0000000002.bz2
3 c522c1db31cc1f90b5d21992fd30e2ab c add_VirtualBox/HardDisks/debian.vdi/0000000003.bz2
4 c522c1db31cc1f90b5d21992fd30e2ab c add_VirtualBox/HardDisks/debian.vdi/0000000004.bz2
5 .....

```

Die erste Spalte enthält die md5 Summen, die zweite *c* (komprimiert / compressed) oder *u* (unkomprimiert / uncompressed) sowie die relative Pfad/Name Kombination eines jeden Blocks in dem Backup.

Schlussfolgerung

Mit Hilfe der vorangegangenen Informationen ist offensichtlich, dass storeBackup ziemlich unempfindlich gegen Manipulationen ist, die Du mit dem Backup anstellen kannst – natürlich nur, wenn Du die Integrität einzelner Backups nicht zerstörst. Da kannst Backups kopieren, Serien („series“) oder sogar einzelne Backups verschieben. Falls Du das tust, musst Du die Änderungen natürlich in den Pfaden und Bezeichnungen („series“) in den Konfigurationsdateien entsprechend nachziehen.

8.3 Late Links Backups

Ein Backup mit der Option `lateLinks` durchzuführen bedeutet, dass die Erstellung des Backups in zwei Teile getrennt wird. Der erste Teil läuft mit `storeBackup.pl`, welches alle Dateien mit neuem Inhalt erkennt und kopiert – der zweite Teil erfolgt mit `storeBackupUpdateBackup.pl`, welches:

- symblock-Links erzeugt und Dateien komprimiert (falls Option `lateCompress` gewählt war),
- Hardlinks setzt, sowie
- Datei- und Verzeichnisrechte setzt.

Der wichtigste Teil – bezogen auf die Struktur von storeBackup – ist das Erzeugen der (noch fehlenden) Hardlinks im Backup. Das bewirkt, dass aus dem noch nicht vollständigen Backup, welches eine Art inkrementelle Sicherung ist,⁶⁵ ein vollständiges Backup wird.

Erstellen wir ein Beispiel:

```

1 $ mkdir /tmp/a
2 $ cd /tmp/a
3 $ mkdir -p s/sub1 b
4 $ cp /bin/ls /bin/pwd s
5 $ cp /bin/ls /bin/pwd s/sub1

```

Wähle ein anderes Verzeichnis Deiner Wahl, falls das Directory `/tmp/a` auf Deinem Rechner schon existiert. Wir haben jetzt ein sourceDir `s` mit ein wenig Inhalt sowie ein backupDir `b` erzeugt.

Jetzt läuft das erste Backup mit Option `--lateLinks` und `--lateCompress`. Sieh auf das Resultat:

```

1 $ storeBackup.pl -s s -b b --lateLinks --lateCompress
2 .....
3 $ find b -print | sort
4 b
5 b/default
6 b/default/2013.08.10_10.14.00
7 b/default/2013.08.10_10.14.00/ls
8 b/default/2013.08.10_10.14.00/.md5CheckSums.bz2
9 b/default/2013.08.10_10.14.00/.md5CheckSums.info
10 b/default/2013.08.10_10.14.00/pwd
11 b/default/2013.08.10_10.14.00/.storeBackupLinks
12 b/default/2013.08.10_10.14.00/.storeBackupLinks/linkFile.bz2

```

Du siehst:

- der gewählte Serienname ist `default`
- die zwei Metadaten Dateien `.md5CheckSums.bz2` und `.md5CheckSums.info` wurden erzeugt

⁶⁵Es ist nicht dasselbe wie eine „richtige“ inkrementelle Sicherung, da hier noch die Informationen über die fehlenden Teile enthalten ist. (Diese werden später über die Hardlinks ergänzt.)

- die Dateien `ls` und `pwd` sind im Backup, allerdings nicht komprimiert; weiterhin fehlen die Dateien `sub1/ls` und `sub1/pwd`
- in `.storeBackupLinks` gibt es die Datei `linkFile.bz2`

Dieses „link file“ beinhaltet die Information, was noch zu tun ist:

```

1 $ bzcat b/default/2013.08.10_10.14.00/.storeBackupLinks/linkFile.bz2
2 # link md5sum
3 #     existingFile
4 #     newLink
5 # compress md5sum
6 #     fileToCompress
7 # dir dirName
8 # symlink file
9 #     target
10 # linkSymlink link
11 #     existingFile
12 #     newLink
13 dir sub1
14 compress c5f89e40c144b6fb8b61f2ef72e4b556
15 pwd
16 compress b5607b4dc7d896c0fab5c4a308239161
17 ls
18 link c5f89e40c144b6fb8b61f2ef72e4b556
19 ./pwd.bz2
20 sub1/pwd.bz2
21 link b5607b4dc7d896c0fab5c4a308239161
22 ./ls.bz2
23 sub1/ls.bz2

```

In den ersten auskommentierten Zeilen wird die Syntax kurz erklärt. Du siehst, dass folgendes ausgeführt werden soll (alle Pfade sind relativ zum aktuellen Backup Verzeichnis):

- erzeuge Directory `sub1`
- komprimiere Datei `pwd` (sie wird dann `pwd.bz2`)
- komprimiere Datei `ls` (sie wird dann `ls.bz2`)
- Hard linke `sub1/pwd.bz2` nach `pwd.bz2`
- Hard linke `sub1/ls.bz2` nach `ls.bz2`

Es gibt keine Abhängigkeiten zu anderen Verzeichnisse, weil dieses das erste Backup war. Nach dem Lauf von `storeBackupUpdateBackup.pl` ergibt sich:

```

1 $ storeBackupUpdateBackup.pl -b b
2 .....
3 $ find b -ls
4 232815 0 drwxrwxr-x 3 hjc hjc 60 Aug 10 10:14 b
5 240109 0 drwx----- 3 hjc hjc 60 Aug 10 12:18 b/default
6 305119 0 drwxr-xr-x 4 hjc hjc 160 Aug 10 12:15 b/default/2013.08.10_11.52.15
7 249568 52 -rwxr-xr-x 2 hjc hjc 49915 Aug 10 10:03 b/default/2013.08.10_11.52.15/ls.bz2
8 249567 16 -rwxr-xr-x 2 hjc hjc 13395 Aug 10 10:03 b/default/2013.08.10_11.52.15/pwd.bz2
9 323964 0 drwxrwxr-x 2 hjc hjc 80 Aug 10 10:03 b/default/2013.08.10_11.52.15/sub1
10 249568 52 -rwxr-xr-x 2 hjc hjc 49915 Aug 10 10:03 b/default/2013.08.10_11.52.15/sub1/ls.bz2
11 249567 16 -rwxr-xr-x 2 hjc hjc 13395 Aug 10 10:03 b/default/2013.08.10_11.52.15/sub1/pwd.bz2
12 306983 4 -rw-rw-r-- 1 hjc hjc 300 Aug 10 11:52 b/default/2013.08.10_11.52.15/.md5CheckSums.bz2
13 305122 4 -rw----- 1 hjc hjc 950 Aug 10 11:52 b/default/2013.08.10_11.52.15/.md5CheckSums.info
14 305120 0 drwx----- 2 hjc hjc 40 Aug 10 12:15 b/default/2013.08.10_11.52.15/.storeBackupLinks
15

```

Du siehst, dass `ls` und `pwd` komprimiert wurden. Die Hardlinks vom Subdirectory `sub1` wurden ebenfalls erstellt. Letztendlich wurden die Berechtigungen korrigiert und `.storeBackupLinks/linkFile.bz2` existiert nicht mehr. Dieses Backup ist nun vervollständigt.

Lassen wir `storeBackup.pl` ein zweites Mal laufen:

```

1 $ storeBackup.pl -s s -b b --lateLinks --lateCompress
2 .....
3 $ find b -print | sort
4 b
5 b/default
6 b/default/2013.08.10_10.14.00
7 b/default/2013.08.10_10.14.00/ls.bz2
8 b/default/2013.08.10_10.14.00/.md5CheckSums.bz2
9 b/default/2013.08.10_10.14.00/.md5CheckSums.info
10 b/default/2013.08.10_10.14.00/pwd.bz2
11 b/default/2013.08.10_10.14.00/.storeBackupLinks
12 b/default/2013.08.10_10.14.00/.storeBackupLinks/linkFrom0
13 b/default/2013.08.10_10.14.00/sub1
14 b/default/2013.08.10_10.14.00/sub1/ls.bz2
15 b/default/2013.08.10_10.14.00/sub1/pwd.bz2
16 b/default/2013.08.10_11.52.15
17 b/default/2013.08.10_11.52.15/.md5CheckSums.bz2
18 b/default/2013.08.10_11.52.15/.md5CheckSums.info
19 b/default/2013.08.10_11.52.15/.storeBackupLinks
20 b/default/2013.08.10_11.52.15/.storeBackupLinks/linkFile.bz2
21 b/default/2013.08.10_11.52.15/.storeBackupLinks/linkTo

```

Du siehst nun eine Änderung im ersten Backup: Es existiert jetzt die Datei `.storeBackupLinks/linkFrom0`. Diese Datei hat folgenden Inhalt:

```

1 $ cat b/default/2013.08.10_10.14.00/.storeBackupLinks/linkFrom0
2 ../2013.08.10_11.52.15

```

Das bedeutet, dass mindestens eine noch zu setzende Referenz mit dem relativen Pfad `2013.08.10_11.52.15` zu diesem Backup existiert. Wenn Referenzen von mehreren anderen Backups existierten, würde es mehrere Dateien dieser Art (`linkFrom1`, `linkFrom2`, ...), die diese Abhängigkeiten beschreiben, geben. Aufgrund dieser Datei oder Dateien erkennt `storeBackupDel.pl`, dass derartige Backups nicht gelöscht werden dürfen, da sonst die Informationen, auf die spätere Backups (über nicht-ausgeführte Links) verweisen, verloren gehen würden.

In dem neuen Backup (`2013.08.10_11.52.15`) listet die Datei `.storeBackupLinks/linkTo` sämtliche anderen Sicherungen auf, zu denen dieses Backup Referenzen hat (in diesem Fall nur eine).

```

1 $ cat b/default/2013.08.10_11.52.15/.storeBackupLinks/linkTo
2 ../2013.08.10_10.14.00

```

Wie Du an der Auflistung der Dateien im zweiten Backup sehen kannst, gibt es in diesem neuen Backup keine Dateien mit Daten aus dem `sourceDir` – es gibt nur Meta Daten von `storeBackup`. Das ist richtig, denn seit dem ersten Backup haben sich keine Dateien im Quellverzeichnis verändert. Sehen wir ins `linkFile`:

```

1 $ bzcata b/default/2013.08.10_11.52.15/.storeBackupLinks/linkFile.bz2
2 # link md5sum
3 #     existingFile
4 #     newLink
5 # compress md5sum
6 #     fileToCompress
7 # dir dirName
8 # symlink file
9 #     target
10 # linkSymlink link
11 #     existingFile
12 #     newLink
13 link c5f89e40c144b6fb8b61f2ef72e4b556
14 ../2013.08.10_10.14.00/sub1/pwd.bz2
15 pwd.bz2
16 link b5607b4dc7d896c0fab5c4a308239161
17 ../2013.08.10_10.14.00/sub1/ls.bz2
18 ls.bz2
19 dir sub1

```



```

20 link c5f89e40c144b6fb8b61f2ef72e4b556
21 ../2013.08.10_10.14.00/sub1/pwd.bz2
22 sub1/pwd.bz2
23 link b5607b4dc7d896c0fab5c4a308239161
24 ../2013.08.10_10.14.00/sub1/ls.bz2
25 sub1/ls.bz2

```

Es ist leicht zu erkennen, dass alle fehlenden Dateien per Hardlink erzeugt sowie das Unterverzeichnis erstellt werden sollen.

Unten sieht man die Änderungen nach einem Lauf von `storeBackupUpdateBackup.pl`:

```

1 $ storeBackupUpdateBackup.pl -b b
2 .....
3 $ find b -ls
4 232815 0 drwxrwxr-x 3 hjc hjc 60 Aug 10 10:14 b
5 240109 0 drwx----- 4 hjc hjc 80 Aug 10 11:52 b/default
6 305119 0 drwxr-xr-x 4 hjc hjc 160 Aug 10 12:15 b/default/2013.08.10_11.52.15
7 249568 52 -rwxr-xr-x 4 hjc hjc 49915 Aug 10 10:03 b/default/2013.08.10_11.52.15/ls.bz2
8 249567 16 -rwxr-xr-x 4 hjc hjc 13395 Aug 10 10:03 b/default/2013.08.10_11.52.15/pwd.bz2
9 323964 0 drwxrwxr-x 2 hjc hjc 80 Aug 10 10:03 b/default/2013.08.10_11.52.15/sub1
10 249568 52 -rwxr-xr-x 4 hjc hjc 49915 Aug 10 10:03 b/default/2013.08.10_11.52.15/sub1/ls.bz2
11 249567 16 -rwxr-xr-x 4 hjc hjc 13395 Aug 10 10:03 b/default/2013.08.10_11.52.15/sub1/pwd.bz2
12 306983 4 -rw-rw-r-- 1 hjc hjc 300 Aug 10 11:52 b/default/2013.08.10_11.52.15/.md5CheckSums.bz2
13 305122 4 -rw----- 1 hjc hjc 950 Aug 10 11:52 b/default/2013.08.10_11.52.15/.md5CheckSums.info
14 305120 0 drwx----- 2 hjc hjc 40 Aug 10 12:15 b/default/2013.08.10_11.52.15/.storeBackupLinks
15 241222 0 drwxr-xr-x 4 hjc hjc 160 Aug 10 10:27 b/default/2013.08.10_10.14.00
16 249568 52 -rwxr-xr-x 4 hjc hjc 49915 Aug 10 10:03 b/default/2013.08.10_10.14.00/ls.bz2
17 249567 16 -rwxr-xr-x 4 hjc hjc 13395 Aug 10 10:03 b/default/2013.08.10_10.14.00/pwd.bz2
18 249566 0 drwxrwxr-x 2 hjc hjc 80 Aug 10 10:03 b/default/2013.08.10_10.14.00/sub1
19 249568 52 -rwxr-xr-x 4 hjc hjc 49915 Aug 10 10:03 b/default/2013.08.10_10.14.00/sub1/ls.bz2
20 249567 16 -rwxr-xr-x 4 hjc hjc 13395 Aug 10 10:03 b/default/2013.08.10_10.14.00/sub1/pwd.bz2
21 238098 4 -rw-rw-r-- 1 hjc hjc 300 Aug 10 10:14 b/default/2013.08.10_10.14.00/.md5CheckSums.bz2
22 241225 4 -rw----- 1 hjc hjc 950 Aug 10 10:14 b/default/2013.08.10_10.14.00/.md5CheckSums.info
23 241223 0 drwx----- 2 hjc hjc 40 Aug 10 12:15 b/default/2013.08.10_10.14.00/.storeBackupLinks

```

Wie Du siehst, ist das Ergebnis ein vollständiges Backup.

Das war bis jetzt alles das normale Verhalten – nichts lief falsch. Ich erzeuge nun ein neues backupDir (`b1`) und lasse es schiefgehen:

```

1 $ storeBackup.pl -s s -b b1
2 .....
3 $ storeBackup.pl -s s -b b1 --lateLinks --lateCompress
4 .....
5 $ find b1 -print | sort
6 b1
7 b1/default
8 b1/default/2013.08.10_13.47.41
9 b1/default/2013.08.10_13.47.41/ls.bz2
10 b1/default/2013.08.10_13.47.41/.md5CheckSums.bz2
11 b1/default/2013.08.10_13.47.41/.md5CheckSums.info
12 b1/default/2013.08.10_13.47.41/pwd.bz2
13 b1/default/2013.08.10_13.47.41/.storeBackupLinks
14 b1/default/2013.08.10_13.47.41/.storeBackupLinks/linkFrom0
15 b1/default/2013.08.10_13.47.41/sub1
16 b1/default/2013.08.10_13.47.41/sub1/ls.bz2
17 b1/default/2013.08.10_13.47.41/sub1/pwd.bz2
18 b1/default/2013.08.10_13.49.21
19 b1/default/2013.08.10_13.49.21/.md5CheckSums.bz2
20 b1/default/2013.08.10_13.49.21/.md5CheckSums.info
21 b1/default/2013.08.10_13.49.21/.storeBackupLinks
22 b1/default/2013.08.10_13.49.21/.storeBackupLinks/linkFile.bz2
23 b1/default/2013.08.10_13.49.21/.storeBackupLinks/linkTo

```

Du erkennst jetzt ein vollständiges Backup (das erste) und ein zweites, nicht-vollständiges Backup. Aus irgendeinem Grund wird das zweite Backup gelöscht. Vielleicht wurde während der Sicherung (nehmen wir an, sie dauerte sehr lange) bemerkt, dass eine Option falsche war und das Backup wurde mit `control-c`

(Steuerung-c) abgebrochen. Anschließend wurde das unfertige Backup einfach gelöscht. In diesem Beispiel lösche ich nun das zweite Backup:

```
1 $ rm -r b1/default/2013.08.10_13.49.21
```

Allerdings existiert die Datei `b1/default/2013.08.10_13.47.41/.storeBackupLinks/linkFrom` noch, so dass die Backup-Serie nicht konsistent ist.

Die Ausführung von `storeBackupUpdateBackup.pl` erzeugt eine Fehlermeldung:

```
1 $ storeBackupUpdateBackup.pl -b b1
2 BEGIN      2013.08.10 13:57:46 475 checking references and backup copying in <b1>
3 VERSION    2013.08.10 13:57:46 475 storeBackupUpdateBackup.pl, 3.4 +
4 INFO       2013.08.10 13:57:46 475 creating lock file </tmp/storeBackup.lock>
5 INFO       2013.08.10 13:57:46 475 scanning directory <b1> for existing backups
6 INFO       2013.08.10 13:57:46 475 scanning directory <b1/default> for existing backups
7 STATISTIC  2013.08.10 13:57:46 475 found 1 backup series, 1 backups, 0 renamed backups
8 ERROR      2013.08.10 13:57:46 475 link <../2013.08.10_13.49.21> to non existing dir in
9 </tmp/a/b1/default/2013.08.10_13.47.41/.storeBackupLinks/linkFrom0>
10 ERROR      2013.08.10 13:57:46 475 found 1 inconsistencies, please repair and check again
```

Die Datei zu löschen, könnte eine schlechte Idee sein, weil in einer realen Umgebung vermutlich noch andere Referenzen zu diesem Backup zeigen würden. Was bedeutet, dass die Anpassung nicht so einfach wäre, die Aktion falsch wäre und die Situation leicht noch schlechter machen würde.

Wir haben mit `storeBackupUpdateBackup.pl` zwei Möglichkeiten:

1. Option `--autorepair` überprüft das Backup und erzeugt die fehlenden Referenzen oder Löscht die, die ins Nichts zeigen.
2. Option `--interactive` fragt immer, was gemacht werden soll, wenn eine Inkonsistenz gefunden wird.

Weil die Option `--autorepair` nicht destruktiv ist, ist sie in 99% der Fälle eine gute Wahl. Aus diesem Grund verwende ich jetzt `--autorepair`:

