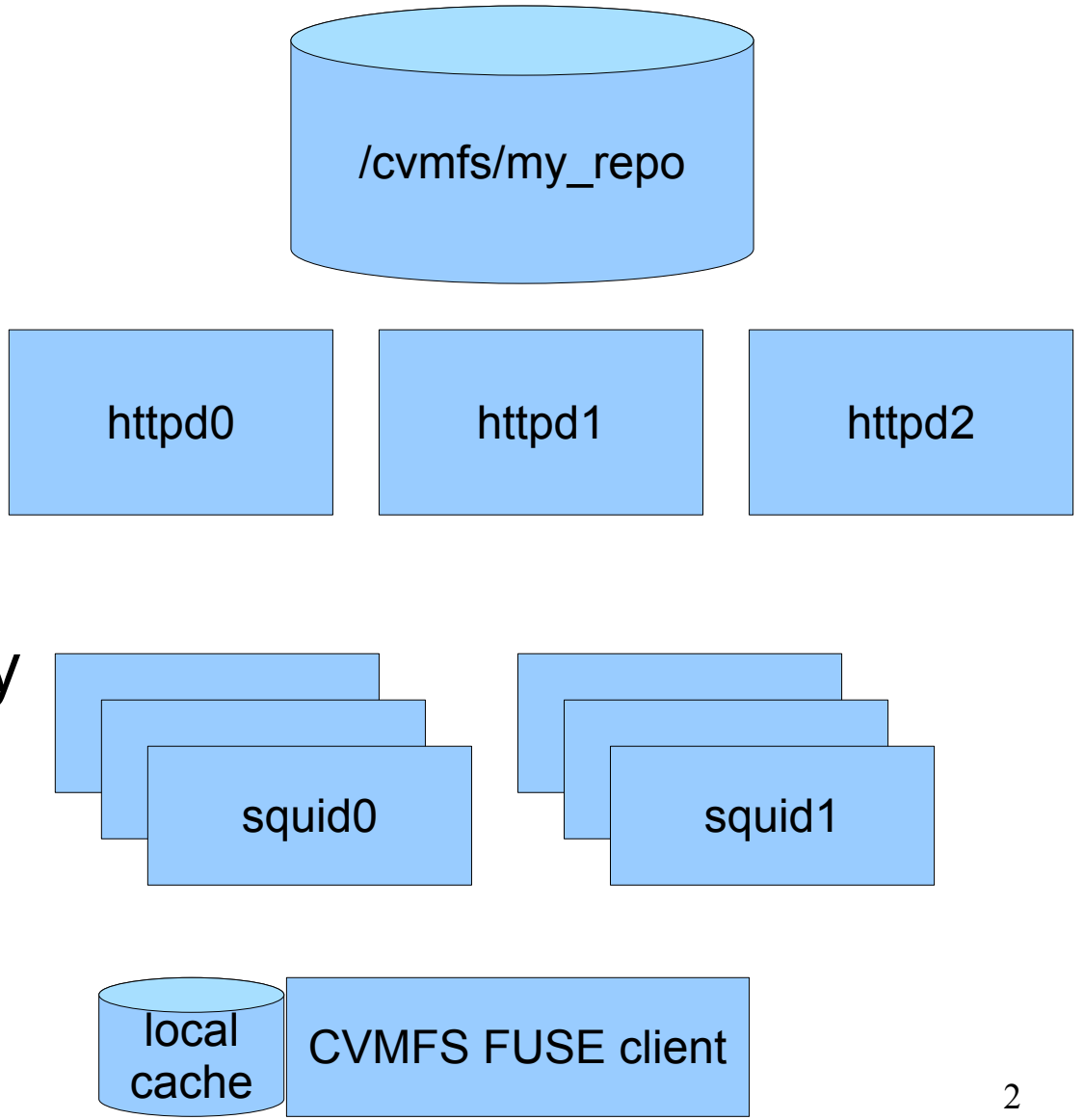
Integration of Parrot Virtual Filesystem and CernVM-FS

Dan Bradley <dan@hep.wisc.edu>
Any Data, Any Time, Anywhere Project

What is CernVM-FS?

