Bacula Success Stories: Einsatz in vielfältigen Umgebungen

Christopher Beppler

25. September 2013

Introduction

Quote

Yes, we have a dress code. You have to dress. — Scott McNealy

Table of Contents

1. Introduction
2. Motivation
3. Success Stories
   - Secure off-site Backup via FTP
   - Free Backups in the cloud
4. Q & A

Motivation

Quote

The computer was born to solve problems that did not exist before. — Bill Gates

Motivation

- Limitations
  - budget
  - space
  - time
- Hacking
  - unique ideas and implementations
  - the joy of going ”off protocol”

Success Stories

Quote

Tell me and I forget. Teach me and I remember. Involve me and I learn. — Benjamin Franklin

Secure off-site Backup via FTP

Environment

- 1 Server
  - Linux (CentOS 6)
  - File Sharing (Samba)
  - customer-specific software using a MySQL database
- 4 Workstations
  - Windows (XP, Vista, 7)
  - no client backup, but that was never *really* critical
- 1 FRITZ!Box off-site
  - External Hard Drive connected via USB
  - Disk Space provided via FTP

old solution

- Backup using FTP
  - unencrypted files transferred using `ncftp`
  - recognition of modified files using `find`
- Limitations
  - no encryption
  - no revision
  - deleted files stayed in backup
  - renamed files stored multiple times in backup
  - some "teething problems" like missed files or filename encoding mismatch

new solution

- Backup of the server using Bacula
  - Backups stored on a dedicated RAID on the same machine
  - this enabled file revision
  - off-site Backup is still mandatory
- Backup of the workstations using Bacula

Implementation

- Bacula-Configuration
  - Limit Volume size to 1024 MB
  - Limit Pool size to the size of the off-site hard disk
  - Enable encrypted storage
  - One Job which doesn't backup anything at all (`/dev/null`) but triggers a script to copy the volumes via FTP
  - This Job runs after the Catalog backup using a higher priority value to ensure it's always the last job of the day
- Shell script only transfers the volumes which have changed since last successful upload.

Conclusion

- Bandwidth limitation requires manual copying of the volumes to the hard disk within the office location after a full backup twice a year.
- Weekend is used for a Differential backup as more time for off-site copying is available.
- Differential Off-site backup time almost always finished before normal office hours began.
- An attacker could steal the files but as these are encrypted they're not worth a lot.

Free Backups in the cloud

Motivation

- Playing with Amazon S3
- Finding a useful task for my Raspberry Pi
- Raspberry Pi only has limited disk space
- external hard drive would be too easy
- Backups to tape are noisy and require discipline

Concerns

- No control over usage of the data by the cloud provider
- Storage outside of Germany

Off-site storage providers

- WebDAV
  - `www.4shared.com`
  - `www.mydrive.ch`
- S3-like
  - Amazon S3
  - Google Storage
  - OpenStack/Swift
  - Rackspace CloudFiles

FUSE filesystems

- `davfs` for WebDAV
- `encfs` for file-encryption
  - encryption directly on filesystem level
- `s3ql` for S3-like
  - encryption
  - compression
  - deduplication

Implementation (WebDAV)

- `sudo mount -t davfs https://webdav.4shared.com/ /mnt/4shared`
- transport security with `https` ist not sufficient. Files would still be stored unencrypted.
- `encfs /mnt/4shared /media/4shared`
- Not even filenames are stored in plaintext using `encfs`.

Implementation (WebDAV)

Storage-Daemon:

```
Device {
  Name = 4SharedStorage
  Media Type = File
  Archive Device = /media/4shared/bacula
  LabelMedia = yes;
  Random Access = yes;
  AutomaticMount = yes;
  RemovableMedia = no;
  AlwaysOpen = no;
}
```

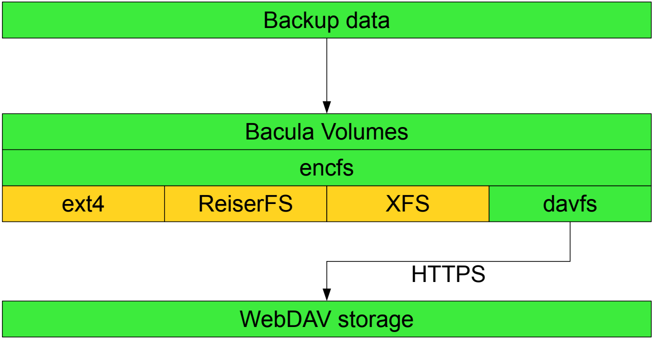
Implementation (WebDAV)

Director:

```
Storage {
  Name = 4SharedStorage
  Device = 4SharedStorage
  [...]
}

Pool {
  Name = 4Shared
  Maximum Volume Bytes = 1024M
  Maximum Volumes = 15
  [...]
}
```

Implementation (WebDAV)



Backup data

Bacula Volumes

encfs

ext4	ReiserFS	XFS	davfs

HTTPS

WebDAV storage

Implementation (Amazon S3)

1. Create Amazon Webservices (AWS) Account.
2. Create AWS keypair.
3. Create Amazon S3 Bucket.
4. Create `s3ql`-filesystem (`mkfs.s3ql`).
5. Mount `s3ql`-filesystem (`mount.s3ql`).

Implementation (Amazon S3)

Storage-Daemon:

```
Device {
  Name = AmazonS3Storage
  Media Type = File
  Archive Device = /media/s3/bacula
  LabelMedia = yes;
  Random Access = yes;
  AutomaticMount = yes;
  RemovableMedia = no;
  AlwaysOpen = no;
}
```

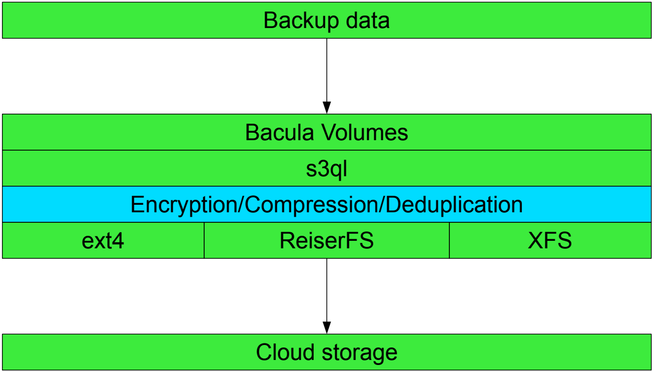
Implementation (Amazon S3)

Director:

```
Storage {
  Name = AmazonS3Storage
  Device = AmazonS3Storage
  [...]
}

Pool {
  Name = AmazonS3
  Maximum Volume Bytes = 100M
  Maximum Volumes = 100
  [...]
}
```

Implementation (Amazon S3)



Backup data

Bacula Volumes

s3ql

Encryption/Compression/Deduplication

ext4 | ReiserFS | XFS

Cloud storage

Conclusion

- For smaller backups providers like 4Shared are a good choice.
- Deduplication, Compression and Encryption with Amazon S3 storage is a lot of load for a Raspberry Pi.
- More appropriate as an addition to an existing backup.

Q & A

Quote

I refuse to answer that question on the grounds that I don't know the answer. — Zaphod Beeblebrox

Thank you for listening.