```
1 $ storeBackupUpdateBackup.pl -b b1 --autorepair
2 BEGIN      2013.08.10 17:06:59 4569 checking references and backup copying in <b1>
3 VERSION    2013.08.10 17:06:59 4569 storeBackupUpdateBackup.pl, 3.4 +
4 INFO       2013.08.10 17:06:59 4569 removing old lock file of process <475>
5 INFO       2013.08.10 17:06:59 4569 creating lock file </tmp/storeBackup.lock>
6 INFO       2013.08.10 17:06:59 4569 scanning directory <b1> for existing backups
7 INFO       2013.08.10 17:06:59 4569 scanning directory <b1/default> for existing backups
8 STATISTIC  2013.08.10 17:06:59 4569 found 1 backup series, 1 backups, 0 renamed backups
9 ERROR      2013.08.10 17:06:59 4569 link <../2013.08.10_13.49.21> to non existing dir in
10 </tmp/a/b1/default/2013.08.10_13.47.41/.storeBackupLinks/linkFrom0>
11 INFO       2013.08.10 17:06:59 4569 autorepair: deleted
12 </tmp/a/b1/default/2013.08.10_13.47.41/.storeBackupLinks/linkFrom0>
13 ERROR      2013.08.10 17:06:59 4569 ----- repeating consistency check -----
14 INFO       2013.08.10 17:06:59 4569 scanning directory <b1> for existing backups
15 INFO       2013.08.10 17:06:59 4569 scanning directory <b1/default> for existing backups
16 STATISTIC  2013.08.10 17:06:59 4569 found 1 backup series, 1 backups, 0 renamed backups
17 INFO       2013.08.10 17:06:59 4569 consistency check finished successfully
18 INFO       2013.08.10 17:06:59 4569 found no references to backups from lateLinks that need
19 storeBackupUpdateBackup run
20 INFO       2013.08.10 17:06:59 4569 everything is updated, nothing to do
21 STATISTIC  2013.08.10 17:06:59 4569                                     duration = 1s
22 STATISTIC  2013.08.10 17:06:59 4569 [sec] |      user|      system
23 STATISTIC  2013.08.10 17:06:59 4569 -----+-----+-----
24 STATISTIC  2013.08.10 17:06:59 4569 process|      0.08|      0.00
25 STATISTIC  2013.08.10 17:06:59 4569 childs |      0.05|      0.00
26 STATISTIC  2013.08.10 17:06:59 4569 -----+-----+-----
27 STATISTIC  2013.08.10 17:06:59 4569 sum    |      0.13|      0.00 => 0.13
28 INFO       2013.08.10 17:06:59 4569 syncing ...
29 INFO       2013.08.10 17:06:59 4569 checking references and copying in <b1>
30 $ find b1 -print | sort
31 b1
32 b1/default
33 b1/default/2013.08.10_13.47.41
34 b1/default/2013.08.10_13.47.41/ls.bz2
35 b1/default/2013.08.10_13.47.41/.md5CheckSums.bz2
36 b1/default/2013.08.10_13.47.41/.md5CheckSums.info
37 b1/default/2013.08.10_13.47.41/pwd.bz2
38 b1/default/2013.08.10_13.47.41/.storeBackupLinks
```

```

39 b1/default/2013.08.10_13.47.41/sub1
40 b1/default/2013.08.10_13.47.41/sub1/ls.bz2
41 b1/default/2013.08.10_13.47.41/sub1/pwd.bz2

```

Die Ausgabe des `find` Kommandos oben zeigt, dass alles in Ordnung ist – es gibt eine vollständige Sicherung.

Nun erzeugen wir nochmals ein zweites Backup mit `lateLinks`:

```

1 $ storeBackup.pl -s s -b b1 --lateLinks --lateCompress
2 -----
3 $ find b1 -print | sort
4 b1
5 b1/default
6 b1/default/2013.08.10_13.47.41
7 b1/default/2013.08.10_13.47.41/ls.bz2
8 b1/default/2013.08.10_13.47.41/.md5CheckSums.bz2
9 b1/default/2013.08.10_13.47.41/.md5CheckSums.info
10 b1/default/2013.08.10_13.47.41/pwd.bz2
11 b1/default/2013.08.10_13.47.41/.storeBackupLinks
12 b1/default/2013.08.10_13.47.41/.storeBackupLinks/linkFrom0
13 b1/default/2013.08.10_13.47.41/sub1
14 b1/default/2013.08.10_13.47.41/sub1/ls.bz2
15 b1/default/2013.08.10_13.47.41/sub1/pwd.bz2
16 b1/default/2013.08.10_17.17.53
17 b1/default/2013.08.10_17.17.53/.md5CheckSums.bz2
18 b1/default/2013.08.10_17.17.53/.md5CheckSums.info
19 b1/default/2013.08.10_17.17.53/.storeBackupLinks
20 b1/default/2013.08.10_17.17.53/.storeBackupLinks/linkFile.bz2
21 b1/default/2013.08.10_17.17.53/.storeBackupLinks/linkTo

```

Diese Mal lösche ich das *erste* Backup:

```

1 $ rm -r b1/default/2013.08.10_13.47.41

```

Hieraus ergibt sich, dass ich nur ein nicht-vollständiges Backup habe (welches mit `lateLinks` erstellt wurde), dass ins Nichts verweist. Ich versuche wiederum `--storeBackupUpdateBackup.pl` mit Option `--autorepair` zu verwenden:

```

1 $ storeBackupUpdateBackup.pl -b b1 --autorepair
2 BEGIN      2013.08.10 17:21:08 5019 checking references and backup copying in <b1>
3 VERSION    2013.08.10 17:21:08 5019 storeBackupUpdateBackup.pl, 3.4 +
4 INFO       2013.08.10 17:21:08 5019 creating lock file </tmp/storeBackup.lock>
5 INFO       2013.08.10 17:21:08 5019 scanning directory <b1> for existing backups
6 INFO       2013.08.10 17:21:08 5019 scanning directory <b1/default> for existing backups
7 STATISTIC  2013.08.10 17:21:08 5019 found 1 backup series, 1 backups, 0 renamed backups
8 ERROR      2013.08.10 17:21:08 5019 FATAL ERROR: link <../2013.08.10_13.47.41> to non existing dir in
9 </tmp/a/b1/default/2013.08.10_17.17.53/.storeBackupLinks/linkTo>
10 ERROR      2013.08.10 17:21:08 5019 found 1 inconsistencies, please repair and check again

```

Jetzt erhalte ich einen *FATAL ERROR*. Es gibt für `storeBackupUpdateBackup.pl` keine Chance, diesen Fehler zu reparieren. Die einzige Möglichkeit wäre, das fehlende (erste) Backup von woanders (z. B. von einer Kopie oder Replikation) an die richtige Stelle zu kopieren⁶⁶ und anschließend `storeBackupUpdateBackup.pl` mit `--autorepair` laufen zu lassen. Wenn es keine andere Sicherung des fehlenden Backups gibt, sind alle Backups (und alle darauffolgenden Backups), die von diesem abhängen, nicht mehr komplettierbar. Du solltest sie löschen oder diese unvollständigen Backups an eine Stelle außerhalb von `backupDir` verschieben, damit sie den Ablauf (Workflow) der `storeBackup`-Programme nicht behindern.

8.4 Replikation

Du solltest zumindest das Kapitel 7.8.6, „Wie `storeBackups` Replikation funktioniert“ gelesen haben – es gibt Dir Erklärungen zu den grundlegenden Konzepten und Konfigurationen.

⁶⁶Es ist eine gute Idee, hier `linkToDirs.pl` für das Kopieren zu verwenden, weil dann der größte Teil der Dateien mit bereits existierenden per Hardlink verbunden werden kann. `linkToDirs.pl` weiß nichts über `storeBackups` Logik und seine Meta Daten – daher kann es für jeglichen Daten verwendet werden; auch mit nicht-vollständigen Sicherungen oder irgendetwas anderem.

Du musst wissen, dass die Replikation als Erweiterung von lateLinks realisiert ist, um zu verstehen, wie sie im Detail funktioniert,

Die Interna der Replikationsfunktionalität werden an Beispielen erläutert. Zuerst erstellen wir ein Backup mit einer Replikation:

```
1 $ mkdir /tmp/a
2 $ cd /tmp/a
3 $ mkdir -p source backup/default repl delta
4 $ cp /bin/ls source
5 $ cp /bin/ls source/ls1
6 $ storeBackupReplicationWizard.pl -m backup -c repl -d delta -S default
7 WARNING 2013.08.17 10:12:44 25548 cannot find any existing backup at master backup directory
   ↪ </tmp/a/backup>
8 INFO    2013.08.17 10:12:44 25548 1 series chosen:
9 INFO    2013.08.17 10:12:44 25548      <default>
10 INFO   2013.08.17 10:12:44 25548 wrote master backup configuration file
   ↪ </tmp/a/backup/storeBackupBaseTree.conf>
11 INFO   2013.08.17 10:12:44 25548 wrote backup copy configuration file
   ↪ </tmp/a/repl/storeBackupBaseTree.conf>
12 INFO   2013.08.17 10:12:44 25548 wrote delta cache configuration file </tmp/a/delta/deltaCache.conf>
```

Schauen wir nach, was storeBackupReplicationWizard.pl generiert hat:

```
1 $ cat backup/storeBackupBaseTree.conf | egrep -v '^\s*$$|^#'
2 backupTreeName='Master Backup'
3 backupType=master
4 seriesToDistribute='default'
5 deltaCache=/tmp/a/delta
6
7 $ cat delta/deltaCache.conf | egrep -v '^\s*$$|^#'
8 backupCopy0='Backup Copy' 'default'
9 ;backupCopy1=
10 ;backupCopy2=
11 ;backupCopy3=
12 ;backupCopy4=
13 ;backupCopy5=
14 ;backupCopy6=
15 ;backupCopy7=
16 ;backupCopy8=
17 ;backupCopy9=
18
19 $ cat repl/storeBackupBaseTree.conf | egrep -v '^\s*$$|^#'
20 backupTreeName='Backup Copy'
21 backupType=copy
22 seriesToDistribute='default'
23 deltaCache=/tmp/a/delta
```

Folgende Dateien sind in den Verzeichnissen:

```
1 $ find . -print | sort
2 .
3 ./backup
4 ./backup/default
5 ./backup/storeBackupBaseTree.conf
6 ./delta
7 ./delta/deltaCache.conf
8 ./repl
9 ./repl/storeBackupBaseTree.conf
10 ./source
11 ./source/ls
12 ./source/ls1
```

Jetzt erstellen wir das erste Backup mit lateLinks:

```
1 $ storeBackup.pl -s source -b backup --lateLinks
2 .....
3 $ find . -print | sort
```

```

4 .
5 ./backup
6 ./backup/default
7 ./backup/default/2013.08.17_10.13.41
8 ./backup/default/2013.08.17_10.13.41/ls1.bz2
9 ./backup/default/2013.08.17_10.13.41/.md5CheckSums.bz2
10 ./backup/default/2013.08.17_10.13.41/.md5CheckSums.info
11 ./backup/default/2013.08.17_10.13.41/.storeBackupLinks
12 ./backup/default/2013.08.17_10.13.41/.storeBackupLinks/linkFile.bz2
13 ./backup/storeBackupBaseTree.conf
14 ./delta
15 ./delta/deltaCache.conf
16 ./repl
17 ./repl/storeBackupBaseTree.conf
18 ./source
19 ./source/ls
20 ./source/ls1

```

Bis jetzt nichts Neues. Eine komprimierte Version von `ls1` in `source` befindet sich in `backup`. Die Datei `ls` fehlt dort aufgrund der Deduplikation. Weiterhin befinden sich die Metadaten sowie `linkFile` (siehe voriges Kapitel) in `backup`.

Wir beenden das Backup mit `storeBackupUpdateBackup.pl`:

```

1 $ storeBackupUpdateBackup.pl -b backup
2 BEGIN    2013.08.17 10:14:59 25914 checking references and backup copying in <backup>
3 VERSION  2013.08.17 10:14:59 25914 storeBackupUpdateBackup.pl, 3.4 +
4 INFO     2013.08.17 10:14:59 25914 creating lock file </tmp/storeBackup.lock>
5 INFO     2013.08.17 10:14:59 25914 reading <backup/storeBackupBaseTree.conf>
6 INFO     2013.08.17 10:14:59 25914 master backup: checking <Master Backup/default>
7 INFO     2013.08.17 10:14:59 25914 copying <2013.08.17_10.13.41> to </tmp/a/delta/default>
8 INFO     2013.08.17 10:14:59 25914 scanning directory <backup> for existing backups
9 INFO     2013.08.17 10:14:59 25914 scanning directory <backup/default> for existing backups
10 STATISTIC 2013.08.17 10:14:59 25914 found 1 backup series, 1 backups, 0 renamed backups
11 INFO     2013.08.17 10:14:59 25914 consistency check finished successfully
12 INFO     2013.08.17 10:14:59 25914 found no references to backups from lateLinks that need
13 storeBackupUpdateBackup run
14 INFO     2013.08.17 10:14:59 25914 (1/0) updating </tmp/a/backup/default/2013.08.17_10.13.41>
15 INFO     2013.08.17 10:14:59 25914 phase 1: mkdir, symlink and compressing files
16 STATISTIC 2013.08.17 10:14:59 25914 created 0 directories
17 STATISTIC 2013.08.17 10:14:59 25914 created 0 symbolic links
18 STATISTIC 2013.08.17 10:14:59 25914 compressed 0 files
19 STATISTIC 2013.08.17 10:14:59 25914 used 0.0 instead of 0.0 (0 <- 0 ; 0.0%)
20 INFO     2013.08.17 10:14:59 25914 phase 2: setting hard links
21 STATISTIC 2013.08.17 10:14:59 25914 linked 1 files
22 INFO     2013.08.17 10:14:59 25914 phase 3: setting file permissions
23 STATISTIC 2013.08.17 10:14:59 25914 set permissions for 2 files
24 INFO     2013.08.17 10:14:59 25914 phase 4: setting directory permissions
25 STATISTIC 2013.08.17 10:14:59 25914 set permissions for 0 directories
26 INFO     2013.08.17 10:14:59 25914 reading <backup/storeBackupBaseTree.conf>
27 INFO     2013.08.17 10:14:59 25914 scanning directory <backup> for existing backups
28 INFO     2013.08.17 10:14:59 25914 scanning directory <backup/default> for existing backups
29 STATISTIC 2013.08.17 10:14:59 25914 found 1 backup series, 1 backups, 0 renamed backups
30 INFO     2013.08.17 10:14:59 25914 consistency check finished successfully
31 INFO     2013.08.17 10:14:59 25914 found no references to backups from lateLinks that need
32 storeBackupUpdateBackup run
33 INFO     2013.08.17 10:14:59 25914 deleting in deltaCache </tmp/a/delta> processedBackups
34 INFO     2013.08.17 10:14:59 25914 age for deletion is > 99d (delete backups older than
35 Fri 2013.05.10 10:14:59)
36 STATISTIC 2013.08.17 10:14:59 25914 duration = 1s
37 STATISTIC 2013.08.17 10:14:59 25914 [sec] | user| system
38 STATISTIC 2013.08.17 10:14:59 25914 -----+-----+-----
39 STATISTIC 2013.08.17 10:14:59 25914 process| 0.07| 0.03
40 STATISTIC 2013.08.17 10:14:59 25914 childs | 0.05| 0.00
41 STATISTIC 2013.08.17 10:14:59 25914 -----+-----+-----
42 STATISTIC 2013.08.17 10:14:59 25914 sum | 0.12| 0.03 => 0.15
43 INFO     2013.08.17 10:14:59 25914 syncing ...
44 END      2013.08.17 10:14:59 25914 checking references and copying in <backup>
45
46 $ find . -print | sort
47 .

```

```

48 ./backup
49 ./backup/default
50 ./backup/default/2013.08.17_10.13.41
51 ./backup/default/2013.08.17_10.13.41/ls1.bz2
52 ./backup/default/2013.08.17_10.13.41/ls.bz2
53 ./backup/default/2013.08.17_10.13.41/.md5CheckSums.bz2
54 ./backup/default/2013.08.17_10.13.41/.md5CheckSums.info
55 ./backup/default/2013.08.17_10.13.41/.storeBackupLinks
56 ./backup/storeBackupBaseTree.conf
57 ./delta
58 ./delta/default
59 ./delta/default/2013.08.17_10.13.41
60 ./delta/default/2013.08.17_10.13.41/ls1.bz2
61 ./delta/default/2013.08.17_10.13.41/.md5CheckSums.bz2
62 ./delta/default/2013.08.17_10.13.41/.md5CheckSums.info
63 ./delta/default/2013.08.17_10.13.41/.storeBackupLinks
64 ./delta/default/2013.08.17_10.13.41/.storeBackupLinks/linkFile.bz2
65 ./delta/deltaCache.conf
66 ./delta/processedBackups
67 ./repl
68 ./repl/storeBackupBaseTree.conf
69 ./source
70 ./source/ls
71 ./source/ls1

```

Nun hat sich vieles verändert:

- Das Master-Backup in backup ist vollständig. Im backup-Verzeichnis gibt es Hardlinks auf die Dateien ls und ls1 – und linkFile ist verschwunden.
- Am Anfang der Ausgabe von storeBackupUpdateBackup.pl sieht man die folgenden Zeilen:

```

1  reading <backup/storeBackupBaseTree.conf>
2  master backup: checking <Master Backup/default>
3  copying <2013.08.17_10.13.41> to </tmp/a/delta/default>

```

Das Programm erkannte, dass das lateLinks Backup noch nicht in den Delta Cache kopiert wurde – daher wurde es vor der Vervollständigung in diesen kopiert. Beachte die Ausgabe von find; sie zeigt die Kopie im Verzeichnis delta. Der Delta-Cache sammelt alle neuen (zu replizierenden) Backups vom Master-Backup (backup), bis sie zu ihrem Bestimmungsort (repl) gelangen, und speichert sie dann noch eine Zeit lang.

Du siehst auch, dass das Verzeichnis processedBackups in delta erzeugt wurde.

Jetzt läuft storeBackupUpdateBackup.pl ein zweites Mal, aber auf dem Verzeichnis repl:

```

1  $ storeBackupUpdateBackup.pl -b repl
2  BEGIN      2013.08.17 10:33:03 28608 checking references and backup copying in <repl>
3  VERSION    2013.08.17 10:33:03 28608 storeBackupUpdateBackup.pl, 3.4 +
4  INFO       2013.08.17 10:33:03 28608 creating lock file </tmp/storeBackup.lock>
5  INFO       2013.08.17 10:33:03 28608 reading <repl/storeBackupBaseTree.conf>
6  INFO       2013.08.17 10:33:03 28608 reading </tmp/a/delta/deltaCache.conf>
7  INFO       2013.08.17 10:33:03 28608 copying </tmp/a/delta/default/2013.08.17_10.13.41> to <repl/default>
8  INFO       2013.08.17 10:33:03 28608 scanning directory <repl> for existing backups
9  INFO       2013.08.17 10:33:03 28608 scanning directory <repl/default> for existing backups
10 STATISTIC  2013.08.17 10:33:03 28608 found 1 backup series, 1 backups, 0 renamed backups
11 INFO       2013.08.17 10:33:03 28608 consistency check finished successfully
12 INFO       2013.08.17 10:33:03 28608 found no references to backups from lateLinks that need
13 storeBackupUpdateBackup run
14 INFO       2013.08.17 10:33:03 28608 (1/0) updating </tmp/a/repl/default/2013.08.17_10.13.41>
15 INFO       2013.08.17 10:33:03 28608 phase 1: mkdir, symlink and compressing files
16 STATISTIC  2013.08.17 10:33:03 28608 created 0 directories
17 STATISTIC  2013.08.17 10:33:03 28608 created 0 symbolic links
18 STATISTIC  2013.08.17 10:33:03 28608 compressed 0 files
19 STATISTIC  2013.08.17 10:33:03 28608 used 0.0 instead of 0.0 (0 <- 0 ; 0.0%)
20 INFO       2013.08.17 10:33:03 28608 phase 2: setting hard links
21 STATISTIC  2013.08.17 10:33:03 28608 linked 1 files
22 INFO       2013.08.17 10:33:03 28608 phase 3: setting file permissions
23 STATISTIC  2013.08.17 10:33:03 28608 set permissions for 2 files
24 INFO       2013.08.17 10:33:03 28608 phase 4: setting directory permissions
25 STATISTIC  2013.08.17 10:33:03 28608 set permissions for 0 directories

```