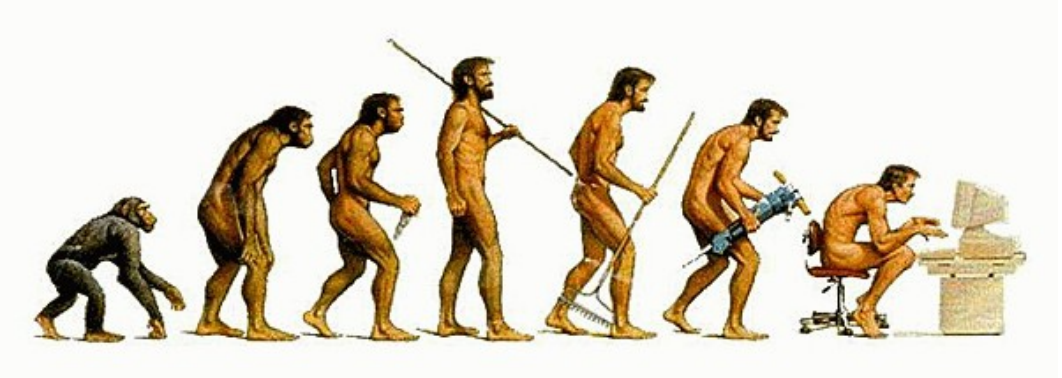
- network filesystem
- read-only POSIX interface
  - FUSE-mounted
- fetches files via http
  - verifies data integrity
- aggressive caching
  - local disk
  - web proxies



CernVM-FS aka CVMFS

- Problem #1
  - It's a mouth-full

CaVeMaN-FS?



Some Early History

- CVMFS originally relied on parrot
  - Used parrot's GROW-FS
- Switched to a new implementation c. 2010
  - Extends basic idea of GROW-FS
  - Designed as FUSE module
- Full circle
  - Now we want CVMFS support in parrot
  - i.e. alternative to GROW-FS in parrot

GROW-FS vs. CVMFS

- Both
  - Read-only http access
  - Server can be standard web server
  - Local on-disk cache on client-side
  - Can benefit from intermediate web caches
  - Client downloads a file catalog
    - Catalog created in explicit publication step
    - Contains strong cryptographic checksum of all files
    - Integrity of catalog is key to verifying all else

GROW-FS vs. CVMFS

GROW-FS

- Catalog in client memory

CVMFS

- Catalog in SQLite
- Supports sub-catalogs
  - Only downloaded if sub-tree accessed

GROW-FS vs. CVMFS

GROW-FS

- Catalog integrity via checksum fetched by https
  - implemented?

CVMFS

- Catalog integrity via signature fetched by http
  - Client must have copy of public key

GROW-FS vs. CVMFS

GROW-FS

- File tree in web directory

CVMFS

- File tree copied into web directory
  - Compressed
  - Named by checksum
- More heavy-weight publication process
  - Even has its own kernel module!

GROW-FS vs. CVMFS

GROW-FS

- Leaves resilience and load-balancing up to the web layer

CVMFS

- Explicit support in client for redundant sources for repository
  - Failover
  - Preference for low-latency source
- Ditto for intermediate http caches
  - Also supports load-balancing

GROW-FS vs. CVMFS

GROW-FS

- POSIX access via parrot
  - Don't need any favors from admin

CVMFS

- POSIX access via FUSE
  - Need admin to allow us to mount /cvmfs/cms.cern.ch

Adding CVMFS to Parrot

- We want both FUSE and parrot modes
  - FUSE on resources we control
  - Parrot on others
- Main integration issue
  - No existing CVMFS client API
  - Code designed for use in FUSE module

Adding CVMFS to Parrot

- Main integration issue
  - No existing CVMFS client API
  - Code designed for use in FUSE module
- Two approaches considered
  - Make parrot support FUSE modules
  - Create CVMFS client API library and use it in parrot

Loading FUSE modules in Parrot?

- Pros
  - No mods to CVMFS required
  - Could support other FUSE filesystems too
- Cons
  - Might have more generic user interface
    - Harder to use
  - Performance concerns:
    - FUSE module running as ptraced process under Parrot?
    - o.w. must modify FUSE library so ptrace not necessary
      - Setup of pipe between “kernel” (parrot) and module
      - Still a little worried about performance, but less

CVMFS client API library

- Implemented in Jan 2012
  - https://github.com/dcbradley/parrot_cvmfs
- Patch accepted and released in CVMFS 2.0
- Major code changes in CVMFS 2.1
  - 2.0 patch can't be used
  - Currently working on libcvmfs patch for 2.1

