Title:

Backup mit tar: So erstellen Sie Archive unter Linux

Description:

Vermeiden Sie Datenverlust mit regelmäßigen Sicherungen. Wir erklären Ihnen, wie Sie inkrementelle Backups mit tar erstellen.

Teaser

Regelmäßige Backups sind einer der wichtigsten Faktoren bei der IT-Sicherheit. Bei Linux-Systemen – die häufig das Rückgrat eines Webservers sind – können Sie sehr effizient Sicherungskopien mit dem Programm tar erstellen. Was kann das Archivierungsprogramm? Wir erklären Ihnen, wie tar funktioniert und was die Besonderheiten sind. Damit auch Sie künftig regelmäßig Backups mit tar erstellen können, erklären wir Ihnen Schritt für Schritt wie Sie ein Backup-Skript erzeugen.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Backup mit tar erstellen: So funktioniert die Archivierung

Das Archivierungsprogramm tar basiert auf einer alten Datensicherungsmethode, die aber auch heute noch überzeugt. Der Name tar ist eigentlich ein Akronym und steht für Tape Archiver, also die Archivierung auf Bandlaufwerken. Auch wenn zumindest Privatanwender diese heutzutage kaum noch benutzen, ist das Programm immer weiterhin das beliebteste Werkzeug zur Archivierung auf Unix-Systemen. Mit dem Packprogramm können auch regelmäßige, inkrementelle Backups eines Servers erstellt werden. Wir erklären Ihnen, wie tar funktioniert und mit welchen Befehlen die Datensicherung klappt.

\_TABLE\_OF\_CONTENT\_

## Wie funktioniert tar?

tar ist ein Programm zur Archivierung unter Linux und verwandten Systemen. Für ein solches Programm untypisch bietet tar dabei standardmäßig keine Komprimierung an. Dennoch ist das Programm sehr populär, denn sein großer Vorteil liegt darin, dass **ganze Verzeichnisse zu einer Datei zusammengeführt** werden können. Diese Technik hängt mit der Geschichte des Programms zusammen: Bei einem Bandlaufwerk werden die Daten nacheinander auf ein Magnetband übertragen. Dies erklärt die sequenzielle, lineare Speicherung beim tar-Format. Neue Dateien werden hinten an das Archiv angehängt. Eine daraus entstehende Datei wird auch Tarball (englisch für Teerkugel) genannt, da die Dateien praktisch „aneinandergeklebt“ sind.

Um dennoch eine **Komprimierung** zu erreichen wird tar häufig in Kombination mit gzip verwendet. Die beiden Programme ergänzen sich dabei perfekt: gzip kann immer nur einzelne Dateien komprimieren. Deshalb wird in der Regel zunächst tar und dann gzip (oder auch ein anderes Programm zur Kompression) eingesetzt. So entstehen schließlich .tar.gz- oder .tzip-Dateien.

### tar installieren

Unter Ubuntu sollte tar bereits vorinstalliert sein. Wenn Sie eine andere Linux- oder Unix-Distribution verwenden, installieren Sie das hilfreiche Programm mit:

*sudo apt-get install tar tar-doc*

Das Paket tar-doc ist dabei optional: Es enthält eine Dokumentation zum Archivierungsprogramm.

### tar benutzen

Wenn Sie tar benutzen möchten, verwenden Sie einfach diese Syntax:

*tar Option Datei*

Die **Optionen** von tar sind folgende:

