

BACULA

(It comes by night and sucks the vital essence from your computers.)

Dietz Pröpper

dietz@gweep.de

Wer der Anwesenden besitzt Backups?

- Backups jünger als 24h?
- Backups jünger als eine Woche?
- Backups jünger als ein Monat?
- Weiß nicht.

Eine Lösung des Backupproblems:

BACULA (It comes by night and sucks the vital essence from your computers.)

- Robustes, skalierendes System (2000-Client Installationen existieren.)
- Netzwerkbasiert (ein tape langt für alle (mehr oder minder).)
- Unterstützt Backup verschiedenster Systeme.
- Einfache Konfiguration (no GUI).
- Automatisches Backup.

- Backup-Grundlagen: The Tao of Backup
- Bacula: Aufbau, Features
- Vorführung

The Tao Of Backup: (nach <http://www.taobackup.com/>)

- Abdeckung
- Häufigkeit
- Verteilung
- Historie
- Zuverlässigkeit
- Sicherheit
- Integrität

- Prinzipiell: alles sichern.
 - Häufig hört man, Bewegungsdaten zu sichern sei ausreichend.
 - Aber: Zeitaufwand für Wiederherstellung des Basissystems kann gewaltig sein.
 - Zudem: Sicherung von statischen Daten nützlich für Integritätsprüfung.
 - Last not least: Bare metal recovery ist nützlich.
- Praktische Umsetzung:
 - zu sichernde Daten identifizieren (/tmp ist normalerweise eher uninteressant.)
 - Diese in Klassen (Wichtigkeit, Umfang, Änderungsfrequenz) gruppieren (z.B. Betriebssystem, Konfiguration, Benutzerdaten, jeweils pro zu sichernder Maschine, Templates sind hierzu recht hilfreich.)
 - In Abhängigkeit von Änderungsfrequenz: Plan für Backup erstellen. (z.B. /home täglich, /usr manuell bei Änderung.)

- Wieder: Im Prinzip so häufig wie möglich.
 - Der beste Backup ist der, den man nicht braucht.
 - Der zweitbeste ist der, welchen man gerade 10min bevor man ihn benötigt gezogen hat.
- Praktisch:
 - Änderungsrate.
 - Wiederbeschaffungskosten.
 - Kosten des Backups.
 - Zeitfenster.
- Allerdings: Insbesondere die Wiederbeschaffungskosten sind schwer zu schätzen.

- Nicht nur Hardware versagt. Gelegentlich brennen ebenfalls Häuser ab und treten Flüsse über ihre Ufer. Oder ein Dieb kommt vorbei. Oder ein Staatsanwalt hat eine IP-Nummer falsch abgetippt und lässt die Maschinen abholen.
- Wenn jetzt alle Backups an einer Stelle gelagert sind dann sind sie auch alle weg.
- Daher: Backups nach Möglichkeit an mehreren Stellen verteilt lagern (offsite backup).
Z.B.:
 - Regelmäßig Backups in einer anderen Firmenfiliale (oder bei Eltern, Freunden etc.) lagern.
 - Banken bieten für wenig Geld sehr gut geschützte Schließfächer

- Häufig ist es nützlich, auf „alte“ Backups zugreifen zu können.
 - Jemand hat vor einer Woche (einem Monat/einem Jahr) eine Datei gelöscht.
 - Datenkorruption wird nicht sofort erkannt.
 - Die letzte Lieferung an Bändern war von schlechter Qualität.
- bei keiner oder einer kurzen Backup-Historie sinken die Chancen, eine konsistente Version der Daten zu finden.
- Praktisch: geschachtelte Intervalle, z.B:
 - Die letzte Woche mit täglicher Granularität, ggf. inkrementell., den letzten Monat wöchentlich, das letzte Jahr monatlich.
- Staging:
 - Für Langzeitbackups sind Bänder nachwievor die beste Alternative.
 - Für Kurzzeitbackups (täglich, wöchentlich) verwendet man inzwischen meistens Festplatten (idealerweise geeignete RAIDs).