CVMFS in Parrot

- Cvmfs patch merged into cctools in late Jan, 2012
- Not yet enabled in binaries released on cctools web page
- To compile it yourself, see
  - https://github.com/dcbradley/parrot_cvmfs

Usage

- parrot_run ls /cvmfs/my_repo

- cern.ch repositories supported out-of-the-box (e.g. cms.cern.ch, atlas.cern.ch, and others)
  - Public key bundled
  - URLs of web server + globally distributed replicas
- A web proxy *must* be configured

Configuring Web Proxy

- Two options:
  - Standard parrot web proxy
    - parrot_run -p or HTTP_PROXY
  - CVMFS-specific web proxy
    - parrot_run -r or PARROT_CVMFS_REPO
    - Supports a failover chain of load-balanced proxies

Usage

- Other repositories can also be configured
  - -r or PARROT_CVMFS_REPO
- Example:
  export PARROT_CVMFS_REPO = "
  cms.hep.wisc.edu:
  pubkey=/home/dan/cms.hep.wisc.edu.pub,
  url=http://cvmfs01.hep.wisc.edu/cvmfs/cms.hep.wisc.edu"
- parrot_run ls /cvmfs/cms.hep.wisc.edu

Limitations

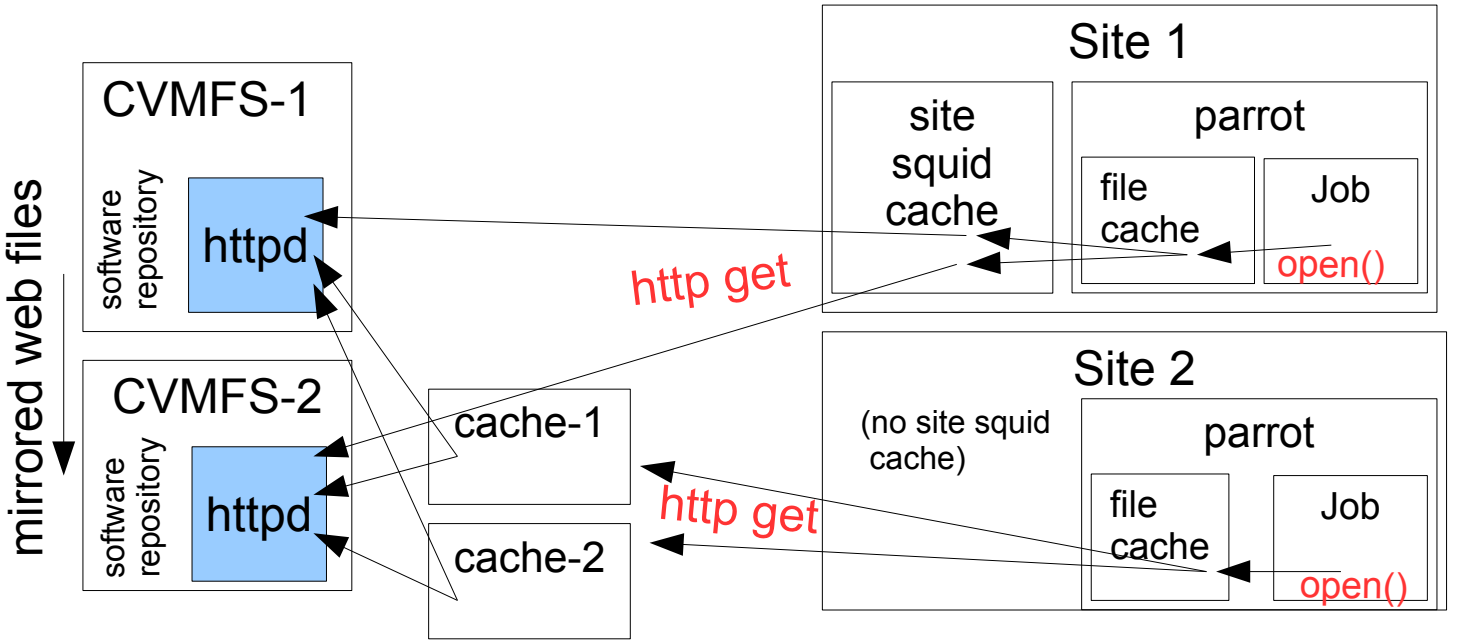
- CVMFS client library currently supports only one repository at a time
  - Globals in libcvmfs from FUSE-oriented code
- Parrot can destruct and re-initialize libcvmfs when switching between repositories
  - Not enabled by default
  - Likely severe performance penalty for frequent switching