|  |  |  |
| --- | --- | --- |
| Option | Beschreibung | Besonderheit |
| --help | Zeigt Ihnen alle Optionen an. |  |
| --version | Gibt die verwendete Version von tar aus. |  |
| -c | Erzeugt ein neues Archiv. | (*create*) |
| -d | Vergleicht Dateien im Archiv und im Dateisystem miteinander | (*diff*) |
| -f | Schreibt ein Archiv in die angegebene Datei beziehungsweise liest die Daten aus der angegebenen Datei aus. | (*file*) Diese Option müssen Sie grundsätzlich als letzte eingeben, da alle nachfolgenden Eingaben als Dateien interpretiert werden. |
| -z | Komprimiert oder dekomprimiert das Archiv direkt mit gzip. | gzip muss bereits installiert sein. |
| -Z | Komprimiert oder dekomprimiert das Archiv direkt mit compress. | compress muss bereits installiert sein. Auf Großschreibung achten. |
| -j | Komprimiert oder dekomprimiert das Archiv direkt mit bzip2. | bzip2 muss bereits installiert sein. |
| -J | Komprimiert oder dekomprimiert das Archiv direkt mit xz. | xz muss bereits installiert sein. Auf Großschreibung achten. |
| -k | Verhindert, dass Dateien beim Extrahieren aus dem Archiv bereits existierende Dateien überschreiben. |  |
| -p | Behält die Zugriffsrechte beim Extrahieren bei. |  |
| -r | Fügt eine Datei zu einem bestehenden Archiv hinzu. | (*recreate*) Die Datei wird hinten angehängt. Funktioniert nur mit einem unkomprimierten Archiv. |
| -t | Zeigt den Inhalt eines Archivs an. | (*table*) |
| -u | Hängt nur Dateien an, die jünger sind als ihre jeweiligen Versionen im Archiv. |  |
| -v | Zeigt die Schritte bei der Archivierung an. | (*verbose*) |
| -vv | Zeigt detailliertere Informationen bei der Archivierung an. | (*very verbose*) |
| -w | Jede Aktion muss bestätigt werden. |  |
| -x | Extrahiert Dateien aus dem Archiv. | (*extract*) Die Dateien bleiben im Archiv enthalten. |
| -A | Hängt die Dateien eines bestehenden Archivs an ein anderes an. | Auf Großschreibung achten. |
| -C | Gibt den Ort an, in dem Dateien extrahiert werden sollen. | Auf Großschreibung achten. |
| -M | Erstellen, anzeigen oder extrahieren eines mehrteiligen Archivs. | Auf Großschreibung achten. |
| -L | Wechselt das Medium nach einer bestimmten Dateigröße. | Die Größe wird in Kilobytes angegeben. Auf Großschreibung achten. |
| -W | Überprüft das Archiv, nachdem es geschrieben wurde. | Auf Großschreibung achten. |
| -P | Archiviert alle Dateien vom Wurzelverzeichnis aus. | Auf Großschreibung achten. |
| --exclude | Schließt Dateien oder Ordner aus. | Wird nach dem Befehl zum Erstellen mit --exclude=<Datei/Ordner> angegeben. |
| -X | Liest eine Liste mit auszuschließenden Dateien. | Verlangt eine zuvor angelegte Liste: -X <Liste>.list. Auf Großschreibung achten. |
| -g | Erstellt ein Log aller Verzeichnisse inklusive Prüfsummen. |  |

Beim Erstellen von tar-Archiven haben Sie zudem die Möglichkeit, mit einem Sternchen Wildcards anzulegen. Wenn Sie ein neues Archiv erzeugen, geben Sie immer zuerst die Optionen an, dann den Dateinamen des Archivs, das Sie erstellen wollen, und schließlich die Dateien und Ordner, die enthalten sein sollen. In dem folgenden Beispiel erstellen Sie ein Archiv (*-c*) aus zwei Text-Dateien, komprimieren es mit gzip (*-z*) und schreiben es in die Datei archiv.tar.gz (*-f*):

*tar -czf archiv.tar.gz beispiel\_1.txt beispiel\_2.txt*

Möchten Sie alle Textdateien im Verzeichnis zu einem Archiv zusammenfassen, verwenden Sie eine entsprechende Wildcard:

*tar -cf text\_archiv.tar \*.txt*

Auch komplette Verzeichnisse und deren Unterverzeichnisse können Sie zu einem Archiv zusammenfassen. Im folgenden Beispiel wird */verzeichnis1* inklusive aller Unterverzeichnisse und der enthaltenden Dateien ausschließlich des Unterverzeichnisses */verzeichnis1/unterverzeichnis\_x* archiviert:

*tar -cf archiv.tar --exclude="/verzeichnis1/unterverzeichnis\_x" /verzeichnis\_1*

Im folgenden Beispiel extrahieren (*-x*) Sie das komprimierte (*-z*) Archiv, das wir im ersten Beispiel erstellt haben, in ein anderes Verzeichnis (*-C*):

*tar -xzf archiv.tar.gz -C /home/verzeichnis1/archiv\_verzeichnis*

Um einem Archiv (das dafür unkomprimiert sein muss) eine weitere Datei hinzuzufügen, geben Sie zum Beispiel folgenden Befehl ein:

*tar -rf archiv.tar beispiel\_extra.txt*

## Wie funktioniert ein Backup mit tar?

Webmaster verwenden tar gern, um Backups zu erstellen: Dabei bleibt die Verzeichnisstruktur erhalten und der Funktionsumfang des Programms lässt zusätzlich **zahlreiche Feinabstimmungen** zu, wie an den vielen Optionen ersichtlich ist. Im Folgenden erklären wir Ihnen, wie Sie ein vollständiges Backup mit tar erzeugen und wie Sie inkrementelle Backups mit dem Programm erstellen.