- Ein nicht getestetes Backupsystem ist wertlos. Typische Fehler:
 - Die Backupsoftware war buggy. Hier sind Low-Level-Tools die notfalls das Backupformat „direkt“ lesen können sehr nützlich.
 - Der Banktresor war nicht antimagnetisch.
 - In der anderen Filiale wurden die Bänder im Regal über der Heizung gelagert.
 - Die Bänder waren schlecht.
 - Versehentlich wurden Bänder nicht getauscht.

→ Sowohl technische, als auch systematische Fehler.

- Die einzige Abhilfe: regelmäßige Tests der Restorefähigkeit.
- Um hierfür nicht Kopien der zu backupenden Systeme vorhalten zu müssen ist es nützlich, wenn das Backupsystem Prüfsummen der gesicherten Daten anlegt, aus welchen man die Konsistenz des Backups ableiten kann.

- Mein einfachster Audit: Wir haben bei der Begehung einfach den Backup der letzten Woche mit genommen.
- Backups sind Kopien der realen Daten! Wir sollten sie also genauso wie die realen Daten schützen.
- Das kann bei verteilter Bandlagerung problematisch sein.
- Vorkehrungen (aufsteigende Nützlichkeit):
 - Medien katalogisieren.
 - Physikalische Sicherheit.
 - Verschlüsselung. Kann bei modernen Backupsystemen zum Bottleneck werden.

- Im Rahmen von Backups: ergeben die gesicherten Daten die Basis für einen funktionsfähigen restore?
 - Wenn ein Programm seine Daten während des Backups ändert dann sind die Chancen groß daß die Dateien nicht in einem konsistenten Zustand gebackupt werden.
 - Ebenfalls wenn Daten über mehrere Dateien konsistent gesichert werden sollen.
- Abhilfe:
 - Problematische Software während des Backups anhalten.
 - Datenbank-Dumps.
 - Dateisystem-Snapshots.

BACULA

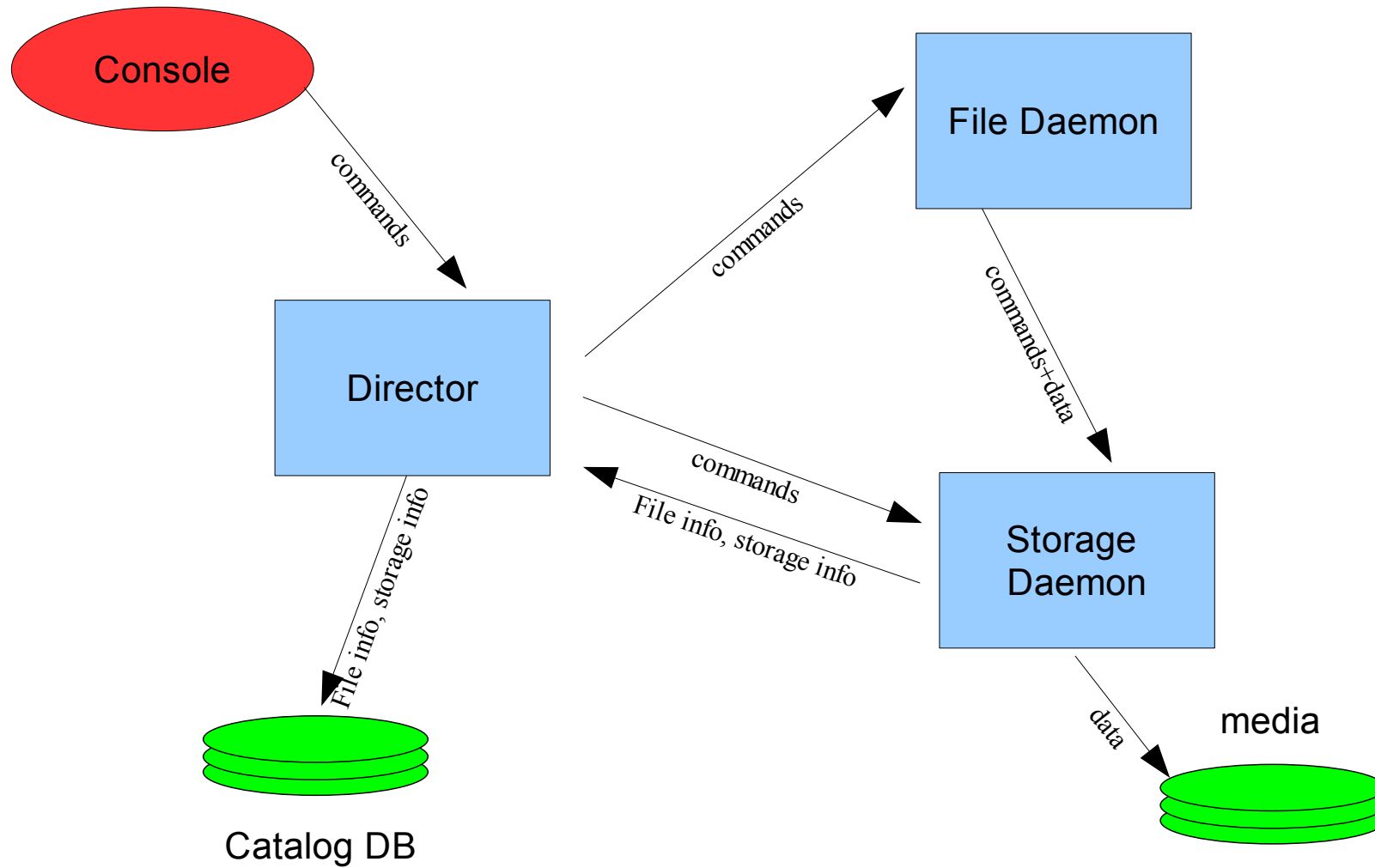
(It comes by night and sucks the vital essence from your computers.)

