From Bacula to Bareos

Migration of a backup environment

by Daniel Holtkamp of
Riege Software International
Content

- Introduction
- Mission Briefing
- Old Systems & New Systems
- Migration Plan
- Testing
- Configuration
- Deployment
- Problems
- Results
# About me

<table>
<thead>
<tr>
<th>Name</th>
<th>Daniel Holtkamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>36</td>
</tr>
<tr>
<td>Occupation</td>
<td>Senior System Administrator @Riege Software International GmbH</td>
</tr>
<tr>
<td>Areas of Expertise</td>
<td>Red Hat Certified Engineer MySQL DBA MongoDB Backup Infrastructure Python Scripting PBX/Voip and more</td>
</tr>
</tbody>
</table>
Riege Software International

- Specialized in software development for the cargo industry
- Family owned and operated since 1985
- Over 30 years of experience in direct forwarding and logistics
- Supported by 80+ employees
- Located in Düsseldorf-Meerbusch/Germany
- 7 branches in Europe, Asia and North America
Mission Briefing

- Rework of Backup Infrastructure

- Hardware is outdated and will be replaced

- Good opportunity to switch to Bareos

- Major configuration rewrite

- Integration of lessons learned

- Accessibility of previous backups
**Old Hardware MB**

**Dell PowerEdge 2900**

1x Xeon E5335 @ 2.00GHz (4-core) / 32GB  
Spool: 200GB Raid-5 @ 4x 15K SAS (Internal)  
Database: 1.4TB Raid-10 @ 12x 300GB 15K SAS  
(Direct Attached Storage)

Storage A: 4.5TB @ EMC AX100 (FC SAN)  
Storage B: 4.5TB @ EMC AX100 (FC SAN)  
PowerVault TL2000 - 2 Drive Autochanger

1.2TB Database  
470 Clients  
~10.343.555.277 files  
~1.806.415.779 unique filenames  
~6.491.669 unique paths

---

**New Hardware MB**

**Dell T620**

2x Xeon E5-2620 v2 @ 2.10GHz (6-core) / 64GB  
Spool: 300GB Raid-1 @ 2x 300GB 15K SAS  
Database: 1.8TB Raid-6 @ 6x 480GB SSD  
Storage: 16TB Raid-6 @ 20x 1TB 7.2K SAS  
PowerVault TL2000 - 2 Drive Autochanger
Old Hardware FRA

Dell PowerEdge 2950

1x Xeon L5420 @ 2.50GHz (4-core) / 32GB
4.8TB Raid-5 @ 6x 7.2K SAS (Internal)

Average backup holding time of 14 days

171 clients
33.820.230 files
185.850.466 unique filenames
14.621.702 unique paths

New Hardware FRA

Dell R720xd

2x Xeon E5-2620 v2 @ 2.10GHz (6-core) / 64GB
Spool: 300GB Raid-1 @ 2x 300GB 15K SAS
Database: 480GB Raid-1 @ 2x 480GB SSD
Storage: 13TB Raid-6 @ 16x 1TB 7.2K SAS
Migration Plan

- Test configuration options
- Get a general idea on what to do
- Migrate test & dev systems to backup test server
- Migrate FRA datacenter
- Migrate MB datacenter
- Migrate HKG datacenter
Testing

- Dedicated test server
  - Virtual Machine on RHEV Cluster
  - Storage requirements not too high (300GB allocated)
- Check out new features
  - SD to SD copy for migration between data centers?
- Try configuration options
  - Make sure storage configuration still works
  - Multiple catalogs
  - Automatic client configuration
- Proof of concept
  - Final configuration skeleton, test server “in production” (backing up test & dev systems)
Testing - results

- SD to SD copy for migration between data centers?
  - Sadly it only works with one director. We prefer one director per Datacenter for autonomy reasons

- Make sure storage configuration still works
  - Storage configuration still works

- Multiple catalogs
  - Multiple catalog feature not really working right now, single catalog is the way to go

- Automatic client configuration
  - Wrote a service that handles new clients - details later in this presentation

- Final configuration skeleton, test server “in production” (backing up test & dev systems)
  - Configuration ready for live deployment, test server configuration working nicely
Director configuration

Director {
    Name = backup.mb-dir
    Maximum Concurrent Jobs = 20
    Password = "password"
    Messages = Daemon
    WorkingDirectory = /var/backup/bareos/director-working
}