### Ein einfaches Backup mit tar erstellen

Es ist sinnvoll für Ihre Sicherungsstrategie, wenn Sie ein Backup-Skript für die Archivierung Ihres Systems erstellen, statt einfach händisch Archive anzulegen. So können Sie **automatisiert mehrere Verzeichnisse archivieren**, komprimieren und auf einen externen Datenspeicher übertragen. Wichtig dafür ist, dass Sie durchgehend schreib- und leseberechtigt in den entsprechenden Verzeichnissen sind. Zunächst legen Sie in Ihrem Home-Verzeichnis ein Verzeichnis *bin* an (falls Sie ein solches noch nicht haben) und erstellen dort das Skript. Das folgende Beispielskript müssen Sie selbstverständlich an Ihre Bedürfnisse und Verzeichnisstruktur anpassen:

*#!/bin/bash*

*DATE=$(date +%Y-%m-%d-%H%M%S)*

*BACKUP\_DIR="/zielverzeichnis/backup"*

*SOURCE="$HOME/quellverzeichnis"*

*tar -cvzpf $BACKUP\_DIR/backup-$DATE.tar.gz $SOURCE*

Damit Sie verstehen, was genau dieses Skript bewirkt, erklären wir es nun Zeile für Zeile:

1. Die erste Zeile ist der sogenannte Shebang, der dem Betriebssystem mitteilt, welches Interpreter-Programm es verwenden soll. In diesem Fall wird bash verwendet.
2. Jedes Backup mit tar erhält einen Zeitstempel. Das ist wichtig, damit mehrere Backups sicher auseinandergehalten werden können. Der Variablen gibt man beispielsweise folgendes Format: Jahr-Monat-Tag-StundeMinuteSekunde, also zum Beispiel 2017-09-07-152833.
3. Hiermit bestimmen Sie das Verzeichnis, in dem das Backup erstellt wird. Das letzte Unterverzeichnis beenden Sie nicht mit „/“.
4. In dieser Zeile legen Sie fest, welche Verzeichnisse Sie in das Archiv aufnehmen möchten. Hier kann man auch mehrere Verzeichnisse aufführen, die nur von einem Leerzeichen getrennt werden: SOURCE="$HOME/quellverzeichnis1 $HOME/quellverzeichnis2 ". An dieser Stelle notieren Sie ebenfalls kein „/“ am Ende der Verzeichnisse. Achten Sie in jedem Fall darauf, ein Leerzeichen vor dem abschließenden Anführungszeichen einzufügen.
5. Die letzte Zeile des Skripts enthält schließlich den tar-Befehl:
   * *-cvzpf* erstellt ein Archiv (*-c*), die Schritte dabei werden angezeigt (*-v*), es wird mit gzip komprimiert (*-z*), die Zugriffsrechte werden beibehalten (*-p*) und alles wird in folgende Datei ausgegeben (*-f*). Vor allem *-v* und *-p* sind dabei optional und Sie haben zudem die Möglichkeit, weitere Optionen einzufügen, um Ihr Backup anders zu gestalten.
   * *$BACKUP\_DIR/backup-$DATE.tar.gz* benennt das Verzeichnis (*$BACKUP\_DIR*) und die Datei, in die das Backup gespeichert werden soll. In unserem Beispiel nennen wir diese *backup*, gefolgt vom aktuellen Zeitstempel. Abgeschlossen wird der Dateiname mit der Angabe des Formats, in dem man die Datei erstellt. Wenn Sie eine andere Komprimierung verwenden möchten, müssen Sie daran denken, sowohl das Dateiformat als auch die Option im Befehl zu ändern.
   * Zum Schluss teilen Sie tar durch die Variable *$SOURCE* mit, was es überhaupt archivieren soll. Denkbar ist es, dass Sie zusätzlich mit *--exclude* oder *-X* Verzeichnisse oder Dateien ausschließen, die nicht in das Backup aufgenommen werden sollen.

