Bacula
The Network Backup Solution

Presented at FOSDEM on 24 February 2007 in Brussels

- Bacula – the Network Backup Tool for Linux, Mac, Unix and Windows

Project History and Statistics

- Bacula = Backup + Dracula

- January 2000 – Project started
- 14 April 2002 – First release to Source Forge (1.16)
- 29 June 2006 – Release 1.38.11
- January 2007 – Release 2.0.0

- 41 project members; 33 with CVS write access
- 656 users registered in the bugs database
- 1,444 users subscribed to bacula-users
- 712 users subscribed to bacula-announce

- Downloads
  - 74 version 1.16
  - 10,919 version 1.36
  - 12,408 version 1.38.11

Introduction

Do you do backups?

- No
- Yes, I did one last month
- Yes, tarballs every week
- Sometimes I rsync ...
- Yes, CDs every week

Problems with this:

- Finding the files you need to restore
- Restoring to a point in time
- Knowing what was backed up when
- Scaling (how do you handle 2000 machines?)
- Bare-metal recovery is complex

Introduction (Cont)

Bacula is a network backup solution, designed for Linux, Mac OS, Unix and Windows systems.

Project goals:

- To be able to backup any client from a Palm to a mainframe computer with the same software
- To have “Enterprise” features similar to what are available in the largest commercial applications
- To assure compatibility of your data for at least 30 years (providing you have the appropriate hardware)
- A Free and Open Source (GPL v2 + additions) license

Bacula components

- Bacula Director
- Bacula File daemon
- Bacula Storage daemon
- Bacula Console
- Bacula Catalog
- Bacula Monitor

Bacula Components

Director

- Knows everything about everything
- Initiates and supervises all activities
- Runs backup, restore, verify and archive jobs
- Maintains the catalog
- Decides what files are backed up on which client and where
- Typically one Director except in very large shops

Bacula Components

File daemon

- Client program that communicates over network with Director and Storage daemon
- Does file backup and restore as requested by Director
- Installed on each system to be backed up
- Needs access to all files to be backed up
- Adapted specifically to each OS
- Typically multiple File daemons per Director

Bacula Components

Storage daemon

- Accepts commands from the Director
- Accepts data from File daemons (FD)
- Reads and writes data on the physical medium
  - Disk
  - Tape
  - DVD
  - USB
- Typically one per Director

Bacula Components

Console

- Allows user or administrator to control Bacula
- Communicates with Directory via network
- Start, cancel, restore, verify backups
- Consoles available
  - TTY
  - wxWidgets (GUI) – Linux, Unix, Win32
  - Gnome (GUI)
  - Several web interfaces
- Restricted consoles (ACL) for security