- Netzwerkbasiertes Backupsystem.
- Unterstützt zur Zeit Linux, Mac OS, diverse Unices und Windows.
- Features vergleichbar mit denen kommerzieller Backupprodukte (Skalierbarkeit, Staging etc.)
- Datenkompatibilität für 30 Jahre. Hardware bitte selber mitbringen.
- Open Source Lizenz (GPL v2 mit Modifikationen).
- Wird seit Januar 2000 entwickelt, große und aktive Entwicklergemeinde, mehr als 30 Entwickler Ende 2007. Aktuelle Version: 5.0.2
- Knappe 1000000 Downloads bei sourceforge.net. (Stand Mitte 2009)

- Seit 2009: kommerzielle Unterstützung der Erfinder: baculasystems.com
- Kern Sibbald, der Projektinitiator ist chairman und CTO, etliche langjährige Entwickler arbeiten mit Bacula Systems zusammen.
- Seit der Gründung neue Versionsnummerierung: „gerade“ Versionsnummern bezeichnen die kommerziell unterstützte Version, „ungerade“ die „freie“ Variante.
- Aktuelle kommerzielle Version: 2.6.1, Version 4 wurde aufgrund des Nummerierungsschemas aus gelassen.
- Bacula Systems bietet kommerziellen Support (zu den üblichen Preisen), Schulung und Beratung.
- Ich habe mit Bacula Ssystems nichts zu tun.

- Verteiltes Backupsystem – viele Clients können gegen einen Backupserver sichern.
- Alle Kommunikation zwischen den Komponenten netzwerkbasierend.
- IPv6-Unterstützung.
- Automatisches Scheduling von Backupjobs, parallele Jobs.
- Interaktives Restore
- Einfache Administration
- Volumelabels, ANSI/IBM-Labels
- Maschinenunabhängiges, erweiterbares Medienformat, Lowlevel-Tools zum Zugriff auf Medien.