```

26 INFO      2013.08.17 10:33:03 28608 reading <repl/storeBackupBaseTree.conf>
27 INFO      2013.08.17 10:33:03 28608 marked <default/2013.08.17_10.13.41> as linked in </tmp/a/delta>
28 INFO      2013.08.17 10:33:03 28608 scanning directory <repl> for existing backups
29 INFO      2013.08.17 10:33:03 28608 scanning directory <repl/default> for existing backups
30 STATISTIC 2013.08.17 10:33:03 28608 found 1 backup series, 1 backups, 0 renamed backups
31 INFO      2013.08.17 10:33:03 28608 consistency check finished successfully
32 INFO      2013.08.17 10:33:03 28608 found no references to backups from lateLinks that need
33 storeBackupUpdateBackup run
34 INFO      2013.08.17 10:33:03 28608 backup </tmp/a/delta/default/2013.08.17_10.13.41> copied to <Backup
    ↳ Copy>
35 INFO      2013.08.17 10:33:03 28608 moving backup to </tmp/a/delta/processedBackups/default>
36 INFO      2013.08.17 10:33:03 28608 deleting in deltaCache </tmp/a/delta> processedBackups
37 INFO      2013.08.17 10:33:03 28608 age for deletion is > 99d (delete backups older than
38 Fri 2013.05.10 10:33:03)
39 INFO      2013.08.17 10:33:03 28608 checking series <default>
40 INFO      2013.08.17 10:33:03 28608 default -> 2013.08.17_10.13.41 - not old enough to delete
41 STATISTIC 2013.08.17 10:33:03 28608 duration = 1s
42 STATISTIC 2013.08.17 10:33:03 28608 [sec] |      user|      system
43 STATISTIC 2013.08.17 10:33:03 28608 -----+-----+-----
44 STATISTIC 2013.08.17 10:33:03 28608 process|      0.09|      0.01
45 STATISTIC 2013.08.17 10:33:03 28608 childs |      0.05|      0.00
46 STATISTIC 2013.08.17 10:33:03 28608 -----+-----+-----
47 STATISTIC 2013.08.17 10:33:03 28608 sum    |      0.14|      0.01 => 0.15
48 INFO      2013.08.17 10:33:03 28608 syncing ...
49 END      2013.08.17 10:33:03 28608 checking references and copying in <repl>
50
51 $ find . -print | sort
52 .
53 ./backup
54 ./backup/default
55 ./backup/default/2013.08.17_10.13.41
56 ./backup/default/2013.08.17_10.13.41/ls1.bz2
57 ./backup/default/2013.08.17_10.13.41/ls.bz2
58 ./backup/default/2013.08.17_10.13.41/.md5CheckSums.bz2
59 ./backup/default/2013.08.17_10.13.41/.md5CheckSums.info
60 ./backup/default/2013.08.17_10.13.41/.storeBackupLinks
61 ./backup/storeBackupBaseTree.conf
62 ./delta
63 ./delta/default
64 ./delta/default/2013.08.17_10.13.41.copied
65 ./delta/default/2013.08.17_10.13.41.linked
66 ./delta/deltaCache.conf
67 ./delta/processedBackups
68 ./delta/processedBackups/default
69 ./delta/processedBackups/default/2013.08.17_10.13.41
70 ./delta/processedBackups/default/2013.08.17_10.13.41/ls1.bz2
71 ./delta/processedBackups/default/2013.08.17_10.13.41/.md5CheckSums.bz2
72 ./delta/processedBackups/default/2013.08.17_10.13.41/.md5CheckSums.info
73 ./delta/processedBackups/default/2013.08.17_10.13.41/.storeBackupLinks
74 ./delta/processedBackups/default/2013.08.17_10.13.41/.storeBackupLinks/linkFile.bz2
75 ./repl
76 ./repl/default
77 ./repl/default/2013.08.17_10.13.41
78 ./repl/default/2013.08.17_10.13.41/ls1.bz2
79 ./repl/default/2013.08.17_10.13.41/ls.bz2
80 ./repl/default/2013.08.17_10.13.41/.md5CheckSums.bz2
81 ./repl/default/2013.08.17_10.13.41/.md5CheckSums.info
82 ./repl/default/2013.08.17_10.13.41/.storeBackupLinks
83 ./repl/storeBackupBaseTree.conf
84 ./source
85 ./source/ls
86 ./source/ls1

```

Ein Blick auf die Dateiliste und den Log von storeBackupUpdateBackup.pl zeigt:

- Das Master-Backup (Verzeichnis backup) ist unverändert.
- Die Deltas wurden in das Replikationsverzeichnis (repl) kopiert, welches das Ziel der Kopie der Daten ist. Zusätzlich wurde diese Kopie verwendet, um das Backup in repl zu vervollständigen. Ein diff zwischen backup und repl zeigt,

```
1 $ diff -r backup repl
```

```

2 diff -r backup/storeBackupBaseTree.conf repl/storeBackupBaseTree.conf
3 21c21
4 < backupTreeName='Master Backup'
5 ---
6 > backupTreeName='Backup Copy'
7 26c26
8 < backupType=master
9 ---
10 > backupType=copy

```

dass nur die Konfigurationsdateien für die Replikation unterschiedlich sind. Die Backups in diesen Verzeichnissen sind völlig identisch. Daher können beispielsweise die Programme `storeBackupDel.pl` oder `storeBackupCheckBackup.pl` auf beiden Verzeichnissen gestartet werden.

- Die Kopie der Delta-Dateien im Delta Cache wurde in das Unterverzeichnis `processedBackups` verschoben. Der *Status* der Replikation wird in folgenden Dateien gespeichert:

```

1 $ cat ./delta/default/2013.08.17_10.13.41.copied
2 Backup Copy
3
4 $ cat ./delta/default/2013.08.17_10.13.41.linked
5 Backup Copy

```

Das bedeutet, dass das Backup der Serie `default` vom 2013.08.17_10.13.41 zur Backup-Kopie `Backup Copy` kopiert und verlinkt wurde. Wenn Du zu mehr als einem anderen Verzeichnis (z. B. auf einer USB-Platte) replizierst, dann findest Du auch deren Namen (so wie `Backup Copy` in diesem Fall) in diesen Dateien (**.copied* und **.linked*) nachdem sie kopiert und verlinkt sind. Auf diese Art kann `storeBackup` entscheiden, ob ein Backup vollständig in alle Replikationsverzeichnissen kopiert und verlinkt wurde. Nachdem dieses sichergestellt ist, werden die betroffenen Sicherungen in das Unterverzeichnis `processedBackups` im Delta Cache verschoben. (Wir sehen später, wie diese Daten verwendet werden können, wenn im Replikationsprozess etwas schief geht.) Diese verarbeiteten Backups werden für eine konfigurierbare Zeit gespeichert. Der Default-Zeitraum von 99 Tagen ist *sehr* konservativ und sollte kürzer gewählt werden.

Mit Hilfe des oben aufgeführten Beispiels ist es einfach zu erkennen, wie das Replikationskonzept umgesetzt ist:⁶⁷

1. Starte `storeBackup.pl -b backup` und erzeuge das lateLinks basierende Delta-Backup in Verzeichnis `backup`.
 - (a) Kopiere die Deltas vom Backup zum Delta-Cache (Directory `delta`).
 - (b) Vervollständige das Backup (erzeuge die Hardlinks usw.) im Verzeichnis `backup`.
2. Starte `storeBackupUpdateBackup.pl -b repl` und
 - (a) kopiere die Deltas aus dem Delta-Cache (Directory `delta`) in das Zielverzeichnis der Replikation (`repl`).
 - (b) schreibe den Namen der Replikation (`Backup Copy`) in Datei `timestamp.copied`
 - (c) vervollständige das Ziel der Replikation im Verzeichnis `repl` (erzeuge Hardlinks usw.). (Im Prinzip dasselbe wie im oben beschriebenen Schritt 1b.)
 - (d) verschiebe das soeben kopierte Backup im Delta-Cache in das Unterverzeichnis `processedBackups` und schreibe den Namen der Replikation (`Backup Copy`) in die Datei `timestamp.linked`

Mit Kenntnis der grundlegenden Prinzipien der Replikation ist es einfach, darauf zu reagieren, wenn etwas schief geht. Es geht immer ums Verschieben, Kopieren oder darum, Zeilen an eine Datei anzuhängen. Simulieren wir nun eine abgebrochene Replikation. (Es folgt ein neues Beispiel von Anfang an.)

Als erstes erzeugen wir ein neues Backup und replizieren es:

⁶⁷Um die Erklärungen einfach und verständlich zu halten, werden die Verzeichnisnamen von oben weiter verwendet.


```

1 $ mkdir /tmp/x
2 $ cd /tmp/x
3 $ mkdir -p source backup/default repl delta
4 $ cp /bin/ls source
5 $ storeBackupReplicationWizard.pl -m backup -c repl -d delta -S default
6 $ storeBackup.pl -s source -b backup --lateLinks
7 $ storeBackupUpdateBackup.pl -b backup
8 $ storeBackupUpdateBackup.pl -b repl

```

Wir fügen noch eine Datei zu source hinzu und starten das Backup sowie die Replikation:

```

1 $ cp /bin/pwd source
2 $ storeBackup.pl -s source -b backup --lateLinks
3 $ storeBackupUpdateBackup.pl -b backup
4 $ storeBackupUpdateBackup.pl -b repl

```

Die folgenden Backups wurden erzeugt:

```

1 $ ls -l backup/default/
2 2013.08.18_09.45.16
3 2013.08.18_09.49.21

```

Nun *stellen wir uns vor*, das zweite Backup sei korrupt – vielleicht wegen einer Unterbrechung des Laufs von storeBackupUpdateBackup.pl, aufgetretenen Schreibfehlern oder warum auch immer. Da wir die Details der Fehler und welche Dateien betroffen wären nicht wüssten, löschen wir das soeben replizierte Backup:

```

1 $ ls -l repl/default/
2 2013.08.18_09.45.16
3 2013.08.18_09.49.21
4 $ rm -r repl/default/2013.08.18_09.49.21

```

Wiederholtes Starten von storeBackupUpdateBackup.pl auf der Replikation (*ein Backup fehlt!*) resultiert in:

```

1 $ storeBackupUpdateBackup.pl -b repl
2 BEGIN      2013.08.18 09:57:46 6852 checking references and backup copying in <repl>
3 VERSION    2013.08.18 09:57:46 6852 storeBackupUpdateBackup.pl, 3.4 +
4 INFO       2013.08.18 09:57:46 6852 creating lock file </tmp/storeBackup.lock>
5 INFO       2013.08.18 09:57:46 6852 reading <repl/storeBackupBaseTree.conf>
6 INFO       2013.08.18 09:57:46 6852 reading </tmp/x/delta/deltaCache.conf>
7 INFO       2013.08.18 09:57:46 6852 scanning directory <repl> for existing backups
8 INFO       2013.08.18 09:57:46 6852 scanning directory <repl/default> for existing backups
9 STATISTIC  2013.08.18 09:57:46 6852 found 1 backup series, 1 backups, 0 renamed backups
10 INFO       2013.08.18 09:57:46 6852 consistency check finished successfully
11 INFO       2013.08.18 09:57:46 6852 found no references to backups from lateLinks that need
12 storeBackupUpdateBackup run
13 INFO       2013.08.18 09:57:46 6852 everything is updated, nothing to do
14 INFO       2013.08.18 09:57:46 6852 deleting in deltaCache </tmp/x/delta> processedBackups
15 INFO       2013.08.18 09:57:46 6852      age for deletion is > 99d (delete backups older
16 than Sat 2013.05.11 09:57:46)
17 INFO       2013.08.18 09:57:46 6852 checking series <default>
18 INFO       2013.08.18 09:57:46 6852      default -> 2013.08.18_09.45.16 - not old enough to delete
19 INFO       2013.08.18 09:57:46 6852      default -> 2013.08.18_09.49.21 - not old enough to delete
20 STATISTIC  2013.08.18 09:57:46 6852                                     duration = 1s
21 STATISTIC  2013.08.18 09:57:46 6852 [sec] |      user|      system
22 STATISTIC  2013.08.18 09:57:46 6852 -----+-----+-----
23 STATISTIC  2013.08.18 09:57:46 6852 process|      0.08|      0.01
24 STATISTIC  2013.08.18 09:57:46 6852 childs |      0.05|      0.00
25 STATISTIC  2013.08.18 09:57:46 6852 -----+-----+-----
26 STATISTIC  2013.08.18 09:57:46 6852 sum    |      0.13|      0.01 => 0.14
27 INFO       2013.08.18 09:57:46 6852 syncing ...
28 END        2013.08.18 09:57:46 6852 checking references and copying in <repl>
29
30 $ find repl -print | sort
31 repl
32 repl/default
33 repl/default/2013.08.18_09.45.16
34 repl/default/2013.08.18_09.45.16/ls.bz2
35 repl/default/2013.08.18_09.45.16/.md5CheckSums.bz2
36 repl/default/2013.08.18_09.45.16/.md5CheckSums.info
37 repl/default/2013.08.18_09.45.16/.storeBackupLinks
38 repl/storeBackupBaseTree.conf

```

– nichts. `storeBackupUpdateBackup.pl` vermutet, dass alles passt, weil nichts mehr zu replizieren ist. (Die Kontrolldateien zeigen an, dass alles perfekt funktionierte.)

Wenn dieser Fehler nicht behoben wird, wird die Replikation beim nächsten Backup in Fehler laufen. Um das zu verdeutlichen, provozieren wir diesen Fehler:

```
1 $ storeBackup.pl -s source -b backup --lateLinks
2 $ storeBackupUpdateBackup.pl -b backup
3 $ storeBackupUpdateBackup.pl -b repl
4 $ storeBackupUpdateBackup.pl -b repl
5 BEGIN      2013.08.18 10:06:06 7050 checking references and backup copying in <repl>
6 VERSION    2013.08.18 10:06:06 7050 storeBackupUpdateBackup.pl, 3.4 +
7 INFO       2013.08.18 10:06:06 7050 creating lock file </tmp/storeBackup.lock>
8 INFO       2013.08.18 10:06:06 7050 reading <repl/storeBackupBaseTree.conf>
9 INFO       2013.08.18 10:06:06 7050 reading </tmp/x/delta/deltaCache.conf>
10 INFO      2013.08.18 10:06:06 7050 copying </tmp/x/delta/default/2013.08.18_10.05.54> to <repl/default>
11 INFO      2013.08.18 10:06:07 7050 scanning directory <repl> for existing backups
12 INFO      2013.08.18 10:06:07 7050 scanning directory <repl/default> for existing backups
13 STATISTIC 2013.08.18 10:06:07 7050 found 1 backup series, 2 backups, 0 renamed backups
14 ERROR      2013.08.18 10:06:07 7050 FATAL ERROR: link <../2013.08.18_09.49.21> to non existing dir in
15 </tmp/x/repl/default/2013.08.18_10.05.54/.storeBackupLinks/linkTo>
16 ERROR      2013.08.18 10:06:07 7050 found 1 inconsistencies, please repair and check again
```

Wie die *FATAL ERROR* Meldung zeigt, wurde ein fehlendes Backup festgestellt – es gibt daher keine Chance, die Hardlinks zu erstellen.

Ein Blick auf die Dateien im Delta Cache (`delta`) und das Zielverzeichnis der Replikation (`repl`) zeigt:

```
1 $ find delta repl -print | sort
2 delta
3 delta/default
4 delta/default/2013.08.18_09.45.16.copied
5 delta/default/2013.08.18_09.45.16.linked
6 delta/default/2013.08.18_09.49.21.copied
7 delta/default/2013.08.18_09.49.21.linked
8 delta/default/2013.08.18_10.05.54
9 delta/default/2013.08.18_10.05.54.copied
10 delta/default/2013.08.18_10.05.54/.md5CheckSums.bz2
11 delta/default/2013.08.18_10.05.54/.md5CheckSums.info
12 delta/default/2013.08.18_10.05.54/.storeBackupLinks
13 delta/default/2013.08.18_10.05.54/.storeBackupLinks/linkFile.bz2
14 delta/default/2013.08.18_10.05.54/.storeBackupLinks/linkTo
15 delta/deltaCache.conf
16 delta/processedBackups
17 delta/processedBackups/default
18 delta/processedBackups/default/2013.08.18_09.45.16
19 delta/processedBackups/default/2013.08.18_09.45.16/ls.bz2
20 delta/processedBackups/default/2013.08.18_09.45.16/.md5CheckSums.bz2
21 delta/processedBackups/default/2013.08.18_09.45.16/.md5CheckSums.info
22 delta/processedBackups/default/2013.08.18_09.45.16/.storeBackupLinks
23 delta/processedBackups/default/2013.08.18_09.45.16/.storeBackupLinks/linkFile.bz2
24 delta/processedBackups/default/2013.08.18_09.49.21
25 delta/processedBackups/default/2013.08.18_09.49.21/.md5CheckSums.bz2
26 delta/processedBackups/default/2013.08.18_09.49.21/.md5CheckSums.info
27 delta/processedBackups/default/2013.08.18_09.49.21/pwd.bz2
28 delta/processedBackups/default/2013.08.18_09.49.21/.storeBackupLinks
29 delta/processedBackups/default/2013.08.18_09.49.21/.storeBackupLinks/linkFile.bz2
30 delta/processedBackups/default/2013.08.18_09.49.21/.storeBackupLinks/linkTo
31 repl
32 repl/default
33 repl/default/2013.08.18_09.45.16
34 repl/default/2013.08.18_09.45.16/ls.bz2
35 repl/default/2013.08.18_09.45.16/.md5CheckSums.bz2
36 repl/default/2013.08.18_09.45.16/.md5CheckSums.info
37 repl/default/2013.08.18_09.45.16/.storeBackupLinks
38 repl/default/2013.08.18_10.05.54
39 repl/default/2013.08.18_10.05.54/.md5CheckSums.bz2
40 repl/default/2013.08.18_10.05.54/.md5CheckSums.info
41 repl/default/2013.08.18_10.05.54/.storeBackupLinks
42 repl/default/2013.08.18_10.05.54/.storeBackupLinks/linkFile.bz2
43 repl/default/2013.08.18_10.05.54/.storeBackupLinks/linkTo
44 repl/storeBackupBaseTree.conf
```

Wir stellen fest:

1. In repl

- (a) Das Backup repl/default/2013.08.18_09.45.16 wurde kopiert und vervollständigt.
(.storeBackupLinks/linkFile.bz2 existiert nicht mehr)
- (b) Das Backup repl/default/2013.08.18_10.05.54 wurde kopiert, aber *nicht* vervollständigt.
(.storeBackupLinks/linkFile.bz2 existiert noch)

2. In delta/default

- (a) Das erste Backup 2013.08.18_09.45.16 ist als kopiert und verlinkt markiert – was passt, siehe 1a oben.
- (b) Das zweite Backup, 2013.08.18_09.49.21 ist als kopiert sowie verlinkt markiert – das ist falsch, weil es in repl fehlt. (Ich hatte es gelöscht, um ein Problem zu simulieren.)
- (c) Das dritte Backup 2013.08.18_10.05.54 ist als kopiert, aber nicht verlinkt gekennzeichnet, was richtig ist, siehe 1b oben.

Um das Problem zu lösen, könnte ich alle Backups vollständig von backup nach repl kopieren. Aber in einem realen Einsatz wäre die sehr zeitaufwändig und könnte eventuell zu Datenverlust aufgrund von unterschiedlichen Löschenzenarien führen.

Ein anderer Weg, das Problem zu lösen, ist die Verwendung von linkToDirs.pl, um das fehlende Backup von backup direkt an den korrekten Platz in repl zu kopieren. Siehe Kapitel 6.16 “linkToDirs.pl” für nähere Informationen.

Der „eleganteste“ und schnellste Weg ist, den Delta-Cache auf die Situation in repl anzupassen:

- 1. Das fehlende Backup (siehe 2b oben) ist als „kopiert“ und „linked“ gekennzeichnet, daher lösche ich diese Markierungsdateien. Siehe hierzu das erste Kommando unten.
ACHTUNG: Das funktioniert nur, weil in diesem Beispiel nur eine Replikation konfiguriert ist. Wenn mehrere Replikationen konfiguriert sind, muss der Name der betreffenden Replikation aus den Dateien gelöscht werden. Wenn Du lediglich die Dateien löschst, werden sehr wahrscheinlich andere Replikationen betroffen sein!
- 2. Aber das Backup (welches in repl fehlt) ist nicht an seinem Platz in delta/default. Glücklicherweise ist es noch in delta/processedBackups/default verfügbar. Ich schiebe es mit dem zweiten Kommando unten an die richtige Stelle.

Zum Schluss zeigt die Ausgabe von find, dass alle Dateien an den gewünschten Stellen sind:

```
1 $ rm delta/default/2013.08.18_09.49.21.copied delta/default/2013.08.18_09.49.21.linked
2 $ mv delta/processedBackups/default/2013.08.18_09.49.21 delta/default
3 $ find delta -print | sort
4 delta
5 delta/default
6 delta/default/2013.08.18_09.45.16.copied
7 delta/default/2013.08.18_09.45.16.linked
8 delta/default/2013.08.18_09.49.21
9 delta/default/2013.08.18_09.49.21/.md5CheckSums.bz2
10 delta/default/2013.08.18_09.49.21/.md5CheckSums.info
11 delta/default/2013.08.18_09.49.21/pwd.bz2
12 delta/default/2013.08.18_09.49.21/.storeBackupLinks
13 delta/default/2013.08.18_09.49.21/.storeBackupLinks/linkFile.bz2
14 delta/default/2013.08.18_09.49.21/.storeBackupLinks/linkTo
15 delta/default/2013.08.18_10.05.54
16 delta/default/2013.08.18_10.05.54.copied
17 delta/default/2013.08.18_10.05.54/.md5CheckSums.bz2
18 delta/default/2013.08.18_10.05.54/.md5CheckSums.info
19 delta/default/2013.08.18_10.05.54/.storeBackupLinks
20 delta/default/2013.08.18_10.05.54/.storeBackupLinks/linkFile.bz2
21 delta/default/2013.08.18_10.05.54/.storeBackupLinks/linkTo
22 delta/deltaCache.conf
23 delta/processedBackups
24 delta/processedBackups/default
```

```

25 delta/processedBackups/default/2013.08.18_09.45.16
26 delta/processedBackups/default/2013.08.18_09.45.16/ls.bz2
27 delta/processedBackups/default/2013.08.18_09.45.16/.md5CheckSums.bz2
28 delta/processedBackups/default/2013.08.18_09.45.16/.md5CheckSums.info
29 delta/processedBackups/default/2013.08.18_09.45.16/.storeBackupLinks
30 delta/processedBackups/default/2013.08.18_09.45.16/.storeBackupLinks/linkFile.bz2

```

Ein Lauf von storeBackupUpdateBackup.pl auf repl ergibt folgende Ausgaben:

```

1  $ storeBackupUpdateBackup.pl -b repl
2  BEGIN      2013.08.18 10:39:08 7958 checking references and backup copying in <repl>
3  VERSION    2013.08.18 10:39:08 7958 storeBackupUpdateBackup.pl, 3.4 +
4  INFO       2013.08.18 10:39:08 7958 removing old lock file of process <7050>
5  INFO       2013.08.18 10:39:08 7958 creating lock file </tmp/storeBackup.lock>
6  INFO       2013.08.18 10:39:08 7958 reading <repl/storeBackupBaseTree.conf>
7  INFO       2013.08.18 10:39:08 7958 reading </tmp/x/delta/deltaCache.conf>
8  INFO       2013.08.18 10:39:08 7958 copying </tmp/x/delta/default/2013.08.18_09.49.21> to <repl/default>
9  INFO       2013.08.18 10:39:08 7958 scanning directory <repl> for existing backups
10 INFO       2013.08.18 10:39:08 7958 scanning directory <repl/default> for existing backups
11 STATISTIC  2013.08.18 10:39:08 7958 found 1 backup series, 3 backups, 0 renamed backups
12 INFO       2013.08.18 10:39:08 7958      1 directory </tmp/x/repl/default/2013.08.18_09.45.16> has no
    ↳ linkFrom entry
13 INFO       2013.08.18 10:39:08 7958 autorepair: wrote linkFrom from
    ↳ </tmp/x/repl/default/2013.08.18_09.45.16> to
14 </tmp/x/repl/default/2013.08.18_09.49.21>
15 INFO       2013.08.18 10:39:08 7958
16 INFO       2013.08.18 10:39:08 7958 ----- repeating consistency check -----
17 INFO       2013.08.18 10:39:08 7958 scanning directory <repl> for existing backups
18 INFO       2013.08.18 10:39:08 7958 scanning directory <repl/default> for existing backups
19 STATISTIC  2013.08.18 10:39:08 7958 found 1 backup series, 3 backups, 0 renamed backups
20 INFO       2013.08.18 10:39:08 7958      1 directory </tmp/x/repl/default/2013.08.18_09.49.21> has no
    ↳ linkFrom entry
21 INFO       2013.08.18 10:39:08 7958 autorepair: wrote linkFrom from
    ↳ </tmp/x/repl/default/2013.08.18_09.49.21> to
22 </tmp/x/repl/default/2013.08.18_10.05.54>
23 INFO       2013.08.18 10:39:08 7958
24 INFO       2013.08.18 10:39:08 7958 ----- repeating consistency check -----
25 INFO       2013.08.18 10:39:08 7958 scanning directory <repl> for existing backups
26 INFO       2013.08.18 10:39:08 7958 scanning directory <repl/default> for existing backups
27 STATISTIC  2013.08.18 10:39:08 7958 found 1 backup series, 3 backups, 0 renamed backups
28 INFO       2013.08.18 10:39:08 7958 consistency check finished successfully
29 INFO       2013.08.18 10:39:08 7958 listing references:
30 INFO       2013.08.18 10:39:08 7958      /tmp/x/repl/default/2013.08.18_09.49.21
31 INFO       2013.08.18 10:39:08 7958      -> /tmp/x/repl/default/2013.08.18_09.45.16
32 INFO       2013.08.18 10:39:08 7958      /tmp/x/repl/default/2013.08.18_10.05.54
33 INFO       2013.08.18 10:39:08 7958      -> /tmp/x/repl/default/2013.08.18_09.49.21
34 INFO       2013.08.18 10:39:08 7958 (1/2) updating </tmp/x/repl/default/2013.08.18_09.49.21>
35 INFO       2013.08.18 10:39:08 7958 phase 1: mkdir, symlink and compressing files
36 STATISTIC  2013.08.18 10:39:08 7958 created 0 directories
37 STATISTIC  2013.08.18 10:39:08 7958 created 0 symbolic links
38 STATISTIC  2013.08.18 10:39:08 7958 compressed 0 files
39 STATISTIC  2013.08.18 10:39:08 7958 used 0.0 instead of 0.0 (0 <- 0 ; 0.0%)
40 INFO       2013.08.18 10:39:08 7958 phase 2: setting hard links
41 STATISTIC  2013.08.18 10:39:08 7958 linked 1 files
42 INFO       2013.08.18 10:39:08 7958 phase 3: setting file permissions
43 STATISTIC  2013.08.18 10:39:08 7958 set permissions for 2 files
44 INFO       2013.08.18 10:39:08 7958 phase 4: setting directory permissions
45 STATISTIC  2013.08.18 10:39:08 7958 set permissions for 0 directories
46 INFO       2013.08.18 10:39:08 7958 reading <repl/storeBackupBaseTree.conf>
47 INFO       2013.08.18 10:39:08 7958 marked <default/2013.08.18_09.49.21> as linked in </tmp/x/delta>
48 INFO       2013.08.18 10:39:08 7958 scanning directory <repl> for existing backups
49 INFO       2013.08.18 10:39:08 7958 scanning directory <repl/default> for existing backups
50 STATISTIC  2013.08.18 10:39:08 7958 found 1 backup series, 3 backups, 0 renamed backups
51 INFO       2013.08.18 10:39:08 7958 consistency check finished successfully
52 INFO       2013.08.18 10:39:08 7958 listing references:
53 INFO       2013.08.18 10:39:08 7958      /tmp/x/repl/default/2013.08.18_10.05.54
54 INFO       2013.08.18 10:39:08 7958      -> /tmp/x/repl/default/2013.08.18_09.49.21
55 INFO       2013.08.18 10:39:08 7958 (2/2) updating </tmp/x/repl/default/2013.08.18_10.05.54>
56 INFO       2013.08.18 10:39:08 7958 phase 1: mkdir, symlink and compressing files
57 STATISTIC  2013.08.18 10:39:08 7958 created 0 directories
58 STATISTIC  2013.08.18 10:39:08 7958 created 0 symbolic links
59 STATISTIC  2013.08.18 10:39:08 7958 compressed 0 files

```

```

60 STATISTIC 2013.08.18 10:39:08 7958 used 0.0 instead of 0.0 (0 <- 0 ; 0.0%)
61 INFO      2013.08.18 10:39:08 7958 phase 2: setting hard links
62 STATISTIC 2013.08.18 10:39:08 7958 linked 2 files
63 INFO      2013.08.18 10:39:08 7958 phase 3: setting file permissions
64 STATISTIC 2013.08.18 10:39:08 7958 set permissions for 2 files
65 INFO      2013.08.18 10:39:08 7958 phase 4: setting directory permissions
66 STATISTIC 2013.08.18 10:39:08 7958 set permissions for 0 directories
67 INFO      2013.08.18 10:39:08 7958 reading <repl/storeBackupBaseTree.conf>
68 INFO      2013.08.18 10:39:08 7958 marked <default/2013.08.18_10.05.54> as linked in </tmp/x/delta>
69 INFO      2013.08.18 10:39:08 7958 scanning directory <repl> for existing backups
70 INFO      2013.08.18 10:39:08 7958 scanning directory <repl/default> for existing backups
71 STATISTIC 2013.08.18 10:39:08 7958 found 1 backup series, 3 backups, 0 renamed backups
72 INFO      2013.08.18 10:39:08 7958 consistency check finished successfully
73 INFO      2013.08.18 10:39:08 7958 found no references to backups from lateLinks that need
    ↪ storeBackupUpdateBackup run
74 INFO      2013.08.18 10:39:08 7958 backup </tmp/x/delta/default/2013.08.18_09.49.21> copied to <Backup
    ↪ Copy>
75 INFO      2013.08.18 10:39:08 7958 moving backup to </tmp/x/delta/processedBackups/default>
76 INFO      2013.08.18 10:39:08 7958 backup </tmp/x/delta/default/2013.08.18_10.05.54> copied to <Backup
    ↪ Copy>
77 INFO      2013.08.18 10:39:08 7958 moving backup to </tmp/x/delta/processedBackups/default>
78 INFO      2013.08.18 10:39:08 7958 deleting in deltaCache </tmp/x/delta> processedBackups
79 INFO      2013.08.18 10:39:08 7958 age for deletion is > 99d (delete backups older than Sat 2013.05.11
    ↪ 10:39:08)
80 INFO      2013.08.18 10:39:08 7958 checking series <default>
81 INFO      2013.08.18 10:39:08 7958 default -> 2013.08.18_09.45.16 - not old enough to delete
82 INFO      2013.08.18 10:39:08 7958 default -> 2013.08.18_09.49.21 - not old enough to delete
83 INFO      2013.08.18 10:39:08 7958 default -> 2013.08.18_10.05.54 - not old enough to delete
84 STATISTIC 2013.08.18 10:39:08 7958 duration = 1s
85 STATISTIC 2013.08.18 10:39:08 7958 [sec] | user | system
86 STATISTIC 2013.08.18 10:39:08 7958 -----+-----+-----
87 STATISTIC 2013.08.18 10:39:08 7958 process| 0.12| 0.01
88 STATISTIC 2013.08.18 10:39:08 7958 childs | 0.04| 0.00
89 STATISTIC 2013.08.18 10:39:08 7958 -----+-----+-----
90 STATISTIC 2013.08.18 10:39:08 7958 sum | 0.16| 0.01 => 0.17
91 INFO      2013.08.18 10:39:08 7958 syncing ...
92 END      2013.08.18 10:39:08 7958 checking references and copying in <repl>

```

Folgendes passierte:

- Das fehlende Backup wurde vom Delta Cache kopiert (aus Log von oben):

```
1 copying </tmp/x/delta/default/2013.08.18_09.49.21> to <repl/default>
```

Das letzte Backup (2013.08.18_10.05.54) wurde nicht kopiert, weil das schon erfolgte.

- Das Programm führte einige Reparaturen ("auto repair") automatisch aus, was bei Replikationen normal ist.
- Das Programm entdeckte zwei unvollständige Backups, in denen Hardlinks usw. zu ergänzen waren:

```

1 listing references:
2 /tmp/x/repl/default/2013.08.18_09.49.21
3 -> /tmp/x/repl/default/2013.08.18_09.45.16
4 /tmp/x/repl/default/2013.08.18_10.05.54
5 -> /x/repl/default/2013.08.18_09.49.21

```

Anschließend vervollständigte es beide Backups.

Zum Schluss überprüfe ich mit storeBackupCheckBackup.pl, ob die Replikation korrekt verlief:

```

1 $ storeBackupCheckBackup.pl -c repl
2 BEGIN      2013.08.18 10:42:50 8042 checking backups in </tmp/x/repl>
3 VERSION    2013.08.18 10:42:50 8042 storeBackupCheckBackup.pl, 3.4 +
4 INFO       2013.08.18 10:42:50 8042 scanning directory <repl> for existing backups
5 INFO       2013.08.18 10:42:50 8042 scanning directory <repl/default> for existing backups
6 STATISTIC  2013.08.18 10:42:50 8042 found 1 backup series, 3 backups, 0 renamed backups
7 INFO       2013.08.18 10:42:50 8042 consistency check finished successfully
8 INFO       2013.08.18 10:42:50 8042 found no references to backups from lateLinks that need
    ↪ storeBackupUpdateBackup run

```

```

9  INFO      2013.08.18 10:42:50 8042 backup directories to check
10 INFO      2013.08.18 10:42:50 8042 /tmp/x/repl/default/2013.08.18_09.45.16
11 INFO      2013.08.18 10:42:50 8042 /tmp/x/repl/default/2013.08.18_09.49.21
12 INFO      2013.08.18 10:42:50 8042 /tmp/x/repl/default/2013.08.18_10.05.54
13 INFO      2013.08.18 10:42:50 8042 -- checking </tmp/x/repl/default/2013.08.18_09.45.16> ...
14 INFO      2013.08.18 10:42:50 8042 -- checking </tmp/x/repl/default/2013.08.18_09.49.21> ...
15 INFO      2013.08.18 10:42:50 8042 -- checking </tmp/x/repl/default/2013.08.18_10.05.54> ...
16 INFO      2013.08.18 10:42:50 8042 -- no WARNINGS OCCURRED DURING THE CHECK! --
17 INFO      2013.08.18 10:42:50 8042 -- no ERRORS OCCURRED DURING THE CHECK! --
18 END       2013.08.18 10:42:50 8042 checking backups in </tmp/x/repl>

```

9 Verwendung von storeBackup (Beispiele)

9.1 Einige Informationen zu Beginn

Bevor hier einige Beispiele erläutert werden: Es ist nicht schlecht, wenn Du Dir darüber im Klaren bist, was Du tust und was passiert. Hier werden zuerst einige wichtige Aspekte darüber, wie storeBackup arbeitet, beschrieben. (Die folgenden Erklärungen betreffen die prinzipielle Abarbeitung; aus Gründen der Performance ist die Implementierung etwas anders. Es gibt diverse Warteschlangen, Parallelitäten und einen kleinen Scheduler in `storeBackup.pl`, die hier nicht beschrieben werden.)

storeBackup verwendet mindestens zwei interne zeilenorientierte Dateien in jedem erzeugten Backup:

- `.md5CheckSums.info` – allgemeine Informationen über das Backup
- `.md5CheckSums[.bz2]` – Information über jede gespeicherte Datei (dir, etc.)

Wenn `storeBackup.pl` gestartet wird, passiert (neben einigem anderen) Folgendes:

1. Lies den Inhalt der vorherigen `.md5CheckSums[.bz2]`-Datei und speichere den Inhalt in zwei Indexdateien:
`dbm(md5sum)` und `dbm(filename)`. (`dbm(md5sum)` bedeutet, dass als Hash die md5-Summe verwendet wird, bei `dbm(filename)` ist es der Dateiname.) Standardmäßig werden diese Tabellen im Hauptspeicher gehalten.
2. Lies den Inhalt der anderen `.md5CheckSums[.bz2]`-Dateien (`otherBackupSeries`) und speichere sie im Hash `dbm(md5sum)`. Speichere immer die letzte Kopie in der `dbm`-Struktur, falls zwei unterschiedliche Dateien (z. B. von unterschiedlichen Backup-Serien) identisch sind. Dieses bewirkt, dass unterschiedliche Versionen derselben Datei aus unterschiedlichen Backups in künftigen Backups zusammengelegt werden.
- Hier wird beschrieben, wie `storeBackup.pl` arbeitet, wenn keine Dateien aus anderen Serien verlinkt werden (einfaches Backup), siehe Beispiel 1, section 9.2 und Beispiel 2, Kapitel 9.3:
 In einer Schleife über alle Dateien mache Folgendes:
 1. Sieh nach, ob in `dbm(filename)` – das alle Dateien vom vorherigen Backup enthält –, exakt dieselbe unveränderte Datei existiert, verwende dazu die Meta-Informationen aus `dbm(filename)`, insbes. die md5-Summe.
 Erzeuge einen Hardlink auf die schon bestehende Datei und gehe zu Schritt 3.
 2. Berechne die md5-Summe von der zu sichernden Datei und suche in `dbm(md5sum)` nach dieser md5-Summe.
 Wenn sie dort existiert, erzeuge einen Hardlink auf die betreffende Datei.
 Wenn sie nicht existiert, kopiere oder komprimiere die Datei.
 3. Schreibe die Informationen über die neue Datei in die entsprechende `.md5CheckSums[.bz2]` Datei.
- Hier wird beschrieben, wie storeBackup arbeitet, wenn Dateien mit einer anderen Backup-Serie verlinkt werden, siehe example 3, section 9.4 and example 4, section 9.5.
 In einer Schleife über alle Dateien mache Folgendes:
 1. Sieh nach, ob in `dbm(filename)` – das alle Dateien vom vorherigen Backup enthält –, exakt dieselbe unveränderte Datei existiert, verwende dazu die Meta-Informationen aus `dbm(filename)`.

(Da jetzt auf mehrere, unabhängige Backups verlinkt wird, kann es sein, dass eine Datei gleichen Inhalts in den anderen Backups existiert. Daher muss `storeBackup.pl` in `dbm(md5sum)` nachsehen, ob dorthin verlinkt werden könnte, um immer auf dieselbe Datei zu verlinken.)

2. Berechne die md5-Summe der Datei, falls sie nicht aus Schritt 1 bekannt ist.
Suche in `dbm(md5sum)` nach dieser md5-Summe.
Wenn sie existiert, erzeuge einen entsprechenden Hardlink.
Wenn sie nicht existiert, kopiere oder komprimiere die Datei.
3. Schreibe die Informationen über die neue Datei in die entsprechende `.md5CheckSums[.bz2]` Datei.

- Hier wird der (prinzipielle) Ablauf bei Verwendung der Option `lateLinks` beschrieben. Siehe auch Beispiel 6, Kapitel 9.7 unten

Wenn Du Deine Backups auf einen NFS-Server speicherst, wird die meiste Zeit für das Setzen der Hardlinks verwendet. Einen Hardlink zu setzen geht sehr schnell, aber wenn das viele tausend sind, dauert es auch seine Zeit. Du kannst das Warten auf die Hardlinks vermeiden, indem Du die Option `lateLinks` verwendest:

1. Erstelle ein Backup mit `storeBackup.pl` und setze `--lateLinks` (or `lateLinks = yes`) in der Konfigurationsdatei. `StoreBackup.pl` wird dann keine Hardlinks generieren; es wird lediglich eine Datei geschrieben, in der die Informationen darüber, was hätte gelinkt werden sollen, enthalten ist.
Es handelt sich beim neuen Backup also erst einmal um ein inkrementelles Backup.
2. in einem eigenständigen Schritt wird `storeBackupUpdateBackup.pl` aufgerufen, um alle benötigten Hardlinks zu setzen, damit aus dem nicht fertiggestellten Backup wieder ein vollständiges Backup wird. Siehe auch Kapitel 7.6, Verwendung der Option `lateLinks` für weitergehende Erklärungen.

Fazit:

1. Lösche kein Backup, in dem die Hardlinks noch nicht gesetzt wurden. `storeBackupUpdateBackup.pl` erzeugt diese. Es ist eine gute Idee, zum Löschen von Backups ausschließlich `storeBackup.pl` oder `storeBackupDel.pl` zu verwenden.
2. Die gemeinsame Verwendung von Plattenplatz über mehrere Backups erfolgt immer über Hardlinks. Das bedeutet:
 - Eine Backup-Serie kann nicht über mehrere Dateisystem verteilt werden.
 - Wenn Du Daten zwischen unterschiedlichen Backup-Serien verlinken willst, müssen diese im selben Dateisystem angelegt sein.
3. Die Informationen über ein Backup in den `.md5CheckSums` Dateien wird mit relativen Pfadnamen gespeichert. Es spielt daher keine Rolle, wenn Du den absoluten Pfad zum Backup änderst oder von einem anderen Rechner sicherst (Server macht Backup vom Client über NFS – Client macht Backup zum Server über NFS).
(Noch) nicht gesetzte Hardlinks zu anderen Backup-Serien (über die Option `lateLinks` werden ebenfalls mit relativen Pfaden gespeichert. Das bedeutet: Da kannst `backupDir` verschieben wie Du willst, aber Du solltest niemals den relativen Pfad zwischen den Backup-Serien, ändern bevor alle Links mit `storeBackupUpdateBackup.pl` erzeugt wurden.

Wenn Du weitere Ideen oder Fragen hast, kannst Du mich gern kontaktieren ([hjclaes\(at\)web.de](mailto:hjclaes(at)web.de)).

Es ist eine gute Idee, die Konfigurationsdatei anstelle der Kommandozeile zu verwenden. Ruf einfach