# Include split config files.
@/etc/bareos/conf.d/catalogs.conf
@/etc/bareos/conf.d/filesets.conf
@/etc/bareos/conf.d/jobdefs.conf
@/etc/bareos/conf.d/messages.conf
@/etc/bareos/conf.d/pools.conf
@/etc/bareos/conf.d/schedules.conf
@/etc/bareos/conf.d/storages.conf

# Clients
@|"sh -c 'cat /etc/bareos/clients/mb/*.conf'"
@|"sh -c 'cat /etc/bareos/clients/fra/*.conf'"
@|"sh -c 'cat /etc/bareos/clients/hkg/*.conf'"
@|"sh -c 'cat /etc/bareos/clients/special/*.conf'"
@|"sh -c 'cat /etc/bareos/clients/retired/*.conf'"
Sample client config

Client {  
Name = clientname  
Address = clientname  
FDPort = 9102  
Password = "clientpassword"  
Catalog = backup.mb.base.catalog  
File Retention = 1080 days  
Job Retention = 3600 days  
AutoPrune = yes  
}  

Job {  
Enabled = yes  
Name = clientname  
Client = clientname  
JobDefs = DefaultJob  
filesetplaceholder  
Pool = clientname  
}  

Job {  
Name = clientname.copy  
Client = clientname  
JobDefs = DefaultCopyJob  
filesetplaceholder  
Pool = clientname  
}  

Pool {  
Name = clientname  
Pool Type = Backup  
Label Format = "clientname."  
Next Pool = TL  
Maximum Volume Jobs = 1  
Maximum Volumes = 40  
Volume Retention = 30 days  
AutoPrune = yes  
Recycle = yes  
}  

Client {  
Name = clientname  
Address = clientname  
FDPort = 9102  
Password = "clientpassword"  
Catalog = backup.mb.base.catalog  
File Retention = 1080 days  
Job Retention = 3600 days  
AutoPrune = yes  
}  

Job {  
Enabled = yes  
Name = clientname  
Client = clientname  
JobDefs = DefaultJob  
filesetplaceholder  
Pool = clientname  
}  

Job {  
Name = clientname.copy  
Client = clientname  
JobDefs = DefaultCopyJob  
filesetplaceholder  
Pool = clientname  
}  

Pool {  
Name = clientname  
Pool Type = Backup  
Next Pool = TL  
Label Format = "clientname."  
Maximum Volume Jobs = 1  
Maximum Volumes = 40  
Volume Retention = 30 days  
AutoPrune = yes  
Recycle = yes  
}  

Client {  
Name = clientname  
Address = clientname  
FDPort = 9102  
Password = "clientpassword"  
Catalog = backup.mb.base.catalog  
File Retention = 1080 days  
Job Retention = 3600 days  
AutoPrune = yes  
}  

Job {  
Enabled = yes  
Name = clientname  
Client = clientname  
JobDefs = DefaultJob  
filesetplaceholder  
Pool = clientname  
}  

Job {  
Name = clientname.copy  
Client = clientname  
JobDefs = DefaultCopyJob  
filesetplaceholder  
Pool = clientname  
}  

Pool {  
Name = clientname  
Pool Type = Backup  
Next Pool = TL  
Label Format = "clientname."  
Maximum Volume Jobs = 1  
Maximum Volumes = 40  
Volume Retention = 30 days  
AutoPrune = yes  
Recycle = yes  
}
FileSet {
  Name = "FullSet"
  Include {
    Options {
      compression=GZIP
      signature = MD5
      aclsupport = yes
      xattrsupport = yes
    }
    File = ":/usr/local/bin/local_partitions"
  }
  Exclude Dir Containing = .backupexclude
}
Exclude {
  File = /var/lib/bareos
  File = /var/backup
  File = /proc
  File = /tmp
  File = /var/tmp
  File = /var/cache
  File = /.journal
  File = /.fsck
}
}
JobDefs {
    Name = "DefaultJob"
    Type = Backup
    Level = Incremental
    FileSet = "FullSet"
    Schedule = "mb-weekly-cycle"
    Messages = Standard
    Priority = 10
    Write Bootstrap = "/var/backup/bareos/bootstraps/%c.bsr"
    Rerun Failed Levels = no
    Max Full Interval = 30 days
    Storage = backup.mb.filestorage
    Allow Mixed Priority = yes
    Accurate = yes
    Allow duplicate Jobs = no
    Cancel Lower Level Duplicates = yes
    Cancel Queued Duplicates = yes
    Cancel Running Duplicates = no
}

