Datensicherung mit Bacula

Datensicherung klingt einfach, kann sich aber in einer großen und heterogenen Umgebung schnell als komplexe Aufgabe erweisen, die man nur mit professioneller Software bewältigt. Dass solche Software nicht Unsummen an Linzenkosten verschlingen muss, beweist das Open-Source-Projekt Bacula. Gerd Müller

Datensicherung vom PDA bis hin zum größten Mainframe-System und zu allen zukünftigen Versionen der Software aufwärtskompatible Sicherungsmedien – das waren von Anfang an die Ziele des Backup-Projektes Bacula (1), das seit Januar 2000 Kern Sibbald leitet. Er wählte das Open-Source-Modell, weil das seiner Meinung nach am besten zum Generationenvertrag von verlässlicher Sicherung und späterer Rücksicherung passt. Inzwischen hat sich Bacula zu einer ernsthaften Alternative für kommerzielle Produkte entwickelt.

Komponenten

Bacula besteht aus mehreren Komponenten. Im einzelnen sind das:

- Catalog (SQL-DB)
- Director (»bacula-dir«)
- Storage (»bacula-sd«)
- File (»bacula-fd«)
- Console

Diese Komponenten lassen sich auf beliebige Server verteilen, sogar eine standortübergreifende Installation ist nicht nur denkbar, sondern gängige Praxis. Mit Ausnahme des Director und des Catalog kann der Admin alle anderen Bestandteile je nach Anforderung beliebig oft aufspielen. Die Kommunikation zwischen den einzelnen Bausteinen erfolgt über wenige, bei IANA eingetragene Ports (**Abbildung 1**). Dadurch stellen Firewalls kein großes Hindernis für die Kommunikation der Backup-Komponenten dar. Eine Verschlüsselung sämtlicher Verbindungen während der Sicherung ist natürlich möglich. Die einzelnen Bausteine authentifizieren sich gegenseitig mit einem Passwort, das auf beiden Seiten hinterlegt sein muss.

Baculas Gedächtnis

Der Catalog ist das zentrale Gedächtnis von Bacula. In ihm sind sämtliche für den Betrieb notwendige Daten gespeichert. Technisch handelt es sich um eine SQL-Datenbank. Zur Zeit unterstützt die Software MySQL, PostgreSQL und SQLite. Das einfache und offene Backend von Bacula ist eine seiner Stärken. So existieren mittlerweile einige freie Überwachungsskripte für Monitoring-Lösungen. Für Nagios bringt Bacula die sogar mit. Die Skripte nehmen dem Administrator die lästige Pflicht ab, täglich die Sicherung auf Erfolg zu prüfen. Dadurch braucht er nur noch einzuschreiten, wenn ein Backup fehlschlägt.

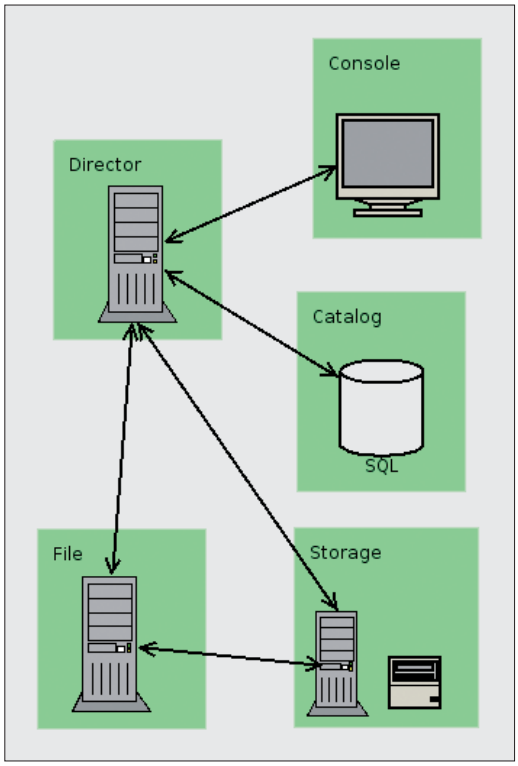

Abbildung 1: So spielen die Komponenten von Bacula zusammen: Die Fäden laufen beim Director zusammen. Bedient wird das System über die Console. Das eigentliche Lesen und Schreiben übernehmen File- und Storage-Daemon.