Bacula Components

Catalog database

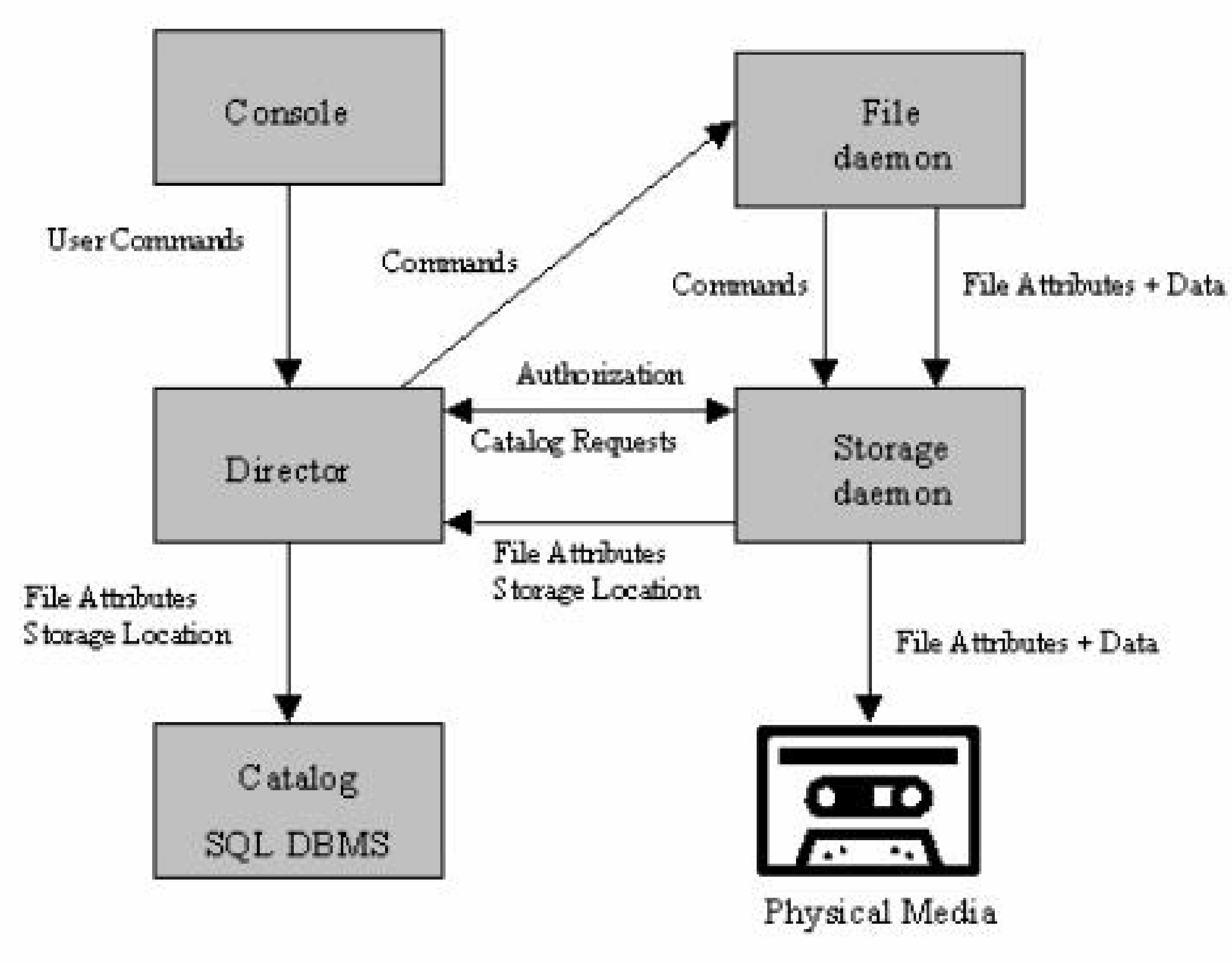
- SQL database
- Maintains indexes and Volume databases
- Permits rapid restores
- Allows inquiry of when and where files were backed up
- Uses MySQL, PostgreSQL, or SQLite
- Supports multiple databases of selected type

Bacula Components

Tray Monitor

- Gnome/KDE/Win32 GUI tray applet to monitor components
- Queries Director, File daemons, Storage daemon
- Near real-time display of activity

Main Components



Features

- Distributed backup with a central server and catalog
- All components communicate via the network and can be deployed separately.
- Internal scheduler for automatic and simultaneous job execution with priorities.
- Restore of one or more files selected interactively either for current backup or a backup prior to a specified time and date (many more options).
- Pool and library volume management.
- Multi-volume saves and multiple saves per volume with checksum and hash code for data security.
- Simple administration with several consoles (command line, GUI, and web)
- Rapid restoration of individual files (one user reported 4 to 6 hours with tar and 3 to 4 minutes with Bacula!).
- Rescue CDROM for “bare metal” recovery.

Features (cont.)

- Support for ANSI / IBM labels
- Support for multiple drive autochangers (libraries)
- Support for Unicode/Win32
- Secured communications with built-in TLS (SSL) between components
- Support for direct writing on DVD
- Integrated Python interpreter for “event” (job start, end, ...) scripting
- Specification of multiple possible devices with automatic selection of a free device
- Multiple simultaneous copies of the same backup on different volumes

Bacula – Hardware Features

- Wide hardware support with configurable Device resources
- Support for multiple drive autochangers (libraries)
- Support for tape barcode readers
- Pool and Volume library management
- Backups can span multiple volumes
- Multiple backups (jobs, clients, OSes) per volume

Bacula – Security Features

- Director and Storage daemon can be run non-root
- MD5, SHA1, ... signatures for each file
- CRC checksum for each Volume block
- CRAM-MD5 daemon authorization
- Restricted consoles and tray-monitors
- Communications (TLS) encryption
- Data (PKI) encryption
- Tripwire like Verify feature

Bacula – Director Configuration File

```
Director {
  Name = bacula-dir
  DIRport = 9101
  Query File = “/usr/local/etc/query.sql”
  Working Directory = “/var/bacula”
  PID Directory = “/var/run”
  Maximum Concurrent Jobs = 20
  Password = “secret”
  Messages = Standard
}
```

Bacula – Director Configuration File (cont)

```
Job {
  Name = “Laptop”
  Type = Backup
  Client = laptop-fd
  FileSet = “Full Set”
  Schedule = “Weekly Cycle”
  Storage = File
  Messages = Standard
  Pool = Standard
  Write Bootstrap = “/var/bacula/laptop.bsr”
  Priority = 10
}
```