JobDefs {
    Name = "DefaultCopyJob"
    Enabled = yes
    Type = copy
    Messages = Standard
    FileSet = EmptySet
    Schedule = "mb-filestorage-copy-cycle"
    Selection Type = PoolUncopiedJobs
    Write Bootstrap = "/var/backup/bareos/bootstraps/%c.bsr"
    Priority = 10
    Allow Mixed Priority = yes
    Storage = backup.mb.copy.filestorage
}

JobDefs {
    Name = "DefaultJobFra"
    Type = Backup
    Level = Incremental
    FileSet = "FullSet"
    Schedule = "fra-weekly-cycle"
    Messages = Standard
    Priority = 10
    Write Bootstrap = "/var/backup/bareos/bootstraps/%c.bsr"
    Rerun Failed Levels = no
    Max Full Interval = 30 days
    Storage = backup.fra.filestorage
    Allow Mixed Priority = yes
    Accurate = yes
    Allow duplicate Jobs = no
    Cancel Lower Level Duplicates = yes
    Cancel Queued Duplicates = yes
    Cancel Running Duplicates = no
}

JobDefs {
    Name = "DefaultCopyJobFra"
    Enabled = yes
    Type = copy
    Messages = Standard
    FileSet = EmptySet
    Schedule = "fra-filestorage-copy-cycle"
    Selection Type = PoolUncopiedJobs
    Write Bootstrap = "/var/backup/bareos/bootstraps/%c.bsr"
    Priority = 10
    Allow Mixed Priority = yes
    Storage = backup.fra.copy.filestorage
}
JobDefs {
    Name = "DefaultJobHKG"
    Type = Backup
    Level = Incremental
    FileSet = "FullSet"
    Schedule = "hkg-weekly-cycle"
    Messages = Standard
    Priority = 10
    Write Bootstrap = "/var/backup/bareos/bootstraps/%c.bsr"
    Rerun Failed Levels = no
    Max Full Interval = 30 days
    Storage = backup.hkg.filestorage
    Allow Mixed Priority = yes
    Accurate = yes
    Allow duplicate Jobs = no
    Cancel Lower Level Duplicates = yes
    Cancel Queued Duplicates = yes
    Cancel Running Duplicates = no
}

JobDefs {
    Name = "DefaultCopyJobHKG"
    Enabled = yes
    Type = copy
    FileSet = EmptySet
    Schedule = "hkg-filestorage-copy-cycle"
    Selection Type = PoolUncopiedJobs
    Write Bootstrap = "/var/backup/bareos/bootstraps/%c.bsr"
    Priority = 10
    Allow Mixed Priority = yes
    SpoolAttributes = yes
    Spool Data = yes
    Storage = backup.hkg.copy.filestorage
}

Job {
    Name = "archive"
    Enabled = no
    Type = copy
    Messages = Standard
    FileSet = EmptySet
    Client = backup.mb.riege.local
    Schedule = archive-copy-cycle
    Selection Type = SQLQuery
    Selection Pattern = "select JobId from toarchive order by Id asc limit 100"
    Pool = TL
    Write Bootstrap = "/var/backup/bareos/bootstraps/%c.bsr"
    Priority = 10
    Allow Mixed Priority = yes
}
#!/bin/bash
#
# Manage bareos archive jobs.
#
# Cleanup old list
mysql bareos -e "delete from toarchive;"

# Split subquery to speed up mysql
PRIORS=$(mysql bareos -N -e"SELECT PriorJobId from Job where PoolId=4 AND Type IN ('B','C') AND JobStatus in ('T','W')")
PRIORS=$(echo $PRIORS|sed s/"/,/g)

# Find jobs to be copied, sort them in the order they are on the source tape.
TOARCHIVE=$(echo $TOARCHIVE|sed s/"/\),\(/g)

# No jobs found - exit.
# Otherwise put them in the table bareos will read from and start the job.
if [ -z ${TOARCHIVE} ]; then
echo "no jobs found"
else
 mysql bareos -e"INSERT INTO toarchive (JobId) VALUES (${TOARCHIVE});"
 /usr/sbin/bconsole <<EOF
 run archive
 yes
 quit
 EOF
 run archive
fi
Pool {
  Name = Default
  Pool Type = Backup
  Recycle = yes
  AutoPrune = yes
  Volume Retention = 30 days
}