Case Study: CMS use of Parrot + CVMFS

- CMS physicists submit analysis jobs to the Open Science Grid
- CMS manages application software at CMS-owned sites
  - Starting to migrate to CVMFS for this
- In theory, CMS could maintain software at non-CMS sites, but in practice this doesn't happen
- By using Parrot + CVMFS, we can make use of non-CMS sites to run jobs that rely on CMS software

  - No special support by site admins is required

System Architecture

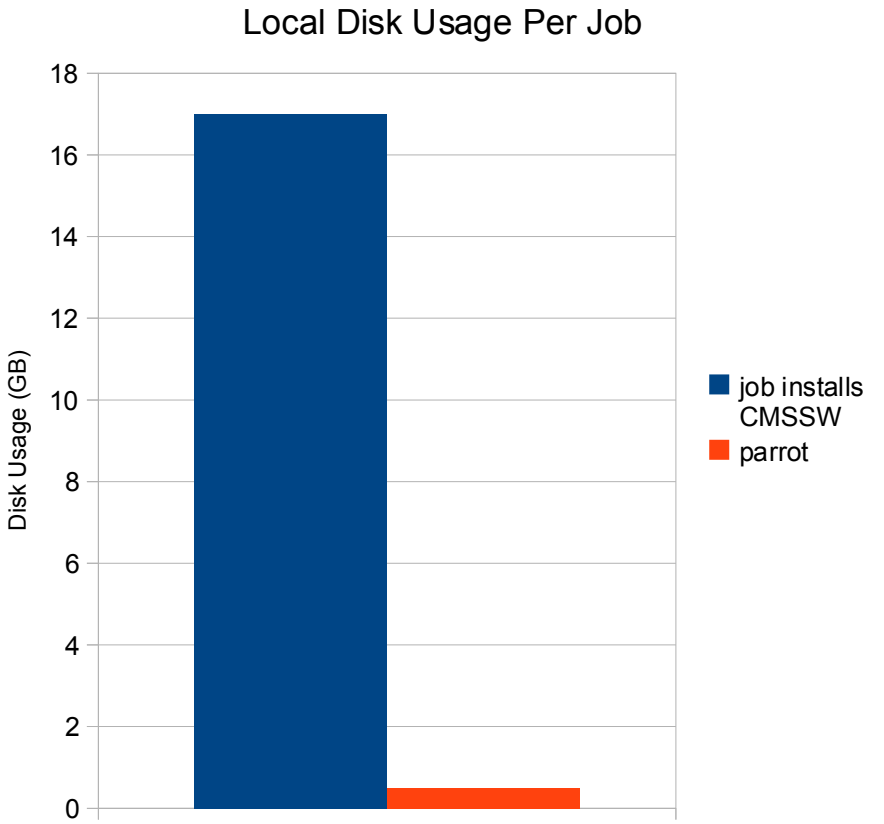


Repository cms.hep.wisc.edu:
4M files, 100GB, 100 sub-catalogs

Also serves as OSG_APP for Wisconsin OSG site
More info: http://www.hep.wisc.edu/cms/comp/ops/cvmfs.html

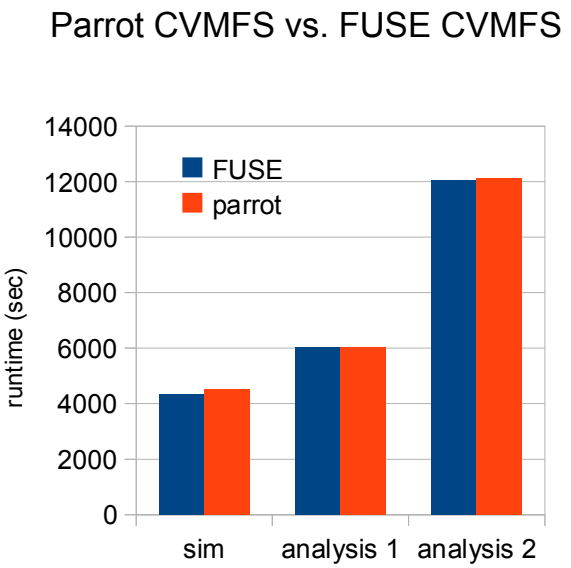
CMS Disk Usage

- Why not just download and install CMS software?
  - Local per-job installation requires unreasonable time and space
- Parrot + cvmfs only needs to download a fraction of the data
  - most files not actually used by most jobs



