Distributed Backup And Disaster Recovery for AFS

A work in progress

Steve Simmons Dan Hyde

scs@umich.edu drh@umich.edu

University of Michigan

Background

- Assigned to work on backup and disaster recovery end of 2005
- Disk-based backup desired to replace TSM
- Funding and political support were strong
- Policy was vague

Some Useless Stats

- 13 dataful servers, plus others
- Two vice partitions per server
- 9.8TB as of today, about 33% full
- Critical university services rely on AFS
- 245,686 volumes (not counting backups, clones, etc)
- Volumes up to 110GB actual usage - and users want much bigger

Process Confusion

- Problem: What is backup, and why do it?
- Solution: there are three separate needs
  - Restore of lost/deleted files - “file restore”
  - Recovery from hardware/software disaster
  - Long-term storage of critical data sets - “archive”

Optimal solution for one is suboptimal for others

File Restore

- Save users from their errors
- Save users from their errors
- Recover data when old user returns
- FOIA/HIPPA complicate the picture
- Predictability of backup and restore process
- A very slow process (days) is usually acceptable.

Disaster Recovery

- When a service is going to be out for a long time, it needs to be restored and quickly
- Involves files **PLUS** hardware, power, cooling, network, dogs, cats, professors…
- Always unplanned
- Only the most recent version is of interest
- As close to real-time data as possible

Archival

- Officially, not our issue
- Unofficially, we expect it to come up
- Timing of archive selected manually by user
- Retention time arbitrarily long (but FOIA, HIPPA, etc complicate things)

Backup To Disk For File Recovery

- Basic design done in Jan 06
- afsdump.pl posted by Matthew Hoskins
- Heavily modified for our use
- Yes, we will post
- Some auxiliary tools as well

File Restore Basic Method

- For every afs server, a bfs server
- Incremental dumps arbitrarily deep
- n days to do a cycle
- Only 1/n full dumps per day
- Per-volume dump schedules that override default

Getting Started

- Initial backup is level 0
- About 20x faster than ancient TSM (1.2 TB in 5 hour vs. 3+ days)
- After first backup, rewrite the db to spread future level 0s across the volumes (reseq_dump)
- Daily/full backup mix, all servers, under 1 hour

Actual Status

- Running in pilot today
- Expected to be in production in 30 days
- Lots more interesting tools to build, but core is done:
  - We can back up
  - We can recover
- Declare victory, start on disaster recovery

Quick Questions

vos shadows (history)

- vos move
  - same volumeID
  - deletes source volume and .backup
- vos copy
  - new volumeID; must not exist
  - doesn’t delete source volume nor .backup
- vos shadow
  - new or old (default) volumeID
  - doesn’t delete source volume nor .backup
  - not in VLDB
  - -incremental

vos shadows (changes)

- volIsShadow (controlled recovery)
  - new bit in volume header
- vos shadow
  - sets volIsShadow
- vos syncvldb
  - error if volIsShadow
  - override with vos syncvldb -forceshadows

vos shadows (future)

- Volume groups/families
  - rw, ro, backup, clone, parentID
  - where do shadows fit in?
- restoredFromID
  - “….to make sure that an incremental dump is not restored on top of something inappropriate”
  - dangers
    - vos shadow a
    - vos shadow b -incremental
- Database vs. listvldb/listvol/paper notes

Quick Questions

Disaster Recovery & Shadows

- One to one relationship of afs server to recovery server
- Spread afs and bfs servers across sites?
- Same host used for disaster recovery and disk-based backup (“file restore”)
- RAID configured for space, not speed
- Shadows don’t appear in volume db
- Incremental refresh of shadows is fast

When Disaster Hits

- Identify which server is down
- Promote all volumes on shadow server to production
- You’re back in service, with loss of only data since last refresh of shadow
- Incrementally update shadows as frequently as load allows - maybe 3-4 times a day

Gotchas I

- Must be careful to delete/create shadows when doing vos move
- vos moves don’t invalidate old backups, but shadows have problem (see later)
- slocate/updatedb is your friend
- Periodic reruns of reseq_dumps needed

Gotchas II

- Need a real db (will have soon)
- When 13 afs servers shadow/dump to 13 bfs servers simultaneously, bandwidth suffers
- Distributing level 0s speeds disk dumps, but makes tracking (and restore?) complex
- Per-volume dump schedules not implemented, but only a few days work

Into The Future

- We want clones of shadows
- Do the backups from the shadows, not the live volume
- Don’t use .backup, use clones - so 24-hour snapshots can be available even if a full backup takes longer than 24 hours

Use Clones Instead of Backups

- Clones should appear in db
- Let users access the multiple clones like .backup
- Have multiple .backups (clones) available

Allow many clones

- 30 at a minimum, hundreds is better
- Avoid performance issue by making many clones from the shadow, not the production
- When you move a shadow (it will happen), preserve the clones

Naming issues for many clones

- Must be comprehensible to users
- May have to be dynamic
- How to make available without training 400,000 people to do fs mkm