Director als Leitstelle

Der Director ist die Leitstelle im Backup-System. Er koordiniert sämtliche Komponenten, wertet ihre Daten aus und archiviert sie im Catalog. Außerdem steuert er die Backups entsprechend bestimmter Konzepte, die in seiner Konfiguration hinterlegt sind.

Bacula unterscheidet sich bei diesen Sicherungskonzepten nicht von anderen Projekten ähnlicher Größe oder kommerziellen Produkten.

Listing 1: Konfiguration der Console

```
01 Director {
02  Name = backup-dir
03  DIRport = 9101
04  address = backup
05  Password = "d87Y2YQsX0PQLyDaUHTg5kV
   oWmk7io52"
06 }
```

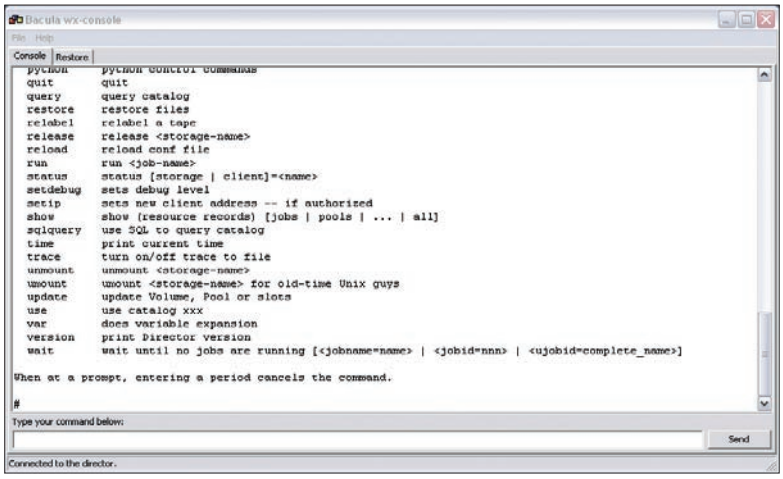

Abbildung 2: Die Windows-Version der Bacula-Console.

Daten lassen sich nach den gängigen Methoden archivieren: vollständig, differenziell oder inkrementell. Ein differenzielles Backup sichert dabei immer sämtliche Daten, die sich seit der letzten Vollsicherung geändert haben. Ein inkrementelles Backup speichert nur die Daten, die sich seit der letzten Sicherung – gleich welchen Typs – geändert haben.

Bei der Steuerung der einzelnen Sicherungen sorgt Bacula immer dafür, dass einer inkrementellen eine Vollsicherung vorausging und diese noch vorhanden ist. Fehlt sie, so befördert Bacula die inkrementelle automatisch zu einer Vollsicherung. Der Catalog dokumentiert sämtliche Abläufe und deren Dateien. Welche Daten zu sichern sind, definieren so genannte File Sets. Sie sind über Jobs den entsprechenden Servern zugeordnet. Zu welchem Termin Bacula sie ausführt, bestimmen Zeitpläne (Schedules), mit denen sie verknüpft sind.

Listing 2: Einstellungen für den File-Daemon

```
01 Director {
02  Name = backup-dir
03  Password = "w4Usp8cV18pitA56WhtUHYSbNat83NgvesmiH"
04 }
05
06 Director {
07  Name = backup-mon
08  Password = "NoGiwMDAU4RWQd4jrqNL4OcHI42EwhtxKEy"
09  Monitor = yes
10 }
11
12 FileDaemon {
13  Name = backup-fd
14  FDport = 9102
15  WorkingDirectory = /var/bacula/working
16  Pid Directory = /var/run
17  Maximum Concurrent Jobs = 20
18 }
19
20 Messages {
21  Name = Standard
22  director = backup-dir = all, !skipped, !restored
23 }
```

Sicherungsmedien

Die nächste Komponente ist der Storage-Daemon. Er stellt dem Director den Zugriff auf die Sicherungsmedien (Volumes) zur Verfügung und übernimmt das Schreiben beim Backup und das Lesen beim Recovery. Bei einem Backup bekommt der Storage-Daemon die Daten direkt vom File-Daemon und gibt sie bei einer Rücksicherung wieder an ihn zurück.

Für die Archivierung kann Bacula bereits mit sehr vielen unterschiedlichen Medien umgehen: vom simplen Plattenplatz auf einer angeschlossenen Festplatte über CDs, DVDs bis hin zu Bändern in großen Libraries (inklusive Barcodelesern), auf die es mit Unterstützung des MTX-Projektes (2) zugreift. Das Format der Sicherungen ist in jedem Fall identisch und – obwohl proprietär – sehr flexibel und auch gut dokumentiert.