- Umfangreiche Verwaltungsfunktionen für Medien-Pools inklusive automatischer Datenmigration (wichtig für Staging).
- Unicodesupport.
- Eventbasiertes Python-Interface.
- Unterstützt bare metal recovery (aka rescue CD mit Backupanbindung).
- Unterstützung der allermeisten Tapelaufwerke und vieler Autochanger.
- Rapid restore.
- Backupverschlüsselung.



- Kontrolliert und steuert die anderen Komponenten.
- Plant die Jobausführung.
- Verwaltet den Katalog.
- Mehrere Kataloge möglich.
- Normalerweise ein Director pro Installation.
- Verschiedene Backuptypen: Voll, inkrementell, differentiell, Migration.

- Residiert als Daemon auf jedem Client.
- Führt alle Dateioperationen (Backup, Restore, Verify) durch.
- Operationen werden durch den Director gesteuert.
- Spricht (wahlweise per TLS) mit Director und Storage Daemon.
- Benötigt (selbstverständlich) Zugriff auf die zu sichernden Daten.
- Prüfsummenbildung, Verschlüsselung und Komprimierung finden hier statt. (Damit: Aufwand auf die Clients verteilt, wichtig für parallele Jobausführung.)
- Muß mit Rootprivilegien laufen, kann seit Version 5 Schreibrechte dropen.

- Schreibt und liest von den physikalischen Medien.
- Unterstützt Backup auf Festplatte, Bandlaufwerk, DVD.
- Einfach anpassbare (shell script) Unterstützung für Bandwechsler (via mtx.)
- Liest und schreibt Daten von und zu den File Daemons.
- Sendet Speicherorte (z.B. Bandoffset) an den Director, der sie im Katalog ablegt (Wichtig für schnellen restore.)

- SQL-Datenbank (momentan MySQL, PostgreSQL oder SQLite.)
- Speichert Informationen zu gesicherten Dateien (storage location, Checksums,...), Jobs, Medien,...
- Automatisches Löschen alter Metadaten.
- Mehrere Kataloge möglich.
- Wichtig: Datenbank passend tunen! Insbesondere bei langem Backlog kommen hier schnell etliche TB zusammen.