```
1 # storeBackup.pl --generate <configFile>
```

auf.

Editiere die Konfigurationsdatei und rufe `storeBackup.pl` folgendermaßen auf:

```
1 # storeBackup.pl -f <configFile>
```

Du kannst die Einstellungen in der Konfigurationsdatei auf der Kommandozeile überschreiben (siehe Beispiel 6).

9.2 Beispiel 1, sehr einfaches Backup

Dies ist eine einfache Konfiguration mit `storeBackup`, bei der nur zwei Optionen verwendet werden (das Quellverzeichnis und das Backup-Zielverzeichnis) sowie als optionaler Parameter der Name der Logdatei. Diese Konfiguration erzeugt ein Backup vom Quellverzeichnis `/home/jim` nach `/backup`:

```
1 # storeBackup.pl --sourceDir /home/jim --backupDir /backup/jim --logFile /tmp/storeBackup.log
```

Die Option `--logFile` ist optional und bewirkt, dass `storeBackup.pl` anstelle nach `stdout` (Terminal) in die Log Datei `/tmp/storeBackup.log`

Die Option `backupDir` gibt das Zielverzeichnis an – ein externes USB Laufwerk oder etwas anderes, auf das die Daten kopiert werden. Siehe Kapitel 3, Erste Schritte für weitere Informationen. Falls Du dann weitere Fragen haben solltest, siehe Kapitel 6.2.1, `storeBackup.pl` Optionen und insbesondere die `--backupDir` Option.

9.3 Beispiel 2, Backup von mehreren Verzeichnissen

Aus historischen Gründen (Kompatibilität) kann `storeBackup.pl` nur ein Verzeichnis sichern. Aber dieser Nachteil wandelt sich in einen Vorteil, wenn die Option `followLinks` verwendet wird, da durch sie die Konfiguration sehr einfach und flexibel wird.

Da kannst ebenso das aus anderen Programmen bekannte Verfahren mit `includeDirs` und `exceptDirs` verwenden. Aber das ist deutlich umständlicher und unschön zu handhaben.

Um `lateLinks` zu verwenden, führe die folgenden Schritte durch. (Ich gehe hier davon aus, dass Du `/home/greg/important`, `/home/jim` und `/etc` nach `/backup/stbu` sichern willst.) Du kannst das später sehr einfach ändern. Zuerst leg ein spezielle Verzeichnis an, z. B. `/opt/stbu`. Lass uns des Weiteren annehmen, dass Du `storeBackup` in `/opt/storeBackup` ausgepackt hast:

```
1 # mkdir /opt/stbu
2 # cd /opt/stbu
3 # ln -s /opt/storeBackup storeBackup
4 # ln -s /home/jim home_jim
5 # ln -s /etc etc
6 # ln -s /home/greg/important home_greg_important
7 # ln -s . backup
```

Mit dem ersten symbolischen Link stellen wir sicher, dass `storeBackup` selbst Teil des Backups sein wird. Daher können wir `storeBackup` selbst später mit `cp` zurückholen und anschließend `storeBackupRecover.pl` für alles andere verwenden.

Der letzte symbolische Link ist ein Trick, um eine exakte Kopie von `/opt/stbu` im Backup zu erhalten. Jetzt solltest Du ein Skript schreiben, um `storeBackup.pl` zu starten. Speichere es als `/opt/stbu/backup.sh`:

```
1 /opt/storeBackup/bin/storeBackup.pl -s /opt/stbu -b /backup/stbu -S . -l /tmp/storeBackup.log
  ↪ --followLinks 1
```

Durch Setzen von `--followLinks` auf 1 wird `storeBackup.pl` die erste Verzeichnis-Ebene von symbolischen Links wie „richtige“ Verzeichnisse behandeln. Daher findest Du dann `home_jim` als Verzeichnis-Eintrag in Deinem Backup.

Nun setze noch die richtigen Rechte auf das Skript:

```
1 chmod 700 /opt/backup/backup.sh
```

Immer wenn Du dieses Skript startest, wird ein (weiteres) Backup von den gewählten Verzeichnissen und Deinem kleinen Skript durchgeführt. Du musst `root` sein, um die notwendigen Rechte zu Sicherung der benannten Verzeichnisse zu haben. Und natürlich solltest Du Schreibrechte in `/backup/stbu` haben.

Jetzt kannst Du die zu sichernden Verzeichnisse durch einfaches Löschen oder Hinzufügen von symbolischen Links in verändern.

9.4 Beispiel 3, ein kleines Backup täglich, ein großes einmal die Woche

In diesem Beispiel willst Du ein großes Backup vom ganzen Rechner mit Hilfe von `exceptDirs` und ein kleines für einige spezielle Verzeichnisse mit `followLinks` einrichten. Natürlich könntest Du das auch anders herum machen, nur `followLinks` verwenden, nur `includeDirs` oder Kombinationen verwenden.

Im Folgenden nehmen wir an, Du willst:

1. Dein Rechner mountet `/net`, was Du nicht sichern willst, von einem anderen Rechner.
2. Du möchtest ebenfalls `/tmp` und `/var/tmp` nicht sichern.
3. Du möchtest ein Backup des gesamten Rechners einmal pro Woche nach `/net/server/backup/weekly` sichern.
4. Du möchtest `/home/jim` und `/home/tom/texts` nach Arbeitsende abends schneller sichern.
5. Selbstverständlich möchtest Du die Dateien zwischen den beiden Backup-Serien mit Hardlinks verlinken.
6. Falls Du beide Skripte gleichzeitig startest, werden neue Dateien nicht zwischen den beiden Backups verlinkt. Über die Zeit wird das trotzdem erfolgen. Aber Du solltest die Skripte vor allem nicht gleichzeitig starten, wenn sie das erste Mal laufen! In diesem Fall würden alle mehrfach vorkommenden Dateien nicht verlinkt!
7. Du willst die Option `lateLinks`, die Deine Backups deutlich schneller machen würde, nicht verwenden, weil Du auf dem NFS-Server keine Skripte laufen lassen kannst (oder warum auch immer).

Um die oben beschriebenen Aktivitäten vorzubereiten, benötigst Du Folgendes:

Für das tägliche Backup erstellst Du ein gesondertes Verzeichnis (wir verwenden `followLinks`) wie in Beispiel 2 beschrieben (storeBackup liegt in diesem Beispiel wiederum in `/opt/storeBackup`:

```

1 # mkdir /opt/small-backup
2 # cd /opt/small-backup
3 # ln -s . small-backup
4 # ln -s /home/jim home_jim
5 # ln -s /home/tom/texts home_tom_texts

```

und schreibst ein Backup-Skript `byBackup.sh`:

```

1 #! /bin/sh
2 /opt/storeBackup/bin/storeBackup.pl -s /opt/small-backup
3 -b /net/server/backup -S daily -l /tmp/storeBackup.log --followLinks 1 0:weekly

```

Dann schreibst Du das Skript für die wöchentlichen Backups:

```

1 #! /bin/sh
2 /opt/storeBackup/bin/storeBackup.pl -s / -b /net/server/backup -S weekly -l
↪ /tmp/storeBackup.log --exceptDirs net -e tmp -e var/tmp -e proc -e sys -e dev 0:daily

```

Die „0“ vor den Pfaden (wie `0:daily`) bedeutet, dass das letzte Backup der anderen Serie zum Suchen nach identischen Dateien verwendet wird.

Und – natürlich – sollten die Verzeichnisse `weekly` und `daily` innerhalb von `/net/server/backup` auf dem NFS-Server existieren.

Wie Du siehst, werden die Kommandozeilenoptionen ein wenig verwirrend. Wenn Du solche Konfigurationen erstellst, solltest Du möglichst eine Konfigurationsdatei mit `storeBackup.pl -g configFile` erstellen und diese stattdessen verwenden.

9.5 Beispiel 4, Backup von unterschiedlichen Rechnern, Daten geteilt

Dieses Beispiel zeigt, wie man nicht-koordinierte Backups von unterschiedlichen Rechnern durchführt und die Daten über Hardlinks teilt.

Stell Dir vor, Du hast die folgenden Randbedingungen:

1. Du hast einen Server namens „server“ mit einer separaten Platte, die als `/disk1` gemountet ist.
2. Du willst den Rechner „client1“, der `/disk1` vom Server nach `/net/server/disk1` mountet, in dieses Verzeichnis als Serie „client1“ sichern.
3. Du willst Rechner „client2“, der `/disk1` vom Server nach `/net/server/disk1` mountet, in dieses Verzeichnis als Serie „client2“ sichern.
4. Das Backup des Servers läuft unabhängig von den anderen nachts.

5. Die Backups von den Clients laufen unkoordiniert, also eventuell gleichzeitig
6. Die identischen Dateien in den Backups sollen über Hardlinks verbunden werden.
7. Du könntest zusätzlich kleine Backups von Teilen der Quellen (inklusive dem Teilen von Daten) machen, aber das wäre dasselbe Verfahren wie das hier beschriebene und wird daher nicht weiter ausgeführt.
8. Wenn Du eine Client/Server-Architektur wie diese hast, empfiehlt es sich, die Option `lateLinks` zu verwenden, um die Backups zu beschleunigen. Beispiel 6 erklärt, wie man das macht.

Schreibe das folgende Skript für den Server:

```
1  #! /bin/sh
2  <PATH>storeBackup.pl -s / -b /disk1 -S server -l /tmp/storeBackup.log -e /tmp -e /var/tmp -e
   ↪ /disk1 -e /sys -e /dev -e /proc 0:client1 0:client2
```

das folgende Skript für Client 1:

```
1  #! /bin/sh
2  <PATH>/storeBackup.pl -s / -b /net/server/disk1 -S client1 -l /tmp/storeBackup.log -e /tmp -e
   ↪ /var/tmp -e /disk1 -e /sys -e /dev -e /proc 0:server 0:client2
```

und das folgende Skript für Client 2:

```
1  #! /bin/sh
2  <PATH>/storeBackup.pl -s / -b /net/server/disk1 -S client2 -l /tmp/storeBackup.log -e /tmp -e
   ↪ /var/tmp -e /disk1 -e /sys -e /dev -e /proc 0:server 0:client1
```

9.6 Beispiel 5, unterschiedliche Aufbewahrungsfristen für verschiedene Verzeichnisse

Du kannst dies sehr leicht mit dem von den vorherigen Beispielen bekannten Trick erreichen. Nehmen wir an, Du willst Dein Backup für 60 Tage und alle Dateien im Verzeichnis „nichtWichtig“ für nur 7 Tage aufbewahren.

Mach einfach zwei Backups, eins mit `--keepAll 60d` und nimm das Verzeichnis „nichtWichtig“ mit `exceptDirs` aus. Mach ein zweites Backup mit `--keepAll 7d` für das fehlende Verzeichnis. Das Zielverzeichnis für beide Backups ist dasselbe. Dadurch sind die nötigen Relationen zwischen den Backups vorhanden - wenn Du jetzt eine Datei zwischen „nichtWichtig“ und den an deren Verzeichnissen kopierst oder verschiebst, wird sie keinen unnötigen Platz im Backup belegen.

9.7 Beispiel 6, Verwendung von `lateLinks`

Nach dem Lesen der vorherigen Beispiele sollte es für Dich kein Problem mehr sein zu verstehen, wie mehrere Verzeichnisse (siehe Beispiel 2) oder über Kreuz verlinkte Backups (siehe Beispiele 3 und 4) zu konfigurieren ist. Ich nehme jetzt an, dass Du eine Konfigurationsdatei erzeugst:

```
1  # storeBackup.pl -g stbu.conf
```

- Konfiguriere `storeBackup` so, dass eine Sicherung in das Backup-Verzeichnis über NFS gemacht wird. Dazu setzt Du in den Einstellungen der Konfigurationsdatei `stbu.conf` u.a. Folgendes:

```
1  lateLinks = yes
2  lateCompress = yes
3  doNotDelete = yes
```

Wenn Du eine hohe Bandbreite zum Server hast, besteht keine Notwendigkeit, `lateCompress` von `no` in `yes` zu ändern. Weil `doNotDelete` auf `yes` gesetzt ist, musst Du nicht auf das Löschen älterer Backups warten.

- Lass die Backup(s) laufen. (Wie immer ist das erste Backup ziemlich langsam.) Das ist alles, was Du von Seiten des NFS Clients zu tun hast.
- Starte (später über `cron`) auf dem Server (NFS-Server = Backup Server):

```
1 storeBackupUpdateBackup.pl -b <backupDirDir> -l /tmp/stbuUpdate.log
```

um die fehlenden Hardlinks (und anderes) zu erzeugen.

- Starte (später via cron) auf dem Server:

```
1 storeBackupDel.pl -f <cf1> -b <backupDirDir> --unset doNotDelete
```

Dies überschreibt (mit „unset“) das `doNotDelete`-Flag in der Konfigurationsdatei.

Du musst `backupDir` in dem Kommando oben auf dasselbe Verzeichnis, das Du in der Konfigurationsdatei von `storeBackup.pl` angegeben hast, setzen. Der Pfad auf dem Client und auf dem Server können dabei – je nach Mountpunkten – unterschiedlich sein. Wenn er gleich ist, musst Du ihn nicht in der Kommandozeile überschreiben. (Du kannst mehr über die Verwendung von Konfigurationsdatei und Kommandozeile im Kapitel 7.1 lesen.)

Du kannst auch das allererste Backup mit der `lateLinks` Option laufen lassen. Natürlich muss danach `storeBackupUpdateBackup.pl` laufen, um ein vollständiges Backup zu erhalten.

Detaillierte Erläuterungen über die Option `lateLinks` finden sich in Verwendung der Option `lateLinks`, siehe Kapitel 7.6.

10 FAQ, Frequently asked Questions

- 1 Ich will keine Dateien im Backup komprimieren
- 2 Wo ist die graphische Oberfläche (GUI)?
- 3 Ich brauche dieses `lateLinks` nicht
- 4 Erstellen eines Backups über ssh (nicht NFS) auf einen entfernten Server
- 5 Ich will dieses blocked file und will es für alle Dateien größer als 50 MB nutzen
- 6 Wie mache ich ein vollständiges Backup meines GNU/Linux Rechners?
- 7 Wie installiere ich `storeBackup` auf einem (Synology) NAS?
- 8 `storeBackup` auf Raspberry Pi
- 9 Können `storeBackup` die Hardlinks ausgehen?

* * * * *

FAQ 1 Ich will keine Dateien im Backup komprimieren

Ich will keine Dateien im Backup komprimieren. Wie kann ich das konfigurieren?

Beim Konfigurieren von `storeBackup.pl` die Option `exceptSuffix` auf `'.*'` setzen. Das ist das Suchmuster für „passt auf alles“.

* * * * *

FAQ 2 Wo ist die graphische Oberfläche (GUI)?

Es gibt einige Gründe, warum `storeBackup` über die Kommandozeile anzusprechen ist:

- Wenn möglich, solltest Du Deine Backup regelmäßig mit einem Automatismus durchführen, z. B. mit cron.
- Falls `storeBackup` auf einem Server läuft, gibt es dort wahrscheinlich keine GUI. Denke auch an die Abhängigkeiten einer Oberfläche zu unterschiedlichen Versionen der GUI-Bibliotheken.
- Wenn Du Daten auf ein irgendwie korruptes System spielen willst, wird die GUI (falls eine lief) eventuell nicht mehr starten. Dann ist es gut, ein Tool zu haben, dass von einer Kommandozeile oder von einer Recovery-CD verwendet werden kann. Es ist auch sinnvoll, `storeBackup` selbst Teil des Backups sein zu lassen (siehe Beispiel 2).
- Wenn Du nur einige Dateien aus dem Backup holen willst, kannst Du einen beliebigen Browser irgendeines Betriebssystems nutzen. Das ist eine Art GUI und das einzige, was Du (neu) lernen musst, ist der Pfad zum Backup.

- Wenn Du eine separate GUI schreiben willst, die `storeBackup` aufruft, unterstütze ich gern!

* * * * *

FAQ 3 Ich brauche dieses `lateLinks` nicht

Ich will einfach nur ein Backup auf ein externes USB Laufwerk machen und will die Option `lateLinks` nicht verwenden. Wie mache ich das?

Du musst Dich überhaupt nicht mit dieser Option (oder mit `storeBackupUpdateBackup.pl`), die für höhere Ansprüche gedacht ist, auseinandersetzen, wenn Du `lateLinks` nicht verwenden willst. Sieh Dir Beispiel 1 an.

* * * * *

FAQ 4 Erstellen eines Backups über ssh (nicht NFS) auf einen entfernten Server

Mit GNU/Linux ist es ebenfalls möglich, ein Backup über eine SSH-Verbindung laufen zu lassen. Das hat den Vorteil, dass kein (eventuell) zusätzliches Netzwerk-Dateisystem (wie bei NFS) konfiguriert werden muss.

Anstelle eines NFS-Mounts auf das Zielverzeichnis wird das Programm `sshfs` verwendet. Es ist für die meisten Distributionen verfügbar, kann jedoch auch von <http://fuse.sourceforge.net/sshfs.html>⁶⁸ heruntergeladen werden.

Das Kommando, das entfernte Verzeichnis `/var/backup` auf dem Rechner `chronos` als User „backup“ nach `/mnt/target` zu mounten ist:

```
1 # sshfs backup@chronos:/var/backup /mnt/target
```

`storeBackup.pl` muss jetzt nur noch so konfiguriert werden, dass das Backup nach `/mnt/target` geschrieben wird. Nach dem Backup kann das Zielverzeichnis mit `fusermount -u /mnt/target` ausgehängt werden.

BESCHLEUNIGUNG EINES REMOTE-BACKUPS ÜBER SSHFS

`sshfs` nutzt eine eigene Netzwerkanfrage für jeden individuellen Hardlink, der gesetzt werden muss, und für jede einzelne Datei, die gelöscht werden muss. Da die Latenz (Dauer) für jede Aktion über das Netzwerk generell um einiges größer ist als für eine lokale, kann ein entferntes (Remote-) Backup ziemlich langsam sein, auch wenn die Bandbreite groß genug ist.

Aus diesem Grund empfiehlt es sich, für derartige Anwendungen von `storeBackup` die Optionen `lateLinks` und `doNotDelete` zu verwenden. Ihre Verwendung ermöglicht es, das Erzeugen von Hardlinks und das (spätere) Löschen lokal auf der entfernten Maschine durchzuführen. Die Verwendung von `lateLinks` resultiert mit `sshfs` generell in einer Verbesserung der Performance um den Faktor 10 bis 75, abhängig von der Anzahl der Änderungen und der Latenz des Netzwerks.

Das Vorgehen ist allgemein:

1. Mounte das Remote-System:

```
1 # sshfs backup@chronos:/var/backup /mnt/target
```

2. Starte das Backup:

```
1 # storeBackup.pl --backupDir /mnt/target --lateLinks --doNotDelete [other options]
```

3. Nimm den Mount wieder weg:

```
1 # fusermount -u /mnt/target
```

4. Setze die Hardlinks auf dem Remote-System:

```
1 # ssh -T -l backup ebox.rath.org 'storeBackupUpdateBackup.pl --backupDir /var/backup'
```

5. Lösche alte Backups auf dem Remote-System:

```
1 # ssh -T -l backup chronos "storeBackupDel.pl --backupDir /var/backup [other options]"
```

⁶⁸<http://fuse.sourceforge.net/sshfs.html>

Beachte, dass dieses Vorgehen erfordert, dass storeBackup auch auf dem Remote-System installiert ist.

* * * * *

FAQ 5 Ich will dieses blocked file und will es für alle Dateien größer als 50 MB nutzen

Um das gewünschte Resultat zu bekommen, setze einfach:

```
1 checkBlocksSuffix = .*
2 checkBlocksMinSize = 50M
```

Diese Konfiguration wird Blocked Files für alle Dateien mit einer Größe von mindestens 50 MB verwenden. Wenn Du das ab einer anderen Größe willst, z. B. 800KB, ändere den Werte von `checkBlocksMinSize` auf 800k.

Erklärung für Experten: `storeBackup.pl` erzeugt von der Konfiguration oben folgende interne Regel:

```
1 '$file =~ /\.*/' and '$size >= 52428800'
```

Du kannst daher auch direkt die folgende Regel definieren:

```
1 '$size >= &::SIZE("50M")'
```

um dasselbe Resultat zu bekommen.

* * * * *

FAQ 6 Wie mache ich ein vollständiges Backup meines GNU/Linux Rechners?

Erzeuge als Erstes eine Konfigurationsdatei:

```
1 storeBackup.pl -g completeLinux.conf
```

Öffne die Konfigurationsdatei mit einem Editor Deiner Wahl und editiere die folgenden Optionen:

```
1 sourceDir = /
```

Setze `sourceDir` auf `/`, so dass das gesamte Dateisystem gesichert wird.