Bei Bacula sind Backup-Aufträge nicht auf die Größe eines einzelnen Volume beschränkt. Umgekehrt können die Datenträger auch die Daten beliebiger Sicherungen enthalten. Damit man nicht ständig neue Volumes angelegen muss, räumt Bacula die bereits genutzten Datenträger auf. Das heißt, sind für bestimmte Dateien oder ganze Volumes die Aufbewahrungsfristen abgelaufen, überschreibt das nächste Backup diesen freien Platz wieder. Organisatorisch lassen sich Volumes zu Gruppen, so genannten Pools, zusammenfassen. Damit ist es etwa möglich, für bestimmte Datenträger andere Aufbewahrungsfristen zu definieren oder einzelne Datenträger für unterschiedliche Sicherungskonzepte zu reservieren. Pools verknüpfen die Volumes mit den Jobs.

Sicherungsagenten

Der File-Daemon ist der Agent, mit dem Bacula auf die Daten der zu sichernden Systeme zugreift. Er liest die Daten beim Backup und schreibt sie bei der Rücksicherung auf dem je-

weiligen Server. Er meldet sämtliche Sicherungs- oder Rücksicherungsaufträge und die damit verbundenen Dateien, Volumes und Pools an den Director, der diese Informationen im Catalog archiviert. Die Dateien tauscht der File-Daemon direkt mit dem Storage-Daemon aus.

Für den File-Daemon unterstützt Bacula eine Vielzahl von Unix-Systemen (Linux, Solaris, FreeBSD, OpenBSD, Mac OS X, Tru64, Irix, AIX, BSDI, HPUX, NetBSD und andere) sowie Windows-Systemen (Win98, Win Me, Win XP, Win NT, Win 2000 und Win 2003). Die Rücksicherung von Daten auf ein anderes Betriebssystem ist für Bacula kein Problem. Backup oder Restore erfolgen immer über einen lokalen Agenten, der sich an die vom Betriebssystem vorgegebenen Rechte hält. Natürlich lassen sich auch NFS- oder Samba-Shares sichern.

Besonders erwähnenswert ist, dass Bacula unter Windows die Sicherung von geöffneten Files ermöglicht und die Interaktion mit Applikationen unterstützt, um etwa konsistente Kopien von Datenbanken zu gewährleisten. Es nutzt hierfür bereits seit längerem den Microsoft Volume Shadow Copy Service (VSS). Dadurch hebt sich Bacula auch ein Stück weit von Amanda ab, einer der wenigen Open-Source-Lösungen der gleichen Kategorie. Diese Funktion ist dort erst in Planung. Zusätzlich bietet Bacula die Möglichkeit, auf sämtlichen Plattformen eigene Aktionen unmittelbar vor und nach jeder Sicherung auszuführen. Dieses Verfahren kommt häufig zum Einsatz, um Datenbanken mit deren Bordmitteln auf Platte zu sichern, bevor Bacula wiederum das so entstandene Backup auf ein weiteres Medium schreibt.

Konsole für den Administrator

Die letzte Komponente von Bacula – die Console – stellt das Administrations-Interface dar. Will der Administrator manuell auf Bacula Einfluss nehmen, so ist die Konsole sein Werkzeug. Neben der Shell-basierten Bconsole gibt es auch noch für Windows und für den Gnome-Desktop jeweils eine grafische Konsole (**Abbildung 2** und **Abbildung 3**). Sie unterscheiden sich allerdings primär nur im Aufbau der Menüs von der Bconsole. Alle Konsolen bieten die gleiche, sehr gute Hilfefunktion.

Das folgende, ausführliche Beispiel zeigt die Installation von Bacula anhand eines minimalen Setup mit einem einzigen Server, der Catalog, Director und Storage-Daemon gleichzeitig beheimatet. Außerdem ist er selbst auch noch das zu sichernde System und bekommt aus diesem Grund auch den File-Daemon installiert. Für den Catalog benutzt das Beispiel eine MySQL-Datenbank.