[Tipp: Prinzipiell spielt es bei Linux und Unix keine Rolle, welche Endung Sie der Skript-Datei geben. Die Systeme lesen die Art der Datei aus, indem sie die Dateistruktur mit einem Magic-File vergleichen. Dabei handelt es sich um eine Datenbank, die sich für gewöhnlich unter */etc/magic* finden lässt. Dennoch hat es sich eingebürgert, Dateiendungen anzugeben, damit Sie als Nutzer leichter den Überblick behalten.]

Nun speichern Sie die Datei mit dem Namen *backup* im *bin*-Verzeichnis und fügen dessen Pfad der PATH-Variable hinzu:

*PATH=$PATH:$HOME/bin*

Das soeben erstellte Backup-Skript müssen Sie noch ausführbar machen:

*chmod u+x $HOME/bin/backup*

Damit machen Sie die Datei nur für Sie selbst (*u*) ausführbar. Sie können die Rechte aber auch an eine Gruppe (*g*), an andere (*o*) oder an alle (*a*) vergeben. Jetzt sind Sie fertig und können das Skript ausführen:

*sudo backup*

Wenn Sie das Backup wieder herstellen möchten, also das Archiv extrahieren, dann reicht hierfür ein Befehl:

*tar -xzf backup.tar.gz -C /*

Das Skript erstellt ein Voll-Backup. Dies ist aber gerade bei der Sicherung eines kompletten Servers nicht immer die richtige Wahl. Deshalb sollten Sie überlegen, ob ein inkrementelles Backup mit tar für Ihre Zwecke nicht sinnvoller ist.

[Hinweis: Beim Erstellen eines Archivs mit absoluten Pfadangaben liefert tar die Meldung: „tar: Entferne führende „/“ von Elementnamen“. Dies ist keine Fehlermeldung, sondern ein Hinweis auf eine Sicherheitsvorkehrung beim Zurückspielen: tar macht aus */home/unterverzeichnis* den Pfad *home/unterverzeichnis*. Wenn Sie sich beim Extrahieren des Archivs nun nicht im Root-Verzeichnis befinden, legt tar eine neue Verzeichnisstruktur an, zum Beispiel: */home/unterverzeichnis/home/unterverzeichnis*. Damit ist die Wahrscheinlichkeit, dass Sie versehentlich Ihr komplettes System überschreiben, verringert. Denken Sie daran: Unix fragt beim Überschreiben nicht nach. Sie müssen also, sollten Sie bestehende Inhalte wirklich ersetzen wollen, zuerst ins Root-Verzeichnis navigieren. Umgehen können Sie dies durch die Option *-P*]

### Was ist ein inkrementelles Backup?

Webmaster erstellen **regelmäßig Backups, um einen Datenverlust zu vermeiden**. Sollte das eigentliche System seinen Dienst verweigern, kompromittiert oder gelöscht worden sein, kann man in solch einem Fall eine funktionierende Version aus dem Backup aufspielen. Je häufiger Sie Speicherpunkte setzen, umso weniger Datenverlust müssen Sie im Notfall hinnehmen. Wenn Sie nun jedes Mal ein vollständiges Backup speichern, also alle Daten des System archivieren, dauert das zum einen sehr lang und zum anderen ist ein großer Speicherbedarf notwendig. Stattdessen können Sie auf inkrementelle Backups setzen.

Ein inkrementelles Backup setzt immer eine Vollspeicherung voraus. Sie müssen das System (oder zumindest den Teil, den Sie sichern möchten) zuerst einmal komplett archivieren. Im Anschluss werden bei einem inkrementellen Backup nur noch **neue oder veränderte Dateien** gesichert. Dies führt zu einem sehr viel kleineren Datenaufkommen, verlangt bei einer Wiederherstellung aber einen größeren Aufwand. Wenn Sie das Backup also wieder aufspielen, benötigen Sie das letzte Voll-Backup und jedes inkrementelle Backup, das Sie seit jener vollständigen Archivierung erstellt haben. Sollte Ihnen eine Datei davon verloren gegangen sein (das ist heutzutage unwahrscheinlicher als noch zu Zeiten von Magnetbändern), ist das Backup lückenhaft.

