Moving To AFS

Simon Wilkinson
school of Informatics
University of Edinburgh
simon@sxw.org.uk

Overview

- Evaluation results
- Cell design
- Deployment process
- Deployment experiences

Background on Informatics

- ~ 2000 active users, ~1500 hosts
- 20 Tb of centrally managed filestore
- Deployed Kerberos and LDAP infrastructure

Our Existing Filesystem

- NFS v3 based with Sun fileservers and predominantly Linux clients
- AMD automounter providing identical filesystem on every machine
- Locally developed mechanisms to populate AMD filesystem maps, manage quotas, and do nightly mirroring
- Developed incrementally over many years.

Weaknesses

- Lack of security
  - Can't allow access from unmanaged machines
  - Can't allow access from beyond the firewall

Weaknesses

- Lack of portability
  - AMD infrastructure required significant modifications to off-the-shelf machines
  - Lack of client availability for some systems

Weaknesses

- Lack of maintainability
  - Local glue required lots of effort just to keep running
  - Dealing with partition filling, and the resultant home directory moves
  - Fileserver failure leads to hung mounts, and lots of rebooting

Criteria

- Secure enough to permit access from foreign machines, and across firewalls
- Flexible ACL model
- Better performance
- Stability
- Linux and Solaris support required, Windows and Mac OS X desirable
- Easily scale to our client & data requirements
- No per-client licensing fees
- Preferably be a self-contained solution

Candidates

- AFS
- CIFS
- Coda
- NFSv4

Feature Comparison

- On paper, most AFS features are present in NFSv4
- Critical absence is volume location independence
- Can't move filespace between servers without the user noticing
- No concept of a global namespace - still needs automounter glue!

Evaluation

- AFS and NFSv4 feature sets very similar on paper, with NFSv4 leading the way
- However, NFSv4 “not quite ready yet” - few implementations of complete feature set
- Linux NFSv4 only did machine based authentication at mount time
- Bugs in NFSv4 implementation caused benchmarks to hang

Benchmarks

- Three benchmarks selected
  - iozone
  - blogbench
  - The Andrew Benchmark
- Only iozone and blogbench eventually used

Benchmarking Results

- NFSv4 won the iozone one every time - by a small margin for files smaller than the AFS cache size
- Much more evenly matched with blogbench
- “Lies, damn lies, and statistics”

Evaluation Results

- NFSv4 just wasn't ready, and would still have required automounter madness.
- "Don't want our data to be their learning experience"
- OpenAFS met the majority of our criteria, with stability as an added bonus!

Cell design Authentication

- AFS is tightly coupled with our authentication infrastructure
- Using RedHat's RH9 vintage pam_krb5 module (but planning on stopping)
- Using Doug Engert's pam_afs2 module (but looking at Russ's pam_openafs_session)

Cell Design Directory

- Debated integrating pts with our existing LDAP directory
- Wrote some proof-of-concept code to backend pts with LDAP
- Decided that our LDAP service wasn't sufficiently reliable to do this in production
- Use 'standard' pts, with hooks into our account management system

Cell Design - Backups

Cell Design
Online Backups

- Our recent history makes us somewhat jumpy
- Off site disk mirrors was a requirement
- So, we use read-only user volumes
- All user volumes have an offsite RO copy which is released nightly.
- Backup volumes are still used to provide 'Yesterday' functionality, and tape backups ...

Cell Design
Tape Backups

- Finding a workable, scalable, tape backup system is a priority
- Currently embroiled in local politics
- At the moment, we just walk the AFS filespace and use our existing EBU licenses
- Not a very pretty bodge!

Deployment Experiences

- Softly, softly ...
- Initially offered additional file space, rather than home directories, to the adventurous
- Gradually shifted computing staff home directories over
- Now creating all new users in AFS
- Starting to bulk move existing users

Things that make our users Sad

- ACLs - especially the fact they are directory only
- Lack support for ‘special’ files such as devices or named pipes.
- Limits on maximum number of files per directory
- Linux’s behaviour with sticky mode temporary directories

Things that caused us pain

- .Xauthority files stored in home directories
- SSH public key files
- System daemons inheriting the PAG of the user starting them.
- Condor
- Beagle

Security Hurts!

- Requirement to gain credentials before accessing files causes problems
  - Cron
  - Web servers
  - Condor and Grid Engine

Security Still Hurts

- Having to renew credentials is not popular
  - Long running jobs
  - Processes left running overnight (Thunderbird, gnome-screensaver!)
- Unix applications aren't good at dealing with unexpected FS failure

Reduce The Pain

- Get your filesystem credentials at login
- Renew them whenever you can (screensavers &c.)
- Don't have credentials expiring in the middle of the day
- Make sure all credentials renewal tools renew AFS tokens, too

Long Running Jobs

- Provide a mechanism for stashing credentials with a subset of permissions on the local disk
- Encourage people to use this to provide credentials for long running jobs
- k5start and krenew are hugely useful tools
- Renewable tickets are great for medium-life jobs!

Conclusions

- Going well so far
- The crunch point is just around the corner!
- Softly, softly has perhaps been too soft
- Ensuring reliability before moving users, and responding rapidly to their concerns has been key

Thanks!

- There's a lot of good code and support out there!

QUESTIONS?