Bacula – Director Configuration File (cont)

```
Schedule {
  Name = “Weekly Cycle”
  Run = Level=Full 1st sun at 2:05
  Run = Level=Differential 2nd-5st sun at 2:05
  Run = Level=Incremental mon-sat at 2:05
}
```

Bacula – Director Configuration File (cont)

```
Client {
  Name = laptop-fd
  Address = laptop.example.org
  FDPort = 9102
  Catalog = MyCatalog
  Password = “secret-fd”
  File Retention = 30 days
  Job Retention = 6 months
  AutoPrune = yes
  Maximum Concurrent Jobs = 20
}
```

Bacula – Director Configuration File (cont)

```
FileSet {
  Name = “Full Set”
  Include {
    File = /
    File = /usr
    File = /var
}
Exclude {
   File = /proc
   File = /tmp
   File = /sys
   File = /.journal
}
```

Bacula – File daemon Configuration File

```
FileDaemon {
  Name = laptop-fd
  FD Port = 9102
  Working Directory = /var/bacula
  PID Directory = /var/run
}

Director {
  Name = bacula-dir
  Password = “secret-fd”
}
```

Bacula – Storage Configuration File

```
Storage {
  Name = bacula-sd
  SD Port = 9103
  Working Directory = /var/bacula
  PID Directory = /var/run
}

Director {
  Name = bacula-dir
  Password = “secret-sd”
}
```

Bacula – Storage Configuration File (cont)

```
Device {
  Name = FileStorage
  Device Type = File       # DVD, Tape, FIFO
  Media Type = File
  Archive Device = /usr/bacula-backups
  Label Media = yes
  Random Access = yes
}

AutoChanger {
  ...
}
```

Technical Highlights

- Multi-OS support : Linux (all versions including the zSeries), Win32, Solaris, *BSD, Mac OS X, Irix, Tru64, AIX, HP-UX
- Catalog : support for MySQL, PostgreSQL or SQLite
- Backup to disk or tape with spooling capability to avoid “shoe-shine” on tapes
- Wide hardware support with configurable Device resources, including support for multiple drive autochangers (libraries).
- Backup/restore of POSIX Access Control Lists (ACL)
- Support for large files (>2GB) and 64 bit architectures
- Secured communication between components with CRAM-MD5 authentication and optional encryption
- Multi-thread implementation
- Written in a subset of C++

Bacula Configuration - Job

Jobs are the basic unifying structure

- Type – what to do, backup, restore, migrate, ...
- Level – level of detail of type
- FileSet – what to and what not to backup
- Client – where to get the files
- Storage – where to put the files (which hardware)
- Pool – which set of Volumes (tapes, files) to use
- Schedule – when to do it

Bacula Configuration – Job Type

- Backup – save files
- Migration – move files from Volume to Volume
- Verify – verify Volumes or Client files
- Admin – run administrative tasks
- Restore – restore files

Bacula Configuration – Job Level

- Full – everything defined in FileSet
- Differential – everything changed since last Full
- Incremental – everything changed since last Full, Diff
- Catalog – Compare Client files to catalog
- InitCatalog – Create base for Client comparison
- VolumeToCatalog – verify Volume files to catalog
- DiskToCatalog – verify Client files to catalog

Bacula Configuration – FileSet

- Include/Exclude files and/or directories
- Regex or wildcard for file/directory name selection
- Compression using similar selection criteria
- Which filesystem types to backup
- ACL support selection
- Sparse file handling
- Signature (MD5, SHA1, ...)

Developers

Most developers believe that backup programs are not very "sexy". However, they **are** extremely complicated:

- Database – SQL
- GUI
- Web
- Networking
- OS (every bit must be restored)
- Restore
  - Full, Diff, Inc
  - Find Volumes
  - Directory tree (100 million records, 5 million files)
  - where is the file?
- Job resource allocation (drives, Volumes)
- Differencing

Project development

Site : http://www.bacula.org/

Development style:

- Open with SourceForge project
- Patches and commits reviewed, and releases planned by K. Sibbald
- Developer Subversion access. Currently 33 developers may commit.
- Comprehensive developer's guide with code style guidelines
- All code tested using a regression test suite.
- Email list for developers (bacula-devel).

Release organization:

- Development code in Subversion HEAD
- Releases are maintained/updated via stable branches

Resources

Downloads : http://download.sourceforge.net/bacula

For IT professionals and system administrators

- Manual: http://www.bacula.org/rel-manual/index.html
- OS and Hardware compatibility lists (in manual)
- Bugs reports: http://bugs.bacula.org/
- Email support list : bacula-users@lists.sf.net

For developers

- Docs : http://www.bacula.org/developers/index.html
- Email list : bacula-devel@lists.sf.net
- CVS at Source Forge

Credits

Thanks

- Dan Langille who created the original presentation
- Karl Cunningham who updated it