Pool {
  Name = Scratch
  Pool Type = Backup
  RecyclePool = Scratch
  Storage = library
}

Pool {
  Name = TL
  Pool Type = Backup
  Next Pool = Archive
  Volume Retention = 3600 days
  AutoPrune = no
  Storage = library
}

Pool {
  Name = Archive
  Pool Type = Backup
  Volume Retention = 3600 days
  AutoPrune = no
  Storage = library
}

Pool {
  Name = Databases
  Pool Type = Backup
  Volume Retention = 30 days
  AutoPrune = Yes
  RecyclePool = Scratch
  Storage = library
}

Schedule {
  Name = "mb-weekly-cycle"
  Run = Differential 1st-5th sun at 21:05
  Run = Incremental mon-sat at 21:05
}

Schedule {
  Name = "fra-weekly-cycle"
  Run = Differential 1st-5th sat at 22:05
  Run = Incremental sun-fri at 22:05
}

Schedule {
  Name = "mb-filestorage-copy-cycle"
  Run = Level=Full sun-sat at 11:00
}

Schedule {
  Name = "fra-filestorage-copy-cycle"
  Run = Level=Full sun-sat at 13:00
}

Schedule {
  Name = "archive-systems-cycle"
  Run = Level=Full Jan 1st at 21:00
  Run = Level=Differential Feb 1st at 21:10
  Run = Level=Differential Mar 1st at 21:10
  Run = Level=Differential Apr 1st at 21:10
  Run = Level=Differential May 1st at 21:10
  Run = Level=Differential Jun 1st at 21:10
  Run = Level=Full Jul 1st at 21:00
  Run = Level=Differential Aug 1st at 21:10
  Run = Level=Differential Sep 1st at 21:10
  Run = Level=Differential Oct 1st at 21:10
  Run = Level=Differential Nov 1st at 21:10
  Run = Level=Differential Dec 1st at 21:10
}
Director storage resource

Storage {
  Name       = library
  Address    = 10.11.0.72
  SDPort     = 9103
  Password   = "password"
  Device     = library
  Media Type = LTO5
  Maximum Concurrent Jobs = 2
  Autochanger = yes
}

Storage {
  Name       = backup.mb.filestorage
  Address    = 10.11.0.72
  SDPort     = 9103
  Password   = "password"
  Device     = backup.mb.filestorage-1
  Device     = backup.mb.filestorage-2
  Device     = backup.mb.filestorage-3
  Device     = backup.mb.filestorage-4
  Device     = backup.mb.filestorage-5
  Device     = backup.mb.filestorage-6
  Device     = backup.mb.filestorage-7
  Device     = backup.mb.filestorage-8
  Device     = backup.mb.filestorage-9
  Device     = backup.mb.filestorage-10
  Media Type = backup.mb.filestorage
  Maximum Concurrent Jobs = 10
}

Storage {
  Name       = backup.mb.copy.filestorage
  Address    = 10.11.0.72
  SDPort     = 9103
  Password   = "password"
  Device     = backup.mb.copy.filestorage-1
  Device     = backup.mb.copy.filestorage-2
  Media Type = backup.mb.filestorage
  Maximum Concurrent Jobs = 2
}

Storage {
  Name       = backup.fra.filestorage
  Address    = 10.11.0.72
  SDPort     = 9103
  Password   = "password"
  Device     = backup.fra.filestorage-1
  Device     = backup.fra.filestorage-2
  Media Type = backup.fra.filestorage
  Maximum Concurrent Jobs = 2
}

Storage {
  Name       = backup.fra.copy.filestorage
  Address    = 10.11.0.72
  SDPort     = 9103
  Password   = "password"
  Device     = backup.fra.copy.filestorage-1
  Device     = backup.fra.copy.filestorage-2
  Media Type = backup.fra.filestorage
  Maximum Concurrent Jobs = 2
}

Storage {
  Name       = backup.hkg.filestorage
  Address    = 10.11.0.72
  SDPort     = 9103
  Password   = "password"
  Device     = backup.hkg.filestorage-1
  Device     = backup.hkg.filestorage-2
  Media Type = backup.hkg.filestorage
  Maximum Concurrent Jobs = 2
}