Installation

Die aktuelle Bacula-Version bekommt man von der Projektwebsite **(1)**. Nach dem Auspacken ist Bacula in nur wenigen Schritten installiert. Für den Backup-Server sind folgende Kommandos notwendig:

```
./configure --with-mysql
make
make install
```

Listing 3: Konfiguration des Storage-Daemon

```
01 Storage {
02  Name = backup-sd
03  SDPort = 9103
04  WorkingDirectory = "/var/bacula/working"
05  Pid Directory = "/var/run"
06  Maximum Concurrent Jobs = 20
07 }
08
09 Director {
10  Name = backup-dir
11  Password = "0KaQ38vP8E8R2jKERde1BIZH5p27jimk62MXl1kZ"
12 }
13
14 Director {
15  Name = backup-mon
16  Password = "bY9RZXmxUIZvhWiENsWcrBCwDEvgri2j2Qqx23Pxv"
17  Monitor = yes
18 }
19
20 Device {
21  Name = FileStorage
22  Media Type = File
23  Archive Device = /tmp
24  LabelMedia = yes;
25  Random Access = Yes;
26  AutomaticMount = yes;
27  RemovableMedia = no;
28  AlwaysOpen = no;
29 }
30
31 Messages {
32  Name = Standard
33  director = backup-dir = all
34 }
```

Soll ein Server nur Komponenten für Backup oder Restore enthalten, so ruft man zum Beispiel »./configure --enable-client-only« auf. Anschließend legt der Admin auf dem Backup-Server den Catalog an. Die notwendigen Skripte finden sich zusammen mit allen Konfigurationsdateien unter »/etc/bacula«. Mit

```
./create_bacula_database
./make_bacula_tables
./grant_bacula_privileges
```

ist die Installation abgeschlossen. Zuletzt sind auf beiden Servern nur noch die Init-Skripte (»bacula-ctl-dir« für den Director, »bacula-ctl-sd« für den Storage-Daemon und »bacula-ctl-fd« für den File-Daemon) zu verlinken. Die Init-Skripte liegen auch unter »/etc/bacula/«.

Bacula installiert immer den File-Daemon auch auf dem Backup-Server selbst. Er dient nicht nur dazu, von diesem Server ein Backup zu erstellen, sondern ist auch notwendig, um den Catalog zu archivieren. Damit ist die Installation abgeschlossen und es lohnt ein Blick in die installierten Konfigurationsdateien.

Konfiguration

Die Konfiguration der Komponenten erfolgt über »bacula-dir.conf«, »bacula-fd.conf«, »bacula-sd.conf« und »bconsole«. Sämtliche Komponenten benötigen für dieses Beispiel keine weitere Einrichtung und können unverändert bleiben. Die einfachste dieser Konfigurationsdateien ist die der Console. Sie regelt nur die Kommunikation zum Director (**Listing 1**).

Listing 4: Konfiguration des Director

```
001 Director {
002  Name = backup-dir
003  DIRport = 9101
004  QueryFile = "/etc/bacula/query.sql"
005  WorkingDirectory = "/var/bacula/working"
006  PidDirectory = "/var/run"
007  Maximum Concurrent Jobs = 1
008  Password = "d87Y2YQsXOPQLyDaUHTg5kVoWmk7io52"
009  Messages = Daemon
010 }
011
012 JobDefs {
013  Name = "DefaultJob"
014  Type = Backup
015  Level = Incremental
016  Client = backup-fd
017  FileSet = "Full Set"
018  Schedule = "WeeklyCycle"
019  Storage = File
020  Messages = Standard
021  Pool = Default
022  Priority = 10
023 }
024
025 Job {
026  Name = "Client1"
027  JobDefs = "DefaultJob"
028  Write Bootstrap = "/var/bacula/working/Client1.
    bsr"
029 }
030
031 Job {
032  Name = "BackupCatalog"
033  JobDefs = "DefaultJob"
034  Level = Full
035  FileSet="Catalog"
036  Schedule = "WeeklyCycleAfterBackup"
037  RunBeforeJob = "/etc/bacula/make_catalog_backup
    bacula bacula"
038  RunAfterJob = "/etc/bacula/delete_catalog_
    backup"
039  Write Bootstrap = "/var/bacula/working/
    BackupCatalog.bsr"
040  Priority = 11
041 }
042
043 Job {
044  Name = "RestoreFiles"
045  Type = Restore
046  Client=backup-fd
047  FileSet="Full Set"
048  Storage = File
049  Pool = Default
050  Messages = Standard
051  Where = /tmp/bacula-restores
052 }
053
054 FileSet {
055  Name = "Full Set"
056  Include {
057   Options { signature = MD5 }
058   File = /usr/local/src/bacula-2.0.3
059  }
060  Exclude {
061   File = /proc
062   File = /tmp
```