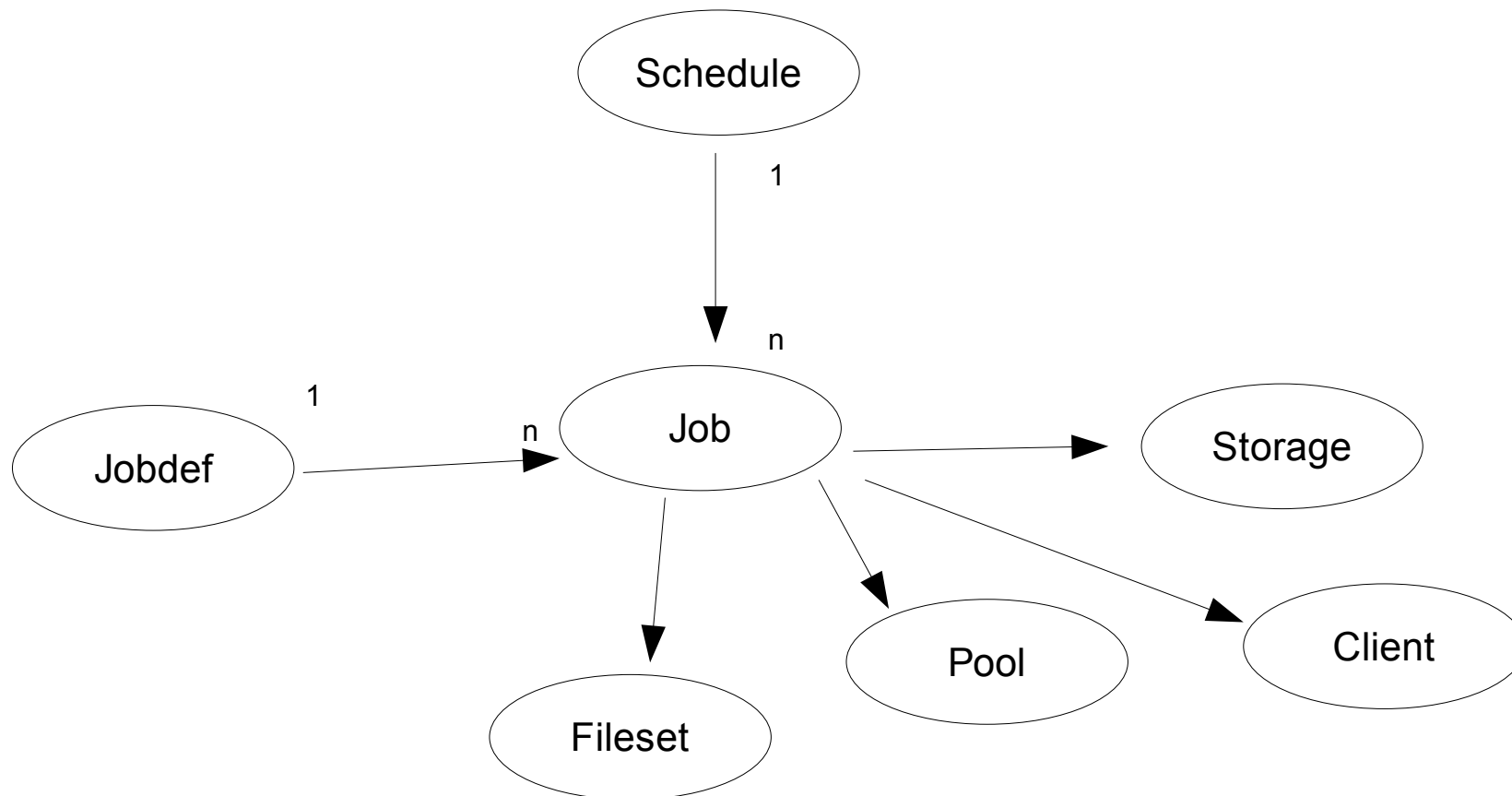
- Erlaubt die Steuerung von Bacula.
 - Starten/Stoppen von Jobs.
 - Katalogabfragen.
 - Dateiauswahl für Restore.
 - Möglichkeit zur Rechtevergabe.

- Verschiedene Consolen
 - TTY (bconsole.)
 - qt-basiert (BAT.)
 - etliche Webinterfaces.
 - gnomebasiert (bgnome.)

- Director und Storage Daemon können non-root laufen. Der File Daemon benötigt Lesezugriff auf zu sichernde Daten und läuft daher üblicherweise als root.
- Daemonen authentifizieren sich via CRAM-MD5.
- TLS zwischen den Daemonen möglich. Tunneling via ssh einfach machbar.
- Verschiedene Signaturen für gesicherte Dateien (MD5, SHA1,...), werden im Katalog gespeichert.
- CRC-Checksummen auf Backupmedien.
- ACLs für die Consolen.
- Asymmetrische Verschlüsselung (RSA+AES) der Medien (Dateinamen werden nicht verschlüsselt.)

- Jeder Daemon wird über eine eigene Konfigurationsdatei konfiguriert.
- Ein Großteil der Konfiguration findet in der Director-Konfiguration statt.
- Der Director kennt (im Wesentlichen) Clients, Jobs, Filesets, Schedules, Pools und Storages.
 - Ein Client ist ein zu sichernder Rechner.
 - Ein Fileset bestimmt die zu sichernden Daten.
 - Ein Pool definiert die Backupmedien.
 - Einem Job sind jeweils ein Fileset, ein Client, ein Schedule, ein Storage und ein Pool zugeordnet. Das Fileset wird für den jeweilige Client auf den entsprechenden Pool gesichert.
 - Ein Schedule definiert, wann und wie eine Gruppe von Jobs ausgeführt wird.
 - An verschiedenen Stellen können minimale Haltezeiten für Medien oder Jobs definiert werden.

Das Ganze als Bild:



- Zwei simulierte Clients, Bert und Fred
- Dateibasiertes Backup

FRAGEN?