### Ein inkrementelles Backup mit tar erstellen

Mit tar lassen sich **regelmäßige inkrementelle Backups** erstellen. Auch hierfür schreiben Sie ein eigenes Backup-Skript. Damit können Sie zum Beispiel festlegen, dass einmal pro Monat ein Voll-Backup erstellt wird und dann täglich ein inkrementelles Backup. Außerdem erreicht man mit dem folgenden Skript, dass alte Backups regelmäßig in nach Datum sortierte Ordner verschoben werden. Zusätzlich zu tar benötigen Sie hierfür auch noch cron. Dieser Daemon (ein Programm, das im Hintergrund läuft) lässt zeitbasierte Ausführungen von anderen Prozessen zu – cron ist bei Ubuntu grundsätzlich enthalten. Zunächst öffnen Sie also wieder einen Texteditor und erstellen Ihr Skript:

*#!/bin/bash*

*BACKUP\_DIR="/zielverzeichnis/backup"*

*ROTATE\_DIR="/zielverzeichnis/backup/rotate"*

*TIMESTAMP="timestamp.dat"*

*SOURCE="$HOME/quellverzeichnis "*

*DATE=$(date +%Y-%m-%d-%H%M%S)*

*EXCLUDE="--exclude=/mnt/\* --exclude=/proc/\* --exclude=/sys/\* --exclude=/tmp/\*"*

*cd /*

*mkdir -p ${BACKUP\_DIR}*

*set -- ${BACKUP\_DIR}/backup-??.tar.gz*

*lastname=${!#}*

*backupnr=${lastname##\*backup-}*

*backupnr=${backupnr%%.\*}*

*backupnr=${backupnr//\?/0}*

*backupnr=$[10#${backupnr}]*

*if [ "$[backupnr++]" -ge 30 ]; then*

*mkdir -p ${ROTATE\_DIR}/${DATE}*

*mv ${BACKUP\_DIR}/b\* ${ROTATE\_DIR}/${DATE}*

*mv ${BACKUP\_DIR}/t\* ${ROTATE\_DIR}/${DATE}*

*backupnr=1*

*fi*

*backupnr=0${backupnr}*

*backupnr=${backupnr: -2}*

*filename=backup-${backupnr}.tar.gz*

*tar -cpzf ${BACKUP\_DIR}/${filename} -g ${BACKUP\_DIR}/${TIMESTAMP} -X $EXCLUDE ${SOURCE]*

Auch für dieses Backup-Skript erklären wir Ihnen Schritt für Schritt, was passiert:

* Zunächst definieren Sie wieder den Interpreter.
* Anschließend legen Sie die Variablen fest. Neu hinzugekommen sind ein Verzeichnis für Rotationen der Backups (eine Art Archiv der Backups) und eine Datei für einen Timestamp.
* In unserem Beispiel illustrieren wir, dass es nicht immer sinnvoll ist, alle Verzeichnisse in das Backup zu übernehmen. Ausgeschlossen haben wir in diesem Fall die Inhalte der Ordner (und nicht die Ordner selbst, daher das „\*“) *mnt*, *proc*, *sys* und *tmp*. Denn die Daten in diesen Verzeichnissen sind temporär oder werden bei jedem Systemstart neu erstellt.
* Damit die Pfade alle richtig interpretiert werden, wechselt das Skript mit *cd /* in das Root-Verzeichnis.
* Mit *mkdir* legen Sie das Backup-Verzeichnis an, falls es noch nicht existiert.
* Nun werden alle Variablen eingelesen. Da Sie Ihre Backups durchnummerieren wollen, ermittelt der Code-Block die Nummer des letzten Backups. Dies geschieht, indem das Skript die anderen Bestandteile des Dateinamens entfernt.
* Sie nehmen immer nur 30 Backups auf, danach verschiebt das Skript alle Archivdateien in den Rotationsordner. Dieser wird zuerst erstellt und dann werden mit *mv* alle Dateien, die mit den Buchstaben *b* und *t* beginnen, in den neuen Ordner verschoben. Die Beschränkung der Buchstaben erklärt sich dadurch, dass ohnehin nur damit gekennzeichnete Dateien in dem Ordner sein sollten: nämlich *backup* und *timestamp*. Schließlich setzt das Skript die Backup-Nummer wieder auf 1. Stellt Ihr Skript fest, dass noch keine 30 Backups erstellt wurden, erhöht es die Dateinummer einfach um 1 (*++*).
* Jetzt macht das Skript quasi wieder das rückgängig, was es am Anfang vorgenommen hat: Die Befehle sorgen dafür, dass der Dateiname wieder – jetzt mit der neuen Nummer – vollständig ist.
* Schließlich führt das Skript den eigentlichen tar-Befehl aus: Im Vergleich zum Kommando des einfachen Voll-Backups ist hier eine weitere Option hinzugekommen. Mit -g wird das inkrementelle Backup ermöglicht. Dazu liest tar den Timestamp jeder Datei aus, gleicht diesen mit den bislang in timestamp.dat aufgezeichneten Daten ab und kann so entscheiden, welche Änderungen seit dem letzten Backup entstanden sind. Nur diese werden Teil des neuen Archivs.