Storage {
  Name       = backup.hkg.copy.filestorage
  Address    = 10.11.0.72
  SDPort     = 9103
  Password   = "password"
  Device     = backup.hkg.copy.filestorage-1
  Device     = backup.hkg.copy.filestorage-2
  Media Type = backup.hkg.filestorage
  Maximum Concurrent Jobs = 2
}
Storage daemon configuration

```
Storage {
    Name = backup.mb-sd
    Maximum Concurrent Jobs = 20
    WorkingDirectory = /var/backup/bareos/storage-working/
}
Director {
    Name = backup.mb-dir
    Password = "password"
}
Autochanger {
    Name = library
    Device = TL-DRIVE-1, TL-DRIVE-2
    Changer Command = "/var/backup/bareos/scripts/mtx-changer %c %o %S %a %d"
    Changer Device = /dev/changer
}
Device {
    Name = TL-DRIVE-1
    Drive Index = 0
    Media Type = LTO5
    Archive Device = /dev/nst0
    AutomaticMount = yes
    AlwaysOpen = yes
    AutoChanger = yes
    LabelMedia = no
    Maximum Filesize = 20G
    Spool Directory = /var/backup/spool
    Maximum Spool Size = 100G
    Maximum Concurrent Jobs = 1
}
Device {
    Name = TL-DRIVE-2
    Drive Index = 1
    Media Type = LTO5
    Archive Device = /dev/nst1
    [...]
All devices configured the same way!

→ 10 write devices
Offsite datacenters get their own devices

2 write devices each - wan bandwidth limitation
Devices for copy jobs

So copies do not interfere with backups
Automatic client configuration

- Python script running as a system service
- Listening to network port, TLS encryption
- Takes a JSON-String and writes client configuration
  - Client name
  - Password
  - Client type
  - Other options
- Deactivates client on previous backup server (same mechanics)
- Small, simple, fast
Deployment

- Rollout in FRA DC first
  - Less special cases
    - Auto-Update old configuration in MB for remote backup (Password change)

- Rollout in MB DC
  - Automatic config deployment
  - Systems register automatically
  - A few clients with special configurations (RunScript, special Fileset)

- Rollout HKG DC
  - Least problematic
  - configuration problems have been worked out already
Problems

- 100+ full backups over 2MBit WAN take a while.

- Multiple “exclude dir containing” definitions did not work. Bug submitted, fixed by Marco.

- Tape Library started to lock up on copy jobs:
  - Seems to be a block size problem.
  - Removing block size configuration solved the problem (using standard block size now)

- Starting copy jobs with a running original backup caused ONE copy job to be canceled (cancel duplicates in original jobdef). Bug reported, fixed by Marco.

- Tape-2-Tape archive jobs always rewound the read tape between jobs even though the jobs were in the correct order. Bug reported, fix submitted, process sped up a lot.
Accessibility of old backups

How do we access the backup history from the old server?

- Made sure all jobs have been copied from disk to tape
- Cleared all disk volumes - because only tapes will be accessible
- Imported the old database on the new server (separate MySQL Schema)
- Copied config to own directory
- Configured director to a different port and made correct devices available for restore (separate LTO5 Drive)
- Wrote wrapper script to handle services (oldbackuprestore)
  - Starts director with the old config
  - Starts console that connects to said director
  - Handle exit cases, shutdown director

Running the script gives you access to entire old servers data and allows restore from tapes.
Results

- Smooth transition with only minor bumps
- Backup performance greatly improved
- Simplified configuration
- Old backups still accessible
Questions?
Overtime!
Retiring clients – 1

How do you retire clients?

Quick & Dirty

- delete volumes
- delete configuration file

Pros & Cons

- Pro: quick
- Con: no further restore possible
- Con: artefacts in the database - pool, client, fileset
- Con: if backups are on tape - using bscan to recreate volume information does not work ("unknown client").
Retiring clients – 2

Retire them and keep available.

- make sure all jobs have been copied to tape
- delete on-disk volumes
- disable client jobs
- move config to directory for retired clients

Pros & Cons

- Pro: restore still possible
- Pro: bscan catalog recreation still works
- Con: database storage requirements
- Con: configuration can get very big over the years
Cycle database

Database gets very big. How can we deal with this?

- Retire them and keep available.
- make sure all jobs have been copied to tape
- delete on-disk volumes
- copy config to retirement directory (bareos-2014)
- rename database & adjust catalog config
- Extend oldbackuprestore to also offer restore from past year

Pros & Cons

- Pro: Start with clean database each year
- Pro: retired clients can be removed during switch
- Con: database storage requirements increase (but can be moved to slower storage)
- Con: restoring cumbersome
Thank you!