```
1 backupDir=/media/drive
```

Hier nehme ich an, dass die angeschlossene Festplatte für das Backup unter dem Pfad `/media/drive` angebunden ist. Dies muss angepasst werden, wenn sie woanders gemountet ist. Die Backups können natürlich z. B. alternativ auf einem NFS-Mount liegen. Falls Du diese Konfiguration hast, findest Du Erläuterungen zu NFS unter Kapitel 7.11. Wenn Du das Backup über NFS machst, solltest Du auch Kapitel 7.6 lesen.

Als nächstes sind die Verzeichnisse anzugeben, die nicht gesichert werden sollen. Wir müssen hier `backupDir` angeben, um eine Rekursion zu vermeiden.

```
1 exceptDirs= tmp var/tmp proc sys media
```

Füge andere Verzeichnisse, die Du nicht sichern willst (z. B. über NFS gemountete Home-Verzeichnisse), an diese Liste.

Nehmen wir weiterhin an, Du willst die Inhalte aller anderen Verzeichnisse, die `tmp` oder `temp` (groß oder klein geschrieben) heißen und sich irgendwo im Dateisystem befinden, ausschließen. Füge dann hinzu:

```
1 exceptRule= '$file =~ m#/te?mp/#i'
```

Um Cache-Dateien auszuschließen, werden alle Verzeichnisse mit `cache` im Namen (groß oder klein geschrieben) hinzugefügt. Ändere die Zeile von oben in:

```
1 exceptRule= '$file =~ m#/te?mp/#i' or '$file =~ m#cache.*/#i'
```

Aber jetzt gibt es das Risiko, dass vielleicht einige wichtige Dateien nicht gesichert werden, weil sie in einem Verzeichnis mit Namen `/tmp/`, `/temp/` oder in einem Verzeichnis mit z. B. `cache` im Namen liegen. Aus diesem Grunde schreiben wir die Namen aller aufgrund dieser Regel ausgeschlossenen Dateien in eine Datei, die wir nach Durchführung des Backups überprüfen können:

```
1 writeExcludeLog=yes
```

Jetzt wird es in jedem Backup eine Datei `.storeBackup.notSaved.bz2` geben, in der die ausgeschlossenen Dateien gelistet sind.

Um alle Dateitypen zu kopieren (insbesondere Block- und zeichenorientierte Devices in `/dev`), setze:

```
1 cpIsGnu=yes
```

Um ein vollständiges Backup durchzuführen, muss auch der Boot Sektor gespeichert werden. Das folgende Skript geht davon aus, dass dieser auf dem Laufwerk `sda` gespeichert ist. Dies muss eventuell an die Anforderungen Deines Systems angepasst werden. Erzeuge das Verzeichnis `/backup` und speichere das folgende Skript `pre.sh` dort:

```
1 #!/bin/sh
2
3 rm -f /backup/MBR.prior
4 mv /backup/MBR.copy /backup/MBR.prior
5 # copy the boot loader
6 dd if=/dev/sda of=/backup/MBR.copy bs=512 count=1 > /dev/null 2>&1
7
8 # copy back with:
9 # dd if=/backup/MBR.copy of=/dev/sda bs=512 count=1
```

Setze die Rechte:

```
1 chmod 755 /backup/pre.sh
```

Setze `precommand` in der Konfigurationsdatei um das Skript aufzurufen:

```
1 precommand = /backup/pre.sh
```

Um während des Backup zu sehen, dass etwas passiert, setze:

```
1 progressReport = 2000
2 printDepth = yes
```

Schau Dir die `keep*`-Optionen an und setze sie auf für Dich sinnvolle Werte. Außerdem solltest Du `logFile` auf einen für Dich passenden Werte setzen – überprüfe ebenso die anderen Einstellmöglichkeiten.

Wie immer wird das erste Backup einige Zeit brauchen, da für alle Dateien md5-Summen berechnet werden müssen – insbesondere aber wegen der Kompression der Dateien. Das nächste Backup wird *sehr* viel schneller sein.

Nach der Durchführung des Backups solltest Du kontrollieren, welche Dateien aufgrund der Option `exceptRule` *nicht* im Backup sind.

* * * * *

FAQ 7 Wie installiere ich storeBackup auf einem (Synology) NAS?

Folgendes Verfahren hat zum Erfolg geführt.

Vorbereitung:

- aktiviere `ssh` auf dem NAS
- installiere den Paketmanager (<http://www.synology-wiki.de/index.php/IPKG>)

Installation:

- Installation von `storeBackup` in ein beliebiges Verzeichnis, z. B.: `/volume1/homes/admin/storeBackup`. (`/volume1` ist der Mountpoint der Festplatte bzw. der nutzbaren Partition; unter `homes` liegen die Home-Verzeichnisse der Userkonten, auch das vom `admin`)
- als `admin` einloggen: `ssh admin@ip-des-nas`
- Sicherstellen, dass das x-Bit gesetzt ist:

```
# chmod u+x /volume1/homes/admin/storeBackup/bin/*
# chmod u+x /volume1/homes/admin/storeBackup/lib/stbuMd5*
```

- Nach `/usr/bin` verlinken:
`ln -s /volume1/homes/admin/storeBackup/bin/* /usr/bin`
- `md5deep` installieren:
`ipkg install md5deep`
- `md5deep` als `md5sum` nach `/usr/bin` verlinken:
`ln -s /opt/bin/md5deep /usr/bin/md5sum`

Damit sollte `storeBackup` lauffähig sein.

* * * * *

FAQ 8 `storeBackup` auf Raspberry Pi

Mir wurde berichtet, dass `storeBackup` auf dem Raspberry Pi (`raspbmc` und Raspbian GNU/Linux 7) läuft. Man sollte auf Folgendes achten:

- setze `noCompress` auf 1 (mehr als ein Komprimier-Job macht keinen Sinn).
- Verwende Option `saveRAM` und stell sicher, dass in Deinem temporären Verzeichnis genügend Platz ist. Es scheint am besten, ein eigenes Directory für temporäre Dateien anzulegen und `storeBackup.pl` über die Option `tmpdir` darauf zu verweisen. Zumindest mit `raspbmc` scheint der temporäre Bereich (`/tmp`) so klein zu sein, dass es ansonsten auch ohne Option `saveRAM` (bei der einige Hash-Tabellen ausgelagert werden) bei Aufruf von `storeBackup.pl` einen Crash mit seltsamen Fehlermeldungen gibt.

Es ist wichtig, Swapping zu vermeiden, weil der Raspberry Pi auf irgend etwas *sehr langsames* (z. B. eine sdcard) auslagert. Natürlich hängt eventuelles Swapping davon ab, welchen RAM-Ausbau Dein Raspberry Pi hat und was für Daten Du sicherst.

Wie immer ist das erst Backup langsam (*sehr langsam*), aber das nächste ist (ziemlich) schnell (natürlich abhängig von der langsamen CPU (verglichen mit heute üblicher PCs) und dem langsamen Speicher, in den geschrieben wird).

* * * * *

FAQ 9 Können `storeBackup` die Hardlinks ausgehen?

Man benötigt für jede Datei mindestens einen Inode, daher spart das Hardlinken von Null-Byte-Dateien zumindest viele Inodes. Dieses kann keine Auswirkungen haben (wenn im Dateisystem genügend statisch reserviert werden (ext)) oder tatsächlich Platz sparen, wenn das Dateisystem sie dynamisch alloziert (reiserfs, btrfs) oder wenn Du ansonsten mit einem ext-Dateisystem die statisch allozierte Anzahl von Inodes erreichst.

Einen Hardlink zu erzeugen sollte auch schneller sein als einen neuen Inode zu erzeugen.

Für `storeBackup` ist die Verwaltung von Hardlinks kein Problem. Es versucht, einen Hardlink zu erzeugen und, falls das nicht erfolgreich ist, wird eine neue Datei erzeugt und anschließend gegen diese verlinkt. Die Grenze für die Anzahl der Hardlinks zu erreichen bedeutet lediglich eine neue identische Datei zu erzeugen. Aufgrund dieses einfachen Algorithmus muss sich `storeBackup` nicht um die Anzahl der Hardlinks, die ein Dateisystem unterstützt, kümmern. Dieses Verhalten unterscheidet sich von dem typischer Unix Tools wie `cp`, `tar` oder `rsync`. Wenn Du mit ihnen ein Verzeichnis mit vielen Hardlinks auf ein Dateisystem kopierst, das nicht genügend Hardlinks unterstützt, bekommst Du eine Fehlermeldung (siehe auch die Erläuterung zum Program `linkToDirs.pl`, das Teil der `storeBackup`-Tools ist und dieses Problem auf die oben beschriebene Art und Weise umgeht).

Btrfs würde ich momentan (2014) nicht für Backups verwenden, weil ich es nicht für stabil genug halte. Aber nichtsdestotrotz habe ich einige Tests gemacht und das Verhalten bezüglich Hardlinks scheint sehr anders als das anderer Dateisysteme zu sein. Ich erreichte die Hardlink-Grenze – mit allen Dateien in einem Verzeichnis – es war aber möglich, weitere Hardlinks auf diese Dateien aus einem anderen Verzeichnis zu erzeugen. Aber wie auch immer – aufgrund von `storeBackup`s simplen Algorithmus ist es auch mit btrfs effizient.

Zusammengefasst denke ich, dass es keinen Grund gibt, Null-Byte-Dateien nicht per Hardlink zu verbinden.

11 Mitwirkende

Mein Dank gilt allen, die ihre Ideen mit mir teilen, mir Fehlerberichte sandten und geduldig genug waren, storeBackup zu dem werden zu lassen, was es heute ist.

Ich möchte hier insbesondere aufführen:

- Francesco Potorti, der mir sehr viel bei Fehlerkorrekturen in Teilen der Version 1.x half.
- Arthur Korn für viele Diskussionen und seine Unterstützung für Debian.
- Nikolaus Rath (Nikolaus at rath.org), der wesentliche Beiträge zur Version 2 machte und mich wieder für die Weiterentwicklung von storeBackup motivierte.
- W. David Shields, ViewMachine Corporation, Florida, USA, (dave at viewmachine.com), der die Dokumentation überarbeitete, über Neuerungen diskutierte und viele Fehler während der Testphase von Version 3 fand.
- Frank Brungräber für das Gegenlesen dieser Dokumentation.

Diese Auflistung soll keine Zurücksetzung all der anderen sein, die mich unterstützt haben.

12 Change Log

```
1 -----
2
3 version 1.0      2002.05.07
4 first public release
5
6 -----
7 version 1.1      2002.05.18
8 statistical output 'over all files/sec' was unclear
9 changed to:
10 over all files/sec (real time) =
11 over all files/sec (CPU time) =
12 CPU usage =
13
14 versions are now (overall checksum):
15 storeBackup.pl -V      => 1.3461
16 storeBackuppls.pl -V   => 1.2583
17 storeBackupVersions.pl -V => 1.4313
18 storeBackupRecover.pl -V => 1.4992
19
20 -----
21 version 1.2      2002.05.19
22 storeBackup.pl:
23 with option --exceptDirs you can also use wildcards
24 added option --contExceptDirsErr
25
26 storeBackupRecover.pl:
27 if you extract a directory (eg. abc) and there exists another
28 directory with a name with the same beginning (eg. abcd), this
29 one will also be extracted -> corrected
30
31 versions are now (overall checksum):
32 storeBackup.pl -V      => 1.3471
33 storeBackuppls.pl -V   => 1.2583
34 storeBackupVersions.pl -V => 1.4313
35 storeBackupRecover.pl -V => 1.5145
36
37 -----
38 version 1.3      2002.05.22
39 all programs:
```



```

40 the usage of the programmes with sensless list parameters
41 (like *.h) was ignored -- now an error message is produced
42
43 storeBackupVersions.pl:
44 improved performance, checks same inodes before calculating
45 md5 sums
46
47 storeBackup.pl:
48 when the time for backup was < 1 sec, a division by zero could happen
49 (thanks to Joerg Paysen for the report)
50 added --keepMinNumberAfterLastOfDay (instead of replacing --keepMinNumber)
51
52 versions are now (overall checksum):
53 storeBackup.pl -V          => 1.3491
54 storeBackuppls.pl -V       => 1.2583
55 storeBackupVersions.pl -V => 1.4483
56 storeBackupRecover.pl -V => 1.5159
57
58 -----
59 version 1.4      2002.05.27
60 all programs:
61 support recovering of hard links in the source tree of storeBackup.pl
62
63 storeBackupRecover.pl
64 fixed some little bugs introduced in version 1.2
65
66 storeBackupConvertBackup.pl
67 new program to convert old backup directories (target) to the new
68 format of .md5CheckSums[.bz2]
69 YOU HAVE TO CALL IT, IF YOU WANT TO USE VERSION 1.4 WITH OLD BACKUPS!
70
71 versions are now (overall checksum):
72 storeBackup.pl -V          => 1.3568
73 storeBackuppls.pl -V       => 1.2583
74 storeBackupVersions.pl -V => 1.4775
75 storeBackupRecover.pl -V => 1.4073
76 storeBackupConvertBackup.pl -V => 1.9776
77
78 -----
79 version 1.5      2002.05.28
80 storeBackup.pl
81 better statistics about freed/used space on disk
82
83 versions are now (overall checksum):
84 storeBackup.pl -V          => 1.3606
85 storeBackuppls.pl -V       => 1.2583
86 storeBackupVersions.pl -V => 1.4775
87 storeBackupRecover.pl -V => 1.4073
88 storeBackupConvertBackup.pl -V => 1.9776
89
90 -----
91 version 1.6      2002.06.10
92 storeBackupVersions.pl
93 added flags:
94 --showAll (same as all below)
95 --size (shows size of found files)
96 --uid (show also uid of source file)
97 --gid (show also gid of source file)
98 --mode (show also mode of source file)
99 --ctime (show also creation time of source file)
100 --mtime (show also modify time of source file)
101 storeBackup.pl

```

```

102 added weekday to INFO output in log file when deleting old dir
103 via parameter --keepOnlyLastOfDay
104 ROADMAP is actualized
105
106 versions are now (overall checksum):
107 storeBackup.pl -V => 1.3617
108 storeBackuppls.pl -V => 1.2583
109 storeBackupVersions.pl -V => 1.4401
110 storeBackupRecover.pl -V => 1.4073
111 storeBackupConvertBackup.pl -V => 1.9776
112
113 -----
114 version 1.7      2002.07.2
115 storeBackup.pl
116 added flag --ignoreReadError
117 added flags --file, --generate, --print: you can now use a
118 configuration file instead of putting all in command line options
119
120 versions are now (overall checksum):
121 storeBackup.pl -V => 1.2871
122 storeBackuppls.pl -V => 1.2972
123 storeBackupVersions.pl -V => 1.3795
124 storeBackupRecover.pl -V => 1.3280
125 storeBackupConvertBackup.pl -V => 2.0308
126
127 -----
128 version 1.8      2002.08.17
129 storeBackupConvertBackup.pl
130 updated program to convert old backup directories (target) to the new
131 format of .md5CheckSums[.bz2] and .md5CheckSums.info
132 YOU HAVE TO CALL IT, IF YOU WANT TO USE VERSION 1.7 WITH OLD BACKUPS!
133 see file bin/_ATTENTION_ for detailed information
134
135 storeBackuppls.pl
136 added option -v for verbose information
137
138 storeBackup.pl
139 - correction of minor errors
140 - added list parameter(s) otherBackupSeries
141 allows you to hard link to older trees from the same backup
142 allows you to hard link to backup trees of another backup series
143 This gives you the possibility to share data via hard link between
144 independent backups. See README file for more information (search
145 for 'otherBackupSeries').
146
147 storeBackupVersions.pl + storeBackupRecover.pl
148 - compatible with new file format
149
150 -----
151 version 1.8.1    2002.08.19
152 Error fixing:
153 storeBackup.pl
154 - didn't build dbm(filename) correctly when first backup with
155 otherBackupSeries
156 - pattern for recognizing of relative part of backup path did not
157 work with some strange path names, pattern replaced with substr
158 and length
159 - if the directory to backup was empty, then no .md5CheckSum.bz2
160 was created
161
162 -----
163 version 1.9      2002.08.26

```

```

164 storeBackup.pl
165 - new option --chmodMD5File
166 - total internal replacement for handling --onlyMD5Check
167 is now handled in ::buildDBMs -> nearly as fast as without
168 --onlyMD5Check
169 - new option --printDepth
170 - options --onlyMD5Check and --onlyMD5CheckOn are now only needed
171 if hard linking with other backups (see otherBackupSeries)
172
173 -----
174 version 1.9.1      2002.08.31
175 storeBackup.pl
176 - performance improvement when copying small files (< 100KB)
177 - error fix: --onlyMD5Check was not as fast as described in v1.9
178 du to an error when making the package (but fortunately the
179 correct version was in my backup)
180
181 versions are now (overall checksum):
182 storeBackup.pl -V          => 1.3138
183 storeBackuppls.pl -V       => 1.2626
184 storeBackupVersions.pl -V  => 1.4091
185 storeBackupRecover.pl -V   => 1.3454
186 storeBackupConvertBackup.pl -V => 2.0844
187
188 -----
189 version 1.10      2002.10.20
190 storeBackup.pl
191 - options --onlyMD5Check and --onlyMD5CheckOn are now obsolete
192 storeBackup decides itself, if the functionality is needed
193 - you do not have to worry when using 'otherBackupSeries' if it's not
194 yet ready. this is recognized automatically
195 - added options --withUserGroupStat --userGroupStatFile
196
197 versions are now (overall checksum):
198 storeBackup.pl -V          => 1.3325
199 storeBackuppls.pl -V       => 1.2966
200 storeBackupVersions.pl -V  => 1.4295
201 storeBackupRecover.pl -V   => 1.3709
202 storeBackupConvertBackup.pl -V => 2.0844
203
204 -----
205 version 1.10.1    2002.10.27
206 storeBackup.pl + storeBackupRecover.pl
207 - replaced syscall lchown with fork-exec chown
208 because of error messages with perl 5.8 (SuSE 8.1)
209
210 versions are now (overall checksum):
211 storeBackup.pl -V          => 1.3334
212 storeBackuppls.pl -V       => 1.2966
213 storeBackupVersions.pl -V  => 1.4295
214 storeBackupRecover.pl -V   => 1.3722
215 storeBackupConvertBackup.pl -V => 2.0844
216
217 -----
218 version 1.11      2003.03.05
219 storeBackup.pl
220 - --exceptSuffix: removed '.bmp', added '.pgp'
221 - changed default of parameter --logFile
222 - new parameters:
223 --plusLogStdout, --saveLogs, --compressWith,
224 --logInBackupDir, --compressLogInBackupDir,
225 --logInBackupDirFileName

```