[Hinweis: Das Skript verschiebt (bei täglicher Archivierung) jeden Monat die Backup-Dateien in einen neuen Archivordner, damit das eigentliche Backupverzeichnis nur die aktuellsten Daten enthält. Es ist allerdings keine Funktion eingebaut, die die Anzahl der Archivordner begrenzt. Das bedeutet, dass Sie diese manuell löschen müssen.]

Damit ist das Skript zum Erstellen eines inkrementellen Backups mit tar fertig: Speichern Sie die Datei als *backup* im *bin*-Verzeichnis. Auch hier müssen Sie den Pfad exportieren und das Skript ausführbar machen:

*PATH=$PATH:$HOME/bin*

*chmod u+x $HOME/bin/backup*

Theoretisch können Sie jetzt Ihr Backup-Skript mit *sudo backup* starten. Die Idee hinter dem inkrementellen Backup ist aber ja, dass der Vorgang **automatisiert und täglich** durchgeführt wird. Dafür greift man zu cron und ändert die sogenannte Crontab. Dies ist die Tabelle, nach der cron Aufgaben ausführt. Sie hat sechs Felder:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Minuten  (0-59) | Stunden  (0-23) | Tage  (1-31) | Monate  (1-12) | Wochentage  (0-7) | Auftrag |

In die Felder können Sie entweder Zahlen entsprechend des Wertebereichs (in Klammern angegeben) oder einen Asterisk (\*) eingeben. Letzterer bedeutet so viel wie: zu jedem möglichen Wert. Eine Besonderheit stellt die Spalte Wochentag dar. Hiermit können Sie festlegen, dass ein Auftrag zum Beispiel jeden Montag (1) oder nur an Werktagen (1–5) ausgeführt wird. Den Sonntag kann man durch zwei unterschiedliche Werte angeben: Sowohl 0 als auch 7 beziehen sich auf Sonntag, da für einige Menschen die Woche mit diesem Tag beginnt und für andere damit endet.

In der Kommandozeile öffnen Sie den Editormodus von cron mit:

*sudo crontab –e*

Hier fügen Sie folgende Zeile ein:

*30 7 \* \* \* /home/bin/backup*

Damit haben wir veranlasst, dass das Backup jeden Tag (und jeden Monat, unabhängig vom Wochentag) um 7.30 Uhr morgens ausgeführt wird. Speichern Sie Ihre Änderungen ab und Ihr tägliches inkrementelles Backup ist einsatzbereit.

[Hinweis: cron funktioniert nur dann, wenn Ihr System läuft. Bei Webservern sollte das ohnehin der Fall sein. Wenn Sie aber planen, das Skript für die Sicherung Ihres PC oder Laptops einzusetzen, müssten Sie garantieren, dass die Geräte auch jeden Tag um 7.30 Uhr laufen. Ist das Gerät nicht eingeschaltet, fällt das Backup einfach aus. Eine Möglichkeit, dies zu umgehen, bietet anacron. Dieses Programm verschiebt bei einem nicht laufenden System die geplante Ausführung auf einen Moment, in dem das Gerät wieder aktiv ist.]

### System aus einem Backup wiederherstellen

Man möchte es niemandem wünschen, aber der Ernstfall kann eintreten und Ihr System muss wiederhergestellt werden. Mit tar geht auch das relativ leicht und benötigt zudem kein zusätzliches Skript. Ein einzelner Befehl wie bei einem einfachen Voll-Backup ist aber nicht möglich: Es liegt in der Natur des inkrementellen Backups, dass mehrere Dateien entpackt werden müssen. In die Konsole geben Sie diese Befehlszeilen ein:

*BACKUP\_DIR=/zielverzeichnis/backup*

*cd /*

*for archiv in ${BACKUP\_DIR}/backup-\*.tar.gz; do*

*tar -xpzf $archiv -C /*

*done*

[Hinweis: Bei Wiederherstellung des Systems aus dem Backup werden alle Verzeichnisse und damit eventuell auch wichtige Dateien überschrieben.]