Der File-Daemon ist nicht nur für dieses Beispiel ausreichend konfiguriert (**Listing 2**), auch im richtigen Einsatz benötigt er kaum weitere Einstellungen. Zusätzlich zur Verbindung zum Director wird lediglich definiert, an welchen Director Statusmeldungen zu übertragen sind.
Beim Storage-Daemon sieht es etwas anders aus. Seine Einstellungen ähneln denen des File-Daemon. Hinzu kommt ein Storage-Device, dass das Installationsskript beigesteuert hat. Im Beispiel ist es ein Device, das Medien unter »/tmp« anlegt (**Listing 3**). Zur Demonstration reicht das aus, für die Praxis jedoch nicht. Daher liefert Bacula in »bacula-sd.conf« noch zahlreiche Beispiele für weitere Geräte mit. Es gibt Vorlagen für DVD-Laufwerke, Bandlaufwerke, Libraries und vieles mehr.

Die Konfiguration des Director (**Listing 4**) ist etwas komplexer. Alle Komponenten finden sich hier wieder und sind über ihre Namen verknüpft. Zusätzlich sind hier die Jobs, Schedules, FileSets und Pools definiert.
Alle Komponenten von Bacula lassen sich nun erstmals starten. Ob die Installation und Konfiguration erfolgreich waren, überprüft der Admin in einer Shell mit der Console »bconsole«. Der Aufruf von »status all« gibt eine Statusübersicht aus. War der Test erfolgreich, kann die erste Sicherung erfolgen.

Die erste Sicherung

Zum Starten eines Backup genügt der Befehl »run«. Zunächst ist der zu sichernde Server auszuwählen. In diesem Beispiel ist dies Client1.

Listing 4: Konfiguration des Director (Fortsetzung)

```
063    File = /.journal
064    File = /.fsck
065   }
066 }
067
068 Schedule {
069  Name = "WeeklyCycle"
070  Run = Full 1st sun at 23:05
071  Run = Differential 2nd-5th sun at 23:05
072  Run = Incremental mon-sat at 23:05
073 }
074
075 Schedule {
076  Name = "WeeklyCycleAfterBackup"
077  Run = Full sun-sat at 23:10
078 }
079
080 FileSet {
081  Name = "Catalog"
082  Include {
083   Options { signature = MD5 }
084   File = /var/bacula/working/bacula.sql
085  }
086 }
087
088 Client {
089  Name = backup-fd
090  Address = backup
091  FDPort = 9102
092  Catalog = MyCatalog
093  Password = "w4Usp8cV18pitA56WhtUHYSbNat83NgvesmiH"
094  File Retention = 30 days
095  Job Retention = 6 months
096  AutoPrune = yes
097 }
098
099 Storage {
100  Name = File
101  Address = backup
102  SDPort = 9103
103  Password = "OKaQ38vP8E8R2jKERde1BIZH5p27jimk62MXl1kZ"
104  Device = FileStorage
105  Media Type = File
106 }
107
108 Catalog {
109  Name = MyCatalog
110  dbname = bacula; user = bacula; password = ""
111 }
112
113 Pool {
114  Name = Default
115  Pool Type = Backup
116  Recycle = yes            # Bacula can automatically recycle Volumes
117  AutoPrune = yes          # Prune expired volumes
118  Volume Retention = 365 days     # one year
119 }
120
121 Console {
122  #...
123 }
```