```

226 - if called with parameter -f ... --print then
227 evaluation of wildcards is performed
228 - correction of litte faults
229
230 versions are now (overall checksum):
231 storeBackup.pl -V                => 1.3435
232 storeBackuppls.pl -V             => 1.3152
233 storeBackupVersions.pl -V        => 1.4406
234 storeBackupRecover.pl -V         => 1.3862
235 storeBackupConvertBackup.pl -V => 2.0844
236
237 -----
238 version 1.12          2003.04.16
239 storeBackup.pl
240 - exception list was not taken into account when checking
241 collisions from options of -t and -s
242 - added parameter --copyBWLimit (uses rsync for copying)
243 - in some cases internal linkage of duplicated files did not
244 working
245 - added parameter --postcommand
246 - added statistical output for used length of queues
247
248 versions are now (overall checksum):
249 storeBackup.pl -V                => 1.3537
250 storeBackuppls.pl -V             => 1.3322
251 storeBackupVersions.pl -V        => 1.4518
252 storeBackupRecover.pl -V         => 1.4001
253 storeBackupConvertBackup.pl -V => 2.0844
254 llt -V                          => 1.4294
255 multitail.pl -V                 => 1.4555
256
257 -----
258 version 1.12.1        2003.05.01
259 storeBackup.pl
260 - When copying files < 100 KB into the backup, owner and permissions
261 were not set correctly. When hard linking in the next backup, this
262 was corrected. -> Error fixed
263 - When problems with forking cp or the compression program occurred,
264 this was not handled correctly.
265
266 versions are now (overall checksum):
267 storeBackup.pl -V                => 1.3545
268 storeBackuppls.pl -V             => 1.3322
269 storeBackupVersions.pl -V        => 1.4518
270 storeBackupRecover.pl -V         => 1.4001
271 storeBackupConvertBackup.pl -V => 2.0844
272 llt -V                          => 1.4294
273 multitail.pl -V                 => 1.4555
274
275 -----
276 version 1.12.2        2003.05.18
277 storeBackup.pl
278 - When copying files < 100 KB into the backup, sometimes the
279 storeBackup internal scheduler slows down the backup -> fixed
280 - Files with size zero where not handled correctly -> fixed
281 - Some complicated if cases where not covered -> fixed
282 - better internal documentation
283 - granularity of the internal scheduler is now finer, prog should be
284 about 5% faster
285 - added /etc/cron.daily/storebackup from Arthur Korn for Debian users
286
287 versions are now (overall checksum):

```

```

288 storeBackup.pl -V          => 1.3554
289 storeBackuppls.pl -V       => 1.3322
290 storeBackupVersions.pl -V   => 1.4518
291 storeBackupRecover.pl -V    => 1.4001
292 storeBackupConvertBackup.pl -V => 2.0844
293 llt -V                     => 1.4294
294 multitail.pl -V            => 1.4555
295
296 -----
297 version 1.13          2003.07.28
298 - BSD is now supported
299 storeBackup.pl
300 - Many new options for managing old backups. New/changed parameters:
301 --noDelete changed to --doNotDelete
302 --keepAll can now handle the 'archive flag'
303 --keepWeekDay can now handle the 'archive flag'
304 --keepFirstOfYear is new
305 --keepLastOfYear is new
306 --keepFirstOfMonth is new
307 --keepLastOfMonth is new
308 --firstDayOfWeek is new
309 --keepFirstOfWeek is new
310 --keepLastOfWeek is new
311 --keepOnlyLastOfDay changed to --keepDuplicate
312 --keepMaxNumber is new
313 --keepMinNumberAfterLastOfDay has gone
314 - Correct error message if you do not have permission to read a
315 file (not being root).
316 - Option --exceptDirs only worked correct when storeBackup was
317 started in the source directory (sourceDir)
318 storeBackupDel.pl
319 - new programm to only delete old backups with the flags described
320 above at storeBackup.pl
321
322 versions are now (overall checksum):
323 storeBackup.pl -V          => 1.3664
324 storeBackuppls.pl -V       => 1.3509
325 storeBackupVersions.pl -V   => 1.3765
326 storeBackupRecover.pl -V    => 1.4154
327 storeBackupConvertBackup.pl -V => 2.0844
328 storeBackupDel.pl -V        => 1.3606
329 llt -V                     => 1.2222
330 multitail.pl -V            => 1.4555
331
332 -----
333 version 1.14          2003.08.26
334 storeBackup.pl
335 - most parts of the statistical output were twice when one ore more
336 old backups were deleted
337 - now runs on AIX
338 - checks, if targetDir has write permissions (better error message)
339 - replace statistic message:
340 additional used space
341 with
342 add. used space in files
343 storeBackupDel.pl
344 - can use the config file of storeBackup.pl to operate
345 storeBackuppls.pl
346 - can use the config file of storeBackup.pl to show analysis of
347 livetime of old backups
348
349 versions are now (overall checksum):

```

```

350 storeBackup.pl -V                => 1.2993
351 storeBackuppls.pl -V            => 1.2102
352 storeBackupVersions.pl -V       => 1.2949
353 storeBackupRecover.pl -V        => 1.3134
354 storeBackupConvertBackup.pl -V => 2.0844
355 storeBackupDel.pl -V            => 1.2795
356 llt -V                          => 1.2222
357 multitail.pl -V                 => 1.4555
358
359 -----
360 version 1.14.1      2003.10.25
361 storeBackup.pl (fixed)
362 - in some cases, setuid and setgid were not stored in the backup
363 - depending on the kernel version, permissions in the backup were
364 not set correctly
365 storeBackupRecover.pl (fixed)
366 - depending on the kernel version, permissions in the backup were
367 not set correctly
368
369 versions are now (overall checksum):
370 storeBackup.pl -V                => 1.3001
371 storeBackuppls.pl -V            => 1.2102
372 storeBackupVersions.pl -V       => 1.2949
373 storeBackupRecover.pl -V        => 1.3147
374 storeBackupConvertBackup.pl -V => 2.0844
375 storeBackupDel.pl -V            => 1.2795
376 llt -V                          => 1.2222
377 multitail.pl -V                 => 1.4555
378
379 -----
380 version 1.15      2004.02.06
381 storeBackup.pl
382 - otherBackupSeries now understands 'from-to' and 'all'
383 --includeDirs is new
384 --exceptPattern is new
385 --includePattern is new
386 --resetAtime (in the source directory) is new
387 - sets atime and mtime in the backup to the same values as in
388 the source directory
389
390 deleting of old backups (storeBackup.pl, storeBackuppls.pl,
391 storeBackupDel.pl)
392 - fixed bug with options --keepMinNumber and --keepMaxNumber
393 - set default value of --keepDuplicate to 7d
394 - result of checking old log files is now write to logfile
395 inside of backup (if wanted)
396
397 storeBackupRecover.pl
398 - restores atime and mtime when restoring backups
399
400 llt
401 - output now in format yyyy.mm.dd, no longer in german format
402
403 configuration file syntax
404 - allows now the use of single quotes
405
406 storeBackupMount.pl
407 - pings server, mounts file systems, calls storeBackup and
408 umounts filesystems
409
410 versions are now (overall checksum):
411 (these values have changed dramatically because I switched from cvs to svn)

```

```

412 storeBackup.pl -V                => 157.8243
413 storeBackuppls.pl -V            => 96.8069
414 storeBackupVersions.pl -V       => 138.2092
415 storeBackupRecover.pl -V        => 171.4032
416 storeBackupConvertBackup.pl -V => 178.6868
417 storeBackupDel.pl -V            => 153.4117
418 storeBackupMount.pl -V          => 129.1638
419 llt -V                          => 103.7589
420 multitail.pl -V                 => 62.3245
421
422 -----
423 version 1.15.1    2004.02.08
424 storeBackup.pl
425 - fixed a bug when reading the config file
426 (affecting exceptPattern, includePattern)
427 - fixed a bug when using 'sourceDir = /' and exceptPattern
428 or includePattern
429
430 versions are now (overall checksum):
431 storeBackup.pl -V                => 183.5295
432 storeBackuppls.pl -V            => 143.9218
433 storeBackupVersions.pl -V       => 171.1896
434 storeBackupRecover.pl -V        => 212.6288
435 storeBackupConvertBackup.pl -V => 178.6868
436 storeBackupDel.pl -V            => 183.3940
437 storeBackupMount.pl -V          => 170.2637
438 llt -V                          => 104.0773
439 multitail.pl -V                 => 116.9386
440
441 -----
442 version 1.16      2004.02.25
443 storeBackup.pl
444 - added parameter --exceptTypes
445 - store data in dbm files with pack / unpack
446 - better handling if maximum number of hard links is exceeded
447 - precommand and postcommand now understand single quotes nested
448 in double quotes in the commandline (like ...Pattern)
449 - storeBackup didn't store the uncompress command correctly since
450 version 1.15. This means, that storeBackupRecover could not
451 restore the original version. This is because of the missing
452 option '-d' in file .md5Checksums.info. Wrong version:
453 uncompress=bzip2
454 but must be
455 uncompress=bzip2 -d
456 Change this line with an editor or use the script correct.sh
457
458 storeBackupRecover.pl
459 - storeBackupConvertBackup.pl had a bug, so that storeBackupRecover
460 did not work any more. storeBackupRecover is now able to
461 handle converted backups (again).
462
463 versions are now (overall checksum):
464 storeBackup.pl -V                => 183.9252
465 storeBackuppls.pl -V            => 144.0733
466 storeBackupVersions.pl -V       => 171.5950
467 storeBackupRecover.pl -V        => 213.5498
468 storeBackupConvertBackup.pl -V => 178.6868
469 storeBackupDel.pl -V            => 183.7625
470 storeBackupMount.pl -V          => 170.9166
471 llt -V                          => 104.0773
472 multitail.pl -V                 => 116.9386
473

```

```

474 -----
475 version 1.16.1    2004.03.07
476 storeBackup.pl
477 - better explanations in the configuration file
478 and for command line options
479 - better error messages
480 - option --print did not work for some values
481 - fixed a bug in the module for reading the
482 configuration file with keepWeekday
483 - when printing to a log file and to stdout
484 simultaneously, a possible error message with exit
485 is now also printed to stdout
486 - option verbose now has the same effekt as debug=1
487
488 versions are now (overall checksum):
489 storeBackup.pl -V          => 184.4928
490 storeBackuppls.pl -V       => 144.6597
491 storeBackupVersions.pl -V  => 172.0055
492 storeBackupRecover.pl -V   => 214.0630
493 storeBackupConvertBackup.pl -V => 178.6868
494 storeBackupDel.pl -V       => 184.5203
495 storeBackupMount.pl -V     => 171.2973
496 llt -V                    => 104.0773
497 multitail.pl -V           => 117.4461
498
499 -----
500 version 1.16.2    2004.04.04
501 storeBackup.pl
502 - exit status is now correct (0) when running successfully
503 - option --verbose now prints some additionally verbose messages
504 it is not similar any more to --debug 1
505 - the log file written into the backup now contains the
506 "delete old backupevaluation"
507 - unsupported file type didn't generate an error message
508 instead, the blew up the backup -> corrected
509 - integer overrun in the statistical output when saving large
510 amounts of data is corrected
511 storeBackup_du.pl added to the package
512 versions are now (overall checksum):
513 storeBackup.pl -V          => 184.6565
514 storeBackuppls.pl -V       => 144.2247
515 storeBackupVersions.pl -V  => 172.0004
516 storeBackupRecover.pl -V   => 214.0566
517 storeBackupConvertBackup.pl -V => 178.6868
518 storeBackupDel.pl -V       => 184.5157
519 storeBackupMount.pl -V     => 171.2909
520 llt -V                    => 104.0773
521 multitail.pl -V           => 116.9386
522
523 -----
524 version 1.17      2004.09.04
525 storeBackup.pl
526 - reduced size of temporary berkeley db files
527 this results in better caching (and therefore better performance
528 for backups with many files)
529 - also print size of the berkely db files into the statistical output
530 - new option --unlockBeforeDel
531 - various little bug fixes (corrected comments and print outputs)
532 storeBackupMount.pl
533 - better exit status, distinguishes between errors in
534 storeBackup und storeBackupMount
535 versions are now (overall checksum):

```



```

536 storeBackup.pl -V => 184.9850
537 storeBackuppls.pl -V => 144.6790
538 storeBackupVersions.pl -V => 173.1101
539 storeBackupRecover.pl -V => 214.4541
540 storeBackupConvertBackup.pl -V => 178.6868
541 storeBackupDel.pl -V => 184.8048
542 storeBackupMount.pl -V => 171.8483
543 storeBackup_du.pl -V => 73.0682
544 llt -V => 104.0773
545 multitail.pl -V => 118.2667
546
547 -----
548 version 1.18      2004.06.03
549 storeBackup.pl
550 - minor corrections to statistical output
551 - fixed a bug with options --includePattern and --exceptPattern:
552 There had to be brackets around a logical expression.
553 storeBackupRecover.pl
554 - restoring of directories with a round bracket in the name did not
555 work sometimes, fixed
556 versions are now (overall checksum):
557 storeBackup.pl -V => 185.0688
558 storeBackuppls.pl -V => 144.6790
559 storeBackupVersions.pl -V => 173.1101
560 storeBackupRecover.pl -V => 215.1446
561 storeBackupConvertBackup.pl -V => 178.6868
562 storeBackupDel.pl -V => 184.8048
563 storeBackupMount.pl -V => 171.8483
564 storeBackup_du.pl -V => 73.0682
565 llt -V => 104.0773
566 multitail.pl -V => 118.2667
567
568 -----
569 version 1.18.1    2004.06.08
570 storeBackup.pl
571 - fixed a silly bug which occurred one did not use option progressReport
572 versions are now (overall checksum):
573 storeBackup.pl -V => 185.1527
574 storeBackuppls.pl -V => 144.6790
575 storeBackupVersions.pl -V => 173.1101
576 storeBackupRecover.pl -V => 215.1446
577 storeBackupConvertBackup.pl -V => 178.6868
578 storeBackupDel.pl -V => 184.8048
579 storeBackupMount.pl -V => 171.8483
580 storeBackup_du.pl -V => 73.0682
581 llt -V => 104.0773
582 multitail.pl -V => 118.2667
583
584 -----
585 version 1.18.2    2004.06.26
586 storeBackup.pl
587 - storeBackup calculated too much md5 sums, corrected
588 - storeBackup had a dependency with perl versions >= 5.8,
589 now it does not depend on this new version any more
590 versions are now (overall checksum):
591 storeBackup.pl -V => 185.4812
592 storeBackuppls.pl -V => 145.1333
593 storeBackupVersions.pl -V => 173.1101
594 storeBackupRecover.pl -V => 215.5421
595 storeBackupConvertBackup.pl -V => 178.6868
596 storeBackupDel.pl -V => 185.0938
597 storeBackupMount.pl -V => 171.8483

```

```

598 storeBackup_du.pl -V          => 73.0682
599 llt -V                      => 104.0773
600 multitail.pl -V              => 118.2667
601
602 -----
603 version 1.18.3    2004.07.06
604 storeBackup.pl
605 - much better performance when used with exceptPattern or
606 includePattern
607 storeBackuppls.pl
608 - if used with option -f, default is to read the the location
609 of the backup from the configuration file
610 this default can be overwritten (if you have different mount
611 points)
612 versions are now (overall checksum):
613 storeBackup.pl -V              => 185.8650
614 storeBackuppls.pl -V           => 173.5429
615 storeBackupVersions.pl -V      => 173.9271
616 storeBackupRecover.pl -V       => 216.1658
617 storeBackupConvertBackup.pl -V => 178.6868
618 storeBackupDel.pl -V           => 185.5475
619 storeBackupMount.pl -V        => 172.4720
620 storeBackup_du.pl -V          => 73.0682
621 llt -V                        => 104.0773
622 multitail.pl -V               => 118.2667
623
624 -----
625 version 1.18.4    2004.07.11
626 storeBackup.pl
627 - (much) better performance because of reducing the number of
628 md5sum calls when using otherBackupSeries
629 - the very first backup of a backup series did not hard link
630 to another backup series defined with otherBackupSeries
631 - some temporary files were not deleted
632 versions are now (overall checksum):
633 storeBackup.pl -V              => 186.1958
634 storeBackuppls.pl -V           => 173.9472
635 storeBackupVersions.pl -V      => 174.1391
636 storeBackupRecover.pl -V       => 216.4308
637 storeBackupConvertBackup.pl -V => 178.6868
638 storeBackupDel.pl -V           => 185.7402
639 storeBackupMount.pl -V        => 172.4720
640 storeBackup_du.pl -V          => 73.0682
641 llt -V                        => 104.0773
642 multitail.pl -V               => 118.2667
643
644 -----
645 version 1.19      2005.08.05
646 storeBackup.pl
647 - in some rare cases filenames were stored with a leading slash
648 in .md5CheckSum. I could not be simulated by me. But the bug
649 should be fixed.
650 - some fixes in handling of directory paths
651 - uid and gid were not set correctly for symbolic links in the
652 backups (in the files, not the description of the files)
653 - formatting of file sizes with human readable number (eg. 3.5k)
654 didn't work properly in all cases
655 - check for symbolic links before opening temporary files
656 - set permissions of backup root directory to 0755
657 (independent of umask)
658 storeBackupRecover.pl
659 - could not restore directory '.' with option -r

```

```

660 - uid and gid were not set correctly for symbolic links when
661 restoring, instead they were changed in the file where the
662 symlink pointed to
663 versions are now (overall checksum):
664 storeBackup.pl -V => 186.1958
665 storeBackup.pls.pl -V => 173.9472
666 storeBackupVersions.pl -V => 174.1391
667 storeBackupRecover.pl -V => 216.4308
668 storeBackupConvertBackup.pl -V => 178.6868
669 storeBackupDel.pl -V => 185.7402
670 storeBackupMount.pl -V => 172.4720
671 storeBackup_du.pl -V => 73.0682
672 llt -V => 107.5789
673 multitail.pl -V => 118.2667
674
675
676 - changed max args for GNU/Linux to 64*1024 because of possible
677 problems when using multibyte character sets
678
679 -----
680 version 1.19.1 2005.10.08
681 storeBackup.pl
682 - reduced the lenght of the command line because of problems
683 with dual byte characters
684 - all temporary file names now have a 64 bit random number
685 all (randomly generated) file names are checked for existence
686 before used
687
688 -----
689 version 1.19.2 2005.11.13
690 storeBackup.pl
691 - when saving with --sourceDir / without using --includeDirs then
692 storeBackup calculated useless md5sums
693
694 -----
695 -----
696 version 2.0 2008.11.09
697 all programs:
698 - changed licence to gpl-3
699 - backup format is compatible to version 1.19,
700 options *have changed*
701 - fixed several bugs
702 - introduction of lateLinks (this is the major change)
703 - new options lateLinks, lateCompress
704 - new module for interpreting command line arguments and
705 configuration file: a combination is now possible
706 - better support for files > 2GB on 64 bit operating systems
707 storeBackup.pl, storeBackupDel.pl:
708 - arguments in command line can overwrite configuration file
709 - new option keepRelative
710 - new option deleteNotFinishedDirs
711 storeBackup.pl:
712 - rewrite of core engine
713 - changed algorithm for linking with old backups
714 - directories specified with exceptDirs will now be created
715 as empty directories
716 - new option ignorePerms
717 - new option cpIsGnu (support for special files)
718 - new option saveRAM (default is now to hold temp. DBs in RAM)
719 - removal of option exceptDirsSep
720 - renamed option withTime to suppressTime
721 - renamed option compressMD5File to doNotCompressMD5File

```