Damit Sie nicht jede Archivdatei einzeln extrahieren müssen, verwenden Sie eine for-Schleife:

1. Im ersten Schritt definieren Sie das Verzeichnis, in dem die Backups liegen.
2. Mit *cd /* wechseln Sie in das Rootverzeichnis, damit sichergestellt ist, dass das Archiv auch am korrekten Platz extrahiert wird.
3. Nun starten Sie eine for-Schleife: Dieser Befehl wiederholt alle Aktionen, die zwischen *do* und *done* stehen, bis alle Möglichkeiten durchlaufen sind. Zur Spezifikation des Befehls geben Sie den Pfad Ihrer Backups wieder mit einem Asterisk als Wildcard an, denn Sie möchten alle Archivdateien in diesem Verzeichnis entpacken.
4. Den tar-Befehl spezifiziert man folgendermaßen: Sie extrahieren (*-x*), bei Beibehaltung der Zugriffsrechte (*-p*), und dekomprimieren (*-z*) das Archiv (-*f $archiv*) in das Wurzelverzeichnis (*-C /*).
5. Mit *done* legen Sie den Endpunkt der Schleife fest.

Da Sie beim Erstellen der Archive diese durchnummeriert haben, werden die Backups nacheinander wieder aufgespielt – und zwar beginnend mit dem ältesten. Das ist wichtig: In den Archiven, die nach dem Voll-Backup entstanden sind, befinden sich jeweils neuere Versionen von Dateien. Das bedeutet, dass während der Schleife zunächst die alte Version extrahiert wird, um dann, beim nächsten Durchlauf, mit einer neueren Fassung überschrieben zu werden. Am Ende haben Sie **das komplette System mit dem Backup überschrieben** und die neuste, archivierte Version jeder einzelnen Datei wiederhergestellt.

Dies ist der eigentliche Sinn, eines inkrementellen Backups: die komplette Wiederherstellung des Systems. Mit einem kleinen Umweg ist es aber sogar möglich, **nur eine einzelne Datei zu retten** und die zuvor archivierte, letzte Version zurückzuholen. Dafür gehen Sie in zwei Schritten vor:

*BACKUP\_DIR=/zielverzeichnis/backup*

*ls -l ${BACKUP\_DIR}*

*for archiv in ${BACKUP\_DIR}/backup-\*tar.gz; do*

*tar -tzf $archiv | grep gesuchte-datei;*

*done*

Auch in diesem ersten Schritt greifen Sie auf eine for-Schleife zurück, die allerdings zum Suchen und nicht zum Extrahieren verwendet wird:

1. Sie definieren erneut Ihr Backup-Verzeichnis.
2. Mit dem Kommando ls lassen Sie sich alle Dateien und Ordner im Backup-Verzeichnis anzeigen. Die Option -l ermöglicht detaillierte Informationen.
3. Sie eröffnen eine Schleife wie beim Wiederherstellen des kompletten Archivs.
4. Die wichtige Änderung ist in den Optionen des tar-Befehls zu finden: Statt ein Archiv zu erstellen (*c*) oder zu extrahieren (*x*), lassen Sie den Inhalt der Archive anzeigen (*t*). Da aber nicht Sie selbst nach der Datei suchen wollen, leiten Sie die Ausgabe mit einem Pipe (senkrechter Strich) an den Befehl *grep* weiter. Dieser sucht innerhalb der Ausgabe (also dem Inhalt der Archive) nach der Datei, die Sie suchen.
5. Beenden Sie die Schleife.

Nun zeigt Ihnen das Terminal die gesuchte Datei an – und vielleicht sogar mehrfach, wenn Sie diese regelmäßig bearbeitet haben und sie daher in jedem inkrementellen Backup auftaucht. Merken Sie sich nun den Pfad zu der Datei und bauen Sie eine weitere Schleife, die dann die letzte gespeicherte Version wiederherstellt:

*for archiv in ${BACKUP\_DIR}/backup-\*.tar.gz; do*

*tar –xzf $archiv -C / zielverzeichnis/backup/gesuchte-datei*

*done*

Nun wird die Datei an ihrem ursprünglichen Ort wiederhergestellt und überschreibt dabei auch eine mögliche aktuellere Version.