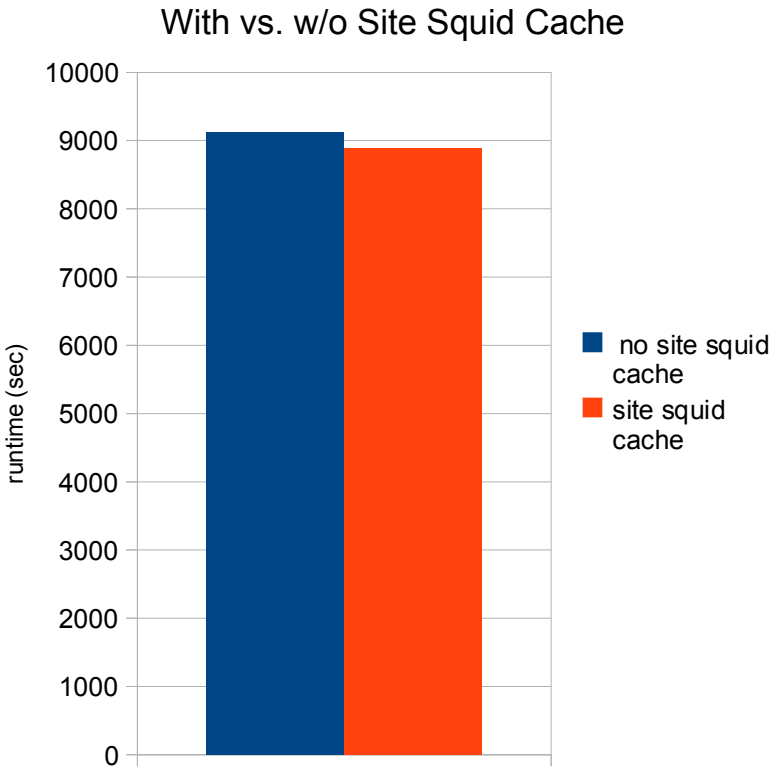
Performance

- For typical CMS analysis jobs, Parrot CVMFS results in comparable performance to FUSE CVMFS
  - For runtime ~1h+



Site-level caching

- Not all OSG sites have web caches
- Does it matter?
  - Tested between UCSD and Wisconsin
  - Not bad
  - But there must be enough bandwidth to central web proxies



Problems Encountered

- CMSSW setup (scram) goes into infinite loop under parrot
  - RH 5.1
  - pwd fails, causing infinite loop
  - Problem: parrot needs to know at time of open() if path is a file or directory
    - Relies on O_DIRECTORY flag
    - Not strictly POSIX behavior
  - Workaround applied
    - Parrot treats “.” and “..” specially in open()

Problems Encountered

- Use of mmap by CMSSW for file buffering
  - Problem: parrot did not flush data written to mmap buffer before munmap
  - Fixed!

Problems Encountered

- sshd fails under parrot
  - affects condor_ssh_to_job in our glideinWMS environment
  - Problem: parrot does not handle some tty operation
  - Not yet fixed
    - Workaround would be to run sshd outside of parrot and wrap user's shell in parrot instead

Problems Encountered

- Some CMS jobs have much larger VSIZE under parrot
  - Gradual build-up, as though memory leak
  - Not yet understood
    - Could be an application problem
      - But only seen under parrot
    - Difficult to debug!

Real Usage

- Deployed parrot + cvmfs job wrapper in Wisconsin → OSG glideinWMS system
  - Opportunistic use of non-CMS OSG sites
  - 180,000 CPU-hours used by CMS physicists
  - 20,000 CPU-hours to failed jobs (VSIZE bloat)
- One of the visions of the Open Science Grid:
  - ATLAS and CMS share resources
  - We are finally doing it!
- Plan to deploy parrot + cvmfs wrapper in broader US CMS glideinWMS system

Conclusions

- Parrot virtual filesystem + CernVM-FS = software access from anywhere
  - Eliminates
    - Maintaining software at remote sites
    - Clunky tar/untar of software packages
    - Installing files that are never used
  - Results
    - Performs well for CMS analysis and simulation
    - But need to solve VSIZE bloat