```

722 - exceptPattern has gone, now there is exceptRule (different syntax)
723 - includePattern has gone, now there is includeRule (different syntax)
724 - new option writeExcludeLog
725 - setting time on (absolute) symbolic link resulted in setting time
726 in the original file -> corrected
727 storeBackupUpdateBackup.pl
728 - new program
729 - sets links asynchronously after running storeBackup with lateLinks
730 storeBackupSearch.pl
731 - new program
732 - allows searching in backups with a free definition depending on
733 filename, size, uid, gid, ctime, mtime and file type
734
735 -----
736 version 2.0.1 2008.12.14
737 storeBackupDel.pl:
738 - option keepLastOfWeek wasn't recognized when set in
739 configuration file
740 storeBackup.pl:
741 - corrected wrong addition for statistical output of
742 option progressReport
743
744 -----
745 -----
746 version 3.0 2009.03.15
747 - support of ';' as comment sign in configuration files
748 (additionally to '#' for better readability)
749 storeBackupCheckBackup.pl
750 - new program, checks consistency of a backup
751 storeBackupDel.pl:
752 - option keepLastOfWeek wasn't recognized when set in
753 configuration file
754 storeBackup.pl:
755 - new options for saving files blocked:
756 checkBlocksSuffix
757 checkBlocksSuffixMinSize
758 checkBlocksSuffixBS
759 checkBlocksCompr
760 - new options for saving files blocked:
761 checkBlocksRule (0-4)
762 checkBlocksBS (0-4)
763 checkBlocksCompr (0-4)
764 checkBlocksRead (0-4)
765 - new options for saving devices blocked:
766 checkDevices (0-4)
767 checkDevicesDir (0-4)
768 checkDevicesBS (0-4)
769 checkDevicesCompr (0-4)
770 - new option to hard link symbolic links:
771 linkSymlinks
772 - new option for defining which files to compress:
773 comprRule
774
775 -----
776 version 3.1 2009.05.24
777 storeBackup.pl
778 - storeBackup did not backup sockets, now it does
779 - for new files, the md5 sum is now calculated before *and*
780 after copying / compressing for safety reasons. The file could
781 have been changed during that time. So the md5 sum would not
782 match the real one. A file with the firstly calculated
783 md5 sum later could be hard linked to the changed file which

```

```

784 means there is no backup of its content.
785 If both md5 sums do not match, an warning is generated and
786 the md5 sum is set to ggggg... which is a not possible value.
787 This problem does not exist for blocked files in v3.0.
788 - improved statistic at the end of a run (sum of warnings
789 and errors)
790 - added options checkBlocksParallel and checkDevicesParallel
791 - added option linkToRecent
792 - name clashes because of compressing files (eg. add .bz2)
793 were not handeld corretly - bug was introduced in 3.0
794 corrected
795 - when making a backup with source=/ while not using
796 includeDirs then the md5 sums of all files were calculated
797 also after the first backup
798 - corrected some issues with the statistical output
799 - option copyBWLimit is now deprecated because
800 - of internal performance optimization
801 - it is useless
802 - new option suppressWarning
803 storeBackupUpdateBackup.pl
804 - if sourceDir=/, for the very first backup with option
805 lateLinks an empty 'linkFrom' file was generated which lead
806 to (useless) error messages. corrected.
807 storeBackupCheckBackup.pl
808 - now also checks if files in the backup are not listed
809 in .md5Checksum
810 storeBackupRecover.pl
811 - the directories in the path to the restored files / directories
812 were not set the original permissions, corrected
813 llt
814 - added option --epoch to calculate human readable dates from
815 epoch based dates
816 man
817 - man pages for all programs (Thanks to Ryan Niebur)
818 all programs
819 - solved issues with single quotes in path and filenames
820
821 -----
822 version 3.2 2009.07.18
823 storeBackup.pl
824 - new option --highLatency, useful on high latency lines
825 - corrected some typos in print statements to log files
826 - now also checks for size of files if files with two equal
827 md5 sums are detected
828 - fixed a bug when using *block* options. storeBackup.pl stopped
829 with an error message when blocked file was existing with same
830 path, filename, contents and times in another series but did
831 not exist in the own series of that backup.
832 - plus some very minor enhancements
833 all programs:
834 - if an option in a configuration file is set to nothing, the
835 default value (if exists) is used
836
837 -----
838 version 3.2.1 2012.02.12
839 storeBackup.pl
840 - replaced File::Copy by own function, because File::Copy did not
841 handle strange filenames (eg. with
842 n) without warnings
843 - read .md5Checksum.info with algorithm for configuration file
844 - changed comments for some options
845 - new parameter fileNameWithLineFeed to option suppressWarning:

```

```

845 suppresses warning if a filename contains a line feed
846 - write logInBackupDirFileName into .md5Checksum.info so it can
847 be identified by storeBackupCheckBackup.pl
848 - deletion of old backups is now done before postcommand
849 - backup of a blocked file (or device) didn't store all md5sums
850 for all blocks in the local .md5Checksum file if two or more
851 block in one blocked file were identical
852 this means it is possible to restore the data with cat or bzip,
853 but *not* with storeBackupRecover.pl !
854 - statistics for storage of blocked files corrected
855 - avoided some (useless) perl warnings about undef'ed variables
856 - avoided some (useless) perl warnings about gotos (happens in
857 new perl versions)
858 - if a file cannot be hard linked, storeBackup.pl makes a new
859 copy of that file. The warning about that fact was shifted to
860 debug output because it confused some users
861 - corrected some confusing code about combinations of compression,
862 lateCompression and lateLinks
863 - solved several possible timing issues (reading of tmp-result files
864 - masking for file names with
n was missing when writing into
865 lateLinks command file
866 - avoid possibility of division by zero when calculating time
867 for run (percentage) in statistics
868 - corrected calculation of 'sum of target all' in statistics
869 - directories with
n in their name didn't get right time stamps
870 in the backup; corrected
871 - permissions on directories with
n in their names were not set
872 correctly
873 storeBackups.pl
874 - in storeBackups.pl option keepLastOfWeek, backupDir and series
875 was ignored in the configuration file
876 - option -v didn't work properly
877 - workaround for timing issue when reading value for inodeBackup
878 storeBackupMount.pl
879 - corrected filtering of output from mount command
880 - added 'rw', 'ro' feature to overwrite read only or read write
881 from /etc/fstab
882 storeBackupCheckBackup.pl
883 - read .md5Checksum.info with algorithm for configuration file
884 - add option includeRenamedBackups
885 - changed option -b to -c (for compatibility to storeBackupRecover.pl
886 - many error corrections (mostly written new)
887 storeBackupRecover.pl
888 - read .md5Checksum.info with algorithm for configuration file
889 - backup of a directory / file starting with '.' didn't work
890 - recovery of blocked files did not work in special cases
891 (depending on size of the blocks and compression flag)
892 - permissions on directories were not restored because if wrong
893 order - they are now set after restoring all files
894 - optimized performance (bigger block size for restoring blocked files)
895 - mtime of restored files was not set to original values because
896 of wrong order of setting permissions (corrected)
897 storeBackupUpdateBackup.pl
898 - replaced File::Copy by own function, because File::Copy did not
899 handle strange filenames (eg. with
n) without warnings
900 - read .md5Checksum.info with algorithm for configuration file
901 - corrected line number when reporting problems with command file
902 (.storeBackupLinks/linkFile.bz2)

```

```

903 - directories didn't get right time stamps when restoring; corrected
904 storeBackupVersions.pl
905 - read .md5Checksum.info with algorithm for configuration file
906
907 -----
908 version 3.3 2012.09
909 general
910 - command line option --unset now also works with list parameters
911 set in configuration files
912 (you can use eg. --unset otherBackupSeries with storeBackup.pl)
913 storeBackup.pl
914 - when saving blocked files or devices with a block size smaller
915 than 1M, then always bzip2 is used as compression algorithm -
916 doesn't matter if you eg. had chosen gzip2. In the backup, the
917 suffix was eg. .gz, but compression algorithm was bzip2.
918 storeBackupRecover cannot restore these backups correctly!
919 Please restore with eg. zcat manually
920 - added rule-function COMPRESS_CHECK
921 - changed option checkDevicesCompr<n> from switch to option with
922 parameter. Possible values are yes, no, check
923 - added option comprSuffix (now there exists a white list and a
924 black list to decide if a file should be compressed or not;
925 the rest of the files is rated by COMPRESS_CHECK)
926 - added option checkBlocksParallel (similar functionality
927 as eg. checkBlocksParallel0)
928 - use DB_File now done in eval. This means, that there is now
929 error message any more if this extension is not available
930 -> should solve problems with several NAS boxes
931 - ignore option 'mergeBackupDir' used by new program
932 storeBackupMergeIsolatedBackup.pl
933 - added statistical output 'COMPR_CHECK' for blocked files
934 - added keys to option 'suppressWarning':
935 use_DB_File, use_IOCompressBzip2
936 - option 'ignoreReadError' didn't work - read errors on
937 directories always were shown as WARNING only; fixed
938 - add some default suffixes to exceptSuffix
939 storeBackup.pl + storeBackupCheckBackup.pl
940 - storeBackup.pl didn't store correct pathname in .md5Checksum when
941 saving blocked devices, therefore storeBackupCheckBackup.pl couldn't
942 check those files
943 - storeBackupCheckBackup.pl needed small enhancement to be able
944 to read correct pathname generated from storeBackup.pl for
945 blocked devices
946 storeBackup.pl + storeBackupUpdateBackup.pl
947 - fixed bug: backup with block + lateLinks; 1st backup complete;
948 2nd backup with *no* changes to blocked file; 3rd backup with
949 changes to blocked file (all without UpdateBackup between 1st,
950 2nd and 3rd run) -> in 3rd run no blocks were linked to
951 existing one
952 - added option --autorepair (-a) to storeBackup.pl
953 - fixed bug: storeBackupUpdateBackup.pl now can handle combination
954 of lateLinks and not finished backups
955 storeBackup.pl + storeBackupDel.pl (+ all others reading config files)
956 - can now read compressed configuration files
957 (recognizing suffix .bz2 and .gz)
958 storeBackupUpdateBackup.pl
959 - added support for replication, new options:
960 --copyBackupOnly, --dontCopyBackup, --archiveDurationCopyStation,
961 --dontDelInCopyStation, --genBackupBaseTreeConf,
962 --genCopyStationConf
963 storeBackupMount.pl
964 - Debian (and Ubuntu) changes all executables to a name without

```

```

965 the suffix '.pl'. storeBackupMount.pl now looks for
966 storeBackup.pl _and_ storeBackup
967 storeBackupCheckBackup.pl
968 - corrected help text / man page
969 - added log file management
970 storeBackupDel.pl
971 - added 'BEGIN' and 'END' to log files for better support
972 through NAGIOS plugin
973 - --plusLogStdout didn't work
974 storeBackupSetupIsolatedMode.pl
975 - new program
976 storeBackupMergeIsolatedBackup.pl
977 - new program
978 storeBackupReplicationWizard.pl
979 - new program
980 linkToDirs.pl
981 - new program
982 storeBackupCheckSource.pl
983 - new program
984
985 -----
986 version 3.3.1 2013.04
987 linkToDirs.pl
988 - added option --saveRAM and --tmpdir
989 - option --ignoreErrors
990
991 storeBackup.pl
992 - removed generation and reading of file backupDir/.md5BlockChecksum
993 This redundant information is not necessary any more
994 - blocked files: permissions and owner/group were not set to root (if
995 backup ran by root) and not to the real owner/group/permissions
996 - option --saveLogs was always switched on
997 - Storing of blocked files didn't work if an existing block had to
998 be hard linked when the number of possible hard links was reached.
999 (In reality, this happened esp. with big sparse files.) corrected
1000 - added parameter noBackupForPeriod to option suppressWarning
1001 (thanks to Oliver Okrongli)
1002 - added option --checkCompr / -C (command line only)
1003 - didn't work when path to storeBackup.pl contained a blank
1004 (this bug could have been present in other programs also -
1005 corrected in lib)
1006
1007 storeBackupCheckBackup.pl
1008 - also check md5 sum entries in case of already checked files
1009 which are there for hard links only
1010 - added enhanced logging like in storeBackup.pl
1011 - added option --wrongFileTables
1012 - at the end, more errors than happen where summarizing
1013 - added option --lastOfEachSeries
1014
1015 storeBackupMount.pl
1016 - newly written. Now allows usage of more programs than
1017 storeBackup.pl only
1018
1019 storeBackupUpdateBackup.pl
1020 - better error correction (option --autorepair)
1021 - added enhanced logging like in storeBackup.pl
1022 - changed useless error message to info
1023 (repair of 'link from' reference from not replicated series)
1024 - blocked files: permissions and owner/group were not set to root (if
1025 backup ran by root) and not to the real owner/group/permissions
1026 - added missing function 'cleanup'

```



```

1027 - compression now runs natively without calling external program bzip2
1028 if possible (bzip2 used + module IO::Compress::Bzip2 available)
1029 - changed 'cp -v' to 'cp -a' when copying delta cache information for
1030 replication. This allows replication on eg. samba shares
1031
1032 storeBackupCheckSource.pl
1033 - added enhanced logging like in storeBackup.pl
1034
1035 storeBackupSetupIsolatedMode.pl
1036 - added option --explicitBackup
1037
1038 dateTools.pl
1039 - corrected wrong calculation in dateTools::sub
1040 this affected storeBackupUpdateBackup deletion of old replicas in
1041 deltaCache for intraday timeframes (not important)
1042
1043 -----
1044 version 3.4 2013.07
1045 storeBackup.pl
1046 - added rule functions MARK_DIR and MARK_DIR_REC
1047 - added options --specialTypeArchiver and --archiveTypes
1048
1049 storeBackupRecover.pl
1050 - now able to restore special files stored with option --archiveTypes
1051 - do not overwrite special files any more if --overwrite is not set
1052
1053 storeBackupCheckBackup.pl
1054 - now able to check backup when special files stored with option
1055 --archiveTypes
1056
1057 storeBackupSetupIsolatedMode.pl
1058 - when using option --configFile, you can now use an already by this
1059 program generated configuration file to copy the metadata of the last
1060 backup (like in the past) again.
1061
1062 linkToDirs.pl
1063 - if flag --ignoreErrors is set, also ignore if directories already
1064 exist
1065
1066 storeBackupRecover.pl
1067 - errors from programs (eg. bzip2, cp) writing data from backup were
1068 not evaluated
1069
1070 lib/stbuMd5Exec.pl
1071 - error messages in called compression program are now transported
1072 to log files
1073 - missing waitpid inserted
1074
1075 -----
1076 version 3.4.1 2013.09
1077 storeBackup.pl
1078 - rule functions MARK_DIR, MARK_DIR_REC now work with option saveRAM
1079 - added parameter use_MLDBM to option suppressWarning
1080 - added error message when running out of disk space by copying
1081 small files (100k)
1082 - when excluding a non-readable directory with exceptDirs no
1083 warning or error message is generated any more
1084 (before this correction it was necessary to set ignoreReadError)
1085 - lockFile is deleted if control-c was pressed
1086
1087 storeBackupCheckBackup.pl
1088 - added missing entry in file list with wrong md5 sums (option -w)

```

```

1089
1090 storeBackupRecover.pl
1091 - fixed bug: called non-existing method getSTDERR on class simpleFork
1092
1093 documentation
1094 - new chapter "internals"
1095
1096 -----
1097 version 3.4.2 2013.09
1098 storeBackup.pl
1099 - fixed bug when reading output files of external programs
1100 (heuristical bug)
1101 - option --progressReport now accepts additionally a time frame
1102
1103 storeBackupUpdateBackup.pl
1104 - option --debug now works like -d (typo)
1105
1106 storeBackupRecover.pl
1107 - new option --createSparseFiles
1108
1109 linkToDirs.pl
1110 - new options --createSparseFiles and --blockSize
1111
1112 -----
1113 version 3.4.3
1114 in library for (mostly) all programs
1115 - in Ubuntu, starting a program with sudo means $PWD is not set
1116 changed subroutine absolutePath to avoid issues
1117
1118 multiTail.pl
1119 - changed program name from multitail.pl to multiTail.pl
1120 to avoid conflicts with another program called multitail
1121 - added options --print, --color, --grep
1122
1123 storeBackup.pl
1124 - changed behavior in case of (non-critical) error messages
1125 file .storeBackupLinks/linkFrom is written even in case of errors
1126 - speedup through caching of already created directories in backup
1127 when using --lateLinks -> reduced checking if directory already
1128 exists on high latency remote line
1129 - sometimes, identical blocks in blocked files were copied instead
1130 of hard linked (problem with parallelisms)
1131
1132 linkToDirs.pl
1133 - added some error messages in case of not being able to read files
1134 (and therefore to calculate md5 sums)
1135 - option --progressReport now accepts additionally a time frame
1136 - added option --printDepth
1137
1138 -----
1139 version 3.5
1140 all storeBackup*.pl programs
1141 - depend on file .md5Checksum.Finished
1142
1143 storeBackupUpdateBackup.pl
1144 - for replication: added support for wildcards in series names
1145 and option --createNewSeries (-C)
1146 - added option --noWarningDiffSeriesInBackupCopy (-N)
1147
1148 linkToDirs.pl
1149 - /tmp (partly) was used for temp. files instead of using $TMPDIR or
1150 special option

```

```

1151 - changed the file ownership and permissions of files being pointed
1152 at by symlinks, instead of the symlink itself / corrected
1153
1154 storeBackupCheckBackup.pl
1155 - added option --tmpdir
1156
1157 storeBackupMergeIsolatedBackup.pl
1158 - added option --tmpdir
1159
1160 storeBackupSetupIsolatedMode.pl
1161 - added option --force
1162
1163 storeBackupMount.pl
1164 - added option --tmpdir
1165 - added options --suppressTime, --maxFilelen, --noOfOldFiles,
1166 --saveLogs, --compressWith
1167
1168 storeBackup.pl
1169 - /tmp (partly) was used for temp. files instead of using $TMPDIR or
1170 special option
1171 - instead series names, now wildcards are also accepted
1172 (option otherBackupSeries)
1173 - option cpIsGnu is set automatically if Linux system is recognized
1174 - ERROR message "no permissions to read ..." does not enforce an exit
1175 of storeBackup.pl any more
1176 - added option stayInFileSystem
1177
1178 storeBackupRecover.pl
1179 - /tmp (partly) was used for temp. files instead of using $TMPDIR or
1180 special option
1181 - library DB_File (better performance) is not a must any more
1182 necessary to support some NAS boxes without additional tweaks
1183
1184 storeBackupReplicationWizard.pl
1185 - /tmp (partly) was used for temp. files instead of using $TMPDIR or
1186 special option
1187
1188 storeBackupSearch.pl
1189 - added option --tmpdir
1190
1191 storeBackupUpdateBackup.pl
1192 - added option --tmpdir
1193
1194 multiTail.pl
1195 - changed option --noOldFiles to --noOfOldFiles
1196 for better compatibility with other programs
1197
1198 -----
1199 version 3.5.1
1200 storeBackup.pl
1201 - linkToRecent didn't work when used for the very first time
1202 in a series
1203 - added option suppressInfo with key readCheckSums
1204 - changed the order of execution:
1205 write backup -> sync -> write 'finished' -> write linkToRecent
1206 -> delete old backups -> start postcommand
1207 - added option '-R' as alias for '--keepRelative'
1208
1209 storeBackupMergeIsolatedBackup.pl
1210 - added option --move
1211
1212 storeBackupSearch.pl

```

```

1213 - option 'backupDir' didn't work (normally not needed)
1214
1215 lib/fileDir.pl
1216 - more detailed error messages when copying of a file does
1217 not succeed
1218
1219 lib/checkParam2.pl
1220 - overwriting settings from config file via commandline didn't
1221 work for options with parameters
1222
1223 storeBackup.pl, storeBackupUpdateBackup.pl, linkToDirs.pl
1224 - added option --maxHardLinks

```

13 License

Copyright (C)

Dr. Heinz-Josef Claes (2000-2018) [hjclaes at web.de]

Nikolaus Rath (2008) [Nikolaus at rath.org] (who made substantial contributions to version 2)

Hinweis:

<http://www.gnu.de/documents/gpl.de.html>

(Dies ist eine inoffizielle deutsche Übersetzung der GNU General Public License, die nicht von der Free Software Foundation herausgegeben wurde. Es handelt sich hierbei nicht um eine rechtsgültige Festlegung der Bedingungen für die Weitergabe von Software, die der GNU GPL unterliegt; dies leistet nur der englische Originaltext. Wir hoffen jedoch, daß diese Übersetzung deutschsprachigen Lesern helfen wird, die GNU GPL besser zu verstehen.)

* * * * *

PREAMBLE

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program—to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too. When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in

other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such

as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- (a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- (b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- (c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

- (d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- (a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- (b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- (c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- (d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- (e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure

that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- (a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- (b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- (c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- (d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- (e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- (f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that

would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.