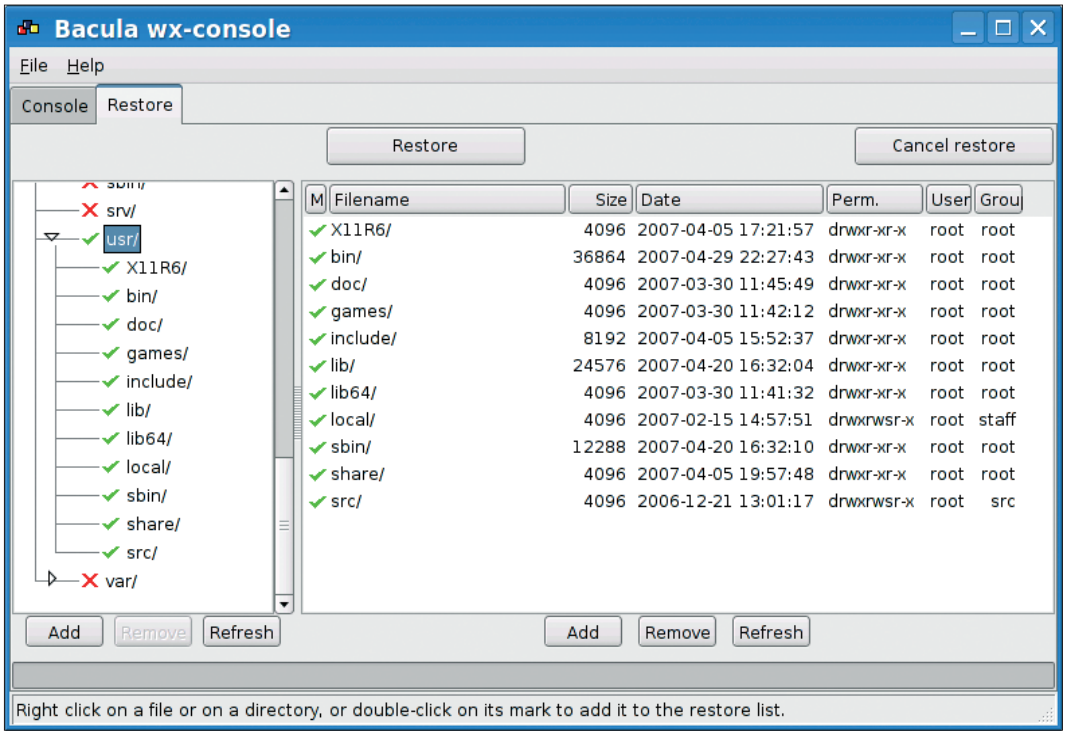

Abbildung 3: Eine grafische Console für Bacula: Sie bietet denselben Funktionsumfang wie die Kommandozeilenversion.

Nun zeigt die Console sämtliche Einstellungen für diesen Job an und fragt, ob er tatsächlich auszuführen ist. Nach »yes« startet das Backup, und die Console gibt die Meldung aus, dass Nachrichten vorliegen.

Sie lassen sich entweder automatisch mit »autodisplay on« oder manuell mit »messages« abrufen. Am Ende der Sicherung liefert Bacula einen Bericht, der alle wichtigen Details zum abgelaufenen Job enthält.

Rücksichern

Ein Restore ist ähnlich einfach und startet nach »restore«. Anschließend kann man festlegen, welche Dateien zurückzusichern sind. Zur Zeit stehen elf Möglichkeiten zur Auswahl, beginnend beim letzten gesicherten Stand und endend bei der Möglichkeit, die Daten mit Hilfe eines SQL-Kommandos im Katalog zu selektieren. Nach Auswahl einer Methode markiert man die gewünschten Dateien, gibt das Ziel der Rücksicherung an und legt fest, ob sie vorhandene Files überschreiben darf. Auch dieser Job erzeugt einen Bericht.

Das hier erläuterte Beispiel nutzt natürlich nur einen Bruchteil der Möglichkeiten von Bacula. Obwohl es nur eine Sicherung in ein Verzeichnis demonstrierte, unterscheidet sich der Ablauf bei einer Sicherung auf Bänder im Großen und Ganzen nicht. Notwendige Vorbereitungen, wie etwa das Labeln der Bänder, kann Bacula entweder automatisch über die voreingestellten Parameter erledigen, oder aber der Admin übernimmt diese Arbeiten selbst und führt sie manuell aus.

Fazit

Das Beispiel zeigt, dass mit Bacula eine Backup-Lösung zur Verfügung steht, die sich kaum hinter den Features kommerzieller Lösungen verstecken muss. Im Unterschied zu den teilweise nicht eben billigen kommerziellen Produkten ist sie zudem kostenlos.

Den Einstieg in Bacula muss auch keiner fürchten. Mit der für ein Open-Source-Projekt fast beispiellos ausführlichen Dokumentation bleiben keine Fragen mehr offen. Man merkt dem Dokument jedenfalls an, dass der Projektleiter Kern Sibbald viel Zeit darauf verwendet. *(jcb)*

Infos

(1) Bacula: (http://www.bacula.org)

(2) MTX-Compatibility: (http://mtx.opensource-sw.net/compatibility.php)

Der Autor

Gerd Müller beschäftigt sich seit über zehn Jahren mit Open Source. Er arbeitet als Consultant bei der NETWAYS GmbH in Nürnberg. Dort leitet er primär Projekte im Nagios- und Bacula-Umfeld.